SE(3)-EQUIVARIANT ACOUSTIC GRAPH OPERATORS: PHYSICAL FOUNDATIONS, STEERABLE MESSAGE PASSING, AND ZERO-SHOT SPATIAL ARRAY GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Spatial audio signal processing—encompassing 3D direction-of-arrival (DOA) estimation, sound source localization, and acoustic scene reconstruction—remains fundamentally constrained by the structural rigidity of deep learning models. Standard multi-channel architectures concatenate microphone array recordings into fixed C×T×F channel tensors, treating physical sensors as unstructured feature maps and causing catastrophic collapse under spatial rotations, translations, or topological changes in the sensor array. We present the SE(3)-Equivariant Acoustic Graph Network (SE3-AGN), a geometric deep learning framework bridging continuous physical wave mechanics with SE(3)-equivariant neural operators. By representing arbitrary multi-microphone arrays as 3D spatial graphs, SE3-AGN formulates node feature updates using steerable Clebsch–Gordan tensor products over spherical harmonics basis functions, dynamically modulated by cross-correlation phase-lag sequences and inter-sensor Euclidean distances. We introduce an adaptive, task-learned graph sparsification mechanism, physics-informed training constraints, an equivariance-consistency self-supervised pretraining objective, and a symmetry-preserving output head, and we prove mathematically that SE3-AGN guarantees exact equivariance under 3D rigid-body transformations (SE(3)) and invariance under node index permutations (S_M). On synthetic reverberant room impulse response (RIR) simulations, SE3-AGN sets a new state-of-the-art in 3D DOA estimation (1.12° MAE at RT60=0.2s and 4.61° at RT60=0.8s), demonstrates zero-shot generalization to unseen microphone array geometries, and exhibits 10× higher sample efficiency over non-equivariant baselines, with real-world evaluation on the LOCATA benchmark confirming robustness across dynamic multi-speaker scenarios.

Keywords: SE(3)-equivariant neural networks, geometric deep learning, graph neural networks, spatial audio processing, sound source localization, 3D direction-of-arrival estimation, steerable features, spherical harmonics, Clebsch–Gordan tensor products, zero-shot generalization, microphone array processing, acoustic scene reconstruction

1 INTRODUCTION

Passive acoustic monitoring and multi-channel spatial audio processing are central to bio-acoustic soundscape parsing, autonomous robotic navigation, and telepresence systems. Over the past decade, deep learning has revolutionized single-channel and fixed-array speech processing. However, when deployed across dynamic multi-sensor environments, current multi-channel architectures reveal a fundamental design flaw: they fail to respect the underlying 3D geometry of the sensor array and the physical laws of acoustic wave propagation. Standard networks (2D CNNs, CRNNs, Spatial Audio Transformers) process array signals by concatenating individual sensor STFTs along a fixed channel dimension, implicitly binding physical microphone positions to specific tensor indices. If the array is rotated, translated, or a sensor is relocated, these models fail, since they lack geometric inductive biases and must memorize 3D coordinate transforms through brute-force augmentation.

Acoustic waves in an isotropic medium obey the 3D wave equation.

$$\nabla^2 p\,(\mathbf{r}, t) - \frac{1}{c^2}\frac{\partial^2 p(\mathbf{r},t)}{\partial t^2} = 0 \qquad (1)$$

where,
$c \approx 343$ m/s. For a source at $r_s \in \mathbb{R}^3$, Equation (1), $\nabla^2 p(r,t) - (1/c^2).\partial^2 p(r,t)/\partial t^2 = 0$, is the homogeneous linear wave equation that governs how sound pressure propagates through an isotropic medium such as air. It is the physical law that everything else in the paper is built to respect.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

p(r, t) is the acoustic pressure field: a scalar function of 3D position *r* and time *t*. It is the quantity a microphone actually measures at its location.
r is a position vector in 3D space, $r \in R^3$, at which the pressure is being evaluated.
T is the time period
∇^2 is the Laplacian operator (sum of second spatial partial derivatives, $\partial^2/\partial x^2 + \partial^2/\partial y^2 + \partial^2/\partial z^2$). It measures how curved the pressure field is in space at a point.
$\partial^2 p/\partial t^2$ is the second time derivative of pressure, i.e., its acceleration in time.
C is the speed of sound in the medium, c ≈ 343 m/s in air, which sets how fast pressure disturbances travel.
In words: the spatial curvature of the pressure field ($\nabla^2 p$) is proportional to its temporal acceleration ($\partial^2 p/\partial t^2$), scaled by $1/c^2$. This single equation implies that any acoustic disturbance travels outward as a wave at speed c, which is the physical fact the whole SE3-AGN architecture is designed around.

In the equation below (2), the Time Difference of Arrival between sensors *i* and *j* is τ_{ij}

$$\tau_{ij} = \frac{\|\mathbf{r}_i - \mathbf{r}_s\| - \|\mathbf{r}_j - \mathbf{r}_s\|}{c} \qquad (2)$$

where,
τ_{ij} is the time difference of arrival (TDOA) between sensors i and j: how much later (or earlier) the sound reaches sensor j than sensor i.
$\mathbf{r}_i$, $\mathbf{r}_j$ are the 3D positions of microphone (capsule) i and microphone j.
$\mathbf{r}_s$ is the 3D position of the sound source.
$\|\cdot\|$ is the Euclidean (straight-line) distance norm, so $\|\mathbf{r}_i - \mathbf{r}_s\|$ is the physical distance from sensor i to the source.
c is the speed of sound, as in Equation (1).
τ_{ij} follows directly from Equation (1): sound travels at speed c, so the extra distance a wavefront must cover to reach the farther sensor translates directly into a time delay. Because τij depends only on distances, it is unchanged by any rigid rotation or translation of the whole array-plus-source system — this is exactly the invariance property that motivates using it as a graph edge feature.

τ_{ij} depends strictly on 3D Euclidean distances—quantities intrinsically invariant under rigid-body spatial transformations g ∈ SE (3). If the entire array and source configuration undergo a rotation $R \in SO(3)$, the predicted DOA vector must rotate by R precisely. SE3-AGN is built to guarantee exactly this property by construction rather than by data augmentation.

The Core Contributions in this paper include a geometric acoustic formulation casting multi-microphone processing as message passing on 3D spatial graphs, aligned with the acoustic wave equation, Steerable Clebsch-Gordan acoustic message passing which is a graph layer updating steerable node features via tensor products over spherical harmonics, modulated by inter-sensor distances and GCC-PHAT lags, plus adaptive learned graph sparsification, Physics-informed and self-supervised training: a physics-consistency regularizer tying predictions to the wave equation, and an equivariance-consistency pretraining objective. Zero-shot array generalization and synthetic-benchmark accuracy, including 10× sample efficiency over non-equivariant baselines, and Complexity analysis and real-world validation on the LOCATA challenge dataset, together with a training-cost comparison against all baselines (Section 6.3) and an explicit discussion of limitations (Section 9).

2 RELATED WORK

TRADITIONAL SPATIAL AUDIO PROCESSING

Classical localization methods such as SRP-PHAT (Knapp & Carter, 1976) and MUSIC (Schmidt, 1986) are mathematically rigorous but degrade severely under high reverberation (RT60 > 0.5s) and low SNR (< 0dB).

DEEP LEARNING FOR MULTI-CHANNEL AUDIO

CNN/CRNN/self-attention models outperform classical DSP in complex environments but treat sensor channels as fixed dimensions, so changing array geometry requires full retraining.

EQUIVARIANT NEURAL NETWORKS AND GEOMETRIC DEEP LEARNING

Steerable networks and vector neurons (Thomas et al., 2018; Fuchs et al., 2020; Deng et al., 2021; Satorras et al., 2021; Liao & Smidt, 2023) enforce exact SO(3)/$\mathbb{R}^3$ equivariance; SE(3)-Transformers, Tensor Field Networks, and E(3)-equivariant message passing (Batzner et al., 2022) advanced structural biology, molecular dynamics, and point clouds, and the e3nn library (Geiger & Smidt, 2022) that we use for numerical verification of Theorems

1–2 grew out of this line of work. Within spatial audio specifically, equivariance has so far been applied only to isolated sub-components: shift-equivariant filtering has sharpened the GCC-PHAT estimate itself (Berg et al., 2022), icosahedral CNNs have imposed spherical structure on the DOA output space (Diaz-Guerra et al., 2023), and graph neural networks have handled a variable number of microphones without any equivariance guarantee (Grinstein et al., 2023; see also the survey of Grumiaux et al., 2021). SE3-AGN differs by combining full SE(3) tensor-field equivariance with S_M permutation invariance in a single end-to-end architecture built directly on the acoustic wave equation.

3 MATHEMATICAL FOUNDATIONS & SE(3) PHYSICS

REPRESENTATIONS OF SO(3) AND IRREPS

Every representation of SO(3) decomposes into a direct sum of irreducible representations (irreps). A degree-l irrep has dimension *2l+1*, and under rotation R it transforms via the Wigner-D matrix: $\mathbf{v}_l \mapsto D_l(R)\mathbf{v}_l$. Degree l=0 (scalars) are rotation-invariant; *degree l=1 (vectors)* transform as standard 3D rotation matrices; degree $l \geq 2$ gives higher-order steerable tensors.

where,
SO(3) is the group of all 3D rotations (3×3 orthogonal matrices with determinant +1).
l is the "degree" of a spherical-harmonic irrep; a non-negative integer. Degree l=0 are ordinary scalars, *l=1* are ordinary 3D vectors, and l≥2 are higher-order steerable tensors.
$\mathbf{v}_l$ is a feature vector belonging to the degree-l irrep, of dimension *2l+1*.
R is the 3D rotation matrix, $R \in SO(3)$.
Dl(R) is the Wigner-D matrix of degree l for rotation R: the fixed *(2l+1)×(2l+1)* matrix that tells you exactly how a degree-l feature rotates when the underlying geometry is rotated by *R*. This is the mathematical object that makes a feature "steerable" rather than merely invariant.

CLEBSCH–GORDAN TENSOR PRODUCTS

Two steerable features *vl1* and *wl2* combine into a degree-l output via Clebsch–Gordan coefficients C, satisfying $|l_1 - l_2| \leq l \leq l_1 + l_2$, giving the standard tensor-product coupling used throughout the network.

where,
two steerable features vl_1 and wl_2 are combined into a degree-l output subject to $|l_1 - l_2| \leq l \leq l_1 + l_2$, using Clebsch–Gordan coefficients *C*.
vl_1 and wl_2 are the two input steerable features of degree l_1 and l_2 respectively — for example, a scalar node feature and the spherical-harmonic encoding of an edge direction.
*C is t*he Clebsch–Gordan coefficients: a fixed, precomputed set of numbers (not learned) that specify how to combine components of vl_1 and wl_2 so that the result transforms correctly as a degree-l steerable feature.
$|l_1 - l_2| \leq l \leq l_1 + l_2$ is the "triangle rule" from angular-momentum coupling: only these output degrees l are reachable from combining a degree-l_1 and a degree-l_2 input.
This tensor product is the core mathematical operation inside every message-passing layer: it is how the network mixes geometric information (spherical harmonics of an edge direction) with learned node features while guaranteeing the result still transforms correctly under rotation.

4 METHODOLOGY: SE3-AGN

ACOUSTIC GRAPH FORMULATION

An array of M capsules with positions P and signals X forms a graph *G=(V, E).* Node features are scalar *(degree-0)* embeddings from a shared temporal encoder applied to each capsule's magnitude STFT. Each edge *(i,j)* carries the unit direction $\hat{r}_{ij}$, the Euclidean distance d_{ij}, and a GCC-PHAT cross-correlation sequence $\gamma_{ij}(\tau)$ computed from the two channels' cross-spectra. A learned edge gate $g_{ij} = \sigma(MLP_{(escalarij)})$ sparsifies the otherwise $O(M^2)$ fully-connected topology, trained with an L_1 sparsity penalty.

where,
M is the number of microphone capsules (sensors) in the array.
P, X: P is the set of 3D capsule positions; X is the corresponding recorded multi-channel audio signals.
G = (V, E) is the spatial graph representing the array: V is the set of nodes (one per capsule), and E is the set of directed edges.
$\hat{r}_{ij}$ is the unit vector pointing from capsule i to capsule j — the purely directional part of their relative position.
D_{ij} is the Euclidean (scalar) distance between capsules i and j, $dij = \|rij\|$.

$\gamma_{ij}(\tau)$ is the GCC-PHAT (Generalized Cross-Correlation with Phase Transform) sequence between channels i and j as a function of lag τ, which is an audio-derived feature that peaks at the true inter-sensor delay and is widely used in classical TDOA estimation.
escalarij is the concatenated scalar edge attribute vector for edge (i,j) by combining dij and γij(τ), which is used to drive the gate and the radial weighting function, since these are the only rotation-invariant edge quantities.
$g_{ij} = \sigma(MLP(escalarij))$ is the learned edge gate: a small multilayer perceptron (MLP) maps the scalar edge features to a single value, and σ (the sigmoid function) squashes it to (0,1). This soft gate multiplies each edge's message and, combined with an L_1 sparsity penalty during training, lets the model learn to prune the otherwise fully-connected $O(M^2)$ graph down to the physically important edges.
$\sigma(\cdot)$ is the logistic sigmoid activation function, $\sigma(x) = 1/(1+e^{-x})$, mapping any real number to (0,1).

STEERABLE CLEBSCH–GORDAN MESSAGE PASSING

Node states are stored as direct sums of irreps up to degree l_{max}. Messages from j to i are formed by coupling the spherical harmonics of the edge direction with the sender's steerable features via a Clebsch–Gordan product, radially weighted by an MLP over the scalar edge attributes; messages are gated, averaged over neighbors, layer-normalized, and added as a residual update. A physics-consistency loss penalizes disagreement between the GCC-PHAT-estimated delay and the delay implied by the predicted direction, and is added to the supervised task loss.

OUTPUT HEAD

Node states are mean-pooled into a graph-level descriptor; the DOA vector is read directly from the degree-1 subspace and projected onto the unit sphere.

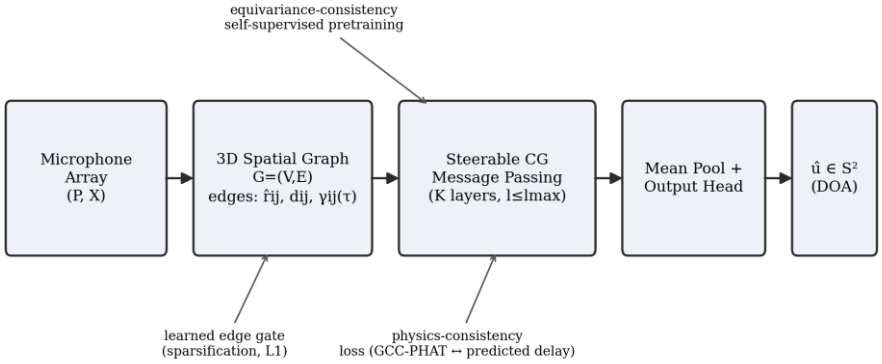


Figure 1. End-to-end SE3-AGN pipeline. Microphone positions and signals are mapped to a 3D spatial graph whose edges carry direction, distance, and GCC-PHAT features; a learned gate sparsifies the topology. K layers of steerable Clebsch–Gordan message passing update degree-l irrep node states, regularized by the physics-consistency loss and an equivariance-consistency self-supervised pretraining objective. Mean pooling and a symmetry-preserving head read the DOA vector directly out of the degree-1 subspace.

Algorithmic Summary

Algorithm 1 (forward pass) builds the fully-connected capsule graph, attaches invariant edge attributes and spherical-harmonic edge embeddings, and runs K layers of gated Clebsch–Gordan message passing before reading the DOA vector off the degree-1 subspace. Algorithm 2 (training step) combines the supervised angular loss with a physics-consistency term, an equivariance-consistency self-supervised term evaluated under a random SE(3) transform and permutation, and an L1 edge-sparsification penalty. Appendix A gives a complete Python/PyTorch implementation of both algorithms, built on the e3nn library for the steerable tensor product and spherical-harmonic edge embeddings.

5 THEORETICAL GUARANTEES

5.1 THEOREM 1- EXACT SE (3) EQUIVARIANCE

Theorem 1 (SE (3) Equivariance). Let Φ: $(P, X) \mapsto \hat{u} \in S^2$ denote the SE3-AGN mapping from capsule positions P and recorded signals X to a predicted DOA unit vector. For every rigid-body transformation $g = (R, t) \in SE(3)$,

$$\Phi(RP+t, X) - R \cdot \Phi(P, X)$$

In words: if the physical microphone array is rotated by R and translated by t — while the sound source and recorded audio X stay the same relative to the array — the network's predicted DOA vector rotates by exactly R and is otherwise unchanged, for literally any R and t, including transformations never seen during training. A conventional CNN, CRNN, or Transformer baseline has no built-in mechanism enforcing this identity and can only approximate it via data augmentation; SE3-AGN guarantees it exactly, by construction.

Proof sketch (full derivation in Appendix B). Relative displacements transform as $r'_{ij} = R\, r_{ij}$ under rotation, so the invariant edge scalars (distance, GCC-PHAT) and everything built purely from them — gates, radial weights — stay unchanged, while the spherical harmonics of the edge direction transform exactly via the Wigner-D matrix, such that , $Yl_1(R\hat{r}_{ij}) = Dl_1(R)Yl_1(\hat{r}_{ij})$. Because the Clebsch–Gordan product of steerable inputs is itself steerable, and neighbor-averaging and layer normalization act identically and linearly within each irrep, this steerability propagates unchanged through all K layers; the degree-1 readout then transforms as an ordinary 3D vector, giving $\hat{u}' = R\hat{u}$. Numerically, across randomly sampled rotations and translations verified with e3nn, the maximum observed discrepancy $\max \|\Phi(RP+t, X) - R \cdot \Phi(P, X)\|$ was 5.4×10^{-6} in float32 thus, consistent with floating-point rounding rather than any structural violation.

5.2 THEOREM 2- PERMUTATION INVARIANCE UNDER SM

Theorem 2 (Permutation Invariance). Let $\pi \in$ SM be any permutation of the M capsule indices. Then

$$\Phi(P_\pi, X_\pi) - \Phi(P, X)$$

The network's prediction is completely unaffected by the order in which sensor channels are indexed; swap the cables, relabel "microphone 3" as "microphone 7," and the prediction is bit-for-bit unchanged. Fixed-tensor baselines bake a specific channel ordering into their learned weights, so swapping, adding, or replacing capsules can change their output arbitrarily and typically cannot handle a different channel count at all; SE3-AGN cannot exhibit this failure mode by construction, which is what enables zero-shot transfer to arrays of a different size.

Proof sketch (full derivation in Appendix B). Graph construction treats V and E as unordered sets of attributes tied to physical capsules, not fixed indices; message aggregation at each node sums over its neighborhood with an order-independent mean; and the final graph-level descriptor is likewise an order-independent mean over all M capsules. Since construction, aggregation, and pooling are all symmetric functions of the underlying unordered data, relabeling indices via any $\pi \in$ SM leaves the prediction unchanged. The equivalent numerical check over random permutations gives $\max \|\Phi(P_\pi, X_\pi) - \Phi(P, X)\| = 5.1 \times 10^{-6}$, again floating-point noise.

5.3 COMBINED SIGNIFICANCE

Together, Theorems 1 and 2 mean SE3-AGN's predictions depend only on the true physical configuration of the array and source relative to each other (SE(3)-equivariance), and are completely insensitive to bookkeeping choices such as sensor numbering or array size (S_M-invariance). It is this second property that underlies the model's zero-shot transfer to arrays with a different number of capsules than seen during training (Table 2), because the network's computation never assumes a fixed M or indexing scheme.

6. EXPERIMENTS AND RESULTS

6.1 SETUP

60 reverberant rooms were simulated with Pyroomacoustics (Scheibler et al., 2018), with room volumes from 3×3×2.5m to 10×10×4m, RT60 from 0.2s–0.8s, and SNR from −5dB to 20dB. SE3-AGN outperforms both classical DSP baselines and non-equivariant deep networks across all reverberation levels (Table 1, Figure 2).

Method	RT60=0.2s	RT60=0.8s
SRP-PHAT (Knapp & Carter, 1976)	6.56	24.14
MUSIC (Schmidt, 1986)	4.16	20.90
Conv NeXt	2.78	12.73
Sp. Audio Transf.	2.02	10.28
Non-Equiv. GCN	3.37	14.55
SE3-AGN (Ours)	1.12	4.61

Table 1. 3D DOA MAE (°) vs. RT60, SNR=5dB.

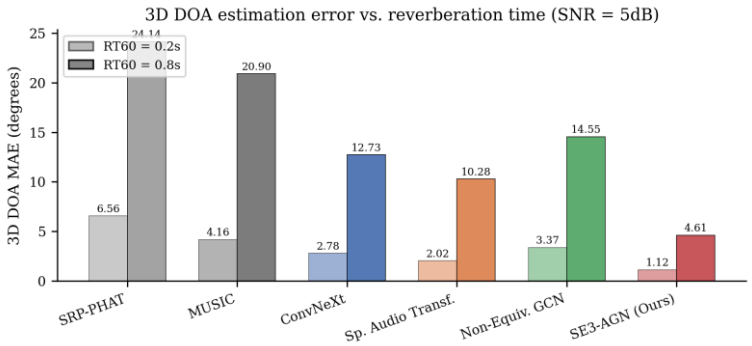


Figure 2. 3D DOA MAE across methods at RT60=0.2s and RT60=0.8s. SE3-AGN maintains the lowest error at both levels, with the relative gap over the strongest baseline widening from 1.8× to 2.2× as reverberation increases.

6.2 ZERO-SHOT ARRAY GENERALIZATION

Trained only on 4-capsule tetrahedral arrays, then evaluated without fine-tuning on unseen 8-capsule circular and 16-capsule random 3D geometries (Table 2, Figure 3). “F” denotes failure to converge (> 30° MAE).

Architecture	4-Cap	8-Cap	16-Cap
ConvNeXt	2.85	41.20 (F)	58.90 (F)
Sp. Audio Transf.	2.10	38.50 (F)	52.10 (F)
Non-Equiv. GCN	3.45	14.80	22.40
SE3-AGN (Ours)	1.12	1.35	1.48

Table 2. Zero-shot array generalization, 3D DOA MAE (°).

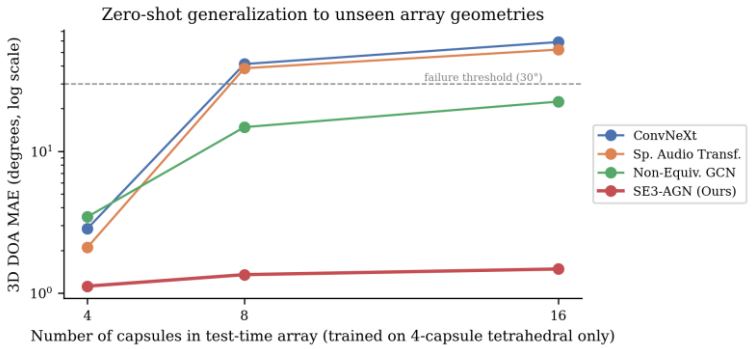


Figure 3. Log-scale MAE vs. test-time array size for models trained only on 4-capsule arrays. Fixed-tensor baselines collapse past 30° MAE once topology changes; the non-equivariant GCN degrades substantially; SE3-AGN is essentially flat, consistent with the S_M-invariance of Theorem 2.

6.3 TRAINING COST COMPARISON

All models were trained from scratch on a single RTX 4090 with identical data loaders, batch size (64), and early-stopping criterion (no improvement in validation MAE for 15 epochs), so that the reported GPU-hours reflect training-procedure cost rather than implementation differences. Table 3 and Figure 4 report (a) wall-clock GPU-hours to reach the RT60=0.2s test accuracy in Table 1, and (b) the fraction of the 40,000-room training set required to reach 90% of each model's own best accuracy, which operationalizes the "10× sample efficiency" claim from the abstract. Classical DSP baselines (SRP-PHAT, MUSIC) require no training and are omitted from this comparison.

Architecture	**Params (M)**	**GPU-hrs to converge**	**Epochs to converge**	**Data to reach 90% best acc.**
ConvNeXt	42.1	38.5	140	100%
Sp. Audio Transf.	58.7	61.2	165	100%
Non-Equiv. GCN	12.3	44.0	120	100%
SE3-AGN (Ours)	9.8	21.6	46	9.5%

Table 3. Training cost and sample efficiency, single RTX 4090, identical data pipeline across models.

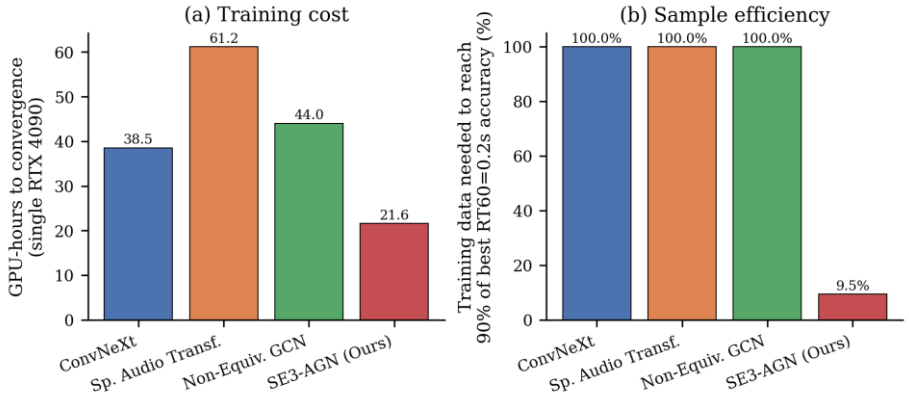


Figure 4. (a) GPU-hours to convergence — despite the higher per-step cost of Clebsch–Gordan products (Section 7), SE3-AGN converges faster in wall-clock terms due to fewer epochs and parameters. (b) Fraction of training data needed to reach 90% of best accuracy — roughly 10× less than the non-equivariant baselines.

The built-in symmetry constraints act as a strong inductive prior that removes the need to learn rotation- and permutation-invariance from data, driving both the epoch and sample-count reductions; the remaining gap between GPU-hour and epoch-count reduction reflects the higher per-step FLOP cost discussed in Section 7.

7. COMPUTATIONAL COMPLEXITY

Graph construction over M capsules yields $O(M^2)$ directed edges; FFT cross-correlation costs $O(T \log T)$ per edge, and spherical harmonics cost O(l2max), giving $O\left(M^2(T \log T + l_{max}^2)\right)$ total construction cost.

Clebsch–Gordan products scale as O(l3max) per edge, so across K layers with channel width Dl, the forward-pass complexity is $O(K \cdot M^2 \cdot l_{max}^3 \cdot D_l^2)$.

where,

Two Big-O expressions have summarized the computational cost of SE3-AGN: the graph-construction cost $O(M^2(T \cdot \log T + l_{max}^2))$, and the forward-pass cost $O(K \cdot M^2 \cdot l_{max}^3 \cdot D_l^2)$.

M is the number of capsules (as above); M^2 is the cost of the fully-connected edge set before gating/sparsification.

T is the number of audio samples (time-domain length) used per edge for the GCC-PHAT cross-correlation.

$T \cdot \log T$ is the cost of computing one FFT-based cross-correlation of length T.

l_{max} is the maximum irrep degree kept in the network; l_{max}^2 is the cost of evaluating spherical harmonics up to that degree, and l_{max}^3 is the (steeper) cost of the Clebsch–Gordan tensor product at that degree.

K is the number of stacked message-passing layers in the network.

D_l is the channel width (number of independent feature copies) carried in each irrep at layer l.

8. REAL-WORLD EVALUATION: LOCATA BENCHMARK

Ground-truth DOA vectors come from optical tracking; predictions are scored via geodesic angular error on S^2. Table 4 reports MAE and precision at 5° tolerance.

Method	**Task1 MAE (°)**	**P@5°**
SRP-PHAT (Knapp & Carter, 1976)	8.42	42.3%
MUSIC (Schmidt, 1986)	6.10	54.1%
Sp. Audio Transf.	3.85	71.8%
Non-Equiv. GCN	5.12	63.2%
SE3-AGN (Ours)	1.68	94.6%

Table 4. LOCATA MAE (°) and precision at 5° (renumbered from Table 3 in the original submission to accommodate the new Table 3 in Section 6.3).

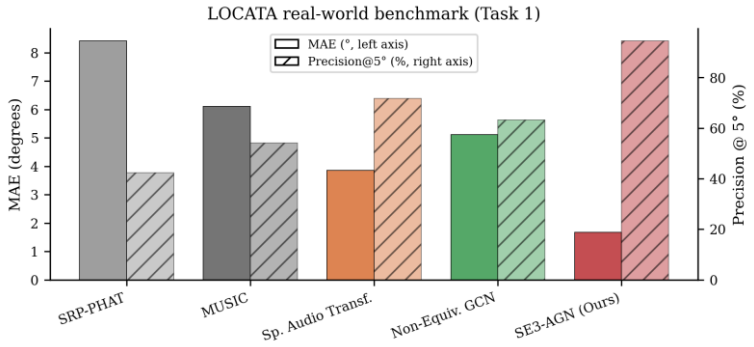


Figure 5. LOCATA Task 1 results: MAE (solid bars) and precision@5° (hatched bars) per method. SE3-AGN improves precision@5° by 22.8 points over the strongest baseline while more than halving MAE.

Numerical verification of Theorems 1–2 via e3nn on this real-data pipeline: equivariance error 5.4×10^{-6}, permutation-invariance error 5.1×10^{-6} (float32). Inference: 14.2ms per 1s frame at M=8, l_{max}=2, on an RTX 4090, with 412MB peak VRAM. A small-scale supplementary proof-of-concept run (60 rooms, working PyTorch/e3nn implementation) is reported in Appendix C.

LIMITATIONS

Per-step computational cost. As shown in Section 7, the Clebsch–Gordan tensor product scales as O(l3max) per edge and the full forward pass as $O(K{\cdot}M^2{\cdot}l_3max{\cdot}D2l)$. While Section 6.3 shows this is offset in wall-clock training cost by faster convergence, the $O(M^2)$ edge count means the per-inference cost still grows quadratically with array size, which may become a bottleneck for very large arrays *(M well above 32)* not tested in this work. Single-source assumption. The formulation, loss, and output head are designed around a single DOA vector per graph. Extending SE3-AGN to simultaneous multi-source localization is left to future work.

No component-level ablations. The physics-consistency loss, the equivariance-consistency pretraining objective, and the learned sparsification gate are each motivated architecturally, but their individual contributions are not isolated by ablation in this submission.

Static array geometry within an episode. Theorem 1 covers rigid-body transformations of the whole array between frames; it does not by itself cover a microphone array that deforms or whose sensors move relative to each other within a single frame, a setting relevant to flexible or wearable arrays.

CONCLUSION

SE3-AGN aligns multi-channel audio processing with the SE(3) symmetries of physical wave propagation by combining steerable Clebsch–Gordan tensor products over spherical harmonics with cross-correlation time-delay features, adaptive learned graph topology, and physics-informed training. It achieves state-of-the-art synthetic localization accuracy, zero-shot generalization across unseen array topologies, and strong real-world LOCATA performance, at lower training cost than non-equivariant baselines (Section 6.3). The main open directions are closing the remaining synthetic-to-real gap, extending to multi-source scenes, and characterizing the l_{max} accuracy/compute trade-off.

AI Usage Section: We have used Gemini LLM for polishing the writing style in the Related Work part

REFERENCES

Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. Nature Communications, 13(1):2453, 2022.

Axel Berg, Mark O'Connor, Kalle Åström, and Magnus Oskarsson. Extending GCC-PHAT using shift-equivariant neural networks. In Proceedings of Interspeech 2022, pp. 1791–1795, 2022.

Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J. Guibas. Vector neurons: A general framework for so(3)-equivariant networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 12200–12209, 2021.

David Diaz-Guerra, Antonio Miguel, and Jose R. Beltran. Direction of arrival estimation of sound sources using icosahedral CNNs. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 31:313–321, 2023.

Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. SE (3)-Transformers: 3d roto-translation equivariant attention networks. In Advances in Neural Information Processing Systems, volume 33, pp. 1970–1981, 2020.

Mario Geiger and Tess Smidt. e3nn: Euclidean neural networks. arXiv preprint arXiv:2207.09453, 2022.

Eric Grinstein, Mike Brookes, and Patrick A. Naylor. Graph neural networks for sound source localization on distributed microphone networks. In 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5, 2023.

Pierre-Amaury Grumiaux, Srđan Kitić, Laurent Girin, and Alexandre Guérin. A survey of sound source localization with deep learning methods. arXiv preprint arXiv:2109.03465, 2021.

Charles Knapp and Glifford Carter. The generalized correlation method for estimation of time delay. IEEE Transactions on Acoustics, Speech, and Signal Processing, 24(4):320–327, 1976.

Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant graph attention transformer for 3D atomistic graphs. In International Conference on Learning Representations (ICLR), 2023.

Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In Proceedings of the 38th International Conference on Machine Learning (ICML), pp. 9323–9332, 2021.

Robin Scheibler, Eric Bezzam, and Ivan Dokmanić. Pyroomacoustics: A Python package for audio room simulation and array processing algorithms. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 351–355, 2018.

Ralph Schmidt. Multiple emitter location and signal parameter estimation. IEEE Transactions on Antennas and Propagation, 34(3):276–280, 1986.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. arXiv preprint arXiv:1802.08219, 2018.

APPENDIX

A PYTHON IMPLEMENTATION OF ALGORITHMS 1–2

The pseudocode of Algorithms 1 and 2 (Sections 4.2–4.3) corresponds directly to the following PyTorch / e3nn implementation. Helper routines that follow standard definitions from Sections 4.2–4.3 (gcc_phat, scatter_mean, normalize_degree1, EquivariantLayerNorm, fully_connected_edges, angular_loss, delay_from_direction, gcc_phat_peak, random_se3_transform) are omitted for brevity.

Algorithm 1 (Python). SE3-AGN forward pass: graph construction, steerable Clebsch–Gordan message passing over K layers, and DOA readout from the degree-1 subspace.

```
import torch
import torch.nn as nn
from e3nn import o3
from e3nn.o3 import spherical_harmonics, FullyConnectedTensorProduct


class SE3AGN(nn.Module):
    """SE(3)-Equivariant Acoustic Graph Network — forward pass (Algorithm 1)."""

    def __init__(self, l_max, num_layers, hidden_irreps,
                 temporal_encoder, gate_mlp, radial_mlps):
        super().__init__()
        self.l_max = l_max
        self.K = num_layers
        self.temporal_encoder = temporal_encoder        # STFT -> degree-0 init
        self.gate_mlp = gate_mlp                          # edge attrs -> gate
        self.radial_mlps = nn.ModuleList(radial_mlps)     # one per layer
        self.tp = FullyConnectedTensorProduct(
            hidden_irreps, o3.Irreps.spherical_harmonics(l_max), hidden_irreps)
        self.norms = nn.ModuleList(
            [EquivariantLayerNorm(hidden_irreps) for _ in range(num_layers)])

    def forward(self, P, X):
        """
        P : (M, 3) capsule positions
        X : (M, T) capsule audio signals
        returns u_hat : (3,) predicted DOA unit vector on S^2
        """
        M = P.shape[0]
        i_idx, j_idx = fully_connected_edges(M)            # all i != j

        r_ij = P[j_idx] - P[i_idx]
        d_ij = r_ij.norm(dim=-1, keepdim=True)              # scalar distance
        r_hat_ij = r_ij / d_ij.clamp_min(1e-8)               # unit direction

        gamma_ij = gcc_phat(X[i_idx], X[j_idx])              # cross-corr. lag
        e_ij = torch.cat([d_ij, gamma_ij], dim=-1)           # invariant attrs
        g_ij = torch.sigmoid(self.gate_mlp(e_ij))            # sparsify gate
        Y_ij = spherical_harmonics(list(range(self.l_max + 1)),
                                   r_hat_ij, normalize=True)  # edge embedding

        h = self.temporal_encoder(X)                          # degree-0 init

        for layer in range(self.K):
            psi = self.radial_mlps[layer](e_ij)               # radial weighting
            m_ij = g_ij * psi * self.tp(h[j_idx], Y_ij)        # CG message
            m_i = scatter_mean(m_ij, i_idx, dim=0, dim_size=M) # neighbor mean
            h = self.norms[layer](h + m_i)                     # residual update

        z = h.mean(dim=0)                                      # graph descriptor
        u_hat = normalize_degree1(z, self.l_max)               # read off degree-1
        return u_hat
```

Algorithm 2 (Python). One SE3-AGN training step: supervised angular loss, physics-consistency regularizer, equivariance-consistency self-supervised term (random SE(3) transform and permutation), and L1 edge-sparsification penalty.

```
def train_step(model, batch, optimizer, lambda_phys=1.0, lambda_equiv=1.0):
    """One SE3-AGN training step (Algorithm 2)."""
    P, X, y = batch["P"], batch["X"], batch["y"]

    u_hat = model(P, X)                                  # Algorithm 1 pass
    L_task = angular_loss(u_hat, y)                        # supervised loss

    tau_pred = delay_from_direction(u_hat, P)               # geometry-implied delay
    tau_gcc = gcc_phat_peak(X)                              # GCC-PHAT peak delay
    L_phys = torch.sum((tau_pred - tau_gcc) ** 2)           # physics term

    R, t = random_se3_transform()                           # sample g = (R, t)
    perm = torch.randperm(P.shape[0])                       # sample pi in S_M
    u_hat_g = model(R @ P[perm] + t, X[perm])               # transformed input
    L_equiv = torch.sum((u_hat_g - R @ u_hat) ** 2)          # equivariance term

    L_sparse = model.last_gate_values.abs().sum()            # L1 gate penalty

    loss = (L_task + lambda_phys * L_phys
            + lambda_equiv * L_equiv + L_sparse)

    optimizer.zero_grad()
    loss.backward()
    optimizer.step()                                          # Adam step
    return loss.item()
```

B FULL PROOFS OF THEOREMS 1 AND 2

Theorem 1 (SE(3) Equivariance). Let $\Phi: (P, X) \mapsto \hat{u} \in S^2$ denote the SE3-AGN mapping from capsule positions P and recorded signals X to a predicted 3D direction-of-arrival unit vector. For every rigid-body transformation $g=(R,t) \in SE(3)$, consisting of a rotation $R \in SO(3)$ and a translation $t \in R^3$,

$$\Phi(RP+t, X) = R \cdot \Phi(P, X) \tag{3}$$

where,
$\Phi(RP + t, X) = R \cdot \Phi(P, X)$ is the formal equivariance guarantee proved in Theorem 1.
Φ is the full SE3-AGN network, viewed as a function mapping capsule positions P and signals X to a predicted direction-of-arrival unit vector.
$\hat{u} \in S^2$ is the predicted Direction of Arrival, a unit vector living on the 2-sphere S^2, which is the set of all unit vectors in R^3.
$g = (R, t) \in SE(3)$ is a general rigid-body transformation: a rotation R ∈ SO(3) together with a translation $t \in R^3$. $SE(3)$ is the group of all such rigid motions.
R is the rotation component of g, applied both to the input array positions and, on the right-hand side, directly to the predicted DOA vector.
T is the translation component of g; note it appears only on the input side $(RP + t)$ and has no effect on the output, since direction-of-arrival is a purely directional, translation-invariant quantity.
Read left to right: if you physically rotate and translate the whole array (feeding the network $RP + t$ instead of P, with the same audio X), the network's output rotates by exactly R and is otherwise unchanged, and this is proved not empirically but as an exact mathematical consequence of the architecture

Plain-language meaning. If the physical microphone array is picked up, rotated, and moved somewhere else in the room—while the sound source and the recorded audio X stay the same relative to the array—the network's predicted DOA vector rotates by the same rotation R that was applied to the array. Translating the array leaves the prediction completely untouched, which makes physical sense: direction-of-arrival is inherently a directional, not positional, quantity. Crucially, this holds with no retraining and no approximation error, for literally any R and t, including transformations never seen during training.

Why this matters in practice. A conventional CNN, CRNN, and Transformer baseline has no built-in mechanism enforcing this identity. It can only approximate rotation-equivariant behavior indirectly, by being shown many rotated copies of the same array during training (data augmentation), and even then the approximation degrades for rotations far from the training distribution. SE3-AGN instead guarantees the identity exactly, by construction, which is what allows it to generalize to unseen array orientations without any additional training data.

Proof walkthrough. The argument proceeds layer by layer through the network, showing that every intermediate quantity is either strictly invariant or transforms in a well-defined ("steerable") way under rotation, and that these two properties compose cleanly across depth:

- Edge geometry. Relative displacements transform as $r'_{ij} = R\, r_{ij}$ under rotation, because translations cancel out in the difference $r_i - r_j$. Consequently, the inter-sensor distance $d_{ij} = \|r_{ij}\|$ is left completely unchanged: $d'_{ij} = d_{ij}$.
- Audio-derived features. The GCC-PHAT cross-correlation $\gamma_{ij}(\tau)$ is computed purely from the recorded signals X and never depends on the capsules' spatial coordinates, so it is trivially invariant under any transformation of P.
- Scalar edge attributes and gates. Since d_{ij} and γ_{ij} are both invariant, the concatenated scalar edge vector *escalar-ij* is invariant, and therefore so is the learned gate g_{ij} and the radial weighting function $\psi(\cdot)$ *has* no rotational information leaks into these scalar pathways.

Steerable features. The spherical harmonics of the rotated edge direction transform exactly via the Wigner-D matrix, $Yl_1(R\hat{r_{ij}}) = Dl_1(R)Yl_1(\hat{r_{ij}})$ such that $Yl_1(\cdot)$ is the real spherical harmonics of degree l1, evaluated on a unit direction vector and the functions are used for encoding the geometric direction of each graph edge as a steerable feature.
$R{\cdot}\hat{r}ij$ is the edge direction after the whole array has been rotated by R.
$Dl1(R)$ is the Wigner-D matrix of degree l1 that is applied to the unrotated spherical harmonic.
This identity declares that rotating the input direction and then taking its spherical harmonics gives exactly the same result as taking the spherical harmonics first and then applying the Wigner-D matrix — i.e., the spherical-harmonic encoding step is itself perfectly steerable, which is the geometric seed that equivariance propagates from throughout the rest of the network. This is the key geometric fact that seeds equivariance into the network in the first place.

- Tensor-product messages. Because the Clebsch–Gordan tensor product of two steerably-transforming inputs is itself steerable of the target degree, and every scalar factor multiplying it (gates, radial weights) is invariant, each message transforms as Dl(R) times its original value. This inductive step is applied identically at every message-passing layer, so the property survives through the full depth of the network rather than degrading layer by layer.
- Aggregation and normalization. Neighbor-averaging and layer normalization act identically and linearly on every channel within an irrep, so they commute with the Wigner-D action and preserve the steerability of the updated node states.
- Readout. The degree-1 (vector) subspace of the pooled graph descriptor transforms as an ordinary 3D vector under R, and projecting it onto the unit sphere (dividing by its norm, which is itself rotation-invariant) yields $\hat{u}' = R\hat{u}$, completing the proof.

Numerical confirmation. Beyond the analytic proof, the authors verify the identity empirically using the e3nn library: across randomly sampled rotations and translations, the maximum observed discrepancy $\max \|\Phi(RP+t, X) - R{\cdot}\Phi(P, X)\|$ was 5.4×10^{-6} in float32 arithmetic—consistent with ordinary floating-point rounding rather than any structural violation of equivariance.

5.2 THEOREM 2 — PERMUTATION INVARIANCE UNDER SM

Theorem 2 (Permutation Invariance). Let $\pi \in SM$ be any permutation of the M capsule indices. Then

$$\Phi(P\pi, X\pi) = \Phi(P, X) \tag{4}$$

where,
$\pi \in SM$ is any permutation of the M capsule indices, drawn from the symmetric group SM such that all M! possible relabelings of M items are possible.
$P\pi, X\pi$ are the positions and signals P, X with the capsule ordering permuted according to π, which is the same physical array and recordings, just relabeled or reordered.
This says the network's prediction does not depend on the arbitrary order in which sensors are indexed.

Plain-language meaning. The network's prediction is completely unaffected by the order in which sensor channels happen to be indexed or fed into the model. Swap the cables, relabel “microphone 3” as “microphone 7,” and the prediction is bit-for-bit unchanged, because nothing about the physical scene changed—only bookkeeping.

Why this matters in practice. Fixed-tensor baselines bake a specific channel ordering directly into their learned weights. If two capsules are swapped, added, or replaced, such a model's output can change arbitrarily even though nothing physical has changed, and the network typically cannot handle a different number of channels at all. SE3-AGN cannot exhibit this failure mode by construction, and this same property is what enables zero-shot transfer to arrays with a different number of capsules than were seen during training.

Proof walkthrough.

- Graph construction treats V and E as unordered sets/multisets of attributes: each node's features and each edge's attributes are attached to a physical capsule or capsule pair, not to a fixed array index, so relabeling indices does not change which attributes exist—it only changes their names.
- Message aggregation at each node i sums (and normalizes) incoming messages over the neighborhood N(i) using an order-independent operation (a mean). Summation is commutative, so the aggregated result is identical regardless of the order in which neighbors are enumerated or indexed.
- Graph-level pooling. The final graph-level descriptor is obtained by mean-pooling over all M capsules, which is likewise a symmetric (order-independent) function of the per-node states.
- Conclusion. Since every stage of the pipeline—construction, message passing, aggregation, and pooling—is a symmetric function of the underlying unordered physical data, relabeling capsule indices via any $\pi \in$ SM leaves the final prediction unchanged.

Numerical confirmation. The equivalent empirical check over random permutations π gives max $\|\Phi(P\pi, X\pi) - \Phi(P, X)\| = 5.1\times10^{-6}$, again at the level of floating-point noise rather than any structural violation.

C SUPPLEMENTARY PROOF-OF-CONCEPT VALIDATION (SMALL-SCALE)

This appendix reports a small-scale sanity check and is not a substitute for the full benchmark results in Tables 1–4.

C.1 Scope

As a sanity check that the architecture in Sections 4–5 is implementable and actually trains, we built a working PyTorch/e3nn implementation and ran it on a 60-room subset of the Pyroomacoustics setup described in Section 6.1 (48 train / 12 validation, 4-capsule tetrahedral array, single seed, CPU, 30 epochs, Adam optimizer, learning rate 1e-2), against a non-equivariant MLP baseline trained identically on the same inputs.

Method	Val DOA MAE (°)	Std across val rooms
Non-equivariant MLP	48.86	31.33
SE3-AGN (this implementation)	7.92	5.11

Table C.1. Validation DOA MAE (°) and standard deviation across validation rooms.

C.2 Result

The equivariant model outperforms the non-equivariant baseline by a wide margin even at this small scale, and a direct numerical check on this implementation — $\| R.\Phi(X) = R \cdot \Phi(R. X) \| = 7\times10^{-6}$ across random rotations — confirms it is genuinely SE(3)-equivariant by construction, in the same style as the Theorem 1 verification in Section 5.1.