Package: boids4R (via r-universe)

May 13, 2026

Title Reynolds-Style Boids and Swarm Simulation

Date 2026-05-08

Version 0.3.1

Author Frederic Bertrand [cre, aut] (ORCID:
<https://orcid.org/0000-0002-0837-8281>)

Maintainer Frederic Bertrand <frederic.bertrand@lecnam.net>

Description Provides deterministic two- and three-dimensional boids and swarm simulations for R. The package implements Reynolds-style separation, alignment, and cohesion rules with optional obstacles, attractors, predators, species parameters, and reproducible frame export. Simulation state is renderer-neutral; optional adapters can hand frame data to visualization packages such as 'ggWebGL'. The model follows Reynolds (1987) <doi:10.1145/37402.37406>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1)

Imports Rcpp

Suggests ggWebGL (>= 0.4.0), htmlwidgets, knitr, pkgdown, rmarkdown, testthat (>= 3.0.0)

LinkingTo Rcpp

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website pkgdown

URL https://fbertran.github.io/boids4R/,
https://github.com/fbertran/boids4R

BugReports https://github.com/fbertran/boids4R/issues

NeedsCompilation yes

Repository https://cran.r-universe.dev

Date/Publication 2026-05-13 21:00:23 UTC

RemoteUrl https://github.com/cran/boids4R

RemoteRef HEAD

RemoteSha d9a660492ff7789b26a017ce6c6cbeddca3bd907

Contents

as_ggwebgl_spec 2
boids_params 3
boids_scenario 4
boids_state 5
boids_world 6
simulate_boids 7

Index **9**

`as_ggwebgl_spec` *Convert an object to a ggWebGL primitive specification*

Description

This generic is defined locally so `boids4R` can offer an optional ggWebGL adapter without depending on ggWebGL at load time.

Usage

```
as_ggwebgl_spec(x, ...)
```

Arguments

`x`	Object to convert.
`...`	Additional arguments.

Value

A `ggwebgl_spec` list for supported methods. For a `boids_simulation`, the list contains point and velocity-vector primitives, labels, WebGL view settings, selection options, and timeline metadata for rendering recorded boids frames with `ggWebGL::ggWebGL()`.

Examples

```
sim <- boids_scenario("schooling_2d", n = 15, steps = 3, seed = 5)

if (requireNamespace("ggWebGL", quietly = TRUE) &&
    utils::packageVersion("ggWebGL") >= "0.4.0") {
  spec <- as_ggwebgl_spec(sim, vector_every = 10)
  names(spec)
}
```

`boids_params` *Build boids rule parameters*

Description

Build boids rule parameters

Usage

```
boids_params(
  dimension = c("2d", "3d"),
  separation_weight = 1.45,
  alignment_weight = 0.85,
  cohesion_weight = 0.72,
  goal_weight = 0.08,
  obstacle_weight = 1.6,
  predator_weight = 2.2,
  separation_radius = 0.18,
  alignment_radius = 0.46,
  cohesion_radius = 0.64,
  obstacle_radius = 0.38,
  predator_radius = 0.72,
  max_speed = 1.25,
  max_force = 0.075,
  noise = 0.003
)
```

Arguments

`dimension`	Simulation dimension, either `"2d"` or `"3d"`.
`separation_weight`, `alignment_weight`, `cohesion_weight`	Rule weights.
`goal_weight`, `obstacle_weight`, `predator_weight`	Optional full-lab forces.
`separation_radius`, `alignment_radius`, `cohesion_radius`	Neighbour radii.
`obstacle_radius`, `predator_radius`	Interaction radii for obstacles and predators.

max_speed, max_force	Speed and steering-force limits.
noise	Random steering noise standard deviation.

Value

A boids_params list.

Examples

```
params <- boids_params(
  "2d",
  separation_weight = 1.2,
  alignment_weight = 0.9,
  cohesion_weight = 0.8,
  max_speed = 1.0,
  noise = 0
)
unlist(params[c("separation_weight", "alignment_weight", "max_speed")])
```

---

boids_scenario *Generate and simulate a named boids scenario*

Description

Generate and simulate a named boids scenario

Usage

```
boids_scenario(
  name = c("murmuration_3d", "predator_avoidance_2d", "obstacle_corridor_2d",
    "schooling_2d", "mixed_species_3d"),
  n = 500L,
  dimension = c("2d", "3d"),
  seed = NULL,
  steps = 120L,
  record_every = 2L
)
```

Arguments

name	Scenario name.
n	Number of boids.
dimension	Scenario dimension. Some scenario names imply a dimension.
seed	Optional integer seed for reproducible scenario initialization and simulation noise. When supplied, the global R random-number state is not modified.
steps	Number of simulation steps.
record_every	Record every record_every steps.

Value

A boids_simulation object.

Examples

```
sim <- boids_scenario(
  "schooling_2d",
  n = 20,
  steps = 5,
  record_every = 1,
  seed = 3
)
frames <- as.data.frame(sim)
table(frames$frame)

sim3d <- boids_scenario("murmuration_3d", n = 15, steps = 3, seed = 4)
range(as.data.frame(sim3d)$z)
```

---

boids_state *Create initial boids state*

Description

Create initial boids state

Usage

```
boids_state(
  n,
  dimension = c("2d", "3d"),
  bounds = NULL,
  positions = NULL,
  velocities = NULL,
  species = "boid",
  seed = NULL,
  .rng = NULL
)
```

Arguments

n	Number of boids.
dimension	State dimension, either "2d" or "3d".
bounds	Optional bounds used for random initialization.
positions, velocities	Optional numeric matrices or data frames.
species	Species labels, recycled to n.

seed	Optional integer seed for reproducible initialization. When supplied, a package-local generator is used and the global R random-number state is not modified.
.rng	Internal package-local random-number generator.

Value

A boids_state data frame.

Examples

```
bounds <- matrix(
  c(-1, -1, 1, 1),
  ncol = 2,
  dimnames = list(c("x", "y"), c("min", "max"))
)
state <- boids_state(6, "2d", bounds = bounds, seed = 1)
head(state)

positions <- matrix(c(-0.5, 0, 0.5, 0), ncol = 2, byrow = TRUE)
velocities <- matrix(c(0.1, 0, -0.1, 0), ncol = 2, byrow = TRUE)
boids_state(2, "2d", positions = positions, velocities = velocities)
```

---

boids_world *Build a boids simulation world*

Description

Build a boids simulation world

Usage

```
boids_world(
  dimension = c("2d", "3d"),
  bounds = NULL,
  boundary = c("wrap", "reflect", "open"),
  obstacles = NULL,
  attractors = NULL,
  predators = NULL,
  species = NULL
)
```

Arguments

dimension	World dimension, either "2d" or "3d".
bounds	Numeric matrix with rows x, y, and optionally z, and columns min and max.
boundary	Boundary behavior: wrap, reflect, or open.
obstacles, attractors, predators	Data frames with coordinate columns.
species	Optional species definition table.

Value

A boids_world list.

Examples

```
bounds <- matrix(
  c(-2, -1, 2, 1),
  ncol = 2,
  dimnames = list(c("x", "y"), c("min", "max"))
)
world <- boids_world(
  "2d",
  bounds = bounds,
  boundary = "reflect",
  obstacles = data.frame(x = 0, y = 0, radius = 0.25),
  attractors = data.frame(x = 1.5, y = 0.4, strength = 0.5)
)
world$boundary
world$obstacles
```

---

simulate_boids *Simulate boids dynamics*

Description

Simulate boids dynamics

Usage

```
simulate_boids(
  state,
  world = NULL,
  params = NULL,
  steps,
  dt = 0.05,
  record_every = 1L,
  engine = c("rcpp_grid", "rcpp_naive"),
  seed = NULL
)
```

Arguments

state	Initial boids_state.
world	A boids_world object.
params	A boids_params object.
steps	Number of integration steps.
dt	Time-step size.

record_every	Record every record_every steps.
engine	Simulation engine. rcpp_grid and rcpp_naive are available.
seed	Optional integer seed for deterministic noise. When supplied, the global R random-number state is not modified.

Value

A boids_simulation object.

Examples

```
state <- boids_state(12, "2d", seed = 1)
world <- boids_world(
  "2d",
  boundary = "reflect",
  attractors = data.frame(x = 0.8, y = 0.2, strength = 0.3)
)
params <- boids_params("2d", max_speed = 0.9, noise = 0)
sim <- simulate_boids(
  state,
  world,
  params,
  steps = 4,
  record_every = 2,
  seed = 2
)
head(as.data.frame(sim))
```

Index

as_ggwebgl_spec, 2

boids_params, 3
boids_scenario, 4
boids_state, 5
boids_world, 6

simulate_boids, 7