Bee Movement Prediction Framework — Technical Specification (v1)

*From occupancy heatmaps → spatio-temporal forecasting → generative scenario simulation*

# 1. Executive Summary

This specification defines a rigorously grounded, scalable pipeline to predict bee presence and movement in mapped areas over time. We model occupancy on a spatial grid (Stage 1), add motion dynamics for short-horizon forecasting (Stage 2), and produce multi-future, uncertainty-calibrated scenarios (Stage 3). The same skeleton extends to pests (e.g., Fall Armyworm), livestock, and humans on trails.

# 2. Problem Statement & Notation

Goal: For a spatial domain Ω discretized into cells c ∈ {1,…,C} and discrete time steps t ∈ {1,…,T}, predict P(bee\_present in cell c at time t+Δ) and, optionally, a distribution over trajectories/flows.

State variables:  
• y\_t(c) ∈ {0,1}: occupancy (presence/absence) or y\_t(c) ∈ ℕ₀: count.  
• x\_t(c) ∈ ℝ^F: feature vector (static + dynamic).  
• u\_t = (u\_t^x, u\_t^y): wind components; θ\_t: wind direction; v\_t: speed.  
• G = (V,E): optional environment graph (hedgerows/rows/trails) where cells are nodes and adjacency defines feasible flow.

Targets:  
• Occupancy: p\_t(c) = P(y\_t(c)=1 | x\_t(c)).  
• Intensity: λ\_t(c) = E[y\_t(c) | x\_t(c)] for Poisson/NB; convert to occupancy via p\_t(c) = 1 − exp(−λ\_t(c)·Δt).

# 3. Data Model & Sensors

Spatial discretization:  
• Regular grid (e.g., 5–10 m cells) or environment graph nodes.  
Temporal discretization:  
• Bins Δt = 5–10 minutes for diurnal foraging patterns.

Sensors (examples):  
• Vision: fixed camera(s) counting detections per cell via simple CV or detection models.  
• Acoustic: microphones detecting bee buzz bands (~190–250 Hz fundamentals + harmonics); transforms to counts.  
• Weather: wind (u,v), temperature, humidity, solar elevation; from on-site sensors or nearby stations.  
• Remote sensing features: NDVI/flower index, shade index, distance to hedges/edges, slope/elevation (DEM).

Minimal long-table schema (one row = one cell × one time):  
time, cell\_id, x, y, count, occ, flower\_ndvi, dist\_hedge, shade\_index, wind\_u, wind\_v, temp\_c, sun\_elev, hour\_sin, hour\_cos, doy\_sin, doy\_cos

# 4. Stage 1 — Area-Based Occupancy/Intensity Models

4.1 Logistic Regression (Occupancy)

Model: logit p\_t(c) = βᵀ x\_t(c).  
Training: maximize Bernoulli log-likelihood with L2 regularization; consider class imbalance with focal loss or sampling.  
Interpretation: sign(β) shows effect direction; odds-ratios = exp(β).

4.2 Poisson / Negative Binomial (Counts)

Model: log λ\_t(c) = βᵀ x\_t(c) (Poisson); for overdispersion use Negative Binomial with dispersion k.  
Occupancy conversion: p\_t(c) = 1 − exp(−λ\_t(c)·Δt).

4.3 Nonlinear Additions

• Generalized Additive Models (GAMs) with spline bases for smooth effects (e.g., hour-of-day).  
• Tree ensembles (Random Forest, Gradient Boosting/XGBoost/LightGBM) for nonlinear interactions.

4.4 Calibration

• Reliability plots (predicted vs. empirical).  
• Isotonic or Platt scaling to correct miscalibration.

# 5. Stage 2 — Spatio-Temporal Movement Forecasting

5.1 Diffusion–Advection on a Grid (Discrete Form)

Start from base probability p̂\_t(c) predicted by Stage 1. Update k steps ahead by iterating:  
• Diffusion (neighbor smoothing): p̂\_t^diff(c) = (1−γ)·p̂\_t(c) + γ·mean\_{n∈N(c)} p̂\_t(n), with 0<γ<1.  
• Advection (wind-driven shift): distribute a fraction α of mass downwind according to (u,v), with 0<α<1.  
Combined step: p̂\_{t+Δ}(c) = A\_α( D\_γ( p̂\_t ) ), where D\_γ is diffusion operator; A\_α is advection operator.  
This discretizes the PDE: ∂p/∂t = D∇²p − v·∇p, with D ∝ γ and v ∝ (u,v).

5.2 Particle Filter (Sequential Monte Carlo)

We represent belief over future bee locations with N particles {s\_t^{(i)}}. Each s\_t contains (cell\_id, optional speed/heading). Algorithm per step:  
1) Propagation: sample s\_{t+1}^{(i)} ~ q(s\_{t+1} | s\_t^{(i)}, x\_t) using a motion prior (attraction to flowers, wind push, noise).  
2) Weighting: w\_{t+1}^{(i)} ∝ w\_t^{(i)} · L( observation\_{t+1} | s\_{t+1}^{(i)} ), where L is a likelihood from counts/detections.  
3) Resampling: normalize weights; resample if ESS < τ to avoid degeneracy.  
Output heatmap: aggregate particle positions into cell probabilities; credible regions from quantiles.

5.3 Deep Spatio-Temporal Models

• ConvLSTM: input tensor X ∈ ℝ^{T×H×W×F}, output next-step probability map ŷ ∈ [0,1]^{H×W}. Loss: focal/CE for occupancy; Poisson deviance for counts.  
• Temporal Transformers: sequence modeling with positional encodings and convolutional tokens for spatial context.  
• Graph Neural Networks: when movement is constrained to edges (rows/hedges/trails), use message passing over G.

# 6. Stage 3 — Generative Scenario Modeling & Uncertainty

We simulate multiple futures and quantify uncertainty:  
• Ensembles: run PF, diffusion–advection, and ConvLSTM; combine via averaging or stacking.  
• Scenario drivers: vary wind/temperature/flower density/hive locations; evaluate sensitivity.  
• Uncertainty products: 50%/90% credible regions, prediction intervals for counts, disagreement maps (ensemble variance).

# 7. Evaluation Protocols & Metrics

Data splits: use forward-chaining or day-based splits to avoid temporal leakage. Spatial k-fold if sites differ.  
Metrics:  
• Occupancy: AUC, Average Precision, Brier score, calibration ECE/MCE.  
• Counts: RMSE/MAE, Poisson/NB deviance, coverage of predictive intervals.  
• Forecast skill: Top-K hit rate (% of times true hot cell is in top-K), Continuous Ranked Probability Score (CRPS) if probabilistic.  
• Stability: drift over horizon; sharpness vs. calibration trade-off.

Ablations: compare (i) baseline, (ii) +diffusion, (iii) +advection, (iv) +PF, (v) +deep model; report gains and compute costs.

# 8. Implementation Blueprint

Data pipeline: sensors → aggregation to (cell,time) → feature engineering (sin/cos hour/doy, NDVI tiles, wind vectors) → model.  
Baseline stack: scikit-learn/StatsModels for GLM; XGBoost/LightGBM; PyTorch/TensorFlow for ConvLSTM/Transformers.  
Real-time loop (Δt): ingest latest counts, update features, predict p\_{t+Δ}, render heatmap + natural-language summary.

Jac/Jaseci integration (agentic):  
• node Cell, node Sensor, node POI; walker Predictor(horizon=Δ) traverses cells, queries model, writes probabilities.  
• byllm bindings for natural-language summaries; jac-cloud for serving and persistence.

# 9. Ethics, Data Protection, and Governance

• Consent & privacy: no individual identification; aggregate cell-level outputs; anonymize locations if sharing.  
• Wildlife welfare: noninvasive sensing preferred; avoid disrupting colonies.  
• Bias & fairness: ensure coverage across microhabitats; audit calibration per subregion/time-of-day.  
• Reliability: publish uncertainty; do not overclaim deterministic precision.

# 10. Extensions to Worms/Pests, Livestock, and People

Worms/Pests: replace flower features with host crop/phenology; add rainfall, interventions (sprays/bio-control) and advection-diffusion with stronger wind coupling.  
Livestock: features include grass biomass (NDVI), water/shade, heat index; graph = paddock paths/fences; outputs are grazing hot zones and water visits.  
People/Forest: graph-constrained movement on trails; features: slope, canopy density, light/time; use GNN/PF-on-graph; safety/SAR simulation as use-case.

# Appendix A — Mathematical Details

A1. Logistic regression:  
 L(β) = ∑\_{t,c} [ y\_t(c)·log σ(βᵀx\_t(c)) + (1−y\_t(c))·log(1−σ(βᵀx\_t(c))) ] − λ||β||².  
A2. Poisson GLM:  
 L(β) = ∑\_{t,c} [ y\_t(c)·(βᵀx\_t(c)) − exp(βᵀx\_t(c)) − log(y\_t(c)!) ].  
A3. Diffusion–Advection PDE:  
 ∂p/∂t = D∇²p − v·∇p, discretized by finite differences; stability via Courant–Friedrichs–Lewy (CFL) condition.  
A4. Particle Filter:  
 Importance weights w\_{t+1}^{(i)} ∝ w\_t^{(i)} · p(y\_{t+1} | s\_{t+1}^{(i)}) · p(s\_{t+1}^{(i)} | s\_t^{(i)}) / q(s\_{t+1}^{(i)} | s\_t^{(i)}, y\_{1:t}).  
 Resample when ESS = 1 / ∑\_i (w\_i²) < τ.  
A5. ConvLSTM cell:  
 i\_t = σ(W\_{xi} \* X\_t + W\_{hi} \* H\_{t−1} + b\_i), similarly f\_t, o\_t; C\_t = f\_t ⊙ C\_{t−1} + i\_t ⊙ tanh(W\_{xc} \* X\_t + W\_{hc} \* H\_{t−1}); H\_t = o\_t ⊙ tanh(C\_t), where \* is convolution, ⊙ is Hadamard product.

# Appendix B — Practical Defaults & Hyperparameters

Grid: 6×6 cells (5 m) for a 30×30 m pilot; Δt=5 min.  
GLM: L2=1e−2; class\_weight=balanced if occupancy is sparse.  
Diffusion γ ∈ [0.1, 0.3]; Advection α ∈ [0.05, 0.2].  
Particle Filter: N=2,000–10,000; Gaussian step noise σ\_step=0.5–1.5 m; resample threshold ESS<0.5N.  
ConvLSTM: 2–3 layers, 16–32 channels, kernel 3×3; horizon 1–6 steps; Adam lr=1e−3.