

Part 1: Symmetry and Groups

Lecturer: Renjie Liao

Revised: Sep. 29, 2026

Note: *LaTeX template courtesy of UC Berkeley EECS dept.***Disclaimer:** *These notes were revised with assistance from GPT-6 Astra.*

An equivariant network specifies how its prediction should change when its input is transformed. The useful starting point is a task, not a list of algebraic definitions. This part assumes familiarity with vectors, matrices, functions, and neural networks; group theory is introduced as needed.

The route through these notes. Part 1 defines transformations and equivariance. Part 2 constructs a rotation-aware CNN by sharing filters. Part 3 explains how to rotate a filter using a small basis. Part 4 handles scalar, vector, and higher-order features. Part 5 assembles and tests complete networks.

1.1 Begin with the output you want

Consider a rigid rotation of an input image or a set of 3D points. The correct output transformation depends on the task:

Task	Input change	Desired output change
Object classification	Rotate the object	Keep the class scores
Segmentation	Rotate the image	Rotate the mask
Velocity prediction	Rotate positions and velocities	Rotate predicted vectors
Energy prediction	Rigidly rotate a molecule	Keep its scalar energy

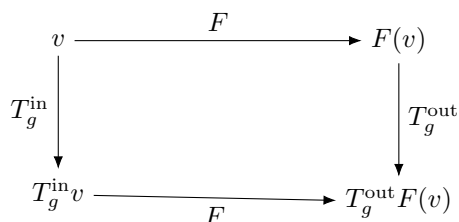
These are modeling assumptions. For example, rotating a handwritten *6* into a *9* need not preserve its label. A gravity direction or a fixed camera can also make some rotations physically meaningful. Choose a symmetry of the prediction problem, not merely a transformation that can be applied to the array.

Write T_g^{in} and T_g^{out} for the input and output transformations associated with g . A predictor F is *equivariant* if

$$F(T_g^{\text{in}}v) = T_g^{\text{out}}F(v) \quad \text{for every } g, v. \quad (1.1)$$

It is *invariant* when T_g^{out} is the identity, so the right-hand side is simply $F(v)$. Here v is the whole network input; we reserve x for a spatial coordinate below. Invariance is a special case of equivariance, with a different output transformation rule.

Our running CNN task. When an image is translated or rotated, a segmentation CNN should move its predicted mask by the same motion. Its hidden features must also follow a specified transformation rule. We will compare a grid CNN using translations and quarter-turns with its continuous-plane counterpart using translations and arbitrary planar rotations.



The two paths must give the same answer. This diagram is also the implementation test used in Part 5.

Architecture versus augmentation. Data augmentation trains on transformed examples. It encourages the desired behavior but does not impose (1.1) for every weight setting and input. An equivariant architecture restricts its layers so that the identity holds by construction in the stated mathematical setting. This can reduce the size of the function class the learner must consider, but it does not guarantee better test accuracy when the symmetry assumption is wrong or only approximate. The geometric viewpoint connects CNNs, permutation-equivariant GNNs, and rotation-equivariant models [1].

1.2 A group is a consistent collection of reversible transformations

We need to compose transformations, do nothing, and undo a transformation. A *group* G is a set with a product satisfying four rules:

1. **Closure:** gh is still in G .
2. **Associativity:** $(gh)k = g(hk)$.
3. **Identity:** an element e satisfies $eg = ge = g$.
4. **Inverse:** each g has g^{-1} with $gg^{-1} = g^{-1}g = e$.

Definition (group action). Let G be a group and X a set. A *left group action* of G on X is a map $\alpha : G \times X \rightarrow X$, written $\alpha(g, x) = g \cdot x$, such that for all $g, h \in G$ and $x \in X$,

$$e \cdot x = x, \quad g \cdot (h \cdot x) = (gh) \cdot x.$$

The codomain X explicitly requires $g \cdot x \in X$ for every $g \in G$ and $x \in X$: every allowed transformation stays within X .

Our convention is that gh applies h first and then g , as in matrix multiplication. There is no requirement that $gh = hg$. A crop that discards pixels is not invertible, so arbitrary cropping is not a group action on fixed-size images.

1.2.1 Quarter-turns: four integers suffice

Use an index $r \in \{0, 1, 2, 3\}$ for a counterclockwise rotation by $r\pi/2$. Indices compose by addition modulo four; the identity index is 0, and the inverse index is $-r \bmod 4$. Thus $(3 + 2) \bmod 4 = 1$: a 270° rotation followed by 180° is a 90° rotation. No continuous-angle interpolation is needed to rotate a square pixel array through these four orientations.

Write $R(\theta)$ for the coordinate matrix of rotation by angle θ :

$$R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}, \quad R_r = R(r\pi/2), \quad C_4 = \{R_0, R_1, R_2, R_3\}.$$

Throughout the mathematics, C_4 means this matrix group, with identity $R_0 = I$. The integer r is its storage index, not the matrix itself. The rule $R_{(r+s) \bmod 4} = R_r R_s$ connects the two implementations; storing r is enough.

1.2.2 Translations and rotations together

Translations form $(\mathbb{Z}^2, +)$ on a pixel grid and $(\mathbb{R}^2, +)$ in continuous space. A translation by t sends a coordinate x to $x + t$; its inverse is translation by $-t$.

To combine a translation and a rotation, write a pose as $g = (t, R)$, where $t \in \mathbb{Z}^2$ and $R \in C_4$, and define $g \cdot x = t + Rx$. Coordinates such as $(1, 0)$ denote column vectors; $(0, 0)$ is the zero translation vector and I is the identity rotation. The identity pose is $e = ((0, 0), I)$. For another pose $h = (u, S)$, composition and inversion give

$$\begin{aligned} (t, R)(u, S) &= (t + Ru, RS), \\ (t, R)^{-1} &= (-R^{-1}t, R^{-1}). \end{aligned} \tag{1.2}$$

The group $\mathbb{Z}^2 \rtimes C_4$ is commonly called $p4$. The symbol $\rtimes$ is a *semidirect product*: rotating the coordinate frame also rotates the translation vector. It is not the independent product $(t + u, RS)$. The notation (t, R) is an ordinary ordered pair, not a homogeneous-coordinate vector or matrix.

Check the order on one point. Let $g = ((0, 0), R_1)$ and $h = ((1, 0), I)$. Then gh sends the origin first to $(1, 0)$ and then to $(0, 1)$. In contrast, hg sends it first to the origin and then to $(1, 0)$. Rotation and translation do not commute. This same Ru term determines the filter indexing in Part 2.

Two motion groups for CNNs. The planar rotation group is $SO(2) = \{R \in \mathbb{R}^{2 \times 2} : R^\top R = I, \det R = 1\}$. It contains the matrices $R(\theta)$ for every rotation angle θ . Adding continuous translations gives $SE(2) = \mathbb{R}^2 \rtimes SO(2)$. These are the continuous counterparts of C_4 and $p4$:

CNN model	Locations X	Motions G	Rotations K
Grid	$\mathbb{Z}^2$	$p4 = \mathbb{Z}^2 \rtimes C_4$	C_4
Continuous plane	$\mathbb{R}^2$	$SE(2) = \mathbb{R}^2 \rtimes SO(2)$	$SO(2)$

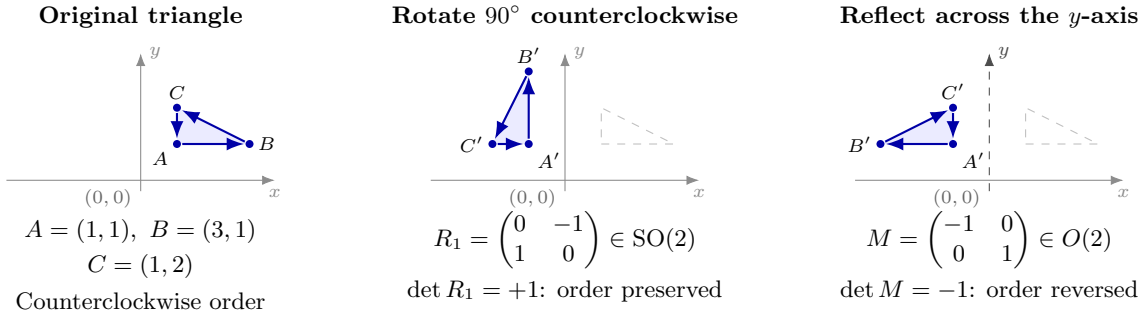
For both models, $g = (t, R)$ with $t \in X$ and $R \in K$; the same product and inverse in (1.2) apply. The *orientation projection* is always denoted by

$$\pi : G \longrightarrow K, \quad \pi((t, R)) = R.$$

It discards translation and retains the rotation matrix. For example, $\pi((t, R_1)) = R_1$; the stored orientation index would be 1. The rotation group K fixes the origin; the motion group G also moves locations. Reflections belong to $O(2) = \{R \in \mathbb{R}^{2 \times 2} : R^\top R = I\}$, and are excluded from both rotation families used here.

The $p4$ model acts on an ideal infinite grid; finite boundaries can affect equivariance. The full $SE(2)$ model acts on the continuous plane. Arbitrary-angle rotations and real-valued translations of sampled pixels require resampling, so numerical equivariance is generally approximate.

A 2D example: turning versus mirroring. Follow the labeled vertices $A \rightarrow B \rightarrow C$ in each panel. Rotation is about the origin.



Both preserve lengths and unsigned angles. Rotation keeps the counterclockwise order; reflection makes it clockwise. Thus $M \notin SO(2)$: no rotation within the plane produces this mirror image of the labeled triangle. Dashed triangles mark the original positions.

1.3 Move coordinates, then work out how the stored signal moves

Let G act on a coordinate set X as defined in Section 1.2. A signal $f : X \rightarrow \mathbb{R}$ stores a value at each coordinate. When a value at x moves to $g \cdot x$, the transformed signal must satisfy

$$[T_g f](x) = f(g^{-1} \cdot x). \quad (1.3)$$

The inverse answers a sampling question: which old coordinate moved to this new coordinate? When $X = G$ and g acts by the group product $h \mapsto gh$, (1.3) is called the *left-regular representation*. For general X , it is the representation on functions associated with the action on X . We explain this terminology in Section 1.5, after introducing representations.

The same sampling rule in both CNN models. For $g = (t, R)$, $[T_g f](x) = f(R^{-1}(x - t))$, on either $\mathbb{Z}^2$ or $\mathbb{R}^2$. A rightward translation $t = (2, 0)$ therefore samples $f(x - t)$: a peak at the origin moves to $(2, 0)$. A quarter-turn rotates the grid by an exact permutation; the continuous model allows every angle.

Proposition 1.1 (The inverse gives the correct composition order) *For an action of G on X , the operators in (1.3) satisfy $T_g T_h = T_{gh}$.*

Proof: For each x , $[T_g T_h f](x) = f(h^{-1} \cdot (g^{-1} \cdot x)) = f((gh)^{-1} \cdot x) = [T_{gh} f](x)$. ■

Implementation consequence. Test the transformation code itself before testing a network. A wrong inverse or composition order can make a correct layer appear incorrect. Keep the mathematical x, y axes separate from array row/column indexing; image rows normally increase downward.

1.3.1 A representation tells channels how to transform

Equation (1.3) specifies how to move a scalar signal between locations. A feature map $f : X \rightarrow \mathbb{R}^C$ stores C numbers, called *channels*, at each location. We must also specify how these numbers transform when the scene moves.

A *linear representation* of G on a vector space V is a map $\sigma : G \rightarrow \text{GL}(V)$, where $\text{GL}(V)$ is the group of invertible linear maps $V \rightarrow V$. It is a *group homomorphism*: it preserves composition, so $\sigma(gh) = \sigma(g)\sigma(h)$ and $\sigma(e) = I$. Thus composing group elements corresponds to composing their linear maps. We revisit homomorphisms in Section 1.5.

In finite-dimensional coordinates, these maps are matrices. The dimension here is that of V , not the number of elements in G : an infinite group can have a finite-dimensional representation. For $V = \mathbb{R}^C$, a channel vector v transforms as $\tilde{v} = \sigma(g)v$. Here *representation* means a transformation rule, not a learned embedding.

Example: two channels can obey different rules. For CNN channels, reserve $\rho : K \rightarrow \text{GL}(\mathbb{R}^C)$ for the representation of the rotation group K , distinct from the projection $\pi : G \rightarrow K$. Thus π extracts a rotation; ρ tells channels how to respond to it. Rotate a scene counterclockwise while keeping the coordinate axes fixed. If the channels store the two velocity components $v = (v_x, v_y)^\top$, then $\rho(R) = R$. A quarter-turn gives

$$\rho(R_1) = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \quad \rho(R_1) \begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} -v_y \\ v_x \end{pmatrix}.$$

Thus eastward velocity $(1, 0)^\top$ becomes northward velocity $(0, 1)^\top$. If the two channels instead store temperature and pressure, each is a scalar: $\rho(R) = I_2$. Both values are carried unchanged to the rotated location. The same two-channel array can therefore encode two scalars or one geometric vector; its channel count does not determine the rule.

Combine location and channel transformations. Write $\tilde{f} = T_{(t, R)} f$ for the transformed field. For these scalar and velocity fields, spatial translations move the stored values without changing their channel components. Their full-pose channel representation is $g \mapsto \rho(\pi(g))$. The operator T_g additionally moves spatial locations, so it represents the motion group on entire *fields*.

If a point x moves to $y = Rx + t$, its feature satisfies $\tilde{f}(y) = \rho(R)f(x)$. Solving for the old location gives

$$\tilde{f}(y) = [T_g f](y) = \rho(\pi(g))f(g^{-1} \cdot y) = \rho(R)f(R^{-1}(y - t)), \quad g = (t, R).$$

For a scalar signal, $\rho(R) = 1$, recovering (1.3). Translation enters through the sampling location $R^{-1}(y - t)$. For a pure translation, $R = I$ and $\rho(I) = I_C$, hence $[T_{(t, I)} f](y) = f(y - t)$: the field generally changes even though its channel action is the identity. A wind map requires rotating its vectors as well as moving their locations; a temperature map requires only spatial resampling.

Translation on the space of signals. Linearity refers to the *space on which the operator acts*. The point transformation $x \mapsto x + t$ is affine, not linear on $\mathbb{R}^2$ when $t \neq (0, 0)$: it does not fix the zero vector. But the signal operator $[T_{(t,I)}f](x) = f(x - t)$ is linear in f :

$$T_{(t,I)}(c_1 f_1 + c_2 f_2) = c_1 T_{(t,I)} f_1 + c_2 T_{(t,I)} f_2, \quad c_1, c_2 \in \mathbb{R}.$$

Also $T_{(t,I)}T_{(u,I)} = T_{(t+u,I)}$ and $T_{(t,I)}^{-1} = T_{(-t,I)}$. Thus $t \mapsto T_{(t,I)}$ is a representation of the translation group on a vector space of functions. On a continuous domain this space is generally infinite-dimensional, so its operators need not be finite matrices.

Optional homogeneous coordinates for poses. The pose $g = (t, R)$ can alternatively be encoded by a 3×3 matrix:

$$A(g) = \begin{pmatrix} R & t \\ (0, 0)^\top & 1 \end{pmatrix}, \quad A(g) \begin{pmatrix} x \\ 1 \end{pmatrix} = \begin{pmatrix} Rx + t \\ 1 \end{pmatrix}.$$

Block multiplication gives $A(gh) = A(g)A(h)$; also $A(g^{-1}) = A(g)^{-1}$. This is a linear representation on $\mathbb{R}^3$. Restricting to points with last coordinate 1 recovers the affine action on $\mathbb{R}^2$. For a direction vector, $A(g) \begin{pmatrix} v \\ 0 \end{pmatrix} = \begin{pmatrix} Rv \\ 0 \end{pmatrix}$: translation has no effect. In particular, a pure translation (t, I) leaves direction components unchanged. This optional coordinate representation differs from the field operators T_g and channel matrices $\rho(R)$. Neither the pair product in (1.2) nor the orientation projection π requires homogeneous coordinates. We use A for these matrices and reserve H for subgroups.

1.4 Symmetrize a predictor by averaging

As a baseline for the CNN task, we can combine predictions from transformed inputs, undoing each output transformation before averaging. This requires no restriction on the predictor's weights. Consider a finite group and any (possibly nonlinear) predictor $F : V_{\text{in}} \rightarrow V_{\text{out}}$, with linear representations σ_{in} and σ_{out} of G on the input and output. These play the roles of T_g^{in} and T_g^{out} in (1.1). Define

$$F_{\text{eq}}(v) = \frac{1}{|G|} \sum_{g \in G} \sigma_{\text{out}}(g)^{-1} F(\sigma_{\text{in}}(g)v). \quad (1.4)$$

Run the predictor on each transformed input, undo the output transformation, then average. For invariant classification, the output undoing step is the identity. For segmentation it is a spatial inverse rotation; without it the masks would be averaged in different frames.

Theorem 1.2 (Finite-group symmetrization) *The predictor in (1.4) is equivariant, even when F itself is not equivariant. If F is already equivariant, $F_{\text{eq}} = F$.*

Proof: For $h \in G$, substitute $k = gh$ in the sum. Since $\sigma_{\text{out}}(kh^{-1})^{-1} = \sigma_{\text{out}}(h)\sigma_{\text{out}}(k)^{-1}$, the sum becomes $F_{\text{eq}}(\sigma_{\text{in}}(h)v) = \sigma_{\text{out}}(h)F_{\text{eq}}(v)$. For an equivariant F , every summand in (1.4) equals $F(v)$. ■

Why use averaging? The theorem gives a direct way to enforce the required equivariance for a finite group, provided transformations and averaging are exact. For our grid CNN, we can use C_4 rotations. The full $p4$ and $\text{SE}(2)$ motion groups are infinite, so this finite-sum formula does not cover them. Averaging costs $|G|$ evaluations. Using a few sampled rotations generally gives only an approximation to an $\text{SO}(2)$ rotation average. Group convolutions instead share computation and constrain the intermediate features; Part 2 builds them explicitly [2].

1.5 Group structure and CNN feature domains

Our CNN starts with values at image locations and builds features indexed by location and orientation. Several poses can reach the same location. We need to describe this redundancy and how motions act on both feature domains. The following definitions and results supply that description; their CNN interpretation follows each result.

Subgroups. A subset $H \subseteq G$ is a subgroup if it contains the identity and is closed under products and inverses. We write $H \leq G$, read “ H is a subgroup of G ,” for this relation. Our CNN rotation subgroups are $C_4 \leq p4$ and $SO(2) \leq SE(2)$, embedded as poses with zero translation. Within C_4 , $C_2 = \{R_0, R_2\}$ keeps only 0° and 180° . Equivariance to G implies equivariance to H ; the reverse is not implied.

Homomorphisms and isomorphisms. A homomorphism $\phi : G \rightarrow G'$ between groups satisfies $\phi(gh) = \phi(g)\phi(h)$. A representation is a homomorphism into the group of invertible linear maps. An *isomorphism* is a bijective homomorphism. We write $G \simeq G'$, read “ G is isomorphic to G' ,” when such a map exists between the groups. The quarter-turn matrices and integers with addition modulo four are isomorphic. For the orientation projection $\pi : G \rightarrow K$, take $g = (t, R)$ and $h = (u, S)$. Expanding the pair product gives

$$\pi(gh) = \pi((t + Ru, RS)) = RS = \pi(g)\pi(h).$$

Thus π is a homomorphism: orientations compose independently of translations. Here RS is ordinary 2×2 rotation-matrix multiplication; no homogeneous coordinates are assumed.

Left-regular representation. Return to (1.3) with $X = G$. The coordinate is now a group element h , and the left group action sends h to gh . Writing L_g for T_g in this case gives

$$[L_g f](h) = f(g^{-1}h), \quad f : G \rightarrow \mathbb{R}.$$

The value stored at h moves to gh . These linear maps obey $L_g L_k = L_{gk}$, just as proved for T_g above; hence $g \mapsto L_g$ is a representation. The word *left* refers to $h \mapsto gh$; *regular* identifies this action of G on itself.

For finite G , store one value per group element: L_g simply permutes these values. For C_4 , $[L_{R_1} f](R_s) = f(R_{(s-1) \bmod 4})$, so

$$(f(R_0), f(R_1), f(R_2), f(R_3)) \mapsto (f(R_3), f(R_0), f(R_1), f(R_2)).$$

For both $p4$ and $SE(2)$, an image is indexed by a location, whereas a lifted feature map is indexed by a full pose (location and orientation). The latter carries the left-regular representation: its orientation index has four values for $p4$ and a continuous angle for $SE(2)$. Part 2 constructs the discrete group convolutions [2].

Orbits and stabilizers. The orbit of $x_0 \in X$ is $\text{Orb}_G(x_0) = \{g \cdot x_0 : g \in G\}$. Its stabilizer is the subgroup $H = \text{Stab}_G(x_0) = \{g \in G : g \cdot x_0 = x_0\}$. For our two CNN actions, the origin’s stabilizer is the rotation subgroup: C_4 inside $p4$, or $SO(2)$ inside $SE(2)$. A stabilizer fixes one chosen point, not every point of the domain.

Cosets and quotient spaces. Different poses can reach the same location. A coset describes the redundancy by collecting poses that differ by a subgroup transformation. We work at the level of sets and group actions; no topology or smooth structure is needed here [3].

Definition 1.3 (Cosets and quotient space) Fix a subgroup $H \leq G$. For each $g \in G$, the left coset of H represented by g is $gH = \{gh : h \in H\}$. The quotient space (or coset space) is

$$G/H = \{gH : g \in G\},$$

read “ $G \bmod H$.” Its elements are cosets, not individual elements of G . Equivalently, it groups elements under the equivalence relation $g_1 \sim g_2$ if and only if $g_1^{-1}g_2 \in H$; this holds exactly when $g_1H = g_2H$.

Thus g and gh describe the same element of G/H : the quotient forgets the change from g to the group product gh for any $h \in H$. The set G/H need not itself be a group.

Homogeneous spaces. The goal is to extend convolution's weight sharing beyond grids: when symmetries connect every location, one local rule can be shared across the domain. Homogeneous spaces formalize this setting.

Definition 1.4 (Homogeneous space) Let X be a nonempty set with a group action $\alpha : G \times X \rightarrow X$. It is a homogeneous G -space if the action is transitive: for every $x, y \in X$, there exists $a \in G$ such that $a \cdot x = y$. Equivalently, starting from any $x_0 \in X$, the group can reach every point:

$$\text{Orb}_G(x_0) = \{g \cdot x_0 : g \in G\} = X \quad \text{for every } x_0 \in X.$$

To see why transitivity means one orbit, the action's codomain gives $\text{Orb}_G(x_0) \subseteq X$. Transitivity supplies the reverse inclusion: each $y \in X$ equals $g \cdot x_0$ for some $g \in G$. Conversely, if $\text{Orb}_G(x_0) = X$ for one x_0 , write any two points as $x = g \cdot x_0$ and $y = h \cdot x_0$. Then $(hg^{-1}) \cdot x = y$, so the action is transitive. Different group elements may send x_0 to the same point, so a subset of G may already produce the whole orbit.

Every coset space G/H is homogeneous under the action

$$a \cdot (gH) = (ag)H.$$

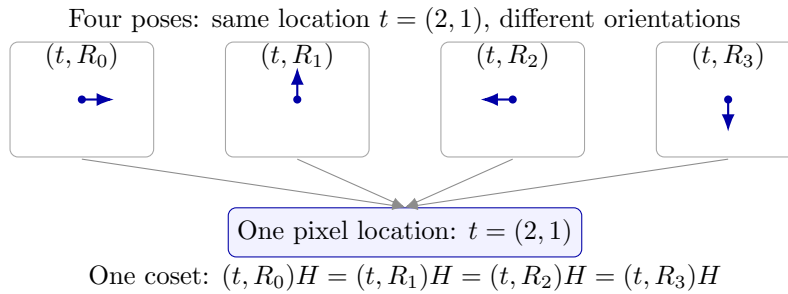
This is independent of the representative g : replacing it by gh , with $h \in H$, leaves $(ag)H$ unchanged. It is transitive because $a = g_2g_1^{-1}$ sends g_1H to g_2H .

Coset correspondence. An image has a feature vector at each location $x \in X$. A lifted feature map has one at each pose $g \in G$, including orientation. This distinction applies to both CNN models: $(p4, C_4)$ on the grid and $(\text{SE}(2), \text{SO}(2))$ on the plane. We need a common description of these domains and how the motion group moves their feature values.

CNN examples. For either model, choose the origin $x_0 = (0, 0)$, with stabilizer $H = \{((0, 0), Q) : Q \in K\}$. These actions are transitive because translations reach every location. For $g = (t, R)$ and $h = ((0, 0), S) \in H$,

$$g \cdot x_0 = t, \quad gh = (t, RS), \quad gH = \{(t, Q) : Q \in K\}.$$

In the group product gh , the orientation becomes RS while the location remains t . Thus one coset collects exactly the poses reaching that location: four orientations for $p4$, and all angles for $\text{SE}(2)$. In the quotient notation $p4/C_4$ and $\text{SE}(2)/\text{SO}(2)$, the rotation groups mean these embedded stabilizers H . At $t = (2, 1)$, the discrete example has four poses (t, R_r) . Here $h = ((0, 0), R_k)$ gives $gh = (t, R_{(r+k) \bmod 4})$:



Arrows inside the boxes indicate orientation. All four poses have exactly the same translation; the downward arrows forget orientation.

Proposition 1.5 (A homogeneous space is a coset space) If G acts transitively on X , choose $x_0 \in X$ and let $H = \text{Stab}_G(x_0)$. Then

$$\Phi : G/H \longrightarrow X, \quad \Phi(gH) = g \cdot x_0$$

is an equivariant bijection. We write $G/H \simeq X$ as G -spaces, read “ $G \bmod H$ is isomorphic to X as a G -space.”

Proof: We have $g_1 \cdot x_0 = g_2 \cdot x_0$ if and only if $g_1^{-1}g_2 \in H$, equivalently $g_1H = g_2H$. Hence Φ is well-defined and injective. Transitivity makes it surjective, and $\Phi(a \cdot gH) = a \cdot \Phi(gH)$ proves equivariance. ■

Nontransitive example. The map Φ reaches precisely the points that the group can move x_0 to. For example, let $G = \text{SO}(2)$ act on $X = \mathbb{R}^2$ by rotations alone, and choose $x_0 = (1, 0)$. Its stabilizer is $H = \{I\}$, and

$$\Phi(R(\theta)H) = (\cos \theta, \sin \theta).$$

Every coset names one point on the unit circle. No rotation reaches the origin or $(2, 0)$ from x_0 , because rotations preserve distance from the origin. Thus G/H describes this circle, the orbit of x_0 , not the entire plane. Adding translations gives $\text{SE}(2)$ and makes the action transitive on the plane.

CNN interpretation. For the two CNN actions, the proposition gives

$$p4/C_4 \simeq \mathbb{Z}^2, \quad \text{SE}(2)/\text{SO}(2) \simeq \mathbb{R}^2,$$

as spaces with their respective group actions. Choose $x_0 = (0, 0)$, so $\Phi((t, R)H) = t$. A G -space is a set equipped with a G -action. “Isomorphic as G -spaces” says that the two descriptions have the same locations and the same motion rules: Φ pairs every coset with exactly one location and vice versa. Transforming the coset then reading its location agrees with transforming that location directly:

$$\Phi(a \cdot (gH)) = a \cdot \Phi(gH).$$

Neither space needs a group product. The statement compares their actions, rather than asserting a group isomorphism.

Grid example. At $t = (2, 1)$, the four poses (t, R_r) all belong to one coset, which Φ labels by the same pixel location $(2, 1)$. Apply the quarter-turn $a = ((0, 0), R_1)$. Taking the group product with a on the left sends every representative to $((-1, 2), R_1R_r)$, so the new coset is labeled $(-1, 2)$. Rotating the pixel location directly gives the same result.

Continuous example. All poses $((2, 1), R(\theta))$, for any angle θ , belong to the coset labeled $(2, 1)$. Let $a = ((1, 0), R(\pi/2))$, a quarter-turn followed by a rightward translation. The new translation is

$$(1, 0) + R(\pi/2)(2, 1) = (1, 0) + (-1, 2) = (0, 2).$$

Every representative therefore moves to $((0, 2), R(\pi/2)R(\theta))$. Reading the new coset’s location gives $(0, 2)$, exactly as applying a to $(2, 1)$ does.

For a CNN, the correspondence means that an image can be indexed by ordinary spatial locations or by these cosets without changing its values or transformation rule. For a scalar image, $\tilde{f}(gH) = f(\Phi(gH))$ stores the same value under a coset label. A lifting layer subsequently computes detector responses at individual poses in G , with four orientation entries at each grid location for $p4$, or a continuous orientation coordinate for $\text{SE}(2)$.

Normal subgroups and quotient groups. We have labeled image locations by poses with orientation discarded. If we know the locations of two poses but not their orientations, can we determine the location of their composition? The following condition determines whether the group product of poses defines an unambiguous product on cosets.

A subgroup H is *normal* if $gHg^{-1} = H$ for every $g \in G$, where $gHg^{-1} = \{ghg^{-1} : h \in H\}$.

Representatives. A representative of a coset is one particular group element in it. To form a product, choose g_1 from the first coset and g_2 from the second. These choices are not unique: g_1h_1 names the same coset as g_1 for any $h_1 \in H$, and similarly for g_2h_2 . The proposed coset product is

$$(g_1H)(g_2H) = (g_1g_2)H$$

and must give the same output coset for every such choice of representatives. This holds if and only if H is normal. To see why normality guarantees this, write

$$(g_1h_1)(g_2h_2) = g_1g_2(g_2^{-1}h_1g_2)h_2.$$

Normality gives $g_2^{-1}h_1g_2 \in H$, so the two trailing factors have a product in H . Therefore $(g_1h_1)(g_2h_2)H = (g_1g_2)H$: changing the representatives does not change the output coset.

Why this is a group. Its elements are cosets, with the product above inherited from G . Closure holds because $g_1g_2 \in G$, and associativity holds because either bracketing gives $(g_1g_2g_3)H$. The identity is $H = eH$, and the inverse of gH is $g^{-1}H$:

$$H(gH) = (gH)H = gH, \quad (gH)(g^{-1}H) = (g^{-1}H)(gH) = H.$$

Thus all group axioms hold. We call this the *quotient group* because each coset identifies elements differing by a right factor in H . The map $g \mapsto gH$ is a homomorphism by the product rule.

Counterexample: forgetting orientation. For the origin rotation subgroup H in either CNN motion group, the proposed coset product depends on the representatives. Let $e = ((0,0), I)$, $q = ((0,0), R_1)$, and $\tau = ((1,0), I)$. In the proposed coset product, take $g_1 = e$ or $g_1 = q$, and $g_2 = \tau$. The poses e and q have the same location, so $eH = qH = H$. But composing each with the same rightward translation τ gives

$$(e\tau) \cdot x_0 = (1,0), \quad (q\tau) \cdot x_0 = (0,1).$$

Remember that $q\tau$ applies τ first, then the quarter-turn q . The output locations differ, so $(e\tau)H \neq (q\tau)H$. Thus the same input coset labels H and τH would give different outputs depending on whether we secretly chose e or q . A location does not contain enough information to compose full poses.

Example: forgetting translation. For either model, let $N = \{(t, I) : t \in X\}$ be the translation subgroup. For the product, take $g_1 = (t, R)$ and $g_2 = (u, S)$. A coset now keeps only orientation:

$$(t, R)N = \{(w, R) : w \in X\}, \quad (t, R)(u, S) = (t + Ru, RS).$$

The remaining orientation is always RS , regardless of the forgotten translations t, u . Hence N is normal, giving

$$p4/\mathbb{Z}^2 \simeq C_4, \quad \text{SE}(2)/\mathbb{R}^2 \simeq \text{SO}(2) \quad \text{as groups.}$$

Here $\mathbb{Z}^2$ and $\mathbb{R}^2$ denote the embedded translation subgroups N . The surviving quarter-turn indices add modulo four; continuous angles add modulo 2π . These orientation labels can be composed.

Conjugation. In ghg^{-1} , apply g^{-1} first, then h , then g . Here h belongs to the subgroup being tested, while g may be any element of the full group. We now identify both transformations explicitly.

Conjugating an origin rotation. For the origin's rotation subgroup H in either motion group, choose $t = (1, 0)$, $g = (t, I)$, and $h = ((0, 0), R_1) \in H$. Thus g translates right by one, and h rotates counterclockwise by 90° about the origin. For any coordinate x ,

$$x \xrightarrow{g^{-1}} x - t \xrightarrow{h} R_1(x - t) \xrightarrow{g} t + R_1(x - t).$$

This is a quarter-turn about the new center t : it leaves t fixed, but sends the origin to $t - R_1t = (1, -1)$. In pose notation,

$$ghg^{-1} = ((1, -1), R_1) \notin H.$$

We chose $h \in H$, yet conjugating it by this g leaves H . Therefore C_4 is not normal in $p4$, and $\text{SO}(2)$ is not normal in $\text{SE}(2)$, when embedded as origin rotations.

Conjugating a translation. For either translation subgroup N , take $g = (t, R)$ and $h = (u, I) \in N$, where $t, u \in X$ and $R \in K$. Thus $g \cdot x = t + Rx$ and $h \cdot x = x + u$. Step by step,

$$\begin{aligned} g^{-1} \cdot x &= R^{-1}(x - t), \\ h \cdot (g^{-1} \cdot x) &= R^{-1}(x - t) + u, \\ (ghg^{-1}) \cdot x &= t + R(R^{-1}(x - t) + u) = x + Ru. \end{aligned}$$

Hence $ghg^{-1} = (Ru, I) \in N$: the displacement rotates, but the result is still a translation. Since R maps the relevant X bijectively to itself, Ru ranges over every translation vector as u varies; therefore $gNg^{-1} = N$ for every pose g , proving normality. For example, $g = ((0, 0), R_1)$ and $h = ((1, 0), I)$ give $ghg^{-1} = ((0, 1), I)$ in both models: the quarter-turn changes a rightward translation into an upward translation.

For CNN feature domains, the action $a \cdot (gH) = (ag)H$ works for *every* subgroup H ; normality is required only if we also want a coset product inherited from G . Thus both image domains $p4/C_4$ and $SE(2)/SO(2)$ support equivariant layers even though they have no group product inherited from their motion groups. The identified $\mathbb{Z}^2$ and $\mathbb{R}^2$ still have their own addition. The purpose of normality here is to distinguish a valid domain of feature locations from a group of transformations with its own composition rule.

Part 2 uses these domains and actions to construct lifting layers and group convolutions, prove their equivariance, and explain pooling and implementation. Here the distinction is geometric: image values live at locations in G/H , while lifted features also retain orientation and live on G .

1.6 Checks before moving on

1. For a segmentation network, state both T_g^{in} and T_g^{out} . *Answer:* rotate the image and the mask in the same spatial frame.
2. Does rotating each filter independently guarantee that a multi-layer network is equivariant? *Answer:* no; intermediate channels must follow a compatible transformation rule.
3. In the $p4$ CNN, an orientation-indexed feature map has shape $[B, C, 4, n_y, n_x]$, with spatial height n_y and width n_x . What changes under a quarter-turn? *Answer:* spatial coordinates rotate and the orientation index cycles; learned channel index C stays fixed.

References

- [1] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. 2021. arXiv:2104.13478.
- [2] T. Cohen and M. Welling. *Group Equivariant Convolutional Networks*. ICML, 2016. PMLR 48:2990–2999.
- [3] E. J. Bekkers. *An Introduction to Equivariant Convolutional Neural Networks for Continuous Groups*. Lecture notes, Section 2.3, 2021. UvA course notes.