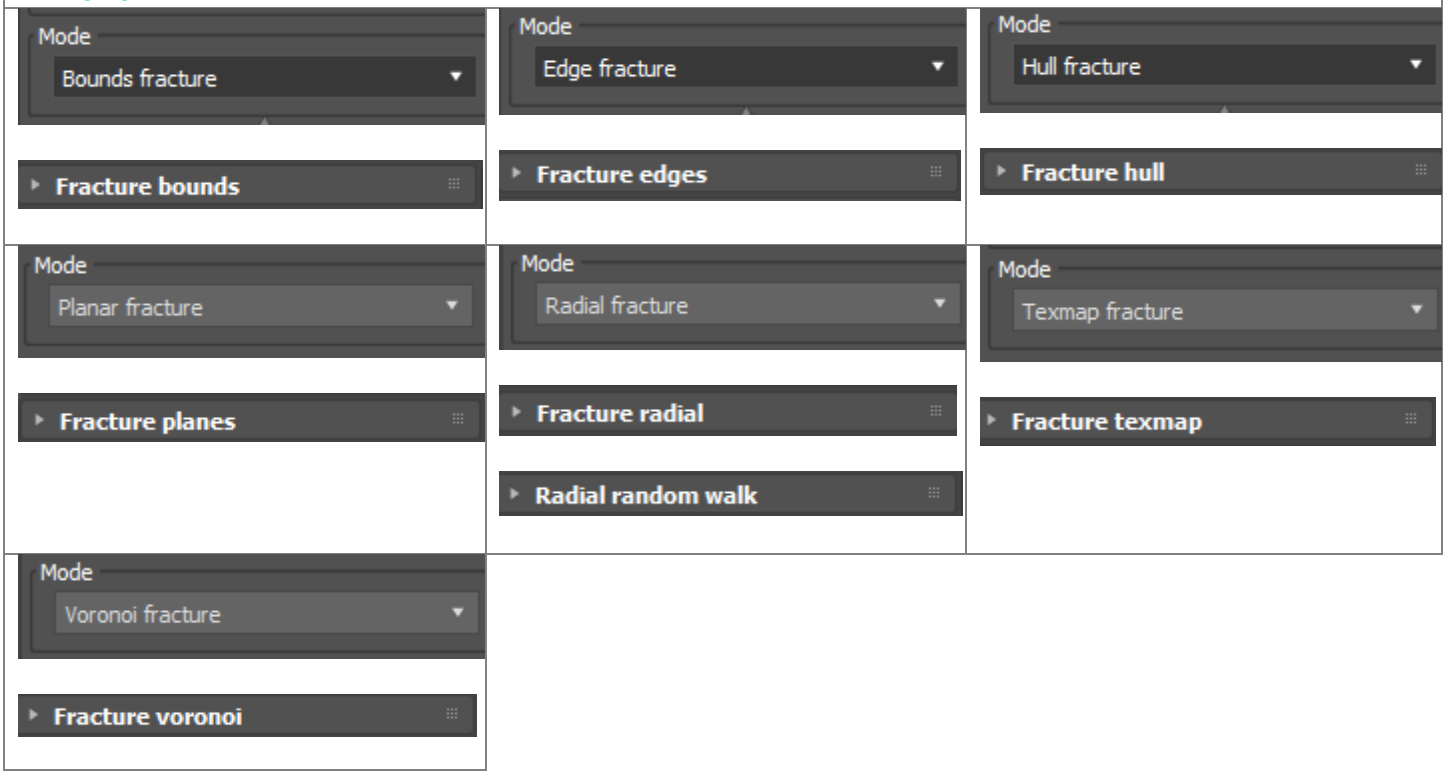
tyFlow MultiFracture Operator Rollout Visibility

QUICK JUMPS - CLICK ON THE IMAGES BELOW TO JUMP TO TOPICS

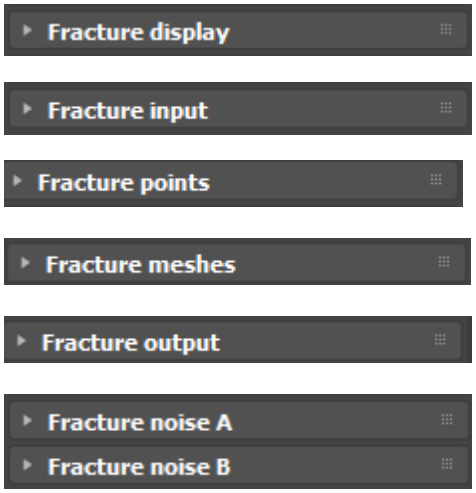
GENERAL MULTIFRACTURE ROLLOUT INFO BELOW ON PAGE 1

MAY 3 2024 – VER 1



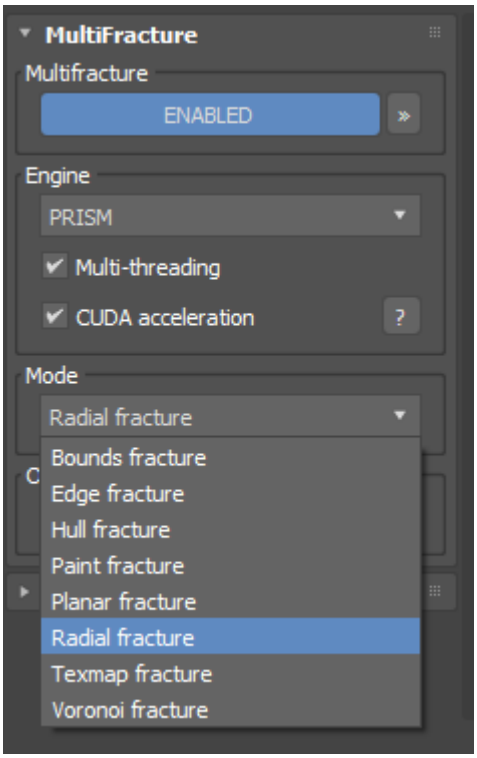
COMMON ROLLOUTS TO ALL MODES

CLICK ON THE IMAGES BELOW TO JUMP TO TOPICS



Multifracture Rollout

BACK



Bounds fracture: fractures will be computed along the object-oriented bounds of a mesh, depending on the ratios between the longest mesh axis, and subsequently smaller axes.

TIP
Use this mode to subdivide overly long meshes into smaller meshes.

Edge fracture: fractures will be computed along the corners/edges of the convex hull of a mesh, depending on the relative angles of those corners/edges

TIP
Use this mode to simulate erosion/damage over the edges of a mesh

Hull fracture: fractures will be computed across random faces on the convex hull of a mesh.

TIP
Use this mode to simulate general erosion/damage over the entire surface of a mesh.

Paint fracture: fractures will be computed along hand-drawn strokes painted over a mesh.

TIP
Use this mode to draw fracture lines directly onto a mesh.

Planar fracture: fractures will be computed along planes scattered across a mesh.

TIP
Use this mode for general, randomized mesh fracturing.

Radial fracture: fractures will be computed using a procedural radial-fracture pattern.

TIP
Use this mode to simulate fractures in brittle material, like glass.

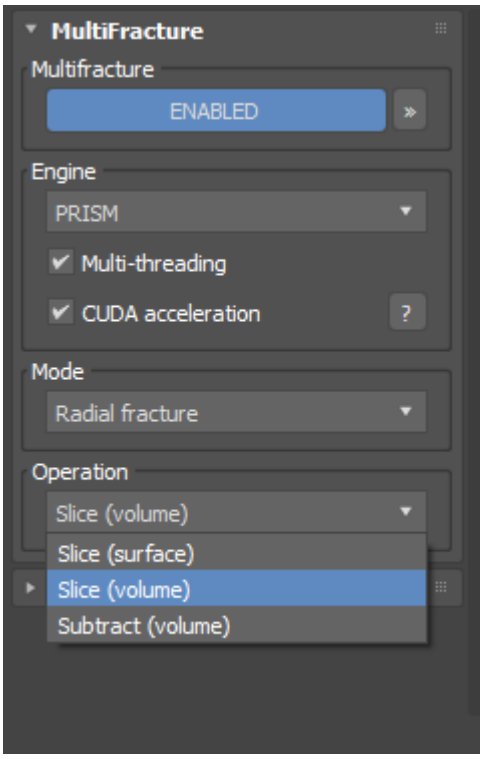
Texmap fracture: fractures will be computed using a specified texture map

TIP
Use this mode to control fracture patterns precisely, using the grayscale values of an image or texture.

Voronoi fracture: fractures will be computed using points in space to generate enclosed cells within a volume.

TIP
Use this mode to gain more fine-tuned control over the size and distribution of fracture chunks.

NOTE
The Voronoi fracture mode requires at least 2 or more fracture points in order to generate proper Voronoi cells. Read the “Fracture points” rollout documentation for more info.



Multi Fracture Rollout Continued:

Operation

Slice (surface): when enabled, meshes will be sliced as if they are thin shells - the resulting fracture chunks will not have thickness or depth.

Slice (volume): when enabled, meshes will be sliced as if they have volume - the resulting fracture chunks will have thickness and depth.

INFO
In order for the volume slice mode to function correctly, input meshes should be composed only of *closed* surfaces. The surfaces may self-intersect, but should not have any open edges. If open edges are present, topological artifacts may appear in the resulting fracture chunks.

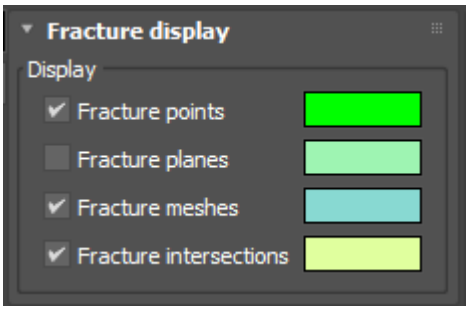
Subtract (volume): when enabled, a boolean subtract operation will be used to remove the fracture mesh from the input mesh, rather than a slice operation.

NOTE
When the subtract operation is selected, fracture meshes will be extruded using shell settings available in the Fracture meshes and Fracture shell noise rollouts. The extruded fracture meshes will then be straight-forwardly subtracted from the input mesh. Typically, this fracture mode produces less optimal/desirable results than the slice modes, however there may be circumstances where it is required.

Fracture display Rollout

BACK

The Fracture display rollout offers controls for how intermediate fracture data is displayed in the viewport. This data can be useful to visualize aspects of the procedural multifracture workflow.



Fracture points: when enabled, individual points used to generate fracture meshes (ex: points used to generate Voronoi cells) will be displayed in the viewport.

Fracture planes: when enabled, planes used to place and orient fracture meshes (ex: gizmo-aligned planes used to place planar fracture meshes) will be displayed in the viewport.

Fracture patterns: when enabled, radial/texmap patters will be displayed in the viewport, aligned to the fracture planes used to place them.

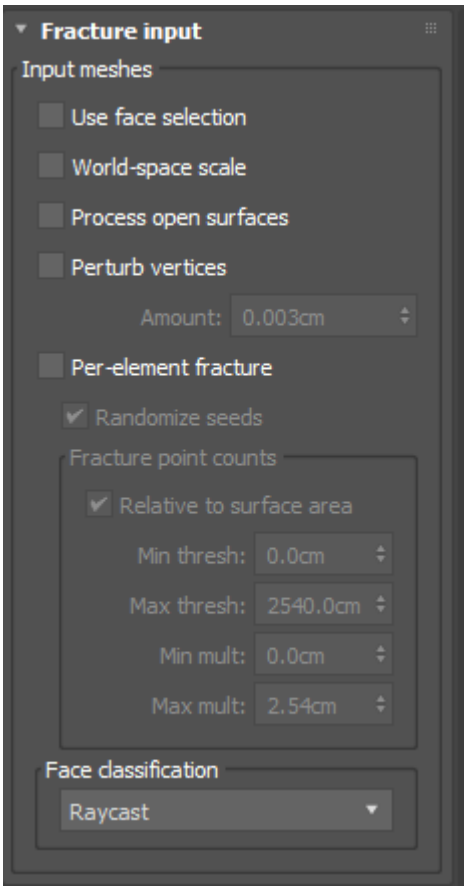
Fracture meshes: when enabled, generated fracture meshes will be displayed in the viewport.

Fracture intersections: when enabled, surface intersections between input meshes and fracture meshes will be displayed in the viewport

Fracture input Rollout

BACK

The Fracture input rollout has parameters which allow you to tune how input meshes are pre-processed, prior to fracturing. Certain parameters are only visible when using the **Multifracture** operator, over the **tyMultifracture** modifier, and vice-versa.



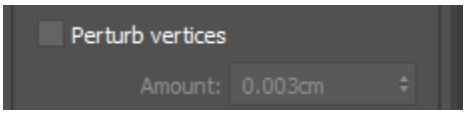
Use face selection: when enabled, only the selected faces/elements of a mesh will be fractured.

INFO
Due to the way the multifracture algorithm processes meshes and their faces, if you enable “use face selection”, the selected faces will be detached from the mesh for the fracture operation and converted into a separate mesh element prior to being fractured. If your face selection consists of full mesh elements only, the detachment process will have no detrimental effect on the resulting mesh. However, if your face selection consists of only partial mesh elements, the detachment process will disconnect the selected faces from the rest of the mesh and those faces will remain disconnected after the fracture operation is complete.

World-space scale: when enabled, various coordinate-relative parameters (ex: noise strength) will be adjusted to match the input mesh’s scale.

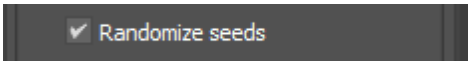
Process open surfaces: when enabled in combination with a subtract operation (instead of a slice operation), open surfaces (which normally cause problems for the subtraction operation) will be treated as thin shells, alleviating those problems in the process.

NOTE
Open surfaces treated as thin shells will not maintain their volume after the subtract operation is completed.

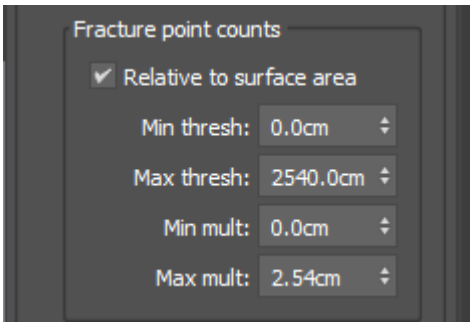


Perturb vertices: when enabled, the vertices of operand meshes will be perturbed by a random amount prior to the fracture operation

Per-element fracture: when enabled, input meshes elements will be split apart and fractured individually, rather than as a single mesh.



Randomize seeds: when enabled, various fracture seeds will be randomized on a per-element basis



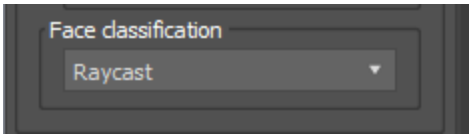
Fracture Input Rollout Continued:

Relative to surface area: when enabled, various fracture point counts will be multiplied by the ratio between an individual element's surface area relative to the specified threshold values.

Min/max threshold: these values determine the threshold range of the ratio multipliers. For example, if you set the "max threshold" value to a surface area of 100, and an element has a surface area of 200, its fracture point count will be multiplied by 2 (prior to being clamped by the "min/max mult" values.

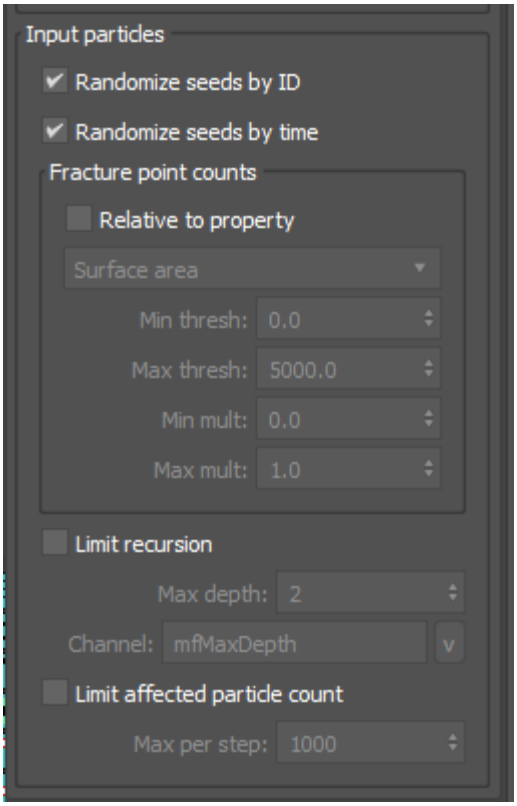
Min/max mult: these values clamp the threshold multipliers. For example, if you set the "max threshold" value to a surface area of 100, and an element has a surface area of 1000, its base fracture point count multiplier will be 10 - but if you set the "max mult" value to 5, the multiplier will be clamped to 5.

TIP
Adjusting the "relative surface area" parameters can give you fairly fine-tuned control over how many fractures an element will undergo, relative to its size. This prevents scenarios where fracture settings which apply to very large elements will also apply equally to very tiny elements, and vice versa. For example, you may want to fracture a large element 1000 times, but fracture tiny elements only once or twice. By tuning the "relative surface area" parameters, you can exert this level of fracture control over elements in a single modifier/operator.



Raycast: when selected, a raycast-based face classification method will run during the operation, to classify faces as either inside or outside of operands. For closed meshes, this is very accurate. For open meshes, this method can produce artifacts.

Fast winding number: when selected, a fast winding number face classification method will run during the operation, to classify faces as either inside or outside of operands. This offers more accuracy than the raycast method when meshes contain holes and open edges (at the cost of some performance). This method does not track nested elements, and is not suitable for situations where meshes are meant to have interior, nested cavities.



THIS IS AVAILABLE IN THE OPERATOR ONLY (Not the Modifier)

Randomize seeds by ID: when enabled, fracture seeds will be randomized on a per-particle basis.

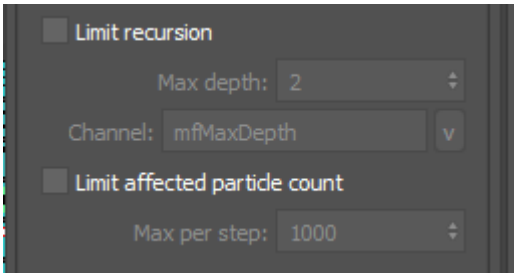
Randomize seeds by time: when enabled, fracture seeds will be randomized depending on the frame the fracture takes place.
Fracture point counts

Relative to property: when enabled, various fracture point counts will be multiplied by the ratio between an individual particle's property relative to the specified property and threshold values.

Min/max threshold: these values determine the threshold range of the ratio multipliers. For example, if you set the "max threshold" value to a surface area of 100, and a particle has a surface area of 200, its fracture point count will be multiplied by 2 (prior to being clamped by the "min/max mult" values.

Min/max mult: these values clamp the threshold multipliers. For example, if you set the "max threshold" value to a surface area of 100, and a particle has a surface area of 1000, its base fracture point count multiplier will be 10 - but if you set the "max mult" value to 5, the multiplier will be clamped to 5.

TIP
Adjusting the "relative property" parameters can give you fairly fine-tuned control over how many fractures a particle will undergo, relative to its size. This prevents scenarios where fracture settings which apply to very large particles will also apply equally to very tiny particles, and vice versa. For example, you may want to fracture a large particle 1000 times, but fracture tiny particles only once or twice. By tuning the "relative property" parameters, you can exert this level of fracture control over particles in a single operator with limited filters.



Limited recursion: when enabled, you can limited how many times a particle can be successive re-fractured by the operator.

Max depth: controls how many times a particle can be refractured by the operator, before it will simply be ignored by the operator.

Channel: the custom float channel where the fracture depth for a given particle (and its children) will be stored.

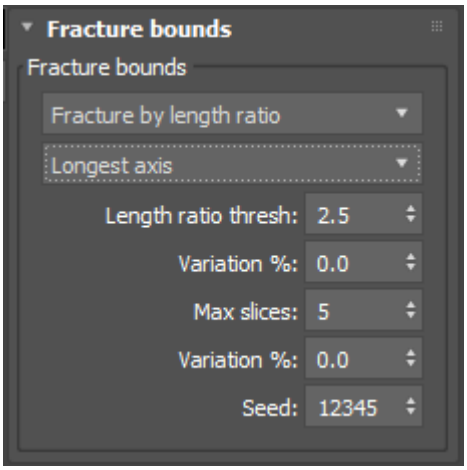
TIP
By enabling "limited recursion", you can have a **Multifracture** operator's timing set to something other than 'on entry', without re-fracting the same particles over and over, leading to an exponential explosion of particles. For example, you may want to ensure a particle is only refractured once (so it will be fractured, and then it's children will be fractured, but its children's children won't be fractured) by setting the "max depth" to 2.

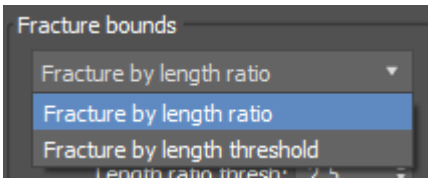
Limit affected particle count: when enabled, the total number of particles fractured by the operator per time step will be limited by the specified value. If the initial number of particles to be fractured is greater than the specified max value, the particles will be sorted by size (largest to smallest) and the sorted list will be clamped to the max value.

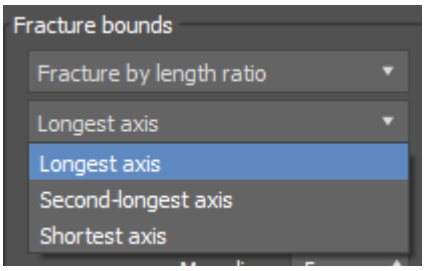
FRACTURE BOUNDS ROLLOUT

BACK

The Fracture bounds rollout contains parameters for the “fracture bounds” fracture mode.







Mode: controls whether the bounds fracture operation will compare the length of the specified fracture axis to the length of the next-shortest fracture axis (and compare that ratio to the specified length ratio thresh), or compare the length of the specified fracture axis to the specified length threshold directly.

Axis: controls which OOBB axis will be used for the bounds fracture computation.

Length thresh: the initial number of slices will be computed by dividing the length of the specified object-oriented bounding box axis by this value.

Length ratio thresh: the initial number of slices will be computed by dividing the ratio between the length of the specified object-oriented bounding box axis and the length of the next-shortest object-oriented bounding box axis by this value.

Variation %: the per-particle percentage of variation to apply.

Max slices: the maximum number of slices that will be assigned to a mesh along the primary object-oriented bounding box axis.

Variation %: the per-particle percentage of variation to apply.

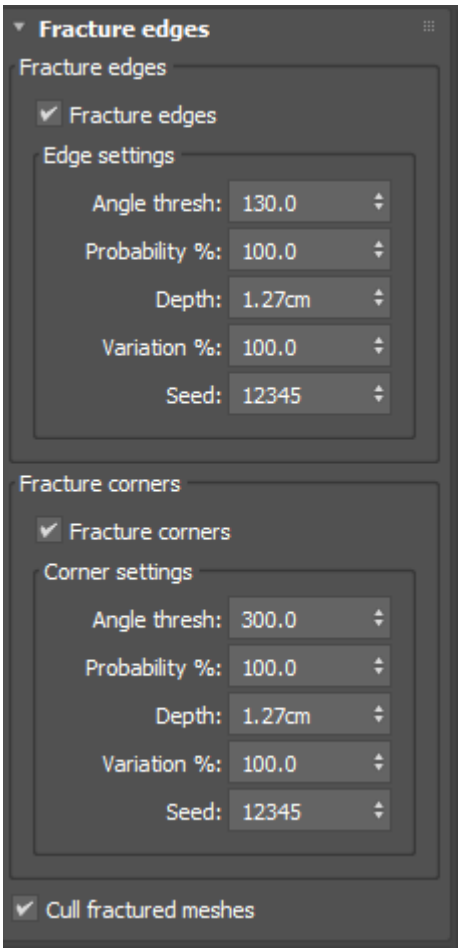
Uniqueness

Seed: the seed value for all varied parameters.

Fracture edges Rollout

BACK

The Fracture edges rollout contains parameters for the “fracture edges” fracture mode.



Edge settings

Edge fractures are generated along edges of the input mesh’s convex hull, that meet the angle threshold criteria.

Angle thresh: the minimum angle that must exist between the faces of an edge on the mesh’s convex hull in order for a fracture mesh to be assigned to the edge.

Probability %: the random probability that a given edge (which meets the angle threshold) will be assigned a fracture mesh.

Depth: fracture meshes will be placed at this depth along the negative surface normal of each qualified edge. Larger values will move fracture meshes inward towards the center of the mesh hull.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.

- -

Fracture corners: when enabled, mesh corners will be fractured.

Corner settings

Angle thresh: the minimum angle that must exist between the faces of a vertex on the mesh’s convex hull in order for a fracture mesh to be assigned to the edge.

Probability %: the random probability that a given vertex (which meets the angle threshold) will be assigned a fracture mesh.

Depth: fracture meshes will be placed at this depth along the negative surface normal of each qualified vertex. Larger values will move fracture meshes inward towards the center of the mesh hull.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.

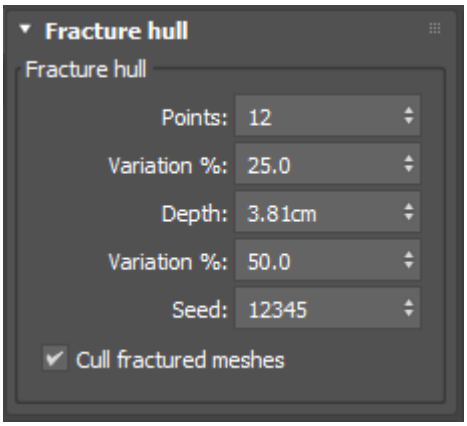
- -

Cull fractured meshes: when enabled, fractured parts of the input mesh that lie above the fracture meshes will be culled. Disable this setting to retain all fracture meshes after the fracture operation is completed.

Fracture hull Rollout

BACK

The Fracture hull rollout contains parameters for the “fracture hull” fracture mode



Points: the number of random fracture points to generate on the convex hull of an input mesh. A fracture mesh will then be placed at every generated fracture point.

Variation %: the per-particle percentage of variation to apply.

Depth: fracture meshes will be placed at this depth along negative surface normal of each random fracture point. Larger values will move fracture meshes inward towards the center of the mesh hull.

Variation %: the per-particle percentage of variation to apply.

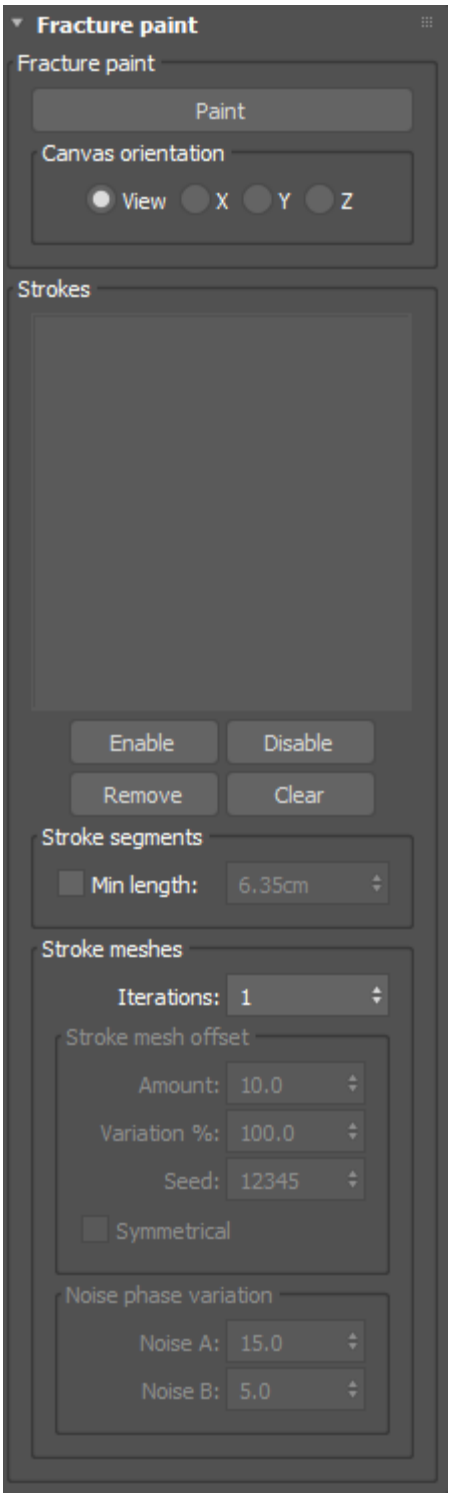
Seed: the seed value for all varied parameters.

Cull fractured meshes: when enabled, fractured parts of the input mesh that lie above the fracture meshes will be culled. Disable this setting to retain all fracture meshes after the fracture operation is completed.

Fracture paint Rollout

BACK

The Fracture paint rollout contains parameters for the “fracture paint” fracture mode.



Canvas orientation: controls whether a view-aligned plane will be used for painting, or an axis-aligned plane.

Stroke segments

Min length: when enabled, stroke segments will be reduced so that the minimum length of any given segment is no less than the specified value.

Stroke meshes

Iterations: when set to a value greater than 1, the stroke meshes will be duplicated N times (where N is the number of iterations greater than 1), and each iteration of stroke meshes will be assigned a different set of per-vertex noise values, offset by the specified phase variation values.

Stroke mesh offset

Amount: controls how much to offset stroke meshes generated *after* the first iteration (ie, iteration 2 and above, unless symmetrical mode enabled - see below), along their vertex normals.

Symmetrical: when enabled, all stroke iterations will be divided into two groups, and each group will be offset in the opposite direction of the other. So, if your iteration count is 2 and your offset amount if 1, one stroke mesh will be offset by 0.5 units along its surface normals, and the other by -0.5 units along its surface normals.

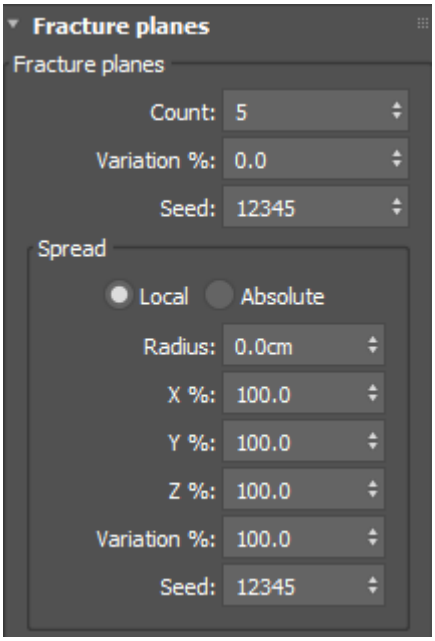
Noise phase variation: phase variation applied to any extra iterations of stroke meshes that are generated, based on the iterations parameter.

Fracture planes Rollout

BACK

The Fracture planes rollout contains parameters for the “fracture planes” fracture mode, as well as any other modes which generate intermediate fracture planes for their fracture meshes (ex: edge fracture mode).

Fracture planes are initially scattered at the center of any fracture points generated by the specified fracture mode. Then, based on the current fracture mesh settings, they are subdivided prior to any noise that is applied and finally are used to fracture the input mesh.



Count: the number of fracture planes to generate at a given fracture point.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.

Spread

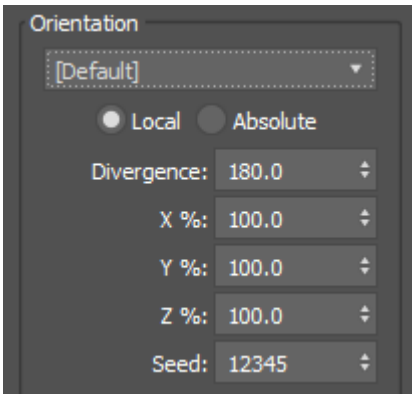
Local/absolute: controls the reference coordinate system that the fracture planes will be spread within.

Radius: controls the maximum spread radius that the fracture planes will be spread within.

X/Y/Z %: controls the percentage of per-axis effect the spread radius will have on the fracture planes.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.



Orientation

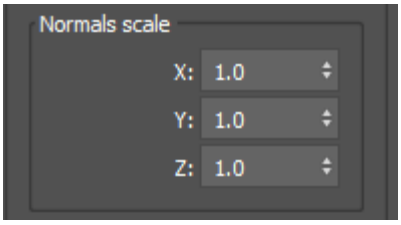
Orientation mode: controls which axis of the input mesh the fracture planes will be aligned to. “[Default]” mode will use the up vector of the fracture points as the alignment vector, which may not be oriented to a particular axis, depending on the fracture mode (for example, fracture planes of the hull fracture mode will be aligned to the nearest convex hull surface normal.

Local/absolute: controls the reference coordinate system that the fracture planes will be aligned within.

Divergence: controls the amount of orientation divergence that will be applied to the fracture planes.

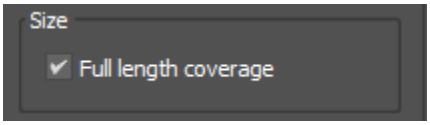
X/Y/Z %: controls the percentage of per-axis effect the divergence will have on the fracture planes.

Seed: the seed value for all varied parameters.

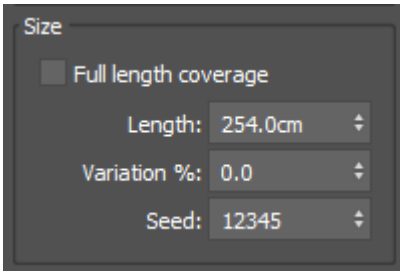


Normals scale

X/Y/Z: these are multipliers applied to the orientation vector for the fracture planes. You can use them to bias the orientation of the fracture planes along a particular axis.



Size

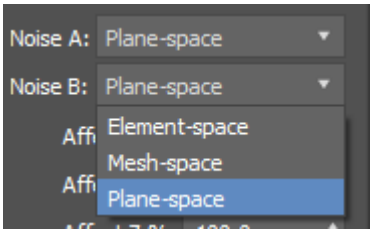


Full coverage: when enabled, the fracture planes will expand to cover the plane-oriented bounds of the underlying input mesh's convex hull. This ensures that all fracture planes will fully cross the input mesh surface boundary, leading to more consistent fracturing. When disabled, depending on the size parameters, planes may not fully cross the mesh surface boundary and therefore may not generate a fracture at those locations.

Length/depth: the size of the fracture planes when full coverage is disabled.

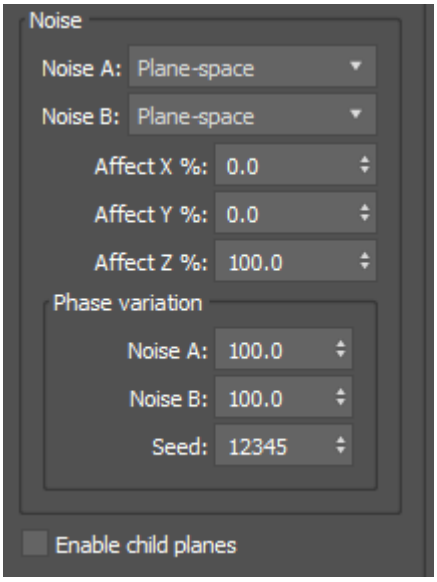
Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.



Noise

[Noise A/B] Element-space: when selected, noise distortion applied to planar cutting meshes will be applied in the coordinate space of the input mesh element (direction of noise displacement from initial plane mesh surface will be independent of plane mesh orientation).



[Noise A/B] Mesh-space: when selected, noise distortion applied to planar cutting meshes will be applied in the coordinate space of the input mesh (direction of noise displacement from initial plane mesh surface will be independent of plane mesh orientation).

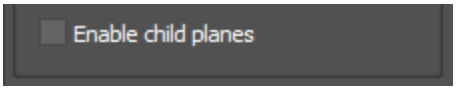
[Noise A/B] Plane-space: when selected, noise distortion applied to planar cutting meshes will be applied in the coordinate space of the planar mesh (direction of noise displacement from initial plane mesh surface will be dependent on plane mesh orientation).

Affect X/Y/Z %: controls how much influence noise values will have on displacement of fracture mesh vertices, relative to the corresponding local fracture plane axis.

Phase variation

Noise A/B: controls the amount of per-plane noise phase variation to apply to fracture meshes generated from fracture planes. Increasing these values ensures that fracture planes do not share identical noise distributions.

Seed: the seed value for all varied parameters.



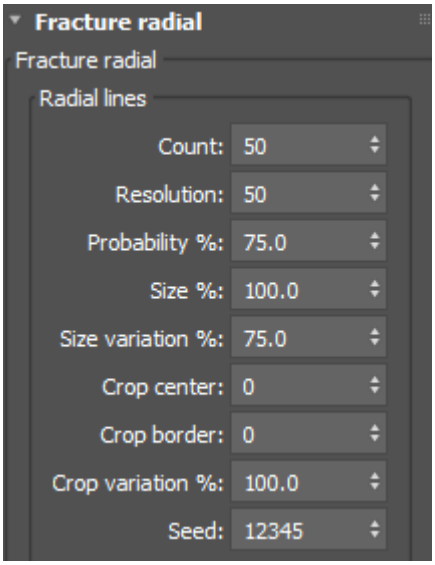
Enable child planes: when enabled, each fracture plane will generate the specified number of child fracture planes. Their corresponding spread and noise parameters have the same effect as those affecting their parent.

Fracture radial Rollout

BACK

The Fracture radial rollout contains parameters for the “fracture radial” fracture mode. In this mode, a radial fracture pattern is generated and applied to fracture planes scattered over fracture points on the input mesh.

Radial fracture patterns consist of two elements: radial lines and concentric lines. Radial lines extend outwards from the center of the fracture pattern, and concentric lines trace around the center of the fracture pattern, connecting the radial lines.



Count: the number of radial lines to generate outward from the center of the fracture pattern.

Resolution: the number of sub-segments each radial line will be composed of.

Probability %: the probability that a given radial line will not be culled. Decreasing this value will increase the chances that a given radial line will be removed from the fracture pattern, thereby increasing its randomness.

Size %: the overall size of radial lines, relative to the size of the underlying fracture plane.

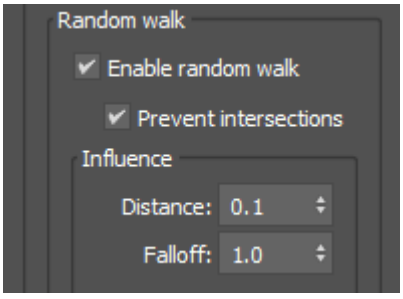
Variation %: the per-particle percentage of variation to apply.

Crop center: crops radial lines, from the center outwards, by removing sub-segments. Increasing this value decreases the density of radial lines closer to the center of the fracture pattern.

Crop border: crops radial lines, from the outward edges inward, by removing sub-segments. Increasing this value decreases the length of radial lines.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.



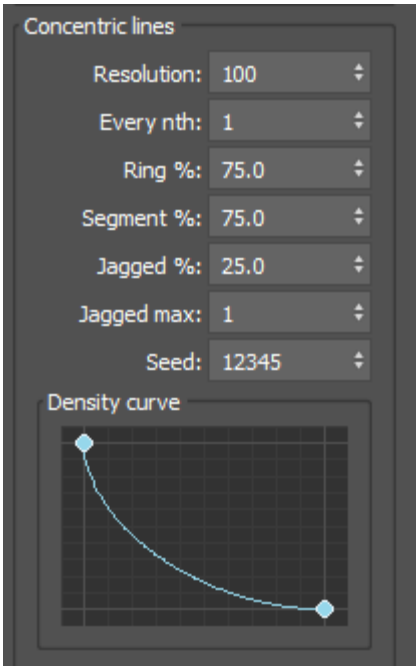
Random walk

Enable random walk: when enabled, radial lines will be affected by random walk noise as they expand outwards, causing them to expand outward in more random patterns.

Prevent intersections: when enabled, radial lines will be prevented from intersecting each other, during their random walk outwards.

Influence

Distance/falloff: controls the amount of influence the random walk noise will have on radial lines, from the center of the fracture pattern outwards.



Resolution: the number of sub-segments a concentric line will be composed of. Note: the actual number of sub-segments a given concentric line will be composed of is relative to its coverage of the fracture pattern, in degrees, multiplied by this value. If resolution is set to 100 and a given concentric line only has 180 degrees of pattern coverage, it will be composed of only 50 sub-segments.

Every nth: controls how many concentric lines will be generated, relative to the resolution of the radial lines. For example, if radial lines is set to 100 and "every nth" is set to 5, then 20 concentric lines will be generated (1 for every 5 radial line sub-segments).

Ring %: controls the probability that a given ring of concentric lines will not be culled. Increasing this value decreases the density of concentric lines.

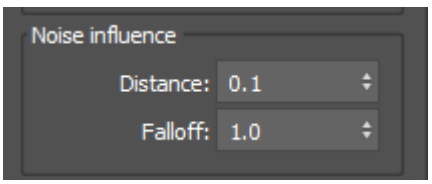
Segment %: controls the probability that a concentric line segment (a concentric line spanning two radial lines) will not be culled. Increasing this value breaks up concentric lines which would otherwise travel all the way around the fracture pattern.

Seed: the seed value for all varied parameters.

Jagged %: controls the probability that a given concentric line segment will undergo "jagged" transformation (that it will connect to mis-aligned sub-segments along its corresponding start and end radial lines, based on the "jagged max" parameter).

Jagged max: controls the maximum amount of mis-alignment that a concentric line segment may undergo.

INFO
Normally, concentric lines will connect at the same sub-segment index located on its start and end radial lines. For example, if two radial lines have a resolution of 100, the concentric line that connects them at their center will start at sub-segment 50 on the first radial line, and connect to sub-segment 50 on the second radial line. However, by increasing the jagged parameters, the sub-segment indices that the concentric line connects at will be more random. If the "jagged max" parameter is set to 5, the start sub-segment index might be 45, and the end sub-segment index might be 55. This would tilt the concentric line relative to its start/end radial lines, giving the interior fracture a more jagged appearance.



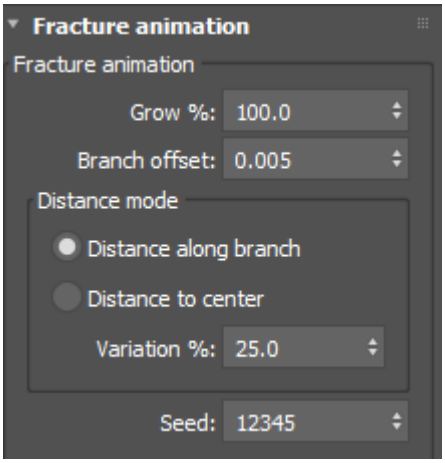
Density curve: this curve allows you to control the density of concentric lines within the fracture pattern, relative to their distance from the center of the fracture pattern.

Noise influence: allows you to control how much influence fracture noise A/B will have on the pattern, relative to the center of the pattern

Fracture animation Rollout

BACK

The Fracture animation rollout contains parameters which allow you to control how fracture patterns (patterns generated in radial/texmap fracture modes) are animated.



Grow %: controls the growth of a fracture pattern, outwards from the center of a given fracture plane.

Branch offset: controls how much to offset the growth of a given pattern branch, randomly.

INFO
Pattern branches are continuous groups of pattern sub-segments. For example, in a radial fracture pattern, each radial line is considered a branch. When “branch offset” is set to a value greater than 0, the growth of each branch will be offset/randomized by the specified amount.

Distance mode

Distance along branch: branches will grow relative to the distance of each sub-segment along the branch.

Distance to center: branches will grow relative to their distance from the center of the fracture pattern.

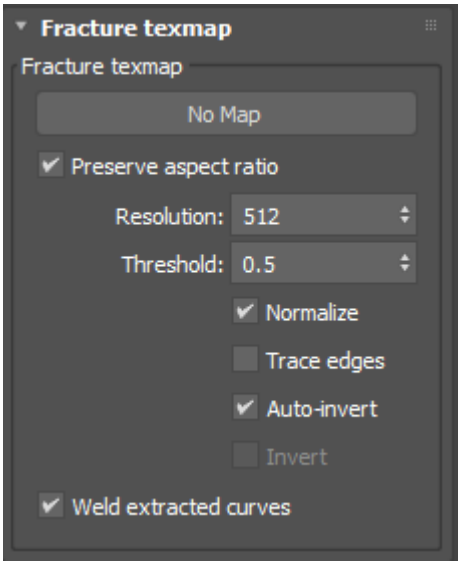
Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.

Fracture texmap Rollout

BACK

The Fracture texmap rollout contains parameters for the “fracture texmap” fracture mode. In this mode, a fracture pattern is extracted from a texture map and applied to fracture planes scattered over fracture points on the input mesh.



Texmap: the texture map from which to derive the fracture pattern.

Preserve aspect ratio: when enabled, the aspect ratio of the texmap will be preserved, even if it does not match the default scale of the underlying fracture plane onto which the fracture pattern is applied.

Resolution: the sampling resolution used to extract the fracture pattern from the texture map.

Threshold: pixel values in the texture map above this luminosity value will be extracted as white, all other pixels will be extracted as black.

TIP
Fracture patterns extracted from texture maps are derived from black/white pixel values only. The threshold value allows you to control how a texture map is converted into a purely black and white image (with no intermediate grayscale values).

Auto-invert: when enabled, the pattern extraction engine will automatically invert the black and white values of the converted image, if it appears as though the black values are what should be skeletonized, based on the ratio of black-to-white pixels within the converted image.

Invert: controls whether the image will be manually inverted, when auto-invert is disabled.

Normalize: when enabled, input texture maps are normalized based on their min/max luminosity, prior to conversion to purely black/white images.

Trace edges: when enabled, the edges of the input texture map will be traced prior to conversion into a fracture pattern.

INFO
The fracture pattern extraction algorithm works by finding the “skeleton” of a black and white image (a “skeleton” in this case is a minimal set of lines that follow shaded areas of the image). By first tracing the edges of the image, a more intuitive “skeleton” may be extracted, if the image is not already composed of lines.

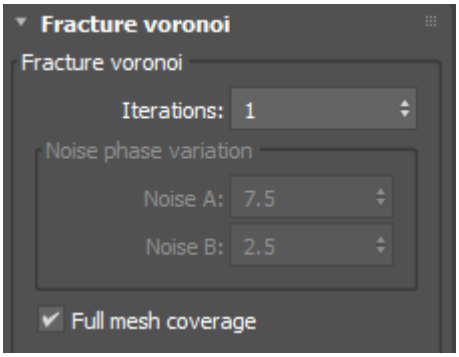
INFO
The fracture pattern extraction algorithm will create a skeleton from white pixels, rather than black pixels (black pixels are assumed to be bordering the patterned areas. Therefore, if the image appears to be inverted (more white pixels exist than black pixels), the auto-invert parameter will flip the values for optimal skeletonization.

Weld extracted curves: controls whether the splines of the extracted fracture pattern will be welded together, prior to meshing.

Fracture voronoi Rollout

BACK

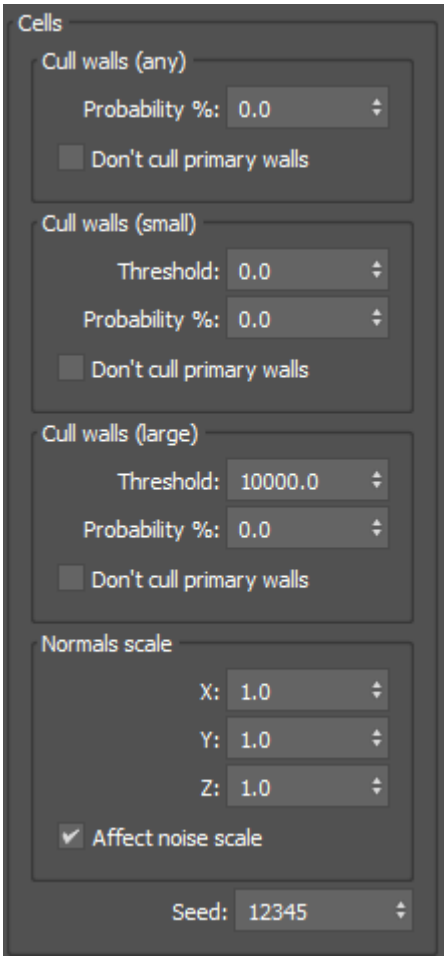
The Fracture voronoi rollout contains parameters for the “fracture voronoi” fracture mode. In this mode, convex cells are generated from fracture points and converted into a fracture mesh.



Iterations: controls the number of unique cell clusters that will be generated from the fracture points, whose overall shape will be varied by the noise phase parameters.

Noise A/B phase variation: controls how noise values applied to voronoi cells will vary on a per-iteration basis.

Full mesh coverage: when enabled, the cell walls generated from the fracture points will have full coverage over the hull of the input mesh. When disabled, cell walls may not have full coverage over the hull of the input mesh, depending on their size.



Cells

Voronoi patterns generate convex cells by default. However, by culling various cell walls, concave cells can be generated - allowing for more interesting fracture patterns.

TIP
When culling cell walls, it is highly recommended to enable the “cull internal open elements” setting in the fracture meshes rollout. More information about that setting can be found in the documentation for that rollout.

Cull walls (any)

Probability %: the probability that any given cell wall will be culled.

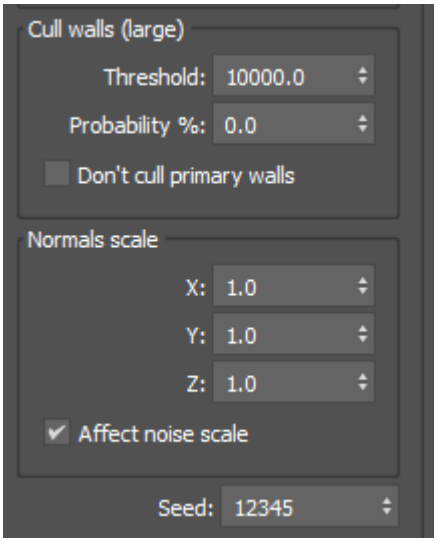
Don’t cull primary walls: when enabled, only cell walls generated in voronoi iteration 2 or above will be culled, leaving the primary cell walls intact.

Cull walls (small)

Threshold: only cell walls with a surface area below this threshold will be culled by the small walls probability.

Probability %: the probability that a given small cell wall will be culled.

Don’t cull primary walls: when enabled, only small cell walls generated in voronoi iteration 2 or above will be culled, leaving the primary cell walls intact.



Cull walls (large)

Threshold: only cell walls with a surface area above this threshold will be culled by the large walls probability.

Probability %: the probability that a given large cell wall will be culled.

Don't cull primary walls: when enabled, only large cell walls generated in voronoi iteration 2 or above will be culled, leaving the primary cell walls intact.

Normals scale

X/Y/Z: these multipliers will bias the angle of voronoi cell walls along each axis. Decreasing the value for a given axis will stretch cell walls along that axis. Increasing the value for a given axis will squash cell walls along that axis.

Affect noise scale: when enabled, noise values will be scaled along each axis by the normals scale values specified. Enabling this setting keeps noise scale consistent with the scale of cell walls, along a particular axis. Disabling this setting means noise scale is independent of cell wall scale.

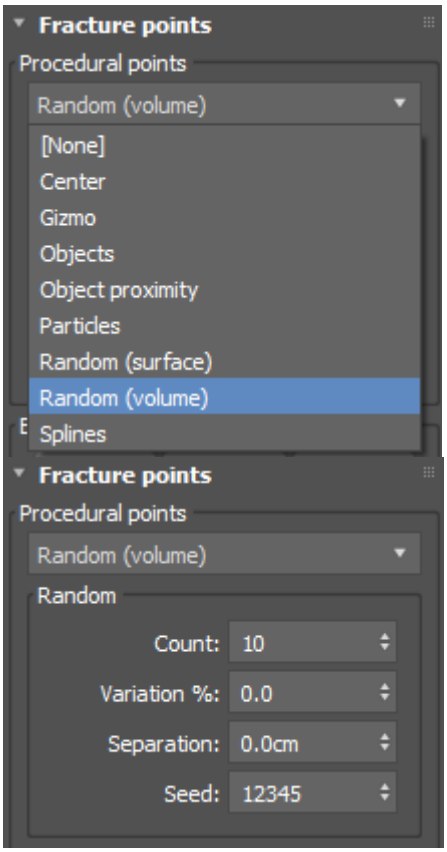
Seed: the seed value for all varied parameters

Fracture points Rollout

BACK

The Fracture points rollout has parameters which provide control over how intermediate fracture points are scattered on meshes. Those points are then used to scatter fracture planes, or generate Voronoi cells, etc. Some fracture modes do not require fracture points (ex: paint fracture), so this rollout may not always appear (depending on which fracture mode you have selected).

Certain parameters are only visible when using the Multifracture **operator, over the** tyMultifracture **modifier, and vice-versa.**



[None]: when selected, only painted points will be used to generate fractures.

Center: when selected, a single point will be generated at the center of the input mesh

Gizmo: when selected, points will be generated inside the modifier's sub-object gizmo.

Gizmo type: controls the shape of the sub-object gizmo.

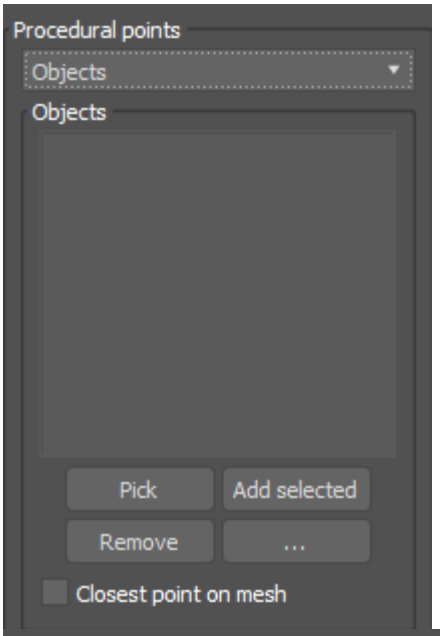
Random

Count: controls the number of points generated in the sub-object gizmo.

Variation %: the per-particle percentage of variation to apply.

Separation: controls the minimum distance that will be enforced between points positions. If a new point is within the minimum distance threshold to previous point, several attempts will be made to move the new point to another position with a valid minimum distance to previous points. If the re-attempts all fail to find a valid position, the point will be culled (ie, setting this value to greater than 0 may result in fewer points being generated than the specified number of points).

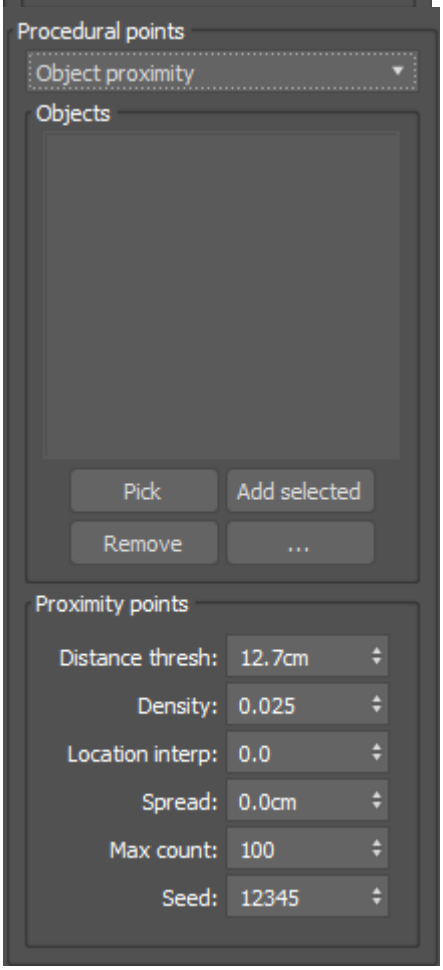
Seed: the seed value for all varied parameters.



Objects

Objects: when selected, points will be generated at the location of specified objects in the scene.

Closest point on mesh: when enabled, generated points will be moved to the nearest location on the input mesh.



Object proximity: when selected, points will be generated on the input mesh, if those locations on the input mesh are within proximity to the specified objects in the scene.

Proximity points

Distance threshold: the maximum distance a proximity point on the input mesh must be to a point on the specified proximity objects' surface, to avoid being culled.

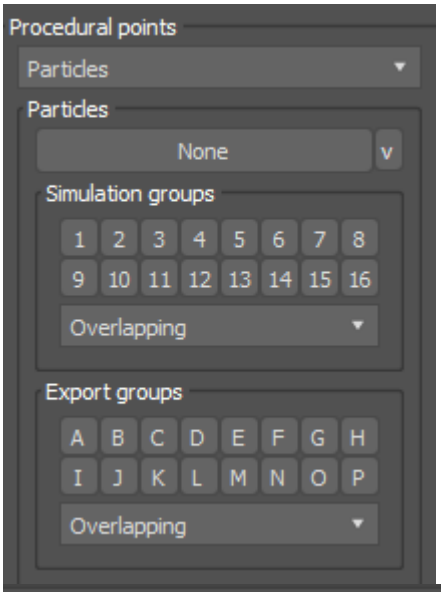
Density: the number of points to generate on the input mesh, for proximity measurements. This value is relative to the surface area of the input mesh. If density is 1.0 and the input mesh has a surface area of 100, then 100 proximity points will be generated.

Location interp: this value interpolates fracture points between their origin on the input mesh and their respective proximal location on the proximity objects' mesh.

Spread: a random spread applied to proximity points, so that they don't all lie uniformly at their point of origin.

Max count: controls the maximum number of non-culled proximity points that will be converted to fracture points.

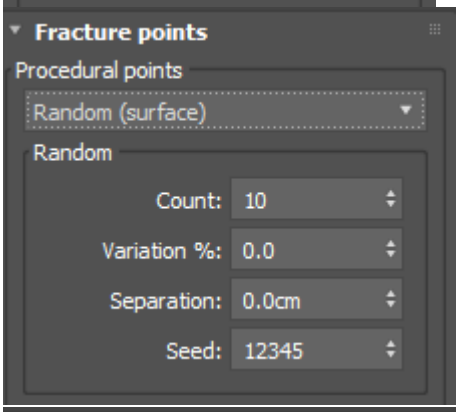
Seed: the seed value for all varied parameters.



Particles: when selected, points will be generated at the location of particles in the specified particle system.

Use this flow's particles: when enabled, only particles from the operator's source flow will be used. When disabled, a user-specified particle system can be chosen.

Simulation/export groups: these settings allow you to filter which particles will be used for generating fracture points



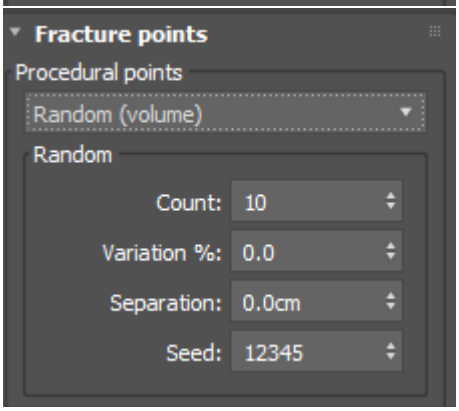
Random (surface): when selected, points will be randomly generated on the surface of the input mesh.

Random

Count: controls the number of points generated on the surface of the input mesh.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.



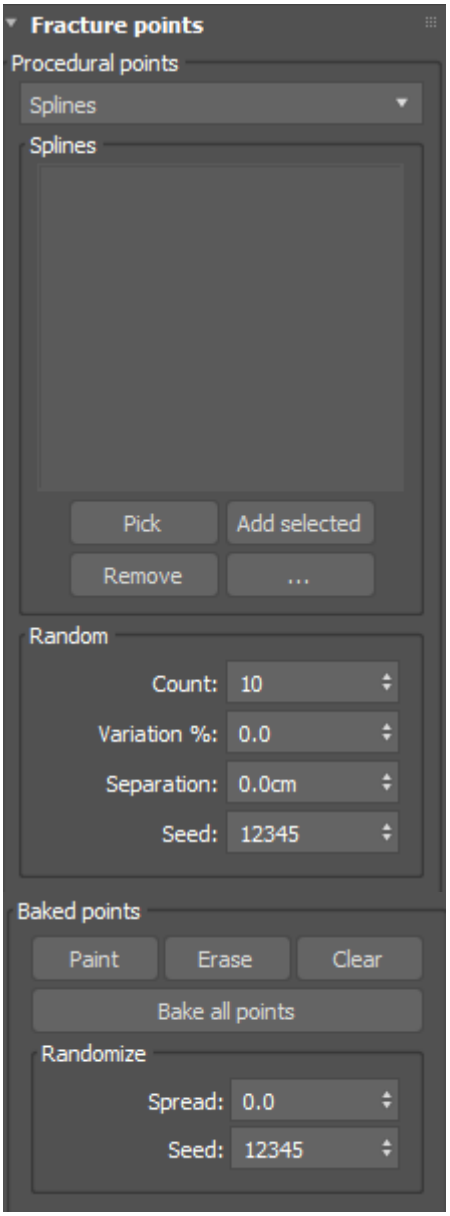
Random (volume): when selected, points will be randomly generated in the volume of the input mesh.

Random

Count: controls the number of points generated in the volume of the input mesh.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.



Splines: when selected, points will be randomly generated along the specified splines.

Random

Count: controls the number of points generated on the splines.

Variation %: the per-particle percentage of variation to apply.

Seed: the seed value for all varied parameters.

Baked points

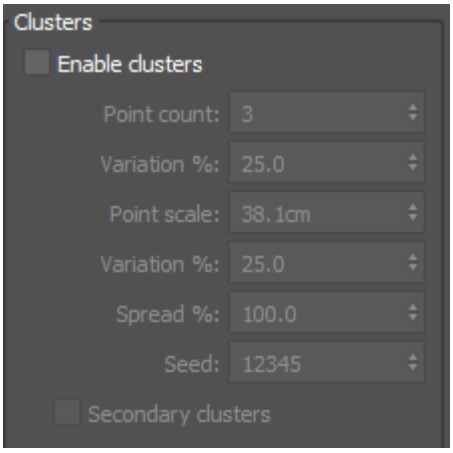
The “baked points” parameters allow you to paint fracture points directly onto input meshes, using your mouse. These baked points can be used in combination with procedural points, or on their own.

Bake all points: when clicked, all procedural points will be converted to baked points. Since baked points can be individually erased with the erase brush, this allows you to erase points which were previously fully procedural (and thus, not individually modifiable).

Randomize

Spread: a random spread applied to baked points, so that they don’t all lie uniformly at their point of origin on the input mesh surface.

Seed: the seed value for all varied parameters.



Clusters

Clusters generate extra points around the initial fracture points, to procedurally increase point density. They are especially useful for the Voronoi fracture mode, which requires at least 2 points to generate fracture cells (some fracture point modes may only generate a single fracture point, so clusters are necessary to achieve a fracture point count of 2+).

Enable clusters: when enabled, cluster points will be generated on the initial fracture points.

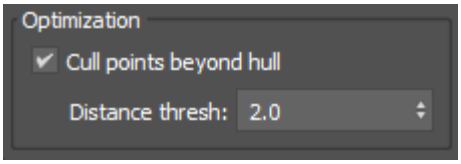
Point count: controls the number of cluster points to generate on a given fracture point.

Variation %: the per-particle percentage of variation to apply.

Point scale: controls the maximum size of an individual cluster of points may be around a given fracture point.

Variation %: the per-particle percentage of variation to apply.

Spread: the maximum amount an individual cluster point's distance may vary, from its parent fracture point to the "point scale" value.



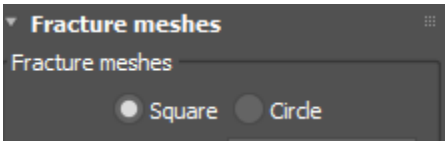
Optimization

Cull points beyond hull: when enabled, generated points that are further from the radius of the input mesh's convex hull, multiplied by the specified distance threshold, will be culled. This prevents distant points (that are too far away from the input mesh to affect its fracture mesh) from slowing down fracture mesh computations.

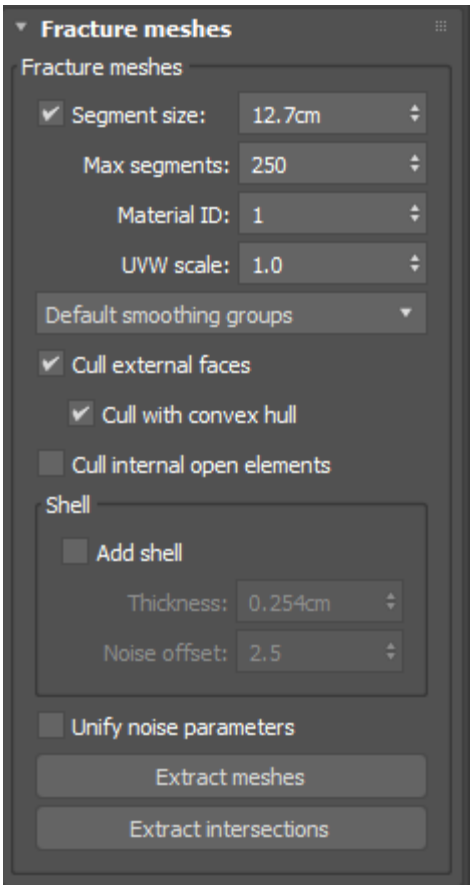
Fracture meshes Rollout

BACK

The Fracture meshes rollout has parameters which provide control over how fracture meshes (meshes generated by the various fracture modes to slice an input mesh) are created.



Square/circle: for fracture modes that use planar meshes to cut input meshes (ex: planar fracture mode), this controls what shape the planar mesh will take. In some situations where having visible edges on planar meshes is undesirable, the "circle" mode can be selected.



Segment size: controls the maximum length of a subdivided edge, on a fracture mesh. When disabled, fracture meshes will not be subdivided. When enabled, lower values generate more dense fracture meshes (resulting in more detailed fractures).

Max segments: controls the maximum number of subdivision segments that may be generated for a given axis or dimension of a fracture mesh. Smaller values will prevent fracture meshes from being generated with too many segments, regardless of what value is specified for the segment size parameter.

Material ID: controls what material ID value will be assigned to all fracture meshes.

UVW scale: this value is a scale multiplier applied to all UVW coordinates assigned to fracture meshes.

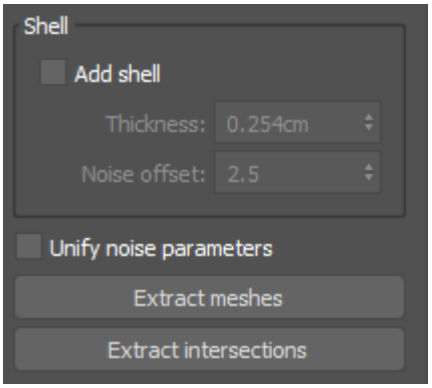
Smoothing mode: controls whether fracture meshes will be assigned smoothing groups.

Cull external faces: when enabled, faces on fracture meshes that do not intersect the hull of the input mesh will be culled, prior to fracture. Typically these faces are not required (as they do not affect the fracture result), so leaving this setting enabled will increase the overall speed of the fracture operation.

Cull with convex hull: when enabled, the convex hull of the input mesh will be used to cull external fracture meshes faces. When disabled, the input mesh itself will be used to compute fracture mesh intersections and cull external faces.

Cull internal open elements: when enabled, open elements (fracture mesh elements with open edges) that are inside of the input mesh hull will be culled prior to fracture.

TIP
Enabling "cull internal open elements" can help prevent fracture artifacts in cases where it is known that fracture meshes have open elements inside the input mesh hull. For example, if you are performing a Voronoi fracture and have enabled cell wall culling, it is likely that orphan cell walls (cell walls that don't connect to other cell walls at a junction) will be generated. Enabling "cull internal open elements" will remove those orphan walls, leading to results that are less likely to contain artifacts (because the **PRISM** volume slice function prefers fracture meshes that fully cross the surface boundaries of input meshes, rather than those that only partially intersect it).



Shell

Add shell: when enabled, fracture meshes will be extruded along their surface normals, giving them volume.

Thickness: controls how thick the shell extrusion will be.

Noise offset: controls how shell noise will be offset, on a per-fracture-mesh basis. Increasing this setting will ensure that multiple fracture meshes will have the appearance of more randomized shell noise.

Unify noise parameters: when enabled, all fracture modes will share the same noise A/B parameters. When disabled, each mode will have its own unique noise A/B parameters.

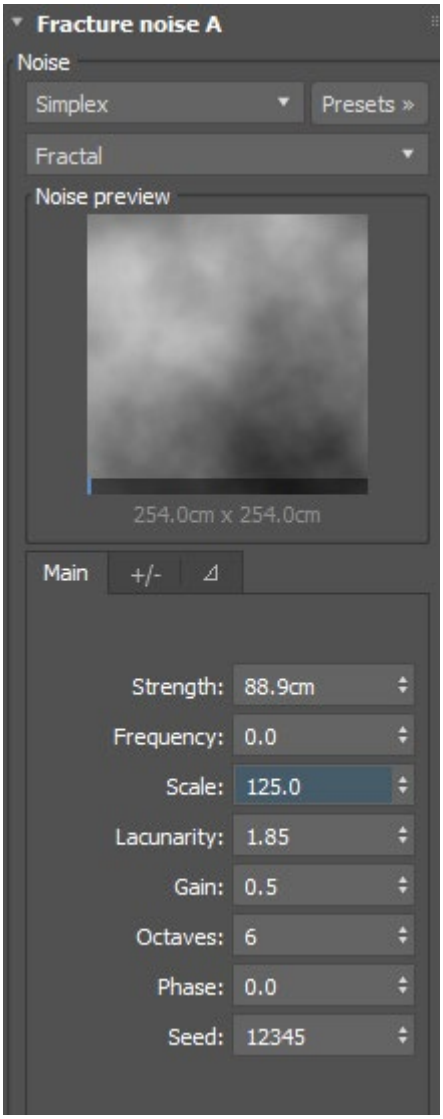
Extract meshes: clicking this will extract fracture meshes to a new editable mesh object.

Extract intersections: clicking this will extract fracture mesh intersection lines to a new editable spline object.

Fracture noise Rollouts

BACK

The Fracture noise rollouts have parameters which allow you to tune how fracture meshes are displaced, prior to fracture. These displacements are what create fracture details.



Fracture noise A/B are two separate levels of noise that you can apply to any fracture mesh (common usage of the two levels are: one level for larger displacements and one level for smaller, more detailed displacements - layered together for more control over the overall shape of the noise through the fracture mesh).

Fracture shell noise specifically applies to fracture meshes that have been extruded with the shell settings available in the Fracture meshes rollout. Fracture noise A/B displace fracture mesh vertices directly, whereas Fracture shell noise varies the distance that a shell operation will extrude a fracture mesh, at a given point in space.

Radial random walk specifically applies to the radial lines of radial fracture patterns.

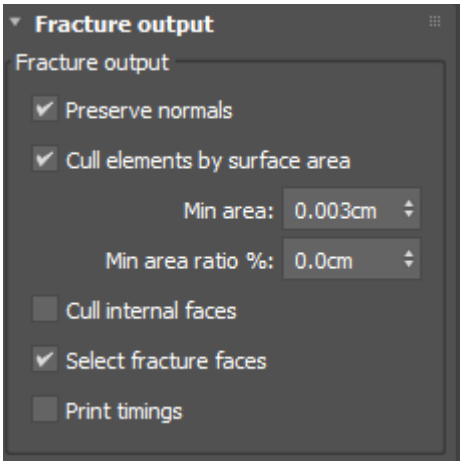
The individual noise parameters themselves are identical to other noise parameters found throughout **tyFlow** (with the exception of some **Multifracture**-specific noise presets available in the noise rollouts, from the “Presets” menu button), and are not outlined further here.

Fracture output Rollout

BACK

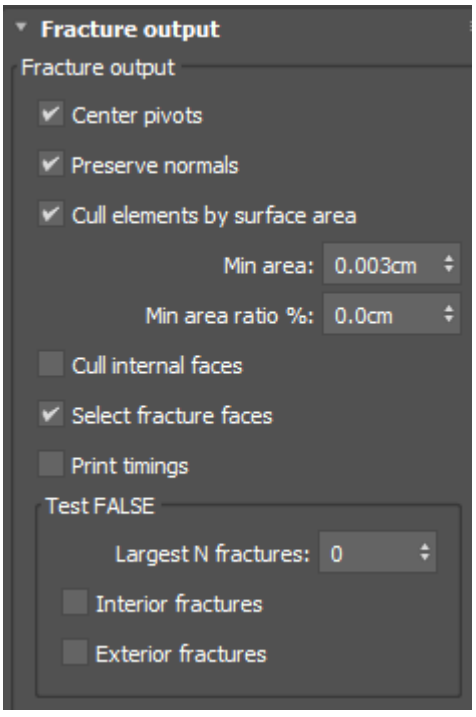
The Fracture output rollout has parameters to control how the output of the modifier/operator is processed.

MODIFIER VERSION



Preserve normals: when enabled, explicit normals of the input mesh will be preserved in the fracture meshes.
Cull elements by surface area: when enabled, fracture meshes will be culled depending on whether or not they have a surface area smaller than the specified thresholds.
Min area: the absolute minimum surface area a fracture mesh must have, to avoid being culled.
Min area ratio %: the minimum surface area a fracture mesh must have, relative to the largest surface area of any given fracture mesh, to avoid being culled. For example, if set to 1%, all fracture meshes whose area is less than 1% the area of the largest fracture mesh will be culled.
Cull internal faces: when enabled, non-manifold faces that are internal to a fractured mesh (ex: remnants of slice mesh that didn't fully cross the input mesh bounds), will be removed.
Select fracture faces: when enabled, fracture faces (faces derived from the fracture mesh(es) in the fracture operation) will be added to the output mesh face selection.
Print timings: when enabled, **PRISM**-specific function timings will be printed to the MAXScript listener, providing a full breakdown of the speed of each **PRISM** function during the fracture operation. These timings do not include the time spent to compute the fracture operand points and meshes themselves - only the actual fracturing operations performed by **PRISM**.

OPERATOR VERSION

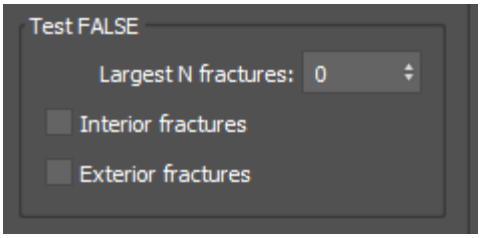


Center pivots: when enabled, fracture particle meshes will have their pivots centered.

Preserve source particle: when enabled, the largest fracture mesh will be assigned to the source particle (and the source particle will not be deleted), thereby preserving any additional properties/binds the source particle may have had which would otherwise be deleted.

Area threshold %: if the ratio between the largest fracture mesh's surface area and the input mesh's surface area is not greater than this percentage, the source particle will not be preserved.

TIP
Preserving the source particle can be useful in cases where you want to preserve binds or other unique properties assigned to it, even after it's fractured. However, there may be some cases where you may not want to preserve those properties - for example, if the source particle is fractured into sufficiently small enough pieces. By adjusting the area threshold %, you can control how big the largest fracture mesh must be in order for it to preserve the source particle's properties/binds. For example, an area threshold value of 90% means the largest fracture mesh must have at least 90% of the surface area of the input mesh, in order for the source particle to be preserved.



Test FALSE

Largest N fractures: controls how many of the largest fractures generated in the fracture operation will test FALSE for any operation conditions (ex: send out).

Interior fractures: when enabled, fractures that originate from inside a fracture mesh (ex: inside of a fracture mesh with volume) will test FALSE for any operation conditions (ex: send out). This setting should only be enabled if a fracture mesh has volume (has shell thickness applied to it).

Exterior fractures: when enabled, fractures that originate from outside a fracture mesh (ex: outside of a fracture mesh with volume) will test FALSE for any operation conditions (ex: send out). This setting should only be enabled if a fracture mesh has volume (has shell thickness applied to it).

TIP
There any many situations where you might only want to send part of a fracture result to another particle event. For example, when performing an edge fracture on a mesh, you may only want to send the resulting edge fragments to the next event, while keeping the rest of the fractured mesh in the current event. You can perform this filtering with Property Test operators, or other methods, but by simply specifying how many of the largest fractures you want to keep in the current event (in that example: 1), you can achieve the same result without having to tweak specific values and settings related to particle size/volume/etc.