Generation of Random Digital Curves Using Combinatorial Techniques

Apurba Sarkar[1] Arindam Biswas[1] Mousumi Dutt[2] and Arnab Bhattacharya[3]

[1]Indian Institute Of Engineering Science and Technology, Shibpur.

[2]Indian Institute Of Information Technology, Kalyani.

[3]Indian Institute Of Technology Kanpur.

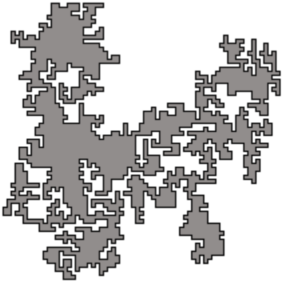
CALDAM 2015

February 9, 2015

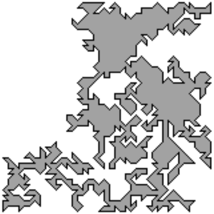
Outline

Objective

- To generate 4- and 8-Connected Random Digital Curves.



(a) 4-connected curve



(b) 8-connected curve

Figure 1: Random Digital Curves on a canvas of size 200 $\times$ 200.

Preliminaries

Definition

(Digital plane, digital point) A digital plane is the set of all points having integer coordinates. A point in the digital plane is called a digital point.

Definition

A digital grid (henceforth referred simply as a grid) $\mathcal{G}$ consists of a set $\mathcal{H}$ of equidistant horizontal (digital) grid lines and a set $\mathcal{V}$ of equidistant vertical (digital) grid lines.

Definitions and Preliminaries

Definition

(Simple closed curve) A closed curve is a curve with no endpoints and which completely encloses an area. A closed curve is simple if it does not cross itself.

Definition

(k-connectedness) Two points p and q are said to be k-connected ($k = 4$ or 8) in a set S if and only if there exists a sequence $\langle p = p_0, p_1, \ldots, p_n = q \rangle \subseteq S$ such that $p_i \in N_k(p_{i-1})$ for $1 \leqslant i \leqslant n$. The 4-neighborhood of a point (x, y) is given by $N_4(x, y) = \{(x', y') : |x - x'| + |y - y'| = 1\}$ and its 8-neighborhood by $N_8(x, y) = \{(x', y') : \max(|x - x'|, |y - y'|) = 1\}$

Basic principle

- The incoming and outgoing direction of the curve at grid points separated by unit grid length will never be the same.
- Movements in opposite direction are always permissible.

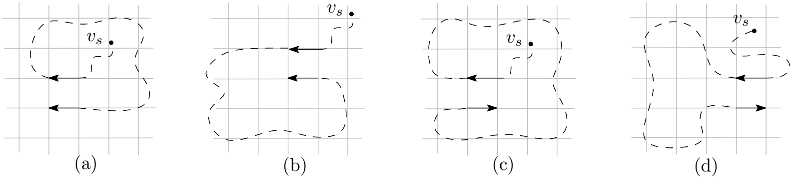


Figure 2: (a) and (b): curves are in the same direction, start of $\mathcal{C}$ is unreachable, (c) and (d): in opposite direction, $\mathcal{C}$ can reach its start.

Generation of random curve

- $v_s =$ the start point chosen randomly, with a timestamp of $t_s = 0$.
- $S=$ set of permissible directions from the current point v_c.
- Computation of S is based on the occupancy and orientation of $\mathcal{C}$ at the earliest visited grid point amongst the **designated** neighbours of v_c denoted by V_d.

Generation of random curves ...

Algorithm

1. Choose the start point v_s randomly;
2. $v_c = v_s, \ t_s = 0;$
3. **do**
4. Compute the set S based on V_d;
5. Randomly pick next direction d_c from S;
6. Advance to the next point v_n along d_c;
7. Update v_c i.e.
$t_n = t_c + 1; \ v_c = v_n;$
8. **while**$(v_c \neq v_s);$

Determining S

1. Depending on the occupancy and orientetaion of vertices in V_d there are three cases.

   Case I: No vertices in V_d are visited.
   Case II: At least one vertex in V_d is visited and let $v_m \in V_d$ has the lower timestamp, v_{m+1}, the grid point next to v_m, lie in N.
   Case III: At least one vertex in V_d is visited and let $v_m \in V_d$ has the lower timestamp, v_{m+1}, the grid point next to v_m, does not lie in N.

Determining S ...

1. To compute S, an array **b** of size 8 is maintained.
2. 8 locations in the array **b** mean 8 directions.
3. **b[i]=0** means direction i is not permissible and **b[i]=1** means direction i is permissible.

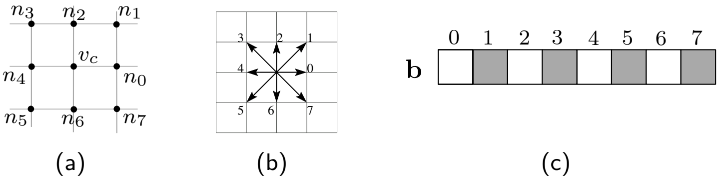


Figure 3: (a) 8 neighbours, (b) 8 directions, and (c) Array **b**

1. For generation of 4-connected random digital curves, the neighbors v_l and v_r of v_c are considered to find S.
2. So, the **designated** set of vertices consists of v_l and v_r.

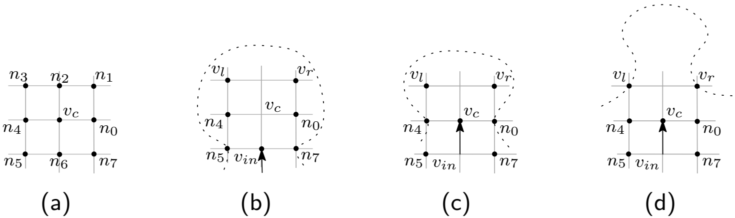


Figure 4: (a) 8 neighbours, (b) trap involving n_5 and n_7, (c) n_0 and n_4, and (d) v_l and v_r

Case I

1. Neither v_l nor v_r is visited.

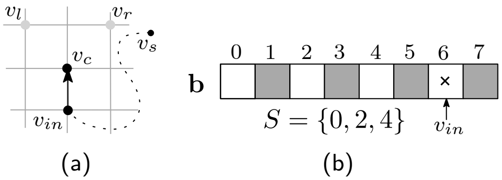
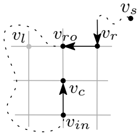


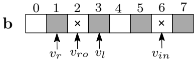
Figure 5: Case I: (a) None of v_l and v_r visited (b) Direction array **b**

Case II

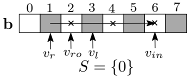
1. At least one in $\{v_l, v_r\}$ is visited and the vertex with the lowest timestamp in $\{v_l, v_r\}$ has its next vertex in N.



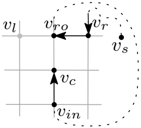
(a) v_r is visited



(b) Initial marking of **b**



(c) Final marking of **b**

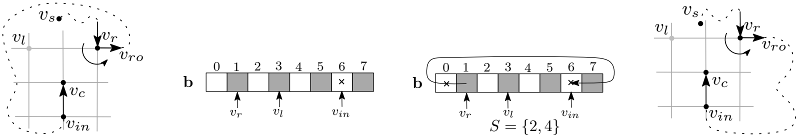


(d) v_r is visited

Figure 6: Illustration of Case II for 4-connected curve

Case III

1. At least one of $\{v_l,\ v_r\}$ is visited and let v_m has the lowest timestamp and $v_{m+1}, \notin N$.



(a) v_r is visited (b) Initial marking of **b** (c) Final marking of **b** (d) v_r is visited

Figure 7: Illustration of Case III for 4-connected curve

1. The curve can move in diagonal direction also.
2. Depending on the incoming direction d_{ci} to v_c, there are two cases
   - If d_{ci} is axis parallel ($d_{ci} \in \{0, 2, 4, 6\}$), $V_d = \{v_l, v_f, v_r\}$.
   - If d_{ci} is diagonal ($d_{ci} \in \{1, 3, 5, 7\}$), $V_d = \{v_{fl}, v_{fr}\}$.

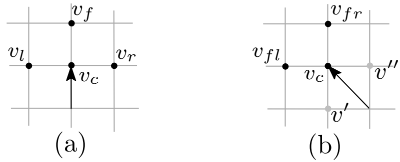


Figure 8: Two cases of 8-connected curves, a) d_{ci} is axis parallel, b) d_{ci} is diagonal

d_{ci} is axis parallel

Case I:

1. None of V_d is visited.

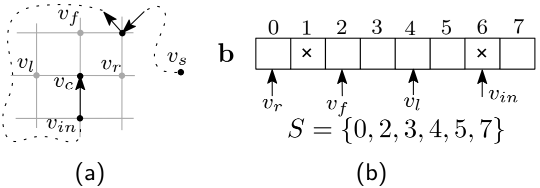


Figure 9: Case I: (a) None of v_l, v_f, and v_r visited (b) Direction array **b**

d_{ci} is axis parallel ...

Case II

1. At least one of V_d is visited and v_m has the lowest timestamp amongst them and its next vertex, say v_{m+1}, is in N.

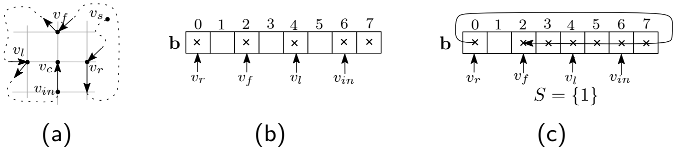
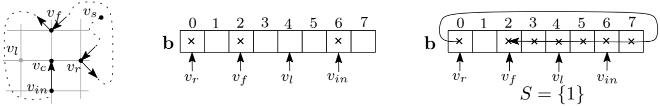


Figure 10: (a) All three v_l, v_f, v_r are visited, (b) Visited neighbors marked, and (c) Permissible set after traversal.

d_{ci} is axis parallel ...

Case III

1. At least one of V_d is visited and let v_m has the lowest timestamp and $v_{m+1}, \notin N$.



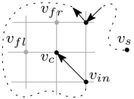
(a) (b) (c)

Figure 11: (a) Case III, (b) Initial marking, and (c) Marking after traversal.

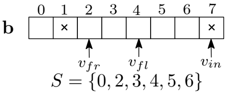
d_{ci} is diagonal

Case I

1. None of V_d is visited.
2. The curve can move in any direction except in the directions of visited neighbors, i.e., n_1 and n_7.



(a) Case I



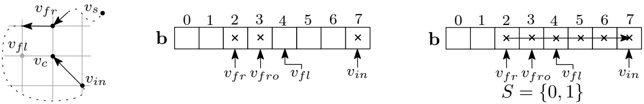
(b) Initial marking

Figure 12: None of v_{fl}, v_{fr} are visited

d_{ci} is diagonal ...

Case II:

1. At least one of V_d is visited and let v_m has the lowest timestamp amongst them and its next vertex, say v_{m+1}, is in N.



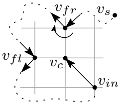
(a) Case II (b) Initial marking (c) Marking after traversal

Figure 13: v_{fr} is visited

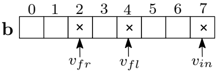
d_{ci} is diagonal ...

Case III:

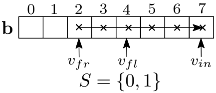
1. At least one of V_d is visited and let v_m has the lowest timestamp and $v_{m+1}, \notin N$.



(a) Case III



(b) Initial marking



(c) Marking after traversal

Figure 14: v_{fr}, v_{fl} both are visited

Boundary condition:

1. The turn $\mathcal{C}$ takes when it meets the boundary for the first time it should take the same turn in all subsequent cases whenever it meets the boundary again.

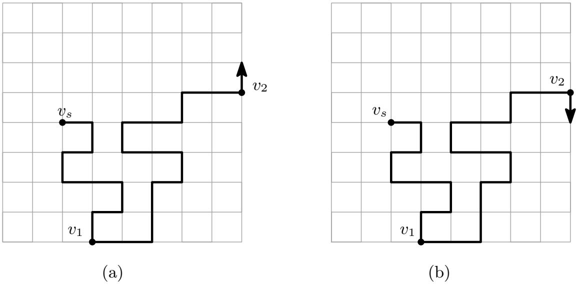


Figure 15: Explanation of boundary condition

Time complexity:

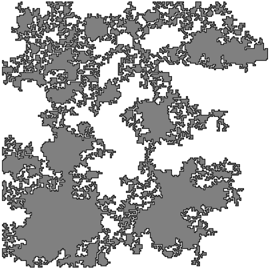
1. A constant number of grid points are checked to determine the next direction from a grid point lying on $\mathcal{C}$.
2. The algorithm requires no backtracking.
3. Thus the algorithm is linear in the number of grid points on the length of the generated curve, given by $O(|\mathcal{C}|/g)$.

Demonstration

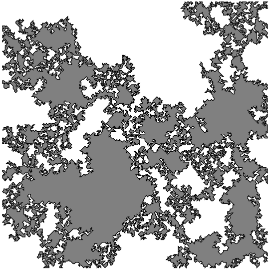
Demonstration

Results:

1. Some random curves generated by the algorithm.



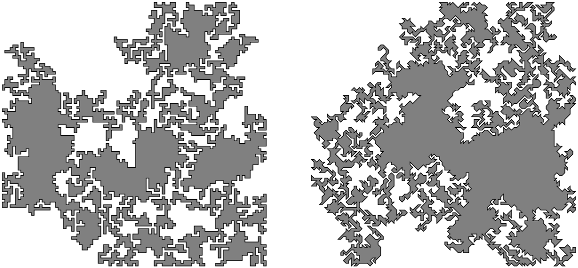
$g = 1$ and $\#$ vertices $= 7308$



$g = 1$ and $\#$ vertices $= 8841$

Figure 16: Instances of 4-connected and 8-connected digital random curves on a canvas of size 200×200.

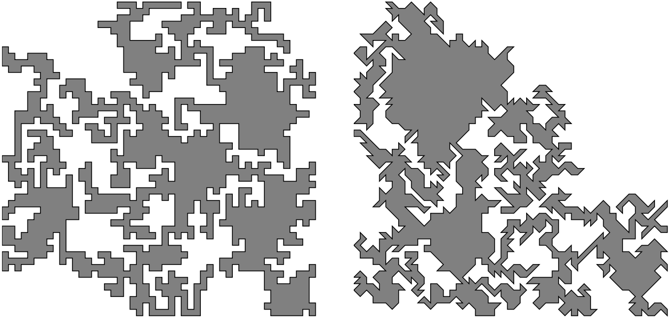
Results:



$g = 2$ and # vertices $= 2310$ $g = 2$ and # vertices $= 2726$

Figure 17: Instances of 4-connected and 8-connected digital random curves on a canvas of size 200×200.

Results:



$g = 4$ and # vertices $= 806$ $g = 4$ and # vertices $= 765$

Figure 18: Instances of 4-connected and 8-connected digital random curves on a canvas of size 200×200.

Test of Randomness

1. Randomness of the generated curves are tested using one-sample runs test.
2. We select 100 equidistant points on $\mathcal{C}$.
3. 0 is assigned for a point lying on a horizontal grid line and 1 is assigned when it lies on a vertical line.
4. A sequence of 0 and 1 of length 100 is generated.
5. Let n_1 be the number of 0s, n_2 be the number of 1s in a run, and r be the number of such runs.

Runs test

$\#\mathcal{C}$	n_1	n_2	r	μ_r	σ_r	z	α in %
$\mathcal{C}_1$	46	54	51	50.68	4.9425	0.0647	94.70
$\mathcal{C}_2$	53	47	50	50.82	4.9565	−0.1654	86.20
$\mathcal{C}_3$	45	55	51	50.50	4.9244	0.1015	92.01
$\mathcal{C}_4$	51	49	51	50.98	4.9726	0.0040	100.00
$\mathcal{C}_5$	62	38	48	48.12	4.6852	−0.0256	98.00
$\mathcal{C}_6$	48	52	51	50.92	4.9666	0.0167	99.01
$\mathcal{C}_7$	49	51	49	50.98	4.9726	−0.3981	69.40
$\mathcal{C}_8$	62	38	49	48.12	4.6852	0.1878	85.14
$\mathcal{C}_9$	49	51	51	50.98	4.9726	0.0040	100.00
$\mathcal{C}_{10}$	53	47	53	50.82	4.9565	0.4398	66.02
$\mathcal{C}_{11}$	54	46	50	50.68	4.9425	−0.1375	89.00
$\mathcal{C}_{12}$	48	52	52	50.92	4.9666	0.2174	83.11
$\mathcal{C}_{13}$	60	40	49	49.00	4.7736	0.0000	100.00
$\mathcal{C}_{14}$	61	39	50	48.58	4.7314	0.3001	76.41
$\mathcal{C}_{15}$	52	48	50	50.92	4.9666	−0.1852	85.50
$\mathcal{C}_{16}$	50	50	51	51.00	4.9746	0.0000	100.00
$\mathcal{C}_{17}$	44	56	50	50.28	4.9023	−0.0571	96.01
$\mathcal{C}_{18}$	53	47	52	50.82	4.9565	0.2380	81.02
$\mathcal{C}_{19}$	44	56	49	50.28	4.9023	−0.2611	79.50
$\mathcal{C}_{20}$	42	58	51	49.72	4.8460	0.2641	79.32

Table 1: Runs test statistic for 20 random curves: $n_1 = \#$ 0s, $n_2 = \#$ 1s, $r = \#$ runs, μ_r and σ_r are mean and standard error of the r statistic respectively, and α is the significance level corresponding to the z value.

Conclusion

1. The algorithm runs in linear-time.
2. Uses combinatorial techniques and does not backtrack.
3. Can be further tuned to generate random paths inside a digital object.

References I

T. Auer and M. Held.
Heuristics for the generation of random polygons.
In *Proc. Canadian Conference on Computational Geometry*, pages 38—-44, 1996.

P. Bhowmick and B. B. Bhattacharya.
Fast polygonal approximation of digital curves using relaxed straightness properties.
IEEE Trans. PAMI, 29(9):1590–1602, 2007.

P Bhowmick, O Pal, and R Klette.
Linear-time algorithm for the generation of random digital curves.
In *Proceedings of the 2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, pages 168–173, 2010.

D. Dailey and D. Whitfield.
Constructing random polygons.
In *Proceedings of the 9th ACM SIGITE Conf, Information Tech. Education*, pages 119–124, 2008.

P. Epstein.
Generating geometric objects at random.
Master's thesis, CS Dept., Carleton University, Canada, 1992.

References II

R. Klette and A. Rosenfeld.
Digital Geometry: Geometric Methods for Digital Picture Analysis.
Morgan Kaufmann, San Francisco, 2004.

Richard I. Levin.
Statistics for Management.
Prentice Hall India, 1986.

Robert D. Mason.
Essentials of Statistics.
Prentice-Hall, 1986.

I. Rennesson, R. Luc, and J. Degli.
Segmentation of discrete curves into fuzzy segments.
Electronic Notes in Discrete Mathematics, 12:372–383, 2003.

J. Rourke and M. Virmani.
Generating random polygons, TR:011, CS Dept.
Smith College, Northampton, 1991.

References III

A. P. Tomas and A. L. Bajuelos.
Generating random orthogonal polygons.
In *Postconference Proc. of CAEPIA-TTIA'2003, LNAI, Springer-Verlag*, 2003.

C. Zhu, G. Sundaram, J. Snoeyink, and J. S. B. Mitchell.
Generating random polygons with given vertices.
CGTA, 6:277–290, 1996.

Thank You

Runs test

- Mean and standard error of the r-statistic are given by $\mu_r = \frac{2n_1 n_2}{(n_1+n_2)} + 1$ and $\sigma_r = \sqrt{\frac{2n_1 n_2(2n_1 n_2 - n_1 - n_2)}{(n_1+n_2)^2(n_1+n_2-1)}}$ are calculated.
- The corresponding z-value is computed as $z = \frac{r-\mu_r}{\sigma_r}$.
- The sampling distribution is approximated by normal distribution.
- From the table of standard normal probability distribution the significance level α is determined as shown in the following table.