FC Feather

author: Fabricio Chamon
e-mail: fabricio.chamon@gmail.com

**if you find it useful, please consider donating*
PayPal fabricio.chamon@gmail.com . thanks!

Donate

1 - Overview

FC Feather is an ICE based feather system Softimage plugin that allows for rapid feather grooming and simulation on geometry characters. The main toolbar contains scripted buttons that automates all the steps necessary to create a basic setup.

2 - Installation

Drag the .xsiaddon file to softimage viewport. Then hit the “update” button on the icetree window to refresh the list of nodes.

3 - Basic Usage

Fire up the main toolbar by clickng ***View > Toolbars > FC_Feathers.***

You now have 5 simple steps, all that is needed to create a basic setup.

tip: Since the system relies on relative cache paths, it is a good idea to save your scene prior to creating the feathers.

3.1 - Creating Tangents

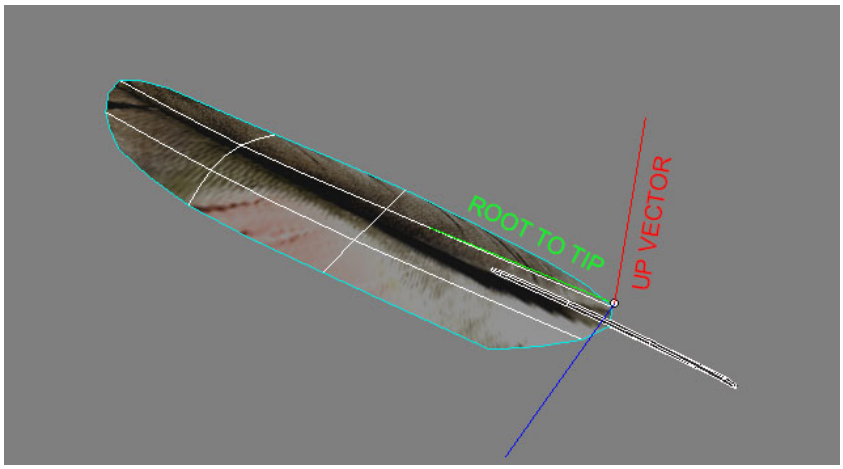
With your mesh in place, it is time to create the basic flow of your particles. Place some curves around the character paying attention to curve direction (reversing curves will revert feathers alignment).

Once the flow is defined, select all curves and click “Create Tangents”. They will be grouped and green colored. *Warning*: Do not rename or move any group/object created by the system before the character is ready to animate.

3.2 - Creating Master Feathers

Master feathers are the objects that will be scattered as feathers on your character. You can have as many master feathers as needed. It can be a mesh or a model with unrestricted hierarchy. It only has to respect a simple orientation rule:

Y axis = feather root to tip
X axis = feather up vector



tip: it is recommended to freeze the master feather scale, so that it relates 1 to 1 to the “particle size” parameter on the pointcloud.

If you have a Model as master particle (like when using complex feather geometry or hair), the orientation of your model null should also match the example above. In this case it is also recommended to build a lowpoly representation of your entire feather and name it “guideMesh”, then hide and place under your model. This is so that the guide feathers have a light topology representation of this specific model.

Once oriented, select all master feathers (middle-click for models!) then click “Set Master Feather”. They will be grouped and yellow colored.

3.3 Creating the Instances Topo Holder

Just click the “Create instances topo holder” button, then select the master feathers group created on the previous step (Grp_Master_Feathers). It will create a null named “Instances_Topo_Holder” in your scene.

tech-explanation: This is a non-artistic, very simple technical procedure necessary to group all the master feathers topology representation into a unified container, that will later be queried by guide feathers to display it’s shape.

3.4 Creating Guide Feathers

Guide feathers let you control specific areas on your character. One can change size, scale, orientation, shape ID and bend of scattered particles. The final size and orientation of all particles is averaged for a smoother result.

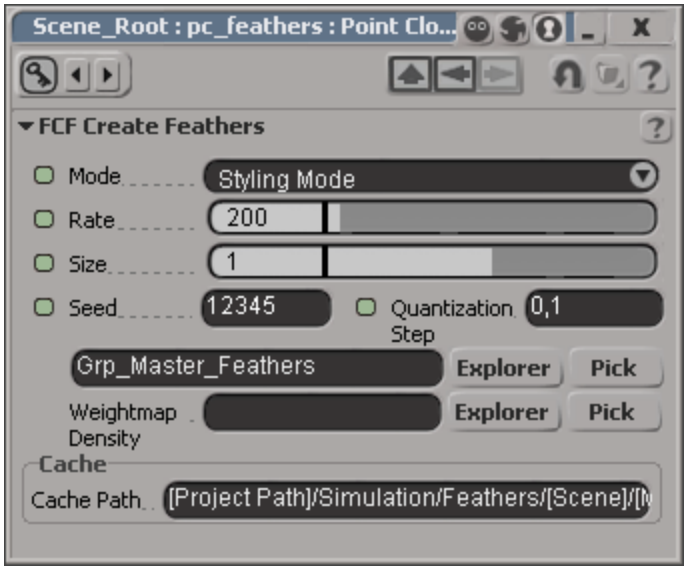
To create your first guide feather select a polygon on your emitter mesh and click “Create Guide Ferather”. You can create more guides at any time by repeating this procedure or by duplicating other guide feathers.

By default they will not render, and are displayed in wireframe mode if the “Override object properties” switch if disabled under viewport display modes.

Each guide also has a parameter set named “Guide Controls”, where you can change the shapeID (master feather to be used) and bend of particles.

3.5 Populating Feathers

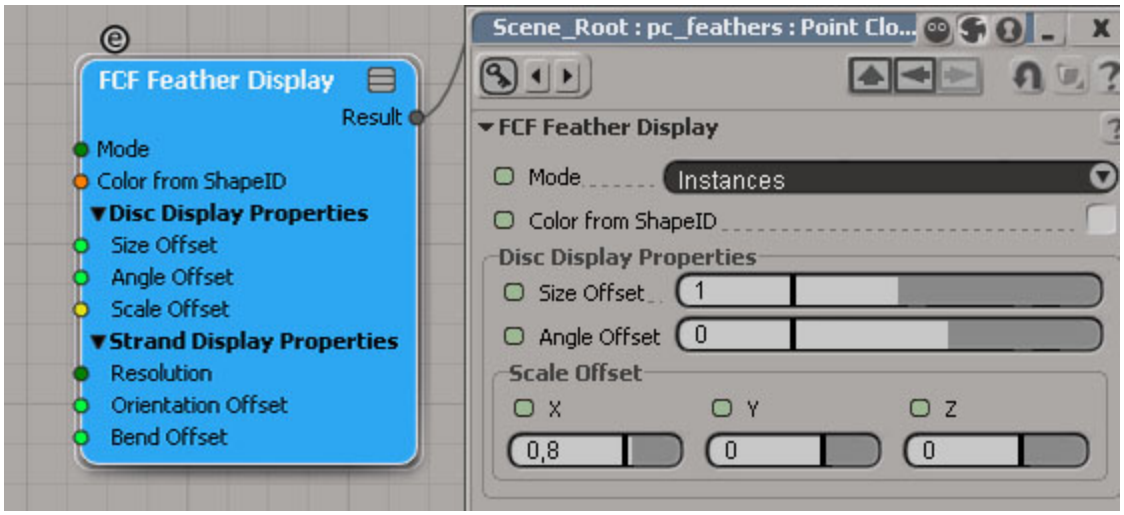
Finally, click the “Populate Feathers” button, and pick the desired emitter geo. The main FC Feather compound will popup.



tip: at this time you can put everything inside a final Character Model and organize your scene.

Mode and ***Rate*** are the two most important parameters. Use *Style Mode* and a low *Rate* to groom your character. After all the guide work is done, crank up the resolution and change it to *Animate Mode*. You are now ready to animate/deform your character!

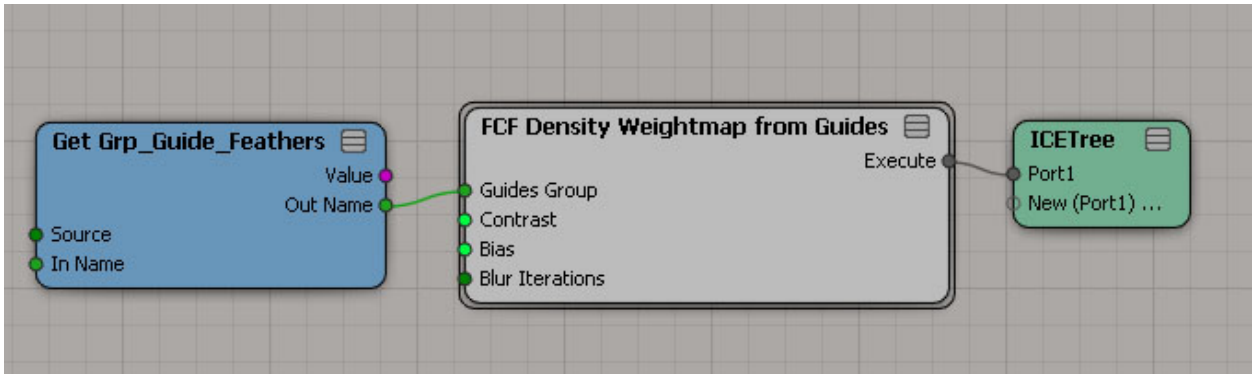
If you suffer form slow viewport updates, connect the *FCF Feather Display* compound in the *pc_feather* pointcloud.



Change the display mode to “Disc Primitive” or “Strand”, then adjust the *Size Offset* so it matches your instances original size. Now the viewport should be lightining fast.

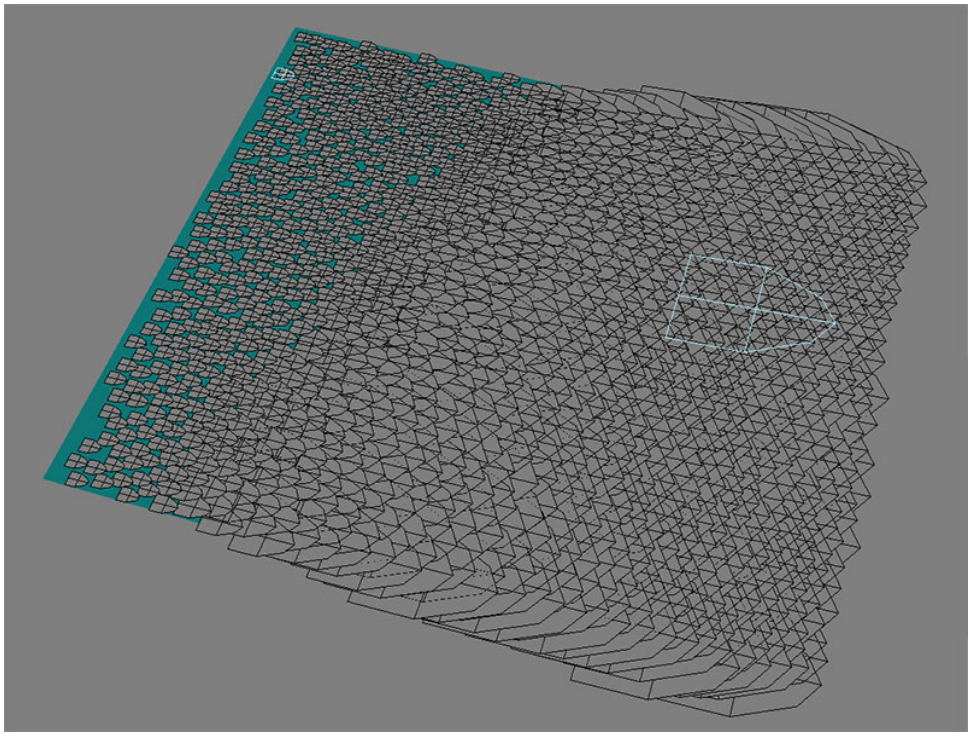
4 - Feather Distribution

FC Feathers distributes particles evenly across the surface to reach optimal coverage at minimum particle rate. But frequently you will need to vary the size of feathers (around the eyes or legs of your character), thus leading to uncovered spots (where you have small feathers). To compensate this problem, you can use the *FCF Density Weightmap from Guides* compound:

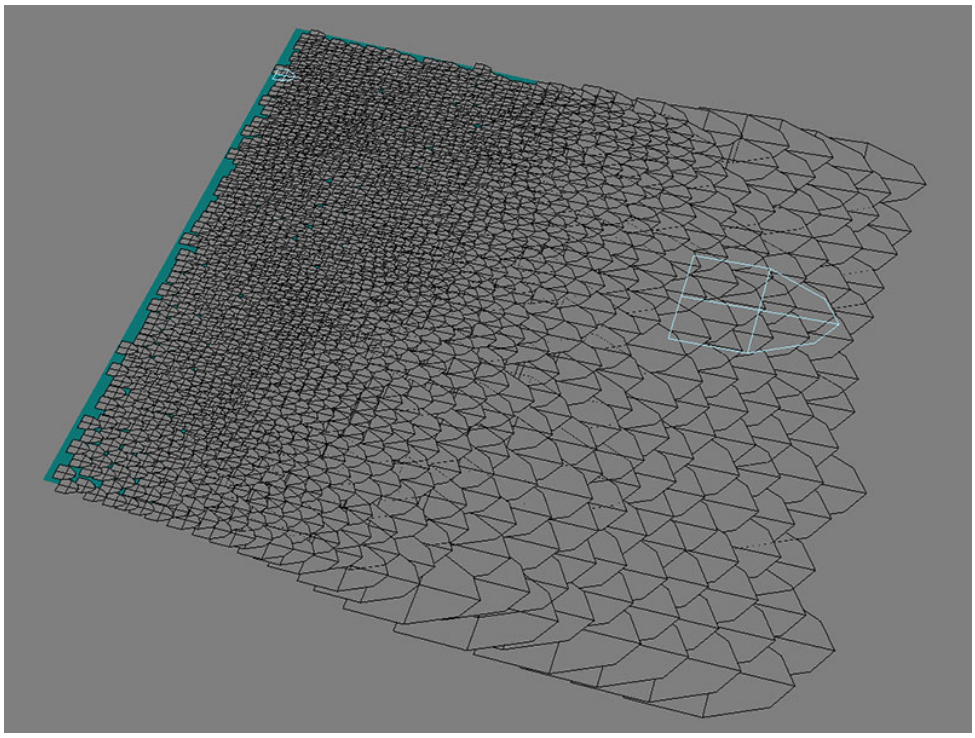


Create a weightmap on your emitter geometry and name it "*Weight_Map_Feather_Density*", then plug the *FCF Density Weightmap from Guides* compound. It will look for the guides size and build a weightmap that you can then connect in the main compound to control feather density.

Example Results:



Default setup. Note the blue spots in between small feathers.



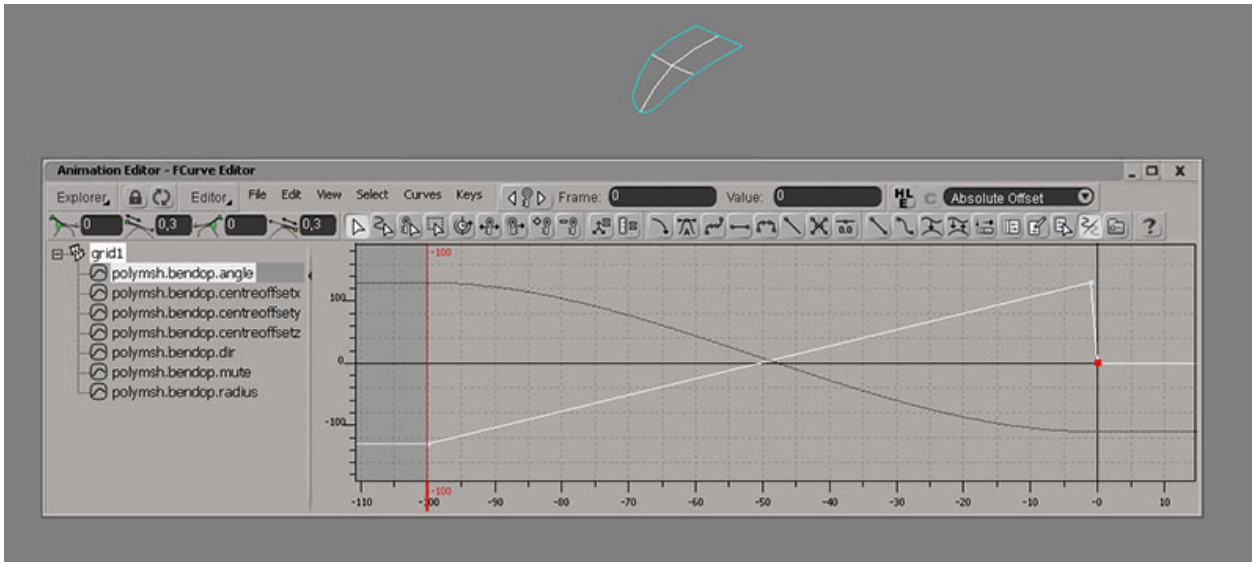
Density weightmap connected. Now the feathers are much more concentrated on small areas. There are no blue spots.

5 - Feather Bending and advanced extra controls

Since ICE can't change instanced particle geometry by itself, a workaround for bending is needed. The solution is to pre-animate a bend operator on the master feather. The icetree then reads guide feathers "bend" parameter and selects the appropriate pre-animated frame.

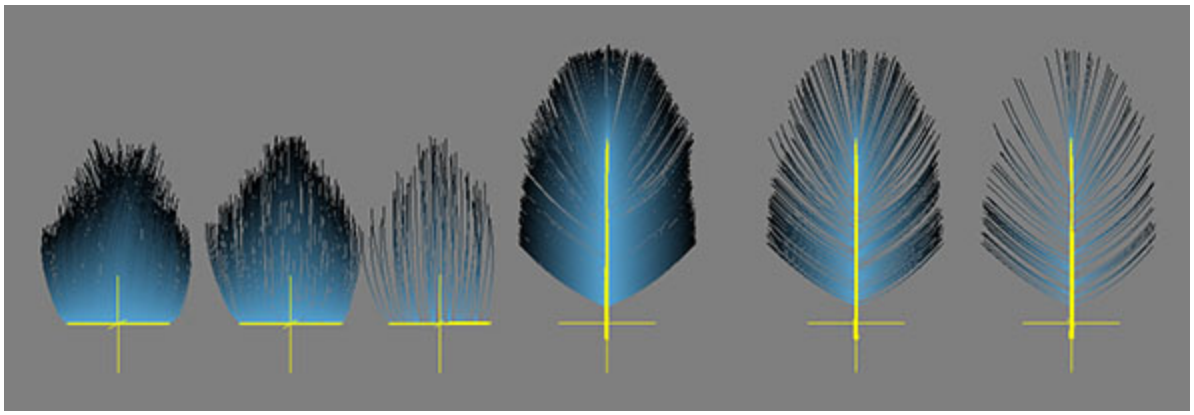
Instructions:

- create a bend operator on the master feather
- bend it down at frame -100, keyframe
- bend it up at frame -1, keyframe
- zero out the bending at frame 0, keyframe



That's it. Now the bending parameter present in the guide controls ppg should work.

You can use the same technique to change other feather properties too, like number of feather barbs on a hair master feather for example:



If you decide using this approach, be aware of the *Quantization Step* parameter present on the main *FCF Create Feathers* compound. For those non-techy, it means reducing (a lot!) memory usage at rendertime. The greater the value, less memory will be used in trade of animation resolution.

6 - Feather Caching

The particles always have to be cached prior to deformation. By default the cache path is set (in the main compound) to:

[Project Path]/Simulation/Feathers/[Scene]/[Model]/[Object]/FC_Feathers

If your character/asset needs to be exported, you'll have to use a constant path, or it will loose the feathers once imported into another scene. Something like this should avoid any path problems:

[Project Path]/Simulation/Feathers/Bird_01

7 - Post Animation Effects (Non-Simulated)

After all animation is done, some effects can enhance feathers look. They are found in the *Feather Animation Controls* icetree category, and don't need to be placed under simulation stack.

8 - Feather Simulation

FC Feather uses a spring based simulation workflow, and needs a second pointcloud for it to work. As this kind of simulation is rarely needed, there's no button to automate the proccess, but should be easy to setup:

- create an empty pointcloud
- create an icetree on the modeling stack
- clone all the points of the feather pointcloud into this pointcloud
- set its particle shape to point or sphere (optional)
- create a new icetree on the simulation stack
- plug *add forces* node

- plug *FCF Spring Force* (randomize the spring factor param and connect the original feathers pointcloud)
- plug *FCF Groom Force*
- plug *drag force*
- plug Simulate Particles node
- now back on the pc_feathers point cloud, create a new icetree on the simulation stack
- plug *FCF Simulate Feathers*
- connect the spring pointcloud
- done.

Enjoy!

f you have any further questions feel free o send an e-mail to fabricio.chamon@gmail.com

Donate