Package 'skymodelr'

June 27, 2026

Type Package

Title Generates and Samples Realistic Terrestrial Atmospheres

Version 0.3.2

Description Generates physically based sky environment maps and radiance samples using the spectral Hosek-Wilkie and Prague atmosphere models. Functions write high-dynamic-range 'OpenEXR' domes in latitude-longitude projections, compute per-direction RGB or 55-channel values, and optionally composite time-accurate star fields and moon phases. Features include automatic sun and moon positioning from date, time and location, support for sea-level and high-altitude observers, wide-spectrum coefficients, and multithreaded C++ acceleration for fast, high-resolution output. For model details, see Hosek and Wilkie (2012) <doi:10.1145/2185520.2185591>, Hosek and Wilkie (2013) <doi:10.1109/MCG.2013.18>, Wilkie et al. (2021) <doi:10.1145/3450626.3459758>, and Vevoda et al. (2022) <doi:10.1111/cgf.14677>.

License GPL-3

Copyright file inst/COPYRIGHTS

Depends R (>= 4.3.0)

Imports Rcpp, swephR, rayvertex (>= 0.14.0), rayimage (>= 0.24.1)

LinkingTo Rcpp, RcppThread, libimath, libopenexr

Encoding UTF-8

LazyData true

Config/build/compilation-database true

Suggests testthat (>= 3.0.0), rayrender (>= 0.40.1)

Config/testthat/edition 3

URL https://www.skymodelr.com

BugReports https://github.com/tylermorganwall/skymodelr/issues

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Tyler Morgan-Wall [aut, cre, cph],
Petr Vevoda [ctb],
Charles University [cph],
Eric Bruneton [ctb, cph],
Lukas Hosek [ctb, cph],
Alexander Wilkie [ctb, cph]

Maintainer Tyler Morgan-Wall <tylermw@gmail.com>

Repository CRAN

Date/Publication 2026-06-27 09:40:02 UTC

Contents

`calculate_sky_radiance` *Sample Prague sky radiance at a chosen wavelength.*

Description

Evaluate the Prague spectral sky model at arbitrary spherical directions and return radiance at a user-specified wavelength. Use `render_mode = "all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.

Usage

```
calculate_sky_radiance(
  phi,
  theta,
  lambda_nm,
  altitude = 0,
  elevation = 10,
  visibility = 50,
  albedo = 0.5,
```

```
  azimuth = 90,
  number_cores = 1,
  wide_spectrum = FALSE,
  render_mode = "all"
)
```

Arguments

phi	Azimuthal angle of the sample, degrees. Vectorized. Range 0 to 360.
theta	Vertical angle of the sample, degrees. Vectorized. Range -90 to 90.
lambda_nm	Wavelength in nanometers. Vectorized. Must lie within the loaded Prague dataset range.
altitude	Default `0`, vectorized. Altitude of the viewer in meters. Range 0 to 15000.
elevation	Default `10`, vectorized. Solar elevation angle above/below the horizon (degrees). Range -4.2 to 90.
visibility	Default `50`, vectorized. Range 20 to 131.8. Meteorological range in kilometers for Prague model.
albedo	Default `0.5`, vectorized. Range 0 to 1. Ground albedo.
azimuth	Default `90`, single value. Solar azimuth (degrees). Defaults South.
number_cores	Default `1`. Number of threads to use in computation.
wide_spectrum	Default `FALSE`. Whether to use the wide-spectrum (55-channel, polarised) coefficients.
render_mode	Default `"all"`. One of `"all"`, `"atmosphere"`, or `"sun"`. Use `"all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.

Value

Numeric vector of radiance values at the requested wavelength(s).

Examples

```
lambda_vals = calculate_sky_radiance(
  phi = c(90, 90),
  theta = c(45, 45),
  lambda_nm = 550,
  altitude = c(0, 10000),
  elevation = 20,
  visibility = 80,
  albedo = 0.1
)
cbind(altitude = c(0, 10000), radiance = lambda_vals)
```

`calculate_sky_values` *Sample a direction from the Prague model.*

Description

Evaluate the Prague spectral sky model at arbitrary spherical directions without writing an image, returning radiance-only samples.

Usage

```
calculate_sky_values(
  phi,
  theta,
  altitude = 0,
  elevation = 10,
  visibility = 50,
  albedo = 0.5,
  azimuth = 90,
  number_cores = 1,
  wide_spectrum = FALSE,
  render_mode = "all"
)
```

Arguments

`phi`	Horizontal angle of the sample, degrees. Vectorized. Range 0 to 360.
`theta`	Vertical angle of the sample, degrees. Vectorized. Range -90 to 90.
`altitude`	Default `0`, vectorized. Altitude of the viewer in meters. Range 0 to 15000.
`elevation`	Default `10`, vectorized. Solar elevation angle above/below the horizon (degrees). Range -4.2 to 90.
`visibility`	Default `50`, vectorized. Range 20 to 131.8. Meteorological range in kilometers for Prague model.
`albedo`	Default `0.5`, vectorized. Range 0 to 1. Ground albedo.
`azimuth`	Default `90`, single value. Solar azimuth (degrees). Defaults South.
`number_cores`	Default `1`. Number of threads to use in computation.
`wide_spectrum`	Default `FALSE`. Whether to use the wide-spectrum (55-channel, polarised) coefficients.
`render_mode`	Default `"all"`. One of `"all"`, `"atmosphere"`, or `"sun"`. Use `"all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.

Value

3-column RGB matrix.

Examples

```
# Generate a basic atmosphere with the Prague model
value_grid = expand.grid(
  phi = seq(0, 360, by = 30),
  theta = seq(0, 90, by = 10),
  altitude = c(0, 10000)
)
vals = calculate_sky_values(
  phi = value_grid$phi,
  theta = value_grid$theta,
  altitude = value_grid$altitude,
  elevation = 45,
  visibility = 120,
  albedo = 0
)
cbind(value_grid, vals)
```

---

calculate_sun_brightness

Sample sun luminance at the disk center.

Description

Evaluate the Prague or Hosek sky model at the sun center and integrate against the CIE Y curve to return a luminance value. This is useful for relative attenuation (e.g., comparing zenith sun to a low sun).

Usage

```
calculate_sun_brightness(
  elevation = 10,
  azimuth = 90,
  albedo = 0.5,
  turbidity = 3,
  altitude = 0,
  visibility = 50,
  hosek = TRUE,
  wide_spectrum = FALSE,
  lambda_nm = NULL
)
```

Arguments

`elevation`	Default `10`. Solar elevation angle above/below the horizon (degrees). Range `[-4.2, 90]` for Prague, `[0, 90]` for Hosek.
`azimuth`	Default `90`. Solar azimuth (degrees).

albedo	Default 0.5. Ground albedo, range 0 to 1.
turbidity	Default 3. Atmospheric turbidity, range 1.7 to 10 (*Hosek only*).
altitude	Default 0. Observer altitude (m), range 0 to 15000 (*Prague only*).
visibility	Default 50. Meteorological range (km); *Prague only*.
hosek	Default TRUE. FALSE selects the Prague model.
wide_spectrum	Default FALSE. Use wide-spectrum (55-channel) coefficients for Prague at sea level only.
lambda_nm	Optional vector of wavelengths for Hosek sampling.

Value

Numeric scalar of sun luminance (CIE Y, relative scale).

Examples

```
calculate_sun_brightness(elevation = 45, hosek = TRUE)
```

---

clear_sky_data *Clear cached sky data files*

Description

Removes files cached by download_sky_data() under tools::R_user_dir("skymodelr", "data").

Usage

```
clear_sky_data(files = NULL, ask = interactive())
```

Arguments

files	Default NULL. Character vector of cached file basenames to remove.
ask	Default interactive(). Whether to ask for confirmation before deleting files.

Value

Invisibly, the paths successfully removed.

Examples

```
clear_sky_data()
```

`download_sky_data` *Download Prague Sky Model Coefficient Data*

Description

This model which allows for sun angles below the horizon, wide spectral ranges including infrared, polarization, as well rendering at altitude) need to download a coefficient file. There are three versions of this dataset, listed here in order of increasing size:

Usage

```
download_sky_data(sea_level = TRUE, wide_spectrum = FALSE)
```

Arguments

`sea_level`	Default `TRUE`. Download the sea-level-only data. Set to `FALSE` to download the full-altitude dataset.
`wide_spectrum`	Default `FALSE`. If `TRUE`, downloads the wide-spectrum (55-channel, polarised) version. Valid only when `sea_level = TRUE`.

Details

Argument combination	File	Size
`sea_level = TRUE, wide_spectrum = FALSE`	`SkyModelDatasetGround.dat`	107MB
`sea_level = TRUE, wide_spectrum = TRUE`	`PragueSkyModelDatasetGroundInfra.dat`	574MB
`sea_level = FALSE`	`SkyModelDataset.dat`	2.4GB

Value

Invisibly, the full path to the data file.

Examples

```
# Standard (11-channel, sea-level) coefficients
download_sky_data()

# Wide-spectrum sea-level coefficients
download_sky_data(wide_spectrum = TRUE)

# Full altitude-range coefficients
download_sky_data(sea_level = FALSE)
```

`generate_moon_latlong` *Generate the atmosphere with the moon*

Description

Note that this is just a scaled version of `generate_sky()`, scaled down by the luminance of the moon as compared to the sun. This function takes the phase of the moon into account, along with the increase in luminosity around a full moon (known as opposition surge). Moonlight attenuation uses the Rozenberg/Krisciunas-Schaefer airmass approximation with configurable `moon_extinction_kV`.

Usage

```
generate_moon_latlong(
  datetime,
  lat,
  lon,
  filename = NA,
  albedo = 0.5,
  turbidity = 3,
  altitude = 0,
  resolution = 2048,
  number_cores = 1,
  moon_atmosphere = FALSE,
  earthshine = TRUE,
  earthshine_albedo = 0.19,
  solar_irradiance_w_m2 = 1300,
  moon_extinction_kV = 0.172,
  hosek = TRUE,
  wide_spectrum = FALSE,
  visibility = 50,
  moon_texture_width = 801,
  moon_texture_height = 801,
  verbose = FALSE
)
```

Arguments

`datetime`	POSIX-compatible date-time.
`lat`	Observer latitude (degrees N).
`lon`	Observer longitude (degrees E; west < 0).
`filename`	Default `NA`. Path to the image file to write.
`albedo`	Default `0.5`. Ground albedo, range 0 to 1.
`turbidity`	Default `3`. Atmospheric turbidity, range 1.7 to 10 (*Hosek only*).
`altitude`	Default `0`. Observer altitude (m), range 0 to 15000 (*Prague only*).
`resolution`	Default `2048`. Image height in pixels (width = 2 * height).

`number_cores` Default `1`. CPU threads to use.

`moon_atmosphere`
Default `FALSE`. If `TRUE`, this generates atmospheric scattering from light from the moon.

`earthshine` Default `TRUE`. If `FALSE`, disable earthshine contribution on the moon.

`earthshine_albedo`
Default `0.19`. Effective Earth albedo term used in the earthshine irradiance approximation.

`solar_irradiance_w_m2`
Default `1300`. Reference solar irradiance at 1 AU (W/m^2) used to normalize earthshine emission intensity.

`moon_extinction_kV`
Default `0.172`. V-band atmospheric extinction coefficient for direct moonlight attenuation.

`hosek` Default `TRUE`. `FALSE` selects the Prague model.

`wide_spectrum` Default `FALSE`. 55-channel Prague coefficients (altitude = 0m only).

`visibility` Default `50`. Meteorological range (km); *Prague only*.

`moon_texture_width`
Default `801`. Internal moon-texture render width.

`moon_texture_height`
Default `801`. Internal moon-texture render height.

`verbose` Default `FALSE`. Whether to print progress bars/diagnostic info.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions `(resolution, 2 * resolution, 4)`.

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
# Moonlit sky (Hosek), mid-evening in DC
generate_moon_latlong(
  datetime = as.POSIXct("2025-09-05 19:30:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  resolution = 400,
  turbidity = 3,
  verbose = TRUE
) |>
  rayimage::render_exposure(15) |>
  rayimage::plot_image()
```

`generate_planets` *Generate a bright-planet image array*

Description

Build a planetary luminance map aligned with the sky dome for compositing within `generate_sky_latlong()`.

Usage

```
generate_planets(
  datetime,
  lon,
  lat,
  filename = NA,
  resolution = 2048,
  turbidity = 3,
  ozone_du = 300,
  altitude = 0,
  color = FALSE,
  planet_width = 1,
  upper_hemisphere_only = TRUE,
  atmosphere_effects = TRUE,
  number_cores = 1,
  verbose = FALSE
)
```

Arguments

`datetime`	POSIXct timestamp used for ephemerides.
`lon`	Observer longitude in degrees (east positive).
`lat`	Observer latitude in degrees.
`filename`	Default `NA`. Destination image path to write. When `NA`, the image array is returned without writing.
`resolution`	Default `2048`. Map half-width (image is `2 * resolution` by `resolution`).
`turbidity`	Atmospheric turbidity for extinction modelling.
`ozone_du`	Column ozone (Dobson Units) for colour shifts.
`altitude`	Observer altitude in metres.
`color`	Render RGB (`TRUE`) stars or monochrome (`FALSE`).
`planet_width`	Approximate point-spread size for planets in pixels.
`upper_hemisphere_only`	If `TRUE`, mask pixels below the horizon.
`atmosphere_effects`	If `TRUE`, apply atmospheric extinction.
`number_cores`	CPU threads used for rendering.
`verbose`	Emit diagnostic output when `TRUE`.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions `(resolution, 2 * resolution, 4)`.

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
# Basic star field over Washington, DC at a fixed time
generate_planets(
  datetime = as.POSIXct("2025-03-21 02:20:00", tz = "America/New_York"),
  lon = -77.0369,
  lat = 38.9072,
  resolution = 400,
  color = TRUE,
  planet_width = 1,
  atmosphere_effects = TRUE,
  upper_hemisphere_only = TRUE,
  number_cores = 2
) |>
  rayimage::plot_image()
```

---

generate_sky *Generate a Hosek-Wilkie sky dome array*

Description

Evaluate either the Hosek-Wilkie or Prague analytic sky models and return a high-dynamic-range image array for the given solar configuration. An image file is written only when `filename` is supplied.

Usage

```
generate_sky(
  filename = NA,
  albedo = 0.1,
  turbidity = 3,
  elevation = 10,
  azimuth = 90,
  altitude = 0,
  resolution = 2048,
  number_cores = 1,
  hosek = TRUE,
```

```
  wide_spectrum = FALSE,
  visibility = 50,
  verbose = FALSE,
  render_mode = "all",
  below_horizon = TRUE
)
```

Arguments

`filename`	Default `NA`. Path to an image file to write. If not given, the array is returned instead.
`albedo`	Default `0.1`. 0.0-1.0 ground albedo. Grass has an albedo of about 0.09, while a landscape covered in snow will have an albedo of 1.0.
`turbidity`	Default `3`. 1.7-10 atmospheric turbidity. Only valid for Hosek model.
`elevation`	Default `10`. Solar elevation above the horizon (degrees).
`azimuth`	Default `90`, sun directly east. Solar azimuth (degrees). The left edge of the image faces north and the middle faces south.
`altitude`	Default `0`. Altitude of the viewer in meters. Valid range: 0 to 15000. Only valid for the Prague model.
`resolution`	Default `2048`. Height of the image. Width is twice this number.
`number_cores`	Default `1`. Number of threads to use in computation.
`hosek`	Default `TRUE`. Set to `FALSE` to enable the Prague 2021-22 spectral sky model.
`wide_spectrum`	Default `FALSE`. Use the 55-channel Prague coefficients (sea level only).
`visibility`	Default `50`. Meteorological range in kilometres for Prague model.
`verbose`	Default `FALSE`. Whether to print progress bars/diagnostic info.
`render_mode`	Default `"all"`. One of `"all"`, `"atmosphere"`, or `"sun"`. Use `"all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.
`below_horizon`	Default `TRUE`. Whether to sample atmospheric scattering below the horizon, which is non-zero when altitude > 0.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions `(resolution, 2 * resolution, 4)`.

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
sky = generate_sky(
  resolution = 8,
  elevation = 30,
  azimuth = 135,
  render_mode = "atmosphere"
)
dim(sky)


# Hosek model (default): clear morning, Sun SE, with solar disk
generate_sky(
  resolution = 400,
  elevation  = 15,
  azimuth    = 135,
  turbidity  = 3,
  render_mode = "all"
) |>
  rayimage::plot_image()

# Same view but hazier and without the solar disk
generate_sky(
  resolution = 400,
  elevation  = 15,
  azimuth    = 135,
  turbidity  = 6,
  render_mode = "atmosphere"
) |>
  rayimage::plot_image()

# Prague model (requires downloaded coefficients)
generate_sky(
  resolution = 400,
  hosek      = FALSE,
  altitude   = 0,
  visibility = 80,
  albedo     = 0.2,
  elevation  = 5,
  azimuth    = 220,
  number_cores = 2
) |>
  rayimage::plot_image()
```

---

generate_sky_latlong *Generate a location and time-specific sky dome (optionally with stars)*

Description

Convenience wrapper around `generate_sky()` that:

1. Computes the Sun's apparent position for `datetime`, `lat`, and `lon` (via Swiss Ephemeris / **swephR**).
2. Renders the corresponding sky model.
3. Optionally overlays a star field using `generate_stars()` or moon with `generate_moon_latlong()`.

Usage

```
generate_sky_latlong(
  datetime,
  lat,
  lon,
  filename = NA,
  albedo = 0.5,
  turbidity = 3,
  altitude = 0,
  resolution = 2048,
  number_cores = 1,
  hosek = TRUE,
  wide_spectrum = FALSE,
  visibility = 50,
  stars = FALSE,
  star_width = 1,
  planets = FALSE,
  moon = FALSE,
  moon_atmosphere = FALSE,
  moon_hosek = TRUE,
  render_mode = "all",
  below_horizon = TRUE,
  verbose = FALSE,
  stars_exposure = 0,
  ...
)
```

Arguments

`datetime`	POSIX-compatible date-time.
`lat`	Observer latitude (degrees N).
`lon`	Observer longitude (degrees E; west < 0).
`filename`	Default `NA`. Path to the `.exr` file to write.
`albedo`	Default `0.5`. Ground albedo, range 0 to 1.
`turbidity`	Default `3`. Atmospheric turbidity, range 1.7 to 10 (*Hosek only*).
`altitude`	Default `0`. Observer altitude (m), range 0 to 15000 (*Prague only*).
`resolution`	Default `2048`. Image height in pixels (width = 2 * height).
`number_cores`	Default `1`. CPU threads to use.
`hosek`	Default `TRUE`. `FALSE` selects the Prague model.

wide_spectrum	Default FALSE. 55-channel Prague coefficients (altitude = 0m only).
visibility	Default 50. Meteorological range (km); *Prague only*.
stars	Default FALSE. If TRUE, composite a star field using generate_stars(). Automatically falls back to a pure star render when the black-sky condition is met (see *Details*).
star_width	Default 1. Passed to generate_stars() to control stellar point-spread size.
planets	Default FALSE. If TRUE, composite bright planets via generate_planets().
moon	Default FALSE. If TRUE, overlay a moon render from generate_moon_latlong().
moon_atmosphere	Default FALSE. If TRUE, this generates atmospheric scattering from light from the moon.
moon_hosek	Default TRUE. Whether to use the faster (but less accurate) Hosek model for atmospheric scattering from the moon. Note that the light scattered from the moon is much less intense than the sun, and thus small inaccuracies are much less likely to be noticable.
render_mode	Default "all". One of "all", "atmosphere", or "sun". Use "all" for atmosphere + solar disk, "atmosphere" for atmospheric radiance only, or "sun" for the solar disk only.
below_horizon	Default TRUE. Whether to sample atmospheric scattering below the horizon, which is non-zero when altitude > 0.
verbose	Default FALSE. Whether to print progress bars/diagnostic info.
stars_exposure	Default 0. Increases star exposure by 2^exposure. Non-physical, this just controls adjustments for artistic effect.
...	Additional **named** arguments forwarded to generate_stars(), and when enabled, generate_planets() and generate_moon_latlong(). Unmatched names are ignored.

Details

Solar angles - altitude (degrees above the horizon) and azimuth (degrees clockwise from east, so 90 degrees = south) - are derived internally; you never have to supply them directly.

Black-sky rule - With the Prague model the sky radiance is defined only down to -4.2 degrees, and with the Hosek model it is defined only about 0 degrees. Below that the function skips the sky render and writes **only stars** when stars = TRUE.

Value

Either the raw data, or the data is invisibly returned if filename is given. The EXR is written to filename.

Examples

```
sky = generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 12:00:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
```

```
  resolution = 8
)
dim(sky)


# Morning sunrise on spring solstice over Washington, DC with Prague model
generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 06:15:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2,
  hosek = FALSE
) |>
  rayimage::plot_image()

generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 12:00:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2
) |>
  rayimage::plot_image()

generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 18:00:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2
) |>
  rayimage::plot_image()

generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 18:30:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2,
  hosek = FALSE,
  verbose = TRUE
) |>
  rayimage::render_exposure(exposure = 2) |>
  rayimage::plot_image()
```

`generate_stars` *Generate a star-field array aligned with* `generate_sky()`

Description

Render a star map for a given observer location, time, and atmospheric conditions so it can be composited with `generate_sky()`. Returns a `(resolution, 2 * resolution, 4)` array with an opaque alpha channel. An image file is written only when `filename` is supplied.

Usage

```
generate_stars(
  lon,
  lat,
  datetime,
  filename = NA,
  resolution = 2048,
  turbidity = 3,
  ozone_du = 300,
  altitude = 0,
  color = TRUE,
  star_width = 1,
  upper_hemisphere_only = TRUE,
  atmosphere_effects = TRUE,
  number_cores = 1
)
```

Arguments

lon	Observer longitude in degrees (east positive).
lat	Observer latitude in degrees.
datetime	POSIXct timestamp used to compute local sidereal time.
filename	Default NA. Path to an image file to write. If NA, the image array is returned without writing.
resolution	Default 2048. Map half-width; the output image is 2 * resolution by resolution.
turbidity	Default 3.0. Atmospheric turbidity controlling aerosol optical depth for extinction/reddening.
ozone_du	Default 300.0. Column ozone in Dobson Units used in atmospheric absorption.
altitude	Default 0.0. Observer altitude above mean sea level in meters.
color	Default TRUE. If TRUE, render RGB star colors; if FALSE, render monochrome luminance.
star_width	Default 1. Approximate stellar point-spread size in pixels (controls apparent star sharpness).
upper_hemisphere_only	Default TRUE. If TRUE, pixels below the local horizon are suppressed to match generate_sky()'s visible hemisphere.
atmosphere_effects	Default TRUE. If TRUE, apply atmospheric extinction and color shift using turbidity, ozone_du, and altitude.
number_cores	Default 1. Number of CPU threads to use.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions (resolution, 2 * resolution, 4).

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
star_map = generate_stars(
  resolution = 8,
  lon = -77.0369,
  lat = 38.9072,
  datetime = as.POSIXct("2025-03-21 02:20:00", tz = "UTC"),
  atmosphere_effects = FALSE,
  number_cores = 1
)
dim(star_map)


# Note: exposure has been increased for all examples (via white_point) for
# ease of visibility in documentation

# Basic star field over Washington, DC at a fixed time
generate_stars(
  resolution = 400,
  lon = -77.0369,
  lat = 38.9072,
  datetime = as.POSIXct("2025-03-21 02:20:00", tz = "America/New_York"),
  color = TRUE,
  star_width = 1,
  atmosphere_effects = TRUE,
  upper_hemisphere_only = TRUE,
  number_cores = 2
) |>
  rayimage::plot_image()

# Monochrome stars, no atmospheric extinction/reddening, full sphere
generate_stars(
  resolution = 400,
  lon = -122.4194,
  lat = 37.7749,
  datetime = as.POSIXct("2025-06-01 08:00:00", tz = "UTC"),
  color = FALSE,
  star_width = 1,
  upper_hemisphere_only = FALSE,
  atmosphere_effects = FALSE
) |>
  rayimage::plot_image()

# Sharper stars (smaller PSF) with ozone/turbidity and altitude
generate_stars(
  resolution = 400,
  lon = 10,
```

```
  lat = 45,
  datetime = as.POSIXct("2025-12-01 22:00:00", tz = "UTC"),
  star_width = 0.5,
  turbidity = 3.5,
  ozone_du = 320,
  altitude = 1000,
  color = TRUE
) |>
  rayimage::plot_image()
```

`list_sky_data` *List cached sky data files*

Description

Lists files cached by `download_sky_data()` under `tools::R_user_dir("skymodelr", "data")`.

Usage

```
list_sky_data()
```

Value

A data frame with columns `file`, `path`, `size`, and `modified`.

Examples

```
list_sky_data()
```

Index