Package: cgvR (via r-universe)

May 11, 2026

Title Interactive 3D Visualization of Large Cayley Graphs via Vulkan

Version 0.1.2

Description Provides interactive 3D visualization for large-scale Cayley graphs. Specifically designed for analyzing state spaces of the 'TopSpin' puzzle. Leverages the 'Datoviz' library and Vulkan-based GPU rendering for smooth real-time exploration of large graphs and complex state transitions. Implements efficient coordinate mapping for high-dimensional permutation groups, allowing users to visualize the connectivity and structural properties of the puzzle's state space. The rendering engine provides high-performance visuals and interactive camera controls, making it suitable for mathematical analysis of group-theoretic puzzles within the R environment.

Depends R (>= 4.1.0)

Imports grDevices, stats

Suggests cayleyR, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

License MIT + file LICENSE

Encoding UTF-8

SystemRequirements C17, C++17, GNU make, pkg-config, libvulkan-dev (Linux) or LunarG 'Vulkan' 'SDK' (Windows, macOS), libglfw3-dev (Linux) or glfw via Homebrew (macOS), ffmpeg (optional, for cgv_record_*). Optional configure flags: --with-vulkan / --without-vulkan, --with-simd for SSE4.1 + PCLMUL fpng.

RoxygenNote 7.3.3

Config/testthat/edition 3

URL https://github.com/Zabis13/cgvR

BugReports https://github.com/Zabis13/cgvR/issues

NeedsCompilation yes

Author Yuri Baramykov [aut, cre] (ORCID:
<https://orcid.org/0009-0000-7627-4217>), Cyrille Rossant [ctb,
cph] (Author of the Datoviz library)

Maintainer Yuri Baramykov <lbsbmsu@mail.ru>

Config/pak/sysreqs make pkg-config

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

Date/Publication 2026-05-11 21:45:38 UTC

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

RemoteRef HEAD

RemoteSha 2c8d7b1159877d6082456bc89c8331545af364f8

Contents

`cgv_background` *Set Background Color*

Description

Set the panel background to a solid color or a 4-corner gradient.

Usage

```
cgv_background(viewer, color)
```

Arguments

viewer	External pointer returned by cgv_viewer.
color	A single color string (e.g. "#FFFFFF", "white"), or a character vector of 4 colors for corners (top-left, top-right, bottom-left, bottom-right).

Value

No return value, called for side effects: updates the panel's background color on the GPU. Returns NULL invisibly.

---

cgv_camera *Set Camera Position and Direction*

Description

Set Camera Position and Direction

Usage

```
cgv_camera(viewer, position = c(0, 0, 5), target = c(0, 0, 0), up = c(0, 1, 0))
```

Arguments

viewer	External pointer returned by cgv_viewer.
position	Numeric vector of length 3 (x, y, z).
target	Numeric vector of length 3 — look-at point.
up	Numeric vector of length 3 — up direction.

Value

No return value, called for side effects: updates the camera's position, look-at target and up vector on the active panel. Returns NULL invisibly.

cgv_camera_mode *Switch Camera Mode*

Description

Switch Camera Mode

Usage

```
cgv_camera_mode(viewer, mode = c("fly", "orbit"))
```

Arguments

viewer	External pointer returned by cgv_viewer.
mode	Character: "fly" (WASD + mouse) or "orbit" (rotate around target with Shift+drag, scroll zoom).

Value

No return value, called for side effects: switches the active camera between fly and orbit interaction modes. Returns NULL invisibly.

cgv_clear_path *Clear Path Highlight*

Description

Remove path highlight and restore original node colors and sizes.

Usage

```
cgv_clear_path(viewer)
```

Arguments

viewer	External pointer returned by cgv_viewer.

Value

No return value, called for side effects: removes the path highlight overlay and restores original node colors and sizes. Returns NULL invisibly.

cgv_close *Close the Viewer*

Description

Close the Viewer

Usage

```
cgv_close(viewer)
```

Arguments

viewer	External pointer returned by cgv_viewer.

Value

No return value, called for side effects: releases the underlying Vulkan application, GPU resources and (in windowed mode) the GLFW window held by viewer. Returns NULL invisibly.

cgv_fly_path *Fly Camera Along a Path*

Description

Smoothly animate the camera along a sequence of 3D waypoints using Catmull-Rom spline interpolation.

Usage

```
cgv_fly_path(viewer, positions, duration = 5, loop = FALSE)
```

Arguments

viewer	External pointer returned by cgv_viewer.
positions	Numeric matrix with 3 columns (x, y, z), one row per waypoint.
duration	Total animation duration in seconds.
loop	Logical: loop the animation?

Value

No return value, called for side effects: starts a Catmull-Rom spline camera animation through the given waypoints. Returns NULL invisibly.

cgv_fly_to *Fly Camera to a Node*

Description

Smoothly animate the camera to center on a given node. Requires that `cgv_set_graph` was called first (for node positions).

Usage

```
cgv_fly_to(viewer, node_id, duration = 1)
```

Arguments

`viewer`	External pointer returned by `cgv_viewer`.
`node_id`	Integer node identifier (1-based R index).
`duration`	Animation duration in seconds.

Value

No return value, called for side effects: starts a camera animation toward the chosen node. Returns `NULL` invisibly.

cgv_highlight_path *Highlight a Path*

Description

Draw a highlighted path between nodes. Path nodes get a distinct color and enlarged size; path edges are drawn as thick colored segments on top of existing edges.

Usage

```
cgv_highlight_path(
  viewer,
  path,
  color = "#FF0000",
  node_scale = 2,
  edge_width = 5
)
```

Arguments

viewer	External pointer returned by cgv_viewer.
path	Integer vector of node IDs forming the path (1-based).
color	Color as hex string "#RRGGBB" or "#RRGGBBAA".
node_scale	Numeric: size multiplier for highlighted nodes (default 2.0).
edge_width	Numeric: line width for path edges (default 5.0).

Value

No return value, called for side effects: adds (or replaces) the highlight overlay for the given path. Returns NULL invisibly.

---

cgv_is_stub_build *Is this a stub build?*

Description

Returns TRUE when cgvR was installed without native rendering support (no Vulkan / GLFW found at install time). In that case all rendering functions raise an error; pure-R helpers like cgv_layout_fr still work.

Usage

```
cgv_is_stub_build()
```

Value

Logical scalar.

---

cgv_layout_fr *Fruchterman-Reingold Force-Directed Layout*

Description

Computes 3D (or 2D) node positions so that connected nodes settle at approximately equal distance ideal_len, while non-adjacent nodes repel each other. Implements the Fruchterman-Reingold algorithm with linearly-cooling temperature, fully vectorized over nodes and edges.

Usage

```
cgv_layout_fr(
  n_nodes,
  edges,
  n_iter = 300L,
  ideal_len = NULL,
  dim = 3L,
  seed = NULL,
  init = NULL,
  cool = 0.98,
  normalize = TRUE,
  verbose = FALSE
)
```

Arguments

n_nodes	Integer. Number of nodes.
edges	Two-column integer matrix (from, to), 1-based. Direction is ignored (forces are symmetric).
n_iter	Integer. Number of iterations (default 300).
ideal_len	Numeric. Target edge length. If NULL, defaults to n_nodes^(1/dim) * 0.8.
dim	Integer, 2 or 3. Output dimensionality (default 3).
seed	Optional integer for reproducible initialization.
init	Optional n_nodes x dim matrix of initial positions. If NULL, uses rnorm.
cool	Numeric in (0, 1]. Per-iteration temperature decay (default 0.98).
normalize	Logical. If TRUE, recenters and scales output to fit in [-15, 15] (matches the demos' camera setup). Default TRUE.
verbose	Logical. Print progress every 50 iterations.

Value

Numeric matrix n_nodes x dim of node coordinates.

Examples

```
edges <- cbind(c(1, 2, 3, 4), c(2, 3, 4, 1))
pos <- cgv_layout_fr(4, edges, n_iter = 200)
```

cgv_layout_fr_bh *Barnes-Hut Fruchterman-Reingold Layout (3D, fast)*

Description

Same energy model as cgv_layout_fr but with O(n log n) repulsion approximated through an octree. Suitable for large graphs (10^4+ nodes).

Usage

```
cgv_layout_fr_bh(
  n_nodes,
  edges,
  n_iter = 200L,
  ideal_len = NULL,
  theta = 1,
  cool = 0.98,
  min_dist = 0.01,
  seed = NULL,
  init = NULL,
  normalize = TRUE
)
```

Arguments

n_nodes	Integer. Number of nodes.
edges	Two-column integer matrix (from, to), 1-based.
n_iter	Integer. Number of iterations (default 200).
ideal_len	Numeric target edge length. If NULL, defaults to n_nodes^(1/3) * 0.8.
theta	Barnes-Hut opening angle (default 1.0). Larger = faster, less accurate.
cool	Per-iteration temperature decay (default 0.98).
min_dist	Minimum distance clamp for repulsion / edge attraction (default 0.01). Avoids division-by-zero when nodes coincide.
seed	Optional integer for reproducible initialization.
init	Optional n_nodes x 3 matrix of initial positions. If NULL, uses rnorm.
normalize	Logical. If TRUE, recenters and scales output to [-15, 15]. Default TRUE.

Value

Numeric matrix n_nodes x 3 of node coordinates.

cgv_record_start *Start Recording the Viewer to a Video File*

Description

Pipes raw RGB frames from the live canvas into ffmpeg, which encodes them on the fly. The recording runs in the background while the user keeps interacting with the viewer (mouse, keyboard).

Usage

```
cgv_record_start(
  viewer,
  file,
  fps = 30L,
  duration = NA_real_,
  ffmpeg_args = NULL
)
```

Arguments

viewer	External pointer from cgv_viewer(). The window must already be running (call this from a frame callback, or after cgv_run() has built the canvas — typically by setting a duration and starting recording before cgv_run()).
file	Output path; extension determines the format.
fps	Frames per second (default 30).
duration	Optional cap in seconds; NA = unlimited.
ffmpeg_args	Optional extra ffmpeg flags spliced before the output path (e.g. "-c:v libvpx-vp9 -b:v 2M"). NULL = use defaults (libx264, yuv420p, veryfast).

Details

Requires ffmpeg on PATH. The container/codec is chosen by the file extension (.mp4 / .mkv / .webm / etc.). Frames are captured at the requested fps; if the rendering loop produces frames faster, extras are dropped.

The recording is automatically stopped when the viewer is closed, cgv_record_stop() is called, or the optional duration elapses.

Value

No return value, called for side effects: stores the recording parameters on the viewer; the ffmpeg pipe is opened lazily on the first rendered frame. Returns NULL invisibly.

cgv_record_stop *Stop the Active Recording*

Description

Closes the ffmpeg pipe (which finalises the output file) and frees the recording state on the viewer.

Usage

```
cgv_record_stop(viewer)
```

Arguments

viewer	External pointer from `cgv_viewer()`.

Value

Invisibly, an integer vector `c(frames, fps)` reporting how many frames were written.

cgv_run *Run the Viewer Event Loop*

Description

Starts the rendering loop.

Usage

```
cgv_run(viewer, n_frames = 0L)
```

Arguments

viewer	External pointer returned by `cgv_viewer`.
n_frames	Maximum number of frames to render. `0` (default) means run until the window is closed (interactive mode). A positive value renders exactly that many frames and returns — useful for smoke tests and scripted rendering on machines with a display.

Value

No return value, called for side effects: drives the render loop. Blocks until the window is closed (interactive mode, `n_frames = 0`) or until exactly `n_frames` frames have been rendered (scripted/offscreen mode). Returns `NULL` invisibly.

cgv_set_graph *Set Graph Data for Rendering*

Description

Provide the full graph (or a subgraph) as adjacency data with optional node colors, sizes, and colormap.

Usage

```
cgv_set_graph(
  viewer,
  nodes,
  edges,
  positions = NULL,
  node_values = NULL,
  node_colors = NULL,
  node_sizes = NULL,
  cmap = 6L
)
```

Arguments

viewer	External pointer returned by cgv_viewer.
nodes	Integer vector of node IDs.
edges	Two-column integer matrix (from, to), 1-based.
positions	Nx3 numeric matrix of 3D coordinates (optional; linear if NULL).
node_values	Numeric vector of length N for automatic coloring via colormap (e.g. BFS depth, group id). Ignored if node_colors is provided.
node_colors	Nx4 integer matrix (RGBA 0-255) for explicit node colors. Takes priority over node_values.
node_sizes	Numeric vector of length N for point sizes (default 10).
cmap	Integer colormap id (default 6 = viridis). Common values: 5 = plasma, 6 = viridis, 7 = inferno, 8 = magma.

Value

No return value, called for side effects: uploads the node and edge buffers to the GPU and caches node positions on the viewer for subsequent cgv_fly_to / picking calls. Returns NULL invisibly.

cgv_set_visibility *Set Visibility Depth*

Description

Controls how many hops from the current focus node are rendered.

Usage

```
cgv_set_visibility(viewer, depth = 10L)
```

Arguments

viewer	External pointer returned by cgv_viewer.
depth	Integer number of hops (default 10).

Value

No return value, called for side effects: updates the visibility depth used to filter rendered nodes by BFS distance. Returns NULL invisibly.

cgv_viewer *Create a 3D Cayley Graph Viewer*

Description

Opens a Vulkan-powered 3D window for interactive graph visualization.

Usage

```
cgv_viewer(width = 1280L, height = 720L, title = "cgvR", offscreen = FALSE)
```

Arguments

width	Window width in pixels.
height	Window height in pixels.
title	Window title.
offscreen	If TRUE, creates the viewer without a window surface (headless mode). Useful for automated tests and CI where no display is available.

Value

An external pointer to the viewer object (invisibly).

Index