DRAWING SPACE-FILLING CURVES IN LOGO

Vladimir Batagelj
Department of mathematics, FMF, University of Ljubljana
Jadranska 19, 1000 Ljubljana, Slovenia
e-mail: `vladimir.batagelj@uni-lj.si`

Abstract

Several space-filling and related curves are described by means of ”simultaneous grammars”. From these descriptions it is very easy to obtain short procedures in logo for drawing the curves.

Keywords

`Space-filling curves, simultaneous grammar, logo, PostScript, recursion`

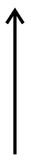
1 Introduction

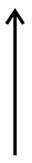
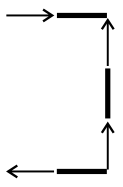
Several procedures for drawing particular space-filiing curves have been proposed in the literature [2, 7, 10, 12, 15, 16, 13]. In this paper we propose to describe space-filling and related curves by means of ”simultaneous (nonlinear) grammars”. By nonlinear we mean that objects described by the grammar are not linear strings but spatial (planar) configurations; and by simultaneous we mean that at each derivation step we choose a rule and apply it simultaneously to all ”nonterminal parts” of the object. For our purposes this intuitive, informal notion of simultaneous grammars is sufficient. The rules governing the growth of a considered curve reflect the recursive patterns that can be observed in the examples of the curve. From this grammatical description of the curve it is very easy to obtain the corresponding drawing procedure.

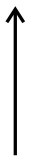
Let us work out an example in details.

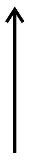
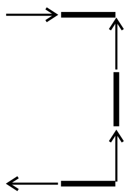
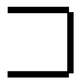
2 Hilbert curve

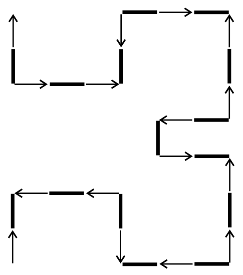
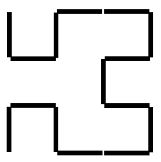
The nonterminals are represented by an arrow and the terminals by heavy lines which remain a permanent part of the objects. The axiom AH and the rules RH1 and RH2 should be understood as embedded in the plane.

AH. 

RH1.  ::= 

RH2.  ::=

We start with AH  by RH1 we obtain  and by RH2 we obtain H_1  , but by RH1

 and by RH2 we obtain H_2  etc.

It is evident that we use the rule RH2 only once – at the last step of a derivation. Let $hi(n)$ be the procedure to draw the n-th Hilbert curve H_n and $ih(n)$ the procedure to draw H_n in the opposite direction. Then we can easily translate our description into the following logo commands:

```
TO Hi :n
  IF :n = 0 [ STOP ]
  RT 90 Ih :n - 1 FD :h LT 90 Hi :n - 1 FD :h
  Hi :n - 1 LT 90 FD :h Ih :n - 1 RT 90
END

TO Ih :n
  IF :n = 0 [ STOP ]
  LT 90 Hi :n - 1 FD :h RT 90 Ih :n - 1 FD :h
  Ih :n - 1 RT 90 FD :h Hi :n - 1 LT 90
END
```

Because `RT` $\alpha =$ `LT` $-\alpha$ we can, introducing another parameter `:d` and denoting `H 1` $\equiv$ `Hi` and `H -1` $\equiv$ `Ih`, combine these two commands into a single command:

```
TO H :d :n
  IF :n = 0 [ STOP ]
  RT :d*90 H (-:d) :n - 1 FD :h LT :d*90 H :d :n - 1 FD :h
  H :d :n - 1 LT :d*90 FD :h H (-:d) :n - 1 RT :d*90
END
```

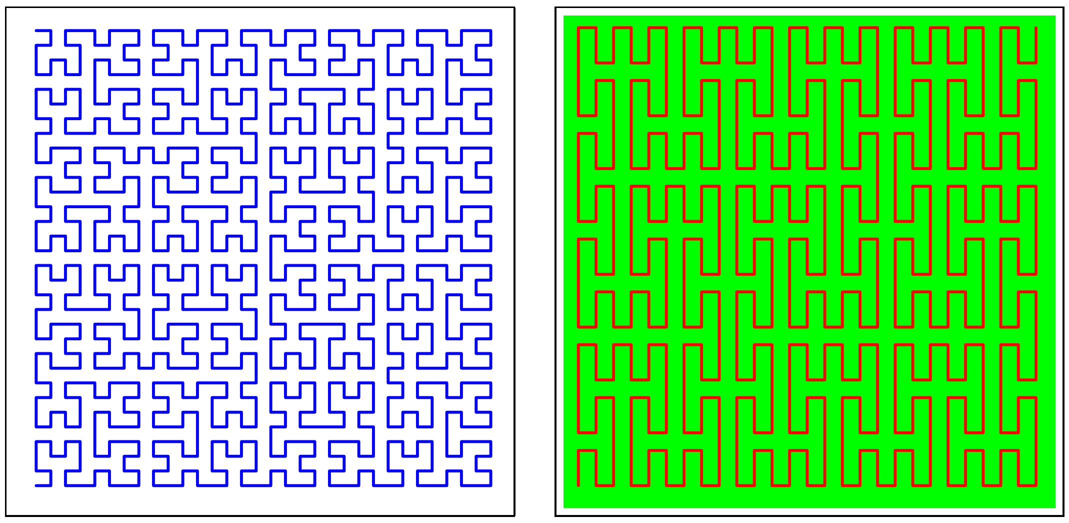


Figure 1: Hilbert curve and Peano curve

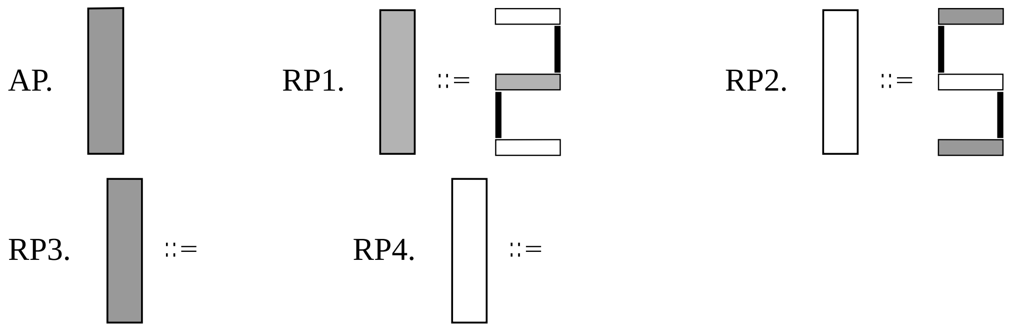
or finally, to avoid multiplication

```
TO Hilb :n :a :h
  IF :n = 0 [ STOP ]
  RT :a Hilb :n - 1 (-:a) :h FD :h LT :a Hilb :n - 1 :a :h FD :h
  Hilb :n - 1 :a :h LT :a FD :h Hilb :n - 1 (-:a) :h RT :a
END

TO Hilbert
  PU SETPOS [-150 -150] PD SETPC [000 000 255]
  Hilb 5 90 10
END
```

In a similar way we can develop drawing procedures also for the following curves. In the examples MicroSoft Windows Logo [11] is used. The PostScript pictures of space-filling curves, presented in figures, were produced by Logo2PS [1].

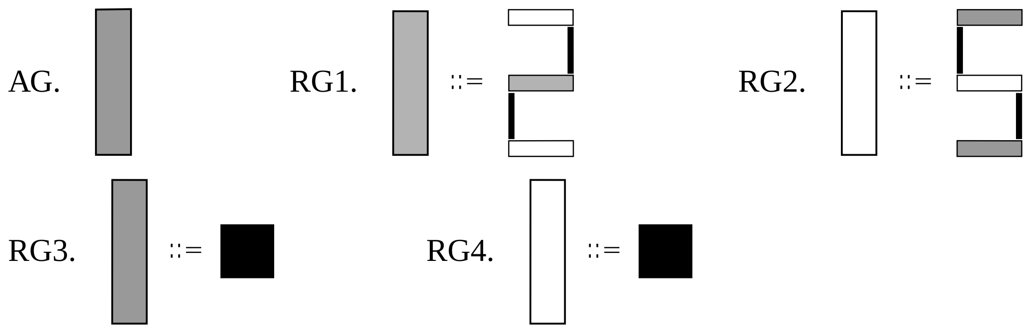
3 Peano curve



```
TO Pean :n :a :h
  IF :n = 0 [STOP]
  RT :a Pean :n - 1 (-:a) :h FD :h Pean :n - 1 :a :h
  FD :h Pean :n - 1 (-:a) :h LT :a
END

TO Peano
  PU SETPOS [ -150 -160 ] PD SETPC [255 000 000]
  SetScreenColor [000 255 000] Pean 6 90 12
END
```

4 Grate curve



```
TO TWO :a :c :w
  IF :c < 1 [ STOP ]
  RT :a FD 1 RT :a FD :w LT :a IF :c > 1 [FD 1]
  LT :a FD :w TWO :a :c - 2 :w
END

TO Square :a :h :w
  FD :w TWO :a :h - 1 :w
END

TO Cag :n :a :w :h
  IF :n = 0 [ Square :a :h :w STOP ]
  RT :a Cag :n - 1 (-:a) :h/4 :w FD :h/8
  Cag :n - 1 :a :h/4 :w FD :h/8
  Cag :n - 1 (-:a) :h/4 :w LT :a
END

TO Cage
  PU SETPOS [ -160 -160 ] PD SETPC [0 0 0]
  SetScreenColor [255 100 100] Cag 4 90 320 320
END
```

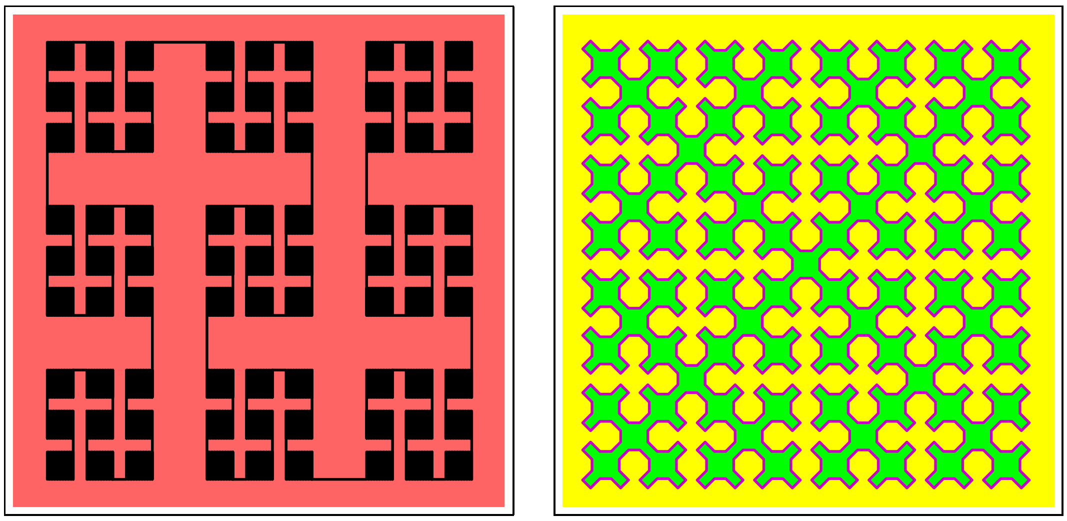
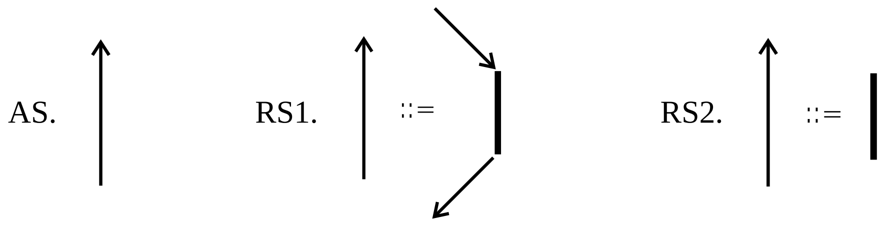


Figure 2: Grate curve and Sierpinski curve

5 Sierpinski curve



```
TO Sierp :n :a :h :k
  IF :n = 0 [ FD :k STOP ]
  RT :a Sierp :n - 1 (-:a) :h :k LT :a FD :h
  LT :a Sierp :n - 1 (-:a) :h :k RT :a
END

TO Sierpinski
  PU SETPOS [ -160 -170 ] PD MAKE "h 12/sqrt 2
  SETPC [200 000 200] SetScreenColor [255 255 000]
  REPEAT 4 [ Sierp 7 45 :h 10 RT 45 FD :h RT 45 ]
  SetFloodColor [000 255 000] PU SetPos [-155 -170] PD FILL
END
```

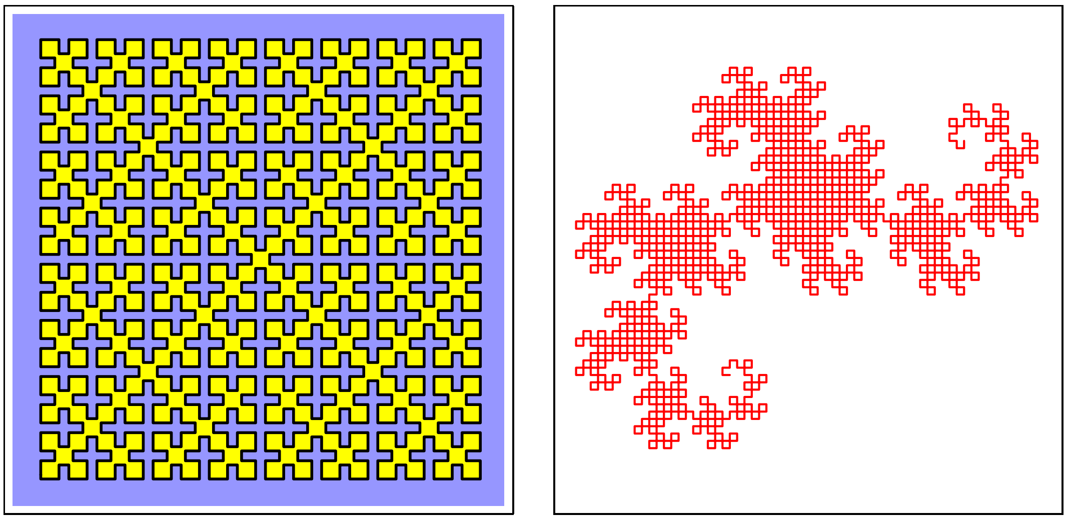
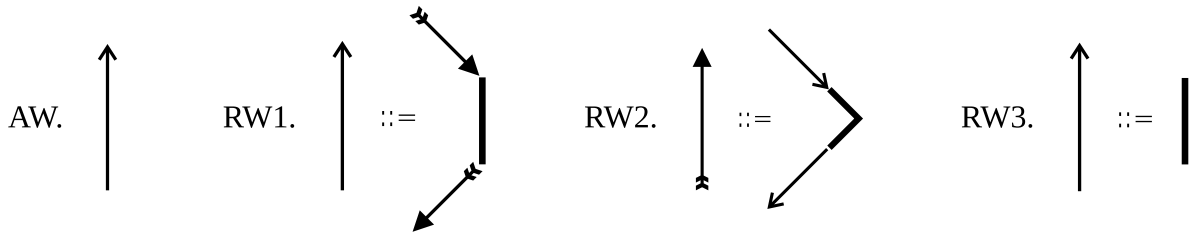


Figure 3: Wirth curve and Dragon curve

6 Wirth curve

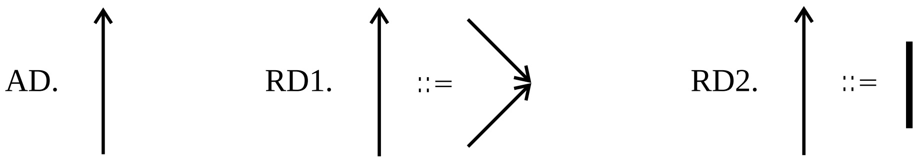


```
TO wi :n :a :h :k
  IF :n = 0 [ FD :h STOP ]
  RT :a iw :n (-:a) :h :k LT :a FD :h
  LT :a iw :n (-:a) :h :k RT :a
END

TO iw :n :a :h :k
  RT :a wi :n - 1 (-:a) :h :k FD :k LT 2 * :a
  FD :k wi :n - 1 (-:a) :h :k RT :a
END

TO Wirth
  PU SETPOS [ -155 -153 ] PD
  SETPC [0 0 0] SetScreenColor [150 150 255]
  REPEAT 4 [ wi 4 45 7 3 FD 3 RT 90 FD 3 ] PU
  SetFloodColor [255 255 0] PU SetPos [-150 -150] PD FILL
END
```

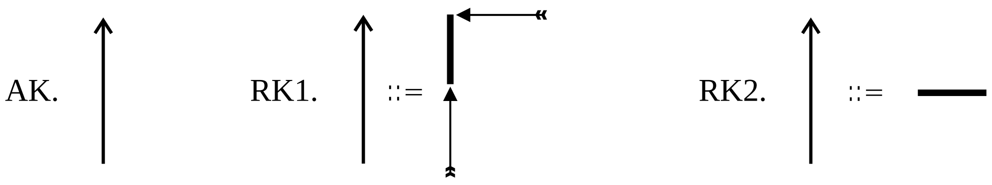
7 Dragon curve

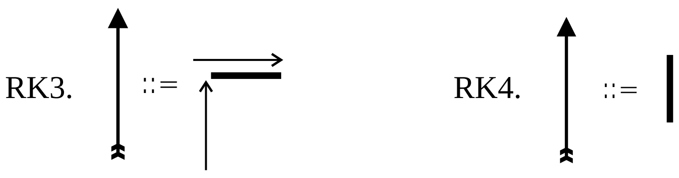


```
TO Drag :n :a :h
  IF :n < 1 [ FD :h STOP ]
  Drag :n - 1 90 :h RT :a Drag :n - 1 -90 :h
END

TO Dragon
  PU SETPOS [ -60 -100 ] PD LT 90
  SETPC [255 000 000] SetPenSize [2 2]
  Drag 11 90 7
END
```

8 Knuth curve





```
TO Knu :n :a :t :h
  IF :n = 0 [ RT 45 + :t FD :h LT 45 + :t STOP ]
  RT 2 * :t + :a Knu :n - 1 2 * :t (-:t) :h
  RT 45 - 3 * :t - :a FD :h  LT 45 - :t + :a
  Knu :n - 1 0 (-:t) :h RT :a
END

TO Knuth
  PU SETPOS [ 250 -130 ] PD LT 90
  SETPC [255 255 255] SetScreenColor [000 000 000]
  Knu 9 -90 45 8
END
```

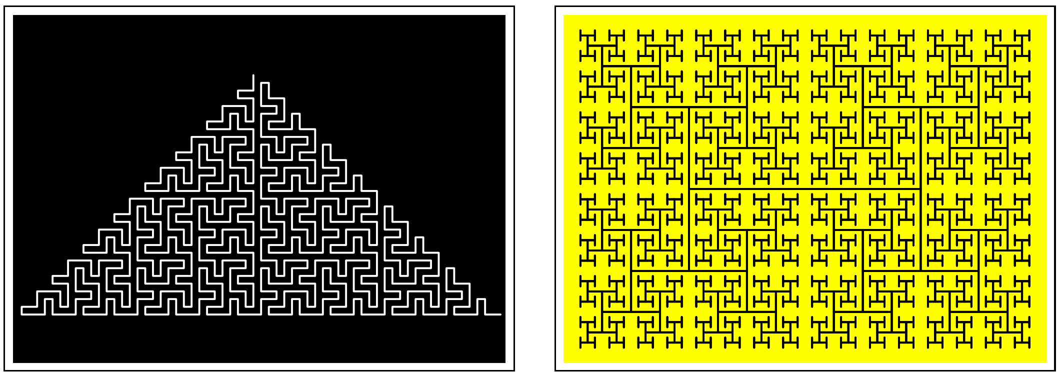
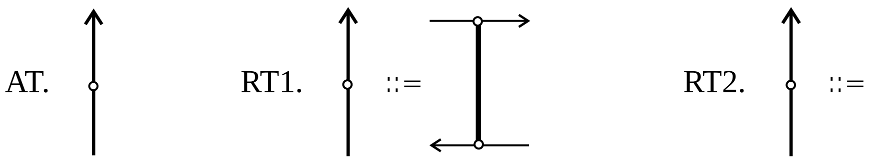


Figure 4: Knuth curve and Antenna tree

9 Antenna tree



```
TO Tr :n :h :q
  IF :n = 0 [ STOP ]
  FD :h LT 90  Tr :n - 1 :q * :h :q LT 90 FD 2 * :h  LT 90
  Tr :n - 1 :q * :h :q LT 90 FD :h
END

TO Tree
  HOME RT 90 SETPC [0 0 0] SetScreenColor [255 255 000]
  Tr 10 120 1 / sqrt 2
END
```

A MSWLogo source file for space-filling curves is available on:
`http://vlado.fmf.uni-lj.si/educa/logo/ex/mix/recurves.lgo`

References

[1] V. Batagelj: *Logo to PostScript.* Paper prepared for Eurologo'97, Ljubljana 1997.

[2] A.J. Cole: *A note on space filling curves.* Software – Practice and Experience, **13**(1983), 1181-1189.

[3] A.J. Cole: *A note on Peano Polygons and Gray Codes.* International Journal of Computer Mathematics, **18**(1985), 3-13.

[4] C. Davis, D.E. Knuth: *Number Representations and Dragon Curves, I-II.* Journal of Recreational Mathematics, **3**(1970), 66-81; **3**(1970), 133-149.

[5] F.M. Dekking, M. Mendes France, A. van der Poorten: *Folds !.* The Mathematical Inteligencer, **4**(1982), 130-138; **4**(1982), 173-181; **4**(1982), 190-195.

[6] F.M. Dekking: *Recurrent Sets.* Advances in Mathematics, **44**(1982), 78-104.

[7] A.J. Fisher: *A new algorithm for generating Hilbert curves.* Software – Practice and Experience, **16**(1986), 5-12.

[8] W.J. Gilbert: *Fractal Geometry Derived from Complex Bases.* The Mathematical Inteligencer, **4**(1982), 78-86.

[9] J. Giles, Jr.: *Construction of Replicating Superfigures.* Journal of Combinatorial Theory, Series A, **26**(1979), 328-334.

[10] L.M. Goldschlager: *Short algorithms for space-filling curves.* Software – Practice and Experience, **11**(1981), 99.

[11] G. Mills: *Logo (Berkeley) for Windows.* Program help file, 1996.

[12] A. Null: *Space-filling curves, or how to waste time with a plotter.* Software – Practice and Experience, **1**(1971), 403-410.

[13] P. Prusinkiewicz, A. Lindenmayer: *The algorithmic beauty of plants.* Springer, New York, 1990.

[14] R.G. Strongin: *Čislennye metody v mnogoekstremal'nyh zadačah.* Nauka, Moskva, 1978, 186-214.

[15] N. Wirth: *Algorithms + Data Structures = Programs.* Prentice-Hall, 1976.

[16] I.H. Witten and B. Wyvill: *On the generation and use of space-filling curves.* Software – Practice and Experience, **13**(1983), 519-525.