User manual ScaleCubeSkeleton2D

Jonàs Martínez
jmartinez@lsi.upc.edu

February 13, 2014

Contents

1 Introduction

ScaleCubeSkeleton2D is a software package (`http://lafarga.cpl.upc.edu/projectes/scalecubeskeleton2d`) that computes the L_∞ Voronoi diagram, cube skeleton, and scale cube skeleton of orthogonal polygons and PPM images. For more details see the related papers:

- Martinez, J., Vigo, M., Pla-Garcia, N., and Ayala, D. Skeleton computation of an image using a geometric approach. In Proceedings of Eurographics (2010), pp. 13–16
- Martinez, J., Pla-Garcia, N., and Vigo, M. Skeletal representations of orthogonal shapes. Graphical Models 75 (2013), 189–207

The *2Dskeleton* executable is implemented in C++ and computes the L_∞ Voronoi diagram and cube skeleton of orthogonal polygons.

The bash script *scaleCubeSkeleton2D.sh* implements the 2D scale cube skeleton and the treatment of PPM images. It also uses the *Orto-brep* python software (`http://devel.cpl.upc.edu/orto-brep/`) to extract the boundary of PPM images, and the *BoxUnion3D* C++ sofware (`http://lafarga.cpl.upc.edu/projectes/boxunion3d`)) to compute the union of rectangles.

2 Prerequisites

Currently, *Linux* is the only supported operating system. The required dependencies can be installed with:

```
# sudo apt-get install g++ git libboost-dev python-imaging
```

3 Compilation

The last version of *ScaleCubeSkeleton2D* can be obtained with:

```
# git clone http://movibio.lsi.upc.edu/gitlab/skeleton/scalecubeskeleton2d.git
```

This will clone a GIT repository called scalecubeskeleton2d with the following structure:

```
scalecubeskeleton2d
└── bin → Contains the adapted precompiled binaries of BoxUnion3D
```

```
src → Contains the C++ code of 2Dskeleton
doc → Contains the available documentation
toBrep2D → Contains the adapted Python code of Ortho-brep
models → Contains some example orthogonal polygons and PPM images
Makefile
README
scaleCubeSkeleton2D.sh
```

The 2Dskeleton executable can be compiled with:

```
# cd scalecubeskeleton2d
# make
```

This will generate the executable in the scalecubeskeleton2d directory.

4 Usage

4.1 Program 2Dskeleton

2Dskeleton is able to compute the L_∞ Voronoi diagram and the cube skeleton of orthogonal polygons. The syntaxis is as follows:

```
./2dskeleton −hmvr −i [input file (txt,brep)] −s [output svg file] −p [output ppm file]
```

Other additional parameters are only useful for the script scaleCubeSkeleton2D.sh

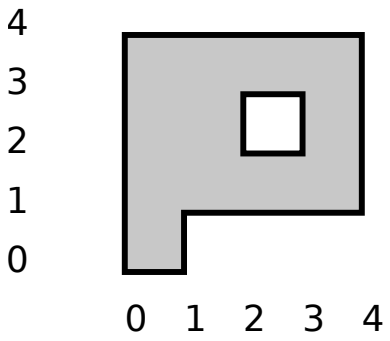
4.1.1 Required parameters

-i [intput file (txt,brep)]
Path to the input file that represents an orthogonal polygon.
Two formats are supported. The .brep is the format of Orto-brep. The .txt, is a simple format that represents the polygon with its boundary contours. Each line of the .txt file represents the two coordinates of a contour vertex. Contours are given by the successive sequence of vertices in the file. The character "h" indicates that a new contour starts. External contours are given in clockwise order. Internal contours, also known as polygon holes, are given in contourclockwise order.
For example the next .txt file:

```
0 0
0 1
1 1
1 4
4 4
```

```
4 0
h
2 2
2 3
3 3
3 2
```

Represents the orthogonal polygon:



4.1.2 Optional parameters

-h
Prints the program syntax.

-m
Computes the L_∞ Voronoi diagram with the less efficient point method.

-v
Computes only the L_∞ Voronoi diagram (by default computes the interior cube skeleton).

-r
The edges of the input orthogonal polygon are also output.

-s [output svg file]
Path to the output Scalable Vector Graphics (SVG) file. Red edges indicate the result of the computation.

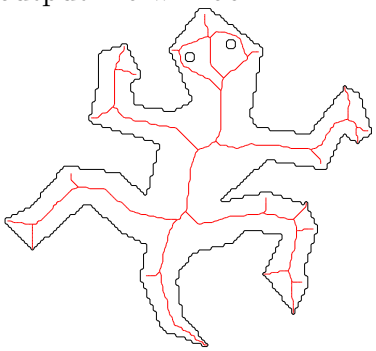
-p [output ppm file]
Path to the output Portable Pixel Map (PPM) file. Red pixels indicate the result of the computation.

4.1.3 Examples

The next call to the 2Dskeleton program:

```
./2Dskeleton -r -i models/lizard.txt -s lizardSkeleton.svg
```

Will compute the interior cube skeleton of an orthogonal polygon. The *lizardSkeleton.svg* output file will be:



4.2 Program scaleCubeSkeleton2D.sh

The script scaleCubeSkeleton2D.sh is able to compute the interior scale cube skeleton of orthogonal polygons and binary images. The syntaxis is as follows:

```
# ./scaleCubeSkeleton2D.sh -hr -i [input file (txt,brep,ppm,pgm,pbm)] -s [output svg file]
  -p [output ppm file] -f [scale]
```

4.2.1 Required parameters

-i [intput file (txt,brep,ppm,pgm,pbm)]
Path to the input file that can be an orthogonal polygon (txt,brep) or a binary image (ppm,pgm,pbm).
The specification of the .txt and .brep files can be found in Section 4.1.1.

4.2.2 Optional parameters

-h
Prints the program syntax.

-r
The edges of the input or scaled (if the parameter -f is enabled) orthogonal polygon are also output.

-s [output svg file]

Path to the output Scalable Vector Graphics (SVG) file. Red edges indicate the result of the computation.

-p [output ppm file]
Path to the output Portable Pixel Map (PPM) file. Red pixels indicate the result of the computation.

-f [scale]
Scale factor to compute the scale cube skeleton. The scale must be greater than 1.

4.2.3 Examples

The next call to the scaleCubeSkeleton2D program:

```
./scaleCubeSkeleton2D -i models/upc.ppm -p upcori.ppm
```

Will compute the interior cube skeleton of a binary image, which is converted to an orthogonal polygon.
Another call to the scaleCubeSkeleton2D program could be:

```
./scaleCubeSkeleton2D -i models/upcscale.ppm -p upcscale.ppm -f 1.26
```

Which will compute the interior **scale** cube skeleton of a binary image. The *upcori.ppm* output file will be (left figure) and the *upcscale.ppm* output file (right figure) will be: