. Documentation

Release 0.1.1

Author

December 01, 2015

CONTENTS

Contents:

CHAPTER

ONE

BRANCH3D MODULE

This module contains the Branch class (one branch of the tree) and the Nodes class

class Branch3D.**Branch**(*mesh*, *init_node*, *init_dir*, *init_tri*, *l*, *angle*, *w*, *nodes*, *brother_nodes*, *Nsegments*)

Class that contains a branch of the fractal tree

Parameters

- **mesh** – an object of the mesh class, where the fractal tree will grow
- **init_node** (*int*) – initial node to grow the branch. This is an index that refers to a node in the nodes.nodes array.
- **init_dir** (*array*) – initial direction to grow the branch. In general, it refers to the direction of the last segment of the mother brach.
- **init_tri** (*int*) – the index of triangle of the mesh where the init_node sits.
- **l** (*float*) – total length of the branch
- **angle** (*float*) – angle (rad) with respect to the init_dir in the plane of the init_tri triangle
- **w** (*float*) – repulsitivity parameter. Controls how much the branches repel each other.
- **nodes** – the object of the class nodes that contains all the nodes of the existing branches.
- **brother_nodes** (*list*) – the nodes of the brother and mother branches, to be excluded from the collision detection between branches.
- **Nsegments** (*int*) – number of segments to divide the branch.

child

list

contains the indexes of the child branches. It is not assigned when created.

dir

array

vector direction of the last segment of the branch.

nodes

list

contains the node indices of the branch. The node coordinates can be retrieved using nodes.nodes[i]

triangles

list

contains the indices of the triangles from the mesh where every node of the branch lies.

tri
int

triangle index where last node sits.

growing
bool

False if the branch collide or is out of the surface. True otherwise.

add_node_to_queue(*mesh*, *init_node*, *dir*)
Functions that projects a node in the mesh surface and it to the queue is it lies in the surface.

Parameters

- **mesh** – an object of the mesh class, where the fractal tree will grow
- **init_node** (*array*) – vector that contains the coordinates of the last node added in the branch.
- **dir** (*array*) – vector that contains the direction from the init_node to the node to project.

Returns **success** – true if the new node is in the triangle.

Return type bool

class Branch3D.**Nodes**(*init_node*)
A class containing the nodes of the branches plus some fuctions to compute distance related quantities.

Parameters **init_node** (*array*) – an array with the coordinates of the initial node of the first branch.

nodes
list

list of arrays containing the coordinates of the nodes

last_node
int

last added node.

end_nodes
list

a list containing the indices of all end nodes (nodes that are not connected) of the tree.

tree
scipy.spatial.cKDTree

a k-d tree to compute the distance from any point to the closest node in the tree. It is updated once a branch is finished.

collision_tree
scipy.spatial.cKDTree

a k-d tree to compute the distance from any point to the closest node in the tree, except from the brother and mother branches. It is used to check collision between branches.

add_nodes(*queue*)
This function stores a list of nodes of a branch and returns the node indices. It also updates the tree to compute distances.

Parameters **queue** (*list*) – a list of arrays containing the coordinates of the nodes of one branch.

Returns **nodes_id** – the indices of the added nodes.

Return type list

collision(*point*)
This function returns the distance between one point and the closest node in the tree and the index of the closest node using the collision_tree.

Parameters **point** (*array*) – the coordinates of the point to calculate the distance from.

Returns **collision** – (distance to the closest node, index of the closest node)

Return type tuple

distance_from_node(*node*)
This function returns the distance from any node to the closest node in the tree.

Parameters **node** (*int*) – the index of the node to calculate the distance from.

Returns **d** – the distance between specified node and the closest node in the tree.

Return type float

distance_from_point(*point*)
This function returns the distance from any point to the closest node in the tree.

Parameters **point** (*array*) – the coordinates of the point to calculate the distance from.

Returns **d** – the distance between point and the closest node in the tree.

Return type float

gradient(*point*)
This function returns the gradient of the distance from the existing points of the tree from any point. It uses a central finite difference approximation.

Parameters **point** (*array*) – the coordinates of the point to calculate the gradient of the distance from.

Returns **grad** – (x,y,z) components of gradient of the distance.

Return type array

update_collision_tree(*nodes_to_exclude*)
This function updates the collision_tree excluding a list of nodes from all the nodes in the tree. If all the existing nodes are excluded, one distant node is added.

Parameters **nodes_to_exclude** (*list*) – contains the nodes to exclude from the tree. Usually it should be the mother and the brother branch nodes.

Returns none

CHAPTER

TWO

FRACTAL_TREE_3D MODULE

CHAPTER

THREE

MESH MODULE

This module contains the mesh class. This class is the triangular surface where the fractal tree is grown.

class mesh.**Mesh**(*filename*)

Class that contains the mesh where fractal tree is grown. It must be Wavefront .obj file. Be careful on how the normals are defined. It can change where an specified angle will go.

Parameters **filename** (*str*) – the path and filename of the .obj file.

verts

array

a numpy array that contains all the nodes of the mesh. verts[i,j], where i is the node index and j=[0,1,2] is the coordinate (x,y,z).

connectivity

array

a numpy array that contains all the connectivity of the triangles of the mesh. connectivity[i,j], where i is the triangle index and j=[0,1,2] is node index.

normals

array

a numpy array that contains all the normals of the triangles of the mesh. normals[i,j], where i is the triangle index and j=[0,1,2] is normal coordinate (x,y,z).

node_to_tri

dict

a dictionary that relates a node to the triangles that it is connected. It is the inverse relation of connectivity. The triangles are stored as a list for each node.

tree

scipy.spatial.cKDTree

a k-d tree to compute the distance from any point to the closest node in the mesh.

loadOBJ(*filename*)

This function reads a .obj mesh file

Parameters **filename** (*str*) – the path and filename of the .obj file.

Returns **verts** – a numpy array that contains all the nodes of the mesh. verts[i,j], where i is the node index and j=[0,1,2] is the coordinate (x,y,z). connectivity (array): a numpy array that contains all the connectivity of the triangles of the mesh. connectivity[i,j], where i is the triangle index and j=[0,1,2] is node index.

Return type array

project_new_point(*point*)

This function projects any point to the surface defined by the mesh.

Parameters **point** (*array*) – coordinates of the point to project.

Returns **projected_point** – the coordinates of the projected point that lies in the surface. intriangle (int): the index of the triangle where the projected point lies. If the point is outside surface, intriangle=-1.

Return type array

CHAPTER

FOUR

SPHERE_PARAMETERS MODULE

Created on Tue Nov 24 10:33:00 2015

@author: fsc

class sphere_parameters.**Parameters**

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

b

m

s