Equivariant & Coordinate Independent Convolutional Neural Networks

Maurice Weiler

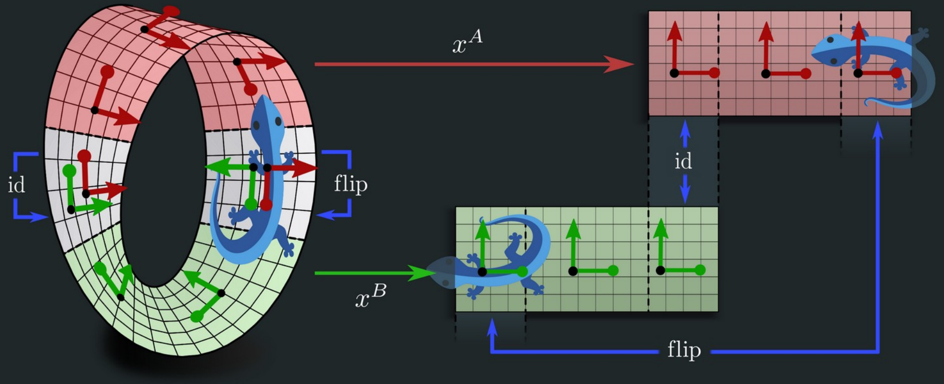
Jaakkola lab

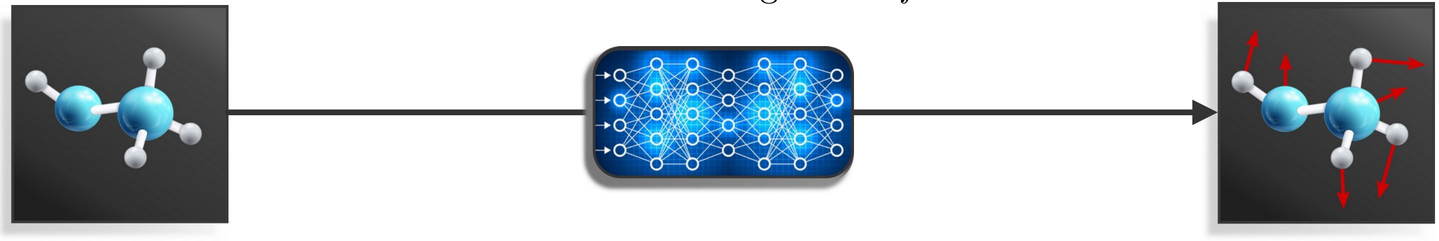
MIT CSAIL

@maurice_weiler

Massachusetts Institute of Technology

Universiteit van Amsterdam



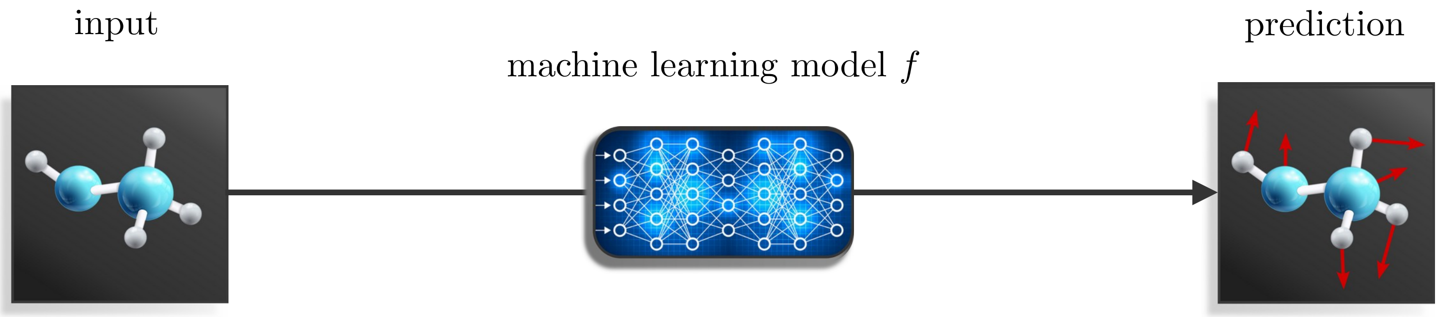


input

machine learning model f

prediction

non-equivariant models explicitly need to learn any transformed sample

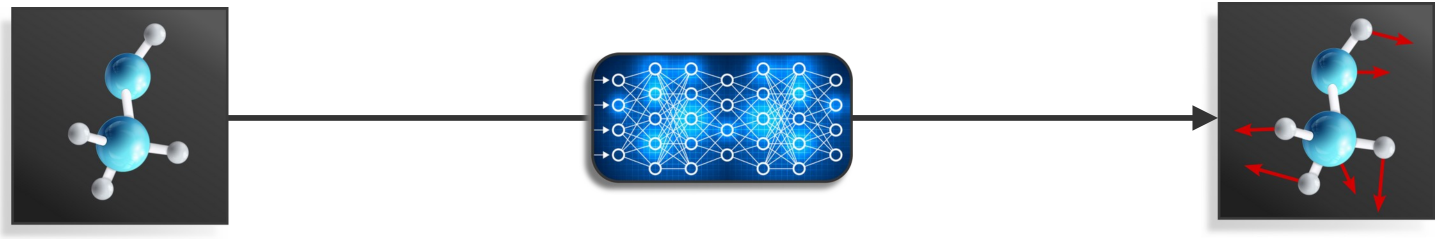


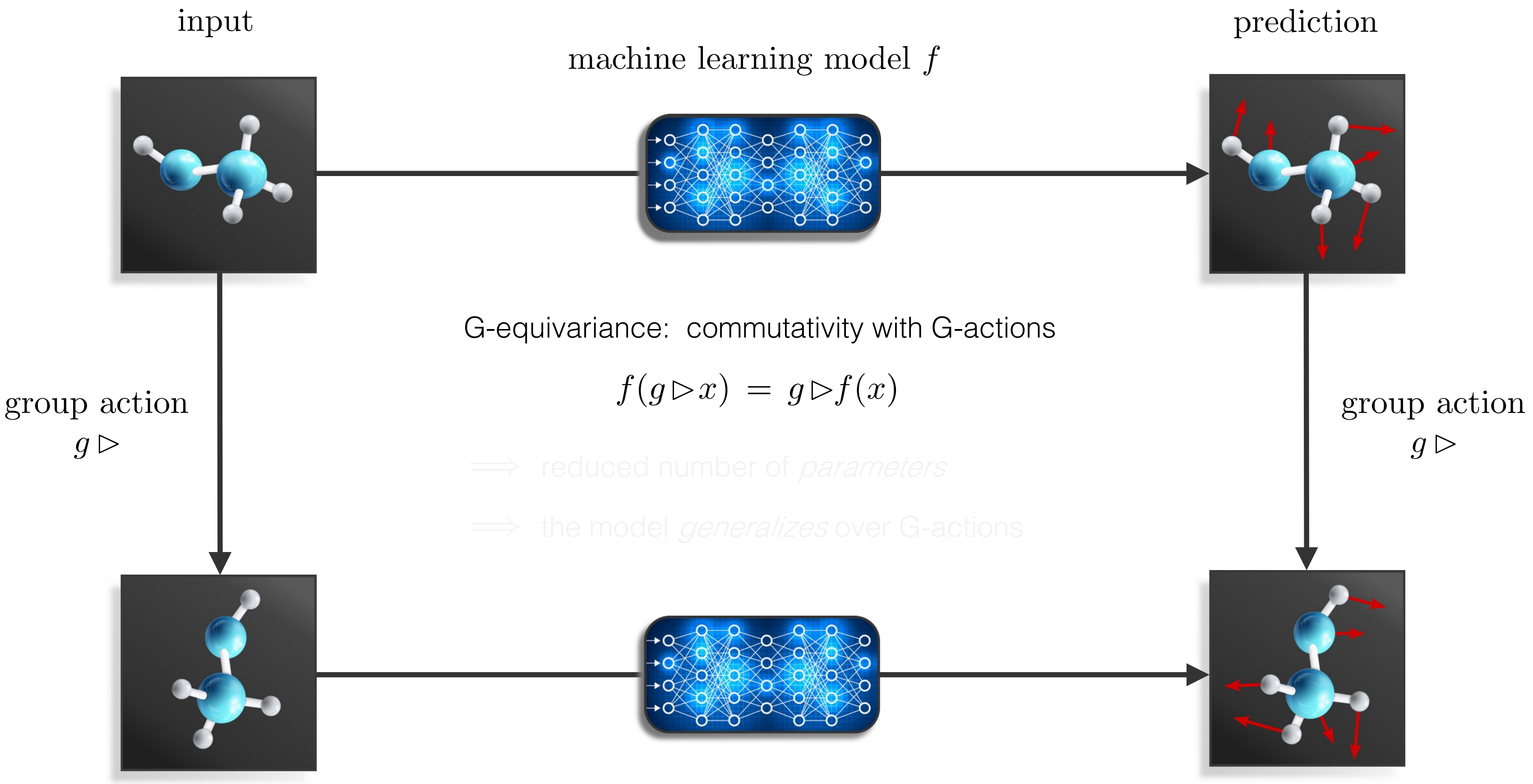
input

machine learning model f

prediction

non-equivariant models explicitly need to learn any transformed sample





input

machine learning model f

prediction

group action
$g \triangleright$

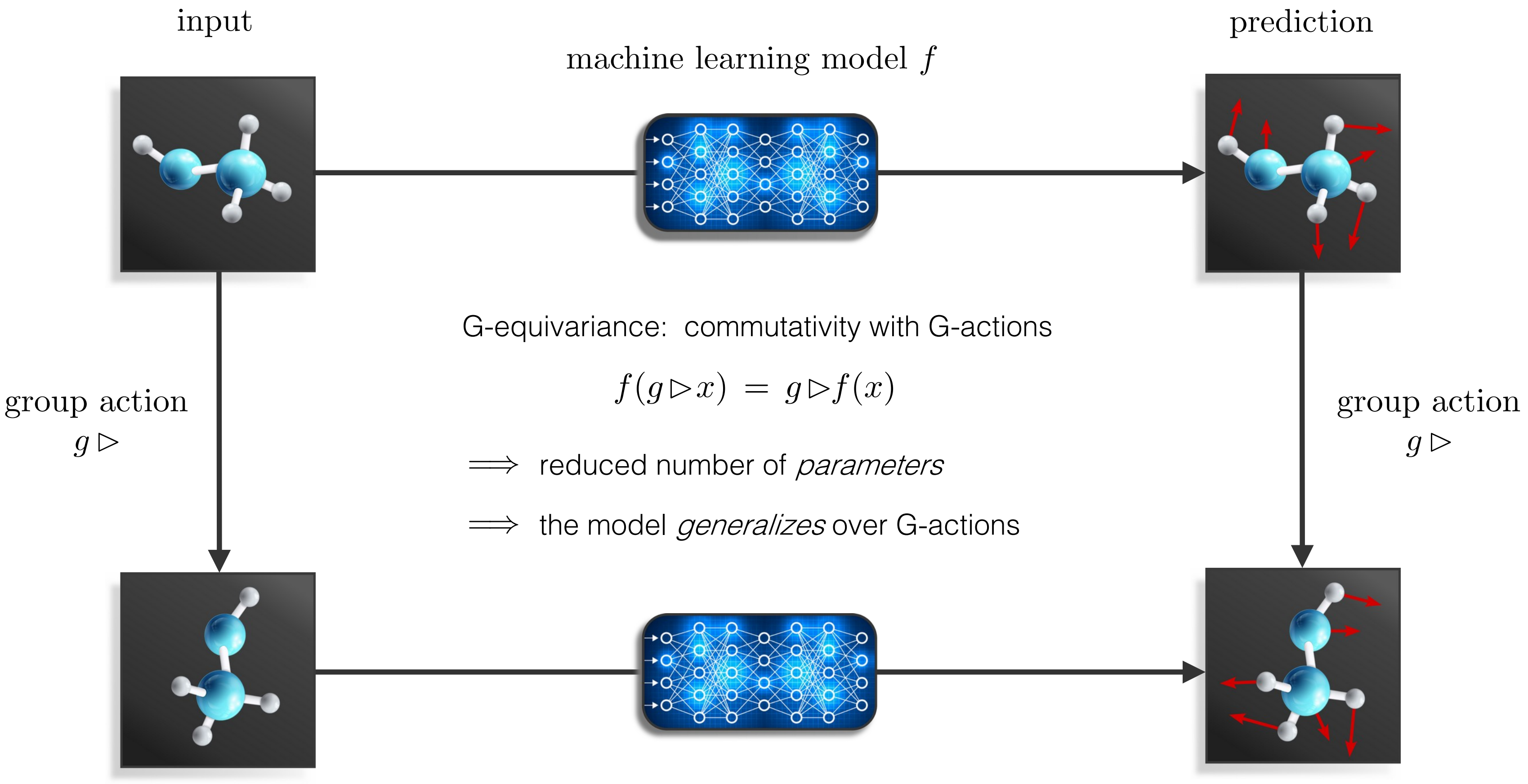
group action
$g \triangleright$

G-equivariance: commutativity with G-actions

$$f(g \triangleright x) \;=\; g \triangleright f(x)$$

$\Longrightarrow$ reduced number of *parameters*

$\Longrightarrow$ the model *generalizes* over G-actions



input

machine learning model f

prediction

group action
$g \triangleright$

group action
$g \triangleright$

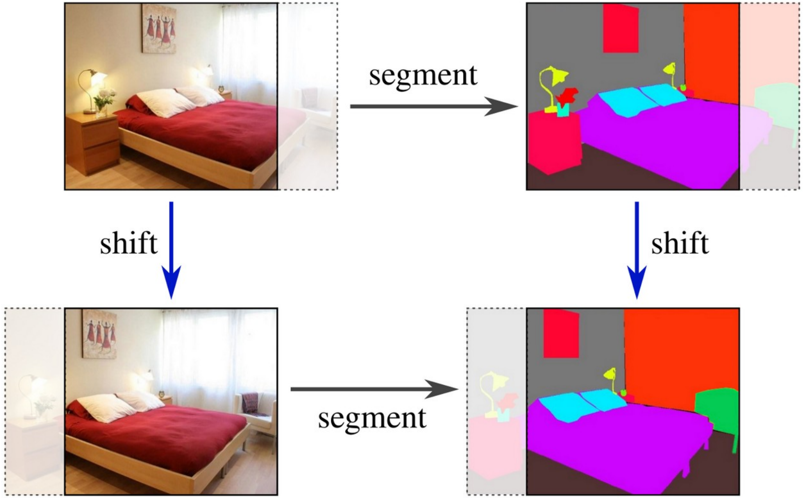
G-equivariance: commutativity with G-actions

$$f(g \triangleright x) \;=\; g \triangleright f(x)$$

$\Longrightarrow$ reduced number of *parameters*

$\Longrightarrow$ the model *generalizes* over G-actions

convolutional neural networks are *translation equivariant*

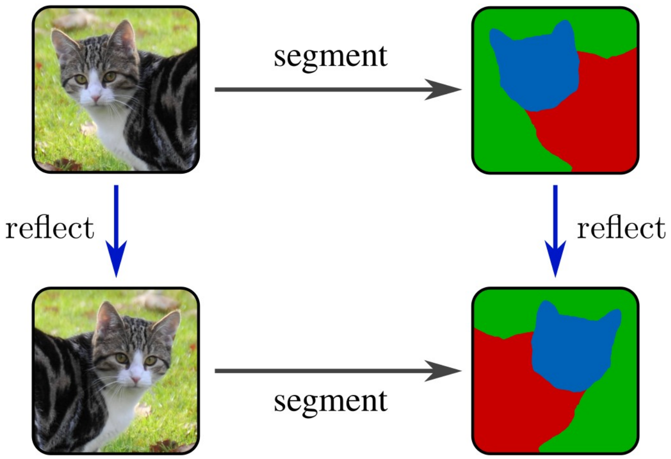


my research: generalize equivariant CNNs to … … larger symmetry groups

… more general manifolds

… gauge symmetries

convolutional neural networks are *translation equivariant*

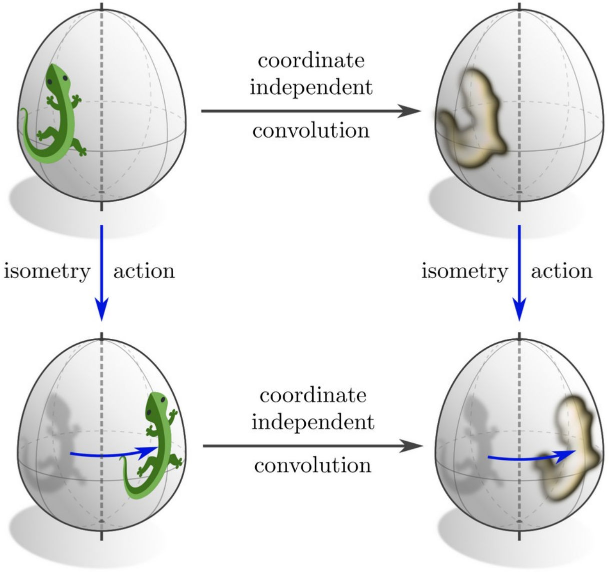


my research: generalize equivariant CNNs to … … larger symmetry groups

… more general manifolds

… gauge symmetries

convolutional neural networks are *translation equivariant*

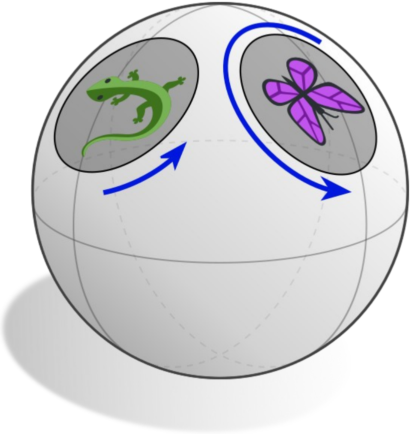

coordinate independent convolution
isometry action
isometry action
coordinate independent convolution

my research: generalize equivariant CNNs to … … larger symmetry groups

… more general manifolds

… gauge symmetries

convolutional neural networks are *translation equivariant*



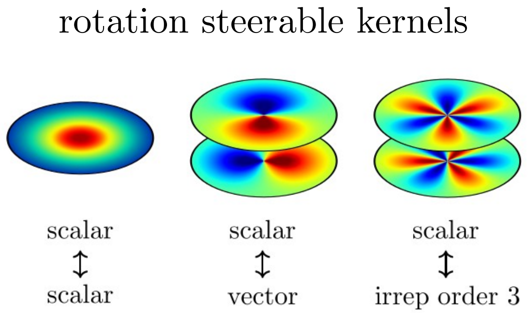
my research: generalize equivariant CNNs to … … larger symmetry groups

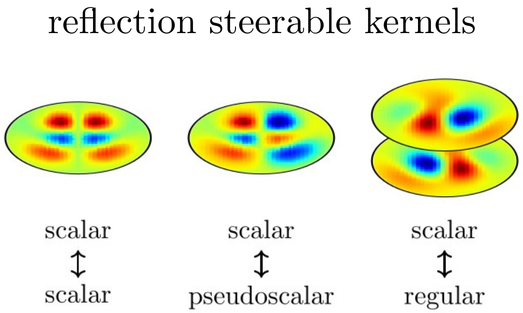
… more general manifolds

… gauge symmetries

convolutional neural networks are *translation equivariant*

how? → *equivariant / steerable kernels*





my research: generalize equivariant CNNs to … … larger symmetry groups

… more general manifolds

… gauge symmetries

Outline

Equivariant Neural Networks & Weight Sharing Patterns

Euclidean CNNs - Translation Equivariance

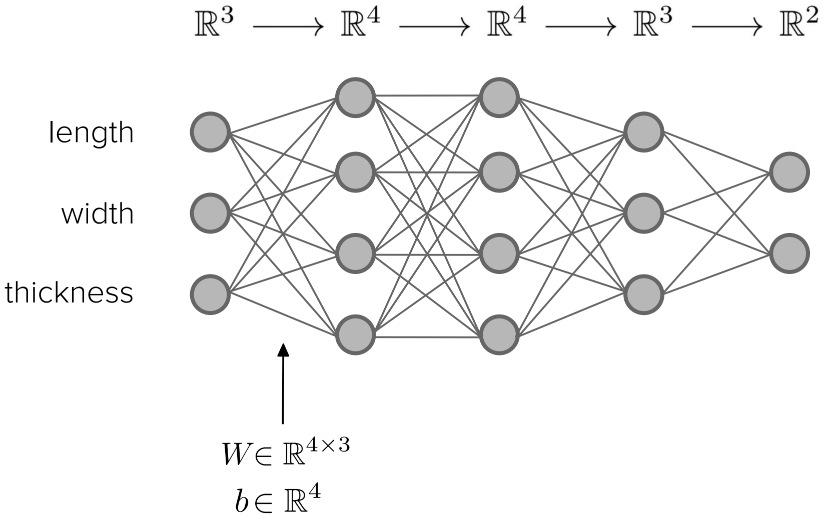
Euclidean CNNs - Affine Group Equivariance

G-steerable Kernels

Manifolds & Gauge Symmetries

Multilayer perceptrons & symmetries

MLPs are universal function approximators $\boldsymbol{f}: \mathbb{R}^N \to \mathbb{R}^M$



$\mathbb{R}^3 \longrightarrow \mathbb{R}^4 \longrightarrow \mathbb{R}^4 \longrightarrow \mathbb{R}^3 \longrightarrow \mathbb{R}^2$

length

width

thickness

$W \in \mathbb{R}^{4\times3}$

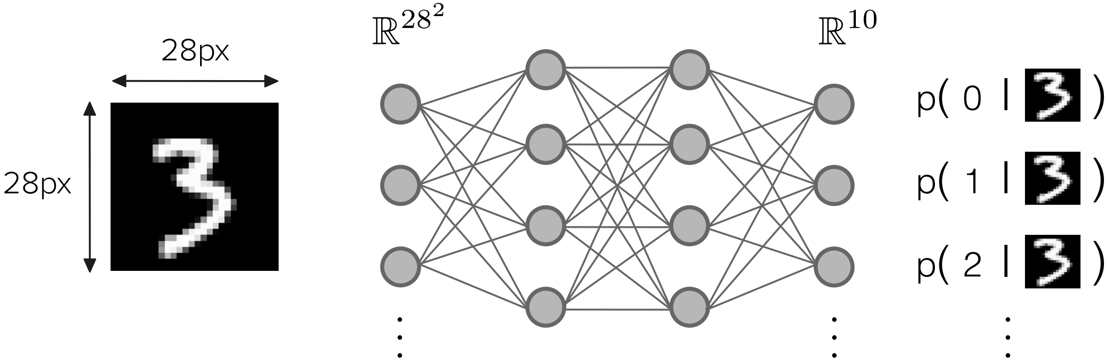
$b \in \mathbb{R}^4$

p(species A |)

p(species B |)

Multilayer perceptrons & symmetries

Images are high dimensional vectors ⟶ can be processed by MLPs

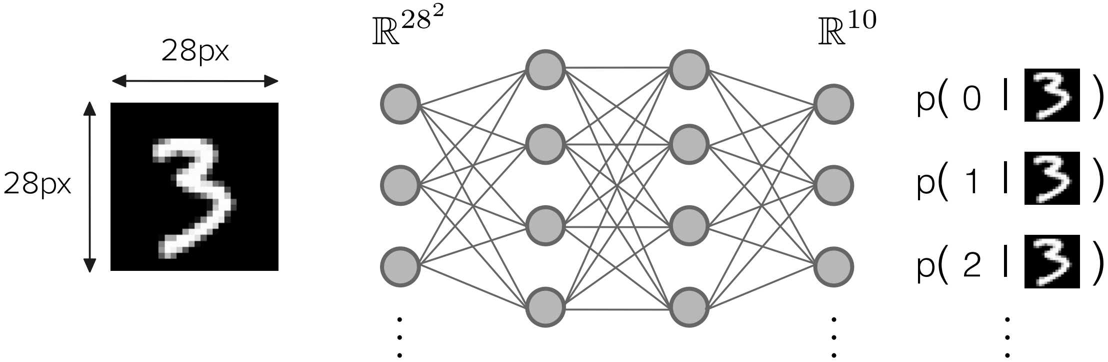


MLPs don't generalize over geometric transformations

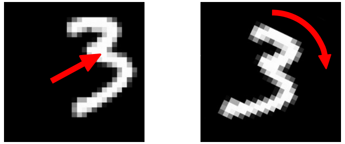
⟶ equivariance!

Multilayer perceptrons & symmetries

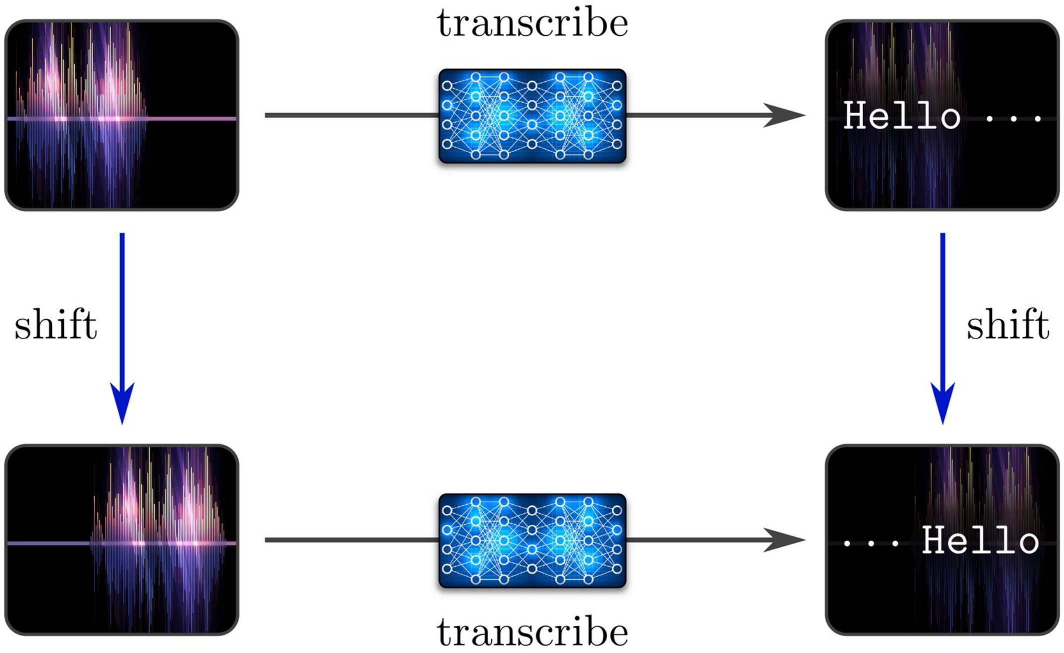
Images are high dimensional vectors ⟶ can be processed by MLPs

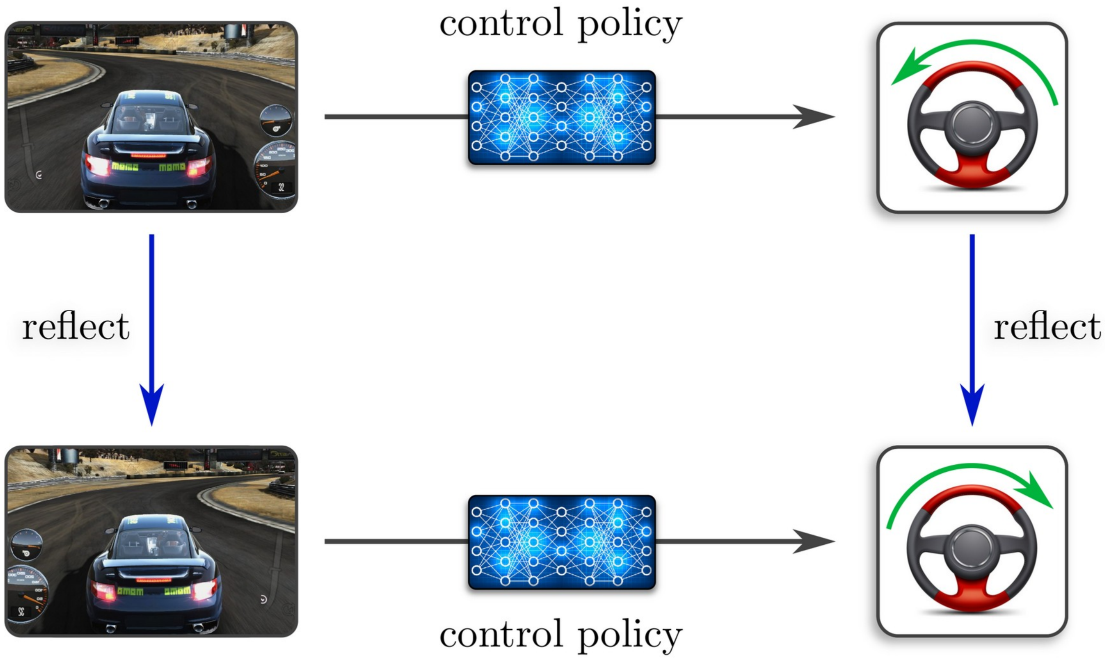


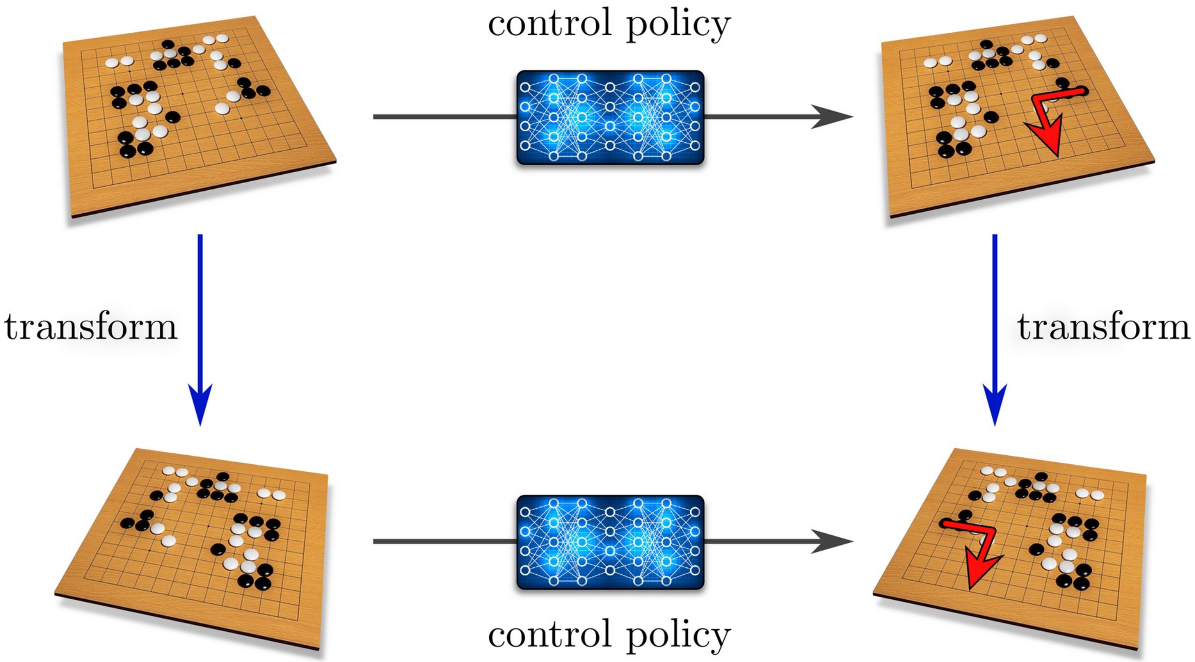
MLPs don’t generalize over geometric transformations

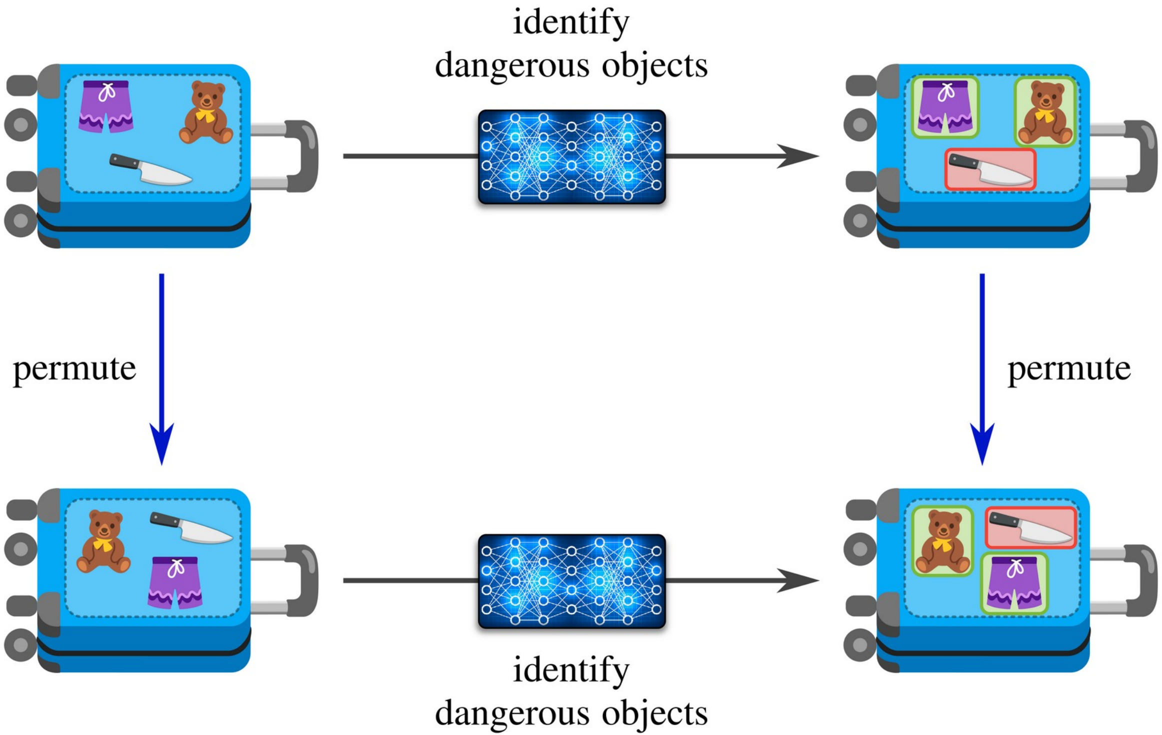


⟶ equivariance!

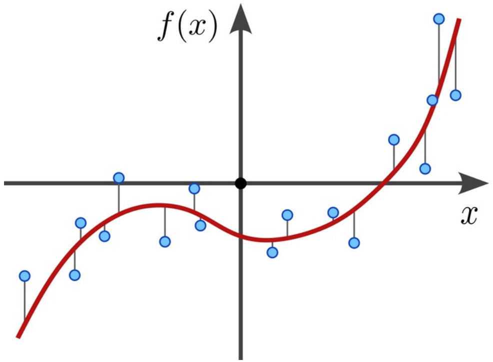

transcribe
Hello ...
shift
shift
... Hello
transcribe


control policy
reflect
reflect
control policy


control policy
transform
transform
control policy


identify
dangerous objects
permute
permute
identify
dangerous objects

Toy example: (anti)symmetric linear regression



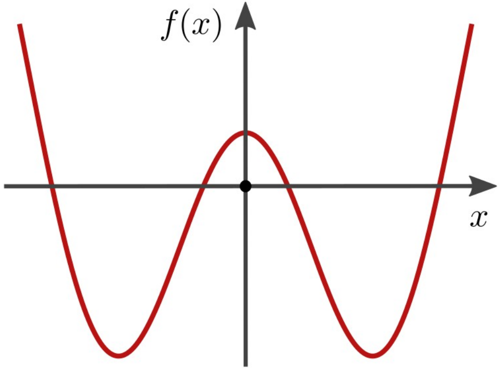
consider curve fitting via polynomial lineaer regression: $f(x) := \sum_n w_n \, x^n$

suppose the ground truth is known to be symmetric: $f(-x) = f(x)$

this prior knowledge is incorporated by constraining the model to $f(x) := \sum_{n \text{ even}} w_n \, x^n$

takeaway: equivariance ⟶ constraint on model parameters (“weight sharing patterns”)

Toy example: (anti)symmetric linear regression



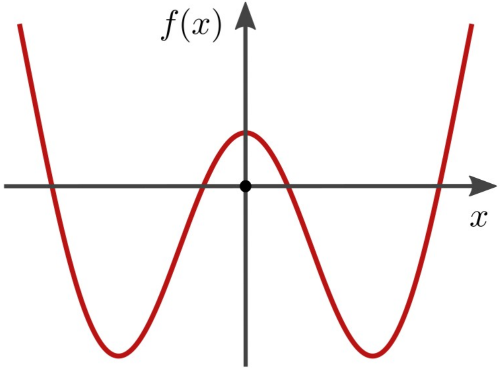
consider curve fitting via polynomial lineaer regression: $f(x) := \sum_n w_n \, x^n$

suppose the ground truth is known to be symmetric: $f(-x) = f(x)$

this prior knowledge is incorporated by constraining the model to $f(x) := \sum_{n \text{ even}} w_n \, x^n$

takeaway: equivariance ⟶ constraint on model parameters (“weight sharing patterns”)

Toy example: (anti)symmetric linear regression



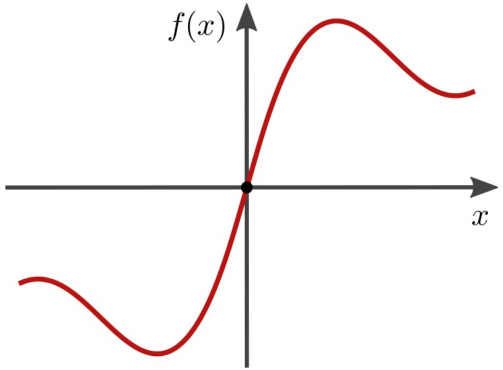
consider curve fitting via polynomial lineaer regression: $f(x) := \sum_n w_n \, x^n$

suppose the ground truth is known to be symmetric: $f(-x) = f(x)$

this prior knowledge is incorporated by constraining the model to $f(x) := \sum_{n \ \text{even}} w_n \, x^n$

takeaway: equivariance ⟶ constraint on model parameters ("weight sharing patterns")

Toy example: (anti)symmetric linear regression



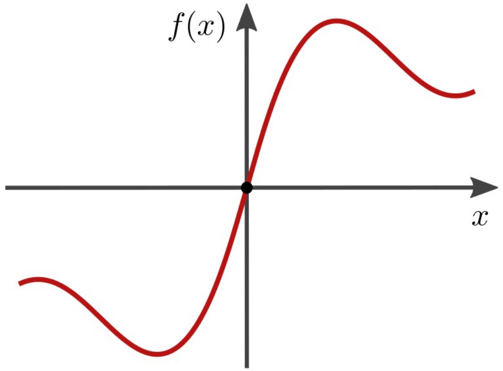
consider curve fitting via polynomial lineaer regression: $f(x) := \sum_n w_n\, x^n$

suppose the ground truth is known to be antisymmetric: $f(-x) = -f(x)$

this prior knowledge is incorporated by constraining the model to $f(x) := \sum_{n\ \text{odd}} w_n\, x^n$

takeaway: equivariance ⟶ constraint on model parameters ("weight sharing patterns")

Toy example: (anti)symmetric linear regression



consider curve fitting via polynomial lineaer regression: $f(x) := \sum_n w_n \, x^n$

suppose the ground truth is known to be antisymmetric: $f(-x) = -f(x)$

this prior knowledge is incorporated by constraining the model to $f(x) := \sum_{n \text{ odd}} w_n \, x^n$

takeaway: equivariance ⟶ constraint on model parameters (“weight sharing patterns”)

Layer-wise equivariant neural networks

a neural network is a sequence of layers (feed forward NN)

common approach: sequence of individually equivariant layers

step 1: specify feature spaces and group actions

step 2: find equivariant maps

$$\begin{tikzcd}
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"] \arrow[d, "t \rhd_{\mathrm{in}}"'] & \mathcal{F}_1 \arrow[r, "L_2"] \arrow[d, "t \rhd_1"'] & \mathcal{F}_2 \arrow[r, "L_3"] \arrow[d, "t \rhd_2"'] & \cdots \arrow[r, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, "L_N"] \arrow[d, "t \rhd_{N-1}"'] & \mathcal{F}_{\mathrm{out}} \arrow[d, "t \rhd_{\mathrm{out}}"] \\
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"'] & \mathcal{F}_1 \arrow[r, "L_2"'] & \mathcal{F}_2 \arrow[r, "L_3"'] & \cdots \arrow[r, "L_{N-1}"'] & \mathcal{F}_{N-1} \arrow[r, "L_N"'] & \mathcal{F}_{\mathrm{out}}
\end{tikzcd}$$

Layer-wise equivariant neural networks

an equivariant neural network is an equivariant sequence of layers

common approach: sequence of individually equivariant layers

step 1: specify feature spaces and group actions

step 2: find equivariant maps

$$\begin{tikzcd}
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"] \arrow[d, "t \triangleright_{\mathrm{in}}"'] & \mathcal{F}_1 \arrow[r, "L_2"] \arrow[d, "t \triangleright_1"'] & \mathcal{F}_2 \arrow[r, "L_3"] \arrow[d, "t \triangleright_2"'] & \cdots \arrow[r, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, "L_N"] \arrow[d, "t \triangleright_{N-1}"'] & \mathcal{F}_{\mathrm{out}} \arrow[d, "t \triangleright_{\mathrm{out}}"] \\
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"'] & \mathcal{F}_1 \arrow[r, "L_2"'] & \mathcal{F}_2 \arrow[r, "L_3"'] & \cdots \arrow[r, "L_{N-1}"'] & \mathcal{F}_{N-1} \arrow[r, "L_N"'] & \mathcal{F}_{\mathrm{out}}
\end{tikzcd}$$

Layer-wise equivariant neural networks

an equivariant neural network is an equivariant sequence of layers

common approach: layer-wise equivariance

step 1: specify feature spaces and group actions

step 2: find equivariant maps

$$\begin{tikzcd}
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"] \arrow[d, swap, "t \,\triangleright_{\mathrm{in}}"] & \mathcal{F}_1 \arrow[r, "L_2"] \arrow[d, swap, "t \,\triangleright_1"] & \mathcal{F}_2 \arrow[r, "L_3"] \arrow[d, swap, "t \,\triangleright_2"] & \cdots \arrow[r, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, "L_N"] \arrow[d, swap, "t \,\triangleright_{N-1}"] & \mathcal{F}_{\mathrm{out}} \arrow[d, "t \,\triangleright_{\mathrm{out}}"] \\
\mathcal{F}_{\mathrm{in}} \arrow[r, swap, "L_1"] & \mathcal{F}_1 \arrow[r, swap, "L_2"] & \mathcal{F}_2 \arrow[r, swap, "L_3"] & \cdots \arrow[r, swap, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, swap, "L_N"] & \mathcal{F}_{\mathrm{out}}
\end{tikzcd}$$

Layer-wise equivariant neural networks

an equivariant neural network is an equivariant sequence of layers

common approach: layer-wise equivariance

step 1: specify feature spaces and group actions

step 2: find equivariant maps

$$\begin{tikzcd}
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"] \arrow[d, "t \,\triangleright_{\mathrm{in}}"'] & \mathcal{F}_1 \arrow[r, "L_2"] \arrow[d, "t \,\triangleright_1"'] & \boxed{\mathcal{F}_2} \arrow[r, "L_3"] \arrow[d, "\boxed{t \,\triangleright_2}"'] & \cdots \arrow[r, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, "L_N"] \arrow[d, "t \,\triangleright_{N-1}"'] & \mathcal{F}_{\mathrm{out}} \arrow[d, "t \,\triangleright_{\mathrm{out}}"] \\
\mathcal{F}_{\mathrm{in}} \arrow[r, "L_1"'] & \mathcal{F}_1 \arrow[r, "L_2"'] & \boxed{\mathcal{F}_2} \arrow[r, "L_3"'] & \cdots \arrow[r, "L_{N-1}"'] & \mathcal{F}_{N-1} \arrow[r, "L_N"'] & \mathcal{F}_{\mathrm{out}}
\end{tikzcd}$$

Layer-wise equivariant neural networks

an equivariant neural network is an equivariant sequence of layers

common approach: layer-wise equivariance

step 1: specify feature spaces and group actions

step 2: find equivariant maps

$$\begin{tikzcd}
\mathcal{F}_{\text{in}} \arrow[r, "L_1"] \arrow[d, "t \rhd_{\text{in}}"'] & \mathcal{F}_1 \arrow[r, "L_2"] \arrow[d, "t \rhd_1"'] & \mathcal{F}_2 \arrow[r, "L_3"] \arrow[d, "t \rhd_2"'] & \cdots \arrow[r, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, "L_N"] \arrow[d, "t \rhd_{N-1}"'] & \mathcal{F}_{\text{out}} \arrow[d, "t \rhd_{\text{out}}"] \\
\mathcal{F}_{\text{in}} \arrow[r, "L_1"'] & \mathcal{F}_1 \arrow[r, "L_2"'] & \mathcal{F}_2 \arrow[r, "L_3"'] & \cdots \arrow[r, "L_{N-1}"'] & \mathcal{F}_{N-1} \arrow[r, "L_N"'] & \mathcal{F}_{\text{out}}
\end{tikzcd}$$

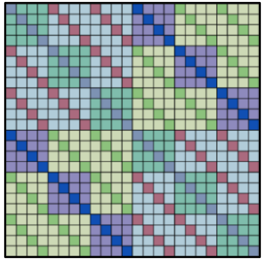
Equivariant linear layers (“intertwiners”)

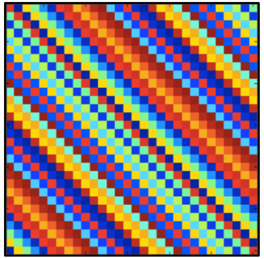
consider a linar layer / matrix multiplication $\mathbb{R}^n \to \mathbb{R}^m, \quad x \mapsto Wx$

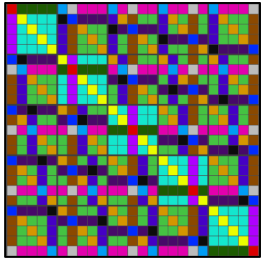
equivariance w.r.t. representations ρ_{in} on $\mathbb{R}^n$ and ρ_{out} on $\mathbb{R}^m$ means:

$$W\rho_{\text{in}}(g) \;=\; \rho_{\text{out}}(g)W \qquad \iff \qquad \left[\rho_{\text{in}}^{-\dagger} \otimes \rho_{\text{out}}\right](g)\,\mathrm{vec}(W) \;=\; \mathrm{vec}(W)$$

equivariant matrices are themselves invariants / symmetric ⟶ characterized by “weight sharing patterns”:



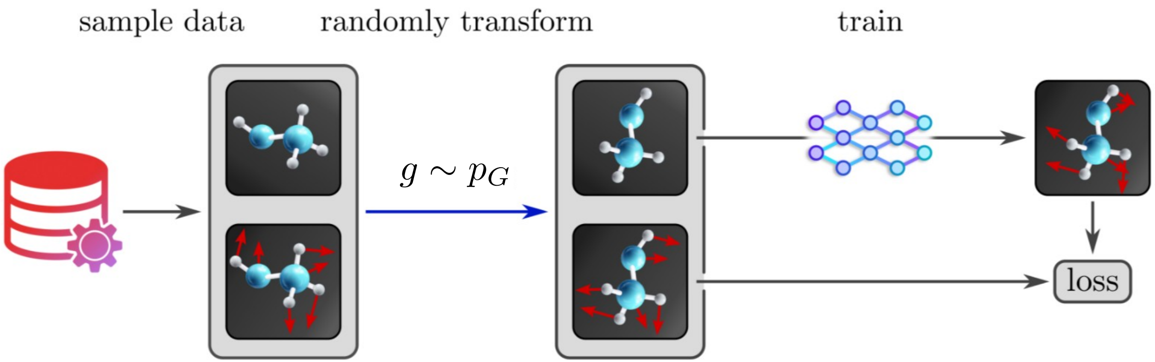




Alternative approaches - data augmentation

apply random transformations to training data + targets

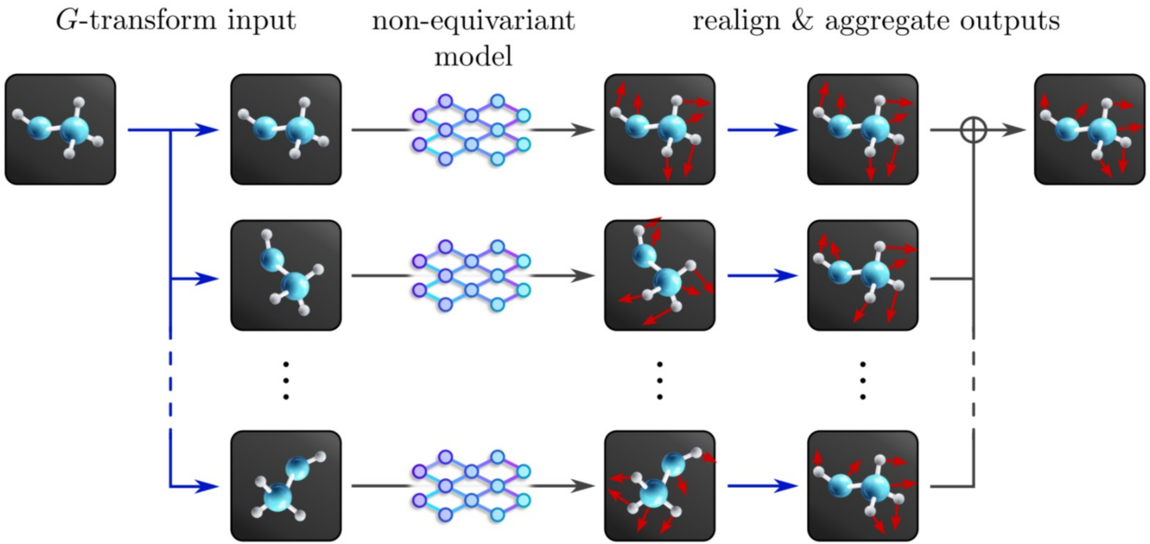
downside: less robust, converges slowly, worse final performance



Alternative approaches - group averaging

symmetrize model by applying it to any transformed inputs & aggregating outputs

downside: expensive for large groups

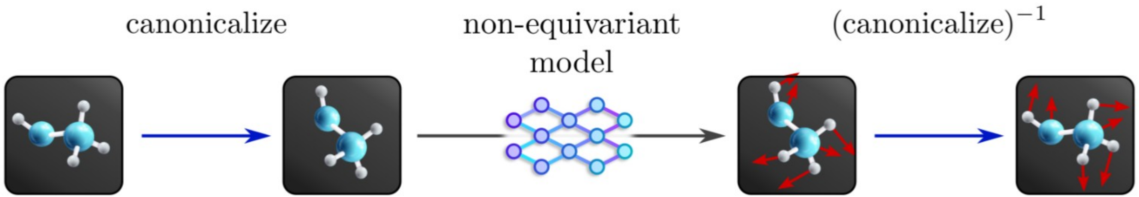


Alternative approaches - group averaging

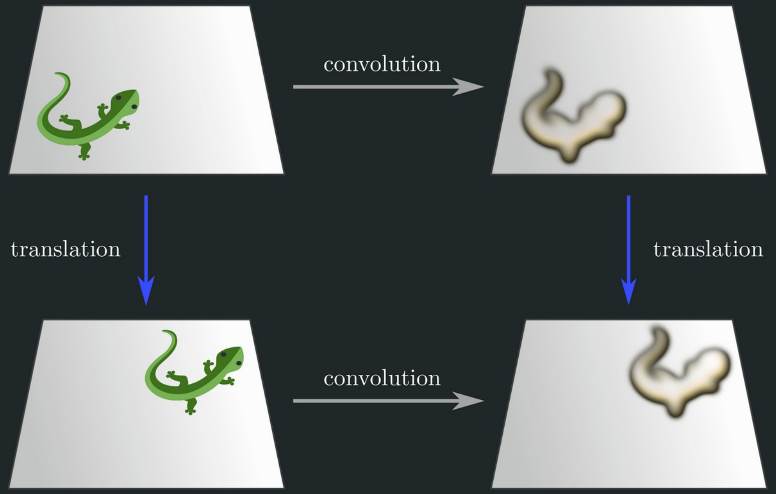
transform input to canonical pose before applying model

downside: non-robust,

continuous canonicalization (choice of orbit representative) topologically impossible



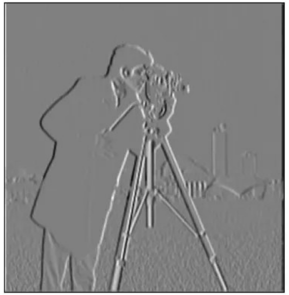
Translation equivariant Euclidean CNNs

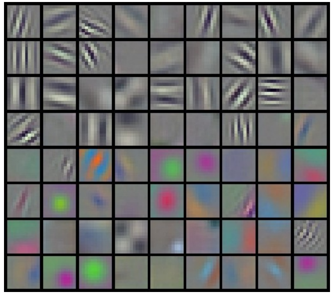


convolution for edge detection

learned filter bank

$$* \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} =$$





usually: convolution $\Longrightarrow$ equivariance (sufficiency)

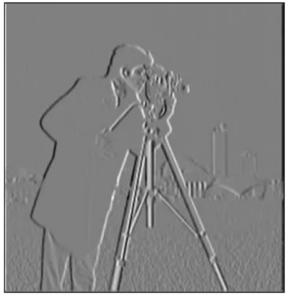
our approach: convolution $\Longleftarrow$ equivariance (necessity) convolution for edge detection

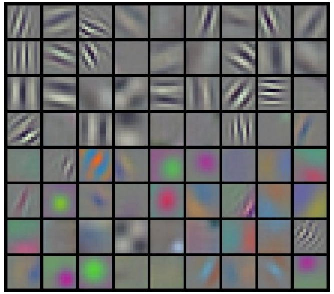
learned filter bank

$*$

1	0	-1
1	0	-1
1	0	-1

$=$





usually: convolution $\Longrightarrow$ equivariance (sufficiency)

our approach: convolution $\Longleftarrow$ equivariance (necessity) recall:

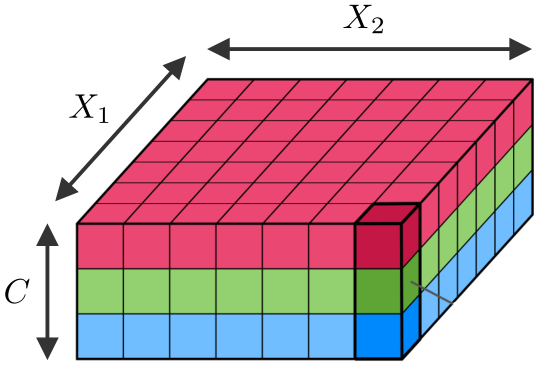
step 1: specify feature spaces and group actions ← feature maps

step 2: find equivariant maps ← convolutions / bias summation / ...

$$\begin{tikzcd}
\mathcal{F}_{\text{in}} \arrow[r, "L_1"] \arrow[d, "t \triangleright_{\text{in}}"'] & \boxed{\mathcal{F}_1} \arrow[r, "\boxed{L_2}"] \arrow[d, "\boxed{t \triangleright_1}"'] & \boxed{\mathcal{F}_2} \arrow[r, "L_3"] \arrow[d, "\boxed{t \triangleright_2}"'] & \cdots \arrow[r, "L_{N-1}"] & \mathcal{F}_{N-1} \arrow[r, "L_N"] \arrow[d, "t \triangleright_{N-1}"'] & \mathcal{F}_{\text{out}} \arrow[d, "t \triangleright_{\text{out}}"] \\
\mathcal{F}_{\text{in}} \arrow[r, "L_1"'] & \boxed{\mathcal{F}_1} \arrow[r, "\boxed{L_2}"'] & \boxed{\mathcal{F}_2} \arrow[r, "L_3"'] & \cdots \arrow[r, "L_{N-1}"'] & \mathcal{F}_{N-1} \arrow[r, "L_N"'] & \mathcal{F}_{\text{out}}
\end{tikzcd}$$

Feature maps as translation group representations

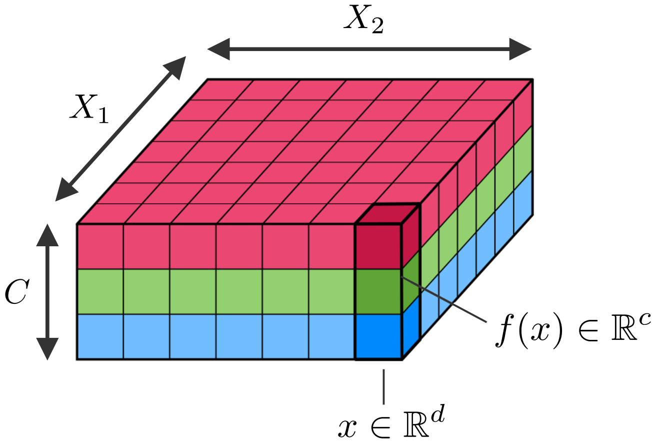
discretized feature maps on $\mathbb{R}^d$ are arrays of shape $(\overbrace{X_1, \ldots, X_d}^{\text{spatial / pixel dimensions}}, \overset{\text{feature channels}}{C})$



Feature maps as translation group representations

discretized feature maps on $\mathbb{R}^d$ are arrays of shape $(\overbrace{X_1, \dots, X_d}^{\text{spatial / pixel dimensions}}, \overset{\text{feature channels}}{C})$

continuous feature maps are functions $f : \mathbb{R}^d \to \mathbb{R}^c$ assigning features $f(x) \in \mathbb{R}^c$ to points $x \in \mathbb{R}^d$

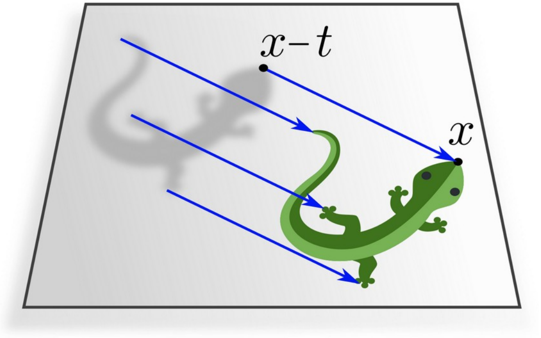


Feature maps as translation group representations

discretized feature maps on $\mathbb{R}^d$ are arrays of shape $(\overbrace{X_1, \dots, X_d}^{\text{spatial / pixel dimensions}},\ \overset{\text{feature channels}}{C})$

continuous feature maps are functions $f: \mathbb{R}^d \to \mathbb{R}^c$ assigning features $f(x) \in \mathbb{R}^c$ to points $x \in \mathbb{R}^d$

feature maps carry a translation *group action* $\big[t \triangleright f\big](x) := f(x-t)$



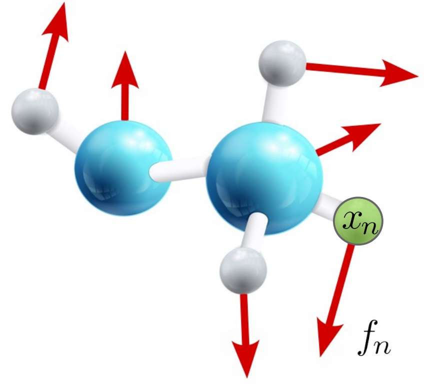
Feature maps as translation group representations

discretized feature maps on $\mathbb{R}^d$ are arrays of shape $(\overbrace{X_1, \dots, X_d}^{\text{spatial / pixel dimensions}},\ \overset{\text{feature channels}}{C})$

continuous feature maps are functions $f : \mathbb{R}^d \to \mathbb{R}^c$ assigning features $f(x) \in \mathbb{R}^c$ to points $x \in \mathbb{R}^d$

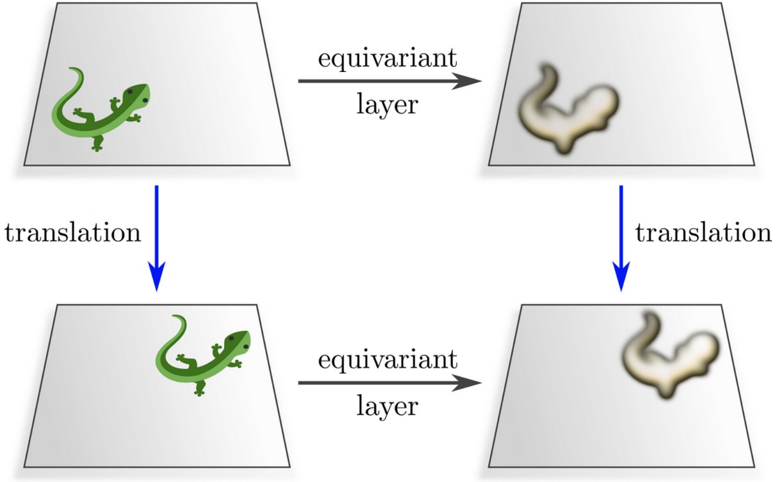
feature maps carry a translation *group action* $\big[t \triangleright f\big](x) := f(x - t)$

this definition includes *point clouds*: $f(x) = \sum_n f_n\, \delta(x - x_n)$



Translation equivariant CNN layers

conventional CNN layer $:=$ any translation equivariant function between feature maps



Translation equivariant CNN layers

conventional CNN layer $:=$ any translation equivariant function between feature maps

general result: translation equivariance $\Longrightarrow$ spatially invariant neural connectivity (“weight sharing”)

examples:

linear map $\xrightarrow{\text{equivariance}}$ convolution (Theorem 3.2.1)

bias field summation $\xrightarrow{\text{equivariance}}$ shared bias (Theorem 3.2.2)

independent nonlinearities $\xrightarrow{\text{equivariance}}$ shared nonlinearity (Theorem 3.2.3)

different pooling windows $\xrightarrow{\text{equivariance}}$ shared pooling window (Theorem 3.2.4)

Theorems proven in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 3.2)

Translation equivariant CNN layers

conventional CNN layer $:=$ any translation equivariant function between feature maps

general result: translation equivariance $\Longrightarrow$ spatially invariant neural connectivity (“weight sharing”)

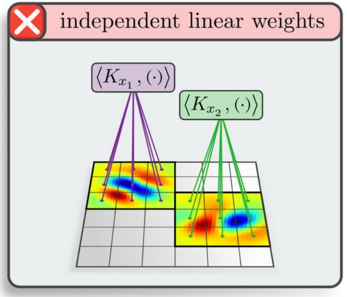
examples:

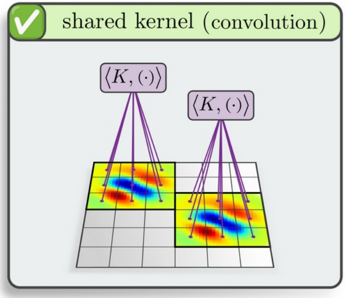
linear map $\xrightarrow{\text{equivariance}}$ convolution (Theorem 3.2.1)

bias field summation $\xrightarrow{\text{equivariance}}$ shared bias (Theorem 3.2.2)

independent nonlinearities $\xrightarrow{\text{equivariance}}$ shared nonlinearity (Theorem 3.2.3)

different pooling windows $\xrightarrow{\text{equivariance}}$ shared pooling window (Theorem 3.2.4)


☒ independent linear weights
$\langle K_{x_1}, (\cdot)\rangle$
$\langle K_{x_2}, (\cdot)\rangle$


✓ shared kernel (convolution)
$\langle K, (\cdot)\rangle$
$\langle K, (\cdot)\rangle$

Theorems proven in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 3.2)

Translation equivariant CNN layers

conventional CNN layer $:=$ any translation equivariant function between feature maps

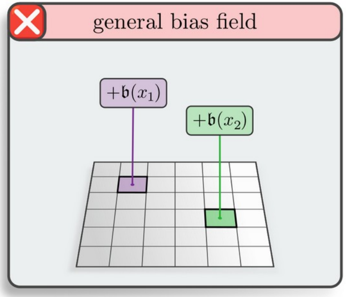
general result: translation equivariance $\Longrightarrow$ spatially invariant neural connectivity (“weight sharing”)

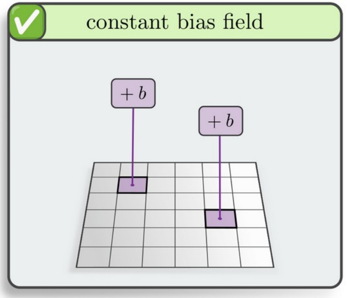
examples: linear map $\xrightarrow{\text{equivariance}}$ convolution (Theorem 3.2.1)

bias field summation $\xrightarrow{\text{equivariance}}$ shared bias (Theorem 3.2.2)

independent nonlinearities $\xrightarrow{\text{equivariance}}$ shared nonlinearity (Theorem 3.2.3)

different pooling windows $\xrightarrow{\text{equivariance}}$ shared pooling window (Theorem 3.2.4)





Theorems proven in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 3.2)

Translation equivariant CNN layers

conventional CNN layer $:=$ any translation equivariant function between feature maps

general result: translation equivariance $\Longrightarrow$ spatially invariant neural connectivity ("weight sharing")

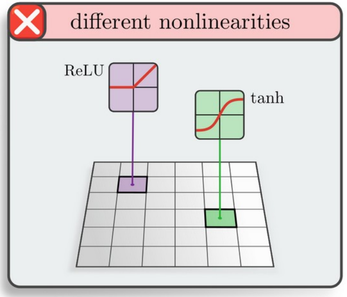
examples:

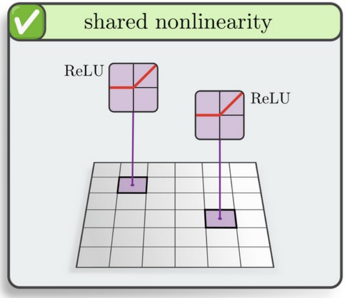
linear map $\xrightarrow{\text{equivariance}}$ convolution (Theorem 3.2.1)

bias field summation $\xrightarrow{\text{equivariance}}$ shared bias (Theorem 3.2.2)

independent nonlinearities $\xrightarrow{\text{equivariance}}$ shared nonlinearity (Theorem 3.2.3)

different pooling windows $\xrightarrow{\text{equivariance}}$ shared pooling window (Theorem 3.2.4)



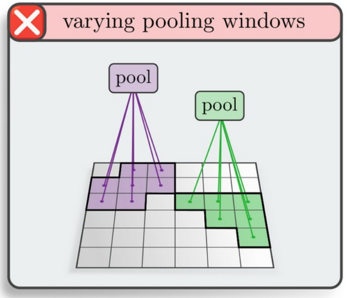


Theorems proven in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 3.2)

Translation equivariant CNN layers

conventional CNN layer $:=$ any translation equivariant function between feature maps

general result: translation equivariance $\Longrightarrow$ spatially invariant neural connectivity (“weight sharing”)



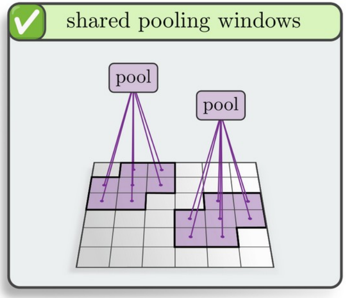
examples:

linear map $\xrightarrow{\text{equivariance}}$ convolution (Theorem 3.2.1)

bias field summation $\xrightarrow{\text{equivariance}}$ shared bias (Theorem 3.2.2)

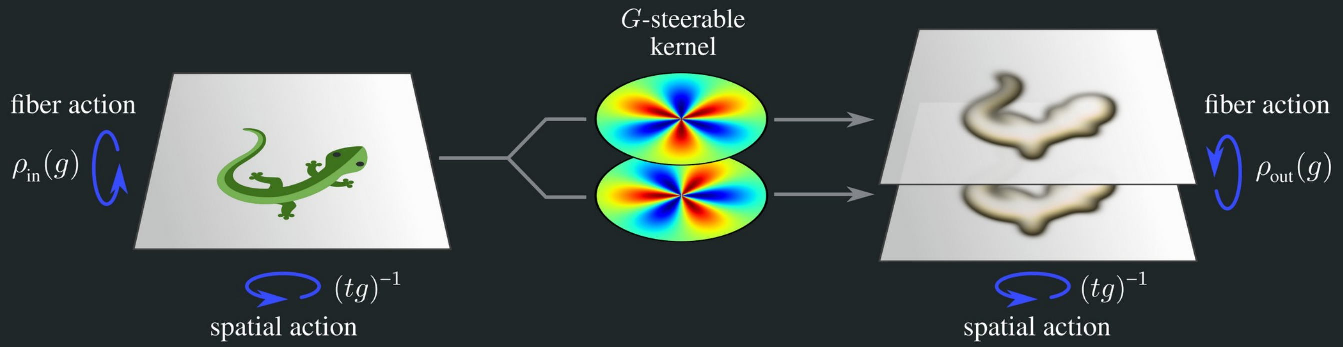
independent nonlinearities $\xrightarrow{\text{equivariance}}$ shared nonlinearity (Theorem 3.2.3)

different pooling windows $\xrightarrow{\text{equivariance}}$ shared pooling window (Theorem 3.2.4)



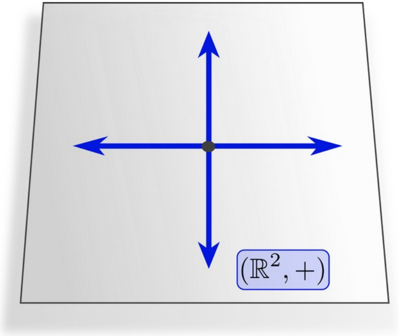
Theorems proven in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 3.2)

Affine group equivariant Euclidean CNNs



Symmetries of Euclidean space - translations

translations $(\mathbb{R}^d, +)$

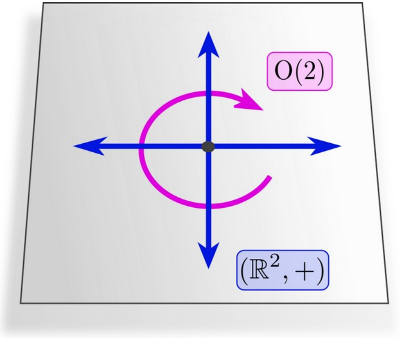


Symmetries of Euclidean space - isometries

translations

rotations + reflections

Euclidean group $\mathrm{E}(d) := (\mathbb{R}^d, +) \rtimes \mathrm{O}(d)$

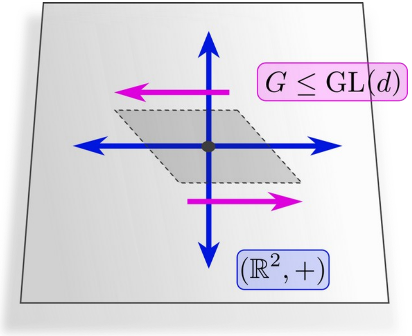


Symmetries of Euclidean space - affine

translations

structure group (rotations / reflections / scaling / shearing / ...)

Affine groups $\mathrm{Aff}(G) := (\mathbb{R}^d,+) \rtimes G \qquad G \leq \mathrm{GL}(d)$



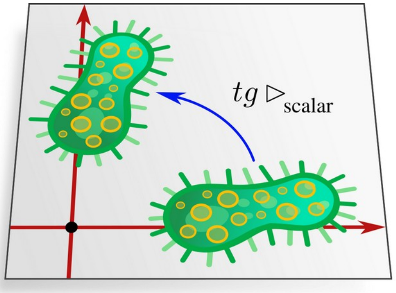
Feature fields as affine group representations

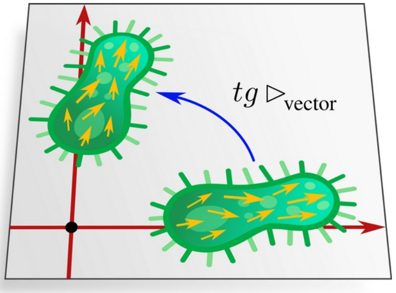
feature vector fields ... ... are functions $f : \mathbb{R}^d \to \mathbb{R}^c$ (like feature maps)

... carry an Aff(G)-action (details depend on *field type* ρ)

examples: scalar fields $s : \mathbb{R}^d \to \mathbb{R}^1$ transform like: $\big[(tg) \rhd s\big](x) \; = \; s\left((tg)^{-1}x\right)$

tangent vector fields $v : \mathbb{R}^d \to \mathbb{R}^d$ transform like: $\big[(tg) \rhd v\big](x) \; = \; g \cdot v\left((tg)^{-1}x\right)$





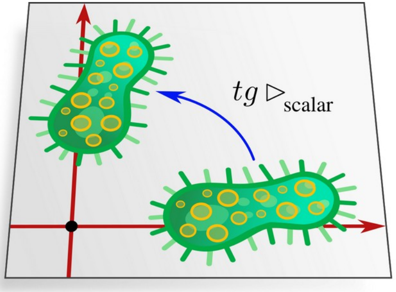
Feature fields as affine group representations

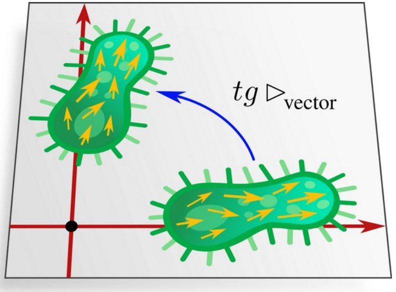
feature vector fields ... ... are functions $f : \mathbb{R}^d \to \mathbb{R}^c$ (like feature maps)

... carry an Aff(G)-action (details depend on *field type* ρ)

examples: scalar fields $s : \mathbb{R}^d \to \mathbb{R}^1$ transform like: $\big[(tg) \rhd s\big](x) = s\big((tg)^{-1}x\big)$

tangent vector fields $v : \mathbb{R}^d \to \mathbb{R}^d$ transform like: $\big[(tg) \rhd v\big](x) = g \cdot v\big((tg)^{-1}x\big)$





Feature fields as affine group representations

feature vector fields ... ... are functions $f : \mathbb{R}^d \to \mathbb{R}^c$ (like feature maps)

... carry an Aff(G)-action (details depend on *field type* ρ)

examples: scalar fields $s : \mathbb{R}^d \to \mathbb{R}^1$ transform like: $\big[(tg) \triangleright s\big](x) \;=\; \boxed{1 \cdot}\; s\big(\boxed{(tg)^{-1}x}\big)$

tangent vector fields $v : \mathbb{R}^d \to \mathbb{R}^d$ transform like: $\big[(tg) \triangleright v\big](x) \;=\; \boxed{g \cdot}\; v\big(\boxed{(tg)^{-1}x}\big)$

Aff(G) acts here by...

1) moving feature vectors on $\mathbb{R}^d$

2) G-transforming feature vectors in $\mathbb{R}^c$

Feature fields as affine group representations

feature vector fields ... ... are functions $f : \mathbb{R}^d \to \mathbb{R}^c$ (like feature maps)

... carry an Aff(G)-action (details depend on *field type* ρ)

examples:

scalar fields $s : \mathbb{R}^d \to \mathbb{R}^1$ transform like: $\big[(tg) \triangleright s\big](x) \;=\; 1 \cdot s\big((tg)^{-1}x\big)$

tangent vector fields $v : \mathbb{R}^d \to \mathbb{R}^d$ transform like: $\big[(tg) \triangleright v\big](x) \;=\; g \cdot v\big((tg)^{-1}x\big)$

ρ-feature fields $f : \mathbb{R}^d \to \mathbb{R}^c$ transform like: $\big[(tg) \triangleright f\big](x) \;=\; \rho(g)\, f\big((tg)^{-1}x\big)$

G-representation

Aff(G) acts here by...

1) moving feature vectors on $\mathbb{R}^d$

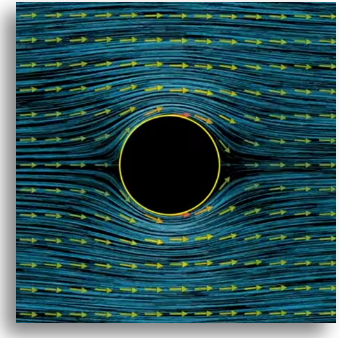
2) G-transforming feature vectors in $\mathbb{R}^c$

Feature fields - examples

fluid flow

(vector)

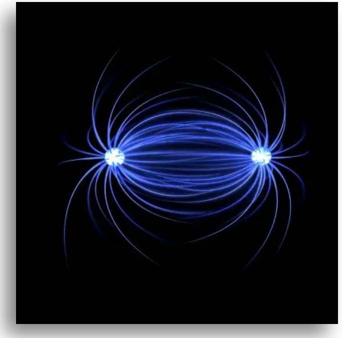
$\rho(g) = g$



EM field strength

(bivector / anti-symm. (0,2)-tensor)

(subspace of) $\rho(g) = g^{-\top} \otimes g^{-\top}$



diffusion tensor image

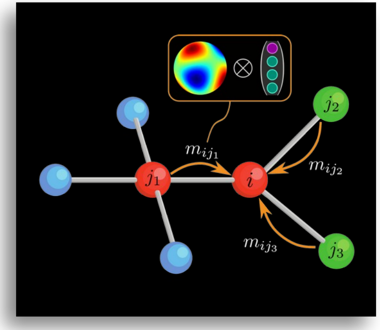
(symm. pos. def. (1,1)-tensor)

(subspace of) $\rho(g) = g \otimes g^{-\top}$

Tensor Field Net features

(SO(3)-irreps)

$\rho(g) = D^j(g)$

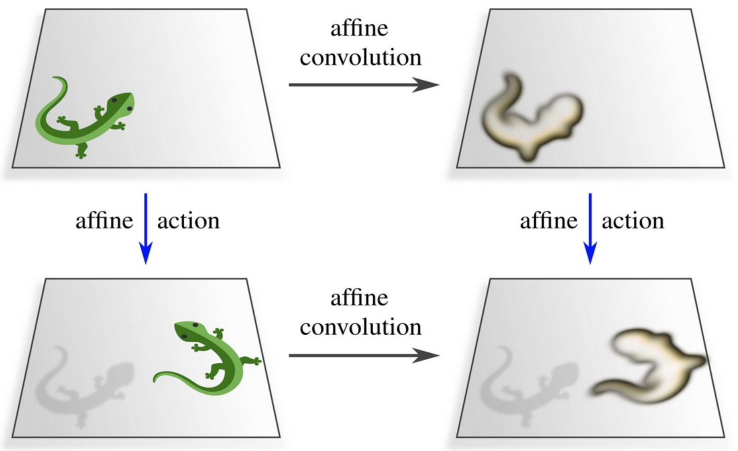


Affine group equivariant CNN layers

steerable CNN layer $:=$ any Aff(G)-equivariant function between feature fields

input field,
type ρ_{in}

output field,
type ρ_{out}



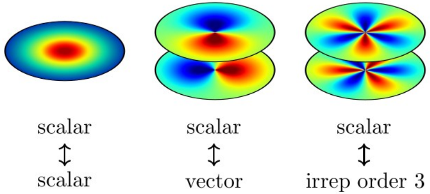
Affine group equivariant CNN layers

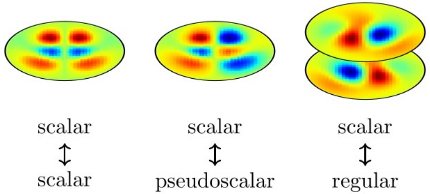
steerable CNN layer $:=$ any Aff(G)-equivariant function between feature fields

general result: Aff(G)-equivariance $\Longrightarrow$ Aff(G)-invariant neural connectivity

1) spatial weight sharing —— $(\mathbb{R}^d, +) \rtimes G =: \mathrm{Aff}(G)$

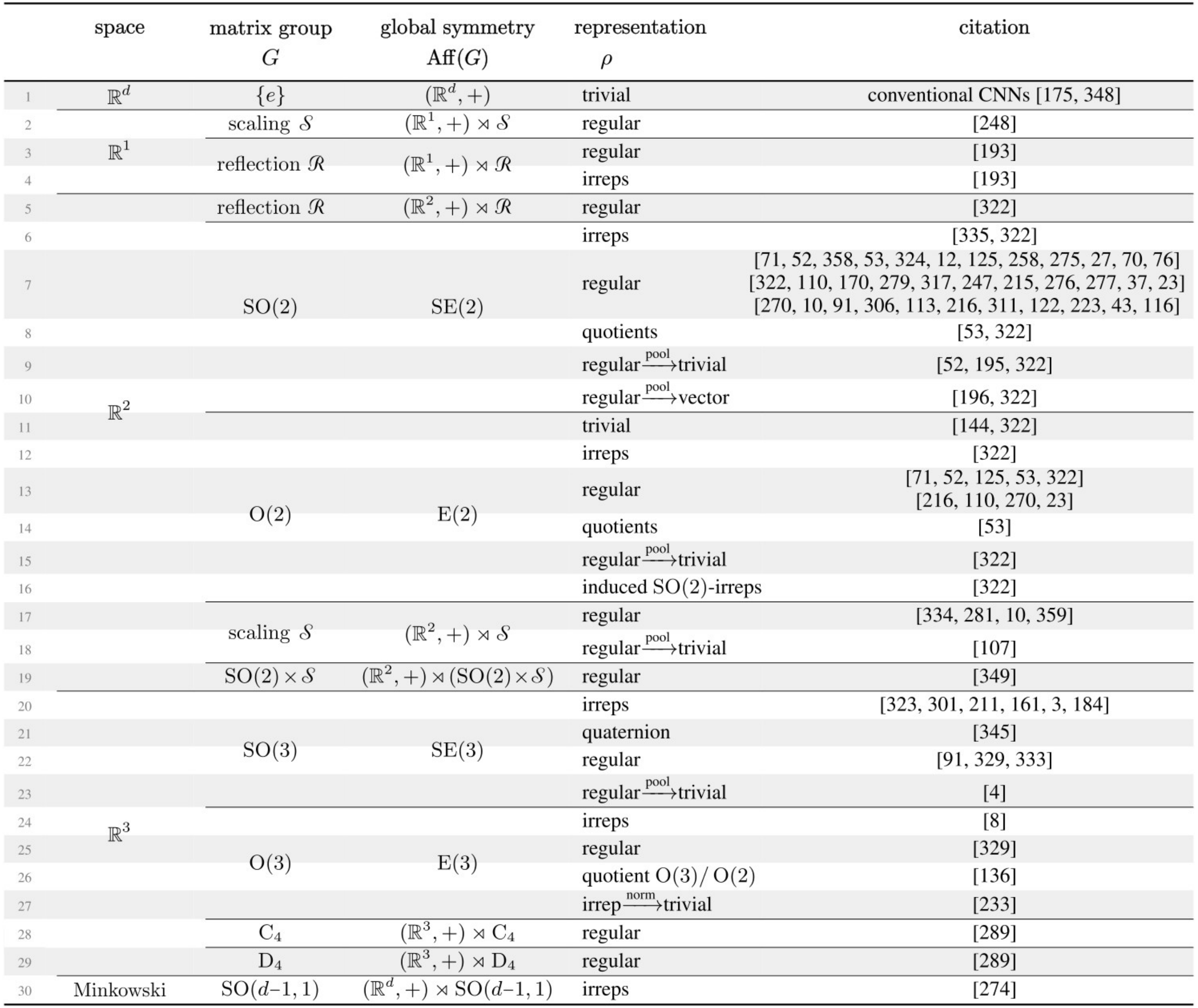
2) G-steerability





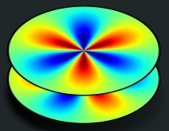
Theorems proven in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 4.3)

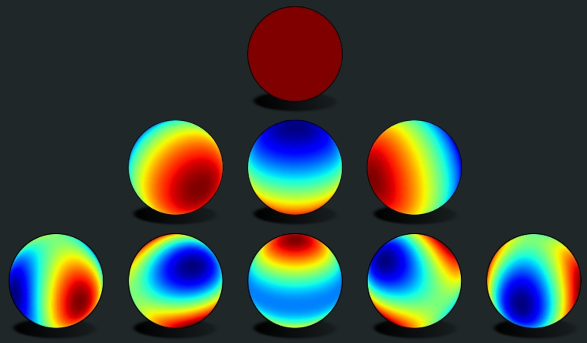
taxonomy of equivariant CNNs

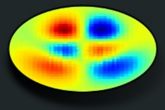


	space	matrix group	global symmetry	representation	citation
		G	$\mathrm{Aff}(G)$	ρ	
1	$\mathbb{R}^d$	$\{e\}$	$(\mathbb{R}^d,+)$	trivial	conventional CNNs [175, 348]
2	$\mathbb{R}^1$	scaling $\mathcal{S}$	$(\mathbb{R}^1,+)\rtimes\mathcal{S}$	regular	[248]
3		reflection $\mathcal{R}$	$(\mathbb{R}^1,+)\rtimes\mathcal{R}$	regular	[193]
4				irreps	[193]
5	$\mathbb{R}^2$	reflection $\mathcal{R}$	$(\mathbb{R}^2,+)\rtimes\mathcal{R}$	regular	[322]
6		SO(2)	SE(2)	irreps	[335, 322]
7				regular	[71, 52, 358, 53, 324, 12, 125, 258, 275, 27, 70, 76] [322, 110, 170, 279, 317, 247, 215, 276, 277, 37, 23] [270, 10, 91, 306, 113, 216, 311, 122, 223, 43, 116]
8				quotients	[53, 322]
9				regular$\xrightarrow{\text{pool}}$trivial	[52, 195, 322]
10				regular$\xrightarrow{\text{pool}}$vector	[196, 322]
11		O(2)	E(2)	trivial	[144, 322]
12				irreps	[322]
13				regular	[71, 52, 125, 53, 322] [216, 110, 270, 23]
14				quotients	[53]
15				regular$\xrightarrow{\text{pool}}$trivial	[322]
16				induced SO(2)-irreps	[322]
17		scaling $\mathcal{S}$	$(\mathbb{R}^2,+)\rtimes\mathcal{S}$	regular	[334, 281, 10, 359]
18				regular$\xrightarrow{\text{pool}}$trivial	[107]
19		$\mathrm{SO}(2)\times\mathcal{S}$	$(\mathbb{R}^2,+)\rtimes(\mathrm{SO}(2)\times\mathcal{S})$	regular	[349]
20	$\mathbb{R}^3$	SO(3)	SE(3)	irreps	[323, 301, 211, 161, 3, 184]
21				quaternion	[345]
22				regular	[91, 329, 333]
23				regular$\xrightarrow{\text{pool}}$trivial	[4]
24		O(3)	E(3)	irreps	[8]
25				regular	[329]
26				quotient O(3)/ O(2)	[136]
27				irrep$\xrightarrow{\text{norm}}$trivial	[233]
28		C_4	$(\mathbb{R}^3,+)\rtimes\mathrm{C}_4$	regular	[289]
29		D_4	$(\mathbb{R}^3,+)\rtimes\mathrm{D}_4$	regular	[289]
30	Minkowski	$\mathrm{SO}(d{-}1,1)$	$(\mathbb{R}^d,+)\rtimes\mathrm{SO}(d{-}1,1)$	irreps	[274]

G-steerable kernels



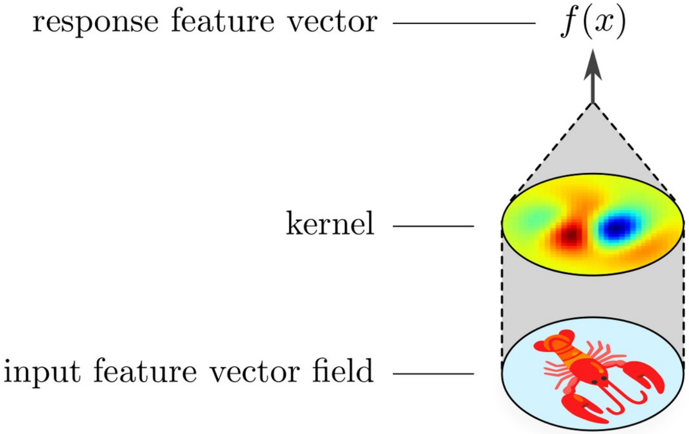




G-steerable kernels - intuition

convolution kernels summarize their field of view around $x \in \mathbb{R}^d$ into a feature vector $f(x) \in \mathbb{R}^{c_\mathrm{out}}$

G-steerable kernels guarantee: G-trafo of their input field of view $\Rightarrow$ G-trafo of the output feature vector



response feature vector — $f(x)$

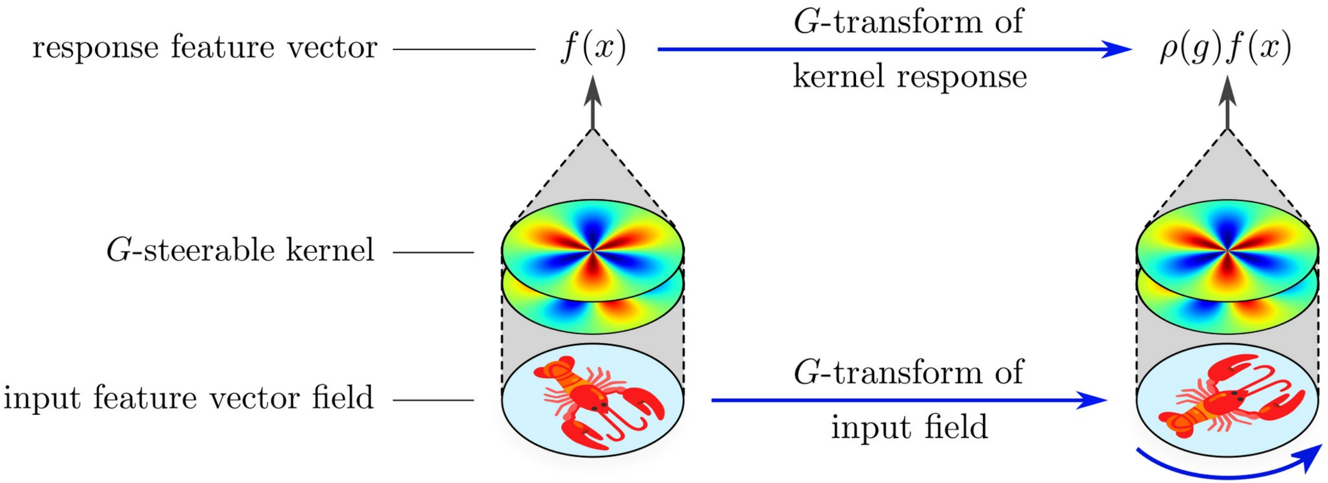
kernel

input feature vector field

G-steerable kernels - intuition

convolution kernels summarize their field of view around $x \in \mathbb{R}^d$ into a feature vector $f(x) \in \mathbb{R}^{c_{\mathrm{out}}}$

G-steerable kernels guarantee: G-trafo of their input field of view $\Rightarrow$ G-trafo of the output feature vector

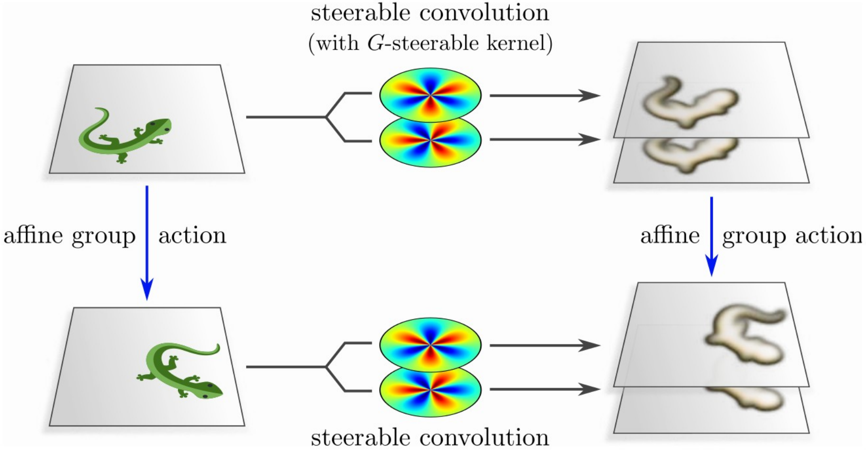


G-steerable kernels - intuition

convolution kernels summarize their field of view around $x \in \mathbb{R}^d$ into a feature vector $f(x) \in \mathbb{R}^{c_{\text{out}}}$

G-steerable kernels guarantee: G-trafo of their input field of view $\Rightarrow$ G-trafo of the output feature vector

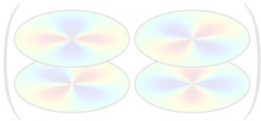
convolutions with G-steerable kernels are Aff(G)-equivariant



G-steerable kernels - mathematical definition

discretized convolution kernels are arrays of shape $(\underbrace{X_1, \ldots, X_d}_{\text{spatial / pixel dimensions}}, \underbrace{C_{\text{out}}, C_{\text{in}}}_{\text{matrix dimensions, mapping in} \to \text{out features}})$

continuous kernels are matrix-valued fields: $K :\ \mathbb{R}^d \longrightarrow \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$



G-steerable kernels satisfy a linear G-equivariance constraint:

$$K(gx) \;=\; \frac{1}{|\det g|} \rho_{\text{out}}(g) K(x) \rho_{\text{in}}(g)^{-1} \qquad \forall\ g \in G,\ x \in \mathbb{R}^d$$

G-action on spatial dimension

G-action on matrix rows / columns

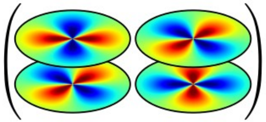
G-steerable kernels - mathematical definition

discretized convolution kernels are arrays of shape

$$(\underbrace{X_1, \ldots, X_d}_{\text{spatial / pixel dimensions}}, \underbrace{C_{\text{out}}, C_{\text{in}}}_{\text{matrix dimensions, mapping in} \to \text{out features}})$$

continuous kernels are matrix-valued fields:

$$K: \ \mathbb{R}^d \longrightarrow \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$$



G-steerable kernels satisfy a linear G-equivariance constraint:

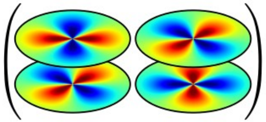
$$\underbrace{K(gx)}_{\text{G-action on spatial dimension}} \ = \ \frac{1}{|\det g|} \underbrace{\rho_{\text{out}}(g) K(x) \rho_{\text{in}}(g)^{-1}}_{\text{G-action on matrix rows / columns}} \qquad \forall\, g \in G,\ x \in \mathbb{R}^d$$

G-steerable kernels - mathematical definition

discretized convolution kernels are arrays of shape $(\underbrace{X_1, \dots, X_d}_{\text{spatial / pixel dimensions}}, \underbrace{C_{\text{out}}, C_{\text{in}}}_{\text{matrix dimensions, mapping in} \to \text{out features}})$

continuous kernels are matrix-valued fields:

$$K: \; \mathbb{R}^d \longrightarrow \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$$



G-steerable kernels satisfy a linear G-equivariance constraint:

$$K(gx) \;=\; \frac{1}{|\det g|} \rho_{\text{out}}(g) K(x) \rho_{\text{in}}(g)^{-1} \qquad \forall\, g \in G,\; x \in \mathbb{R}^d$$

G-action on spatial dimension

G-action on matrix rows / columns

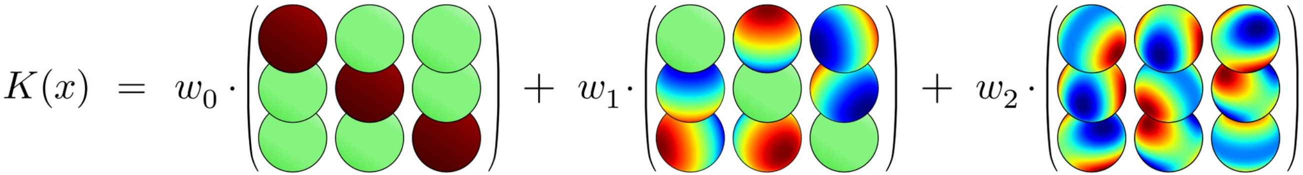
G-steerable kernel bases

convolution kernels $K: \mathbb{R}^d \longrightarrow \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$ form a *vector space*

the G-steerability constraint is *linear*

$\Longrightarrow$ steerable kernels ... ... form a vector subspace

... can be expanded from a steerable basis set (params = expansion weights)



Reflection steerable kernels

reflection group: $\mathbb{Z}_2 \; = \; \{e, r\}$ with $r^2 = e$ or $r^{-1} = r$

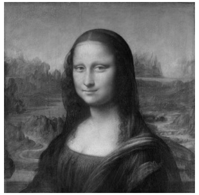
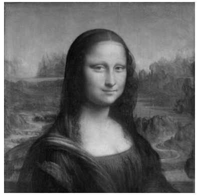
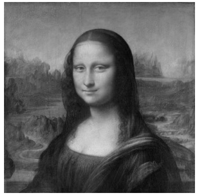
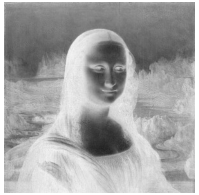
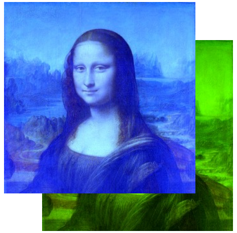
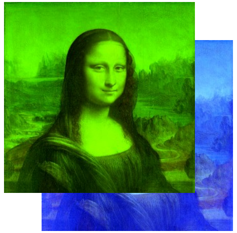
the general G-steerability constraint

$$K(gx) \;=\; \frac{1}{\cancel{|\det g|}} \rho_{\text{out}}(g) K(x) \rho_{\text{in}}(g)^{\cancel{-1}} \qquad \forall\; \cancel{g \in G},\; x \in \mathbb{R}^d$$

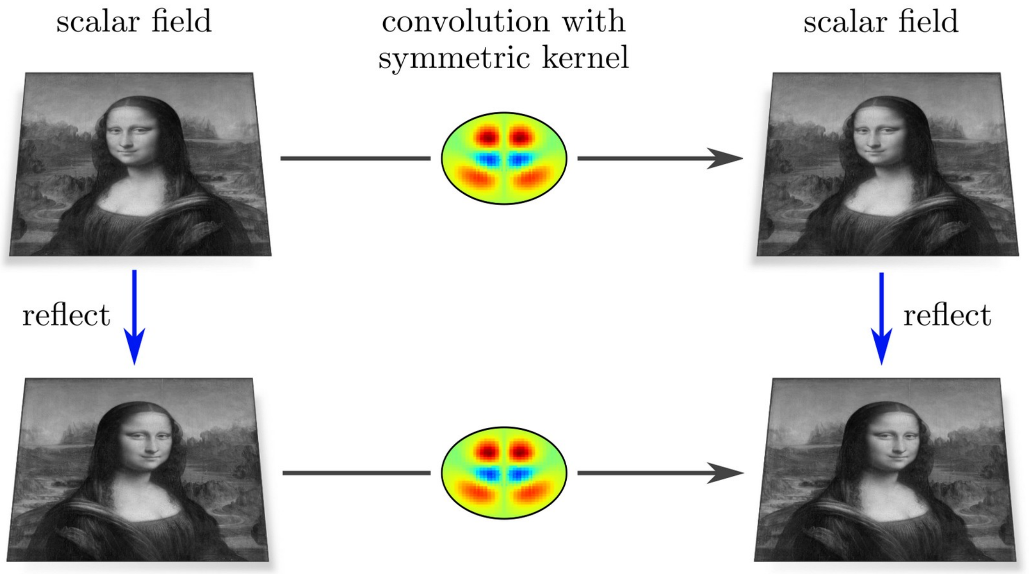
simplifies to:

$$K(rx) \;=\; \rho_{\text{out}}(r) K(x) \rho_{\text{in}}(r) \qquad \forall\; x \in \mathbb{R}^d$$

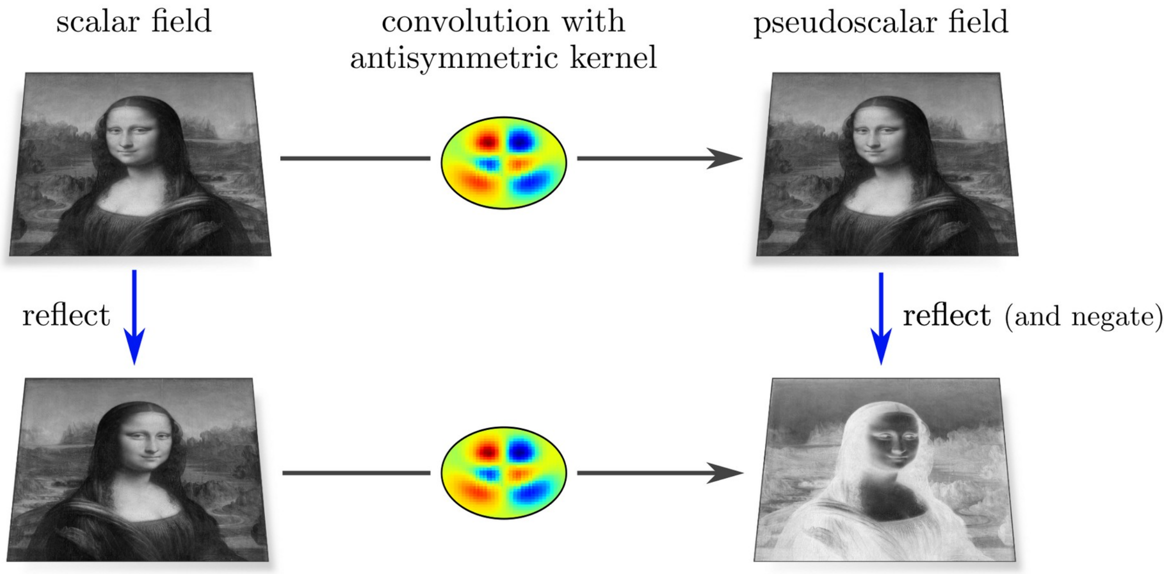
Reflection steerable kernels

field type ρ	$\rho(e)$	$\rho(\mathrm{reflect})$	original field	reflected field
trivial / scalar	(1)	(1)		
sign-flip / pseudo-scalar	(1)	(-1)		
regular	$\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$	$\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$		

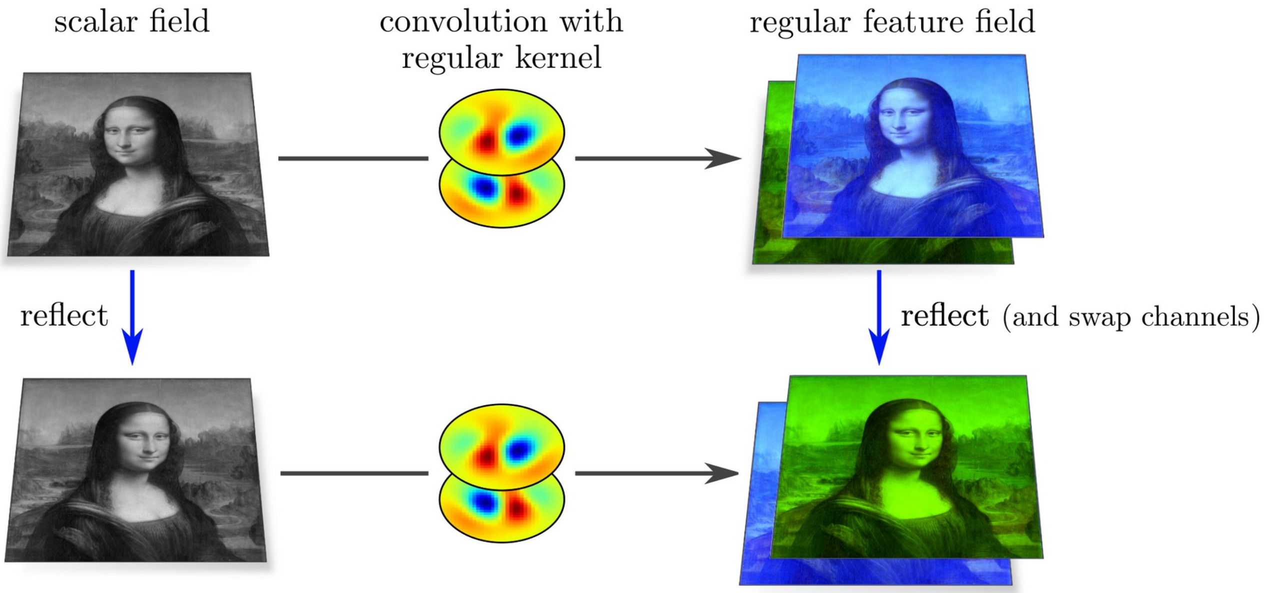
Reflection steerable kernels



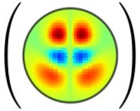
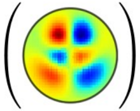
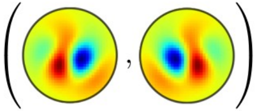
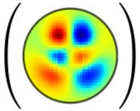
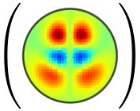
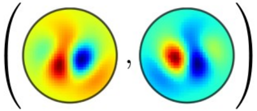
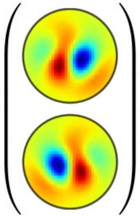
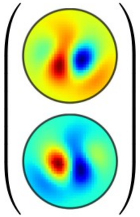
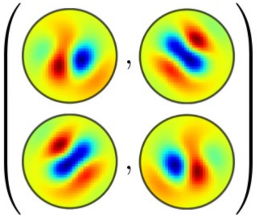
Reflection steerable kernels



Reflection steerable kernels



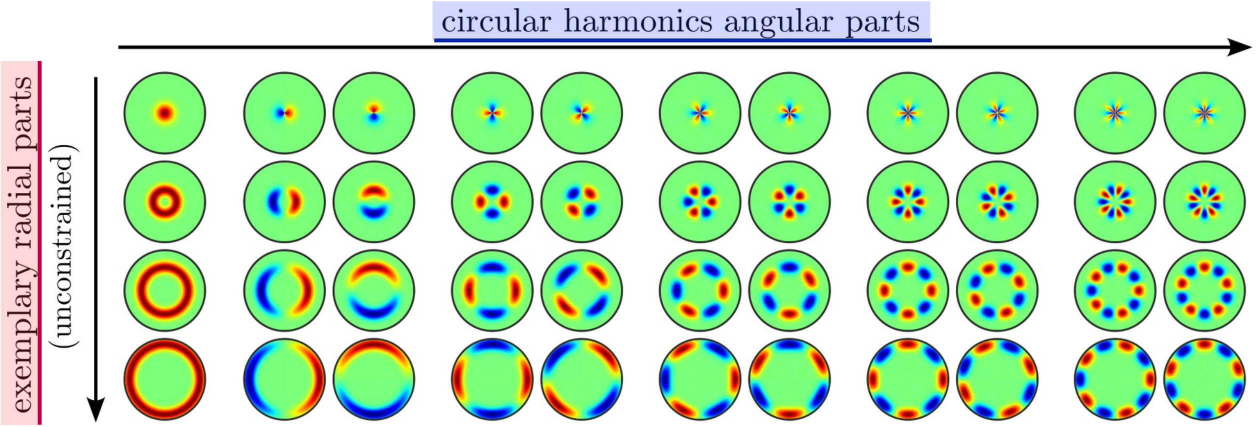
Reflection steerable kernels

ρ_{in} ρ_{out}	trivial	sign-flip	regular
trivial			
sign-flip			
regular			

Full derivation of these examples in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 5.2)

SO(2)-steerable kernels

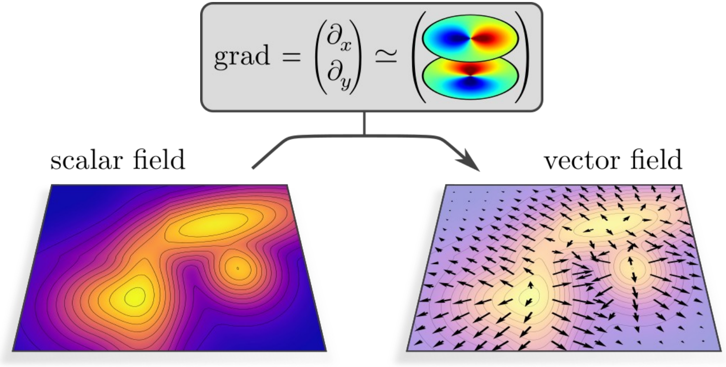
rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained



Full derivation for SO(2) in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 5.3.4)

SO(2)-steerable kernels

rotational symmetry constraint $\implies$ affects only *angular* part, *radial* part unconstrained

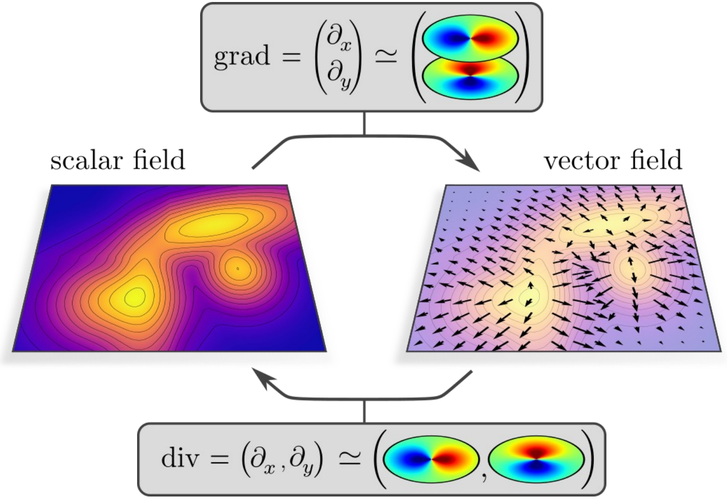


$\mathrm{grad} = \begin{pmatrix}\partial_x\\ \partial_y\end{pmatrix} \simeq$
scalar field
vector field

Full derivation for SO(2) in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 5.3.4)

SO(2)-steerable kernels

rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained

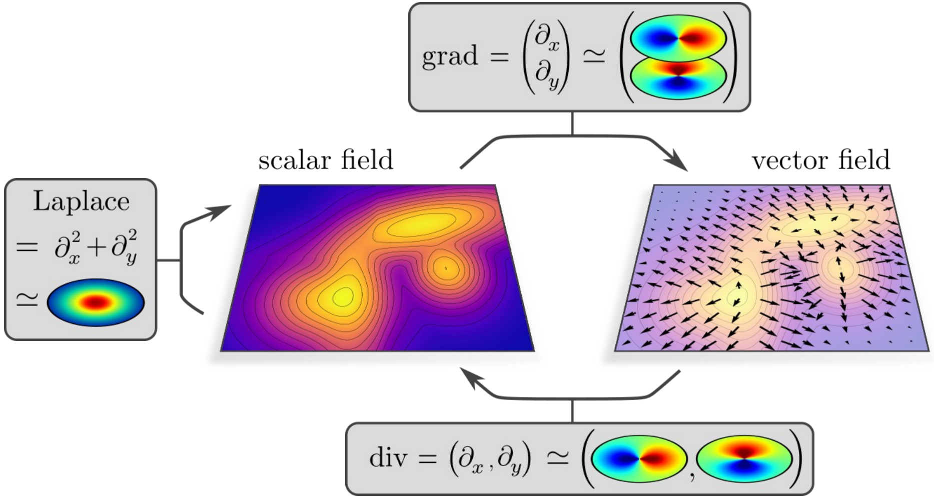

$\mathrm{grad} = \begin{pmatrix}\partial_x \\ \partial_y\end{pmatrix} \simeq$
scalar field
vector field
$\mathrm{div} = (\partial_x, \partial_y) \simeq$

Full derivation for SO(2) in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 5.3.4)

(this diagram is *not* commutative)

SO(2)-steerable kernels

rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained



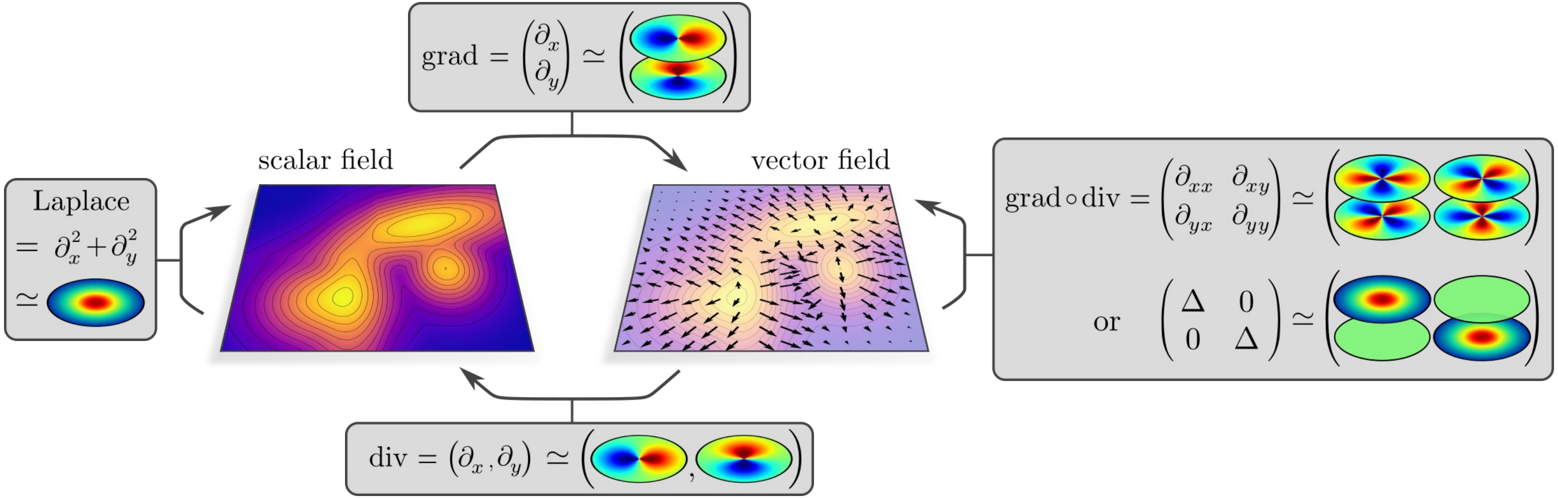
grad $= \begin{pmatrix}\partial_x\\ \partial_y\end{pmatrix} \simeq$
scalar field
vector field
Laplace
$= \partial_x^2 + \partial_y^2$
$\simeq$
div $= (\partial_x, \partial_y) \simeq$

Full derivation for SO(2) in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 5.3.4)

(this diagram is *not* commutative)

SO(2)-steerable kernels

rotational symmetry constraint $\implies$ affects only *angular* part, *radial* part unconstrained



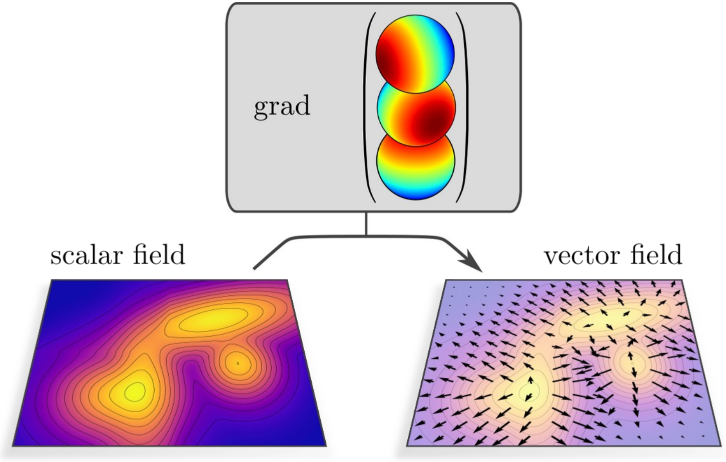
$\text{grad} = \begin{pmatrix} \partial_x \\ \partial_y \end{pmatrix} \simeq$
scalar field
vector field
Laplace
$= \partial_x^2 + \partial_y^2$
$\simeq$
$\text{grad} \circ \text{div} = \begin{pmatrix} \partial_{xx} & \partial_{xy} \\ \partial_{yx} & \partial_{yy} \end{pmatrix} \simeq$
or $\begin{pmatrix} \Delta & 0 \\ 0 & \Delta \end{pmatrix} \simeq$
$\text{div} = (\partial_x, \partial_y) \simeq$

Full derivation for SO(2) in (Weiler et al. 2023, *Equivariant and Coordinate Independent Convolutional Networks*, Section 5.3.4)

(this diagram is *not* commutative)

SO(3)-steerable kernels

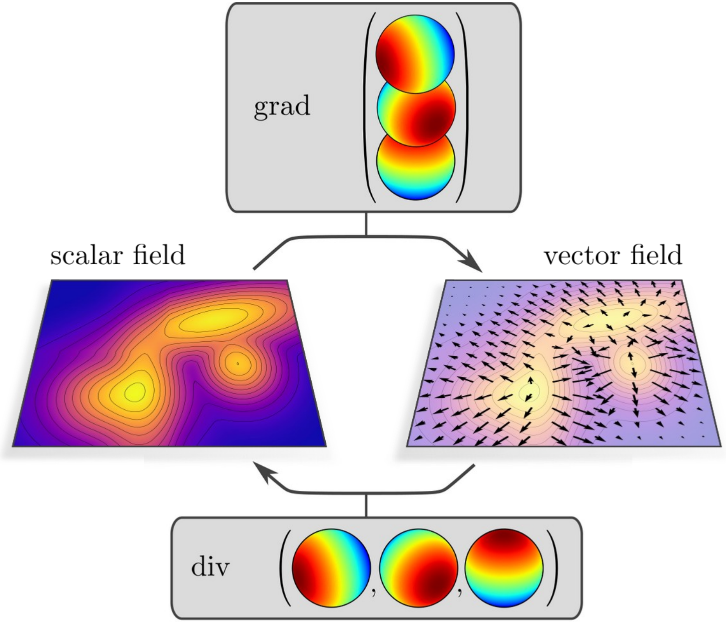
rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained



Full derivation for SO(3) in (Lang & Weiler 2021, *A Wigner-Eckart Theorem for Group Equivariant Convolution Kernels*, Appendix E.5)

SO(3)-steerable kernels

rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained

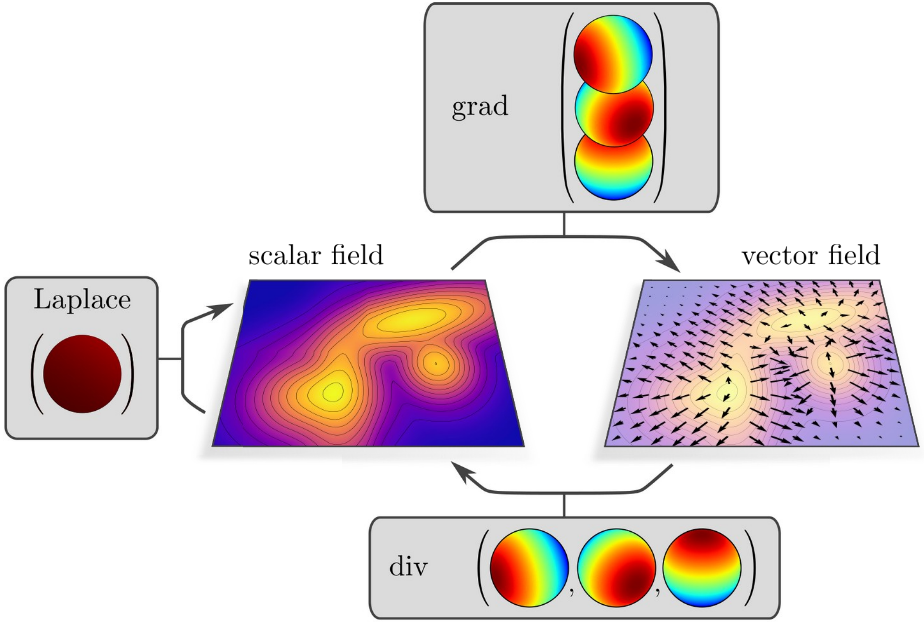


Full derivation for SO(3) in (Lang & Weiler 2021, *A Wigner-Eckart Theorem for Group Equivariant Convolution Kernels*, Appendix E.5)

(this diagram is *not* commutative)

SO(3)-steerable kernels

rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained

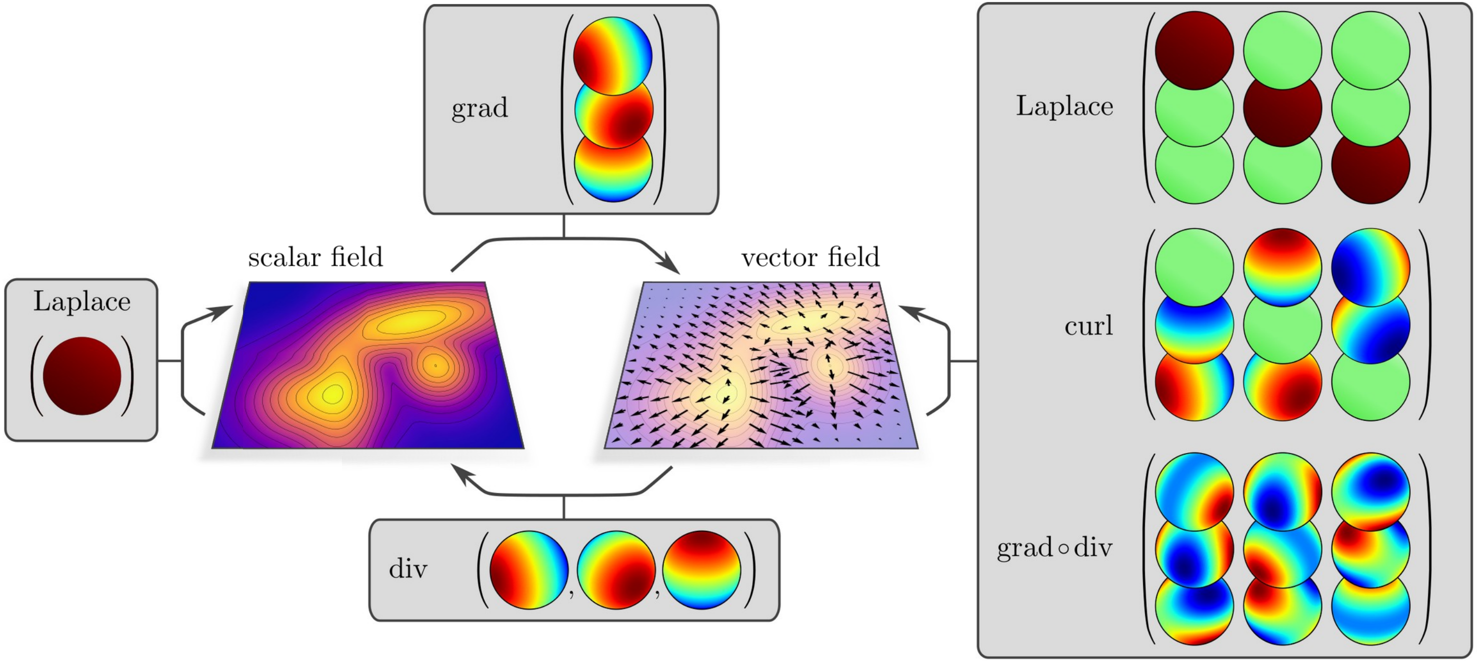


Full derivation for SO(3) in (Lang & Weiler 2021, *A Wigner-Eckart Theorem for Group Equivariant Convolution Kernels*, Appendix E.5)

(this diagram is *not* commutative)

SO(3)-steerable kernels

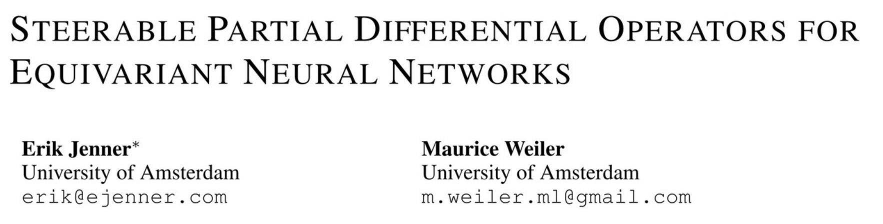
rotational symmetry constraint $\Longrightarrow$ affects only *angular* part, *radial* part unconstrained



Full derivation for SO(3) in (Lang & Weiler 2021, *A Wigner-Eckart Theorem for Group Equivariant Convolution Kernels*, Appendix E.5)

(this diagram is *not* commutative)

SO(3)-steerable kernels

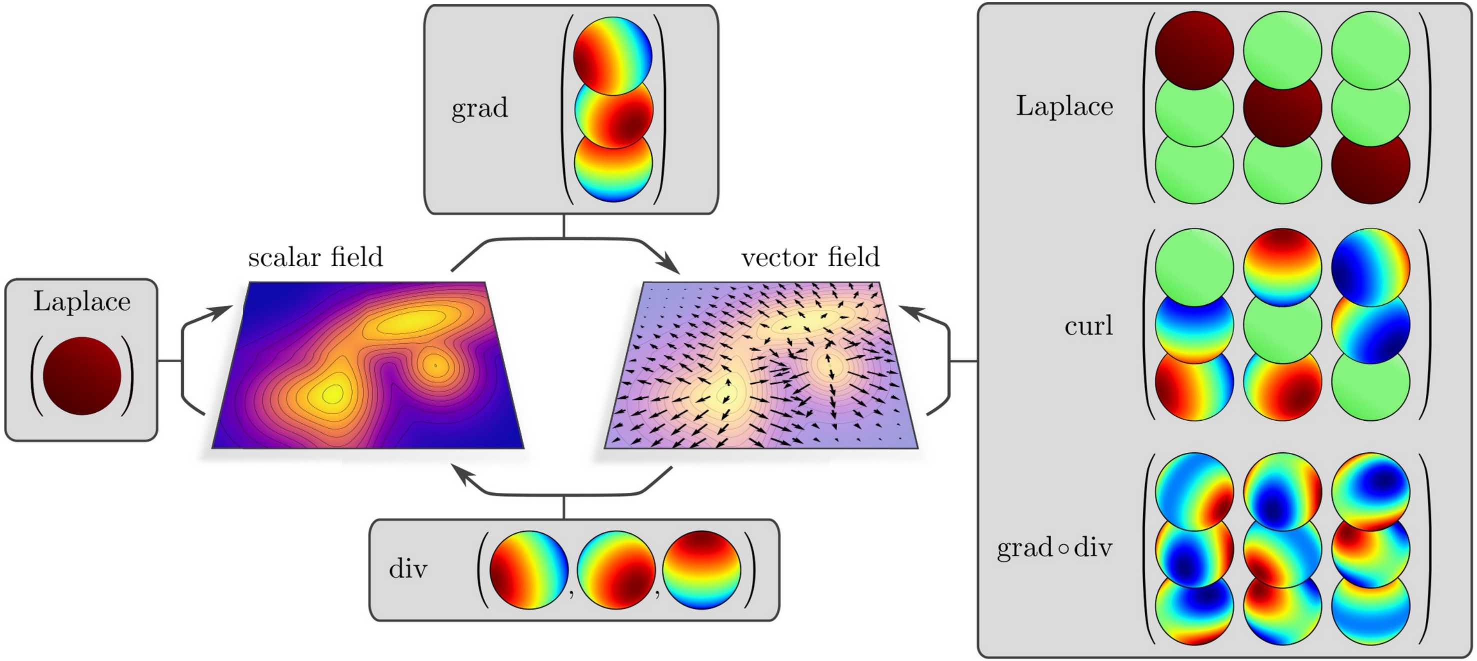


Steerable Partial Differential Operators for Equivariant Neural Networks

Erik Jenner[*]
University of Amsterdam
erik@ejenner.com

Maurice Weiler
University of Amsterdam
m.weiler.ml@gmail.com

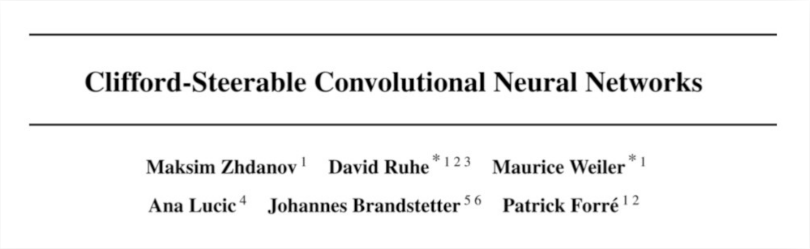
rotational symmetry constraint ⟹ affects only *angu*



Full derivation for SO(3) in (Lang & Weiler 2021, *A Wigner-Eckart Theorem for Group Equivariant Convolution Kernels*, Appendix E.5)

(this diagram is *not* commutative)

Lorentz group steerable kernels

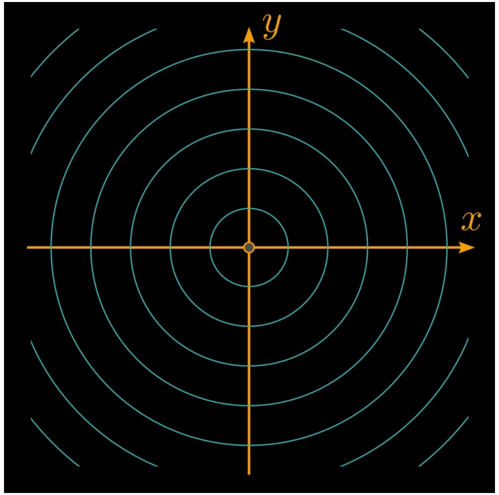


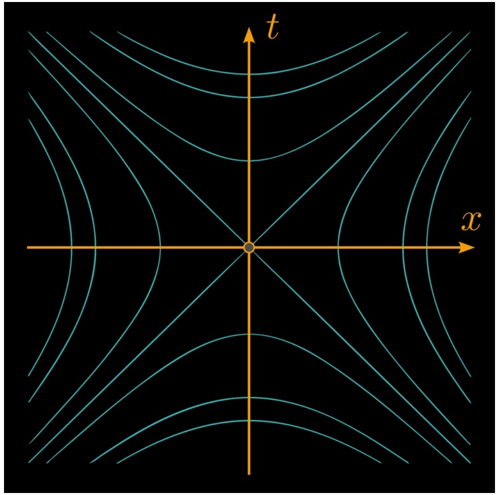
Clifford-Steerable Convolutional Neural Networks

Maksim Zhdanov[1] David Ruhe*[1,2,3] Maurice Weiler*[1]
Ana Lucic[4] Johannes Brandstetter[5,6] Patrick Forré[1,2]

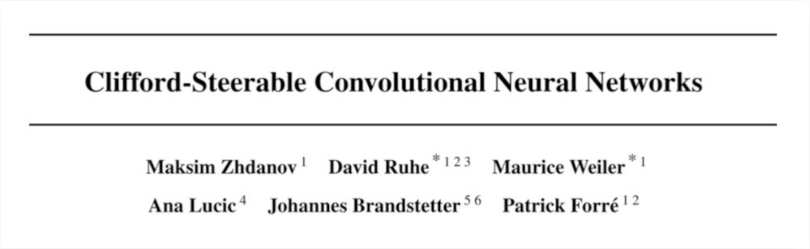
Euclidean space $\mathbb{R}^2$, metric $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Minkowski spacetime $\mathbb{R}^{1,1}$, metric $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$



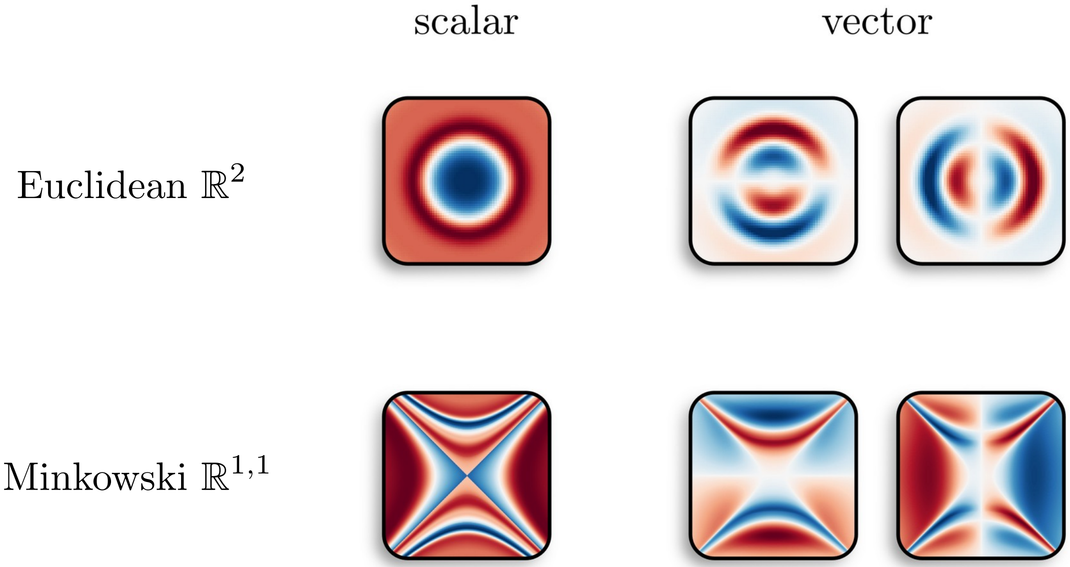


Lorentz group steerable kernels

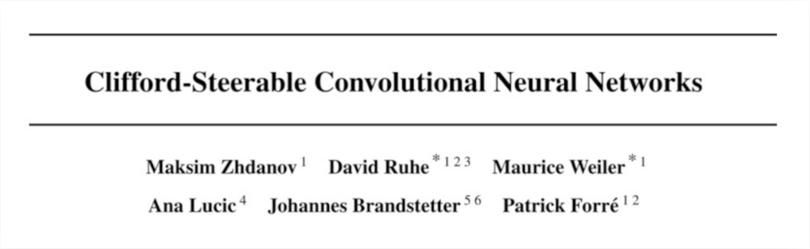


Clifford-Steerable Convolutional Neural Networks

Maksim Zhdanov[1] David Ruhe[*1 2 3] Maurice Weiler[*1]
Ana Lucic[4] Johannes Brandstetter[5 6] Patrick Forré[1 2]

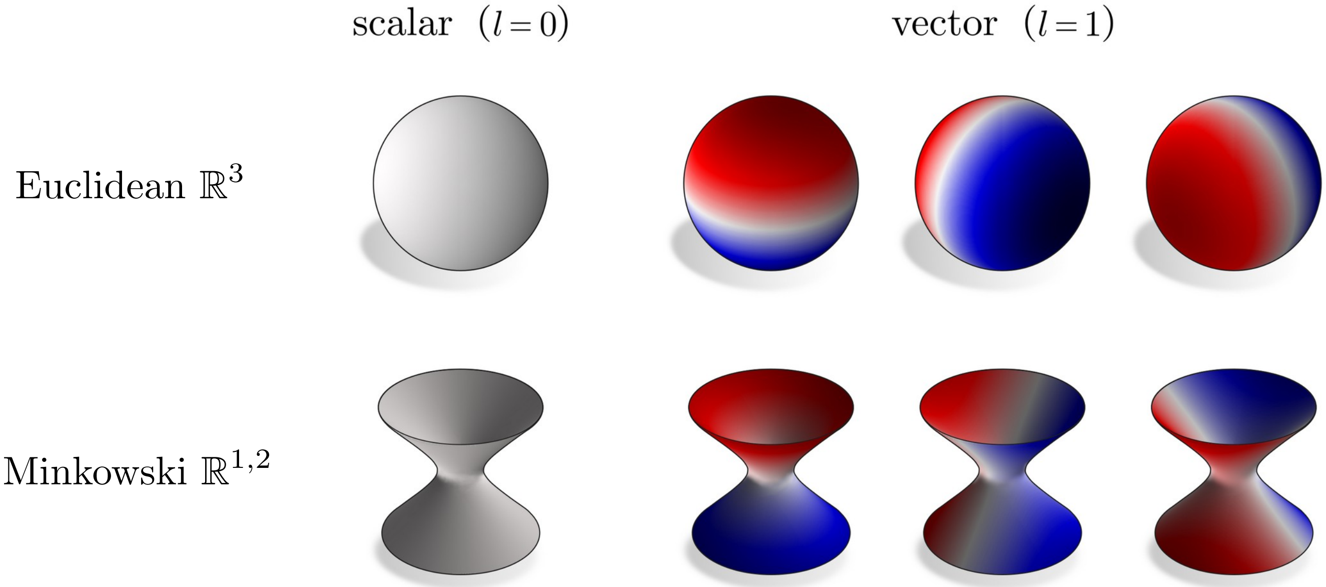


Lorentz group steerable kernels



Clifford-Steerable Convolutional Neural Networks

Maksim Zhdanov[1] David Ruhe[*1 2 3] Maurice Weiler[*1]
Ana Lucic[4] Johannes Brandstetter[5 6] Patrick Forré[1 2]



Quantum Mechanics

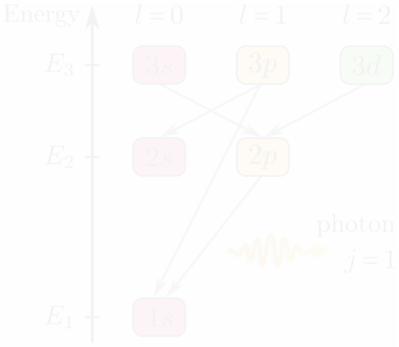
operators $\widehat{A}$ acting on states $|\psi\rangle$

symmetries $\Rightarrow$ representation operator constraint

$$\sum\nolimits_j g_{ij}\widehat{A}_j \;=\; \widehat{U}(g)^\dagger\,\widehat{A}_i\,\widehat{U}(g)$$

(e.g. vector operator)

selection rules for quantum state transitions



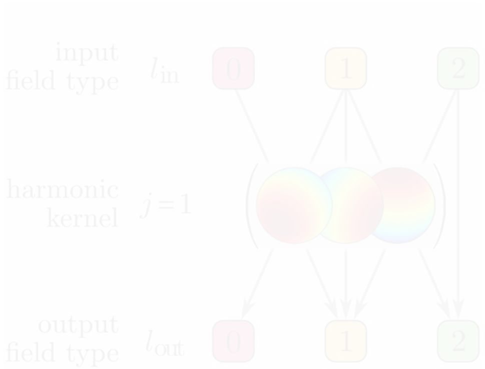
Deep Learning

kernels K acting on features $f(x)$

symmetries $\Rightarrow$ steerable kernel constraint

$$K\big(g^{-1}x\big) \;=\; \rho_{\mathrm{out}}(g)^\dagger\,K(x)\,\rho_{\mathrm{in}}(g)$$

selection rules for equivariant message passing



Quantum Mechanics

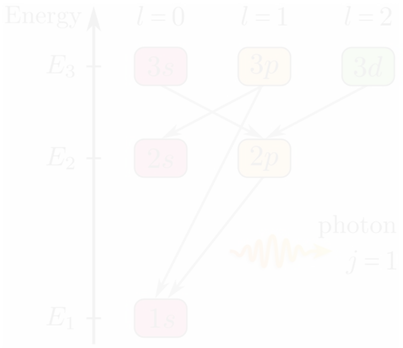
operators $\widehat{A}$ acting on states $|\psi\rangle$

symmetries ⇒ representation operator constraint

$$\sum\nolimits_j g_{ij} \widehat{A}_j \;=\; \widehat{U}(g)^\dagger\, \widehat{A}_i\, \widehat{U}(g)$$

(e.g. vector operator)

selection rules for quantum state transitions



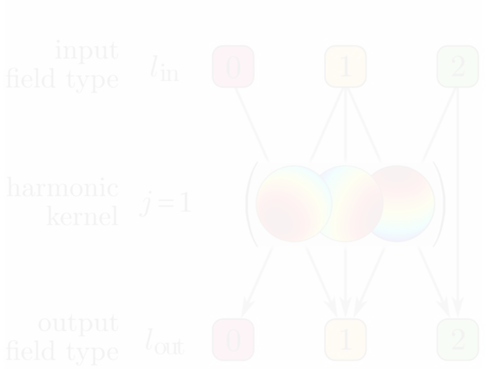
Deep Learning

kernels K acting on features $f(x)$

symmetries ⇒ steerable kernel constraint

$$K\big(g^{-1}x\big) \;=\; \rho_{\mathrm{out}}(g)^\dagger\, K(x)\, \rho_{\mathrm{in}}(g)$$

selection rules for equivariant message passing



Quantum Mechanics

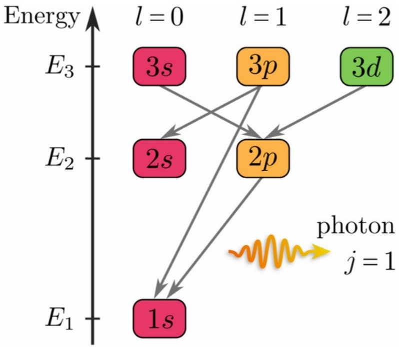
operators $\widehat{A}$ acting on states $|\psi\rangle$

symmetries $\Rightarrow$ representation operator constraint

$$\sum_j g_{ij}\widehat{A}_j \;=\; \widehat{U}(g)^\dagger\,\widehat{A}_i\,\widehat{U}(g)$$

(e.g. vector operator)

selection rules for quantum state transitions



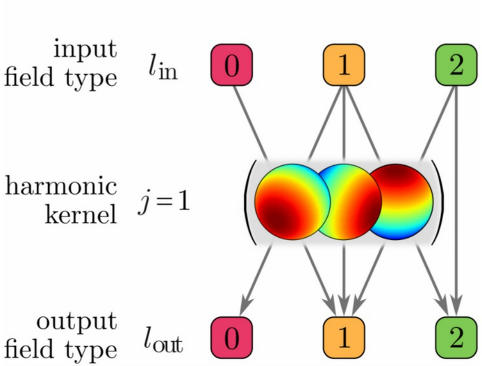
Deep Learning

kernels K acting on features $f(x)$

symmetries $\Rightarrow$ steerable kernel constraint

$$K\big(g^{-1}x\big) \;=\; \rho_{\text{out}}(g)^\dagger\, K(x)\, \rho_{\text{in}}(g)$$

selection rules for equivariant message passing



Quantum Mechanics

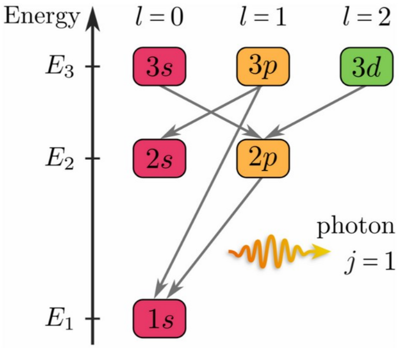
operators $\widehat{A}$ acting on states $|\psi\rangle$

symmetries $\Rightarrow$ representation operator constraint

$$\sum_j g_{ij}\widehat{A}_j \;=\; \widehat{U}(g)^\dagger\, \widehat{A}_i\, \widehat{U}(g)$$

(e.g. vector operator)

selection rules for quantum state transitions



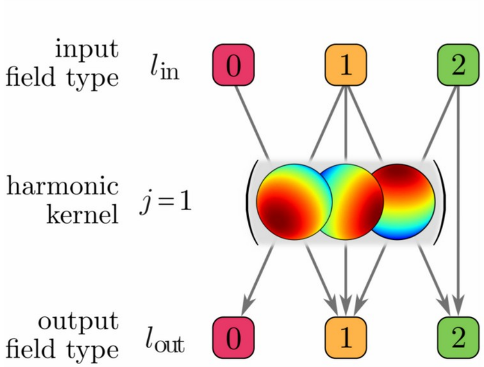
Deep Learning

kernels K acting on features $f(x)$

symmetries $\Rightarrow$ steerable kernel constraint

$$K\big(g^{-1}x\big) \;=\; \rho_{\mathrm{out}}(g)^\dagger\, K(x)\, \rho_{\mathrm{in}}(g)$$

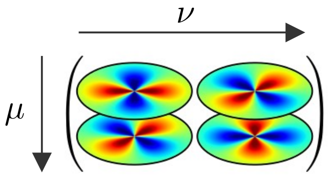
selection rules for equivariant message passing



non-zero
matrix elements!

Wigner–Eckart theorem

operators / kernels are fully described by their *matrix elements*: $\langle\mu|\widehat{A}|\nu\rangle$ or $K_{\mu\nu}(x)$



symmetries couple matrix elements $\Rightarrow$ reduced *degrees of freedom / parameters*

Wigner-Eckart theorem (G=SO(3) / spherical tensor operators) :

$$\langle JM|\,\widehat{A}^{(j)}_{m}\,|ln\rangle \;=\; \underbrace{\lambda^{(Jlj)}}_{\text{“reduced matrix element”}}\,\overbrace{\langle JM\,|\,jm;ln\rangle}^{\text{Clebsch-Gordan coeffs. (fixed)}}$$

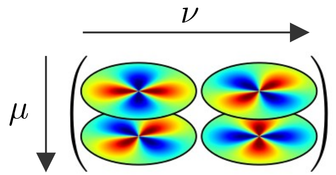
(single d.o.f. instead of $(2J{+}1)(2l{+}1)(2j{+}1)$)

generalized Wigner-Eckart theorem for G-steerable kernels :

$$K^{(J\leftarrow l)}_{Mn}(x) \;:=\; \underbrace{\langle JM|K(x)|ln\rangle}_{\substack{\textit{kernel}\\ \textit{matrix elements}}} = \sum_{j\in\widehat{G}}\sum_{i=1}^{m_j}\sum_{s=1}^{[J(jl)]}\sum_{m=1}^{d_j}\sum_{M'=1}^{d_J} \underbrace{\langle JM|c_{jis}|JM'\rangle}_{\substack{\textit{irrep}\\ \textit{endomorphisms}}}\cdot\underbrace{\langle s,JM'|jm;ln\rangle}_{\substack{\textit{Clebsch-Gordan}\\ \textit{coefficients}}}\cdot\underbrace{\langle i,jm|x\rangle}_{\substack{\textit{harmonics}\\ \textit{(Peter Weyl)}}}$$

Wigner–Eckart theorem

operators / kernels are fully described by their *matrix elements*: $\langle\mu|\widehat{A}|\nu\rangle$ or $K_{\mu\nu}(x)$



symmetries couple matrix elements $\Rightarrow$ reduced *degrees of freedom / parameters*

Wigner-Eckart theorem (G=SO(3) / spherical tensor operators) :

$$\big\langle JM\big|\,\widehat{A}^{(j)}_m\,\big|ln\big\rangle \;=\; \underbrace{\lambda^{(Jlj)}}_{\text{“reduced matrix element”}}\,\overbrace{\big\langle JM\,\big|\,jm;ln\big\rangle}^{\text{Clebsch-Gordan coeffs. (fixed)}}$$

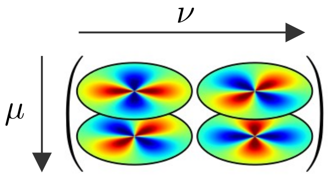
(single d.o.f. instead of $(2J{+}1)(2l{+}1)(2j{+}1)$)

generalized Wigner-Eckart theorem for G-steerable kernels :

$$K^{(J\leftarrow l)}_{Mn}(x) \;:=\; \underbrace{\langle JM|K(x)|ln\rangle}_{\textit{kernel matrix elements}} \;=\; \sum_{j\in\widehat{G}}\sum_{i=1}^{m_j}\sum_{s=1}^{[J(jl)]}\sum_{m=1}^{d_j}\sum_{M'=1}^{d_J}\underbrace{\big\langle JM\big|c_{jis}\big|JM'\big\rangle}_{\textit{irrep endomorphisms}}\cdot\underbrace{\big\langle s,JM'\big|jm;ln\big\rangle}_{\textit{Clebsch-Gordan coefficients}}\cdot\underbrace{\big\langle i,jm\big|x\big\rangle}_{\textit{harmonics (Peter Weyl)}}$$

Wigner–Eckart theorem

operators / kernels are fully described by their *matrix elements*: $\langle\mu|\widehat{A}|\nu\rangle$ or $K_{\mu\nu}(x)$



symmetries couple matrix elements $\Rightarrow$ reduced *degrees of freedom / parameters*

Wigner-Eckart theorem (G=SO(3) / spherical tensor operators) :

$$\big\langle JM \big|\, \widehat{A}^{(j)}_m \,\big| ln \big\rangle \;=\; \lambda^{(Jlj)} \big\langle JM \,\big|\, jm; ln \big\rangle$$

Clebsch-Gordan coeffs. (fixed)

"reduced matrix element"

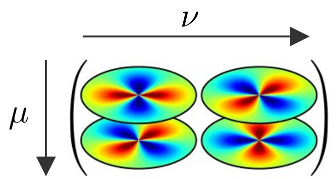
(single d.o.f. instead of $(2J+1)(2l+1)(2j+1)$)

generalized Wigner-Eckart theorem for G-steerable kernels :

$$K^{(J\leftarrow l)}_{Mn}(x) \;:=\; \underbrace{\langle JM|K(x)|ln\rangle}_{\substack{\textit{kernel}\\ \textit{matrix elements}}} \;=\; \sum_{j\in\widehat{G}} \sum_{i=1}^{m_j} \sum_{s=1}^{[J(jl)]} \sum_{m=1}^{d_j} \sum_{M'=1}^{d_J} \underbrace{\big\langle JM\big|c_{jis}\big|JM'\big\rangle}_{\substack{\textit{irrep}\\ \textit{endomorphisms}}} \cdot \underbrace{\big\langle s, JM'\big|jm; ln\big\rangle}_{\substack{\textit{Clebsch-Gordan}\\ \textit{coefficients}}} \cdot \underbrace{\big\langle i, jm\big|x\big\rangle}_{\substack{\textit{harmonics}\\ \textit{(Peter Weyl)}}}$$

Wigner–Eckart theorem

operators / kernels are fully described by their *matrix elements*: $\langle\mu|\widehat{A}|\nu\rangle$ or $K_{\mu\nu}(x)$



symmetries couple matrix elements $\Rightarrow$ reduced *degrees of freedom / parameters*

Clebsch-Gordan coeffs. (fixed)

Wigner-Eckart theorem (G=SO(3) / spherical tensor operators) : $\big\langle JM\big|\,\widehat{A}^{(j)}_m\,\big|ln\big\rangle \;=\; \lambda^{(Jlj)}\,\big\langle JM\,\big|\,jm;ln\big\rangle$

"reduced matrix element"

(single d.o.f. instead of $(2J{+}1)(2l{+}1)(2j{+}1)$)

generalized Wigner-Eckart theorem for G-steerable kernels : $\longleftarrow$ assuming *compact* $G \leq \mathrm{U}(d)$

$$K_{Mn}^{(J\leftarrow l)}(x) \;:=\; \underbrace{\langle JM|K(x)|ln\rangle}_{\textit{kernel matrix elements}} \;=\; \sum_{j\in\widehat{G}}\sum_{i=1}^{m_j}\sum_{s=1}^{[J(jl)]}\sum_{m=1}^{d_j}\sum_{M'=1}^{d_J} \underbrace{\big\langle JM\big|c_{jis}\big|JM'\big\rangle}_{\textit{irrep endomorphisms}}\cdot\underbrace{\big\langle s,JM'\big|jm;ln\big\rangle}_{\textit{Clebsch-Gordan coefficients}}\cdot\underbrace{\big\langle i,jm\big|x\big\rangle}_{\textit{harmonics (Peter Weyl)}}$$

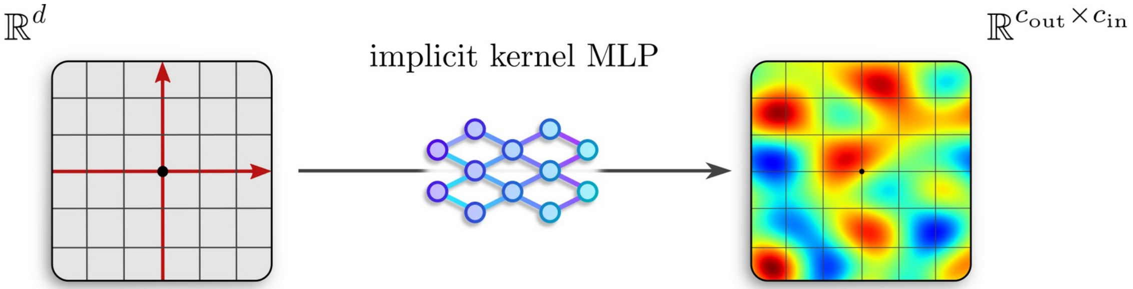
Implicit steerable kernels

convolution kernels are functions $K: \; \mathbb{R}^d \longrightarrow \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$

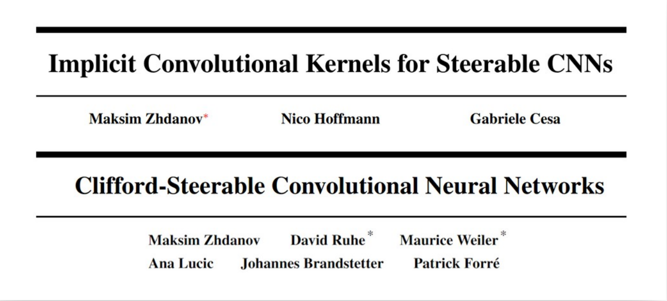
they can be implemented via MLPs

G-steerable kernels are G-equivariant functions

they can be implemented via G-equivariant MLPs



Implicit steerable kernels



Implicit Convolutional Kernels for Steerable CNNs

Maksim Zhdanov* Nico Hoffmann Gabriele Cesa

Clifford-Steerable Convolutional Neural Networks

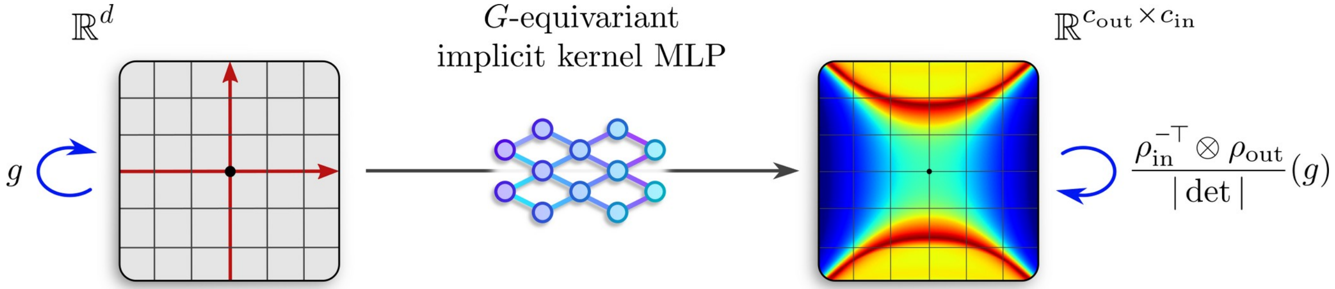
Maksim Zhdanov David Ruhe* Maurice Weiler*
Ana Lucic Johannes Brandstetter Patrick Forré

convolution kernels are functions $K:\ \mathbb{R}^d \longrightarrow \mathbb{R}^{c_\text{out}\times c_\text{in}}$

they can be implemented via MLPs

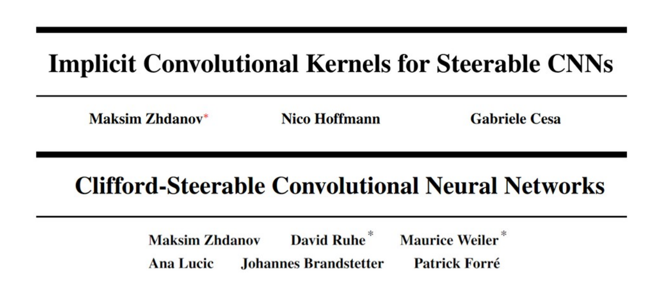
G-steerable kernels are G-equivariant functions

they can be implemented via G-equivariant MLPs



$\mathbb{R}^d$
g
G-equivariant
implicit kernel MLP
$\mathbb{R}^{c_\text{out}\times c_\text{in}}$
$\dfrac{\rho_\text{in}^{-\top}\otimes\rho_\text{out}}{|\det|}(g)$

Implicit steerable kernels



Implicit Convolutional Kernels for Steerable CNNs

Maksim Zhdanov* Nico Hoffmann Gabriele Cesa

Clifford-Steerable Convolutional Neural Networks

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

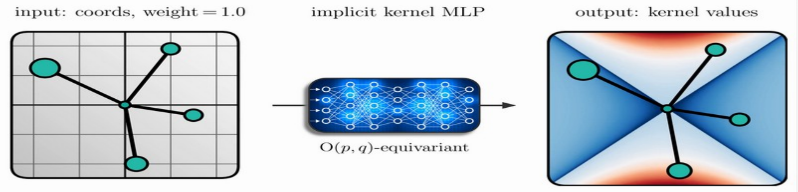
convolution kernels are functions $K: \ \mathbb{R}^d \longrightarrow \mathbb{R}^{c_\text{out} \times c_\text{in}}$

they can be implemented via MLPs

G-steerable kernels are G-equivariant functions

they can be implemented via G-equivariant MLPs

advantage: can additionally be made
input feature dependent

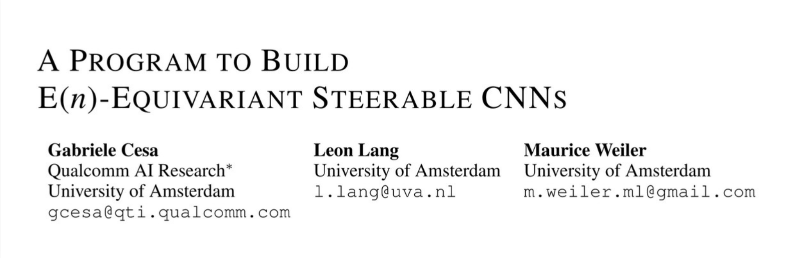


escnn PyTorch library

equivariant CNNs / MLPs for ...

... any groups G ≤ O(d)   (d=1,2,3)

... arbitrary field types ρ



A PROGRAM TO BUILD
E(n)-EQUIVARIANT STEERABLE CNNS

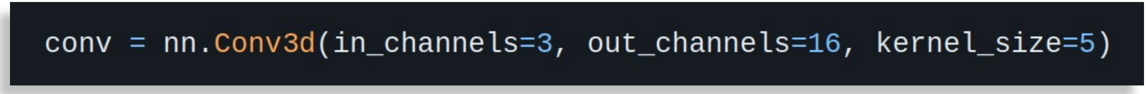
Gabriele Cesa
Qualcomm AI Research*
University of Amsterdam
gcesa@qti.qualcomm.com

Leon Lang
University of Amsterdam
l.lang@uva.nl

Maurice Weiler
University of Amsterdam
m.weiler.ml@gmail.com

https://github.com/QUVA-Lab/e2cnn
https://github.com/QUVA-Lab/escnn

native PyTorch:



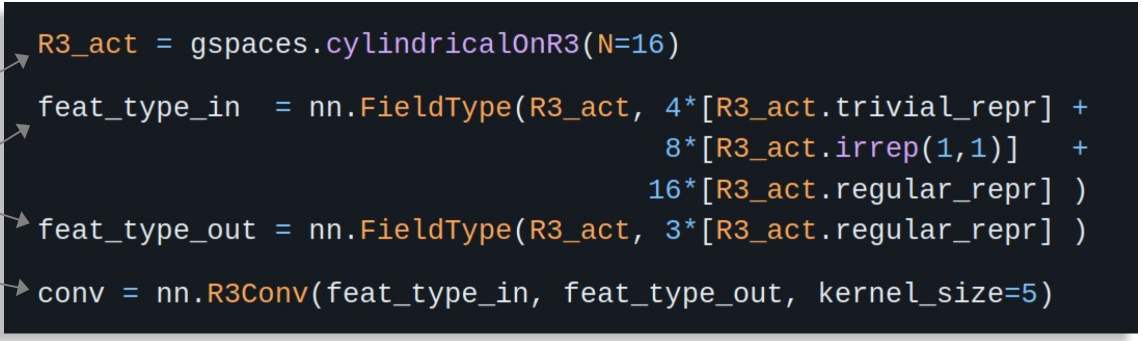
```
conv = nn.Conv3d(in_channels=3, out_channels=16, kernel_size=5)
```

escnn:

fix G and G-action on $\mathbb{R}^d$

specify field types

construct $\mathrm{Aff}(G)$-convolution

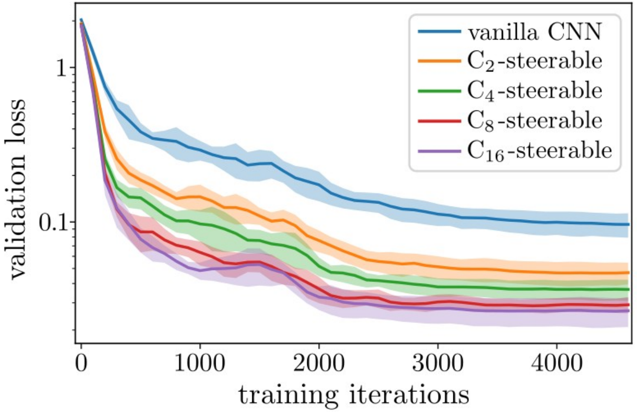


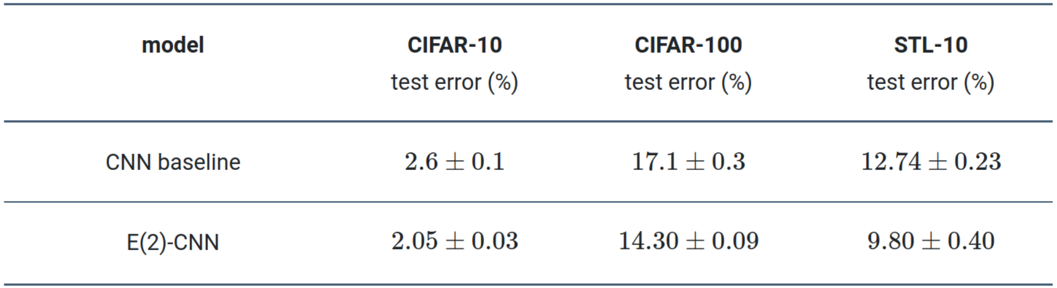
```
R3_act = gspaces.cylindricalOnR3(N=16)

feat_type_in  = nn.FieldType(R3_act, 4*[R3_act.trivial_repr] +
                                     8*[R3_act.irrep(1,1)]   +
                                    16*[R3_act.regular_repr] )
feat_type_out = nn.FieldType(R3_act, 3*[R3_act.regular_repr] )

conv = nn.R3Conv(feat_type_in, feat_type_out, kernel_size=5)
```

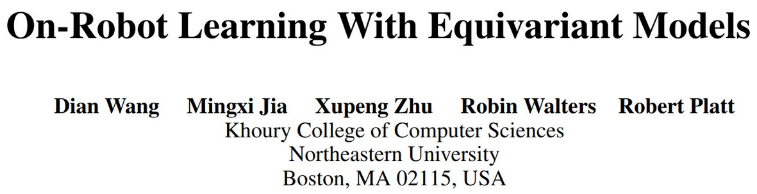
Emperical results - image classification





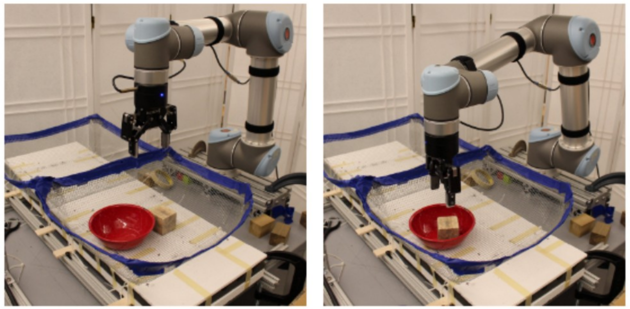
model	CIFAR-10 test error (%)	CIFAR-100 test error (%)	STL-10 test error (%)
CNN baseline	2.6 ± 0.1	17.1 ± 0.3	12.74 ± 0.23
E(2)-CNN	2.05 ± 0.03	14.30 ± 0.09	9.80 ± 0.40

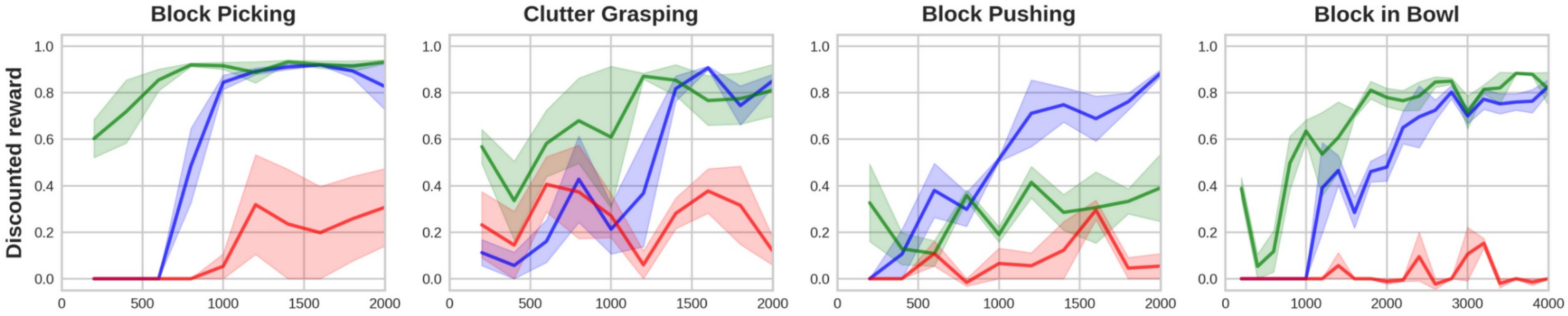
Emperical results - reinforcement learning



On-Robot Learning With Equivariant Models

Dian Wang Mingxi Jia Xupeng Zhu Robin Walters Robert Platt
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02115, USA





🟩 equivariant
(simulation pre-trained)

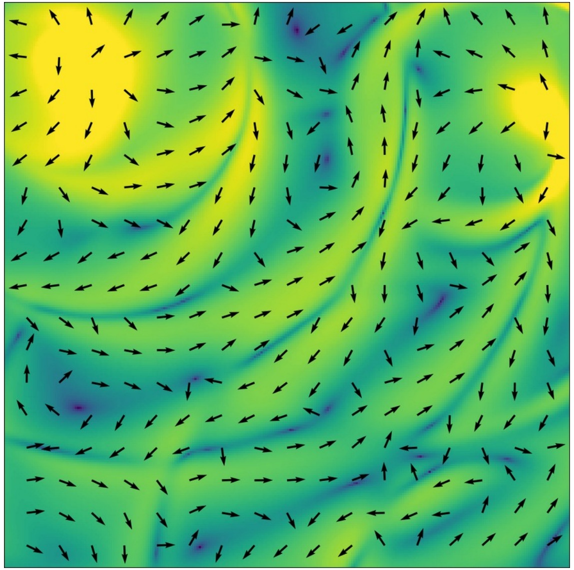
🟦 equivariant
(on-robot)

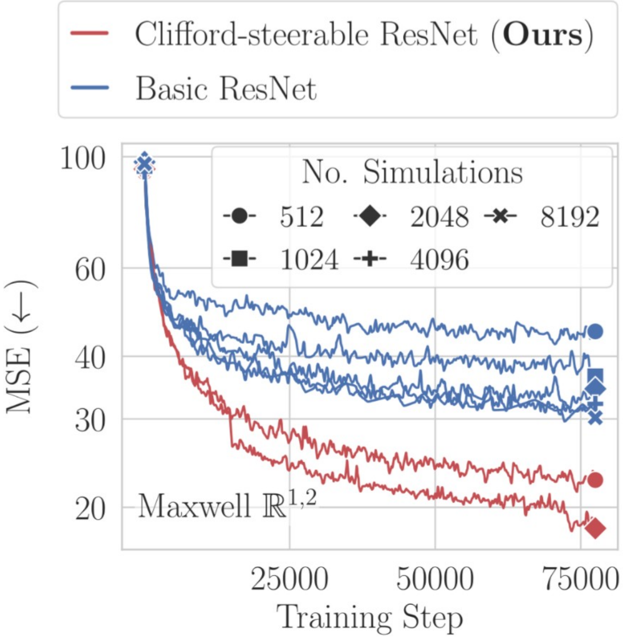
🟥 non-equivariant
(on-robot)

Emperical results - electrodynamics (relativistic)

EM field, induced by moving source charges

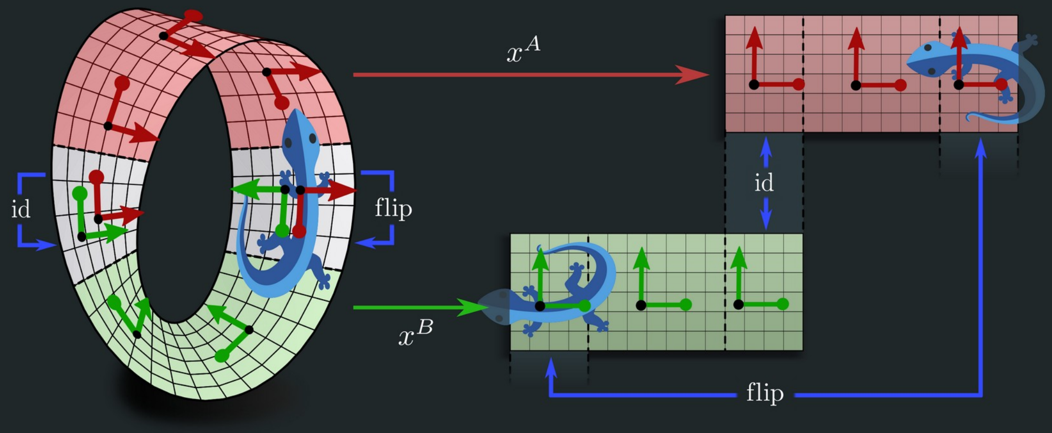
simulate next time steps given previous time steps





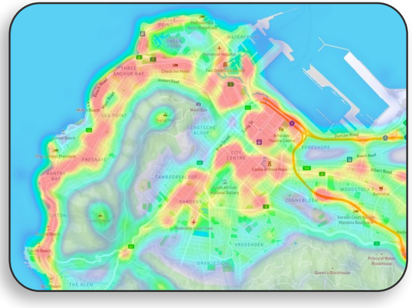
>2x lower MSE

Convolutions on homogeneous spaces & manifolds

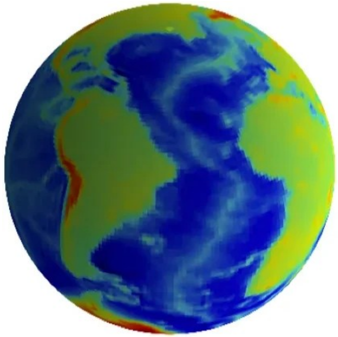


Feature fields on non-Euclidean spaces

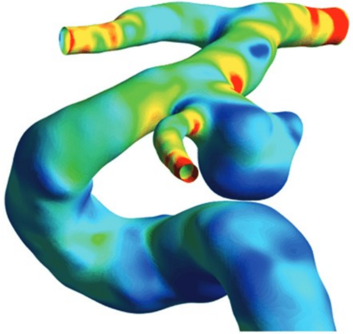
Euclidean



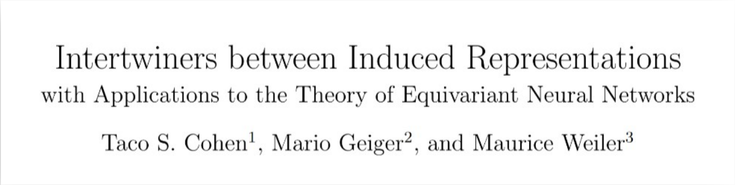
homogeneous



general surface



Homogeneous spaces



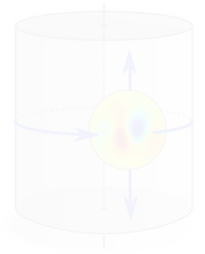
Intertwiners between Induced Representations
with Applications to the Theory of Equivariant Neural Networks

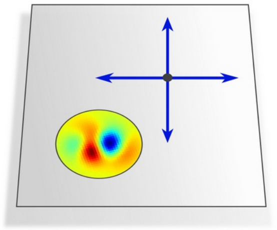
Taco S. Cohen[1], Mario Geiger[2], and Maurice Weiler[3]

idea: equivariance $\Longrightarrow$ weight sharing (convolution)

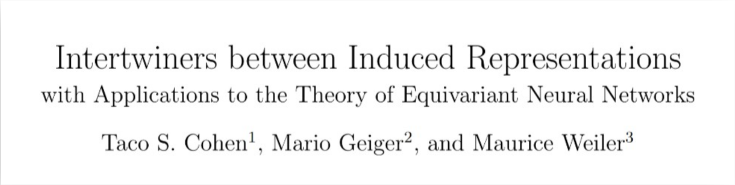
this works for any *homogeneous space* (space with transitive group action)

kernels need to be steerable w.r.t. *stabilizer subgroup*





space	cylinder	$\mathbb{R}^d$	$\mathbb{R}^d$	S^2
symmetry	$(\mathbb{R},+)\times\mathrm{SO}(2)$	$(\mathbb{R}^d,+)$	$\mathrm{Aff}(G)$	$\mathrm{SO}(3)$
stabilizer subgroup	$\{e\}$	$\{e\}$	G	$\mathrm{SO}(2)$



Intertwiners between Induced Representations
with Applications to the Theory of Equivariant Neural Networks

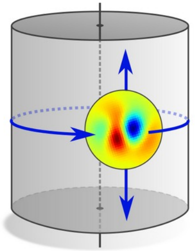
Taco S. Cohen[1], Mario Geiger[2], and Maurice Weiler[3]

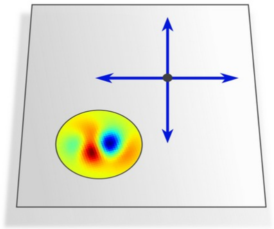
Homogeneous spaces

idea: equivariance $\Longrightarrow$ weight sharing (convolution)

this works for any *homogeneous space* (space with transitive group action)

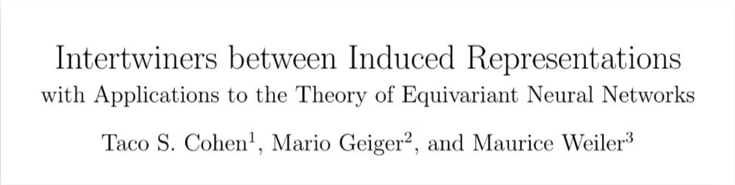
kernels need to be steerable w.r.t. *stabilizer subgroup*





space	cylinder	$\mathbb{R}^d$	$\mathbb{R}^d$	S^2
symmetry	$(\mathbb{R},+)\times\mathrm{SO}(2)$	$(\mathbb{R}^d,+)$	$\mathrm{Aff}(G)$	$\mathrm{SO}(3)$
stabilizer subgroup	$\{e\}$	$\{e\}$	G	$\mathrm{SO}(2)$

Homogeneous spaces



Intertwiners between Induced Representations

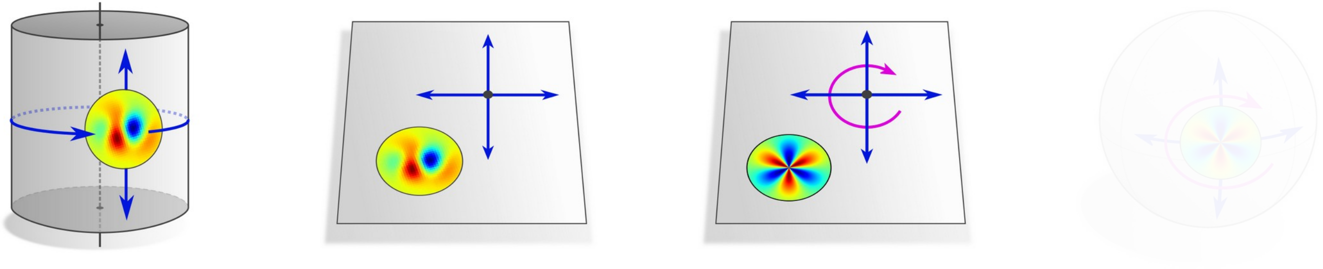
with Applications to the Theory of Equivariant Neural Networks

Taco S. Cohen[1], Mario Geiger[2], and Maurice Weiler[3]

idea: equivariance $\implies$ weight sharing (convolution)

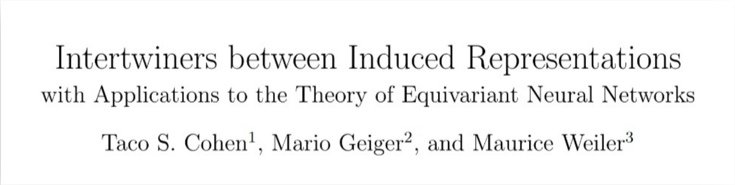
this works for any *homogeneous space* (space with transitive group action)

kernels need to be steerable w.r.t. *stabilizer subgroup*



space	cylinder	$\mathbb{R}^d$	$\mathbb{R}^d$	S^2
symmetry	$(\mathbb{R},+)\times\mathrm{SO}(2)$	$(\mathbb{R}^d,+)$	$\mathrm{Aff}(G)$	$\mathrm{SO}(3)$
stabilizer subgroup	$\{e\}$	$\{e\}$	G	$\mathrm{SO}(2)$

Homogeneous spaces



Intertwiners between Induced Representations

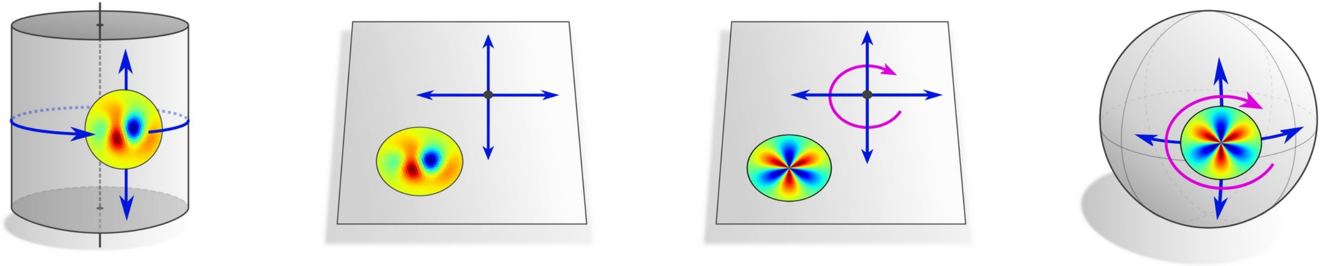
with Applications to the Theory of Equivariant Neural Networks

Taco S. Cohen[1], Mario Geiger[2], and Maurice Weiler[3]

idea: equivariance $\Longrightarrow$ weight sharing (convolution)

this works for any *homogeneous space* (space with transitive group action)

kernels need to be steerable w.r.t. *stabilizer subgroup*



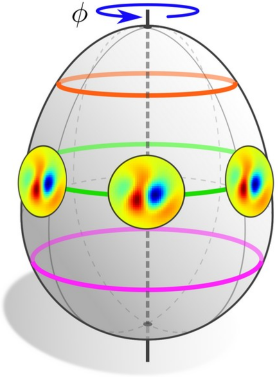
space	cylinder	$\mathbb{R}^d$	$\mathbb{R}^d$	S^2
symmetry	$(\mathbb{R},+)\times\mathrm{SO}(2)$	$(\mathbb{R}^d,+)$	$\mathrm{Aff}(G)$	$\mathrm{SO}(3)$
stabilizer subgroup	$\{e\}$	$\{e\}$	G	$\mathrm{SO}(2)$

Riemannian manifolds - convolutions via isometry equivariance?

idea: equivariance $\Longrightarrow$ weight sharing (convolution)

Riemannian manifolds are in general *asymmetric* (no transitive actions)

$\Longrightarrow$ weight sharing only over *symmetry orbits*



SO(2) orbits

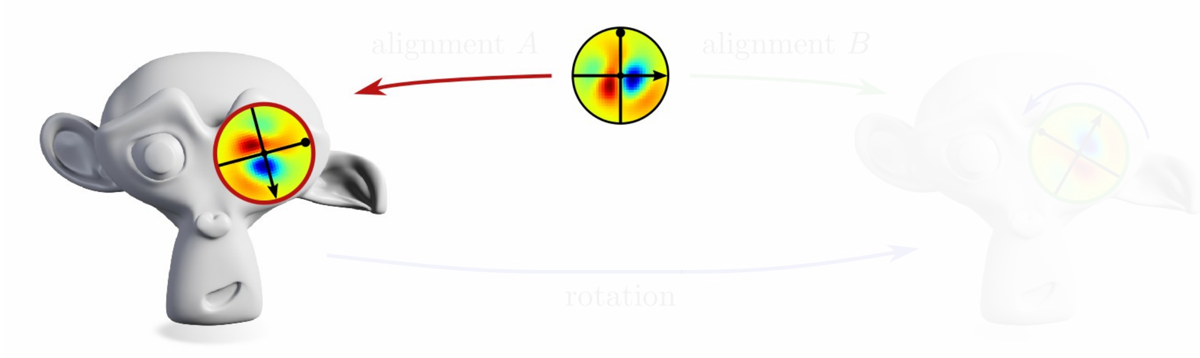
trivial orbits

Riemannian manifolds - convolutions via spatial weight sharing?

idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments ⟷ ambiguity of reference frames

solution: G-steerable kernels

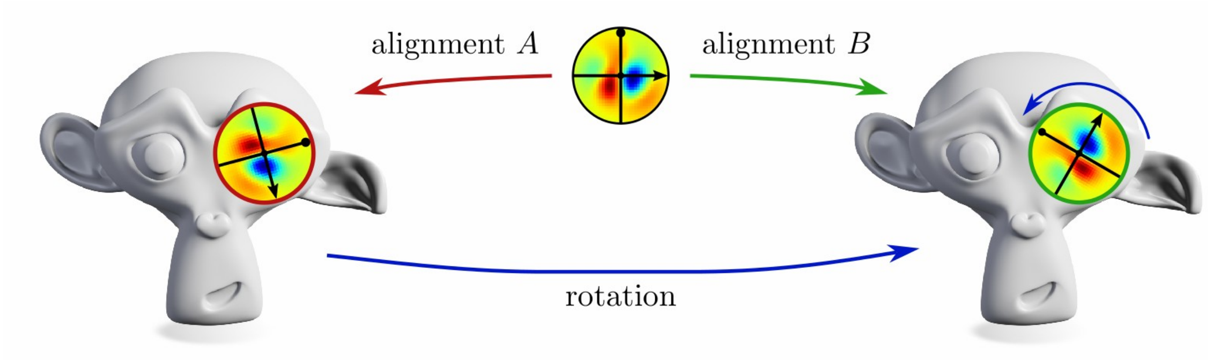


Riemannian manifolds - convolutions via spatial weight sharing?

idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments ⟷ ambiguity of reference frames

solution: G-steerable kernels

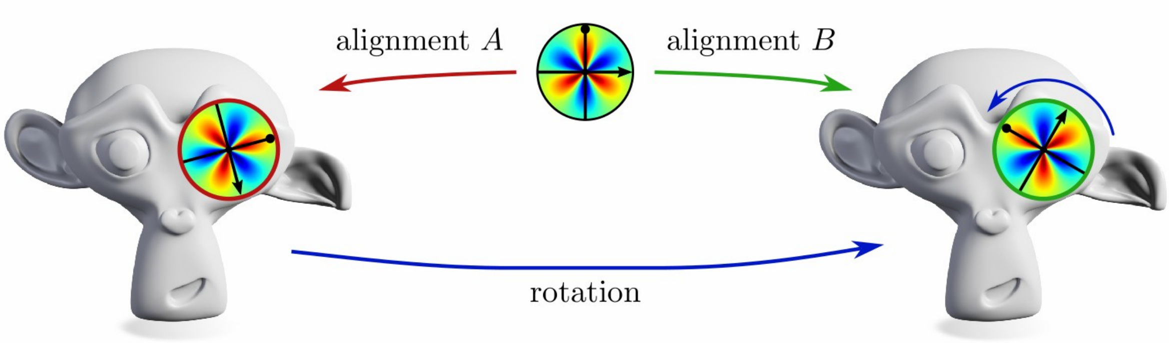


Riemannian manifolds - convolutions via spatial weight sharing?

idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments ⟷ ambiguity of reference frames

solution: G-steerable kernels

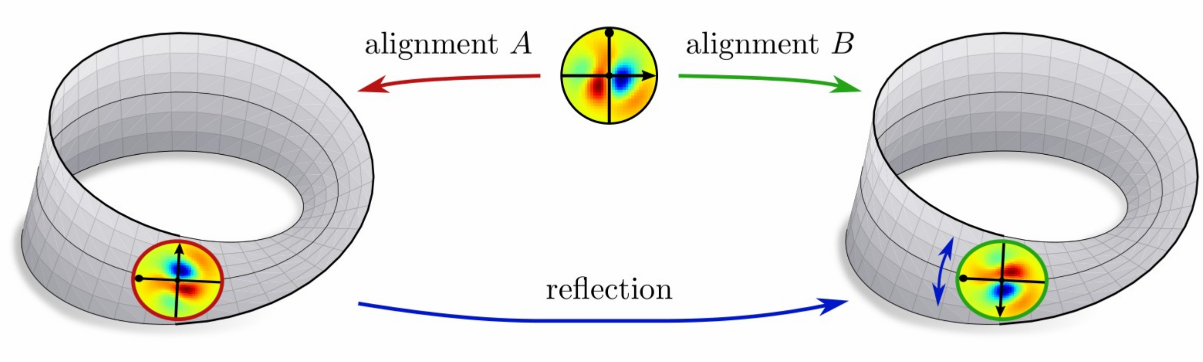


Riemannian manifolds - convolutions via spatial weight sharing?

idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments ⟷ ambiguity of reference frames

solution: G-steerable kernels

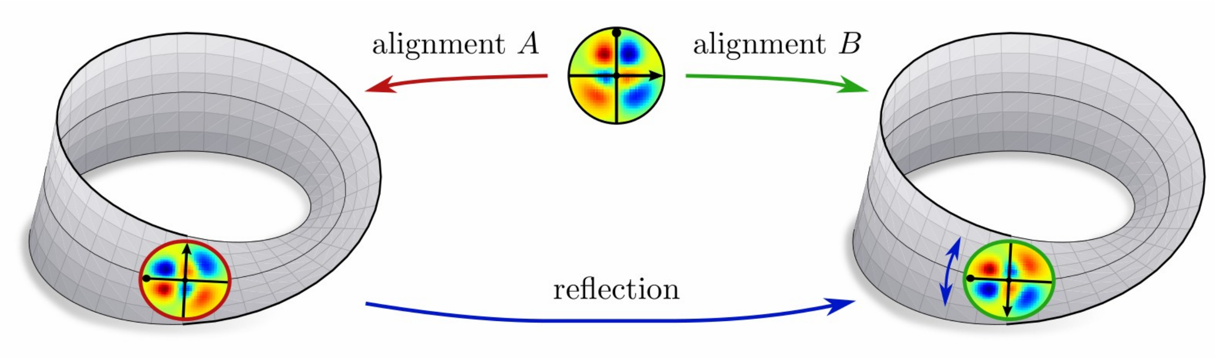


Riemannian manifolds - convolutions via spatial weight sharing?

idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments ⟷ ambiguity of reference frames

solution: G-steerable kernels

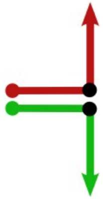


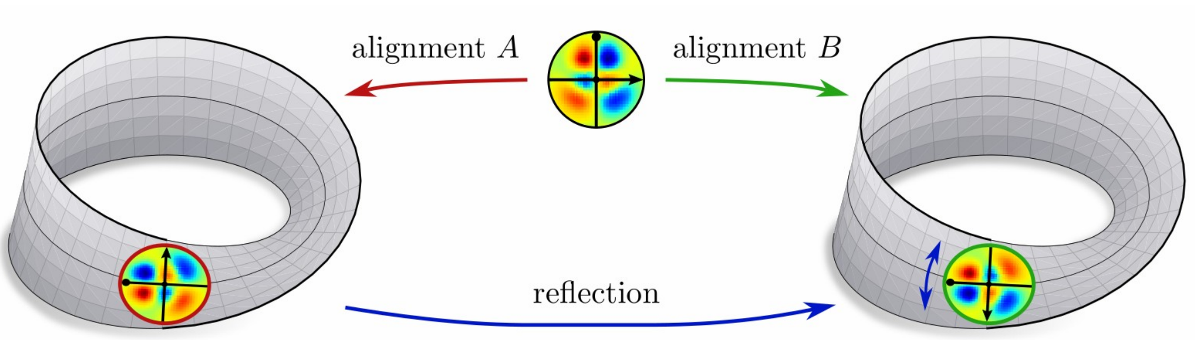
Riemannian manifolds - convolutions via spatial weight sharing?

idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments $\longleftrightarrow$ ambiguity of reference frames

solution: G-steerable kernels



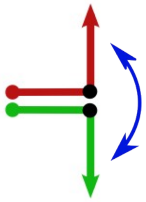


Riemannian manifolds - convolutions via spatial weight sharing?

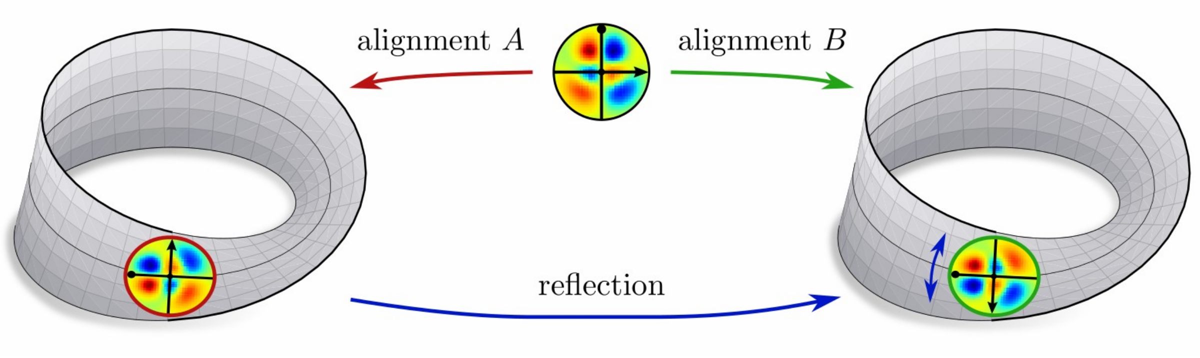
idea: despite lack of symmetries, apply kernel at each point

issue: ambiguous kernel alignments ⟷ ambiguity of reference frames

solution: G-steerable kernels



GAUGE FREEDOM !



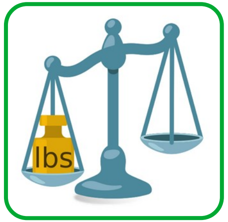
Gauge theory ?

“objects” often have *no canonical numerical representation*

gauge = *arbitrary* choice of such (“measurement units”)

gauge theories ensure consistent predictions among gauges

Gauge theory ?



“objects” often have *no canonical numerical representation*

gauge = *arbitrary* choice of such (“measurement units”)

gauge theories ensure consistent predictions among gauges

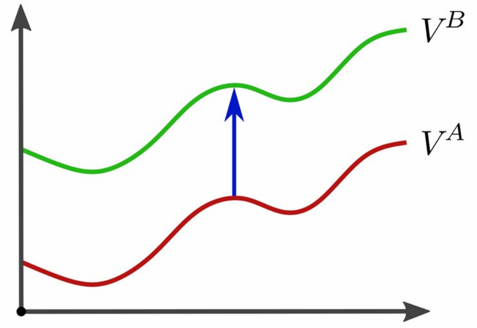
example	gauge fixing	gauge transformation
physical mass	weighing unit	unit conversion
potential energy	reference potential	potential offset
set	ordering	permutation
space / manifold	coordinate chart	chart transition map

Gauge theory ?

“objects” often have *no canonical numerical representation*

gauge = *arbitrary* choice of such (“measurement units”)

gauge theories ensure consistent predictions among gauges



example	gauge fixing	gauge transformation
physical mass	weighing unit	unit conversion
potential energy	reference potential	potential offset
set	ordering	permutation
space / manifold	coordinate chart	chart transition map

Gauge theory ?

“objects” often have *no canonical numerical representation*

(🧸, 🧨, 🛼)

gauge = *arbitrary* choice of such (“measurement units”)

(🧨, 🧸, 🛼)

gauge theories ensure consistent predictions among gauges

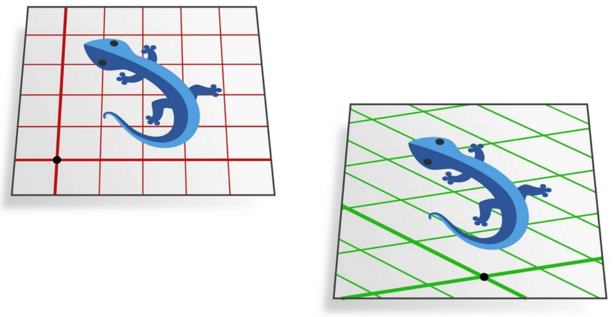
example	gauge fixing	gauge transformation
physical mass	weighing unit	unit conversion
potential energy	reference potential	potential offset
set	ordering	permutation
space / manifold	coordinate chart	chart transition map

Gauge theory ?

“objects” often have *no canonical numerical representation*

gauge = *arbitrary* choice of such ("measurement units")

gauge theories ensure consistent predictions among gauges



example	gauge fixing	gauge transformation
physical mass	weighing unit	unit conversion
potential energy	reference potential	potential offset
set	ordering	permutation
space / manifold	coordinate chart	chart transition map

Reference frames & coordinate independence

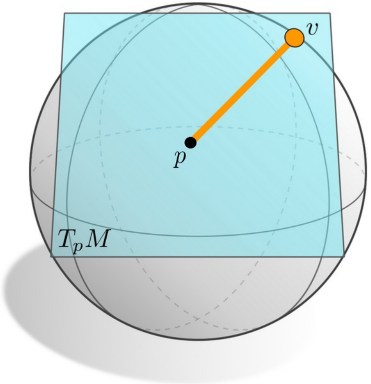
tangent vectors $v \in T_pM$ are *coordinate free*

in gauge A, v is expressed by numerical *coefficients* $v^A \in \mathbb{R}^d$

in gauge B, v is expressed by numerical *coefficients* $v^B \in \mathbb{R}^d$

different numbers,
same information content !

gauge trafos $g^{BA} \in \mathrm{GL}(d)$ relate coefficients: $v^B = g^{BA} v^A$



Reference frames & coordinate independence

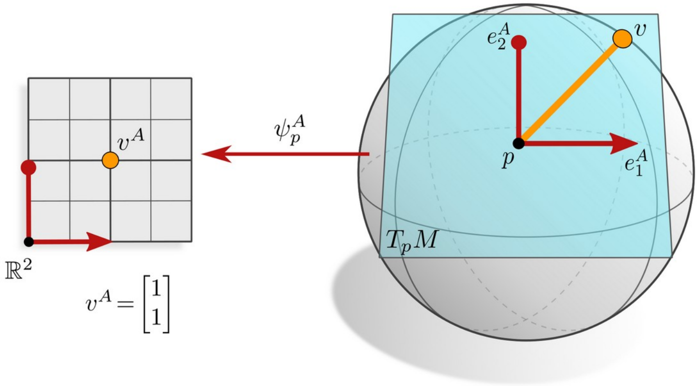
tangent vectors $v \in T_pM$ are *coordinate free*

in gauge A, v is expressed by numerical *coefficients* $v^A \in \mathbb{R}^d$

in gauge B, v is expressed by numerical *coefficients* $v^B \in \mathbb{R}^d$

different numbers,
same information content !

gauge trafos $g^{BA} \in \mathrm{GL}(d)$ relate coefficients: $v^B = g^{BA} v^A$



Reference frames & coordinate independence

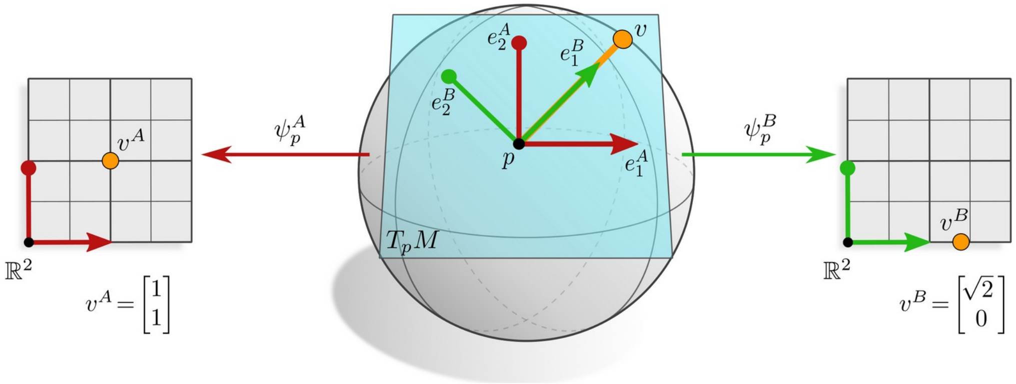
tangent vectors $v \in T_pM$ are *coordinate free*

in gauge A, v is expressed by numerical *coefficients* $v^A \in \mathbb{R}^d$

in gauge B, v is expressed by numerical *coefficients* $v^B \in \mathbb{R}^d$

different numbers,
same information content !

gauge trafos $g^{BA} \in \mathrm{GL}(d)$ relate coefficients: $v^B = g^{BA} v^A$



Reference frames & coordinate independence

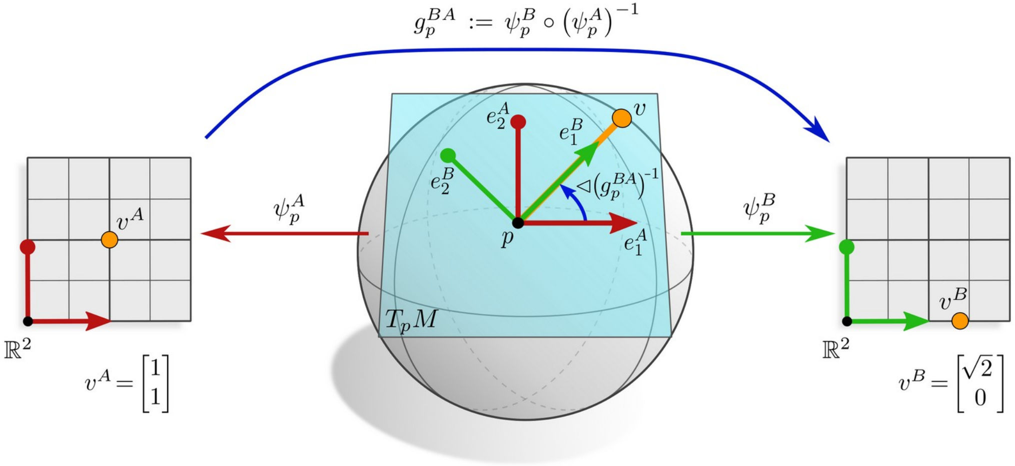
tangent vectors $v \in T_pM$ are *coordinate free*

in gauge A, v is expressed by numerical *coefficients* $v^A \in \mathbb{R}^d$

in gauge B, v is expressed by numerical *coefficients* $v^B \in \mathbb{R}^d$

different numbers,
same information content !

gauge trafos $g^{BA} \in \mathrm{GL}(d)$ relate coefficients: $v^B = g^{BA} v^A$



$g_p^{BA} := \psi_p^B \circ \left(\psi_p^A\right)^{-1}$
ψ_p^A
ψ_p^B
e_2^A
e_1^B
v
e_2^B
$\lhd (g_p^{BA})^{-1}$
p
e_1^A
T_pM
v^A
$\mathbb{R}^2$
$v^A = \begin{bmatrix}1\\1\end{bmatrix}$
v^B
$\mathbb{R}^2$
$v^B = \begin{bmatrix}\sqrt{2}\\0\end{bmatrix}$

Reference frames & coordinate independence

tangent vectors $v \in T_pM$ are *coordinate free*

in gauge A, v is expressed by numerical *coefficients* $v^A \in \mathbb{R}^d$

in gauge B, v is expressed by numerical *coefficients* $v^B \in \mathbb{R}^d$

different numbers,
same information content !

gauge trafos $g^{BA} \in \mathrm{GL}(d)$ relate coefficients: $v^B = g^{BA} v^A$

similar for feature vectors $f^A, f^B \in \mathbb{R}^c$: $f^B = \rho\big(g^{BA}\big) f^A$ (“associated G-bundles”)

Reference frames & coordinate independence

tangent vectors $v \in T_pM$ are *coordinate free*

in gauge A, v is expressed by numerical *coefficients* $v^A \in \mathbb{R}^d$

in gauge B, v is expressed by numerical *coefficients* $v^B \in \mathbb{R}^d$

different numbers,
same information content !

gauge trafos $g^{BA} \in \boxed{\mathrm{GL}(d)}$ relate coefficients: $v^B = g^{BA} v^A$

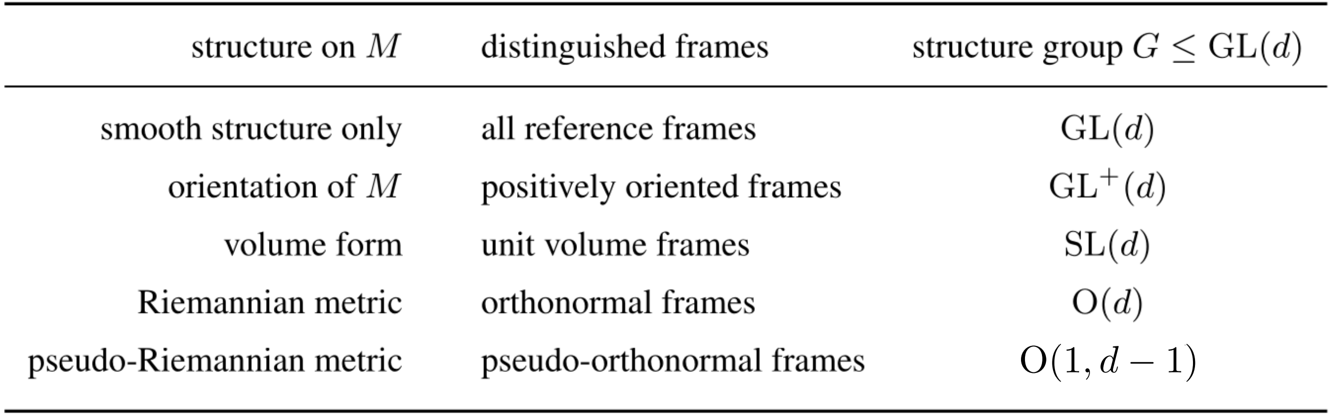
can often reduce to subgroup $G < \mathrm{GL}(d)$

similar for feature vectors $f^A, f^B \in \mathbb{R}^c$: $f^B = \rho\big(g^{BA}\big) f^A$ ("associated G-bundles")

Gauge freedom? $\longleftrightarrow$ G-structures!

ambiguity of frames on a manifold depends on its G-structure

existence of G-structure may be obstructed by manifold's topology

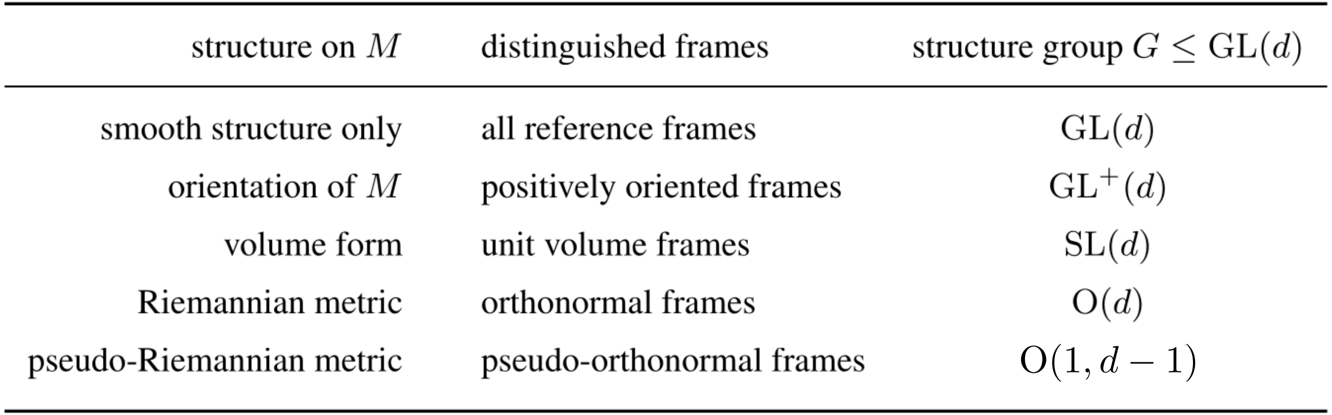


structure on M	distinguished frames	structure group $G \leq \mathrm{GL}(d)$
smooth structure only	all reference frames	$\mathrm{GL}(d)$
orientation of M	positively oriented frames	$\mathrm{GL}^+(d)$
volume form	unit volume frames	$\mathrm{SL}(d)$
Riemannian metric	orthonormal frames	$\mathrm{O}(d)$
pseudo-Riemannian metric	pseudo-orthonormal frames	$\mathrm{O}(1, d-1)$

Gauge freedom? $\longleftrightarrow$ G-structures!

ambiguity of frames on a manifold depends on its G-structure

existence of G-structure may be obstructed by manifold's topology

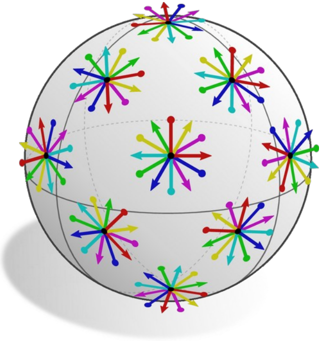


structure on M	distinguished frames	structure group $G \leq \mathrm{GL}(d)$
smooth structure only	all reference frames	$\mathrm{GL}(d)$
orientation of M	positively oriented frames	$\mathrm{GL}^+(d)$
volume form	unit volume frames	$\mathrm{SL}(d)$
Riemannian metric	orthonormal frames	$\mathrm{O}(d)$
pseudo-Riemannian metric	pseudo-orthonormal frames	$\mathrm{O}(1, d-1)$

Gauge freedom? ⟷ G-structures!

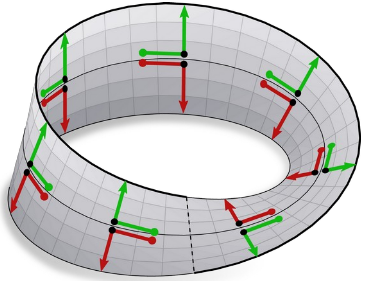
ambiguity of frames on a manifold depends on its G-structure

existence of G-structure may be obstructed by manifold’s topology



SO(2)-structure

(frames unique up to rotation)



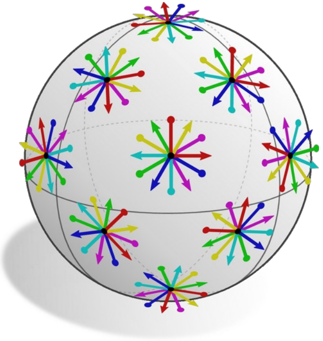
reflection group structure

(frames unique up to reflections)

Gauge freedom? ⟷ G-structures!

ambiguity of frames on a manifold depends on its G-structure

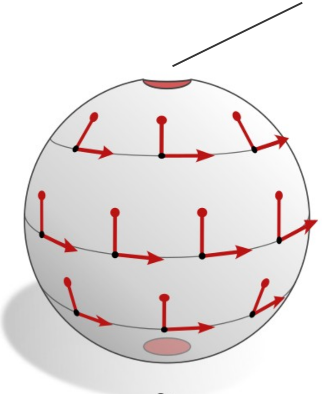
existence of G-structure may be obstructed by manifold's topology



SO(2)-structure

(frames unique up to rotation)

singularity,
non-continuous NN behavior!

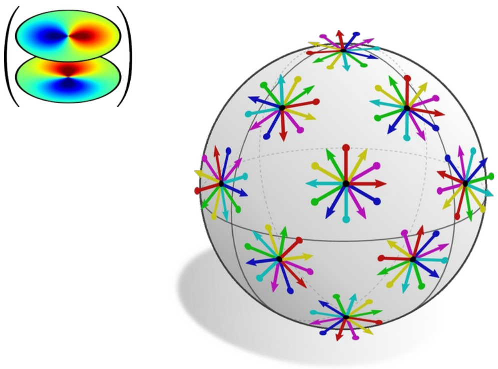


{e}-structure

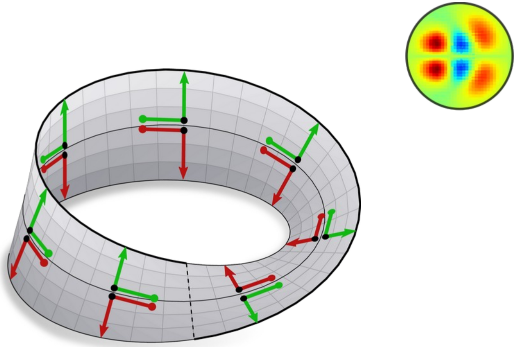
(unique frames)

Coordinate independent convolutions

Theorem: manifold with G-structure
remain coordinate independent $\Longrightarrow$ kernels need to be G-steerable



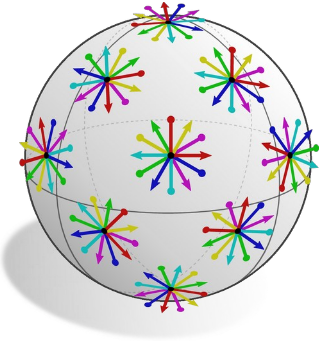
SO(2)-structure



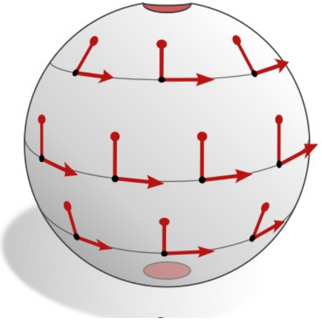
reflection group structure

Global symmetries

Theorem: equivariant w.r.t. *symmetries of G-structure* (“principal bundle automorphisms”)



SO(3)-equivariant

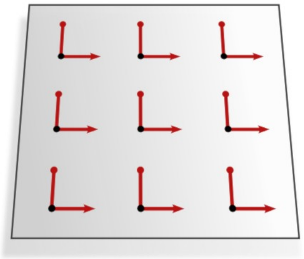


SO(2)-equivariant

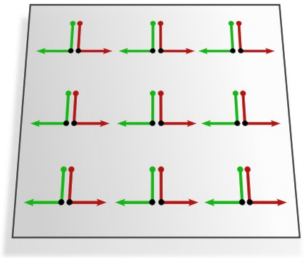
Global symmetries

Theorem: equivariant w.r.t. *symmetries of G-structure* (“principal bundle automorphisms”)

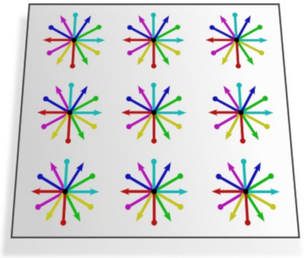
recovers Aff(G)-equivariant CNNs on Euclidean spaces



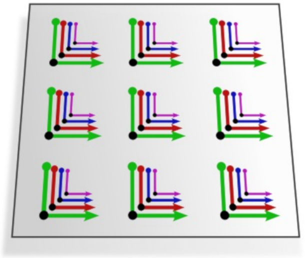
G = {e}



G = reflections



G = SO(2)

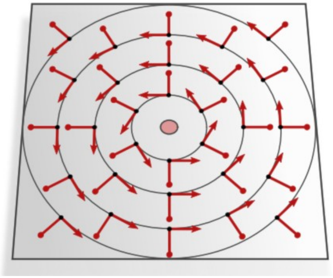


G = scaling

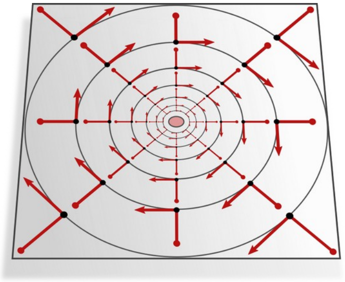
Global symmetries

Theorem: equivariant w.r.t. *symmetries of G-structure* ("principal bundle automorphisms")

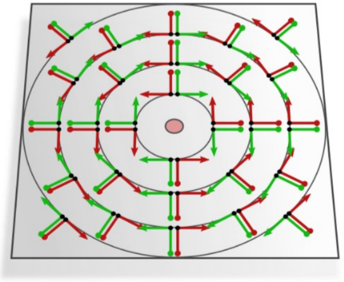
recovers Aff(G)-equivariant CNNs on Euclidean spaces & more exotic moels!



polar

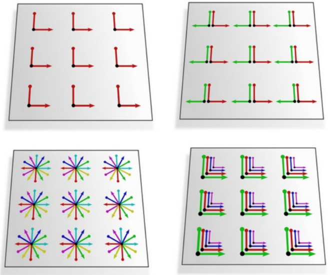


log-polar

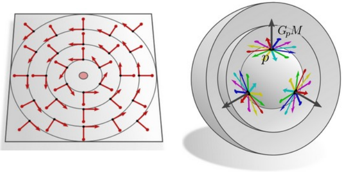


polar + reflections

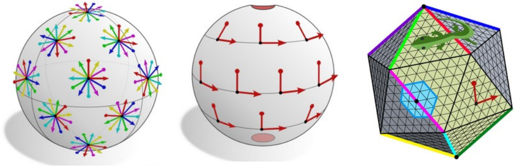
Euclidean steerable CNNs



punctured Euclidean

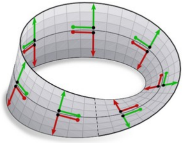


spherical / icosahedral



general 2d surfaces / meshes

Möbius



	manifold M	structure group G	global symmetry Aff_{GM} or Isom_{GM}	representation ρ	citation
1	$\mathbb{E}_d$	$\{e\}$	$\mathcal{T}_d$	trivial	[130, 253] + any conventional CNN
2	$\mathbb{E}_1$	$\mathcal{S}$	$\mathcal{T}_1 \rtimes \mathcal{S}$	regular	[186]
3	$\mathbb{E}_2$	$\mathcal{R}$	$\mathcal{T}_2 \rtimes \mathcal{R}$	regular	[234]
4		SO(2)	SE(2)	irreps	[244, 234, 231]
5				regular	[51, 33, 257, 34, 236, 8, 93, 192, 234, 79, 125, 210, 232, 185, 158, 201, 7, 67, 227, 83, 159, 231, 92, 50, 206, 19, 207, 208, 164, 29, 86]
6				quotients	[34, 234]
7				regular$\xrightarrow{\text{pool}}$trivial	[33, 143, 234]
8				regular$\xrightarrow{\text{pool}}$vector	[144, 234]
9		O(2)	E(2)	trivial	[110, 234]
10				irreps	[234]
11				regular	[51, 33, 93, 34, 234, 159, 79, 201]
12				quotients	[34]
13				regular$\xrightarrow{\text{pool}}$trivial	[234]
14				induced SO(2)-irreps	[234]
15		$\mathcal{S}$	$\mathcal{T}_2 \rtimes \mathcal{S}$	regular	[243, 212, 7, 258]
16				regular$\xrightarrow{\text{pool}}$trivial	[77]
17	$\mathbb{E}_3$	SO(3)	SE(3)	irreps	[235, 224, 156, 120, 2, 6]
18				quaternion	[250]
19				regular	[67, 241, 242]
20				regular$\xrightarrow{\text{pool}}$trivial	[3]
21		O(3)	E(3)	regular	[241]
22				quotient O(3)/ O(2)	[103]
23				irrep$\xrightarrow{\text{norm}}$trivial	[174]
24		C_4	$\mathcal{T}_3 \rtimes \mathrm{C}_4$	regular	[219]
25		D_4	$\mathcal{T}_3 \rtimes \mathrm{D}_4$	regular	[219]
26	$\mathbb{E}_{d-1,1}$	$\mathrm{SO}(d{-}1,1)$	$\mathcal{T}_d \rtimes \mathrm{SO}(d{-}1,1)$	irreps	[205]
27	$\mathbb{E}_2\backslash\{0\}$	$\{e\}$	SO(2)	trivial	[30, 67]
28			$\mathrm{SO}(2) \times \mathcal{S}$		[62, 67]
29	$\mathbb{E}_3\backslash\{0\}$	O(2)	O(3)	trivial	[178]
30		$\{e\}$	$\{e\}$	trivial	[13]
31	S^2	SO(2)	SO(3)	irreps	[122, 64]
32				regular	[35, 111]
33		O(2)	O(3)	trivial	[61, 169, 245]
34	$S^2 \setminus$ poles	$\{e\}$	SO(2)	trivial	[39, 222, 254, 149, 105, 217, 218, 55, 131]
35	icosahedron	C_6	I ($\approx$ SO(3))	regular	[38]
36	ico $\setminus$ poles	$\{e\}$	C_5 ($\approx$ SO(2))	trivial	[251, 139]
37	surface ($d{=}2$) (e.g. meshes)	SO(2)	$\mathrm{Isom}_+(M)$	irreps	[238]
38				regular	[173, 220, 246, 48]
39				regular$\xrightarrow{\text{pool}}$trivial	[150, 151, 160, 220]
40		D_4	$\mathrm{Isom}_{\mathrm{D}_4 M}$	trivial	[98]
41		$\{e\}$	$\mathrm{Isom}_{\{e\}M}$	trivial	[160, 194, 106, 221, 133]
42	Möbius strip	$\mathcal{R}$	SO(2)	irreps	Section 5
43				regular	

?

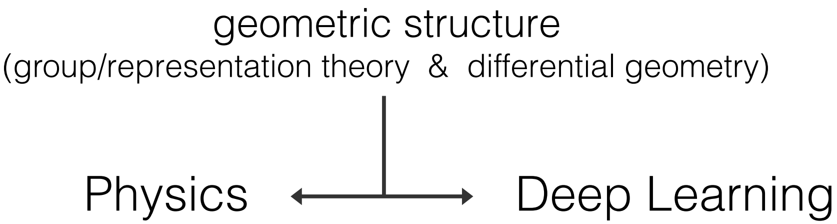
Physics	Deep Learning
tensor fields	feature fields
Minkowski space + global Poincaré symmetry	Euclidean space + global Aff(G) symmetry
curved spacetime + local Lorentz trafos	Riemannian manifold + local gauge trafos
invariant laws of nature (relativity)	invariant neural connectivity
equivariant system dynamics	equivariant inference
scalar / vector / tensor operators in QM	G-steerable kernels
quantum state transition rules	feature transition rules

Physics $\overset{?}{\longleftrightarrow}$ Deep Learning

tensor fields	feature fields
Minkowski space + global Poincaré symmetry	Euclidean space + global Aff(G) symmetry
curved spacetime + local Lorentz trafos	Riemannian manifold + local gauge trafos
invariant laws of nature (relativity)	invariant neural connectivity
equivariant system dynamics	equivariant inference
scalar / vector / tensor operators in QM	G-steerable kernels
quantum state transition rules	feature transition rules

Physics	$\xleftrightarrow{?}$	Deep Learning
tensor fields		feature fields
Minkowski space + global Poincaré symmetry		Euclidean space + global Aff(G) symmetry
curved spacetime + local Lorentz trafos		Riemannian manifold + local gauge trafos
invariant laws of nature (relativity)		invariant neural connectivity
equivariant system dynamics		equivariant inference
scalar / vector / tensor operators in QM		G-steerable kernels
quantum state transition rules		feature transition rules

Physics	$\overset{?}{\longleftrightarrow}$	Deep Learning
tensor fields		feature fields
Minkowski space + global Poincaré symmetry		Euclidean space + global Aff(G) symmetry
curved spacetime + local Lorentz trafos		Riemannian manifold + local gauge trafos
invariant laws of nature (relativity)		invariant neural connectivity
equivariant system dynamics		equivariant inference
scalar / vector / tensor operators in QM		G-steerable kernels
quantum state transition rules		feature transition rules



geometric structure

(group/representation theory & differential geometry)

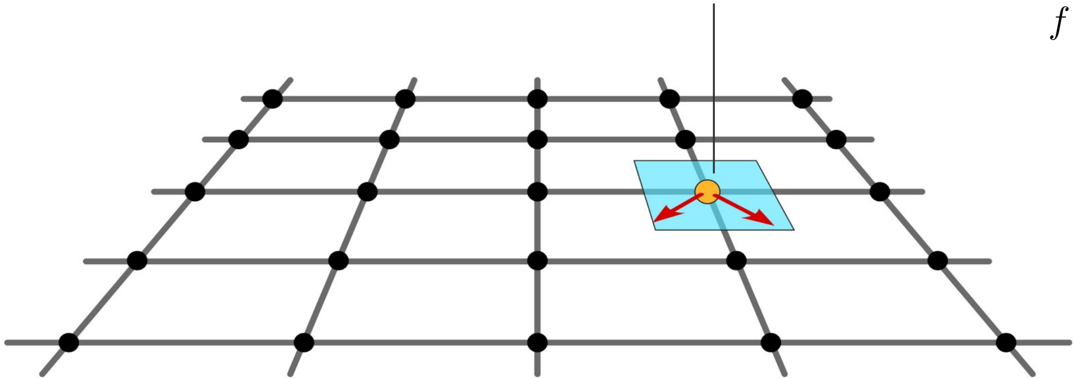
Physics	Deep Learning
tensor fields	feature fields
Minkowski space + global Poincaré symmetry	Euclidean space + global Aff(G) symmetry
curved spacetime + local Lorentz trafos	Riemannian manifold + local gauge trafos
invariant laws of nature (relativity)	invariant neural connectivity
equivariant system dynamics	equivariant inference
scalar / vector / tensor operators in QM	G-steerable kernels
quantum state transition rules	feature transition rules

(Lattice) gauge field theory - physics vs. ML

nodes: ML: feature vectors, associated to TM

physics: fermions, internal quantum space

$$f(x) \mapsto \rho(g_x) f(x)$$



edges: ML: parallel transporters ← given by geometry

physics: gauge bosons ← dynamical variables

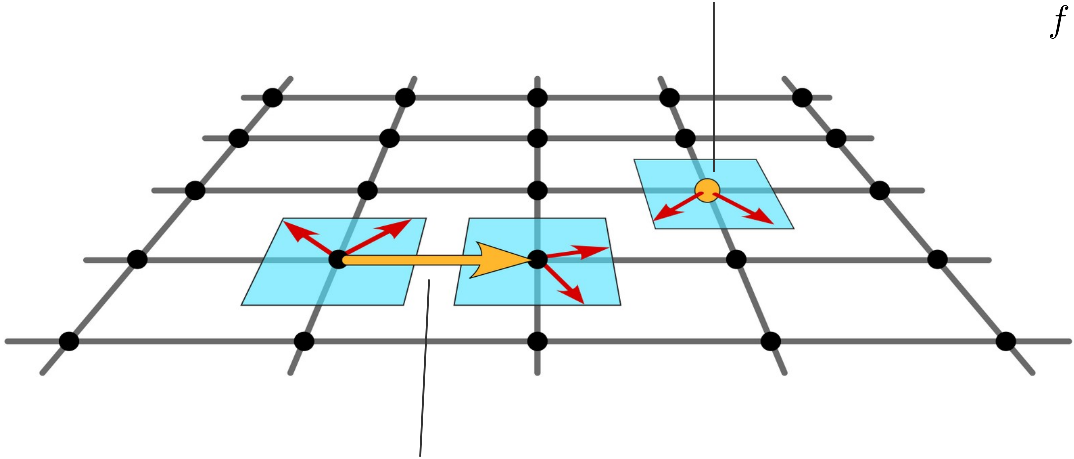
$$U_\mu(x) \;\mapsto\; g_{x+\mu}\, U_\mu(x)\, g_x^{-1}$$

(Lattice) gauge field theory - physics vs. ML

nodes: ML: feature vectors, associated to TM

physics: fermions, internal quantum space

$$f(x) \mapsto \rho(g_x) f(x)$$



edges: ML: parallel transporters ← given by geometry

physics: gauge bosons ← dynamical variables

$$U_\mu(x) \;\mapsto\; g_{x+\mu}\, U_\mu(x)\, g_x^{-1}$$

Thank you!

Maurice Weiler

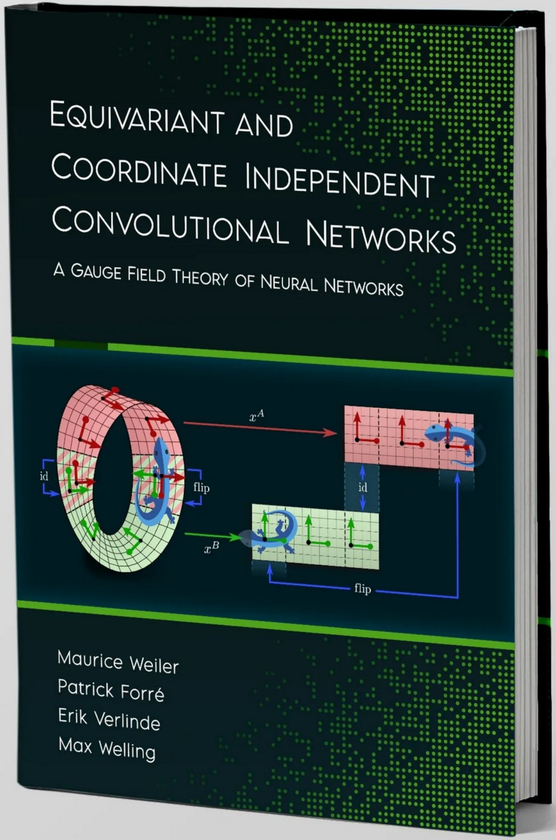
Jaakkola lab

MIT CSAIL

@maurice_weiler

Massachusetts Institute of Technology

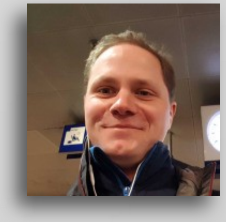
UNIVERSITY OF AMSTERDAM



EQUIVARIANT AND COORDINATE INDEPENDENT CONVOLUTIONAL NETWORKS

A GAUGE FIELD THEORY OF NEURAL NETWORKS

Maurice Weiler
Patrick Forré
Erik Verlinde
Max Welling



Patrick Forré

Erik Verlinde

Max Welling

https://maurice-weiler.gitlab.io/#cnn_book