

Autodesk®
Maya®
2012



Autodesk

nParticles Advanced Techniques

Contents

Chapter 1	1
nParticles Advanced Techniques	1
Overview	1
Preparing for the lessons	2
Understanding the tutorial components	2
Lesson 1 Creating the molten slag	4
Overview	4
Setting up the simulation	4
Creating the nParticle slag	5
Converting the nParticles to an output mesh	11
Adjust the Nucleus solver settings	14
Assigning a texture to the slag	15
Lesson 2 Creating the slag smoke	17
Overview	17
Setting up the simulation	17
Creating the fluid smoke and flames	19
Setting fluid shading attributes	27
Increasing the quality of the fluid simulation	29
Lesson 3 Creating flying sparks	32
Overview	32
Setting up the simulation	33
Creating the nParticle collision event	34
Creating the nConstraint	37
Setting up the second stage of the sparks effect	38

Creating the nParticle sparks	41
Setting the nParticle Sparks attributes	45
Setting the nParticle sparks shading attributes	47
Controlling the nParticle spark behavior	51
Beyond the tutorial	55

nParticles Advanced Techniques



Overview



This tutorial shows you how to create the effect of molten metal being poured down a chute and into a collecting tub. The tutorial combines a number of different effects including three types of nParticle systems, an nParticle output mesh, a 3D fluid with container, and an external gravity field. You can use these components to create effects that combine nParticle systems and fluids such as lava flows, flowing liquids, or other effects.

To watch a video of the rendered simulation, open *SlagPourFoundryFinal.mp4v*, included with the tutorial lesson files.

Preparing for the lessons

This tutorial assumes that you:

- You understand the basic concepts of polygon modeling, animation, fluids, and nParticles.
- Copied the nParticles Advanced Techniques folder from the following location: <http://www.autodesk.com/maya-advancedtechniques>. Then, set the `nParticlesAdvancedTutorials` directory as your Maya project.
- Selected the **nDynamics** menu set. Unless otherwise noted, the instructions in this chapter assume you've already selected the **nDynamics** menu set.

Understanding the tutorial components

This tutorial shows you how to set up a simulated slag foundry that is composed of the following components:

nParticle objects

You create a total of three nParticle systems to simulate the slag foundry scene. All three systems are controlled by the same Nucleus solver node. The first nParticle system (*nParticle_slag*) added to the scene simulates the slag that flows down the chute. You convert this **Ball** style nParticle object to an nParticle output mesh then shade and texture it with a fractal texture. The assigned texture gives the output mesh the appearance of molten metal when rendered. You create, adjust, and shade the *nParticle_slag* and output mesh objects, in Lesson 1. See [Lesson 1 Creating the molten slag](#) (page 4).

The remaining two nParticle systems simulate the metal sparks that fly off the slag flow. To generate the sparks, you create a two-stage effect that is initiated by a particle collision event. The *nParticle_slag* object that you create in Lesson 1 acts as the source particle system for the collision event.

After the resulting target particle system (*nParticle_sparks_emitter*) is generated by the event, it becomes a surface emitter for the nParticle sparks (*nParticle_sparks*). To create the look and streaking behavior of the sparks, you use the **Tube (s/w)** particle render type. The two-stage sparks effect is created in Lesson 3. See [Lesson 3 Creating flying sparks](#) (page 32).

nParticle output mesh

An nParticle output mesh is generated from the nParticle_slag object. This mesh is the visible slag that flows down the chute into the tub. It is also used as a surface emitter for a 3D fluid. The fluid effect creates the smoke that rises above the molten slag as it slides down the slag chute into the tub. The output mesh is textured with a 2D fractal texture to give it the orange glow of molten metal when the scene is rendered. The mesh is created and adjusted in Lesson 1. See [Lesson 1 Creating the molten slag](#) (page 4).

3D fluid effect

The 3D fluid is emitted into the scene from the surface of the nParticle output mesh. It simulates the smoke and steam that rises from the molten slag as it flows down the chute. The fluid container uses Fluid Auto Resize so that it can follow the slag while maintaining a manageable container size. The fluid is created in Lesson 2. See [Lesson 2 Creating the slag smoke](#) (page 17).

External gravity field

You apply an external **Gravity** field to the nParticle_Sparks object to create the arcing behavior of the sparks as they fly off the slag. Also, to create realistic spark behavior, you will simulate with **Ignore Solver Gravity** turned on. The gravity field is added to the scene in Lesson 3. See [Lesson 3 Creating flying sparks](#) (page 32).

Non-dynamic objects

Lights, a camera, and render settings are added to the scene so that you can render out a frame or part of a sequence at various stages of the tutorial.

Tutorial workflow

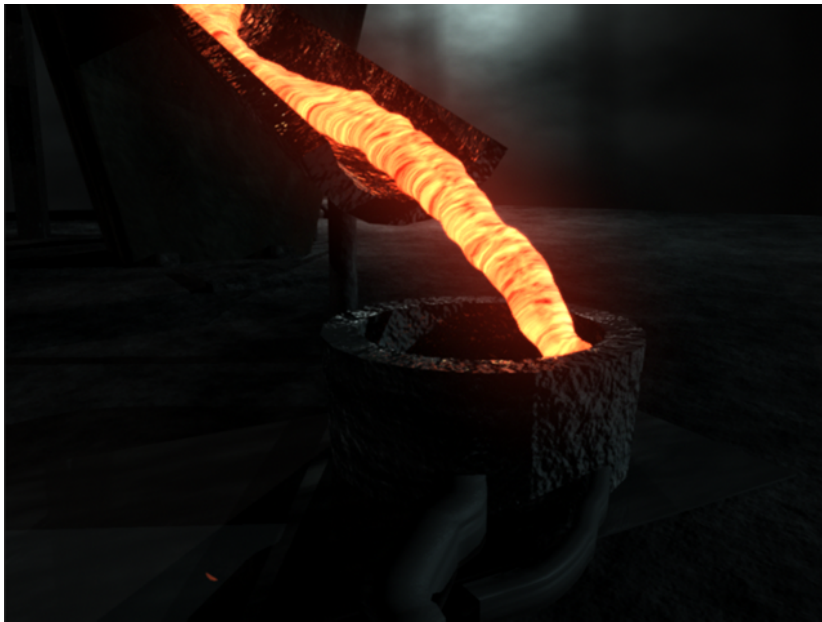
The scene files for each lesson let you complete specific aspects of the tutorial. You can start the tutorial using the scene file for Lesson 1 and continue to add the tutorial components until you complete all four lessons. Otherwise, each lesson can be completed separately using the following scene files:

- For Lesson 1: Creating the molten slag, use SlagPourFoundry_1.mb
- For Lesson 2: Creating the slag smoke, use SlagPourFoundry_2.mb
- For Lesson 3: Creating flying sparks, use SlagPourFoundry_3.mb
- For Beyond the tutorial, use SlagPourFoundry_4.mb

To watch a video of the rendered simulation, open *SlagPourFoundryFinal.mp4v*, included with the tutorial lesson files.

Lesson 1 Creating the molten slag

Overview



In this lesson, you create the nParticle object and output mesh that simulate molten slag. After adjusting the nParticles and the output mesh so the slag flows down the chute, you apply a texture to the mesh, which gives the surface the molten red-orange glow. You render single frames of the simulation to view its overall appearance.

Setting up the simulation

Set up the lesson by doing the following:

- 1 Open *SlagPourFoundry_1.mb*.

- 2 In the **Preferences** window do the following:
 - Under **Categories**, select **Time Slider**.
 - In the **Playback** section, set **Playback speed** to **Play every frame**.
 - Set **Max Playback Speed** to **Real-time [24 fps]**.
- 3 In the **Outliner**, select the *geo_chute* and *geo_tub* objects and convert them to passive collision objects by selecting **nMesh > Create Passive Collider**.

When you create the passive collision objects, a nucleus node (nucleus1) is also created to control all Nucleus objects in the simulation.
- 4 Rename *nRigid1* and *nRigid2* to *nRigid_chute* and *nRigid_tub*.

Renaming your objects makes them easy to identify in the **Outliner** and **Attribute Editor**.
- 5 In the **Outliner**, select one of the nRigid objects.
- 6 In the **Attribute Editor**, switch to the nucleus1 tab, and in the **Ground Plane** section, set the following:
 - Turn on **Use Plane**.
 - For **Plane Origin**, set the Y axis to -3.0.
 - Set **Plane Bounce** to 0.7
 - Set **Plane Friction** to 0.16.
 - Leave other nucleus node settings to the default values.

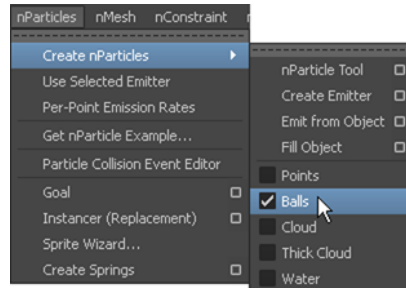
This **Plane Origin** sets the Nucleus plane level with the foundry floor. nParticles that collide with the Nucleus plane will appear to collide with the floor so you do not need to convert the floor geometry to a passive collision object. Collisions with the Nucleus plane are computed faster than collisions with passive collision object, which can decrease simulation time.

Creating the nParticle slag

To create the nParticle object that simulates the slag flow, you use the **Ball** style nParticles. Note that later in the lesson, you convert the nParticle object to an output mesh.

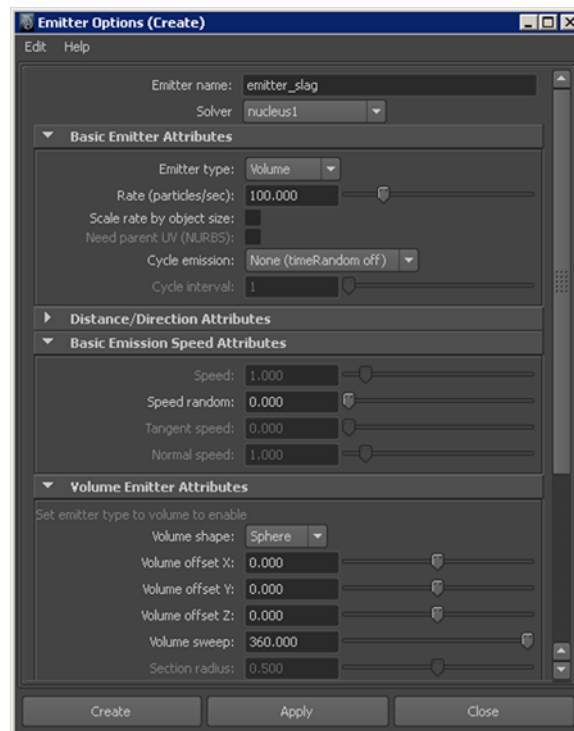
To create the nParticle object

- 1 Set the nParticle style to **Balls** by selecting **nParticles > Create nParticles > Balls**.



- 2 Create the nParticle and emitter objects by selecting **nParticles > Create nParticles > Create Emitter >** .

The **Emitter Options (Create)** window appears.

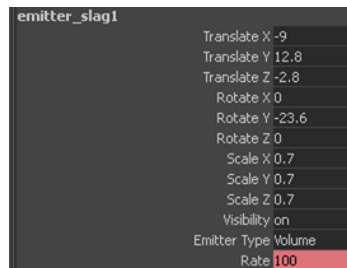


- 3 In the **Emitter Options (Create)** window, reset to the default values by selecting **Edit > Reset Settings** and do the following:
 - For **Emitter name**, type *emitter_slag*.
 - If the **Solver** list appears, select nucleus1.
 - In the **Basic Emitter Attributes** section, set **Emitter type** to **Volume**.
 - In the **Volume Emitter Attributes** section, select **Sphere** from the **Volume shape** list.
 - In the **Volume Speed Attributes** section, set **Away from center** to 0 and **Directional speed** to 5.0.
- 4 Click **Create**.
- 5 In the **Outliner**, rename the new nParticle object to *nParticle_slag*.

Next, position the emitter_slag object at the end of the spout (geo_slag_bin_spout) so that the slag appears to pour from the bin onto the slag chute.

To position and set the emitter

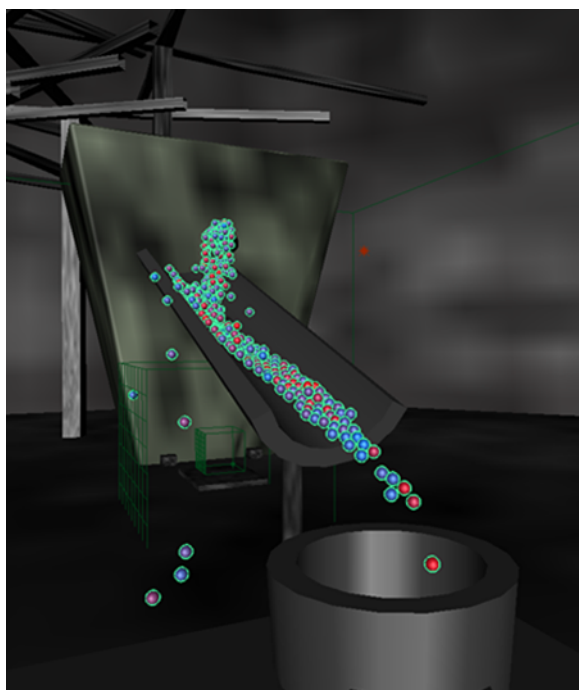
- 1 In the **Outliner** select the emitter_slag object.
- 2 In the **Channel Box** enter the following values:
 - **Translate X:** -9.0
 - **Translate Y:** 12.8
 - **Translate Z:** -2.8
 - **Rotate Y:** -23.6
 - **Scale X:** 0.7
 - **Scale Y:** 0.7
 - **Scale Z:** 0.7



For this simulation, only a brief stream of nParticles is necessary, rather than a continuous flow. To create the slag stream, you key the emitter_slag object so that it stops emitting particles at frame 85. This produces enough slag to slide down the chute and into the tub before the simulation ends.

- 3 Rewind the simulation to the start frame.
- 4 In the **Channel Box**, set **Rate** to 100.
- 5 Right-click **Rate** and select **Key Selected** from the pop-up menu. A keyframe is added to the Time slider.
- 6 Play the simulation to frame 85, right-click **Rate** and select **Key Selected**.
- 7 Step the simulation forward one frame and in the **Channel Box**, set **Rate** to 0.
- 8 Right-click on **Rate** and create another keyframe.
- 9 Play the simulation.

Most nParticles emit onto the chute, slide down the incline, and land in the slag tub. Notice that some of the nParticles fall off the chute and land on the foundry floor.



Since you will later convert the nParticles to an output mesh, you want to ensure that no particles move too far away from the location of the nParticle mass as this enlarges the bounding box area of the resulting mesh. The **Output MeshMax Triangle Resolution** defines the area of the mesh bounding box.

If stray nParticles cause the bounding regions of the mesh to grow beyond the area set by **Max Triangle Resolution**, Maya increases the **Mesh Triangle Size** to compensate. A sudden increase in **Mesh Triangle Size** can quickly reduce the quality of the mesh and distort its appearance. The distortions in the mesh are often seen as flickering or popping on the mesh surface. For this reason, nParticle output meshes work best for small scale effects, such as liquid flows that do not cover a large area of the scene.

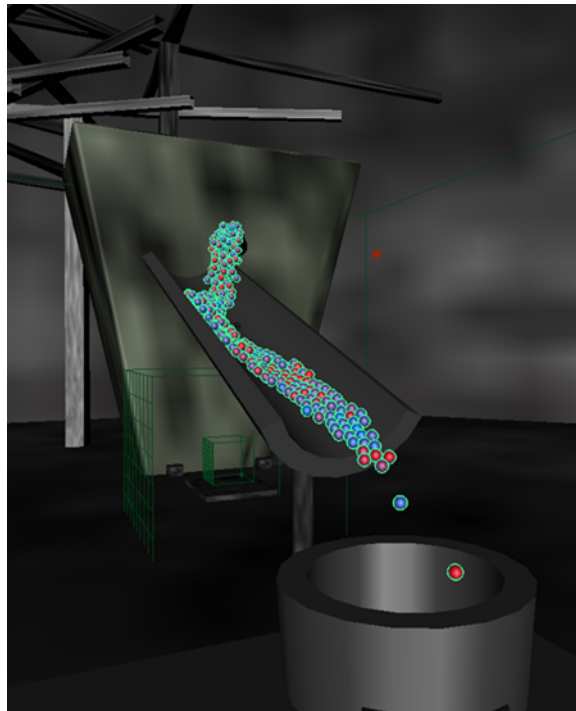
For information about **Max Triangle Resolution** and **Mesh Triangle Size**, see *Output Mesh* in the *nParticleShape node* section of the *nDynamics* Help.

In this section of the lesson, you turn down the **Bounce** attribute so that the particles have no additional inertia making them fall off the slag chute.

To set **Bounce**

- 1 In the **Outliner**, select *nParticle_slag*.
- 2 In the **Attribute Editor**, switch to the *nParticle_slagShape* tab, and in the **Collisions** section, set **Bounce** to 0.
- 3 Rewind the simulation and play it back.

With **Bounce** at 0, the nParticles stay on the slag chute. Also, the particles are causing some side-to-side movement, which will make the texture look more realistic when applied.



If the problem with particles escaping persists after you turn down the **Bounce**, you can use either **Friction** or **Stickiness** to further stabilize the particles on the chute.

Converting the nParticles to an output mesh

In this section of the lesson, you convert the `nParticle_slag` object to an output mesh. Converting the nParticle system to an output mesh lets you take greater advantage of rendering effects, such as shader networks, textures, and motion blur, to finish off the simulation.

After the nParticles are converted to an output mesh, you can still use the nParticle object attributes to adjust the behavior of the slag.

To convert the nParticle object to an output mesh

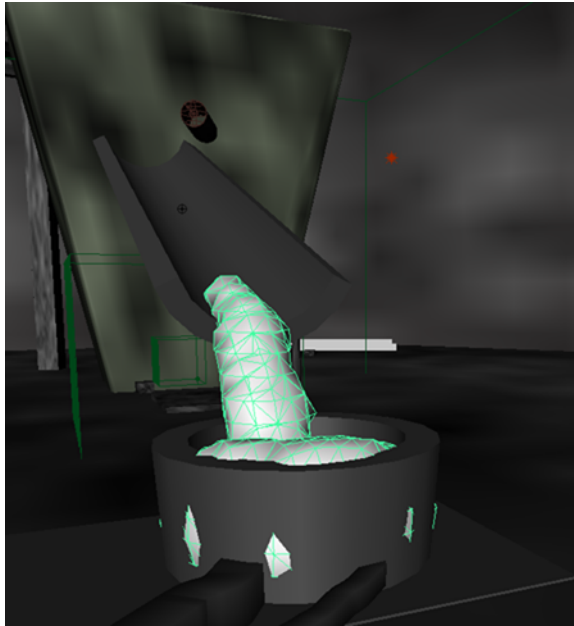
- 1 In the **Outliner**, select the `nParticle_slag` object and select **Modify > Convert > nParticle to Polygons**.

Notice that the particles are no longer displayed in the scene. When converted to a mesh, the `nParticle_slag` object becomes an intermediate object, which is not visible. This reduces simulation time and makes it easier to see how **Output Mesh** attribute adjustments affect the mesh behavior.

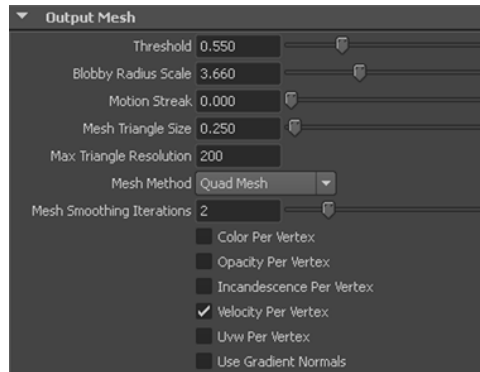
NOTE To make the nParticle object appear in the scene, in the **Object Display** section of the `nParticle_slagShape` node **Attribute Editor**, turn off **Intermediate Object**. You will need to do this later in the tutorial.

- 2 In the **Outliner**, rename the `polySurface1` to `geo_slag`.
- 3 Rewind and play the simulation.

Notice that the output mesh is interpenetrating the `nRigid_tub` geometry. Adjusting the nParticle object **Bloppy Radius Scale** and **Threshold** values can help fix the interpenetrations.



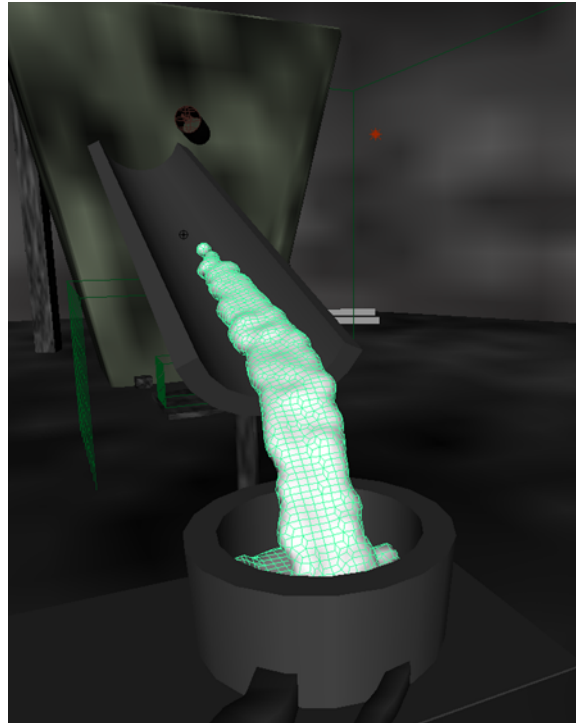
- 4 In the **Outliner**, select the *nParticle_slag* object, then click the **nParticle_slagShape** tab in the **Attribute Editor**.
- 5 In the **Output Mesh** section, set the following:
 - **Threshold:** 0.55
 - **Bloppy Radius Scale:** 3.66
 - **Mesh Triangle Size:** 0.25
 - **Max Triangle Resolution:** 200
 - **Mesh Method:** Quad Mesh
 - **Mesh Smoothing Iterations:** 2



For information about Output Mesh attributes, see *Output Mesh* in the *nParticleShape* node section of the *nDynamics* Help.

6 Rewind and play the simulation.

Notice that the output mesh no longer interpenetrates the *nRigid_tub* geometry.



Also notice that generating an nParticle mesh slows down the simulation. In particular, low **Mesh Triangle Size** values and **Mesh Smoothing Iterations** greater than 3 can quickly slow simulation time and decrease overall performance. However, both settings increase the quality of the mesh. For larger simulations involving output meshes, caching the nParticles can decrease simulation time.

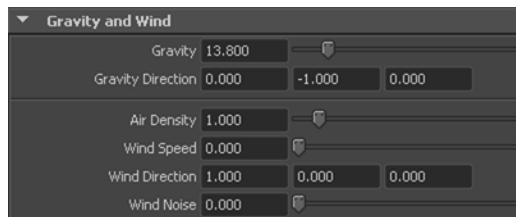
In the next section you set Nucleus solver attributes to further improve the appearance and behavior of the mesh.

Adjust the Nucleus solver settings

In next steps, you will improve the mesh behavior by increasing the Nucleus solver **Substeps**. You also increase the Nucleus **Gravity** so that the slag slides down the chute faster.

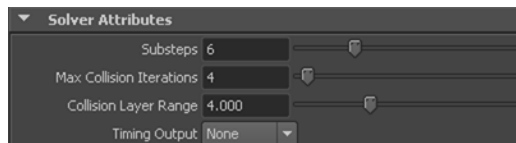
To set nucleus node attributes

- 1 In the **Outliner**, select the *nParticle_slag* object, then click the **nucleus** tab in the **Attribute Editor**.
- 2 In the **Gravity and Wind** section, set **Gravity** to 13.8.



For the other nParticle objects created in this simulation, **Ignore Solver Gravity** will be turned on, meaning that the **Gravity** will not affect them.

- 3 In the **Solver Attributes** section, set **Substeps** to 6.



This controls how the simulation time is broken up into calculation segments. Simulation quality and collision accuracy generally improve

with increasing **Substeps** values. A high number of **Substeps** may result in slower solving.

- 4 Rewind and play the simulation.

Assigning a texture to the slag

Assigning a texture to the mesh gives it the orange glow of molten metal when the scene is rendered. The material that you assign is made from an orange blinn shader with a fractal texture.

When assigned to the mesh, the textures use the UVW coordinates generated by the **Uvw Per Vertex** attribute of the nParticle object.

- 1 In the **Outliner**, select the *nParticle_slag* object, then click the **nParticle_slagShape** tab in the **Attribute Editor**.
- 2 In the **Output Mesh** section, turn on **Uvw Per Vertex**.

With **Uvw Per Vertex** on, Maya generates UVW texture coordinates when you convert an nParticle object to a polygon mesh. The texture coordinates let you map a texture to the surface of your output mesh.

NOTE Be aware that when you use **Uvw Per Vertex**, smoothing the output mesh using **Mesh > Smooth** alters the per-vertex mapping of the UVW coordinates used by the assigned texture.

- 3 Play the simulation until the mesh is in the scene, then stop.
- 4 In the scene view, right-click the mesh and select **Assign Existing Material > blinn_slag**.

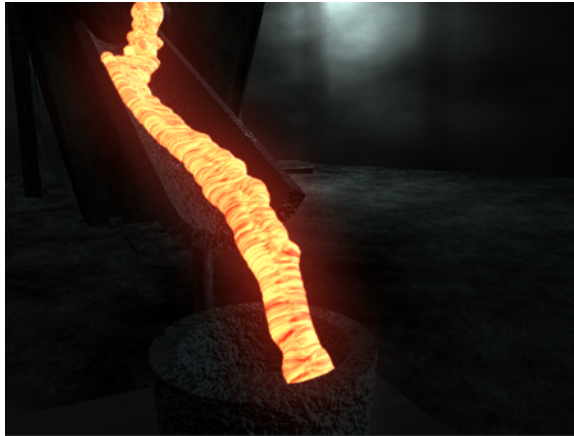
The mesh turns black to indicate that the material is assigned to it. To see the results of the material, you render a single frame of the simulation.

Rendering a single frame

Rendering single frames of your simulation let you observe how attribute adjustments affect its behavior and appearance. For these renders, use the **Maya Software** render with default settings. With these settings, your frames will render quickly, allowing you to move on to the next section of the lesson. If you prefer, you can use **mental ray** for rendering.


- 1 Rewind and play the simulation to about frame 90, then click the

Render the current frame  icon.



You can adjust the material attributes such as **Reflectivity** and **Glow Intensity**, to get the desired look of the molten metal.

You can also improve the quality of the rendered frame by changing the render settings.

- 2 To change the render settings, click the  icon to open the **Render Settings** window.
- 3 Click the **Maya Software** tab and set the following:
 - In the **Anti-aliasing Quality** section, set **Quality** to **Production quality**.
 - In the **Raytracing Quality** section, turn on **Raytracing**.
- 4 Re-render to see the results.

You have now created a realistic simulation of molten slag flowing into a tub. This concludes the section on creating the slag flow. In the next lesson, you create a fluid to simulate smoke and flames coming off the molten slag.

Lesson 2 Creating the slag smoke

Overview



In this lesson, you set up a 3D fluid to emit from the surface of the nParticle output mesh. The fluid simulates smoke and flames that drift off the molten slag as it slides down the chute. The fluid uses **Auto Resize** to dynamically change the container size as it follows the slag down the chute. This keeps the fluid container to a minimal size, which decreases simulation time.

Since you do not modify the nParticle_slag object in this lesson, you can cache it to help decrease the overall simulation time. Be aware that when you continue to Lesson 3, you will need to disable the nParticle cache before starting the lesson.


Setting up the simulation

You can either continue the tutorial using your scene file from the previous lesson or use the scene file designed for the lesson (*slagPourFoundry_2.mb*). If

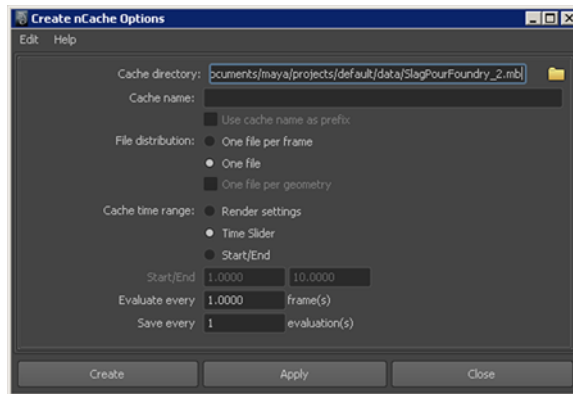
you do not want to do the steps that focus on creating the fluid effect, go to [Lesson 3 Creating flying sparks](#) (page 32). In the scene for this lesson (*slagPourFoundry_3.mb*), the fluid effect is complete.

Before starting the lesson, cache the nParticle_slag object.

To cache the nParticle_slag object

- 1 Open *SlagPourFoundry_2.mb*.
- 2 Rewind the simulation, and in the **Outliner**, select the *geo_slag*.
- 3 Select **nCache > Create New Cache > **.

The **Create nCache Options** window appears.



- 4 In the **Create nCache Options** window, do the following:
 - Set the **Cache directory** to the folder where you want your caches saved.
 - Either type a **Cache name** or leave it set to the default name (*nParticle_slagShape*).


NOTE If you want use a **Cache name** other than the default name, set the **File distribution** attribute first. (see next step). Otherwise, you will need to re-type your custom **Cache name**.

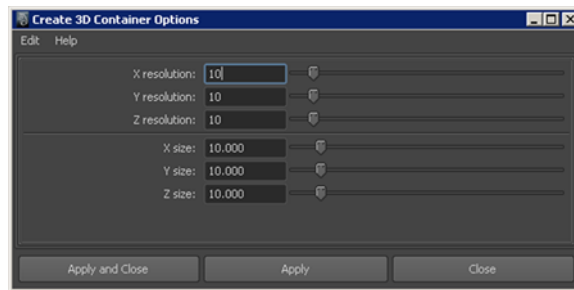
- For **File distribution**, select **One file**.
- Click **Create**.

Creating the fluid smoke and flames

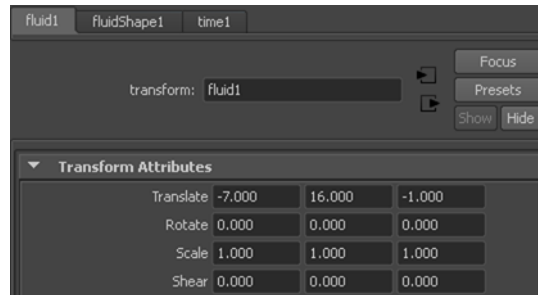
In the next steps, you create a 3D fluid object that emits from the surface of the `geo_slag` polygon object.

To create the 3D fluid container

- 1 Switch to the **Dynamics** menu set.
- 2 Select **Fluid Effects > Create 3D Container** >  .
The **Create 3D Container Options** window opens.




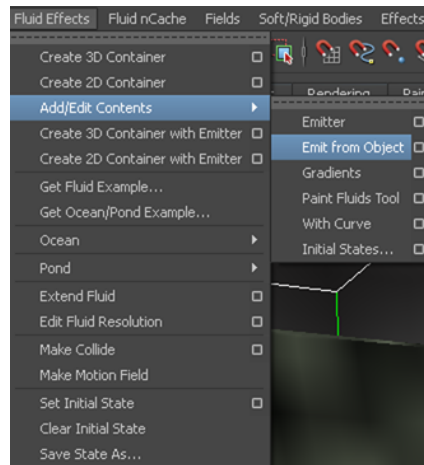
- 3 In the **Create 3D Container Options** window, select **Edit > Reset Settings** and ensure the X, Y, and Z resolution values are 10.
Keeping the fluid resolution values set to the default value of 10 ensures that the fluid simulates quickly. Later in the lesson, you increase the resolution values to improve the quality and overall appearance of the effect.
- 4 Click **Apply and Close**.
- 5 In the **Outliner**, rename fluid1 object to *fluid_slag_smoke*.
- 6 To position the container at the top of the slag chute, in the **Attribute Editor**, click the **fluid_slag_smoke** tab.
- 7 In the **Transform Attributes** section, set the **Translate** values to:
 - X: -7.0
 - Y: 16.
 - Z: -1



The **Translate** values position the fluid container at the top of the slag chute. You can now add the emitter to the fluid container.

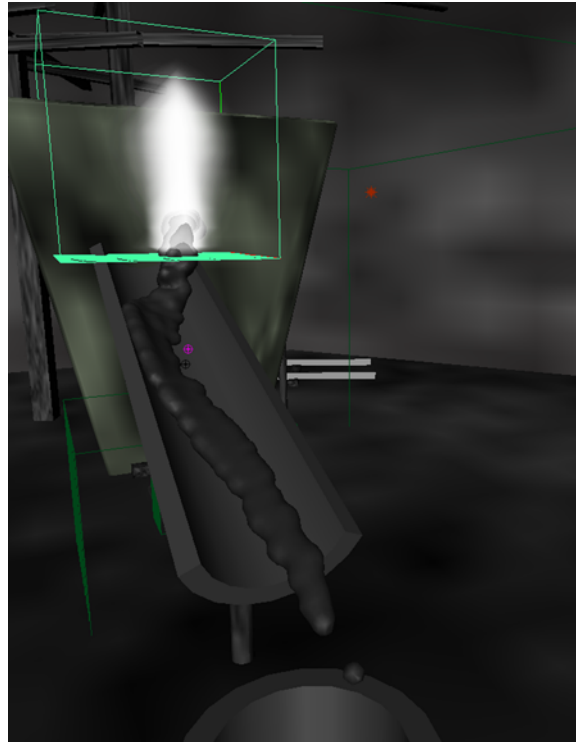
To emit fluid from the slag mesh

- 1 In the **Outliner**, Ctrl-click *fluid_slag_smoke* and *geo_slag*, then select **Fluid Effects > Add/Edit Contents > Emit from Object** >  from the main menu bar.

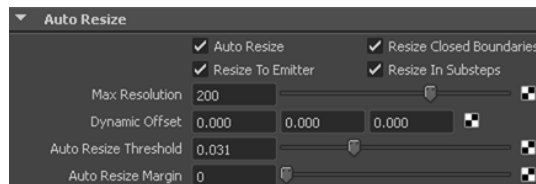


- 2 In the **Emit from Objects Options** window that appears, select **Edit > Reset Settings**, then set the following:
 - For the **Emitter name**, type *emitter_slag_smoke*.
 - In the **Basic Emitter Attributes** section, **Set Emitter type** to **Surface**.
- 3 Click **Apply and Close**.
- 4 Rewind and play the simulation.

Notice that the fluid emits from the slag mesh, but only while the slag passes through the fluid container. Turning on **Auto Resize** enables the fluid container to follow the slag mesh as it emits fluid.



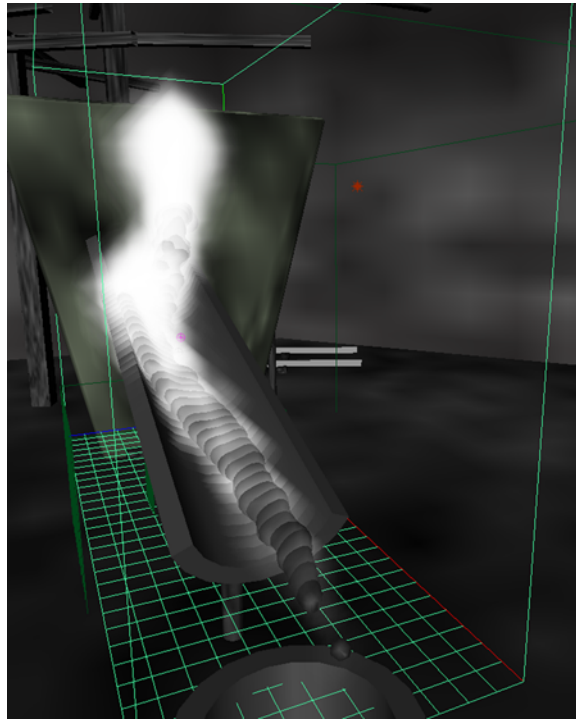
- 5 In the **Outliner** select *fluid_slag_smoke*, then switch to the *fluid_slag_smokeShape* tab in the **Attribute Editor**.
- 6 In the **Auto Resize** section, turn on **Auto Resize**.



- 7 Rewind and play the simulation.
Notice that the fluid container now resizes and follows the movements of the slag. When **Auto Resize** is on, the boundaries (**Size** and **Resolution**) of a fluid container dynamically resize as the fluid density

increases or decreases. **Auto Resize** keeps the fluid container relatively small, so it works well with fluid effects that move quickly, such as a missile vapor trail or the rolling smoke of an explosion.

Turning on **Auto Resize** can increase simulation speed, reduce memory use, creating smaller fluid cache files, and decrease fluid render time.



Notice that the fluid does not look or behave like smoke and flames. To improve the effect you will do the following:

- Reduce the fluid emission rate so that the fluid emits less density and temperature into the container. This helps create the wispy smoke suitable for the effect.
- Adjust the density **Dissipation** attribute, so that the smoke gradually dissipates after it is emitted. For this effect, the smoke density should dissipate fairly quickly.

Adjusting fluid emission

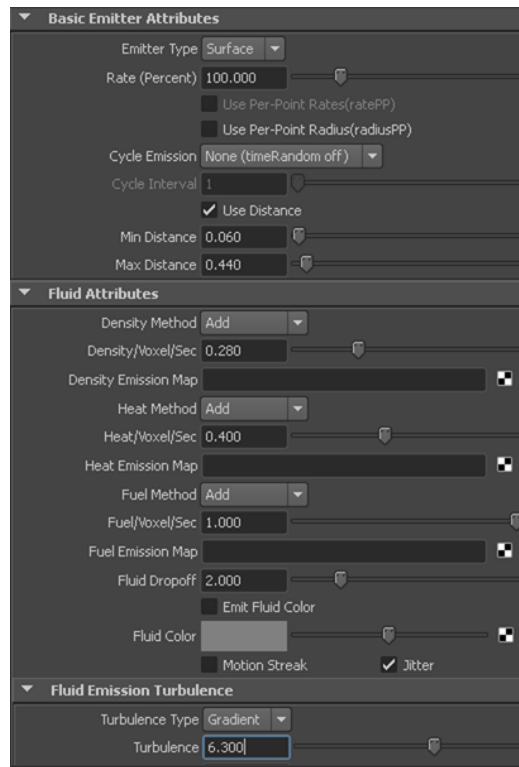
In this section of the lesson, you set attributes to control how the fluid contents emit into the container. For this effect, you emit **Density** and **Heat** into the container. You also add **Turbulence**, to add swirling motion to the fluid.

For more information about fluid effects, see the *Fluid Effects* section in the Maya Help.

To adjust the fluid emission

- 1 In the **Attribute Editor**, click the `emitter_slag_smoke` tab.
If you do not have the fluid selected, you access the fluid emitter object through the **Outliner** by expanding `geo_slag` and selecting `emitter_slag_smoke1`.
- 2 In the **Basic Emitter Attributes** section, do the following:
 - Turn on **Use Distance**.
 - Set **Min Distance** to 0.06.
 - Set **Max Distance** to 0.44.

These values define the minimum and maximum distance that fluid is emitted from the mesh surface.
- 3 In the **Fluid Attributes** section, set the rate of **Density** and **Heat** emission in the fluid container as follows:
 - **Density/Voxel/Sec**: 0.28
 - **Heat/Voxel/Sec**: 0.4
- 4 In the **Fluid Emission Turbulence** section, set **Turbulence** to 6.3.
This adds turbulence to the **Density** and **Heat** as they emit into the container, which generates additional movement in the fluid.



- 5 Rewind and play the simulation.

Notice that the fluid emission is less dense, but there is little motion in the fluid. In the next section, you set fluidShape node attributes to add detail and movement to the smoke.

Setting fluidShape node attributes

Next, you will adjust the **Buoyancy** and **Dissipation** attributes to define the behavior of the **Density** and **Heat** in the fluid container after emission.

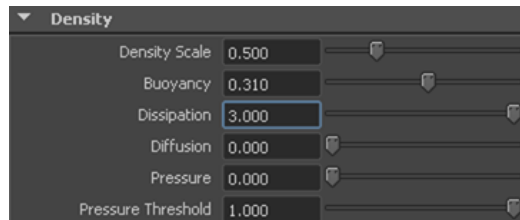
To set the fluidShape node attributes

- 1 In the **Attribute Editor**, click the fluid_slag_smokeShape tab.
- 2 In the **Contents Method** section, set **Temperature** to **Dynamic Grid**.
- 3 In the **Contents Details** section, click **Density**, and set the following:
 - **Buoyancy**: 0.31

This determines the tendency of the **Density** to rise in the container, similar to the way a lighter-than-air gas rises in the atmosphere. Decreasing **Buoyancy** will keep the fluid from rising and increasing in size.

■ **Dissipation:** 3.0

This determines how quickly **Density** is removed from the container similar to evaporation. Increasing **Dissipation** causes the **Density** to dissipate quicker into the air.



4 In the **Temperature** section, set the following:

■ **Temperature Scale:** 0.5

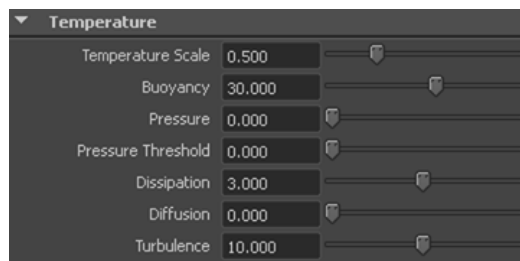
■ **Buoyancy:** 30.0

■ **Dissipation:** 3.0

■ **Diffusion:** 0.0

■ **Turbulence:** 10

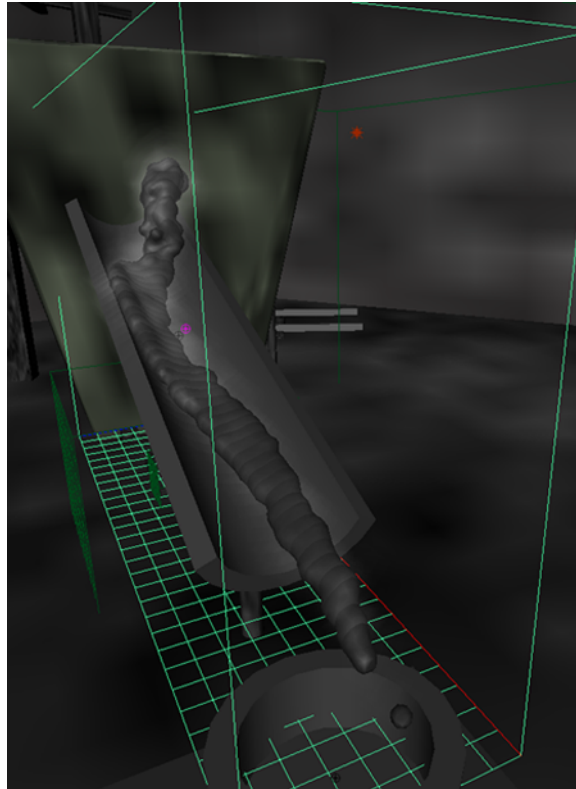
Setting these attributes adds movement to the **Heat** in the container, but they do not affect **Density**.



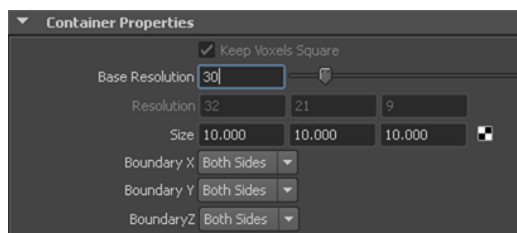
5 Rewind and play the simulation.

The fluid appears more subtle in the scene, but it is difficult to see how the **Density** and **Temperature** attribute adjustments have affected its behavior. Before setting the fluid **Shading** attributes to color the fluid, you can increase the fluid resolution to make the fluid's behavior easier

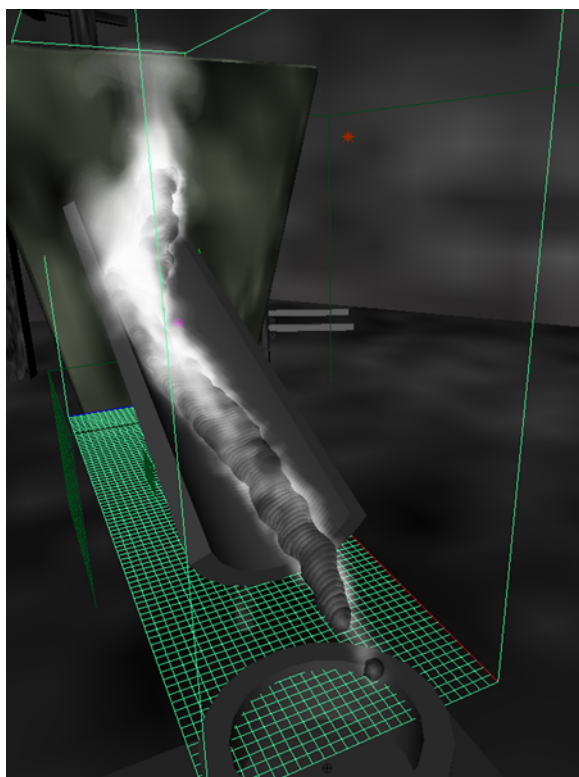
to see. Be aware that increasing fluid resolution increased simulation time.



6 In the **Container Properties** section, set the **Base Resolution** to 30.



7 Rewind and play the simulation.
The fluid behavior is now much easier to assess.



Setting fluid shading attributes

You can set the fluid shading properties using attribute ramps. In this section, you add color and incandescence to the **Density** and **Temperature** and adjust the fluid opacity.

To set fluid shading attributes

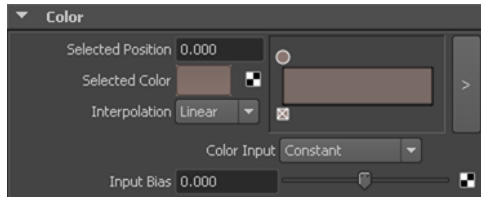
- 1 In the **Shading** section of the fluidShape **Attribute Editor**, click the **Transparency** color swatch, and set the color to the following values:
 - R: 0.5
 - G: 0.5
 - B: 0.5

With a low transparency value, you can clearly see the smoke being emitted from the slag. After you set the **Incandescence** ramp, you can re-adjust the transparency if necessary.

- 2 Set **Edge Dropoff** to 0.025.
- 3 In the **Color** section, click the color swatch and set the color to following values:
 - R: 0.546
 - G: 0.466
 - B: 0.443



- 4 Set **Color Input** to **Constant**.



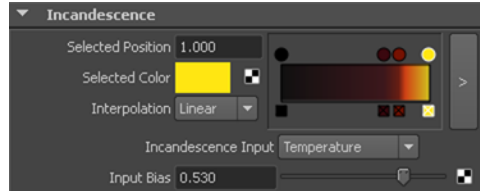
- 5 In the **Incandescence** section, create a ramp using the following values:

Marker	Selected Position	RGB color values
1	0.5	black
2	0.7	R: 0.23, G: 0.035, B: 0.06
3	0.8	R: 0.46, G: 0.071, B: 0.011
4	1.0	R: 3.0, G: 0.90, B:0.072

Leave all of the marker **Interpolation** values set to **Linear**.

- 6 Set **Incandescence Input** to **Temperature**.

- Set **Input Bias** to 0.53.



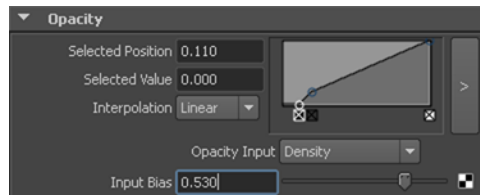
- In the **Opacity** section, create a ramp using the following values:

Marker	Selected Position	Selected Value
1	0.1	0
2	0.2	0.2
3	1.0	1.0

Leave all of the marker **Interpolation** values set to **Linear**.

- Set **Opacity Input** to **Density**.

- Set the **Input Bias** to 0.5.



- Rewind and play the simulation.

The fluid still does not look like real smoke and flames. This is because the fluid resolution is still too low. In the next section, you increase the resolution and improve the solve quality.

Increasing the quality of the fluid simulation

To create more smoke and add flames to the effect, increase the quality of the fluid simulation by increasing the Base Resolution value of the container. You

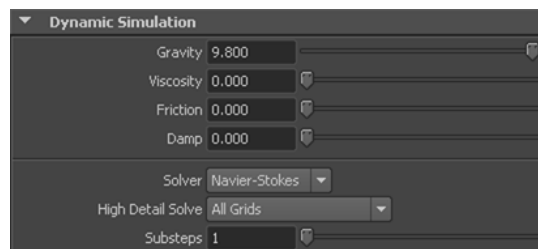
can also use **High Detail Solve** to add detail and use **Self Shadowing** to highlight this detail.

Be aware that making these adjustments to the fluid increases the simulation time.

To increase fluid simulation quality

- 1 In the **Container Properties** section, set the **Base Resolution** to 50.
For systems with low processing power, set the **Base Resolution** set to 30.
- 2 In the **Dynamic Simulation** section, set **High Detail Solve** to **All Grids**.

Using **High Detail Solve** gives simulations more detail without increasing resolution. It is ideal for creating effects such as explosions, rolling clouds, and billowing smoke.



- 3 In the **Lighting** section, do the following:
 - Turn on **Self Shadow**.
 - Set **Shadow Opacity** to 1.0.

This sets the fluid to cast internal shadows and determines the darkness of the shadows cast from the fluid. When **Shadow Opacity** is 1.0, shadows are completely black and the fluid is totally in shadow. The effect of these attributes is most noticeable when the fluid is rendered.

- 4 Rewind and play the simulation to about frame 80, then click the

Render the current frame  icon.

Render the frame again, but this time change the camera perspective.

- 5 In the **Render View** window, select **Render > Render > persp**.



You've now completed the lesson on using a fluid effect to add smoke and flames to the slag foundry. In the next lesson, you create two particle systems which simulate sparks flying off the molten slag as it slides down the chute.

Caching the fluid simulation

Before you render out the final simulation, you need to cache the fluid effect. If you plan to use the current scene for the remaining lessons, you can either cache the fluid now or before you start Lesson4.

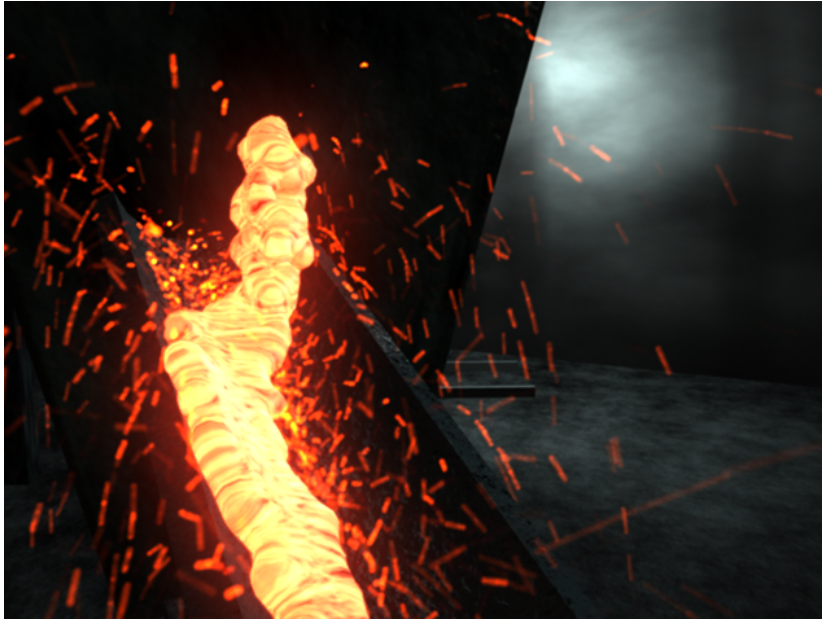
NOTE Depending on the processing speed of your system, caching the fluid can take up to 15 minutes.

To cache your fluid

- 1 Rewind the simulation to the start frame, select the fluid, and do the following:
 - From the **Dynamics** menu set, select **Fluid nCache > Create New Cache >** .
 - In the **Create Fluid Cache Options**, set the **Cache directory** to your project folder.
 - For **File distribution**, select **One file**.
 - Click **Create**.
- 2 After Maya caches your fluid simulation, play it back.

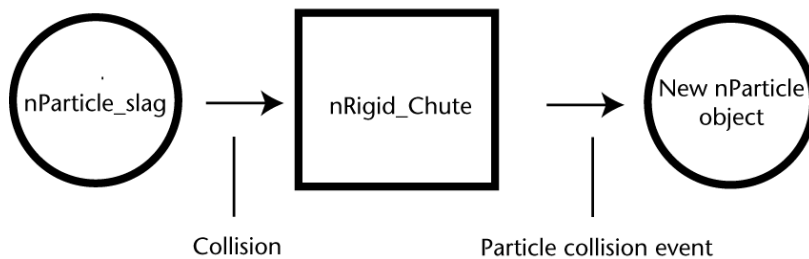
Lesson 3 Creating flying sparks

Overview

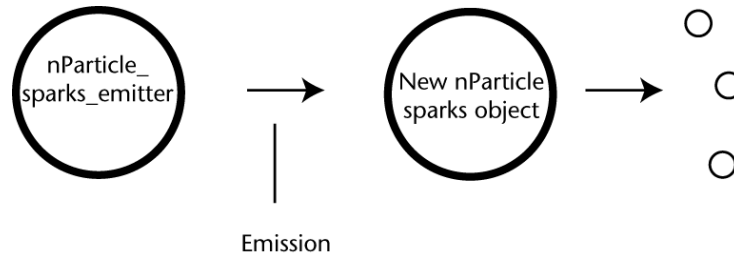


In this lesson, you create sparks that fly from the molten slag as it slides down the chute. To add realism to the emission and behavior of the sparks without using particle expressions, you set up a two-stage nParticle effect.

In the first stage of the sparks effect, the nParticle slag (*nParticle_slag*) collides with the nRigid_chute, and triggers a particle collision event. The particle collision event creates a new nParticle object called *nParticle_sparks_emitter*.



In the second stage, the `nParticle_sparks_emitter` object is used as a surface emitter source for a third particle system called `nParticle_sparks`. Particles from the `nParticle_sparks` object are the simulated sparks visible in the scene. The `nParticle_sparks_emitter` object does not display in the scene.



To ensure that sparks are emitted in locations and at times that appear random, and in the correct amount, you need to control the number of `nParticle_sparks_emitter` particles. To do this, you use the **Emission Overlap Pruning** attribute to emit particles at specific areas along the chute. The result is an effect of randomization without the use of a particle expression or scripting.

It is important that the `nParticle` systems do not collide with each other. `nParticle` collisions cause the mesh to break apart as well as create unwanted collision events, which generate too many sparks. To avoid this, you will use **Exclude Collide Pairs** between each system to prevent `nParticles` from colliding, while still enabling them to collide with the slag chute.

Setting up the simulation

Before starting the lesson, set up the simulation by doing the following:

- 1 Open `SlagPourFoundry_3.mb`.
- 2 Hide the `fluid_slag_smoke` object by selecting it in the **Outliner** and selecting **Display > Hide > Hide Selection**.
Hiding the fluid object speeds up the simulation while you add and adjust new components.
- 3 Turn off **Intermediate object** on the `nParticle_slag` object. To do this, in the **Outliner**, select `nParticle_slag`, and in the **Object Display** section of the `nParticle_slagShape` **Attribute Editor**, turn off **Intermediate Object**.

You need to select this object for the collision event and to add `nConstraints`.

- 4 If you cached the `nParticle_slag` object in the previous lesson, disable the cache by selecting the `nParticle_slag` in the **Outliner**, and selecting **nCache > Disable All Caches On Selected**.

Disabling the cache allows you to make changes on the `nParticle_slagShape` node, such as creating a collision event with this object.

Creating the nParticle collision event

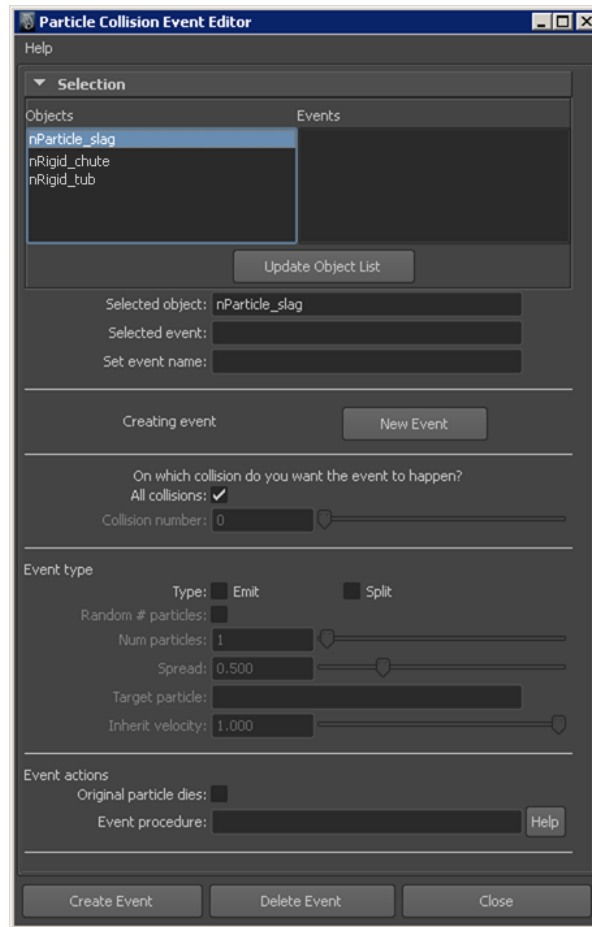
A collision event involves a source particle system (the object that executes an event when it collides with collision geometry) and a target particle system (the particle system that emits particles when the event is executed). For this collision event, the `nParticle_sparks` object is the source particle system, the `nRigid_chute` is the collision geometry, and the `nParticle_sparks_emitter` is the target particle system.

For more information about Particle Collision Events, see *Particle collision events* in the *Dynamics* section of the Maya Help.

To create the collision event

- 1 In the **Outliner**, select the `nParticle_slag` object, then select **nParticles > Particle Collision Event Editor**.

The **Particle Collision Event Editor** appears.



- 2 In the **Particle Collision Event Editor**, do the following:
 - Ensure that *nParticle_slag* is selected in the **Objects** panel.
 - For **Set event name**, type Sparks_Emission.
 - In the **Event Type** section, set **Type** to **Emit**. In this case, when the **Event Type** is **Emit**, the source particle object continues to live after the collision event. When set to **Split**, the object dies after the event.
 - Set **Num Particles** to 1.
This specifies that one particle is created for each collision event.
 - Set **Spread** to 1.

This sets the emission spread angle to 180 degrees.

- For the **Target Particles**, type nParticle_sparks_emitter.

This specifies that the collision event creates an nParticle object.

- Set **Inherit Velocity** to 0.

This sets new nParticles to not inherit velocity from the nParticle_slag object.

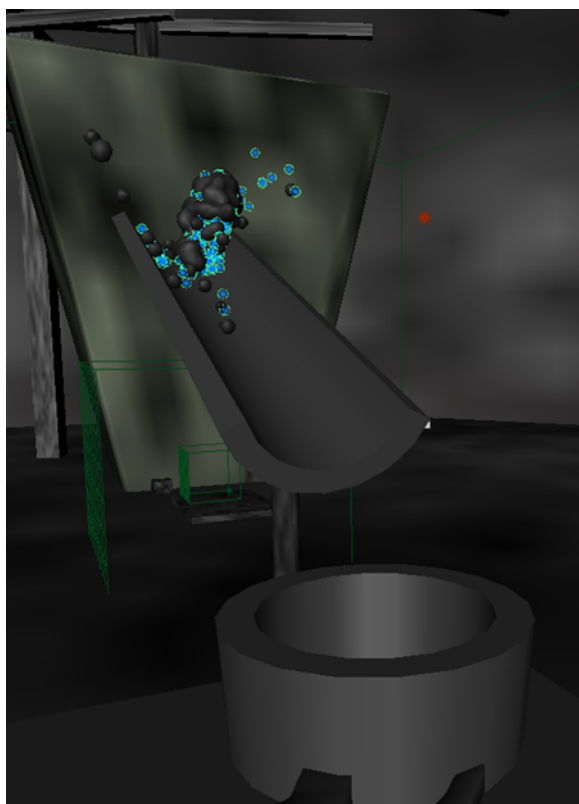
- Click **Create Event** and close the **Particle Collision Event Editor**.

The new nParticle object appears in the **Outliner**.

3 Rewind and play the simulation.

A new **Balls** style nParticle system is created (since **Balls** was the last nParticle style selected in lesson 1) when the nParticle_slag collides with the nRigid_chute object. Notice the nParticles are colliding with the slag and breaking it apart. To stop the collisions between the nParticle_sparks_emitter and nParticle_slag, you will create an **Exclude Collide Pairs** constraint.

NOTE If you do not see the new nParticle system emitting into the scene, ensure that you have disabled the nParticle_slag cache, and create the collision event again.



NOTE Be aware that adding constraints increases simulation time.

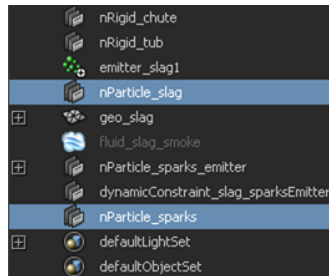
Creating the nConstraint

For this simulation, you want both nParticle systems (nParticle_slag and nParticle_sparks_emitter) to interact with the chute, but you do not want the two nParticle systems to collide. Since both nParticle systems need to interact with the chute, you cannot disable collisions on either nParticle object or assign each system its own Nucleus solver. Creating an **Exclude Collide Pairs** constraint is the best solution for this simulation, because it lets you disable collisions between objects or object components.

To create an Exclude Collide Pair constraint

- 1 In the **Outliner**, Ctrl-select the *nParticle_sparks_emitter* and *nParticle_slag* objects, then select **nConstraints > Exclude Collide Pairs**.

A *dynamicConstraint1* object appears in the **Outliner**.



- 2 Rename the new constraint to *dynConstraint_slag_sparksEmitter*.
- 3 Rewind and play the simulation.
The two nParticle systems no longer collide with each other. Leave the dynamicConstraintShape node attributes set to the default values for this simulation.
- 4 Select the *nParticle_slag* object, and in the **Attribute Editor**, turn on **Intermediate Object** to hide it from the scene.

Hiding the nParticle slag makes it easier to see how the nParticles sparks work in the scene.

In the next section, you refine the behavior for the *nParticle_sparks_emitter* object and complete the second stage of the sparks effect.

Setting up the second stage of the sparks effect

For a realistic simulation, the sparks need to appear randomly, not in a single steady stream. At the same time, the sparks need to appear to be produced by the collision between molten slag and the cooler slag chute. Creating too many sparks can easily make the simulation look too staged and unrealistic.

To create these conditions, you might use particle expressions, but in this lesson you will use only nParticle attributes. This means that only a few particles are required and they must appear stationary at various locations

along the chute. Also, they must not self-collide nor be visible in the simulation, but still participate in the effect.

- 1 In the **Outliner**, select the nParticle_sparks_emitter and in the **Attribute Editor**, click the nParticle_sparks_emitterShape tab.

- 2 In the **Lifespan** section, set the following:

- **LifeSpan Mode: Random range**

- **Lifespan: 0.1**

- **Lifespan Random: 1.0**

Since this nParticle object needs to be in the scene just long enough to emit a burst of sparks, you give it a short Lifespan.

- 3 In the **Collision** section, turn off **Self Collide**.

Although the individual nParticles in this system will not be close enough to collide, turning off **Self Collide** saves the Nucleus solver from calculating these collision iterations.

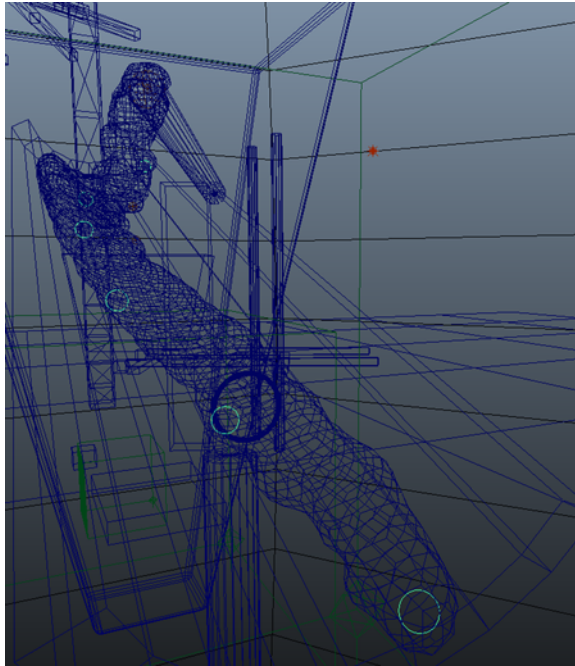
- 4 In the **Emission Attributes** section, set **Emission Overlap Pruning** to 10.

This attribute keeps nParticles from self-colliding while being emitted. A value of 1 guarantees that no self collisions occur with other particles on emission. This value ensures that only a few, evenly spaced particles are emitted from the collision event.

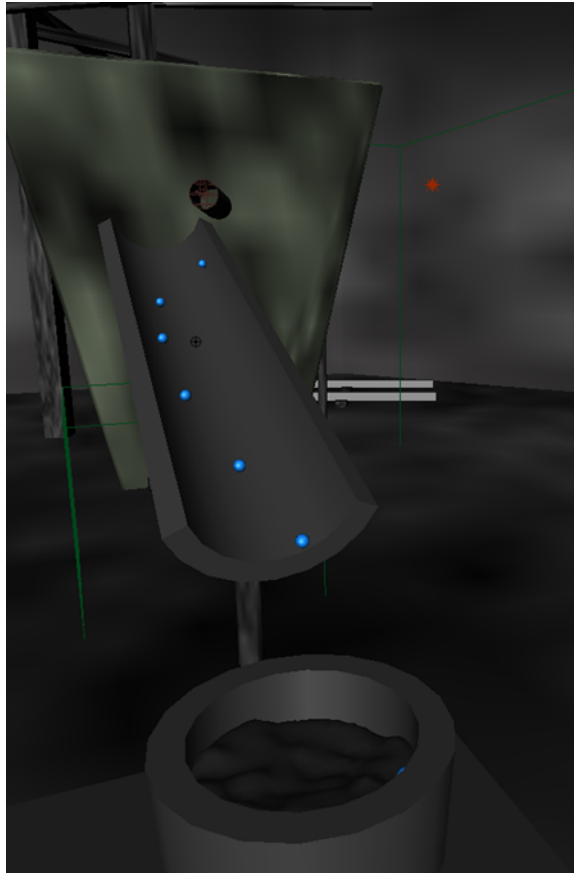
- 5 Rewind and play the simulation.

A good number of particles are created on the chute, spread out evenly along its length. Notice however that the particles move down the chute with the slag. Since these nParticles will emit other nParticles, they need to remain fixed to the chute.

NOTE To make it easier to see where on the chute the new nParticles are, and how many are being emitted into the scene, set the scene shading to Wireframe.



- 6 In the **Dynamic Properties** section, turn on **Ignore Solver Gravity**.
- 7 Rewind and play the simulation.
With gravity disabled on the object, the nParticles remain fixed at various locations along the slag chute. These locations will emit the nParticle sparks.



- 8 To ensure that these particles are not visible in the scene, in the **Shading** section, set **Opacity** to 0.
Next, you will create and adjust nParticles to simulate the flying sparks.

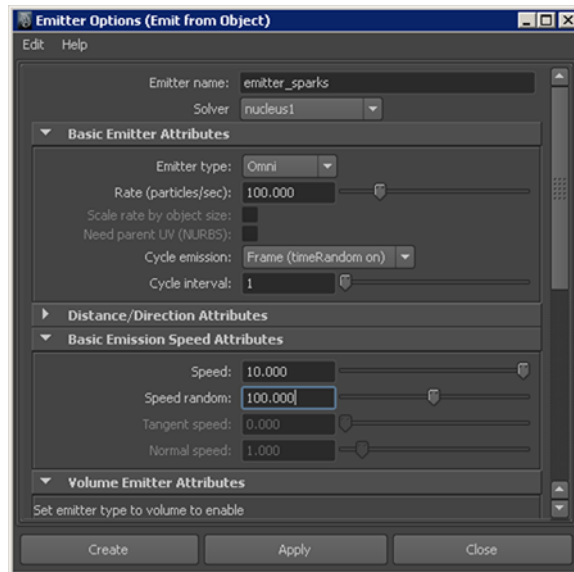
Creating the nParticle sparks

Creating the nParticle sparks is the final stage to the two-part particle effect. The nParticle sparks are emitted from the nParticle_sparks_emitter object at various locations along the chute using **Emit From Object**. An external gravity field is added to give the sparks an arcing motion as they fly up and fall to the foundry floor.

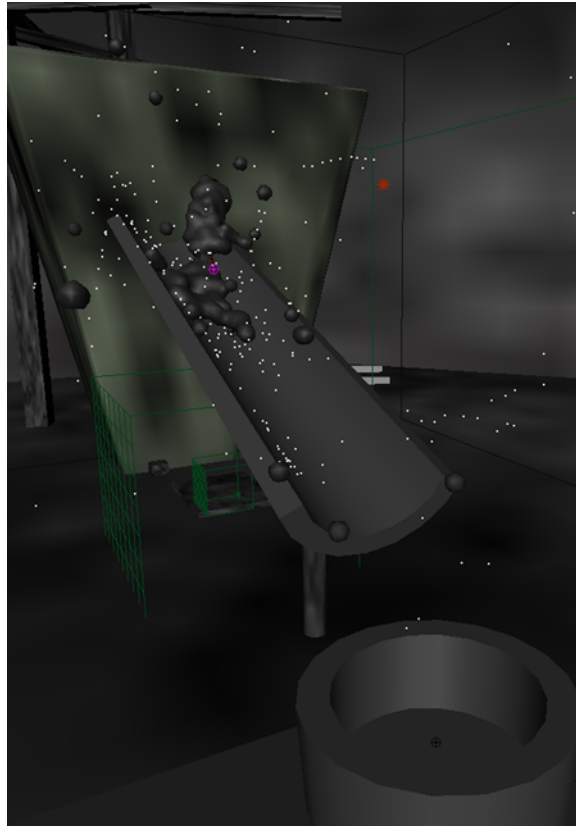
You also add two more **Exclude Collide Constraints** so that the sparks do not collide with the slag or the nParticle_sparks_emitter objects.

To emit the nParticle sparks

- 1 In the **Outliner**, select the *nParticle_sparks_emitter* object and select **nParticles > Create nParticles > Points**.
- 2 Select **nParticles > Create nParticles > Emit from Object >** .
- 3 In the **Emitter Options (Emit from Object)** window, select **Edit > Reset Settings** and do the following:
 - For the Emitter name, type *emitter_sparks*.
 - If the **Solver** list appears, ensure that it is set to *nucleus1*.
- 4 In the **Basic Emitter Attributes** section, set the following:
 - **Rate (particle/sec):** 100
 - **Cycle emission: Frame (timeRandom on)**
When **Cycle Emission** is set to **Frame**, the emission sequence is restarted after the number of frames specified by **Cycle interval**. For this effect, **Cycle interval** is left at 1, so the emitted sparks cycle restarts each frame.
- 5 In the **Basic Emission Speed Attributes** section, set the following:
 - **Speed:** 10.0
 - **Speed random:** 100.0



- 6 Click **Create**.
- 7 In the **Outliner**, rename the new nParticle object to *nParticle_sparks*.
- 8 Rewind and play the simulation.
Notice again that the new nParticle system collides with the slag, breaking the mesh apart. To solve this, you add two more **Exclude Collide Pairs** constraints.



- 9 In the `nParticle_slag` object **Attribute Editor**, turn off **Intermediate Object**.

You can now select the `nParticle` object for the `nConstraint`.

- 10 In the **Outliner**, Ctrl-select the `nParticle_slag` and `nParticle_sparks` objects, then select **nConstraints > Exclude Collide Pairs**.

Name this constraint `dynConstraint_slag_sparks`.

- 11 Add another **Exclude Collide Pairs** constraint between the `nParticle_sparks_emitter` and `nParticle_sparks` objects. Name this constraint `dynConstraint_sparksEmitter_sparks`.

- 12 Rewind and play the simulation.

The particle systems no longer collide, but the `nParticle_sparks` need adjustments to make them look and behave like red-hot sparks.

Setting the nParticle Sparks attributes

To make the particles behave more like sparks, they need to have a limited lifespan, and they cannot self-collide. To simulate the streaking effect of fast moving sparks, you change the particle render type to **Tube (s/w)**. Note that this streaking effect is only visible after rendering a frame.

- 1 In the **Outliner**, select the *nParticle_sparks* object, then click the *nParticle_sparksShape* tab in the Attribute Editor.
- 2 In the **Lifespan** section, set the following:
 - **Lifespan Mode: Random range**
 - **Lifespan Random: 6**

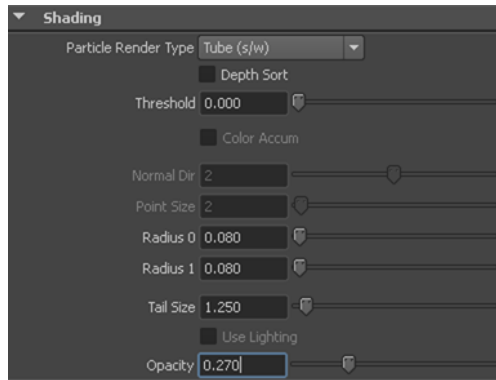
Increasing **Lifespan Random** makes the particle lifespan more random, like real sparks. Randomizing the spark lifespan decreases the number of sparks that are visible in the scene at any one time, making the spark effect less dense. To compensate for the random lifespans, you will increase the nParticle emission rate later.
- 3 In the **Dynamic Properties** section, turn on **Ignore Solver Gravity**. Later in the tutorial, you add an external **Gravity** field to control the behavior of the sparks. If the Nucleus gravity is also affecting the behavior of the sparks, the external **Gravity** field will not have the desired effect.
- 4 In the **Shading** section, set the following:
 - **Particle Render Type: Tube (s/w)**

This type works well for simulating sparks, since you can control streaking effects using the **Tail Size** and **Opacity** attributes.
 - **Threshold: 0**

This makes the sparks display individually with no surface blending.
 - **Radius0: 0.08**
 - **Radius1: 0.08**

This sets the radius for the tube at its starting point and at its ending point. For this effect, the sparks have the same radius at their starting and end point.
 - **Tail Size: 1.25**


This scales the length of the tail, controlling the length of the streaking effect.
 - **Opacity: 0.27**

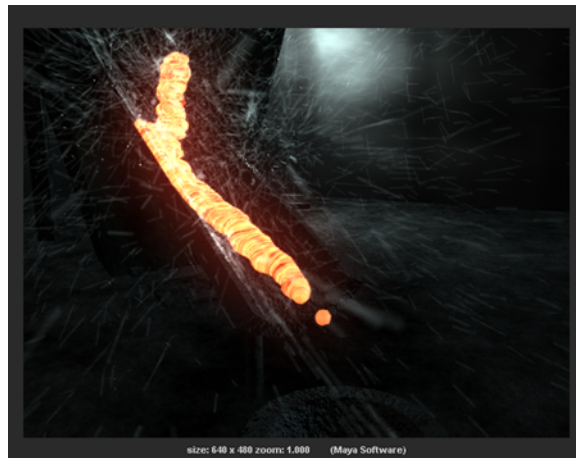


For more information about **Tube (s/w)** type particles, see *Particle nodes* in the *Dynamics* guide of the Maya Help.

- 5 To increase the number of flying sparks, in the **Attribute Editor**, click the emitter_sparks tab, and in the **Basic Emitter Attributes**, set **Rate (Particles/Sec)** to 2000.
- 6 Rewind and play the simulation to about frame 65.
This render lets you view the general appearance of sparks, such as the amount of streaking density.

NOTE For the purposes of this tutorial, use the **Maya Software** renderer with default settings. See [Rendering a single frame](#) (page 15).

- 7 Click the  **Render Current Frame** icon.
The particles appear more like sparks, but they need to be properly shaded to appear red-hot.



Notice also that some sparks are colliding with the slag bin.

- 8 In the **Outliner**, select *geo_bin*, then select **nMesh > Create Passive Collider**.

This converts the *geo_slag* mesh to passive collision object to ensure that these sparks collide with the bin rather than traveling through it and out of the scene.

- 9 Rename *nRigid1* to *nRigid_bin*.

Setting the nParticle sparks shading attributes

To shade the sparks, you set **Color** and **Incandescence** ramps to color the particles according to their speed and age, and create an **Opacity Scale** ramp that sets the nParticles opacity based on its speed. The opacity values help generate more particle streaking.

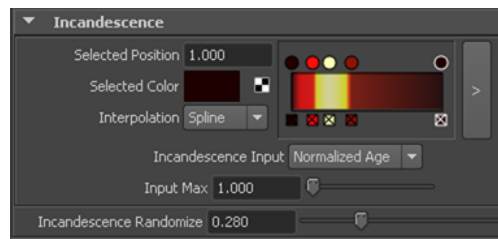
- 1 In the **Attribute Editor**, click the *nParticle_sparksShape* tab.
- 2 Under **Shading**, in the **Color** section, do the following:
 - Click the **Selected Color** swatch and in the **Color Chooser**, set the color to black.
 - Set the **Color Input** to **Speed**.
 - Set the **Input Max** to 9.7.

3 In the **Incandescence** section, set a ramp using the following values:

Marker	Selected Position	RGB color values	Interpolation
1	0	R:0.146, G:0, B: 0	Linear
2	0.14	R:2.25, G: 0.053, B: 0.053	Smooth
3	0.25	R: 39.22, G: 7.77, B: 0.71	Smooth
4	0.4	R: 0.552, G: 0.079, B:0.026	Linear
5	1.0	R: 0.128, G: 0, B:0	Spline

4 Set **Incandescence Input** to **Normalized Age**.

5 Set **Incandescence Randomize** to 0.28.



6 Rewind and play the simulation to about frame 65.

7 Render the current frame.

The **Color** and **Incandescence Scale** ramps color the nParticles so they look red-hot. However, there is not enough streaking to give the sparks the appearance of cooling off as they age.



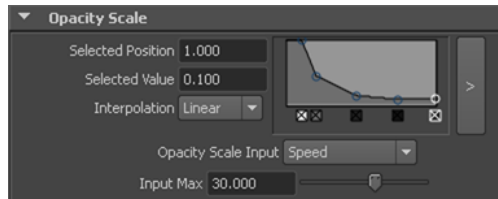
You can use an **Opacity Scale** ramp and an opacity value for each spark based on its speed. For example, as the sparks begin to slow down, their **Opacity** values decrease, until they eventually fade out.

- 8** In the **Opacity Scale** section, set a ramp using the following values:

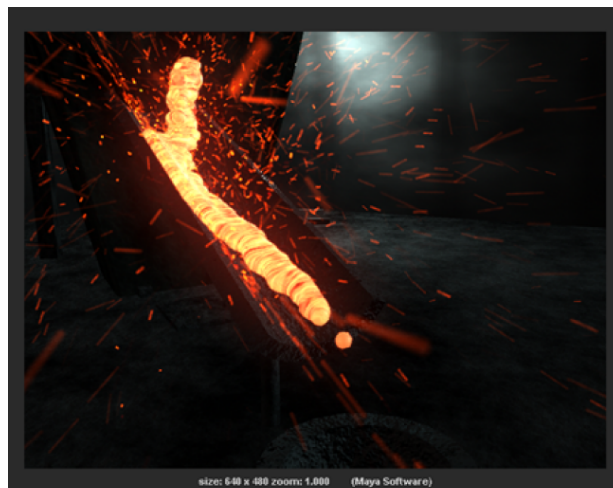
Marker	Selected Position	Selected Value
1	0.1	1.0
2	0.2	0.44
3	0.47	0.14
4	0.75	0.08
5	1.0	0.10

Leave the Interpolation set to **Linear** for each marker.

- 9** Set the **Opacity Scale Input** to **Speed**.
10 Set the **Input Max** to 30.0.

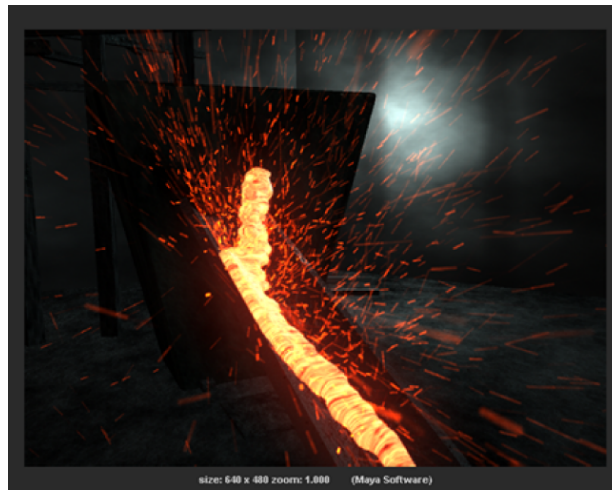


- 11 Rewind and play the simulation to about frame 65.
- 12 Render the current frame.
Each spark now fades out as it loses momentum, creating noticeable streaks.



Rate (Particles/Sec): 2000

Notice that the per-particle **Opacity** values also make the sparks less dense. To increase the density of the spark, you can increase the emitter's **Rate (Particles/Sec)** attribute to 4800.



Rate (Particles/Sec): 4800

In the next section, you make the sparks behave more realistically by increasing bounce and applying an external gravity field to make them travel in an arcing motion.

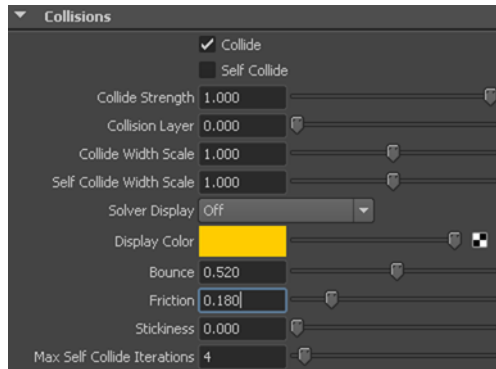
Controlling the nParticle spark behavior

For a spark effect, it's important to generate the arcing motion of typical hot sparks. Real sparks are a very small amount of red-hot matter which allows them to travel quickly through air. As sparks cool they slow down and eventually begin to fall. This creates the characteristic arcing motion.

To create this motion, you decrease the mass of the sparks and increase the amount of drag acting on them. You then add an external **Gravity** field to pull the sparks to the foundry floor.

- 1 In the **Attribute Editor**, click the nParticle_sparksShape tab.
- 2 In the **Collisions** section, set the following:
 - **Bounce:** 0.52
 - **Friction:** 0.18

Increasing the **Bounce** and **Friction** makes the sparks behave more realistically when they collide with the foundry floor, slag chute, and bin.



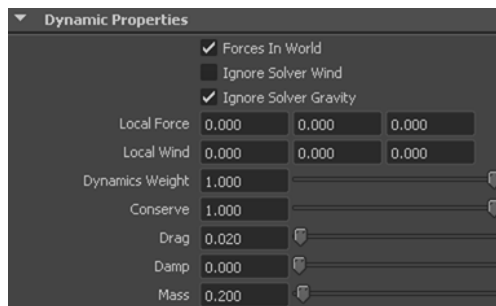
3 In the **Dynamic Properties** section, set the following:

- **Drag:** 0.02

Increasing **Drag** adds air resistance to the particles as they fly, slowing them down a little. This will allow the external **Gravity** field, which you add later, to have more of an effect on the sparks.

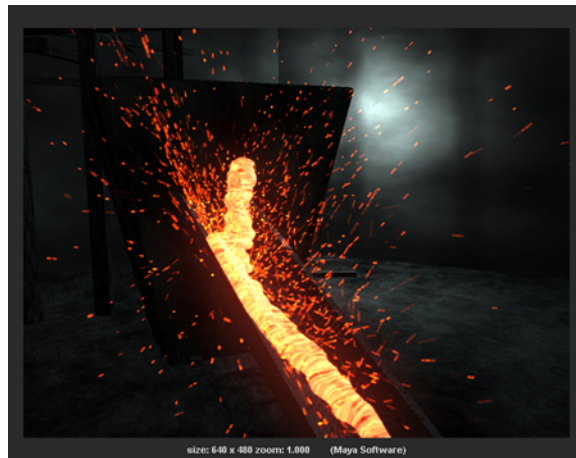
- **Mass:** 0.2

Decreasing the **Mass** lets the sparks fly higher, generating a more pronounced arc effect when external **Gravity** pulls them down.



4 Rewind and play the simulation.


The sparks are flying a little higher and less chaotically than before.



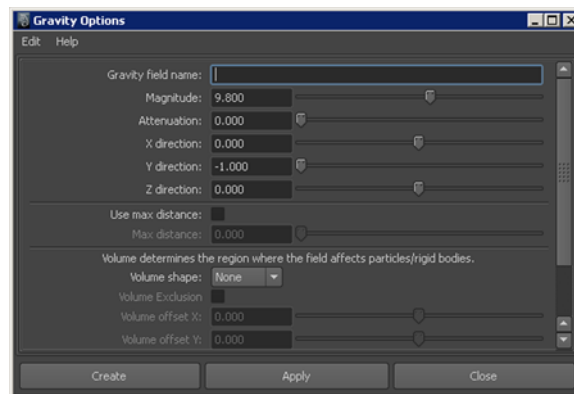
Adding an external Gravity field

By default, the nucleus node applies forces such as wind and gravity to nParticle. You can turn these forces off, and instead apply Maya fields to nParticles. When creating an nDynamics simulation, external fields are useful for controlling individual Nucleus objects.

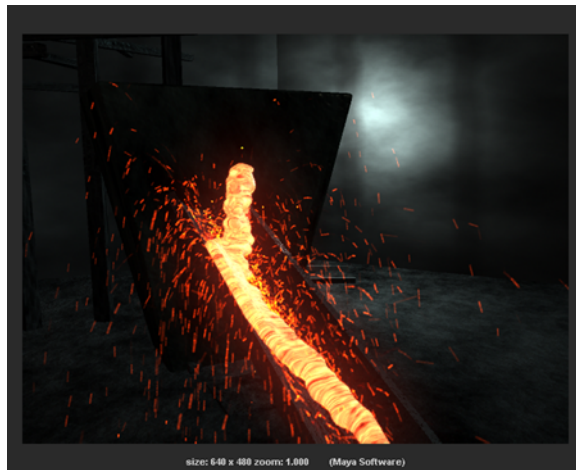
To create an external gravity field

- 1 In the **Outliner**, select the *nParticle_sparks* object, then select **Fields > Gravity** > .

The **Gravity Options** window appears.

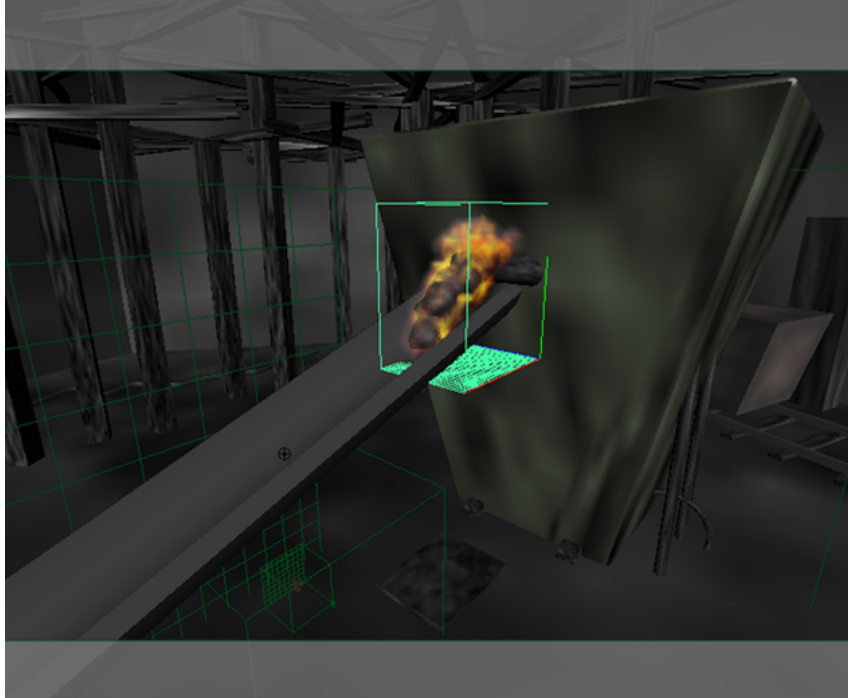


- 2 In the **Gravity Options** window, select **Edit > Reset Settings** and do the following:
 - For **Gravity field name**, type gravity_sparks.
 - Set **Magnitude** to 31.4
 - Click **Create**.
- 3 Rewind and play the simulation to about frame 65.
- 4 Render the current frame.
The sparks now travel in an arcing motion as they fall to the foundry floor.



The nParticle sparks effect is now complete.

Beyond the tutorial



To see the final slag foundry simulation, you can render the completed scene. To watch a video of the rendered simulation, open *SlagPourFoundryFinal.mp4v*, included with the tutorial lesson files.

If you want to render the scene you completed in Lesson 3, change the scene view perspective so it will render with an animated camera. To do this, select **Panels > Perspective > persp** in the **Panel** menu. As an alternative, you can open the *SlagPourFoundry_4.mb* file, which has the perspective set for rendering.

NOTE Before rendering the scene, cache each object separately, then batch render the scene using your favorite renderer. Depending on the processing speed of your system, rendering the final simulation can take over an hour.

Caching the final simulation

Before you render the final simulation, you need to cache each effect.

Cache each nParticle object individually by doing the following:

- 1 Select the nParticle object you want to cache.
- 2 From the **nDynamics** menu set, select **nCache > Create New Cache >** .

Cache the fluid simulation by doing the following:

- 1 Select the fluid object you want to cache.
- 2 Switch to the **Dynamics** menu set, then **Fluid nCache > Create New Cache >** .

You can now batch render the scene using your favorite renderer.