

Package: raybonsai (via r-universe)

August 28, 2024

Type Package

Title Procedurally Generate and Render 3D Trees

Version 0.1.0

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

Description Generate procedural 3D trees and render them in 3D. Based on the `flametree` package by Danielle Navarro.

Imports dplyr, rayrender, tibble, magrittr, purrr, tidyr

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Repository <https://tylermorganwall.r-universe.dev>

RemoteUrl <https://github.com/tylermorganwall/raybonsai>

RemoteRef HEAD

RemoteSha c21b4c2f88c99c28bdb55cfb89d9e327f3883adb

Contents

generate_tree	1
render_tree	4

Index	6
--------------	----------

generate_tree	<i>Generate Tree</i>
---------------	----------------------

Description

Generates a procedural tree with

Usage

```

generate_tree(
  x = 0,
  y = NULL,
  z = NULL,
  seed = 2000,
  midpoint = TRUE,
  branch_depth = 6,
  branch_scale = c(0.8, 0.9),
  branch_angle = c(-30, 30),
  branch_angle_vert = seq(-45, 45, by = 5),
  branch_split = 2,
  branch_prune_prob = 0,
  branch_color = "#603000",
  branch_radius_shrink = 15,
  leaf_color = "green",
  leaf_depth_start = NULL,
  leaf_size = 0.2,
  leaf_prob = 1,
  scale = 1
)

```

Arguments

x	Default '0'. Either the x-coordinate, or if a length-3 vector the x,y, and z coordinates of the base of the tree.
y	Default 'NULL'. The y-coordinate of the base of the tree. Ignored if the 'x' is a length-3 vector.
z	Default 'NULL'. The z-coordinate of the base of the tree. Ignored if the 'x' is a length-3 vector.
seed	Default '2'. Random seed for generating the tree.
midpoint	Default 'TRUE'. Method of extending branches. If 'FALSE', it grows directly to the next node. Else, it first extends a midpoint given the previous orientation and grows from there to the end point.
branch_depth	Default '6'. Number of branch splits to end tree.
branch_scale	Default 'c(0.8,0.9)'.
branch_angle	Default 'c(-30, 30)'. Horizontal branching angle from previous branch.
branch_angle_vert	Default 'seq(-45,45, by=5)'. Vertical branching angle from previous branch.
branch_split	Default '2'.
branch_prune_prob	Default '0'.
branch_color	Default '#603000'.
branch_radius_shrink	Default '15'. Constant that determines the rate the radius shrinks. Higher values result in less shrinking.

leaf_color Default 'NULL'.
 leaf_depth_start
 Default 'NULL', automatically set
 leaf_size Default '0.01'.
 leaf_prob Default '1'.
 scale Default '1'. Uniformly scale the tree.

Examples

```

library(rayrender)
generate_tree(seed=1) %>%
  render_tree()

#Change the branch angle choices
generate_tree(seed=1,branch_angle_vert = c(-15,15)) %>%
  render_tree()

#Change the horizontal branch angle choices
generate_tree(seed=6,branch_angle = seq(-90,90,by=10)) %>%
  render_tree()

#Increase the number of layers
generate_tree(seed=1,branch_depth = 8) %>%
  render_tree()

#Have the leaves start appearing at branch 6 to fill in the tree
generate_tree(seed=1,branch_depth = 8, leaf_depth_start = 6) %>%
  render_tree()

#Change the color and seed to get a different structure
generate_tree(seed=2,branch_depth = 6, leaf_depth_start = 4, leaf_color="pink") %>%
  render_tree()

#Shorten the branches at each junction by random values
generate_tree(seed=2,branch_depth = 6, leaf_depth_start = 4, leaf_color="pink",
              branch_scale = c(0.5,0.6)) %>%
  render_tree()

#Lengthen the branches at each junction by random values (this results in a wild tree)
generate_tree(seed=2,branch_depth = 6, leaf_depth_start = 4, leaf_color="red",
              branch_scale = c(1.1,1.2)) %>%
  render_tree()

#All angles one sign make the tree lean over, and here we double the size of the leaf
generate_tree(seed=2,branch_depth = 6,
              leaf_color="purple", leaf_size=0.4,
              branch_angle_vert = c(15,5)) %>%
  render_tree()

#Include a random chance to not grow branches

```

```
generate_tree(seed=4,branch_depth = 8, leaf_depth_start = 6, leaf_color="red",
             branch_prune_prob = 0.5) %>%
  render_tree()
```

render_tree

Render Tree

Description

Automatically plots the tree with a camera position and field of view that includes the full model. For more control over the scene, pass the scene to `'rayrender::render_scene()'` and specify the camera position manually. Note: spheres and cylinders in the scene are used to automatically compute the field of view of the scene—adding additional sphere (e.g. with `'rayrender::generate_ground()'`) will change this calculation. Use `'rayrender::render_scene()'` instead if this is a problem.

Usage

```
render_tree(
  scene,
  ground_radius = 10,
  ground = TRUE,
  ground_color1 = "darkgreen",
  ground_color2 = "lightgreen",
  fov = NULL,
  lookfrom = NULL,
  lookat = NULL,
  angle = c(0, 0, 0),
  order_rotation = c(1, 2, 3),
  lights = TRUE,
  lightintensity = 60,
  clamp_value = 10,
  width = 600,
  height = 600,
  ...
)
```

Arguments

scene	Scene of tree model, to be passed to <code>'rayrender'</code> .
ground_radius	Default <code>'10'</code> . Radius of the ground.
ground	Default <code>'TRUE'</code> . Whether to add a grassy ground scene to the tree.
ground_color1	<code>'darkgreen'</code> . Primary ground color.
ground_color2	<code>'lightgreen'</code> . Secondary ground color.
fov	Default <code>'NULL'</code> , automatically calculated. Camera field of view.
lookfrom	Default <code>'NULL'</code> . Camera position. Automatically calculated unless specified.

lookat	Default 'NULL'. Position camera is directed at. Automatically calculated unless specified.
angle	Default 'c(0,0,0)'. Degrees to rotate the tree around the X, Y, and Z axes. If this is a single number, it will be taken as the Y axis rotation.
order_rotation	Default 'c(1,2,3)'. What order to apply the rotations specified in 'angle'.
lights	Default 'TRUE'. If 'FALSE', removes all default lights.
lightintensity	Default '80'. Light intensity.
clamp_value	Default '10'. Amount of clamp the light intensity. Finite values help reduce rendering artifacts, set to 'Inf' to turn off this feature.
width	Default '600'. Width, in pixels, of the rendered image.
height	Default '600'. Height, in pixels, of the rendered image.
...	Other arguments to pass to rayrender::render_scene()

Value

Rayrender scene.

Examples

```
# Generate a basic scene with the default tree.
library(rayrender)

generate_tree() %>%
  render_tree()

#Rotate the whole scene
generate_tree() %>%
  render_tree(angle = c(0,90,0))

#Specify a custom camera position/direction/field of view/aperture
generate_tree() %>%
  render_tree(lookfrom = c(3, 0, 1), lookat = c(0,4,1), fov=30, aperture=1)

#Change the ground color
generate_tree() %>%
  render_tree(ground_color1 = "brown", ground_color2 = "orange")

#Turn off lights and add our own
generate_tree() %>%
  add_object(sphere(x=20,material=light(color="magenta",intensity=400))) %>%
  add_object(sphere(x=-20,material=light(color="lightblue",intensity=400))) %>%
  render_tree(lights = FALSE, clamp_value = 10)
```

Index

`generate_tree`, 1

`render_tree`, 4