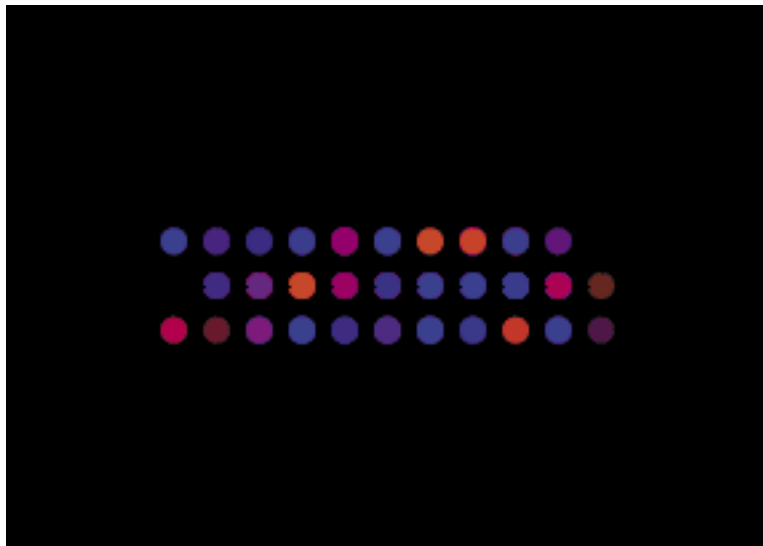


# running in circles

---

Run in circles with GridSquares! This is a great tutorial for learning the Geomancy grid system.



a tutorial for : [gridsquares](#) from geomancy



**shapes and lines for broadcast design.**

[ from Digital Anarchy ]

f/x tools for revolutionaries.



GridSquares is the easiest of the Geomancy filters to learn because it behaves most like a traditional particle system. In fact, the only real difference is its use of a Grid to control how the particles behave.

Even better, GridSquare's 'Randomness' options allow an organic change to the particles without having to set keyframes, unless you want to. [ figure 1 ]

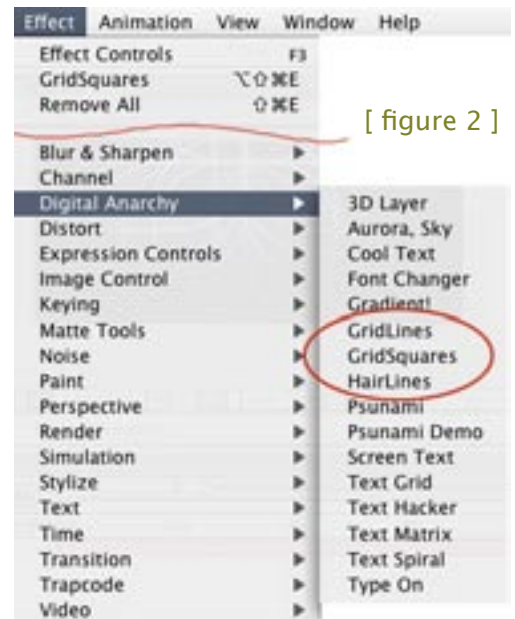


[ figure 1 ]

## 00- download & install

Before you start this tutorial, you will want to download the [geo\\_squares-tute.zip](#) file from our website. This ZIP file contains an After Effects .aep file and QuickTime example movies.

You also need to install our Geomancy plugin set into your After Effects/Plugins folder. The plugins will appear in the 'Effect' dropdown menu, in a 'Digital Anarchy' submenu. [ figure 2 ]



[ figure 2 ]

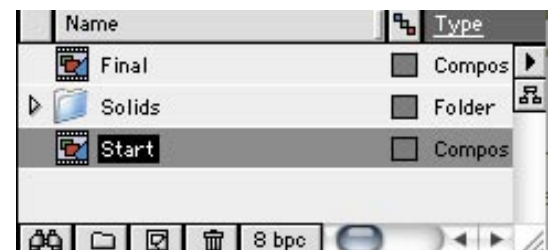
If you are working with the demo version of Geomancy, a red 'X' will watermark your footage.

## 01- project setup

From your download folder, open up the [geo\\_squares.aep](#) project file in After Effects. The 'Final' comp shows your finished piece.

You can also play the QuickTime movie called [geo\\_squares-final.mov](#) to see the final composition that you will create.

The 'Start' comp is simply a 640x480 project with a new Solid layer. Alternately, you can just create a 640x480 comp and add a Solid layer. [ figure 3 ]



[ figure 3 ]



## 02- about the grid

The grid in GridSquares exists to help you wrangle particles into going where you want them to go.

Normal particle systems (like Particle Systems II from Final Effects) are very good at letting you define the attributes and behavior of a particle at birth. But they aren't good at giving you control over that particle after it's been born.

Our grid gives you an extra level of control. It allows you to set up a structure that the particles have to follow. Of course, you can spew out particles in many directions and almost completely ignore the grid if you want to use GridSquares like a regular particle system.



[ figure 2 ]

## 03- the space between

An important aspect of GridSquares, however, is that you do not really work with the grid itself. If you click the 'Show Grid' checkbox in the 'Grid Setup' section, a bunch of red lines will display the rows and columns. [ figure 2 ]

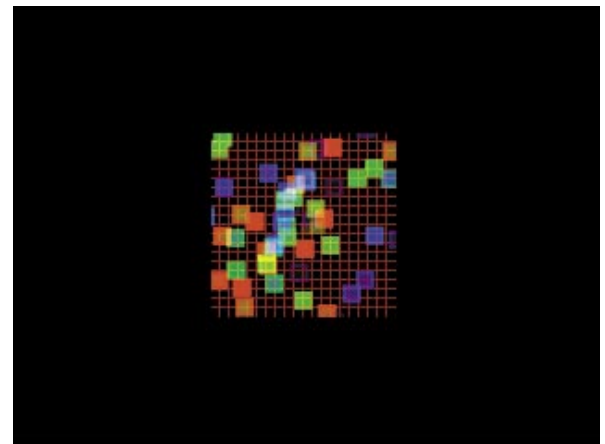
The empty space between the rows and columns (inside the red lines) is what actually matters. By default, GridSquare particles are created in the empty space between the lines. [ figure 3a ]

If you turn on 'Make Size Match Layer' for a moment, you will see a better representation of how the particles fit into the grid spaces. [ figure 3b ]

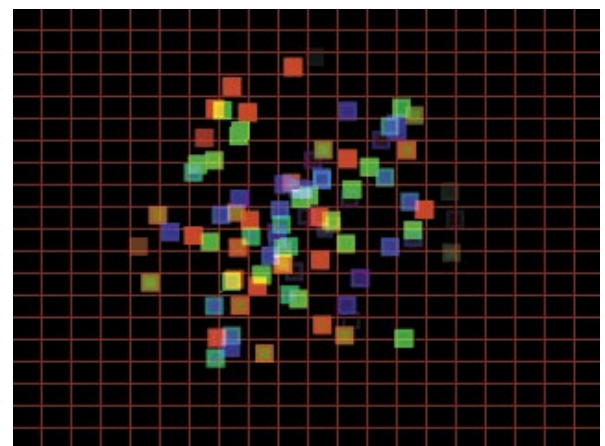
Turn both parameters OFF. You can change this behavior, and we'll talk about how a bit later.

### NOTE

-----  
This behavior of GridSquares differs from the other two Geomancy plugins' use of rows and columns.  
-----



[ figure 3a ]



[ figure 3b ]



## 04- choose the shape

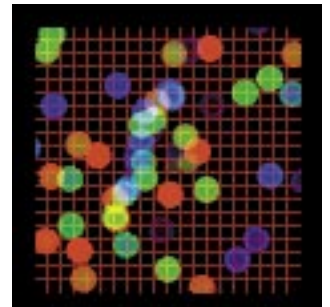
To get rid of the default square shape and change our particles into circles, let's go into the 'Square Setup' section for a moment.



[ figure 4a ]

Twirl down the 'Square Setup'. We don't want to use the default square shape (hey, we didn't pick the filter's name out of a hat, ya know). For this project, we want circles, so set the 'Shape' pop-up to '32-sides'. [ figure 4a ]

This option creates circular shapes, which works really well for small- to medium-sized circles. If you need really large circles, go with the 64-sided polygons. These will appear smoother at larger sizes. For our purposes, the 32-sided polys will render faster.

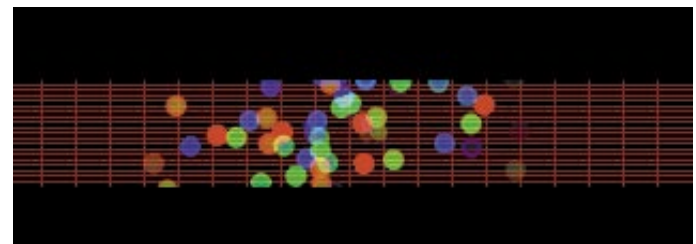


[ figure 4b ]

Since we don't want the outlines on, deselect (uncheck) the 'Outlines' checkbox. NOTE: An important parameter is the 'Grid Unit'. We're not going to cover it here, but make sure you read the GridSquares manual before changing it.

## 05- resize the grid space

Go back into the 'Grid' section and change the 'Rows' to 3. The empty spaces get bigger because there are fewer spaces for particles to be created.



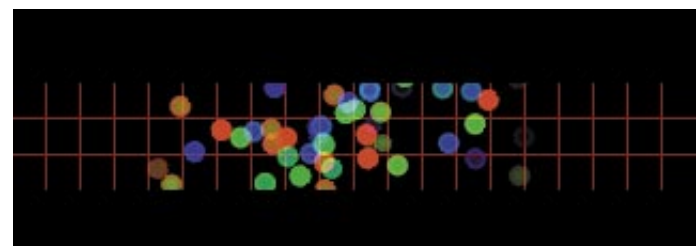
[ figure 5 ]

Making the empty spaces bigger will NOT increase the size of your particles. Particles have their own size control and the grid spacing won't affect them.

## 07- grid dimensions

Now let's set up the foundation.

Set the 'Grid Height' to 100. Set the 'Grid Width' to 640, the width of our comp. [ figure 5 ] Set 'Rows' to 3; and 'Columns' to 20. [ figure 6 ] The option 'Make Size Match Layer' should be turned off (unchecked).



[ figure 6 ]



We want our GridSquare circles to move across the screen in a thin band. There's no need to have the grid bigger than the band, since our particles aren't moving out of formation. So, we've set the grid dimensions to the exact size of the band (3 rows of circles = 3 grid rows). [ figure 7 ]

NOTE

As you will see shortly, we can alternately set the Grid to have more rows, and use its 'Producer Point' to limit where circles get created. There are times when you'll want a grid with extra surrounding space; for instance, if the particles are changing their location.



[ figure 7 ]

### 08- set the producer point

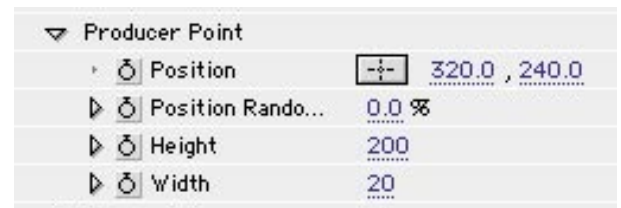
The 'Producer Point' controls where the particle lines generate from.

If the Producer Point is really small, all of the particles will be created in a single grid space (which is the empty space between the rows and columns). They will appear to come from a single point. The larger the 'Producer Point', the more grid spaces it will cover, and the wider an area those particles will appear in.

Imagine a radar screen with its bright line revolving around. We want the Producer Point to be that bright line, leaving a trail of particles behind, but only creating those particles along the line.

Twirl down the arrow for the 'Producer Point' section. Set the 'Height' to 200 and the 'Width' to 20. This gives us a tall, thin producer point. [ figure 8 ]

If our Grid was larger than 200 pixels in height, the particles would still only be created in the 3 rows the Producer Point occupies. This is because the Producer Point is only 100 pixels high, even though the Grid is larger (at 200).



[ figure 8 ]

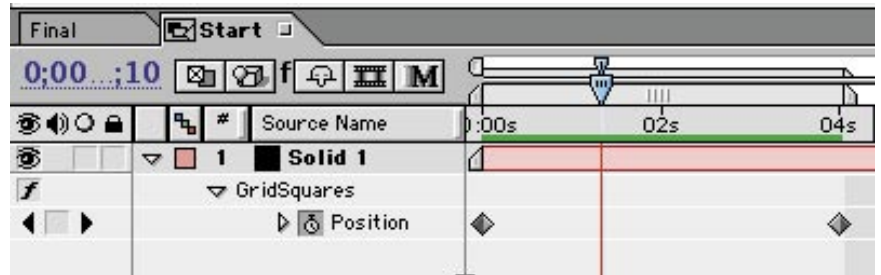


## 09- animate the producer point

Now let's get the particles moving across the screen by animating the Producer Point.

When we animate the Producer Point movement, it will create particles that slowly appear across the screen, just as old ones slowly fade off.

In your Timeline at 00:00, set the 'Producer Point Position' to 10, 320. Move the Time Marker to 04:00, and keyframe values of 630, 320. [ figure 9 ]

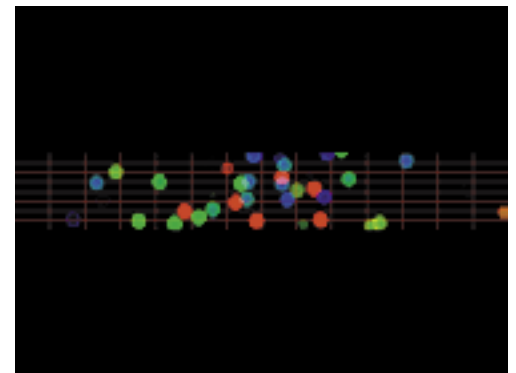


[ figure 9 ]

## 10- preview animation

If you play the animation back now, or open our QuickTime movie called [geo\\_squares-half.mov](#). You'll see that our particles are indeed created as the Producer Point moves across the screen. [ figure 10 ]

Unfortunately, they don't stay put; they're all sorts of different sizes; and they're flying around and flying off the edge of the grid. Now you can see why you sometimes may want the grid larger than the Producer Point!



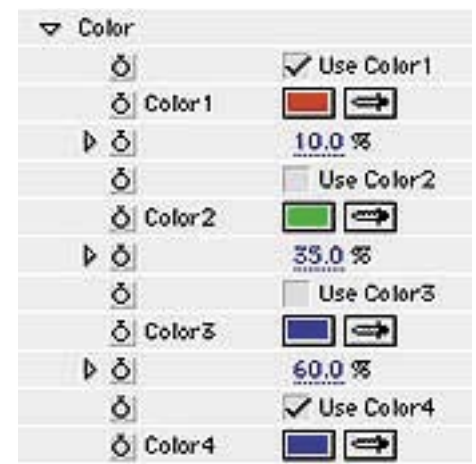
[ figure 10 ]

Well, we don't want particles that are flying around and we certainly don't want them at different sizes. We'll deal with these issues in the final 'Square Setup'.

## 11- color parameters

Let's choose a color palette for the circles. Go to the 'Color' section.

Make 'Color 1' red and 'Color 4' blue. Now turn off Colors 2 & 3 by unchecking their 'Use' boxes. This will create a gradient that goes from red to blue. The circles will get their colors from this gradient. [ figure 11 ]



[ figure 11 ]

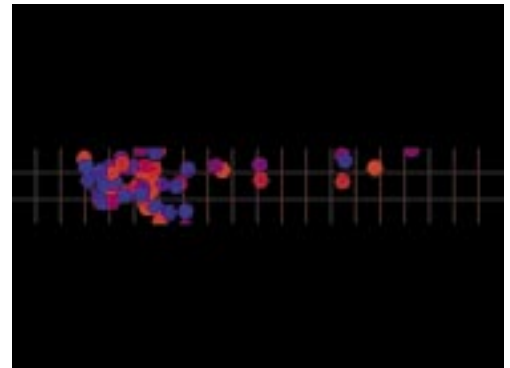


## 12- change blend mode

Still in the 'Color' section, change 'Blend Mode' to 'In Front'. You are essentially setting a transfer mode within the plugin. [ figure 12 ]

This will cause any circle that gets created in the same Grid space to be rendered on top of any circle that's already there. Like creating an stack of opaque objects.

If we left 'Blend Mode' at 'Normal', the circles would blend together, creating other colors or adding them together to create white. It's tough to get excited about a transfer mode, but 'In Front' is pretty cool and useful, so if you're going to get excited... this is it.



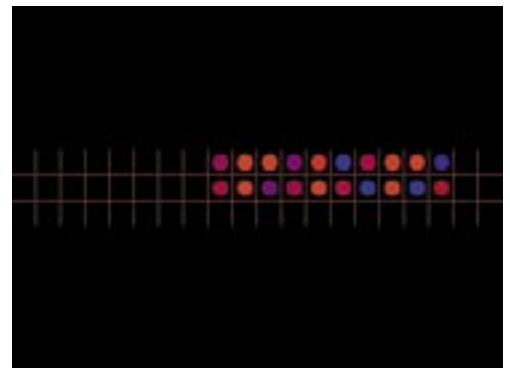
[ figure 12 ]

## 13- set the speed

Open up the 'Square Attributes' section. We need to tell GridSquares how its particles should look and behave.

'Speed' controls how fast the circles move in a given direction once they're created. Since we want the circles to stay still, set 'Speed' to 0.0. [ figure 13 ]

With a speed of 0, they're parked at home and ain't going nowhere. This also confines them to staying within the empty grid spaces.



[ figure 13 ]

## 14- min/max width & height

Next, let's make sure the particles have a nice uniform size. Set the 'Minimum' and 'Maximum Width' to 20 and set the 'Minimum' and 'Maximum Height' to 20 also. All circles, no ovals. [ figure 14 ]

▶	⊗	Speed	0.0
▶	⊗	Speed Randomness	0.0 %
▶	⊗	Horz Growth Speed	3.0
▶	⊗	Horz Growth Speed Ra...	0.0 %
▶	⊗	Vert Growth Speed	3.0
▶	⊗	Vert Growth Speed Ra...	0.0 %
▶	⊗	Spin Speed	0
▶	⊗	Spin Speed Randomness	0.0 %
		<input checked="" type="checkbox"/> Squares can Overlap	
		<input checked="" type="checkbox"/> Squares can Overwrite	

[ figure 14 ]



### 15- min/max growth speed

Further down you'll notice 'Horizontal' and 'Vertical Growth Speed'.

With 'Min/Max Width' and 'Min/Max Height' set to the same size - as they are here - there's no room to grow. The 'Growth Speed' is ignored, so we won't bother with that parameter.

If Min and Max values are different for Height or Width, the 'Growth Speeds' control how fast particles grow from their Min size to the Max size.

These are great parameters to experiment with. For instance, if you set 'Min Width' to 5 and 'Max Width' to 20, the circles would grow from 5 pixels wide to 20 pixels wide.

Of course, with all four values set to 20 in this project, our circles start off and end up at 20. [ figure 15 ]

Square Attribute		
▶	⊗	Minimum Width 20
▶	⊗	Maximum Width 20
▶	⊗	Width Randomness 0.0 %
▶	⊗	Minimum Height 20
▶	⊗	Maximum Height 20
▶	⊗	Height Randomness 0.0 %
▶	⊗	Birth Rate 3.0
▶	⊗	Birth Rate Randomness 0.0 %
▶	⊗	LifeSpan 60
▶	⊗	LifeSpan Randomness 15.0 %

[ figure 15 ]

### 16- birth rate

If you play back the animation now, you'll notice there are grid spaces that don't get filled up. We can solve this problem by cranking the 'Birth Rate' up a little.

Set 'Birth Rate' to 3. This will cause 3 particles to be created every frame. [ figure 16 ]

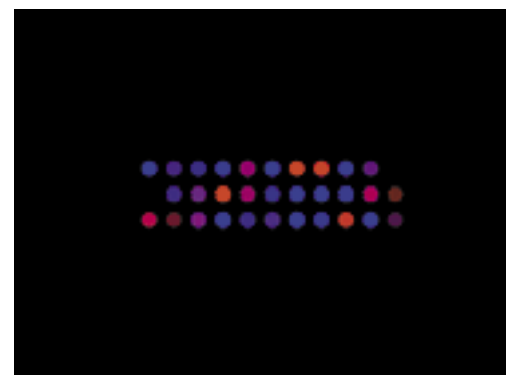
The more that particles are being created, the more likely one will be created in every grid space. Since particles are random, there's no way to guarantee that they will fill every space, unless you've made a lot of 'em.

NOTE:

-----

Remember that each empty space is a POTENTIAL location for a particle to be created. That doesn't mean a particle WILL be created there. Whether a particle is created in any given location is a result of where the 'Producer Point' is. Also keep in mind that once a particle is created, it can move all over the Grid, unless you set things up otherwise.

-----



[ figure 16 ]



### 17- about randomness params

A 'Randomness' parameter helps you vary the values of the parameter its associated with. These options are great for easily adding a little chaos to your animation without having to set keyframes or give up overall control.

Randomness is essentially a built-in Wiggler for every parameter. Convenient, but they increase the number of parameters, which makes the filter look a bit more daunting than it is.

### 18- lifespan randomness

The particles at the end of our GridSquares stream - the ones that were created first - all seem to fade out evenly. This is OK, but we really want the composition to look more 'organic'. We can accomplish this by fading out the particles somewhat randomly.

To get them to do this, set 'LifeSpan Randomness' to 15. Keep 'LifeSpan' at 60. [ figure 17 ]

For the most part, we want our circles to live for 60 frames. The Randomness parameter will vary their LifeSpan just a bit, between roughly 50 and 70, without having to set any keyframes. There you have it... better living through plugins. ;-)

### 19- fade out

Fianlly, set 'Fade Out' to 10 and 'Fade Out Randomness' to 65. [ figure 18 ] This will further vary the tail of the particle movement.

'Fade Out' is the number of frames that the particles fades out over when it dies off. By giving it a Randomness of 65, we create a pretty wide range of possible values that each particle will get. Some will fade out over 3 frames, some over 16, which effectively varies how long each particle is on screen.

LifeSpan Random Value	Range of LifeSpan Value
0	All Particles have a value of 60
10	54 to 66
50	30 to 90
100	0 to 120

[ figure 17 ]

An example of Randomness is the 'LifeSpan' parameter. Set to 60, all particles created will exist on the screen for 60 frames.

But set its associated 'LifeSpan Randomness' parameter to 50%. Now the particles will get a randomly selected lifespan, ranging between 30 and 90 frames (60 +/- (60 x 50%)).

▶  Viscosity	0
▶  Viscosity Randomness	0.0 %
▶  Opacity	100.0 %
▶  Opacity Randomness	0.0 %
▶  Fade In	5
▶  Fade In Randomness	0.0 %
▶  Fade Out	10
▶  Fade Out Randomness	65.0 %

[ figure 18 ]



## conclusion

Check out the final Running Circles sequence below. You can also view the final movie [geo\\_squares-final.mov](#) in your download folder, or render out the composition that you've just built. [ figure 19 ]

The sheer number of parameters in GridSquares – and Geomancy overall – can initially be overwhelming. Keep in mind that many are 'Randomness' options which simply assist other parameters.

[ figure 19 ]

