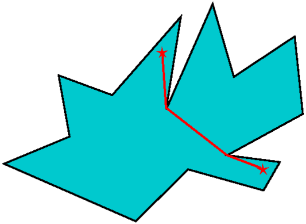

Random Polygons

2. Zwischenpräsentation

Freie Universität Berlin

6th December 2011

Goals

- Testing framework for algorithms related to random polygons
- Generate random polygons (fair, representative)
- Run shortest path algorithm on polygon
- Statistical analysis

2. Milestone

- accomplish 6.12.-12.12.
- all polygon generation algorithms
- shortest path algorithm
- GUI basic version

Geometry Framework

- needed for some polygon generation algorithms
- existing:
    - needed accuracy not given
    - too big
    - partly inconsistent
    -
- therefore small own implementation

Generation algorithms

Permute and Reject by Auer and Held

input: size n or n points

1. generate points, if necessary
2. permute points
3. test if polygon simple. if not, continue with ii.

- simple to implement
- generates all simple polygons with uniform distribution
- runntime depends on polygons needed to be generated to encounter simple polygon (max. n=15 for fast results)
- not suited for practical use

Generation algorithms

2opt.-Moves by Auer and Held

input: size n or n points

i. generate random permutation of n points
ii. for every self intersection of two edges $\overline{ab}$ and $\overline{cd}$
    a. remove $\overline{ab}$ and $\overline{cd}$
    b. insert $\overline{ac}$ and $\overline{bd}$

- needs special treatment for polygons not in general position
- generates all polygons, but not with uniform distribution

Space Partitioning by Auer and Held

input: set S of n points

- i. choose points $s_f, s_t \in S$ at random
- ii. draw line $\overline{s_f s_t} \rightarrow$ 2 sets
- iii. recursively for all subsets S', s'_f first, s'_t last point:
  - a. if S' only consists of s'_f, s'_t, return $\overline{s'_f s'_t}$
  - b. choose s' from S' at random
  - c. draw random line through s' intersecting $\overline{s'_f s'_t}$, thereby creating subsets S", S"', first and last point s'_f, s' and s', s'_t

- ▶ generates not every possible simple polygon

Generation algorithms

Incremental Construction & Backtracking by Auer and Held

input: n points

i. set of all possible edges, all unmarked, randomly choose current point s

ii. recursively for current point:

   a. add next possible unmarked edge to polygon, mark all intersecting edges
   b. backtrack if no unmarked edges left and incompleted polygon

- speed depends on backtracking
- complex to optimize backtracking
- currently not suited for practical use

Generation algorithms

Generating random Polygons by O'Rourke and Virmani

input: size n, radius circle, max. speed, steps t

i. generate random points on circle → regular polygon
ii. t times:
    a. move each vertex with random speed and direction
    b. test if move violates 2 conditions:
        - polygon is simple
        - vertices in bounding region
    c. discard last step if violation of conditions

- still not accessible from GUI
- seems to generate specific class of polygons
- interesting for statistical analysis

Generation algorithms (cont.)

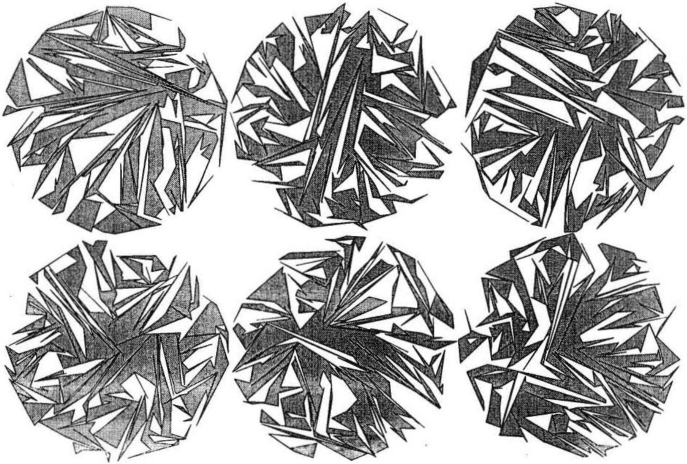

Figure 6: $n = 360$, $t = 1000$, six different random seeds.

8

Generation algorithms

Random Polygon Algorithm by Dailey and Whitfield

input: size n

i. generate 3 random points → random n-gon P size 3
ii. randomly choose and discard edge $\overline{ab}$
iii. determine region P' in polygon visible from $\overline{ab}$
iv. randomly choose point c in P'
v. add edges $\overline{ac}$, $\overline{bc}$ to polygon P

- main reason for geometry framework
- nontrivial problem to determine visible region P'
- most complex of all generation algorithms

Nontrivial resulting algorithmic problems

- order of points representing polygon: cc-wise
- triangularization
- random point in polygon
- intersection line with polygon
- surface area of polygon

Demonstration GUI

- basic version
- next important step: better way to pass parameters

Again Milestones

This Milestone:

- 2 polygon generation algorithms missing
- shortest path generator missing
- possible in 1 week

Next Milestone:

- history objects
- step-by-step visualization
- statistic backend