Particles

A particle system is a commonly used technique using large number of sprites or small moving objects to simulate fuzzy phenomena, like fluid, fire, dust, clouds or smoke. You can render C4D’s native particles systems (Emitter object, Thinking Particles) using C4DtoA. The third-party X-Particles plug-in is also supported.

By default, particle systems are rendered as points. To render spheres or change any Arnold specific parameter you must add an Arnold tag to the emitter object:
Main

Render Points As

Choose to render the particles as disks, spheres or quads.

Beside the built-in modes you can use any shapes from the scene to render particle instances with by selecting the custom render mode and adding the shapes to the Custom shapes field. If there are multiple shades defined their order can be specified via the Shape mode option. The available modes are:

- **sequential**: shapes in the defined order (as they appear in the Custom shapes field).
- **random**: uses the Random seed to define a random shape order.
- **blend**: divides the shapes in equal amounts.
- **first**: uses only the first shape.

You can also setup particle geometry by using child objects under the Emitter which is the native C4D way. If you have an Arnold tag on the Emitter you must change Render points as to child objects.

When using Render points as with child objects, Arnold uses instances which is slower than rendering with built-in spheres or quads.
Built-in spheres (Render points as set to spheres) have no UV coordinates, you have to use a Sphere child object instead.

Object generators do not render correctly as child objects in Emitter when the Radius multiplier is not 1.0.

**Radius Multiplier**

Applies a global multiplier to the radius size of the particles.

**Min. Pixel Width**

If this value is non-zero, particles in points mode with a small on-screen width will be automatically enlarged so that they are at least the specified size in pixels. The enlargement fraction is then used in the particle shader to adjust the opacity so that the visible thickness of the particle remains the same. For a given number of AA samples, this makes it a lot easier to anti-alias very small particles, at the expense of render time (because of the additional transparency/depth complexity). Good values are in the range 0.2 to 0.7. Values closer to 0 are faster to render but need more AA samples. So if your scene already uses very high AA settings, you should use a low value like 0.1.
**Volume Step Size**

This attribute is for use with the *standard_volume* shader.

Particles switch to rendering in volume mode when using this attribute. This controls the rate at which the volume shader gets called. The value indicates the average distance between adjacent volume shader samples along a given ray, in world space units. Generally you will want this setting to be as large as possible before visibly noticeable artifacts occur, and in the case of voxelized volumes no smaller than about half the size of your voxels (in world space). Low Step Sizes will increase render times. For example, when the step size is 0.1 and the volume is 10 units big in world space, there are approximately 100 primary samples and therefore the volume shader is called 100 times.

![Volume Step Size 0.1](image1.jpg)  ![Volume Step Size 1](image2.jpg)

The volume step size has a fixed size, therefore, the larger the container the more steps will be taken and the longer it will take to render. Also beware that if the step size is too small your render times will increase. If you cannot see anything rendered, you may need to reduce the Step Size value.

**SSS** will not work with particles using the built-in render modes (points, spheres, quads). The workaround is to use a custom shape instead (e.g. Sphere).