Introduction to Python and pyPLaSM

Computational Visual Design Laboratory
(https://github.com/cvlab) "Roma Tre" University, Italy

Computational Graphics – Lecture 5 – March 11, 2013

Starting Python

Geometric Programming

Starting Python

Install Python, Scipy, and pyOpenGL

- About Python

Install Python, Scipy, and pyOpenGL

- About Python
- Python Scientific Lecture Notes

Install Python, Scipy, and pyOpenGL

- About Python
- Python Scientific Lecture Notes
- The Python Tutorial

Install Python, Scipy, and pyOpenGL

- About Python
- Python Scientific Lecture Notes
- The Python Tutorial
- PyOpenGL: The Python OpenGL Binding

Install Python, Scipy, and pyOpenGL

- About Python
- Python Scientific Lecture Notes
- The Python Tutorial
- PyOpenGL: The Python OpenGL Binding
- Why Python Is the Last Language You'll Have to Learn

Install IPython as your IDE

- The official IPython site

Install IPython as your IDE

- The official IPython site

- Introducing IPython

Getting started

```
paoluzzi$ ipython
Python 2.7.2 (default, Jun 20 2012, 16:23:33)
Type "copyright", "credits" or "license" for more information.

IPython 0.14.dev -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra deta

In [1]:
```

Geometric Programming

The design language PLaSM

The design language PLaSM is a geometry-oriented extension of a subset of FL.

FL Language

FL (programming at Function Level) is a language developed by the Functional Programming Group of IBM Research Division at Almaden (USA) [@BWW90, @BWWLA89]. The FL language, on the line of the Backus' Turing lecture [Backus78] introduces an algebra over programs and has an awesome expressive power.

The design language PLaSM

The design language PLaSM is a geometry-oriented extension of a subset of FL.

FL Language

FL (programming at Function Level) is a language developed by the Functional Programming Group of IBM Research Division at Almaden (USA) [@BWW90, @BWWLA89]. The FL language, on the line of the Backus' Turing lecture [Backus78] introduces an algebra over programs and has an awesome expressive power.

PLaSM Language

PLaSM, (the Programming LAnguage for Solid Modeling) is a "design language" for geometric and solid parametric design, developed by the CAD Group at the Universities "La Sapienza" and "Roma Tre" [PS92, PPV95]. The language is strongly inFLuenced by FL. With few sintactical differences, it can be considered a geometric extension of a FL subset.

PLaSM Language

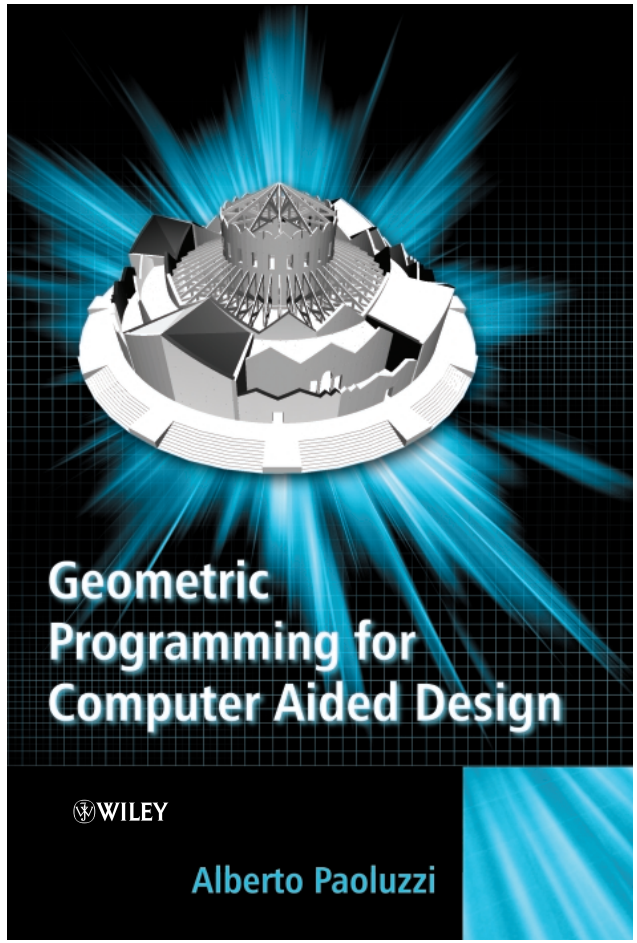
PLaSM Language

PLaSM Language

Paoluzzi, A., Pascucci, V. & Vicentino, M. (1995). Geometric programming: a programming approach to geometric design. *ACM Trans. Graph.* **14** (3), 266–306.

PLaSM Language

Paoluzzi, A., Pascucci, V. & Vicentino, M. (1995). Geometric programming: a programming approach to geometric design. *ACM Trans. Graph.* **14** (3), 266–306.



Motivations for a Python port of PLaSM

- Python: multi-paradigm language with efficient built-in data structures and simple/effective approach to OO programming.

Motivations for a Python port of PLaSM

- Python: multi-paradigm language with efficient built-in data structures and simple/effective approach to OO programming.
- Python's elegant syntax and dynamic typing, and its interpreted nature, make it ideal for scripting and RAD

Motivations for a Python port of PLaSM

- Python: multi-paradigm language with efficient built-in data structures and simple/effective approach to OO programming.
- Python's elegant syntax and dynamic typing, and its interpreted nature, make it ideal for scripting and RAD
- We wished for easy access to Biopython, NumPy, SciPy, Femhub, and the geometry libraries already interfaced with Python

Motivations for a Python port of PLaSM

- Python: multi-paradigm language with efficient built-in data structures and simple/effective approach to OO programming.
- Python's elegant syntax and dynamic typing, and its interpreted nature, make it ideal for scripting and RAD
- We wished for easy access to Biopython, NumPy, SciPy, Femhub, and the geometry libraries already interfaced with Python

The easiest solution?

Pyplasm: Plasm $\rightarrow$ Python

First pyplasm tests

generate and view a geometric object (hpc type) in pyplasm

```
In [1]: from pyplasm import *
Evaluating fenvs.py..
...fenvs.py imported in 0.006975 seconds

In [2]: VIEW(CUBOID([1,4,9]))
```

First pyplasm tests

```
from pyplasm import *
VIEW(CUBOID([1,4,9]))
VIEW(COLOR(BLACK)(CUBOID([1,4,9])))
```

COLOR is a second order function: needs TWO applications

First pyplasm tests

```
a = [[0,0],[4,2],[2.5,3],
 [4,5],[2,5],[0,3],
 [-3,3],[0,0]]
VIEW(POLYLINE(a))
```

First pyplasm tests

```
b = [[0,3],[0,1],[2,2],
 [2,4],[0,3]]
c = [[2,2],[1,3],[1,2],
 [2,2]]
AA(POLYLINE)([a,b,c])
VIEW(STRUCT(AA(POLYLINE)([a,b,c])))

polylines = AA(POLYLINE)([a,b,c])
polygon = SOLIDIFY(STRUCT(polylines))
VIEW(polygon)

cells = SKELETON(1)(polygon)
VIEW(cells)

solid = PROD([polygon, Q(0.5)])
VIEW(solid)

solid = PROD([polygon, QUOTE([0.5,-2.5,0.5])])
VIEW(solid)

complement = DIFFERENCE([ BOX([1,2,3])(solid), solid ])
VIEW(complement)
```

Assignments

- install python

Assignments

- install python
- install scipy

Assignments

- install python
- install scipy
- install pyplasm

Assignments

- install python
- install scipy
- install pyplasm
- explore The Python Tutorial