For user reading this in PDF or TXT formats: This documentation may not be up-to-date. Unless you have trouble accessing the internet or Google, preferably use the online version (https://docs.google.com/document/d/117Dspq-Lk9wd4W6yAQ6ESPdzMkzZS5A2wkki2ca1nrM)
.

Aurora by Moonflower Carnivore is a Shuriken particle effect library for Unity 5.5 onward. It is not compatible with older versions because all particle effects employ new particle functions in 5.5. To use this manual more efficiently for troubleshooting, you can press Ctrl+F to look for the keyword of your question. If nothing matches your question about this asset, you can email us (moonflowercarnivore@gmail.com) for help. Also follow us on Facebook and Tumblr for latest news and knowledges.

!!! IMPORTANT !!!

Due to a security vulnerability, update your Unity editor to the latest patch version.

Table of content

Change log

- V1.00 (10th March 2017):
    - First release
- V1.01 (15th March 2017):
    - Minor fix of texture glitch; Distinguish between aurora “green-purple” and “green-red” variants.
- V1.02 (April 2017):
    - Fix of particle system structure for Unity 5.6
    - Disable mip map of aurora texture to avoid horizontal banding.
- V1.03 (July 2017):
    - Updated all aurora (meteorological) prefabs for consistent scaling result.
    - Updated shader "Particles Additive Intensify" for slightly improving performance.
    - Added "Aurora 000 custom gradient", "Particle petal area" and "Particle petal area mobile".
- V1.04 (July/August 2017):
    - New shader “Particle Additive Intensify Rotate Shear” for rotating and shearing aurora pillars.
- V1.05 (October 2017):
    - New 100 series Aurora prefabs with custom shader for better performance.

Basic usage of particle system

All effect prefabs are contained in the “Prefabs” folder which can be instantly used in your project. Selected few prefabs are provided with mobile variants in the subfolder “mobile effects”. If scaling of the effect does not match your project, you can adjust the scale values in the parent transform component. Doing so will affect all child particle systems because all of them have their scaling mode defined as “hierarchy”. If you do not want the specific child particle system to inherit the scale value from any parent object, change the scaling mode to either “local” or “shape” (parent scale affects size of particle emitter shape only) depending on your need.

To preview a particle effect, you drag the prefab from project folder to the scene hierarchy or scene view, select the particle prefab in hierarchy and click “simulate” button of “particle system” controller in the scene view. When the scene still is not played, you can click “apply” to save your changes to the prefab in the scene, this effectively overwrites the same prefab in project folder. If you want to create a new prefab instead of overwriting the existing one, rename the prefab parent in hierarchy window and drag it back to the project folder.

To simulate the same one-off magical effect in multiple instances in your game, you can use Instantiate on the fly, but this practice undermines performance for frequently used objects even when the object is promptly destroyed. A lighter way is to expect the amount of instances required in the same scene and instantiate the prefab to an array in Start(), remember to SetActive(false) of all particle prefabs initially. When the magical effect needs to be simulated, look for the “next” instance in the array and move this instance to the desired position, finally do a consecutive SetActive(false) and SetActive(true) to trigger particle system simulation. Instead of using ParticleSystem.Play, we recommend our method with SetActive because some of our scripts attached to the particle system object triggers the required actions OnEnable. We could use ParticleSytstem.IsAlive in our scripts, but it needs to be checked in Update() which is another performance pitfall.

Intensify shaders and HDR Bloom

This asset comes with 2 customized “Intensify” Particles shaders (Additive and Diffuse) which are the default shaders with one additional “intensity” parameter. This parameter does 2 things: It multiplies RGBA or albedo values for brighter color; In our public desktop demo, when the camera and its Bloom components have “HDR” options enabled (if you are using “forward” rendering path then you need to either change to “deferred” rendering path or disable “AntiAliasing” in Edit-Project Settings-Quality), the post-processing effect calculates color value greater than 1 (or FF or 255) instead of capping it. When you set the bloom threshold greater than 1, you can achieve “selective bloom”. Keep in mind that HDR bloom is slow on mobile platform. Also Unity seems to treat the excessive RGBA values as negative when HDR is not enabled, resulting hollowness of the texture. In such case, change the intensity value to 1 or just use the original “(Mobile/)Particles/Additive” shader instead.

Our asset cannot includes Bloom scripts from the official image effect package because of approval policy of Asset Store. Please download the free package from Asset Store and follow its instruction carefully to add the desired image effect because its basic setup is quite different from the previous version which was included in the Unity installer. Here is the setting of Bloom script in the desktop demo:

• Quality: High • Mode: Complex • Blend: Add • HDR: On • Intensity: 0.5 • Threshold: 1.25 • RGB Threshold: White • Blur Iterations: 4 • Sample Distance: 10	• Lens Flares: Anamorphic • Local Intensity: 1 • Threshold: 0.1 • Stretch width: 2 • Rotation: 1.55 • Blur Iterations: 4 • Saturation: 1 • Tint Color: CCC166FF

Take extra care that if your camera has MSAA enabled, uncheck it otherwise it would cause over-saturation glitch to the render result.

You can pick HDR color per particle (instead of per material only) with Custom Data module in Unity 5.6 but the setup is very fiddly and, as Unity technician has claimed, performance taxing.

FX camera

Referring our demo scenes, “FX camera” is a separate camera object parented to the “Main camera” object. FX camera only renders objects of the “TransparentFX” layer which is excluded from Main camera’s culling mask. Because the “depth” of FX camera is higher than Main camera, objects rendered by FX camera will always appear before those by Main camera. Its purpose is to avoid undesirable intersection between meshes and z-fighting between alpha objects. Most big glow and spark particle systems of our effects are assigned to TransparentFX layer for achieving cheap glowing effect without Bloom which is especially helpful on mobile platform. The real drawbacks of this practice are that object rendered by higher depth camera will cover absolutely everything by lower depth camera; draw calls batching of objects with same material does not happen when they are rendered by different cameras. These 2 issues are negligible for effect lasts very short.

If you have no intention to set up an FX camera in your scene, you can still use all particle effects of this package as usual unless you have changed the culling mask setting of TransparentFX layer in your Main camera. In such case you should change the layer of all particle systems back to default.

Custom shaders

This asset includes custom shaders in the “Shaders” folder. Some of them like Additive works very similar to the built-in counterpart but ours all provide the parameters “Intensity” and “Saturation”.

Intesnity is simply for boosting the RGB value of the resultant color. Different shaders have slightly different formula to deal with Intensity (_Glow) and should not be used as a way to control the general opacity, instead use _TintColor. When “HDR” of your main camera is disabled, Intensity should be reset to either 1 (aurora and particle atlus) or 0 (flower and vine). Intensity is mainly for post-processing effect HDR Bloom so only pixel exceeds any RGB value of 1 will be bloomed.

Saturation (_Saturation) is for dealing with the issue of gamma and linear color space with exponentiation (pow), just like how color space conversion is handled. Unity’s default new project always uses gamma which is considered “antique” to these days (Unreal Engine 4 on the other hand mandates linear more or less.) What looks okay in gamma will look very greyish in linear, conversely over-saturated. Linear may not be supported by legacy devices, the point is its lighting is more realistic. In many cases, you only retain gamma if you really want your game to be compatible with very old devices and you do not consider lighting or lightmap baking at all.

Since this asset was originally developed under gamma, this gives us a very tough situation to decide if we should re-adjust everything for linear. The quick solution is to have the Saturation parameter in our custom shaders so you can adjust both Intensity and Saturation’s values until you get your desired color tone.

“Displacement Vertex-Fragment Aurora.shader” is designed for the 100-series Aurora effects to move away from sub emitter of particle system which improves performance, not least because they now use much lower particle emission and subsequently produce lower transparency overdraw. Mobile variants primarily use the mesh “aurora_curtain025” with fewest vertices, but you can change it to variants with more vertices for better definition in particle system renderer module.

Particle texture atlas

There are 2 sets of additive particle texture atlas: aurora pillars and general magical textures. Using texture atlas instead of separate texture-materials allows draw call batching which enhances performance and also it is easier to organize textures. To pick the target “tile” of the particle texture atlas, enable “texture sheet animation” module in particle system, modify the correct x and y tile divisions of the atlas, change value type of “frame over time” from curve to constant and input the correct tile index. The first tile (top left) is indexed 0 (zero). If the atlas tiling is 4 by 4, then the last tile (bottom right) is 15 instead of 16.

If you want to randomize between multiple tiles (referring “rainbow arc” in Harmonize and Panmend III magical effects), those textures must be adjacent tiles of same size in the atlas, then change the value type of frame over time to “random between 2 constants” and input the tile indices of the first and last tiles. In this case the last tile index is not the same as the one used for single tile (frame). Instead it should be the intended integer plus 0.9996, otherwise the last tile will never appear.

The general particle texture atlas still has some vacant space for more additive particle textures. You can integrate those from your own project and enable texture sheet animation of other particle effects to improve performance. You subdivide the atlas if it is nearly full but you do not want to increase its dimension. As shown in the general particle texture atlas, larger texture tiles use 4 by 4 division and smaller ones 8 by 8.

If you have quite some alpha blended particle textures in your project, you can group them into one atlas as well. It cannot not be merged with the additive particle atlas because they use different shaders, hence different materials.

Texture format

Nearly all textures in this package are provided in Adobe Photoshop PSD format. Their component layers are retained for your customization. For example, if you do not like the “diffraction” of the glow, spark and ring textures, you can delete or hide those layers in Photoshop and save.

No need to worry about the eventual size of the build because Unity generates flattened copies from these images upon importing and it is these converted copies actually used in the Unity editor and included in the final build, not the pre-conversion source.

Script: ScrollingUVs_Particle

This script is modified from the one available on Unity3D Wiki which no longer works on particle system since Unity 5.3 because of change of scripting API. Also you must now use GetComponent to call for any renderer, not just particle system renderer.

Our version has also added an option to reset UVs to the original before scrolling when the particle system object is activated again (OnEnable), this ensures the UVs always start scrolling from the same position whenever the particle system is simulated anew. This function is particularly designed for the mesh globes in Fortify, Repel and Harmonize magical effects.

When this script is in effect, a new instance of the material is automatically created for the object with this script. This means multiple usages of the originally same material are independent to each other (IOW barring draw call batching) once the scene starts playing. This is the reason the script provides an additional “Starting UV offset” parameter to override the offset values in the material for greater flexibility in application without storing multiple materials in the project folder.

Script: Random Particle Rotation

This script randomly rotates checked axis of the object Transform when the object is activated again (OnEnable). This script is mostly employed to “encircle” particle systems to randomize start rotation of their emitters. “Start rotation” of particle system only affects individual particles instead of emitter shape, that is why we need this script to increase the variety of the effects.

You can remove this script if you really want to for whatever reason. It is just that the particle system will always begin emission from the exact same angle relative to the parent center every time it simulates.

Aurora effects (meteorological phenomenon)

The meteorological effect of “Aurora” itself is provided in 5 variants. The basic one is for demo purpose which allows instant switch between colorized pillar textures without loading a separate material. If you are only using one single color of aurora in your project, you can just use the other variant of aurora.

To customize the aurora pillar color, open the “aurora_sheet.psd” file in graphic editor and choose a desired layer, check “lock transparent pixel”, then apply your own linear gradient on the texture and save. You may crop the edited portion by the guidelines and save as a new texture. Back in Unity, edit the new texture properties to change “alpha source” to “none” because it is for additive shader. If you have cropped the texture, create a new material which use additive or additive intensify shaders, use any non-switch aurora prefab as base and change the materials of slave emitters.

“Aurora 000 custom gradient” allows you to customize aurora pillar color in the material inspector or Custom data module of Unity editor instead of external graphic editor, but in most cases we do not recommend it because of the unnecessary calculation added to rendering. Use it as a quick preview only and you should eventually add an additional aurora pillar texture matching your preview result.

To change the amount of aurora strands, modify the values of “Max Particles” (main module) and/or “rate over time” (emission module) of the “particle master aurora #”. Changing the start colors of the master emitters affects the resultant color. If you want slightly more variant of the aurora color, add more non-white color markers in the start color gradient.

To slant the whole aurora effect, you cannot change any rotation value of particle system itself. Instead you need to make sure the aurora pillar material is using our customized shader which provides the extra “shear” and “rotate” parameters, then fiddle with either value freely.

The aurora effects may generate glitchy banding which can be fixed in different ways. The horizontal banding is an artifact of low resolution texture Mip map which can be solved by disabling Mip maps in the setting of correlated aurora pillar texture in the "Textures" folder.

The issue of vertical banding is slightly more complicated. If you are not adding HDR Bloom to your camera, you can first change the shader of aurora pillar material which is assigned to the "particle slave aurora trail/sparse" children of the Aurora prefab from "Particles/Additive Intensify" to just "(Mobile/)Particles/Additive". If you want the aurora trail even smoother you can increase the emission rate of particle slave aurora trail from 0.4 to 0.8, but you may want to reduce the opacity of the “start color” of the slave emitter as well because the trails will get brighter as more pillar particles stack. Increasing time scale can cause jitter to the effect too.

Take extra care when scaling aurora effect with Transform component, because the slave trail particle system uses “distance emission”. When scaled, the emission value should be divided instead of multiplied. For example, if the parent scale is 10, the distance emission rate of slave should be divided by 10, which results 0.04 from 0.4.

Each aurora effect has 2 “particle master” objects which primarily differ for the sizes of their emitter shapes. “1” is the smaller one to ensure more aurora strands distributed closer to the center of the parent object. You can remove either “particle master” object to suit your preferred distribution of aurora strands. To have this effect works, it requires at least one “ particle master” object and one “particle slave” object.

Sorting layers of all aurora effects in “particle slave” sub emitters which actually emits visible particles are set to negative two, because in most cases this effect is viewed from ground upward behind the cloud particles. If your camera views the aurora from top (universe) to bottom (earth) or you want to use this effect in the UI then you may change the sorting layer to higher value. Sorting layer ordering is a compromise to avoid z-fighting glitch between translucent objects which plagues nearly all AAA game engines even in 2016.

Lighting of aurora does not use the “lights” module of particle system for performance and consistency concern. Instead it is just one directional light with the script “Aurora Light Fade” to control the fading of its intensity. Admittedly the aurora light is not very substantial and can be removed without harming aesthetics too much. Aurora Light Fade script checks the active state of both “particle master aurora” 1 and 2 objects per frame. If either one is active then the light intensity gradually rise to the target value per frame. If both are inactive then the light fades out. If you use this script but have removed “particle master aurora 2”, then you use “Aurora Light Fade Single” script instead but remember to assign “Particle master aurora 1” to the script variable.

For the 100-series Aurora effects, changing color should be done to “Top/Mid/Bot Colors” of the material, so you may duplicate any existing “aurora_disp_~” material, custom your new color and assign it to your duplicated 100-series Aurora effect prefab. “Highlight” and “Intensity” are notably tuned to default values for mobile variants, because we assume in most cases HDR and post-processing effects are disabled for mobile games and making the highlight visable without HDR looks very ugly.

The 100-series take advantage of Custom Data and Custom Vertex Streams. Custom1.x is for scrolling opacity mask; Custom1.y is multiplier of vertex displacement frequency; Custom1.z scrolls the offset of vertical gradient middle position. In Custom Vertex Streams of renderer module, after Custom1.xyz we also add Size to scale the vertex displacement according to current particle size.

Magical effects

All magical effects are themed after mainstream fantasy games. In case the prefab (skill) names are too fancy or obscure to be comprehended, here is the explanations of their recommended functions in game:

- Stimulate: + physical strength/stamina.
- Fortify: + physical defense or erects shield for damage blocking/absorption.
- Enlighten: + magical power/intelligence.
- Determine: + magical defense or resistance against psychological ailments.
- Agitate: + agility/speed.
- Aim: + accuracy/hit-rate.
- Alert: + evasion or negates debuffs.
- Repel: Erects shield for damage reflection.
- Metabolize: Regenerates health gradually.
- Purify: Cures ailments, primarily physiological.
- Harmonize: Negates positive buffs.
- Mend I/II/III: Restore health.
- Panmend I//II/III: Restore health. Area of effect.

Except for Panmend effects, the rest is supposed to be applied on a single character only.

Panmend III is the only magical effect provided with mobile variant. Its desktop version poses challenge for even recent mobile models because of its high amount of mesh particles. The rest should do fine in recent mobile models. You can even extract the child particle objects from the prefabs to mix and match and customize your own new effect.

Optimization of particle effects

To further optimize for older devices, you can reduce particle start lifetimes, emission rates, uncheck “lights” module (you can remove the source light object in the prefab if it is no longer used by any particle system), uncheck “noise” module or remove dispensable child particle system from the effect prefab. If a particle system uses “Sub Emitters” module which has “master” in its object name, when you change its start lifetime, modify the “duration” value of its “birth” sub emitter (with “slave” in its object name) to be equal to the start lifetime of the master particle system.

Before removing “slave” particle system object, be sure to check if it is attached to more than 1 “master”. Slave is usually placed below its master object for identification. If you need to preserve the master without any sub emitter, detach all slave objects from Sub Emitters module and uncheck the module for a clean break up.

Effect for UI Canvas

To make any object including particle effect which is not UI element by default appears before other UI objects under the Canvas, you need to change the Canvas render mode to “Screen Space - Camera” and assign the camera object which renders “UI” layer (it can be the default Main Camera or a separate camera which only renders objects of UI layer exclusively). Finally change the layer of all particle system objects to “UI” layer (click auto apply to all children when changing the parent object layer). Adjust the “order in layer” value in particle system renderer module if needed, higher value makes that particle system appears before those with lower layer value.

If you use NGUI instead which comes with the “widget” component or similar variant, this component must only be added to the child object of the particle effect prefab which actually has a material assigned in the renderer module. Also this must be done to all similar child objects for the visible particles to appear on the UI. Adding the widget component to the prefab parent which usually does not load a material will give you the “missing material” error.

Day-night cycle scripts

The day-night cycle in the demo scenes is not performance friendly when being automated especially on mobile platforms. It is never intended to be the selling point of this asset as it has already been stated in the Asset Store description page. As such we are not responsible to any complaint regarding scripts solely associate with the day-night cycle in the demo scenes, you are still eligible to use them in your own project, though. Cloud has its material tint (black-clay-white) cycled, while stars have their "max particle size" in particle renderer module cycled.

Known issues

- ***Rigidbody messes with particles***: If the particle system with emission rate over distance or inherit velocity attached to an object with Rigidbody component and you translate the parent position of either Transform or Rigidbody, particles may behave weirdly or not be emitted. Read our Tumblr post for details. Simply put, only use Rigidbody physics if it is absolutely needed when in other cases it can be replaced by Character Controller.
- ***Particle trails in buff effect are broken when I drag the object around***: The path of dragging an object is by default not interpolated which results in jittering trail. To allow smoother particle trails when the parent object moves, you need to interpolate (repositioning afterward) or extrapolate (repositioning beforehand by predicting object position in the next frame) its path. Rigidbody provides both options but are only functional when the movement is controlled via velocity instead of position.
- Scaling is prone to a Unity bug (874051) which is only fixed in Unity 2017.1. First make sure you have already updated this asset to 1.03 at least. Then only change transform-scale value of the prefab parent. Finally change the "rate over distance" (emission module) of all "particle slave" sub emitters. For example, if you **divide** the parent scale by 10 to 0.1, you **multiply** the rate over distance by 10 from 0.4 to 4 of particle slave aurora trail and 0.2 to 2 of particle slave aurora trail sparse.

Issues of Unity 5.5 but only fixed in 5.6 without backporting:

- ***Noise offset upon simulation***: Offset of particle noise module resets every time the effect is being simulated again, leading to same particle movement pattern modified by the noise module. However, the offset does randomize every time the scene starts. This is part of the reason "Random Particle Rotation" script is written to make this glitch less noticeable. Precisely this is not a "glitch" per se, it just makes the effect with noise module less exciting after repeated observation.
- ***Randomized seed of sub emitter***: Emission seed of sub emitter is not randomized every time the effect is being simulated again, leading to same particle starting behaviour.

Issues of Unity 5.6 onward:

- ***Assertion error of particle system structure***: Fixed in v1.02.

Terms of use

Disclaimer: This section only summarizes a portion of Unity's Asset Store Terms of Service and End User License Agreement (EULA). It does not override the agreement detailed in Unity's website if there is contradiction between the two.

This asset published by Moonflower Carnivore (the licensor) is intended to be only distributed publicly in your (the end user's) electronic game and interactive media, in the form of an executable which cannot be modified in Unity editor again.

You are free to use and modify this asset if you obtain this asset legally via Unity Asset Store firsthand and distribute your executable containing this asset legally.

You are not allowed to publicly reproduce, distribute, sublicense, rent, lease or lend this asset, either untouched or retouched, or any resource included in this asset for commercial or non-commercial purposes.

If you obtain this asset secondhand, not via Unity Asset Store firsthand, you are not authorized to use this asset, unless you obtain this asset legally via Unity Asset Store again.

Per EULA, sharing this asset to other computers is only allowed:
(...) *provided that these computers are either all (i) physically located at a single physical location ("Site") belonging to END-USER, or (ii) laptops belonging to END-USER which have been made available by END-USER to its employees that are employed at the same Site provided all such computers have appropriately licensed Unity software installed.*

If you really want to include any single content of our assets in your own asset, whether for profit or not, you must email us to negotiate a separate contract for permission.