Oaktree Manual

March 12, 2013

Contents

Chapter 1

Introduction

Chapter 2

Installation

Chapter 3

Running

Chapter 4

Input

Oaktree input language extends Python. Subroutines and objects related to input processing are listed below.

4.1 SIMULATION

SIMULATION object stores data specific to one simulation.

obj = SIMULATION (outpath, duration, step, cutoff)

- **obj** - SIMULATION object
- **outpath** - output directory path
- **duration** - simulation duration
- **step** - time step
- **cutoff** - cutoff length below which geometrical details are not resolved

4.2 SPHERE

A sphere shape.

obj = SPHERE (center, r, scolor)

- **obj** - SHAPE object
- **center** - tuple (x, y, z) defining the center
- **r** - radius
- **scolor** - integer surface color

4.3 CYLINDER

A cylinder shape.

obj = CYLINDER (base, h, r, scolor)

- **obj** - SHAPE object
- **base** - tuple (x, y, z) defining the base center
- **h** - height along z
- **r** - radius
- **scolor** - integer tuple $(s_{base}, s_{side}, s_{top})$of surface colors

4.4 CUBE

A cube like shape.

obj = CUBE (corner, u, v, w, scolor)

- **obj** - SHAPE object
- **corner** - tuple (x, y, z) defining the minimum coordinate corner
- **u** - length along x
- **v** - length along y
- **w** - length along z
- **scolor** - integer tuple $(s_{xmin}, s_{ymin}, s_{zmin}, s_{xmax}, s_{ymax}, s_{zmax})$ of surface colors

4.5 POLYGON

A shape extruded from a polygon.

obj = POLYGON (polygon, h, scolor)

- **obj** - SHAPE object
- **polygon** - list $[(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)]$ of vertices defining a closed polygon in counter clock-wise order when looking down from a positive z point
- **h** - height along z
- **scolor** - integer tuple $(s_{base}, s_1, s_2, ..., s_n, s_{top})$ of surface colors

4.6 MLS

A shape defined by moving least square fit to oriented points.

obj = MLS (op, r, scolor)

- **obj** - SHAPE object
- **op** - list $[(x_1, y_1, z_1, nx_1, ny_1, nz_1), (x_2, y_2, z_2, nx_2, ny_2, nz_2), ..., (x_n, y_n, z_n, nx_n, ny_n, nz_n)]$ of oriented points
- **r** - smoothing radius
- **scolor** - integer surface color

4.7 COPY

Copy shape.

obj = COPY (shape)

- **obj** - SHAPE object
- **shape** - input SHAPE

4.8 UNION

Set theoretic union of two shapes.

obj = UNION (shape1, shape2)

- **obj** - SHAPE object
- **shape1** - first input SHAPE object
- **shape2** - second input SHAPE object

4.9 INTERSECTION

Set theoretic intersection of two shapes.

obj = INTERSECTION (shape1, shape2)

- **obj** - SHAPE object
- **shape1** - first input SHAPE object
- **shape2** - second input SHAPE object

4.10 DIFFERENCE

Set theoretic difference of two shapes.

obj = DIFFERENCE (shape1, shape2)

- **obj** - SHAPE object
- **shape1** - first input SHAPE object
- **shape2** - second input SHAPE object

4.11 MOVE

Move shape linearly.

MOVE (shape, vector)

- **shape** - input SHAPE object
- **vector** - tuple (u, v, w) defining the translation

4.12 ROTATE

Rotate shape about an axis.

ROTATE (shape, point, vector, angle)

- **shape** - input SHAPE object
- **point** - tuple (x, y, z) defining axis point
- **vector** - tuple (u, v, w) defining axis direction
- **angle** - oriented angle in degrees

4.13 FILLET

Fillet a pair of surfaces.

FILLET (shape, c, r, fillet, scolor)

- **shape** - input SHAPE object
- **c** - centre (x, y, z) of surface picking sphere
- **r** - radius of surface picking sphere
- **fillet** - fillet radius
- **scolor** - integer fillet surface color

4.14 DOMAIN

A domain is created in a simulation.

obj = DOMAIN (simu, shape | label, grid)

- **obj** - DOMAIN object
- **simu** - simulation in which the domain is created
- **shape** - domain shape
- **label** - domain label (default: none)
- **grid** - maximal discretization grid size for this domain (default: none)

Chapter 5

Output

Chapter 6

Viewer

Chapter 7

Tutorials

Chapter 8

Theory