

Listing 1: Annotated configuration for Graph-Augmented Neural Cellular Automata

```
{
  "data": {
    "emojis_dir": "data/emojis",           // Folder containing RGBA PNG assets used as targets
    "targets": [ "bacteria.png", "heart.png", "gecko.png", "tooth.png",
                  "star.png", "smile.png", "four_leaf_clover.png", "squid.png",
                  "eye.png", "real_gecko.png" ], // All available target filenames
    "active_target": "gecko.png",          // Target actually used for training/testing
    "img_size": 40                         // Canvas size (HxW) in cells/pixels
  },

  "model": {
    "n_channels": 16,                      // Total channels = RGBA(4) + hidden(12)
    "perception": { "sobel": true, "identity": true }, // Fixed depthwise filters: identity + Sobel
    "update_mlp": { "hidden_dim": 128, "layers": 2 }, // 1x1 conv MLP width and depth
    "layer_norm": false,                   // (Unused here; GroupNorm is applied on dx)
    "fire_rate": 0.5,                      // Default firing prob at test (training uses range below)
    "update_gain": 0.06,                   // Step size after tanh(dx) bounding
    "alpha_thr": 0.12,                     // Alive threshold on alpha (after 3x3 max-pool) for dynamics
    "use_groupnorm": true                   // Apply GroupNorm(1,C) to dx before bounding
  },

  "training": {
    "pool_size": 1024,                     // Size of SamplePool (diversity/memory of states)
    "batch_size": 16,                       // Mini-batches per epoch
    "steps_per_epoch": 800,                 // Steps per epoch
    "nca_steps_min": 48,                    // Short rollout length range (min)
    "nca_steps_max": 80,                    // Short rollout length range (max)
    "long_rollout_prob": 0.4,               // Chance to use a long rollout for a batch
    "long_rollout_steps_min": 200,          // Long rollout length (min)
    "long_rollout_steps_max": 400,          // Long rollout length (max)
    "fire_rate_min": 0.5,                   // Training: lower bound for random firing prob
    "fire_rate_max": 0.9,                   // Training: upper bound for random firing prob

    "num_epochs": 1000,                     // Planned number of epochs
    "learning_rate": 0.0002,                 // Adam step size
    "optimizer": "Adam",                     // Optimizer type
    "weight_decay": 1e-5,                    // L2 regularization
    "gradient_clip": 1.0,                    // Max global grad-norm (stability)

    "loss": "premult_rgba_mse",              // Full-canvas MSE on premultiplied RGBA
    "loss_alpha_thr": 0.12,                  // (Unused by premult RGBA loss) kept for masked-loss ablations
    "loss_lam_area": 0.0,                    // (Unused with premult RGBA) keep 0
    "stability_enabled": true,                // Optional: K-step stability rollouts for near-target states
    "stability_K": 48,                       // Stability rollout length (used only if enabled in code)
    "stability_threshold": 0.03,              // If per-sample loss < this, include in stability phase
    "stability_weight": 0.65,                // Weight of stability loss term
  }
}
```

```

"reset_worst_prob": 0.10,           // Fraction of worst samples in batch to reseed each step
"random_reseed_prob": 0.05,        // Chance to reseed one random sample (prevents collapse)
"loss_lam_bg_alpha": 0.0,          // (Unused with premult RGBA) keep 0
"loss_lam_bg_rgb": 0.0,            // (Unused with premult RGBA) keep 0

"scheduler": {                     // LR schedule
  "type": "StepLR",                // StepLR: decay LR by gamma every step_size epochs
  "step_size": 150,
  "gamma": 0.85
},

"logging": {
  "checkpoint_interval_epochs": 5,  // Save model every N epochs
  "log_interval": 500,              // Print/train-scalar interval (steps)
  "save_interval": 1000,            // Image save interval (steps)
  "results_dir": "outputs",         // Root folder for images/diagnostics
  "checkpoint_dir": "outputs/checkpoints", // Unused by current trainer path mapping
  "visualize_interval": 1000       // Add image to TensorBoard every N steps
},

"misc": {
  "device": "cuda",                // "cuda" or "cpu"
  "seed": 42                       // RNG seed for reproducibility
},

"graph_augmentation": {
  "d_model": 16,                   // Dim of Q/K projections used for attention over offsets
  "attention_radius": 4,           // Max |dx|,|dy| for sampled mid-range neighbor offsets
  "num_neighbors": 8,              // Offsets sampled each step (from the radius ring)
  "gating_hidden": 32,             // Width of optional channel-wise gate MLP
  "message_gain": 0.25,            // Scale after tanh on graph message (residual strength)
  "hidden_only": true,             // If true, graph message updates hidden channels only
  "message_rate": 0.2,            // Probabilistic use of graph message when msg_every=1
  "message_every": 3              // Temporal sparsity: use graph every N steps (if >1)
},

"damage": {
  "start_epoch": 100,              // Start applying damage as a curriculum from this epoch
  "prob": 0.3,                     // Global chance that a batch sees damage this step
  "per_sample_prob": 0.4,          // Within a damaged batch, fraction of samples damaged
  "kinds": {                       // Mixture weights of damage types
    "square": 0.35, "circle": 0.25, "stripes": 0.10,
    "alpha_drop": 0.15, "saltpepper": 0.05, "gaussian": 0.10
  },
  "size_min": 6,                   // Damage size range (pixels/cells)
  "size_max": 18,
  "stripe_width": 6,               // Stripe thickness for striped damage
  "alpha_thr": 0.2,                // Alive threshold used by damage ops that depend on alpha

```

```
"alpha_dropout_p": 0.15,           // Probability for channelwise alpha dropout damage
"salt_pepper_p": 0.02,             // Probability per pixel for salt-and-pepper noise
"gaussian_softness": 0.35,         // Edge softness for Gaussian/blurred damages
"hidden_noise_sigma": 0.0          // Std dev of noise injected into hidden channels (if any)
}
```
