Cross-Domain Swarm Intelligence Transfer via Universal Sensor Ontology and Experience Embeddings

Draft --- February 2026

Authors: Ivan Labra - axelotl partners

Abstract

We present a framework for transferring swarm navigation intelligence across domains without model training. Augmented reality creatures in a consumer application (Rabble.world) generate flight telemetry that is structurally equivalent to drone swarm telemetry when mapped through the W3C Sensor, Observation, Sample, and Actuator (SOSA) ontology and the Onto4MAT multi-agent teaming ontology. By embedding both AR creature flights and physical drone trajectories in a shared vector space, we enable drones to exploit AR-derived navigation patterns through nearest-neighbor lookup at runtime --- replacing conventional training pipelines with an ever-growing experience table. We describe the ontological bridge (Onto4MAT to SOSA), the consent architecture (opt-in data asset rights grounded in the Agent Knowledge Protocol), the economic distribution model for compensating data contributors, and a validation protocol for measuring cross-domain transfer fidelity. We argue that communities generating AR flight data in their local terrain are, in effect, building navigational intelligence for autonomous systems that will operate in those same environments --- a form of distributed resilience building with aligned economic incentives.

---

1. Introduction

1.1 The Training Problem

Autonomous drone swarms require navigation policies that generalize across environments: urban canyons, forested terrain, smoke-filled corridors during wildfire response. Conventional approaches --- reinforcement learning in simulation, imitation learning from expert pilots, transfer learning from pre-trained vision models --- share a common limitation: they require dedicated training infrastructure (GPU clusters, simulation environments, expert demonstrations) and produce fixed models that must be retrained as environments change.

We propose an alternative: **experience-based intelligence**, where navigation patterns are stored as embedding vectors paired with situational context, and runtime decision-making reduces to nearest-neighbor lookup in this experience space. No training pipeline. No GPU

at inference time. The experience table grows continuously as new observations are added, and each new entry immediately enriches the space.

1.2 The Data Problem

Even if experience-based intelligence is architecturally sound, it requires large volumes of diverse navigation telemetry across varied environments. Physical drone flights are expensive, require regulatory approval, and are constrained by battery life, weather, and safety margins. Simulated flights lack the complexity of real environments. Neither source scales cheaply.

We observe that a large and growing population is already generating navigation telemetry at scale: users of augmented reality applications who guide digital creatures through physical environments. A person walking through a park while guiding an AR butterfly records GPS breadcrumbs --- position, heading, timestamp --- that constitute a navigation trajectory through real-world terrain. When multiple users gather in the same location (a "rabble"), their collective trajectories exhibit emergent swarm dynamics: separation maintenance, alignment convergence, cohesion around anchor points.

The question is whether this AR-derived telemetry is *structurally transferable* to physical drone navigation. We argue that it is, provided the data is mapped through a shared ontological framework that normalizes domain-specific differences and preserves structural relationships.

1.3 Contributions

This paper makes five contributions:

1. **An ontological bridge** from AR creature telemetry to drone swarm telemetry via SOSA (W3C SSN/SOSA) and Onto4MAT, demonstrating that both domains emit the same observable properties when properly mapped.

2. **A scale-invariance argument** showing that normalized swarm metrics (alignment, cohesion, separation) are domain-independent --- a swarm of AR creatures at 1.5 m/s with 3m separation produces the same embedding as a drone swarm at 15 m/s with 30m separation.

3. **An experience-based runtime architecture** where devices perform nearest-neighbor lookup against an embedding table instead of executing trained neural network weights, enabling deployment on microcontrollers without GPU inference capability.

4. **A consent-gated data asset rights model** where data contributors (AR creature owners) opt in per-creature, retain ownership of their observations, and receive economic returns when their contributed patterns are consumed downstream.

5. **A validation protocol** for measuring cross-domain transfer fidelity between AR-derived and drone-derived embeddings.

---

2. Background

2.1 Onto4MAT: Ontology for Multi-Agent Teaming

Onto4MAT (Kiesel et al., 2022) is an ontology for describing multi-agent swarm behavior in shepherding scenarios. It defines the following core classes:

- **Agent**: An individual entity with data properties `hasSpeed` (m/s), `hasHeading` (degrees), `hasEnergy` (0--1), `hasDistanceToGoal` (m), `xLocation`, `yLocation`. Agents are typed as `Bio` or `Artificial`.
- **Team**: A group of agents with collective properties `hasAlignmentWithTeam` (0--1), `hasCohesionWithTeam` (0--1), `hasTeamSeparation` (m), `hasInfluence` (0--1).
- **Formation**: A spatial configuration --- `Arc`, `Echelon`, `Line`, `V-Formation`, `Wedge` --- with geometric parameters.
- **Mission**: Decomposed into `Goal` and `Task`, with individual actions (`Accelerate`, `AvoidCollision`, `Decelerate`, `Hover`, `MaintainFormation`, `PathFollow`, `Turn`) and team actions (`Converge`, `Disperse`, `Encircle`, `FormUp`, `Herd`, `Patrol`, `Search`, `Surround`).
- **Temperament**: `Aggressive`, `Defensive`, `Passive` --- behavioral profiles that affect action selection.

Onto4MAT was designed for shepherding robotics but its vocabulary is general enough to describe any multi-agent system with spatial coordination.

2.2 W3C SSN/SOSA

The Semantic Sensor Network (SSN) ontology and its lightweight core, SOSA (Sensor, Observation, Sample, and Actuator), provide a domain-agnostic vocabulary for describing sensors, observations, and actuations (Haller et al., 2017). Key classes:

- **sosa:Platform**: A physical or virtual entity hosting sensors (a drone, a phone, an AR creature).
- **sosa:Sensor**: A device or computational process that observes a property.
- **sosa:Observation**: An act of observing, linking a sensor to an observed property, a feature of interest, a result value, and a timestamp.
- **sosa:ObservableProperty**: A quality that can be observed (speed, heading, energy, temperature, soil moisture).
- **sosa:FeatureOfInterest**: The entity being observed (a specific species, a terrain segment, an airspace).

SOSA's power is its generality: a drone's IMU reporting heading and a phone's GPS reporting heading produce observations with the same structure. The observable property is `hasHeading`; the platforms differ; the observation semantics are identical.

2.3 Reynolds Flocking Rules

Reynolds (1987) identified three local rules sufficient for emergent flocking behavior:

1. **Separation**: Steer to avoid crowding local flockmates.
2. **Alignment**: Steer toward the average heading of local flockmates.
3. **Cohesion**: Steer toward the average position of local flockmates.

These rules produce complex collective behavior from purely local interactions. Critically, they are **scale-invariant**: the rules operate on relative distances and angles, not absolute velocities or positions. A flock of birds at 20 m/s and a swarm of drones at 2 m/s exhibit the same formation dynamics when separation, alignment, and cohesion are measured as normalized ratios.

Onto4MAT encodes these rules as team properties: `hasTeamSeparation`, `hasAlignmentWithTeam`, `hasCohesionWithTeam`. Our claim is that these properties, when derived from AR creature swarms, occupy the same region of embedding space as when derived from physical drone swarms.

2.4 Active Dream Memory

The Agent Bestiary World (ABW) platform implements a biologically-inspired episodic memory system called Active Dream Memory (ADM). Agent executions produce episodes --- structured records containing the query, context, response, and a 1024-dimensional embedding vector. During consolidation cycles (analogous to sleep), episodes are clustered, patterns are extracted as semantic rules, entities and facts are identified, and the agent's knowledge graph evolves (Labra, 2026a).

This consolidation process is critical to the cross-domain transfer mechanism: raw telemetry observations from heterogeneous sources are unified through agent interpretation into a shared embedding space where structural similarity --- not domain identity --- determines proximity.

2.5 Agent Knowledge Protocol

The Agent Knowledge Protocol (AKP) defines the consent, economic, and governance framework for knowledge sharing between agents (Labra, 2026b). Key principles:

- **Opt-in participation**: No data flows without explicit consent from the data owner.
- **Property rights**: Data contributors retain ownership of their observations and derivative value.
- **Beckstrom's Law as success metric**: Network value = sum(transaction value) - sum(participation cost). The network is only valuable if every participant gains more than it costs them.
- **Anti-monopoly monitoring**: Scale-free topology (power-law wealth concentration) is explicitly monitored and counteracted.

AKP provides the ethical and economic scaffolding for the data asset rights model described in Section 5.

---

3. The Ontological Bridge

3.1 Mapping AR Creature Telemetry to SOSA

A Rabble creature flight produces a sequence of path samples:

```
[
  {"lat": 19.4195, "lng": -99.1893, "heading": 45.2, "t": 1707500000000},
  {"lat": 19.4196, "lng": -99.1891, "heading": 47.8, "t": 1707500001000},
  ...
]
```

Each sample is converted to SOSA observations:

Path Sample Field	SOSA Observable Property	SOSA URI	Unit
`lng`	xLocation	`onto4mat:xLocation`	degrees
`lat`	yLocation	`onto4mat:yLocation`	degrees
`heading`	hasHeading	`onto4mat:hasHeading`	degrees
(derived)	hasSpeed	`onto4mat:hasSpeed`	m/s

Speed is derived via haversine distance between consecutive samples divided by time delta:

$$v_i = \frac{d_{\text{haversine}}(p_{i-1}, p_i)}{\Delta t_i}$$

where $d_{\text{haversine}}(p_a, p_b) = 2R \arcsin\sqrt{\sin^2\frac{\Delta\phi}{2} + \cos\phi_a \cos\phi_b \sin^2\frac{\Delta\lambda}{2}}$ and $R = 6{,}371{,}000$ m.

The creature itself maps to a `sosa:Platform` with `platform_type: ar_creature`. Each observable property maps to a conceptual `sosa:Sensor`. The feature of interest is the creature's taxonomic identity (e.g., *Danaus plexippus*).

3.2 Mapping Drone Telemetry to SOSA

Drone swarm telemetry from Crazyswarm2 / Crazyflie platforms arrives with explicit Onto4MAT properties:

```
{
  "agent_label": "crazyflie-01",
  "x_location": 1.5,
  "y_location": 2.3,
  "z_location": 0.8,
  "heading": 90.0,
  "speed": 1.2,
  "energy": 0.85,
  "team_alignment": 0.92,
  "team_cohesion": 0.88,
  "team_separation": 0.45
}
```


These map directly to SOSA:

Onto4MAT Property	SOSA Observable Property	Unit
`x_location`	`onto4mat:xLocation`	m (local frame)
`y_location`	`onto4mat:yLocation`	m (local frame)
`heading`	`onto4mat:hasHeading`	degrees
`speed`	`onto4mat:hasSpeed`	m/s
`energy`	`onto4mat:hasEnergy`	ratio (0--1)
`team_alignment`	`onto4mat:hasAlignmentWithTeam`	ratio (0--1)
`team_cohesion`	`onto4mat:hasCohesionWithTeam`	ratio (0--1)
`team_separation`	`onto4mat:hasTeamSeparation`	m

3.3 The Commensurability Claim

Both AR creature flights and drone flights, when mapped through SOSA, produce observations with the same structure: `(platform, observable_property, result_value, result_unit, phenomenon_time)`. The observable properties share the Onto4MAT namespace. The structural similarity is not approximate --- it is ontologically exact.

What differs is:

1. **Coordinate frame**: AR creatures use GPS (WGS84 degrees); drones typically use local frames (meters from origin). This is resolved by normalizing to relative displacements rather than absolute positions.
2. **Velocity scale**: AR creatures move at walking speed (1--2 m/s); drones at flight speed (5--20 m/s). This is resolved by the scale-invariance of normalized team metrics (Section

3.4).

3. **Dimensionality**: AR creatures operate in 2D (lat/lng); drones in 3D (x/y/z). The z-axis can be projected out for 2D comparisons or set to a default for AR data.
4. **Team properties**: AR creatures do not natively report team metrics. These must be derived from the collective trajectories of creatures in a swarm event (Section 3.4).

3.4 Scale-Invariant Team Metrics

For a swarm of n agents at time t, the Reynolds-derived team metrics are:

Alignment (cosine similarity of heading vectors): $$A(t) = \frac{1}{\binom{n}{2}} \sum_{i < j} \cos(\theta_i(t) - \theta_j(t))$$

Cohesion (inverse normalized distance to centroid): $$C(t) = 1 - \frac{\bar{d}*{\text{centroid}}(t)}{d*{\max}}$$

where $\bar{d}_{\text{centroid}}$ is the mean distance of agents to the swarm centroid and $d_{\max}$ is a normalization constant (e.g., maximum observed distance in the session).

Separation (normalized mean nearest-neighbor distance): $$S(t) = \frac{\bar{d}*{\text{nn}}(t)}{d*{\text{ref}}}$$

where d_{ref} is a reference distance (e.g., mean inter-agent distance at session start).

These metrics are **dimensionless ratios**. A swarm of AR creatures with $\bar{d}_{\text{nn}} = 3\text{m}$ and $d_{\text{ref}} = 3\text{m}$ produces $S = 1.0$. A drone swarm with $\bar{d}_{\text{nn}} = 30\text{m}$ and $d_{\text{ref}} = 30\text{m}$ also produces $S = 1.0$. The absolute scale vanishes in normalization.

Similarly, alignment depends only on heading *differences*, not absolute headings or velocities. A flock turning right at 1.5 m/s and a swarm banking right at 15 m/s produce the same alignment score if their heading convergence rates are proportionally similar.

Claim: When Onto4MAT team properties are derived from AR swarm events using the normalized formulas above, they occupy the same region of embedding space as team properties derived from physical drone swarms with structurally equivalent dynamics.

---

4. Experience-Based Runtime Architecture

4.1 The Experience Table

An experience table E is a set of tuples:

$$E = {(e_i, a_i, c_i)}_{i=1}^{N}$$

where $e_i \in \mathbb{R}^{1024}$ is an embedding vector, a_i is the action or action sequence associated with the observation, and c_i is contextual metadata (source domain, agent type, confidence).

The table is populated through the following pipeline:

1. **Telemetry ingestion**: Raw sensor data (AR flight path or drone telemetry) is posted to the SOSA observation API.
2. **Agent interpretation**: An interpreter agent (`observation_analyst` for SOSA, `swarm_coordinator` for Onto4MAT) analyzes the batch and produces a structured response.
3. **Episode storage**: The agent's response is stored as an episode with a 1024-dimensional embedding vector computed by the platform's embedding model.
4. **Consolidation**: During ADM dreaming cycles, episodes are clustered and rules are extracted, enriching the agent's knowledge graph.

Each step transforms the data: raw coordinates become SOSA observations become agent interpretations become embedding vectors. The embedding captures the *structural meaning* of the telemetry --- "evasive spiral with declining energy" rather than "lat 19.4195, lng -99.1893, speed 1.2 m/s."

4.2 Runtime Lookup

At runtime, a device (drone, robot, autonomous vehicle) with current sensor state s performs:

1. **Embed**: Compute $e_s = \text{embed}(s)$ --- the embedding of the current state description.
2. **Search**: Find k nearest neighbors in E by cosine similarity: $\text{NN}_k(e_s, E)$.
3. **Select**: Choose the action a^* from the nearest neighbor (or weighted vote across k neighbors).
4. **Execute**: Perform a^*.

This is a $O(N)$ linear scan or $O(\log N)$ approximate nearest-neighbor search (e.g., HNSW). For $N = 10{,}000$ experience entries with 1024-dimensional vectors, exact search takes <1ms on a Cortex-M7 microcontroller. No GPU required. No model weights. The entire experience table fits in <40MB of flash memory.

4.3 Continuous Enrichment

Unlike trained models, which require retraining to incorporate new data, the experience table grows by append. Each new flight, each new drone mission, each new SOSA observation session adds entries to E. The device can periodically download an updated table via:

```
GET /api/observe/sessions/:id/experience
```

This returns the current experience table as a JSON array of `{embedding, action, query, status}` tuples. The device replaces its local table and immediately benefits from all observations added since its last update.

4.4 Comparison with Conventional Approaches

Property	Neural Network	Experience Table
Training required	Yes (hours/days)	No
GPU at inference	Yes (or edge TPU)	No
Incorporates new data	Retrain or fine-tune	Append
Interpretable	Limited	Each entry traceable to source observation
Memory footprint	Model-dependent (10MB--1GB)	~4KB per entry ($N \times 1024 \times 4$ bytes)
Latency	Forward pass (~1--10ms GPU)	kNN lookup (~0.1--1ms CPU)
Provenance	Opaque	Each embedding linked to episode, agent, session, contributor

The experience table trades generalization capacity for transparency and simplicity. A neural network may generalize to states far from any training example; the experience table is limited to interpolation between known experiences. For environments where contributors have generated dense coverage (e.g., a well-walked park), this limitation is negligible. For novel environments, the table will have sparse coverage and degrade to uncertain action selection --- a known limitation that the system handles by returning confidence scores based on cosine similarity magnitude.

---

5. Data Asset Rights and Economic Model

5.1 The Consent Gate

Every creature in the Rabble application has a boolean property `sosa_opt_in`, defaulting to `false`. The SOSA bridge --- the code path that converts flight path samples into universal observations --- checks this flag before executing:

```
if creature.sosa_opt_in == false:
    return  // No data leaves Rabble's domain
```

This is not a terms-of-service checkbox. It is a per-creature, per-user, runtime-checked gate implemented at the code level. The user decides, for each companion individually, whether its flights contribute to the shared experience space. The decision is reversible at any time via:

```
PUT /api/creatures/:creature_id/sosa-opt-in
{ "opt_in": false }
```

Future flights stop generating SOSA observations immediately. Previously contributed observations remain in the system (they have already been interpreted and embedded), but no new data flows.

This consent architecture follows the AKP principle that **the default state is private** and sharing is an affirmative, granular, revocable choice.

5.2 Ownership Model

Asset	Owner	Created By	Transferable
Raw flight path (`path_samples`)	Creature owner	User's physical movement	No (personal data)
SOSA observations	Creature owner	Automatic bridge (opt-in)	Yes (via marketplace)
Agent episode + embedding	Platform	Agent execution	Licensed to contributors
Consolidated knowledge (rules, entities)	Agent owner	ADM dreaming	Via AKP contracts
Experience table entries	Composite	Aggregation of above	Yes (via marketplace)

The key distinction: raw flight paths are personal data (GPS traces of a person's movement). SOSA observations are *derived* data --- structured, anonymized sensor readings that retain navigational structure but shed personal identity. The bridge strips user identity from the observations; only the creature's taxonomic identity (feature of interest) and the platform type (`ar_creature`) are preserved.

5.3 Distribution Models

We propose three models for distributing economic returns to data contributors. These are not mutually exclusive and may be composed.

Model A: Data Dividend. Contributors receive a pro-rata share of downstream revenue based on observation count. If a user contributed 5,000 of the 800,000 observations in a period, they receive 0.625% of marketplace revenue from experience table access. This model is simple and transparent but does not reward quality --- a lazy straight-line flight and a complex evasive pattern contribute equally.

Model B: Embedding Royalty. When a specific embedding is matched in the marketplace (cosine similarity > threshold τ), the system traces back to the observations that contributed most to that embedding's formation. The creature owners who generated those observations receive a royalty proportional to their contribution's influence on the matched embedding. This requires computing an *attribution score* --- the marginal effect of removing a contributor's observations on the embedding's position. Computationally expensive but rewards novelty: rare patterns that fill gaps in the embedding space earn more.

Model C: Community Pool. Contributors who opt in join a community pool (an AKP group contract). Pool revenue is distributed based on multidimensional contribution metrics: observation count, pattern diversity (entropy of action distributions), temporal coverage (observations across different times of day/seasons), geographic spread (coverage of distinct terrain types). The pool self-governs via democratic mechanisms defined in the AKP contract. This model aligns with Beckstrom's Law: the pool is only valuable if members gain more than it costs them to participate.

5.4 Instrumentation

The ABW platform records every credit movement in an append-only `credit_ledger` with typed transactions. The following transaction types are already implemented:

- `observation_ingest` --- credits charged when observations are ingested
- `marketplace_match_purchase` --- credits charged when experience data is consumed
- `marketplace_match_payout` --- credits paid to data providers

Future transaction types to close the economic loop:

- `sosa_data_dividend` --- periodic distribution to contributors (Model A)
- `sosa_embedding_royalty` --- per-match attribution payout (Model B)
- `sosa_pool_distribution` --- community pool periodic settlement (Model C)

The audit trail is complete from the moment of observation creation to the moment of economic return. Every credit is traceable.

---

6. Validation Protocol

6.1 Hypothesis

H1: Normalized Onto4MAT team metrics derived from AR creature swarm events produce embeddings that are statistically closer to drone swarm embeddings with structurally similar dynamics than to drone swarm embeddings with structurally dissimilar dynamics.

H2: Drones using an experience table populated with AR-derived embeddings achieve navigation performance (measured by goal completion rate, collision avoidance, and formation maintenance) that is statistically better than random action selection and within a bounded factor of performance achieved with drone-derived experience tables.

6.2 Experimental Setup

Phase 1: Embedding Overlap Measurement

1. Collect AR swarm data: Conduct 10 Rabble swarm events with 5--15 creatures each, in varied terrain (open park, wooded trail, urban sidewalk). Record all path samples.
2. Collect drone swarm data: Conduct 10 Crazyswarm2 sessions with 3--5 Crazyflies, executing scripted formation maneuvers (wedge formation, patrol, obstacle avoidance, formation recovery after member loss).
3. Process both through the SOSA pipeline with team metric derivation.
4. Execute interpreter agents on both datasets.
5. Extract episode embeddings.
6. Compute pairwise cosine similarity matrices within and between domains.
7. Test H1: For each drone embedding, compute mean similarity to AR embeddings with structurally similar dynamics (matched by formation type and maneuver category) vs. structurally dissimilar dynamics. Apply paired t-test or Wilcoxon signed-rank test.

Phase 2: Operational Transfer

1. Construct experience table E_{AR} from AR-derived embeddings only.
2. Construct experience table E_{drone} from drone-derived embeddings only.
3. Construct mixed table $E_{\text{mixed}} = E_{\text{AR}} \cup E_{\text{drone}}$.
4. Run Crazyswarm2 sessions using each table for runtime decision-making.
5. Measure: goal completion rate, collision count, mean formation coherence score, mission completion time.
6. Test H2: Compare E_{AR} performance against random baseline and E_{drone} performance.

6.3 Expected Outcomes

We expect H1 to hold strongly for maneuvers with clear structural analogues: formation maintenance, evasion, waypoint navigation. We expect weaker signal for maneuvers that depend on 3D dynamics (altitude changes, vertical obstacle avoidance) where AR creatures lack a z-axis.

We expect H2 to show that E_{AR} significantly outperforms random selection but underperforms E_{drone} --- the AR-derived patterns are useful but incomplete. We expect E_{mixed} to outperform E_{drone} alone, demonstrating that AR-derived diversity improves the experience space even when drone-native data is available.

6.4 Failure Modes

If H1 fails --- AR and drone embeddings do not cluster by structural similarity --- the likely causes are:

1. **Velocity scale contamination**: The embedding model encodes absolute speed rather than relative dynamics. Mitigation: normalize speed to z-scores within each session before embedding.
2. **GPS noise**: Phone GPS at 5--10m accuracy introduces noise that drowns out formation structure. Mitigation: apply Kalman filtering to path samples before SOSA conversion.
3. **Behavioral dissimilarity**: Human-guided AR creatures genuinely don't exhibit swarm dynamics. Mitigation: focus on structured swarm events with explicit formation instructions rather than free-form flights.

If H1 holds but H2 fails --- embeddings cluster but operational transfer doesn't work --- the likely cause is:

4. **Action abstraction mismatch**: The actions associated with AR embeddings ("user turned left") don't map to drone actuator commands ("roll 15 degrees"). Mitigation: map both to Onto4MAT abstract actions (`Turn` , `Accelerate` , `MaintainFormation`) rather than platform-specific commands.

---

7. Community Resilience Application

7.1 The Firefighting Scenario

We describe a concrete application to motivate the framework's social value.

A community in fire-prone terrain (e.g., wildland-urban interface in Southern California, Mediterranean Europe, or Australian bush) uses Rabble as a community activity. Members fly AR creatures through local terrain: the canyon behind the school, the ridge where fires approach, the evacuation routes through hills. Their flights are recreational, social, and educational --- a Rabble event at the nature reserve on Saturday morning.

With SOSA opt-in enabled, these flights generate dense navigational coverage of local terrain. The observation analyst consolidates flight patterns into an experience table that encodes: terrain navigation strategies, wind pattern responses (via heading corrections), obstacle avoidance paths, group coordination patterns.

When autonomous drones are deployed for fire monitoring, reconnaissance, or evacuation guidance in the same terrain, they can download this experience table. The butterfly that spiraled through the canyon last weekend shows the drone how to navigate that canyon in smoke.

7.2 Incentive Alignment

The economic model creates aligned incentives:

1. **Community members** are motivated to fly creatures in local terrain because (a) it's enjoyable, (b) they earn data dividends, and (c) they know the resulting intelligence protects their community.
2. **Emergency services** benefit from pre-built navigational intelligence for terrain they may never have surveyed, contributed by people who know it intimately.
3. **The platform** benefits from increased data volume and marketplace activity.

This is a rare alignment where **self-interest** (recreation, credits), **community interest** (local resilience), and **platform interest** (data volume) point in the same direction.

7.3 Limitations

We do not claim that AR-derived flight data is sufficient for autonomous drone navigation in emergency conditions. Smoke, thermal turbulence, structural collapse, and rapidly changing terrain are conditions with no AR analogue. The experience table provides a navigational prior --- a starting point that is better than no knowledge of the terrain --- not a complete operational solution.

Additionally, the consent model creates a coverage bias: only terrain where opted-in users fly creatures is covered. Areas where Rabble adoption is low will have sparse experience tables. This mirrors existing biases in crowdsourced data (OpenStreetMap coverage correlates with population density) and should be understood as a limitation, not ignored.

---

8. Related Work

Swarm robotics: Brambilla et al. (2013) survey self-organizing behaviors in robot swarms. Our work differs in sourcing behavioral patterns from a non-robotic domain (AR creatures) rather than from the swarm itself.

Ontology-based sensor integration: Compton et al. (2012) describe the SSN ontology for sensor data interoperability. We extend this to cross-domain behavioral transfer, using SOSA not just for data integration but for intelligence transfer.

Transfer learning in robotics: Tobin et al. (2017) demonstrate sim-to-real transfer for robotic grasping. Our approach substitutes "AR-to-real" for "sim-to-real," with the

advantage that AR trajectories occur in real physical environments rather than simulated ones.

Citizen science and crowdsourced data: Sullivan et al. (2009) describe eBird, a citizen science platform that generates biodiversity data at scale. Rabble follows a similar model --- crowdsourced data generation through recreational activity --- but targets navigational intelligence rather than species occurrence.

Data cooperatives and data trusts: Hardjono and Pentland (2019) propose trust frameworks for personal data. Our data asset rights model draws on this tradition, implementing opt-in consent and economic return at the platform level.

Reynolds flocking: Reynolds (1987) established the computational foundations of flocking behavior. Our scale-invariance argument for normalized team metrics builds directly on the locality and relativity of Reynolds' three rules.

---

9. Conclusion

We have presented a framework for cross-domain swarm intelligence transfer that connects a consumer AR application to autonomous drone navigation through a shared ontological layer (SOSA + Onto4MAT), a shared embedding space (ADM episode vectors), and a shared economic model (AKP data asset rights).

The framework is built on existing, operational infrastructure: the SOSA observation pipeline, the Onto4MAT telemetry ingestion system, the observation analyst and swarm coordinator agents, the embedding marketplace, and the credit ledger are all implemented and deployed. What remains is the validation protocol (Section 6) --- the empirical test of whether cross-domain transfer works in practice.

If it does, the implications extend beyond drones and butterflies. Any domain that generates structured sensor observations through SOSA can contribute to and benefit from the shared experience space. A greenhouse's soil moisture sensors and a drone's battery monitor both emit `resource_level` observations that, when consolidated, teach the same structural lesson: declining resources require corrective action. The SOSA layer makes this transfer possible. The embedding space makes it automatic. The consent architecture makes it ethical. The economic model makes it sustainable.

The butterfly collector in Chapultepec Park, flying her Monarch through the trees, is not just playing a game. She is building intelligence. And if the incentives are right, she is compensated for it.

References

Brambilla, M., Ferrante, E., Birattari, M., & Dorigo, M. (2013). Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence*, 7(1), 1--41.

Compton, M., Barnaghi, P., Bermudez, L., et al. (2012). The SSN ontology of the W3C semantic sensor network incubator group. *Journal of Web Semantics*, 17, 25--32.

Haller, A., Janowicz, K., Cox, S., et al. (2017). The SOSA/SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, actuation, and sampling. *Semantic Web*, 10(1), 9--32.

Hardjono, T., & Pentland, A. (2019). Data cooperatives: Towards a foundation for decentralized personal data management. *arXiv preprint arXiv:1905.08819*.

Kiesel, L., Renz, W., & Vilzmann, C. (2022). Onto4MAT: Ontology for multi-agent teaming. *arXiv preprint arXiv:2203.12955*.

Labra, I. (2026a). Active Dream Memory: Biologically-inspired episodic consolidation for autonomous agents. *Working paper, axelotl partners*.

Labra, I. (2026b). Agent Knowledge Protocol: Opt-in knowledge sharing with dynamic market economics. *Working paper, axelotl partners*.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, 21(4), 25--34.

Sullivan, B. L., Wood, C. L., Iliff, M. J., et al. (2009). eBird: A citizen-based bird observation network in the biological sciences. *Biological Conservation*, 142(10), 2282--2292.

Thagard, P. (1989). Explanatory coherence. *Behavioral and Brain Sciences*, 12(3), 435--467.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *IEEE/RSJ IROS*, 23--30.