“white” versus “brown” noise in roguelike maps

Jeremy Mates

May 13, 2018

White noise

is simply random[Bulmer(2000)]. In the context of a roguelike given a grid of arbitrary size white noise may be achieved by placing objects randomly on the grid. The file `noise.lisp` details the complete code; only snippets will be shown here.

```
(defparameter *rows*  20)
(defparameter *cols*  72)

(defparameter *board* (make-array (list *rows* *cols*) ...)

(dotimes (n 100))
  (setf (aref *board* (random *rows*) (random *cols*)) #\x)))
```

This produces no apparent pattern (and a human may dismiss it as “oh, random”).

```
...xxx...............................x..x......x..............x.........
.............x....x...............x................................x....
.........................xx...............x..........x.........x........
....x................x.....x.x............x.........x.....x..........x..
............xx...............x..........................................
..x...............................................................x..x..
....x.................................................................x.
............................................x..............xx...........
...............x............x....................................x......
......x...........x...x.......x............x.................x..........
...............x.............................x.....................x....
.x....x.................................xx..x..............x.....x......
......................x....x.........x..........x.x.................x...
.x.x................................xx...xx.x.........x...x.............
.........................x..............................................
x....x.....................................x.....x.............x........
......x......................x..x..........................x............
...............................................x........................
..............xxx......x..............x......x...........x............x.
............x......x.........x.......x.....x...................x......x.
```

Brown noise

by contrast will locate each new object relative to the previous object. Care must be taken that the objects do not walk off of the grid (unless they wrap around) and that there is no bias (unless such is desired) that makes the walk favor a particular direction.

```
(defun if-legal (new max)
  (and (>= new 0) (< new max) new))

(defun nearby (x max range)
  (do ((new nil))
    ((numberp new) new)
    (setf new (if-legal (+ x (- (random range)
                                (truncate (/ range 2)))) max))))

(let ((r (random *rows*)) (c (random *cols*)))
  (dotimes (n 100)
    (setf (aref *board* r c) #\x)
    (setf r (nearby r *rows* 5))
    (setf c (nearby c *cols* 9))))
```

A notable feature here is that the randomization is different for the rows and columns; rows move ±2 and columns ±4. Neither this nor the previous implementation care if an object overwrites another.

```
......................................xx.x....x.......x.................
..................................x..xx..x...xx.x.x.....x...............
.......................................xxx.xx..........x................
.....................................xx.xx.................xx...........
......................................x..x..x..........x..x.............
...........................................x............................
...............................................x......xxx...............
.............................................xxxx.......................
............................................x..x.x.x......x.............
...........................................x.x.xx...x...................
............................................x.......x.........x.........
...........................................x.xx.......x........x........
....................................................xx..................
....................................................x.x............x....
.................................................x................xx....
...................................................x...x..........x...x.
........................................................x.x...........x.
.....................................................x........x..x.x....
........................................................x..xx...........
.........................................................x...xxxx.x.....
```

This is obviously quite different from the white noise; moreover, repeated runs of the white noise version will look more or less the same while the brown noise will vary greatly depending on the starting point and exact sequence of random values.

With multiple object types the differences between white and brown noise can be made most notable; instead of `x` the following place plants, fungus, and rocks.

```
............#.......P.....................f..f......#P..f........P..#..P
.......#...P...........#.........P....P..PP........f......P..#...P..#..f
.....P..P.P..........#f.f.f...f............#....#P...#..f..#............
f.P.P..#..f#...#.....P........#....#....f.P.fP...P....#f......f.f#.f....
...#.P.f...#.P.....P............f......P...............f.f...f..........
#...................Pf.....................#.....#...f.f.P.........f...P
.....#.................P.f#...#..f.............#...#.f.#f...f......f....P
..P....Pf........ff..................f#.........ff..##.f........#........
.P.....#..#...P..........f.....f.......f.P...#ff.#..#.#............f...
..f.P...#f..........fP.....f............#......fP....#..................
...f......P....f.....f.P...#....P............P.P..f..P#.......P..........
....#.f#........#......f..f...fP......f.....P..#.#........#..##......#.P
#ff......#......#................f................#.....................
.....P........@.......#.P......P....P..f......#...........#.............
.#f#..#..........f....P..f....P...P.....P.P.P...P..P....#.......f.....#.
.........P...f......P....##....P#......#....P..#.#......fP..........#...
#f....f.....#.P.........f#...#..f#P...#........#............PP....f....#
.P...................f.#......f.....P..P.f...f.#....###....f........f..f
.PP.#PP.#..............##..#P.#.....................................fP..#.
.#...#...................#..P....f...........f.......#.P...P....P.f.P..f
```

and a brown noise version

```
......f......ff...f.ff........#...#####.................................
......f.....ff.......##............##.#.#.#.............................
...f...f..f.....f....#..#..#...######.#..##.............................
..........f.......#....#...#..##....###.#...............................
f...f....ff.......#......#...###.##.##.##...............................
......f...........#...#..#.#....#.#.#.................f................
.f..................P...#..#.##..#.#####..#.............................
...................P....P.......###....##...#.........f.................
.P.P.f..P.....P.......P..P..##..#.#.....#.................f..............
.P..P.....P..............#..#.##.......#...............f.f.f............
....P..fP..........P.P.PP.P###.............#............f..f............
..P...P.PP.fP....PP..P........#.............................f...........
....PP.f..f...@...f..Pf.....PP...............ff.f..............f.........
...PP..PP...f.....P..P.Pf..................f.f.f..........f..f..f........
P.P...P........f...f.f...f..............f........fff.f..................
.P.PPPP.........fP...f...f..f....ff.....f.f.f.f..f.fff..................
..PPP...P.....................f......f.....f.f.f.ff.f..f.................
PPPPPP...P......P..PP......f..........f.f.f.......f..f..................
PPPPPPP.PP.............................f.f...f..ff...f.................
P.P.PPP.....P..............................fff.........................
```

Other complications are possible under brown noise. The randomization rules could vary depending on the object, or could vary over time–close randomization punctuated by the occasional long leap would be an obvious thing to try (this may approach "pink noise").

Evolved landscapes

may in part replicate the real world; a first pass could use white noise to distribute a random spread of colonizing species, and subsequent passes use brown noise to place secondary growth. One could also include rules for shade, nutrients, etc. though a simple mix of white and brown noise will likely produce acceptable results. The following used 3% plants (white noise), 10% fungus (brown noise), and 20% rocks (brown noise).

```
..P.........................P.....P..f.Pfff.ff..f#f#f..#.fff.ff...ff.P
.............................................ff.ffff#fffff..f#f..##..#fP
...............P..........P......P....ff......f.f.f#..ff.f..#..#.#...#
.............P.............................fP.ffff.#f#ffffff...#.#####..
........................P.........................f#..###f.#f....###..#
.......................P..P................f..f#..#.#f#f#P.#.####.#..
.P................P.................................ff...###f##..#.##..
.......................PP...........................#.#.####.###.......
.................................................P#..##.####.##.###.#.
.............@.....................P..........#........##.#####.##......
..P.....P.........................P......##.......#.#.###.....###..#..
..........P.......................P...P.#......#######.#...#.#.#..#...#
.............P..........P..................##.#########.##.###.#.#.#.
.........................P..........#...#.....###.#.##.#..####.#......
```

Cleanup passes may be required to sculpt the random results; look at cellular automata algorithms for examples. Pathfinding may also need to be done if an impassible map is not desired: hunting trails could be carved out to join areas together. The real world does have various amounts of impassable terrain; having some inaccessible areas may add realism or interest to a game. (Note that while it may warm your black heart to have *The +17 Grand Scythe of Über Slaying* generate in an inaccessible area players tend to be critically lacking in humor over such quirks of the RNG. . .)

An improvement might be to generate random numbers of fungus and rocks from random starting points, and using different rules for how much the fungus and rocks can move. Code that does this can be found in `noise.lisp`.

```
........#.......##...fPf..........f....P.........f.f.####.f.fPf.f...f...
.......#.#.#P..####..f#ff...f.f...................##..##..f...f......fff
.P...#.####......#.###.#fff.P.....PP.........fPff..##.......ff..f..f..ff
......###...........###.#.f.f.f.........ff.f#.....f#.ff...fff..ffff....f
......#............P..###.....................##.f.......f.ff.f......f.f...
.................f...f#####.f.........f...ff..#f......f.ff....f.f.f.f....
...#..........@...ff###.###..f.....f...f..f...f...f...f.ff.Pff..f...#..#
....####..........####..####f..fffff..f.ff.fff...f.........f...f..f.####
..###.###.......P....#...#f#f.f.......f...f..........#..f.ffff....f..###
..##.####.##.............#.###f..f......f.ff.........#.#..f.ff.fff..f####
..##..#.########....ff.##f###f.f.f.....f....f...P###..##...ff.f..ff#f#f.
...#.....#####.#.f..f#####.##f.ff..fff.f.#f.....###...##.ff.fffffff####.
.P..#.#fff.###f......#######...f..f.f...P#.#...........###.f.ff..f#f###f
....##.####f...........f.####..fff....f..####.P......ff####ff.f.f..f#f.#.
```

Vaults

enclose a particular amount of space with a particular structure; the structures or rooms may be pre-designed or could be generated by some algorithm. Either the vault or the noise could be drawn first, depending on whether a definite vault structure or a ruin now overrun by jungle is necessary. Some vaults may designate what points are available to be filled in; these would be used to mask where the noise is allowed to appear.

```
##.####    white ##.####    brown ##.####
##...##    noise ##.f.##    noise ##.f.##
#.....#     fill #...f.#     fill #.....#
#......          #....f.          #.f..f.
#.....#          #.....#          #.....#
##...##          ##f..##          ##.f.##
#######          #######          #######
```

White noise will benefit smaller vaults where there is not enough space available for a pattern to emerge. The above both used a 20% fill (some fungus were drawn under the walls) of which the white noise is much more efficient to generate. If a human cannot tell the difference, use the less expensive function.

Other improvements

would be to where necessary maintain a count of objects placed; this count could be used to ensure that the field has a suitable texture: loop placing different types of objects, check the object count, repeat if the output is still too sparse. This will be necessary if routines with a high degree of variance are used (a 10% chance of ending itself `decay` function usually exits well before 20 iterations, until it does not. . .) or if the object placing functions only place a fraction of the total required.

The code used here does not check if a square is already occupied by something; these checks will be necessary if there are structural elements that must not change, or skipped if ornamental lichen are being placed on an otherwise empty field.

"Pink noise" was alluded to above; this might be an interesting line of study. For more information, see the

References

[Bulmer(2000)] Michael Bulmer. Music from fractal noise. *Proceedings of the Mathematics 2000 Festival*, pages 10–13, 2000.