

Blender 4.5 の Fire Simulation 基本

Francesco Milanese

このPDFは無料で、自由に配布できます。

内容を気に入っていただけたら、私の[YouTube](#)チャンネルを登録して、動画に「いいね」をしていただけると嬉しいです。ありがとうございます！

1. [基礎](#)
2. [Principled Volume, Blackbody](#)
3. [Cache, Baking, Offset](#)
4. [Geometry, Inflow, Outflow](#)
5. [Domain Settings, 1/2 \(Gas, Fire, Collections\)](#)
6. [Domain Settings, 2/2 \(Weights, Data, Render, Display\)](#)
7. [Emission, Attributes, Volume Info](#)
8. [Effectors \(Guides, Colliders\)](#)
9. [Force Fields \(Turbulence, Wind, Curve Guide\)](#)
10. [Motion Blur, Filmic, Compositing, Eevee](#)

Fire Simulations in Blender 4.5 basics | 1/10: 基礎

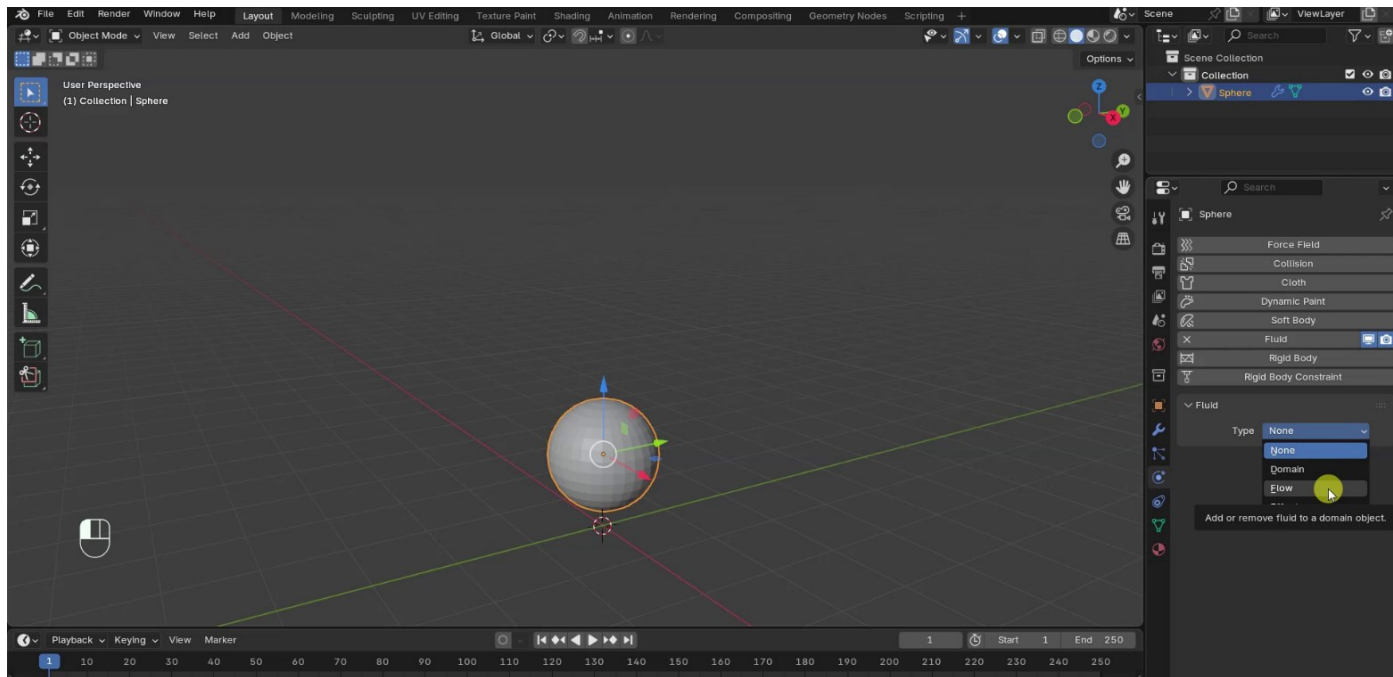
<https://youtu.be/PcqKmidV6Xc>

皆さん、こんにちは！このシリーズの最初のチュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の基本について解説します。ここでは、Blender で simulation を作成するために必要な2つの基本要素を見ていきます。

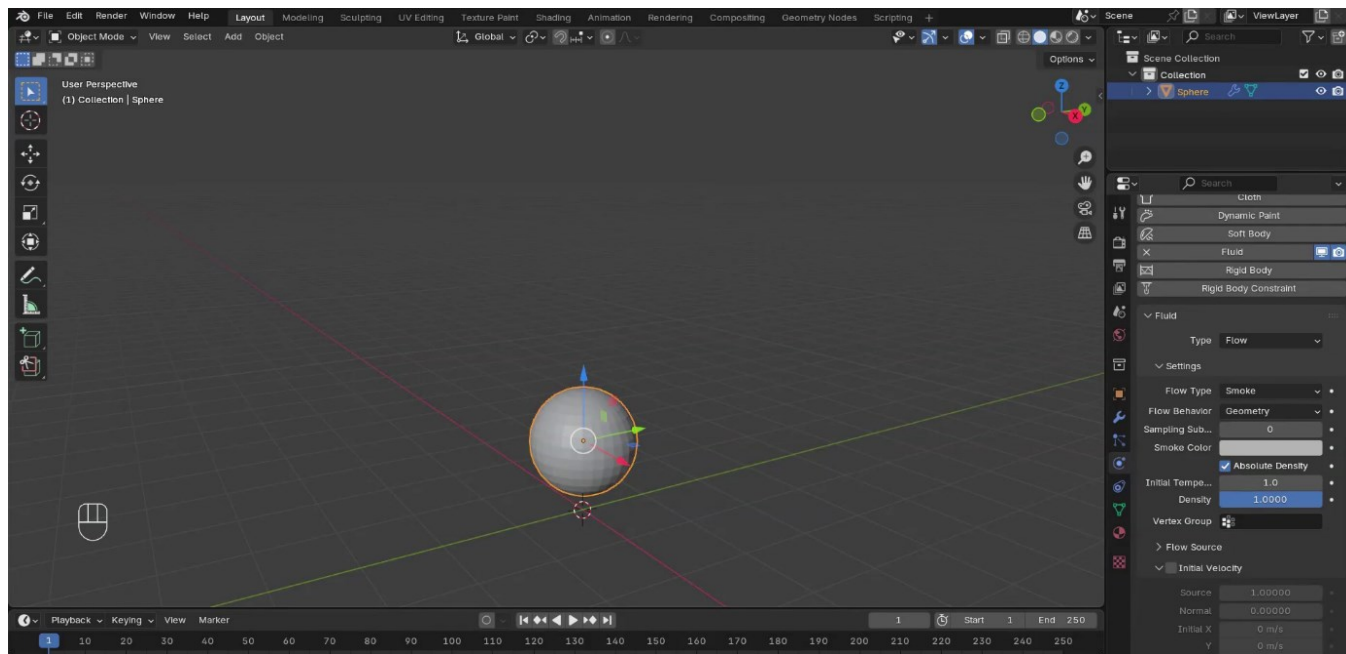
まずは空の Blender シーンから始めましょう。シーンに fire や smoke を追加するには、まずそれらを発生させるオブジェクトを配置する必要があると思うかもしれません。それは正しいのですが、実際には simulation を実行するために必要な基本要素の半分にすぎません。それでも、まずは emitter となるオブジェクトをシーンに追加してみましょう。

Blender の fire と smoke の simulation ツールは、各オブジェクトの Physics タブにあります。追加する必要があるコンポーネントのタイプは Fluid です。Fluid は liquid だけでなく、fire や smoke の simulation も扱います。最初に表示されるのは Type というメニューだけで、まだ fluid や他のカテゴリーを区別しているわけではなく、simulation の中での役割を列挙している形になっています。

各タイプのツールチップを見れば、その役割のヒントが分かります。特に Flow タイプは、シーンに要素を追加または削除するために使われます。先ほど作成した最初のオブジェクトは fire と smoke の emitter にしたいので、この Flow タイプを選びます。



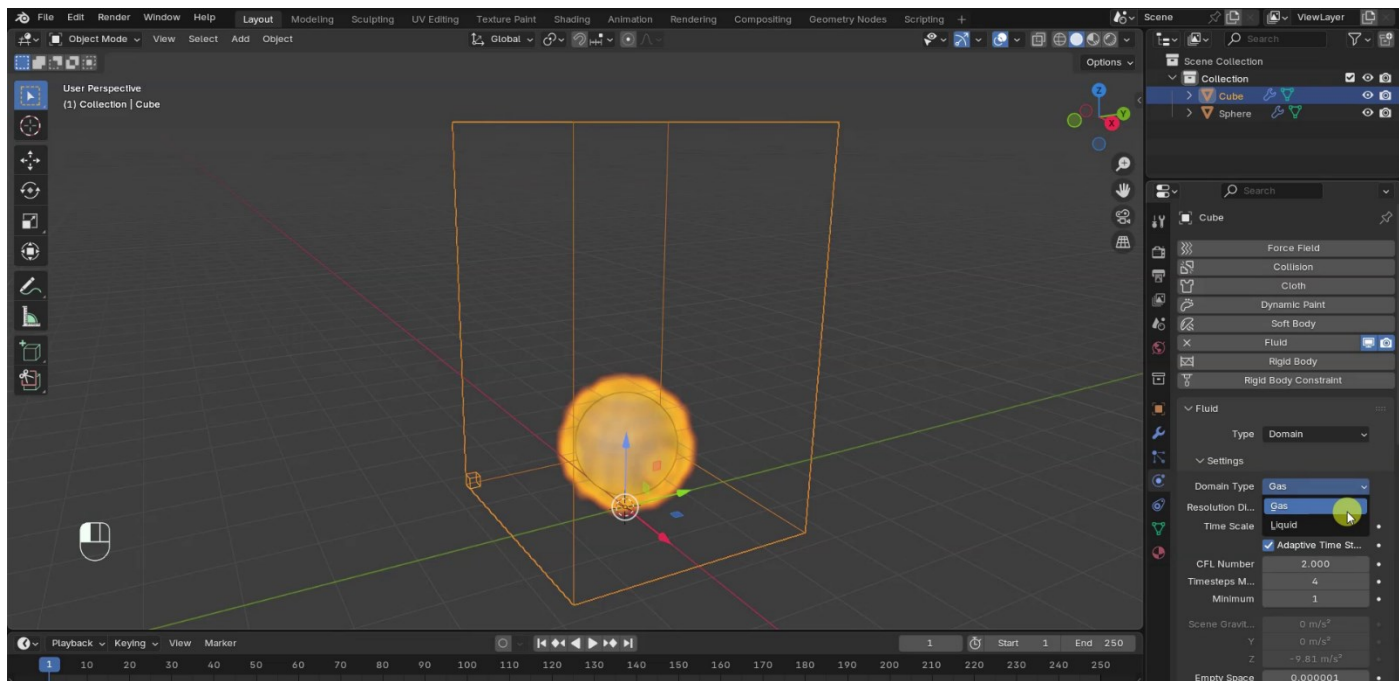
コンポーネントのパネルに新しいオプションがいくつか表示されます。最初の項目はFlow Type で、デフォルトでは Smoke に設定されています。メニューを開くと、Liquid には1つしかオプションがないのに対し、fire と smoke には3つの選択肢があります。それぞれを個別に扱うか、両方を組み合わせるかです。Fire And Smoke を選ぶと、両方のためのパラメータと設定が表示されます。そこで、このチュートリアルシリーズの最初の部分では、このオプションを選びます。



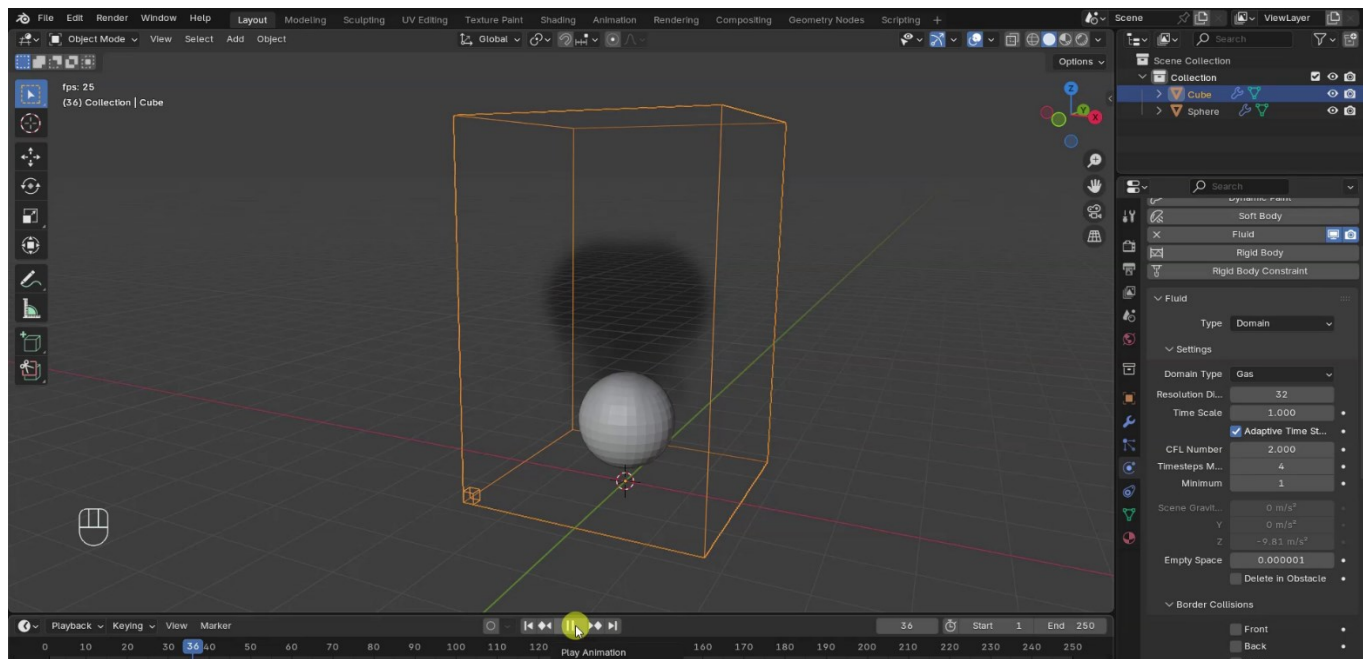
もし Blender の Cloth や Collision などの他の physics simulation に慣れている場合、この段階で Timeline の最初のフレームに移動し、Play ボタンを押せば fire と smoke の simulation が始まると期待するかもしれませんが。しかし実際には何も起こりません。そこで Timeline の最初のフレームに戻り、Fluid simulation に必要なもう1つの基本要素、Domain を導入しましょう。

すべての Fluid simulation は、メッシュによって定義された 3D 空間、つまり volume 内で行われます。Blender ではこの volume を simulation の Domain と呼びます。これは主にパフォーマンスのために必要です。というのも、こうした simulation には大量のメモリと計算が必要だからです。Domain はどのような形状のオブジェクトでも構いませんが、最終的に Blender はその Bounding Box を使用します。このため、最も分かりやすい選択肢は Cube を適切にスケールして使うことです。

emitter オブジェクトを Domain として使う Cube 内に入れたら、その Cube を選択し、Fluid コンポーネントを追加して Type を Domain、Subtype を Gas に設定します。すると Cube はすぐに透明になり、その構造が見えるようになります。また、下の角のひとつに小さな Cube が現れるのにも気づくでしょう。これについてはすぐに説明します。しかし最も重要な変化は、Domain 内の Inflow オブジェクトが炎に包まれることです！



Timeline の最初のフレームを現在のフレームに設定し、Play をクリックして simulation を開始します。確かに何かが起こりますが、最初の fire と smoke の simulation としては少々がっかりかもしれません。なぜなら、fire は最初の数フレームしか表示されず、その後は smoke だけが残るからです。



この挙動の原因は emitter オブジェクトの設定のひとつにあります。対象となる Fluid コンポーネントのパラメータは Flow Behavior で、デフォルトでは Geometry に設定されています。ツールチップによると、Geometry は現在のジオメトリを使って fire を生成し、Inflow モードは fire をシーンに継続的に追加します。そこでモードを Inflow に切り替え、Timeline の最初のフレームに戻ってもう一度 Play をクリックしましょう。

Inflow における「adds fire」という表現こそが重要です。なぜなら Inflow オブジェクトは各フレームで新しい simulation 要素をジオメトリから継続的に生成するからです。

一方で Geometry モードは最初のフレームでジオメトリを参照し、それをすぐに「消費」してしまいます。

言い換えると、Geometry モードは小さな紙切れを燃やす、あるいはスモークマシンが一度だけ煙を吐き出すようなものです。それに対して Inflow はガスコンロのバーナーや、煙を継続的に出し続けるスモークマシンのようなものです。

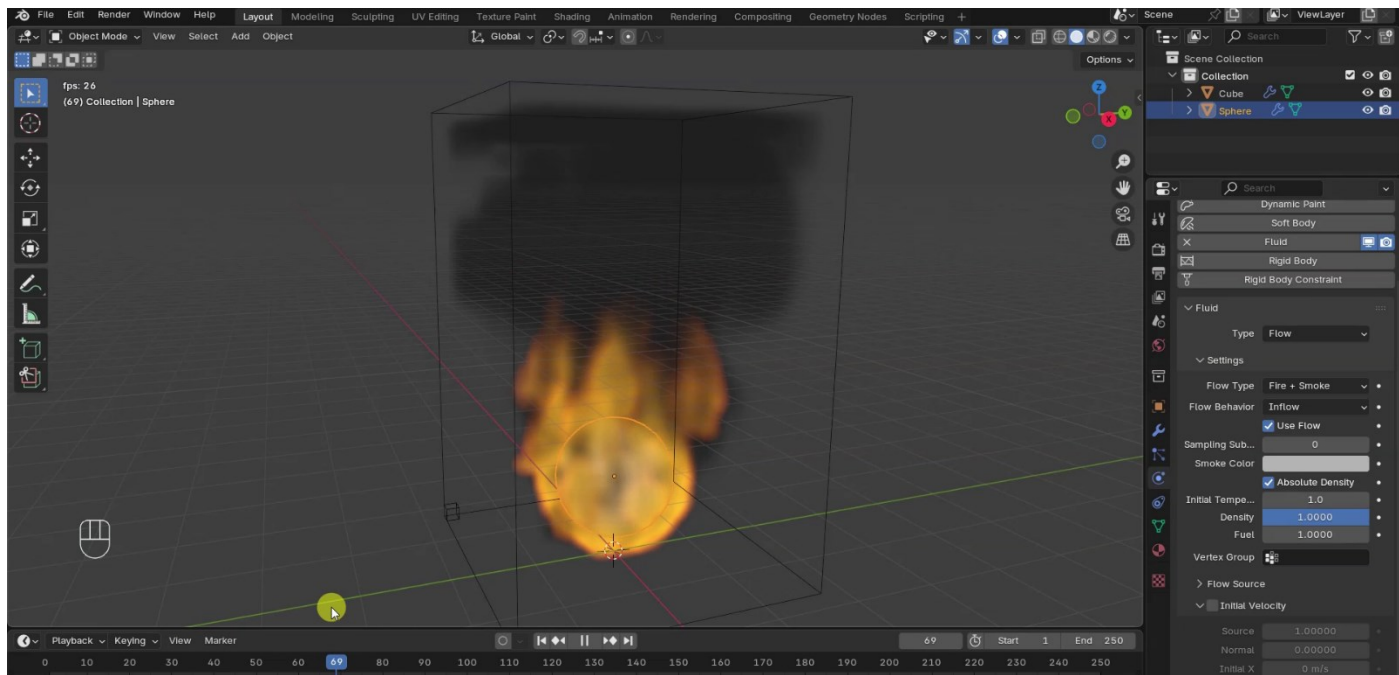
炎、とくにオブジェクトの下部が少しブロック状、つまり3D なので立方体的に見えることに気づいたかもしれません。

この見え方の原因は、Domain のパラメータのひとつである Resolution にあります。

Resolution は Domain の volume が小さな Cube にどのように分割されるかを決定します。

この解像度は、Domain の下の角に現れる小さな Cube で視覚的に表されます。

特に Resolution 値が低いと、炎や煙がそのサイズの Cube に分割されているのがはっきり分かります。

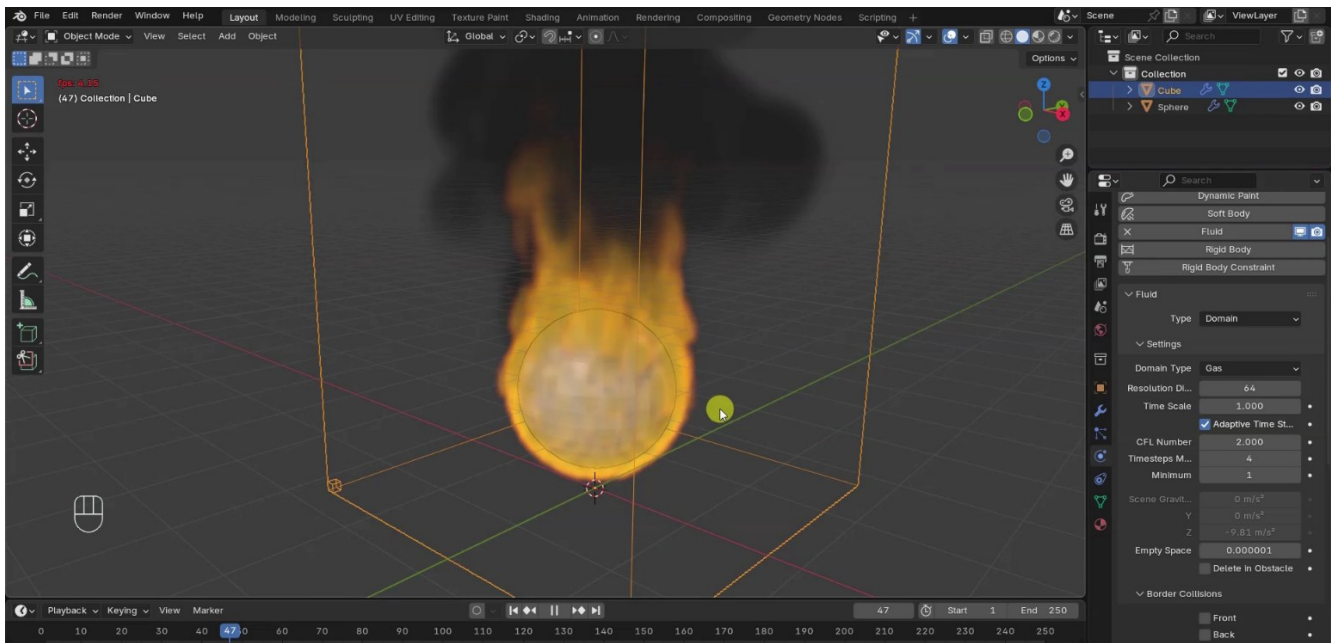


Fluid simulation は Domain の volume を Voxel と呼ばれる小さな Cube に分割して処理します。Voxel は simulation を計算するための最小単位であり、2D 画像における pixel の3D 版です。pixel と同じように、同じ volume に多くの Voxel があれば品質は向上しますが、その分計算コストとディスクメモリが増えます。最も良い方法は、まずデフォルトの Resolution 値 32 から始め、fire と smoke の形状が望むものになるまで

他のパラメータを調整することです。その後 Resolution をさらに上げてテストし、最終的な結果のために非常に高い値に設定します。

メモリ使用に関連して、Fluid simulation のデータがどこに保存されるのかを見てみましょう。特に、まだプロジェクトファイルを保存していない状態です。この情報は Domain オブジェクトの Cache セクションにあり、そこに表示されたパスを見ると、Blender がディスク上に simulation ファイルを保存するフォルダーを作成していることが分かります。現在のプロジェクトファイルを保存していないので、このフォルダーは一時的な場所に作られます。つまり、私のようにしてはいけません。Domain や他の Fluid simulation 要素を作成する前に、必ず Blender のプロジェクトファイルを保存してください。そうすれば cache フォルダーはプロジェクトファイルと同じディレクトリに作成され、ディスクの中を探し回る必要がなくなります。

この時点で viewport shading モードを切り替えたり、Timeline の途中のフレームでレンダリングを実行しても、fire も smoke も表示されません。これは、現在 Solid モードの 3D Viewport に見えているのは fire と smoke の Voxel 表示にすぎず、レンダリング用の Material がまだ定義されていないからです。この内容については次のエピソードで扱います！



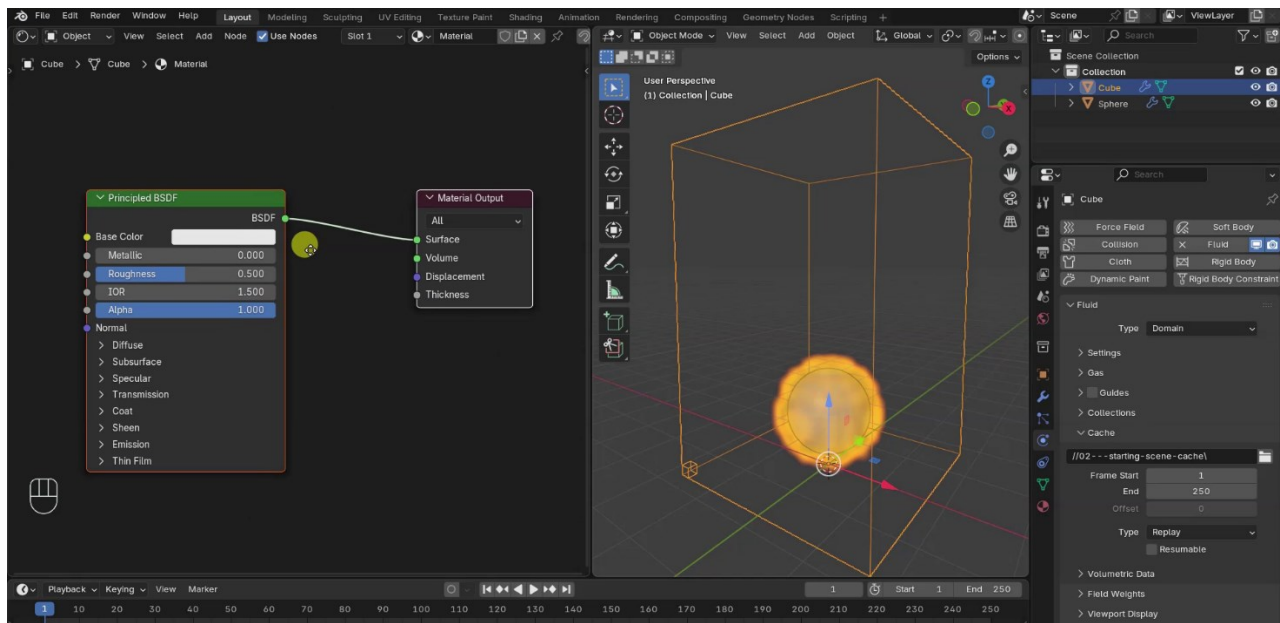
Fire Simulations in Blender 4.5 basics | 2/10: Principled Volume, Blackbody

<https://youtu.be/GSzj7fMeAyc>

皆さん、こんにちは！このシリーズの第2回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の基本として、Cycles でレンダリングするための基本的な Material の設定方法を見ていきます。具体的には、Principled Volume Shader と、この Material が提供する2つのモード、Emission と Blackbody を解説します。

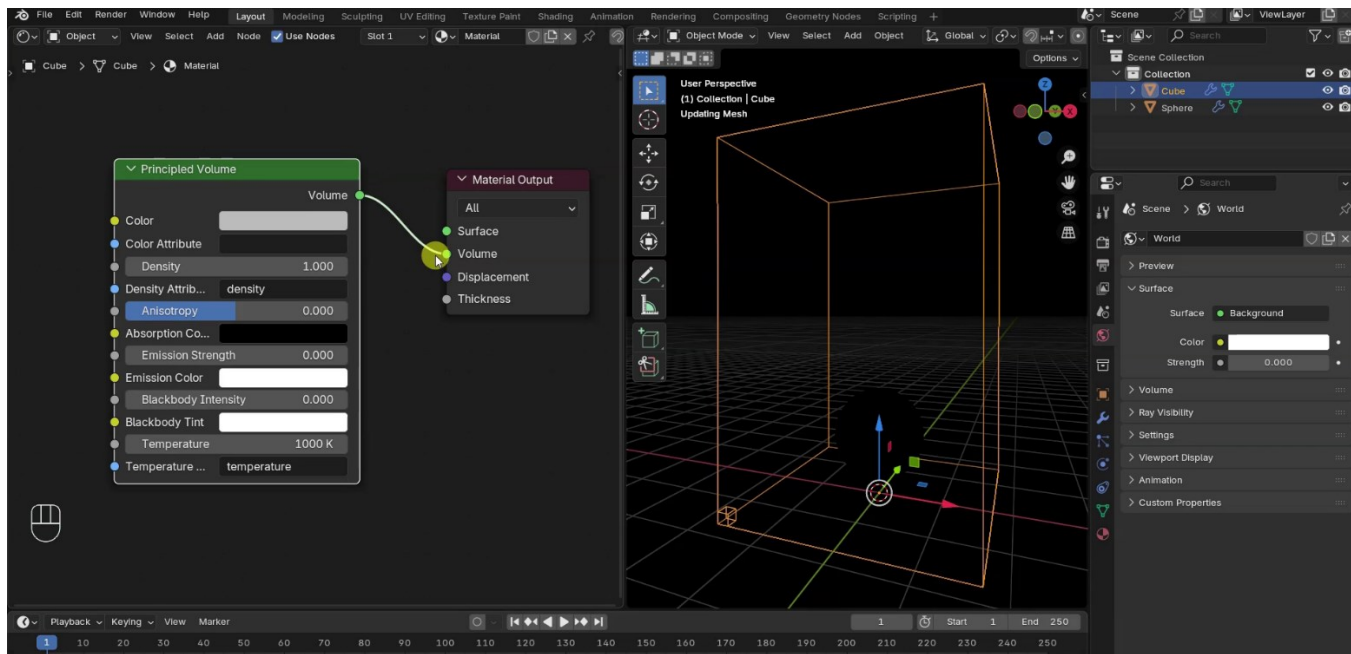
今回は前回のチュートリアルの最後で使用したシーンから始めますが、今回は Blender ファイルを保存し、プロジェクトファイルと同じディレクトリ内に cache フォルダーを設定してあります。

fire と smoke の Material は、シーン内の Flow オブジェクトではなく Domain に設定しなければなりません。そこで Domain を選択し、インターフェースに Shading Editor を追加して Rendered ビューモードに切り替え、基本的な Material を作成します。

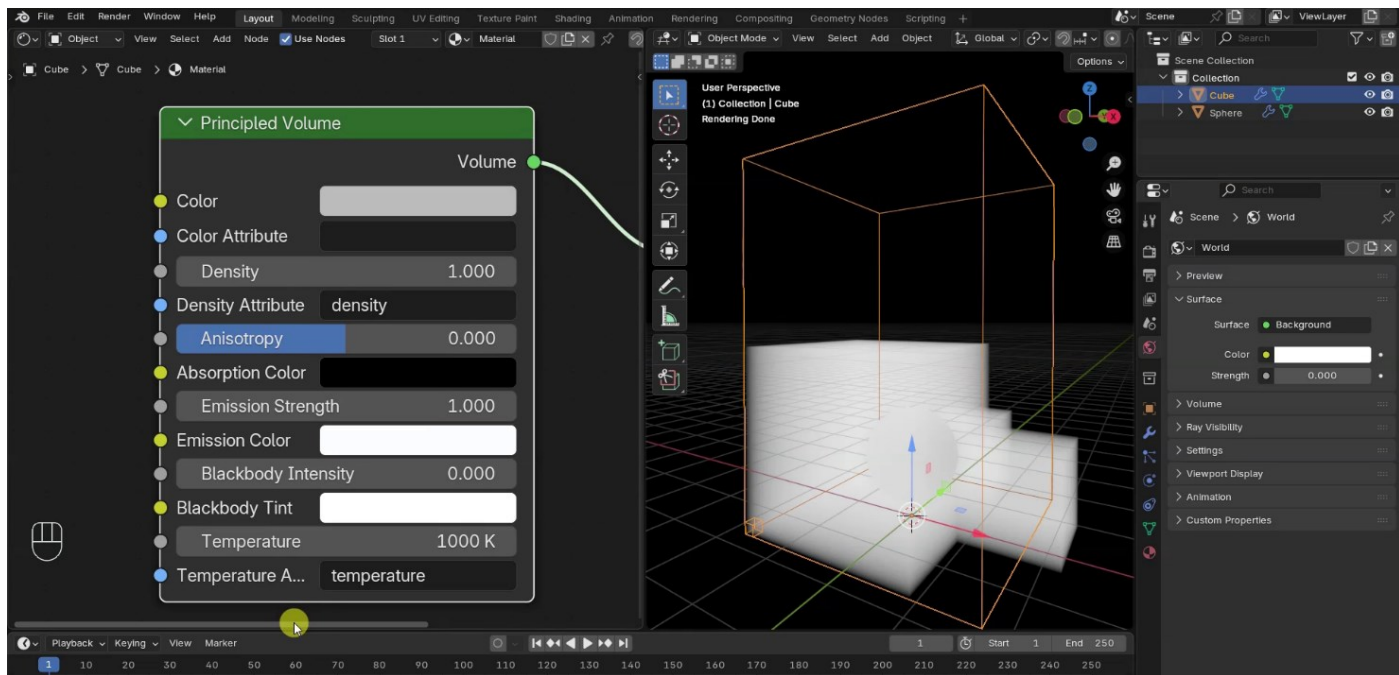


World タブで Background の Strength を 0 に設定し、炎が際立つようにします。アニメーションを再生しても、まだ何も表示されません。これは、デフォルトの Material に含まれる Principled Shader が Volumetric Material に適していないためです。

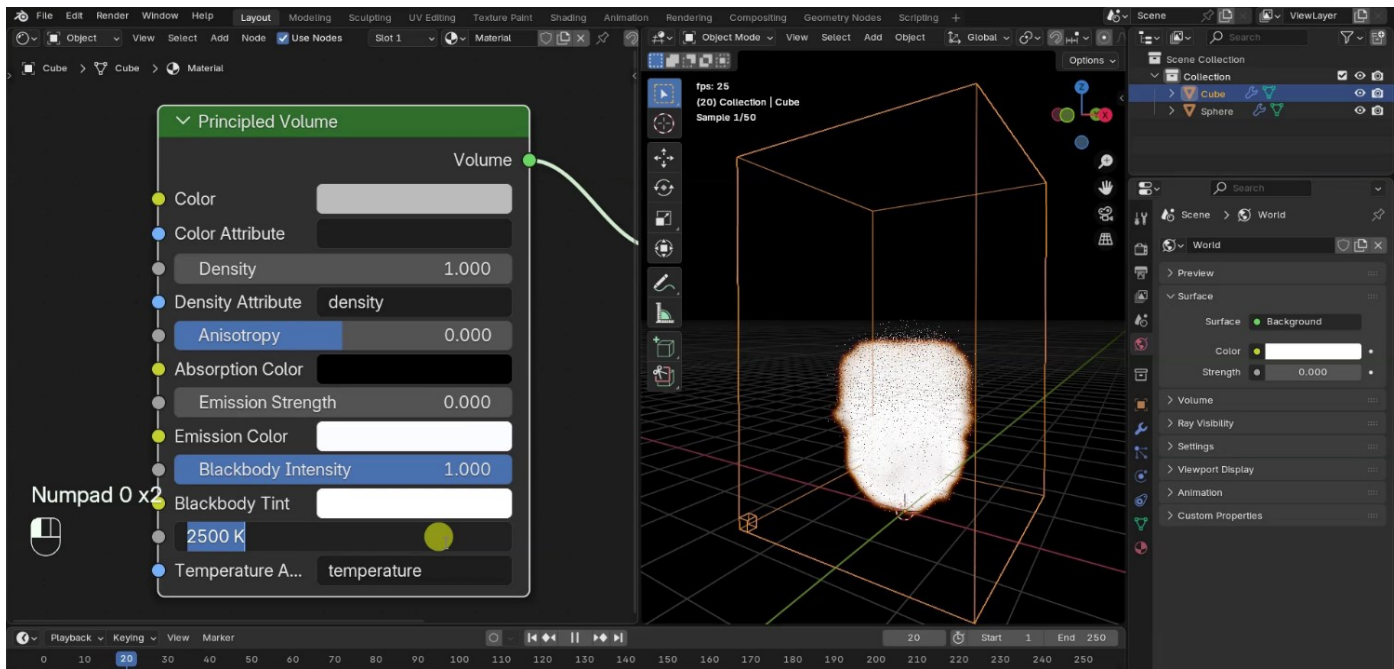
Principled Shader を削除し、代わりに Principled Volume Shader を挿入して、その出力を Material Output ノードの Volume 入力に接続します。ですが、この状態でアニメーションを再生しても、Rendered プレビューにはまだ何も表示されません。ここで、炎に色を付ける方法を見ていきましょう。



Blender には炎を着色する2つの方法が用意されています。最初の方法は Emission と呼ばれるものです。これは物理的には正確ではなく、最初の見目は少々残念に感じるかもしれませんが、実際には炎の強度や色を柔軟に調整できるため非常に便利です。このモードとその用途については、後のエピソードでさらに詳しく解説します。



2つ目の方法は Blackbody と呼ばれるもので、こちらは物理的に正確です。特に Strength を 1 に設定し、適切な温度値をフィールドに入力すれば、正しい結果が得られます。このツールを使うと、Rendered プレビューですぐに美しい炎が確認できます。Temperature の値はケルビン単位で表され、インターネット上には実際の数値とそれに対応する炎の色の例が数多く公開されています。



続ける前に、Render タブでプレビュー用の Denoise を有効にします。

物理学において Blackbody とは、すべての電磁放射を吸収し、それを温度だけに依存するスペクトル、ここでは単純に「色」と呼びますが、それとして再放射する理想的な物体を意味します。

温度が上がると、赤熱したストーブの赤から黄色や白に変化し、さらに上がると色は青に近づいていきます。

実際、表面温度が最も高い星は青白い巨星です。

Intensity と Tint パラメータについて説明します。まず Intensity は直感的に理解できる値で、先ほど触れたとおり、物理的に正しい値は 1 です。

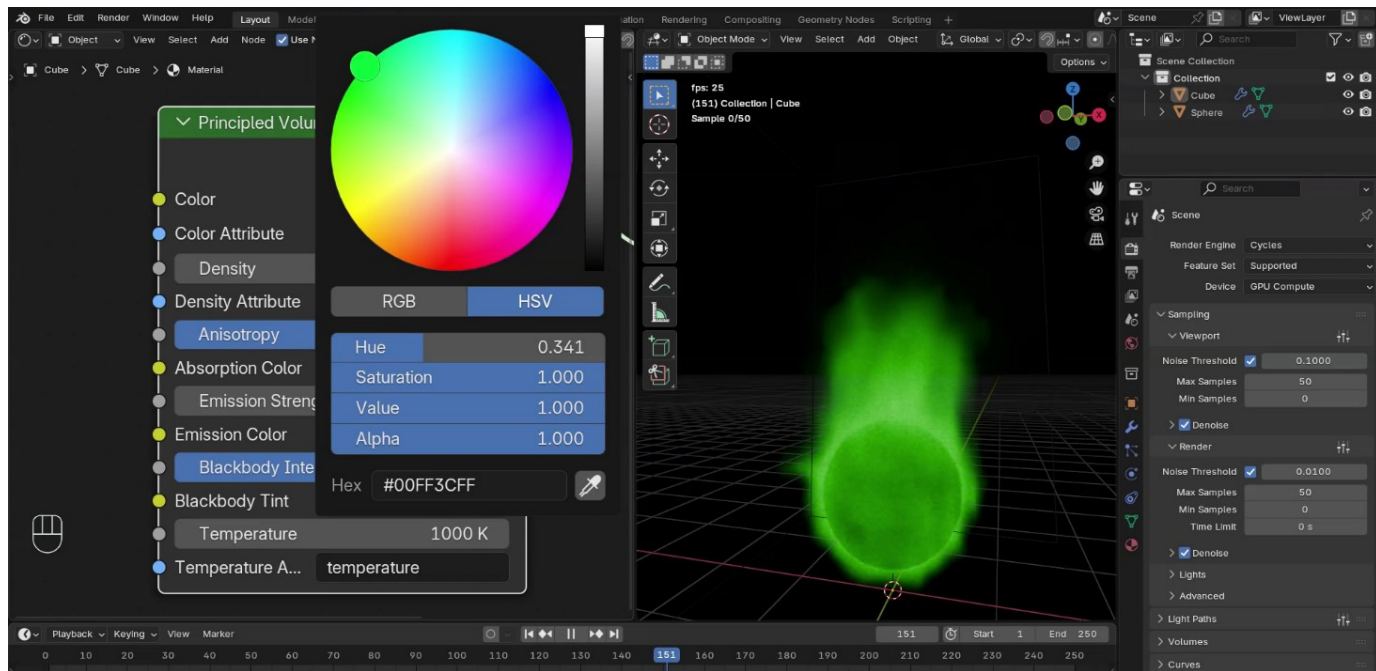
1 より大きくすれば fire は明るくなりますが、物理的には正確ではありません。

Tint フィールドでは炎の色を変更できます。

もちろん、これは物理的に正確な表現ではありません。

物理的な正確さを求めるなら、Temperature の値を使い、Tint は白のままにしておくべきです。

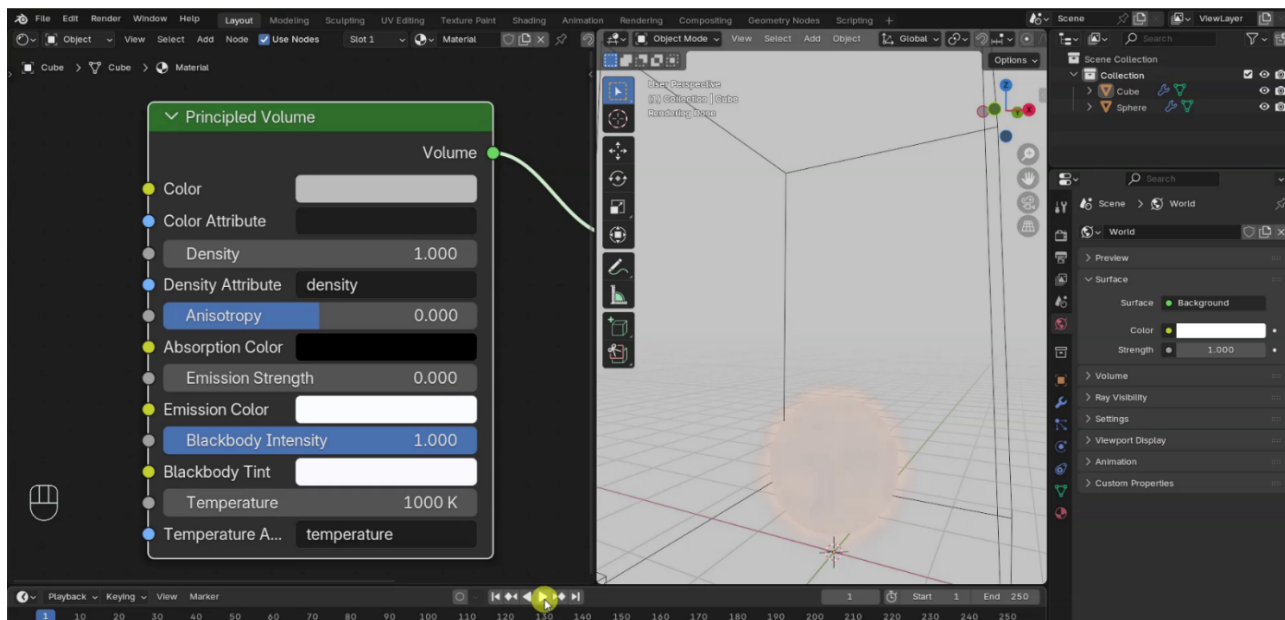
しかし、Emission でノードやパラメータを設定することなく手早く炎に色を付けたい場合、このオプションは便利です。



ここで「Inflow オブジェクトごとに異なる炎の色を付けるにはどうすればいいのか？」と疑問に思うかもしれません。なぜなら、色は Inflow オブジェクトではなく Domain の Material で定義されるからです。基本的には2つの方法があります。1つは Domain を別々に作成することですが、これは炎同士が相互作用しない場合にしか適しません。もう1つの方法は、emitter オブジェクトからのパラメータや Attribute を使ってそれぞれの炎に異なる色を与える方法ですが、必ずしも上手くいくとは限りません。これについては、他

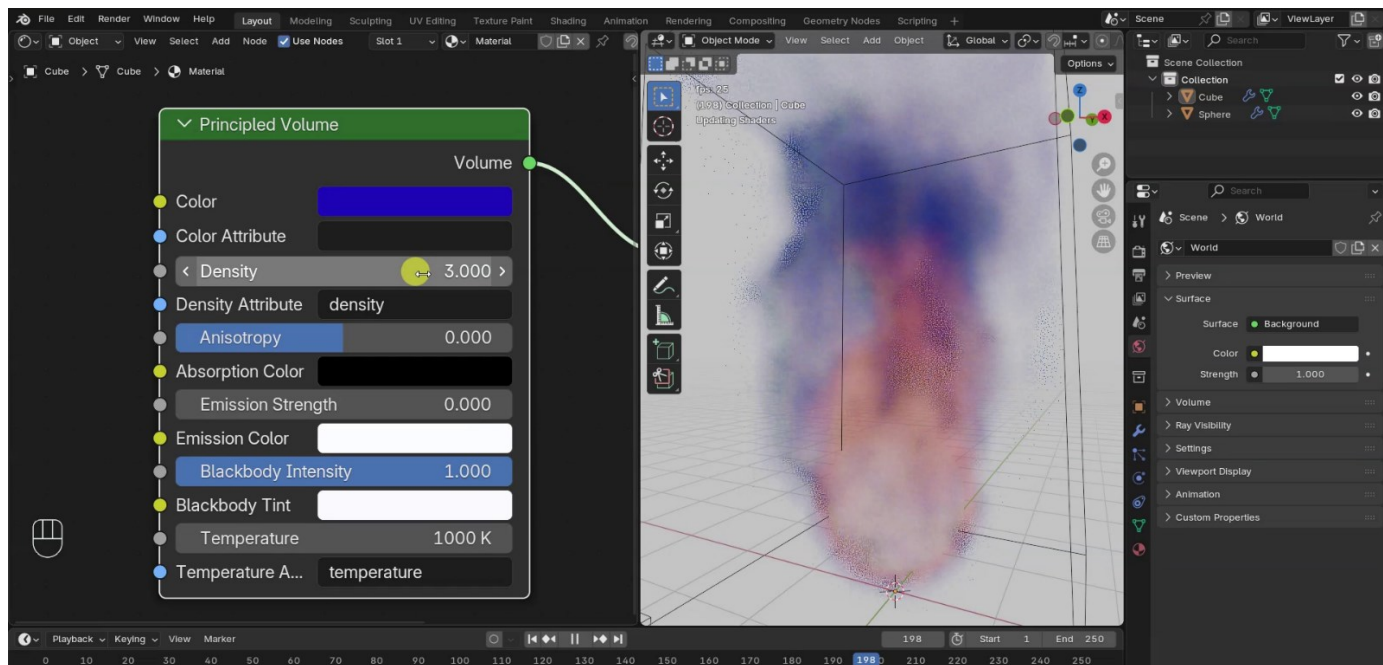
のオブジェクトパラメータや Principled Volume をさらに詳しく見た後、別のエピソードで可能な解決策を紹介します。

Principled Volume ノードの上部には Color というフィールドがあり、デフォルトでは灰色になっています。これは smoke の色を表しています。Principled Volume は単一のノード内で flame と smoke の見た目を個別のパラメータで定義できるからです。Density の値も smoke に関係します。smoke を見せるために、World Background Strength を 1 に戻します。



煙粒子を通過する光の散乱は Anisotropy と Absorption Color で調整できます。

Anisotropy では散乱の方向を制御でき、値が負の場合は後方、正の場合は前方に強く散乱します。その違いを分かりやすくするために、Density の値を 2 に設定して smoke を濃くし、flame が smoke の後ろに来るようにオブジェクトを配置しました。Absorption Color フィールドでは、光が smoke に吸収される際に色味を加えることができます。



ノードで利用できる Attribute について説明する前に、まず Domain が保存する情報を解説する必要がある
ので、このエピソードはここで終了します。

今回見たのはレンダリング時における fire と smoke の見た目を定義するための最初のステップにすぎません。

しかし、前回と今回の内容で、炎を作成してレンダリングするために必要な基本知識はすでに揃っています。

数回後のエピソードでは、特に Emission を活用するために Material が提供する他のツールも分析していき
ますが、その前に flame と smoke をよりコントロールするための simulation の基本をさらに扱う必要があ
ります。

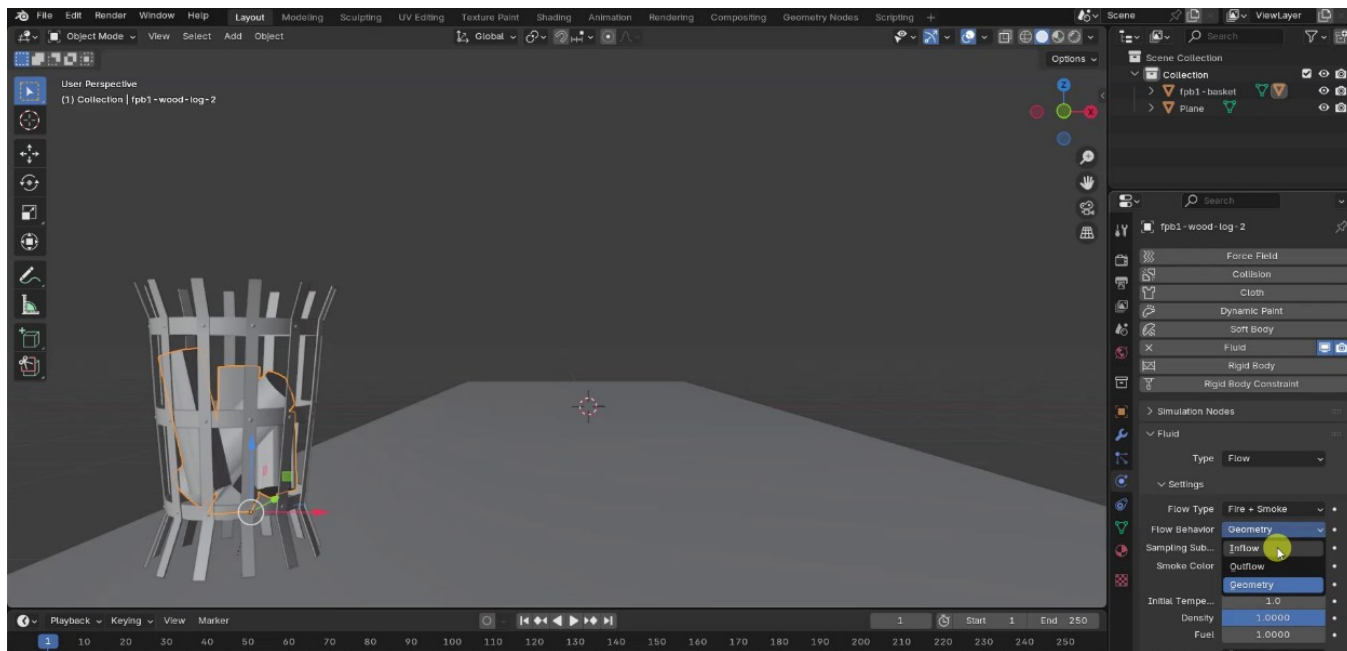
Fire Simulations in Blender 4.5 basics | 3/10: Cache, Baking, Offset

<https://youtu.be/uOJ6TA0rUOA>

皆さん、こんにちは！このシリーズの第3回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke において、Bake 一つまり fire と smoke の物理シミュレーションがどのように実行され、どのように保存されるかを解説します。Domain の simulation を一度計算すれば、シーン内の他の場所に複製しても再計算する必要がない方法も見ていきます。最後に、simulation の再生に Offset を設定し、炎の点火から始めるのではなく、すでに燃えている状態からアニメーションを開始できるようにする方法を紹介します。

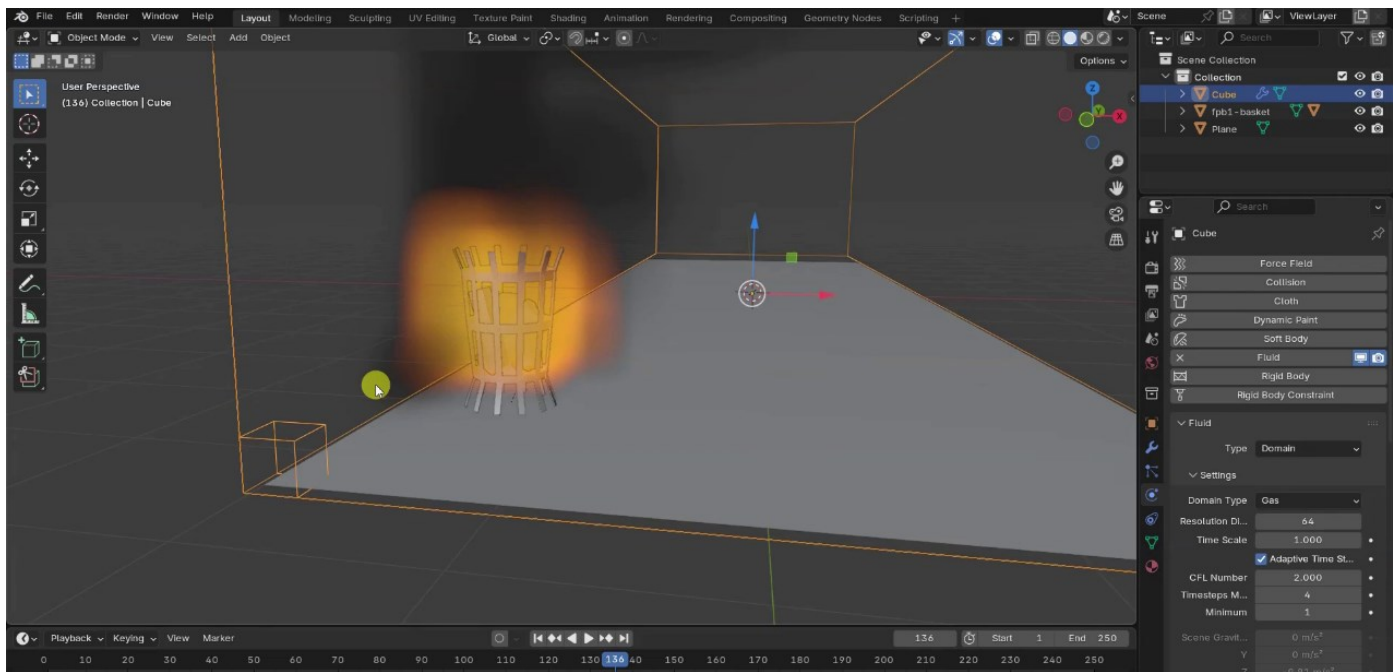
今回のチュートリアルの開始シーンは、薪を入れた brazier（火鉢）です。これらは別々のメッシュで、薪は1つのメッシュとしてまとまっています。brazier は長方形の Plane の上に置かれていて、これが床の役割を果たし、後で他の brazier もここに配置します。World の Strength は 0 に設定されており、光源もないため、Rendered モードではシーンが真っ黒に見えます。

前回までの内容から、オブジェクトに炎を発生させるには Fire And Smoke Flow タイプの Fluid コンポーネントを割り当てる必要があることを知っています。そこで薪に Physics Fluid コンポーネントを追加し、その挙動を Inflow に設定します。これにより薪は各フレームで flame を発生させるようになります。

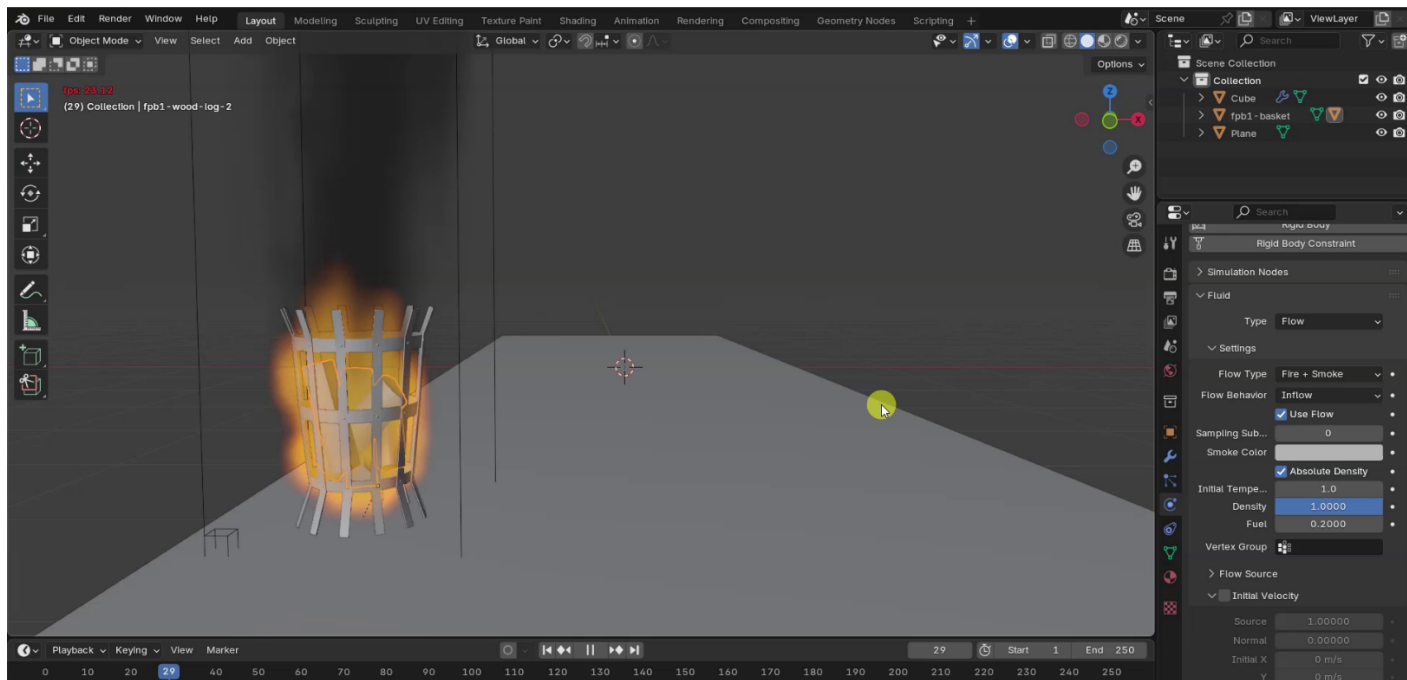


次に Cube を作成し、Flow コンポーネントを Domain タイプに設定してシーン全体を最初に囲みます。するとすぐに問題に気づきます。デフォルトの resolution が低すぎて Voxel が大きすぎるため、炎が emitter となるオブジェクトに比べて不自然に大きくなってしまいます。

Resolution を 64 に上げてても大きな改善はなく、simulation にも時間がかかるようになります。これ以上上げるのは試すまでもなく正しいアプローチではないと分かります。



最も分かりやすい解決策は、Domain をリサイズし、brazier とその上に広がる flame と smoke を収める程度の大きさにすることです。すると今度は resolution が効きすぎるほどで、むしろ過剰かもしれません。Resolution を 32 に戻しても炎は十分にくっきりしています。むしろ問題は、薪が fire を出しすぎていることです。



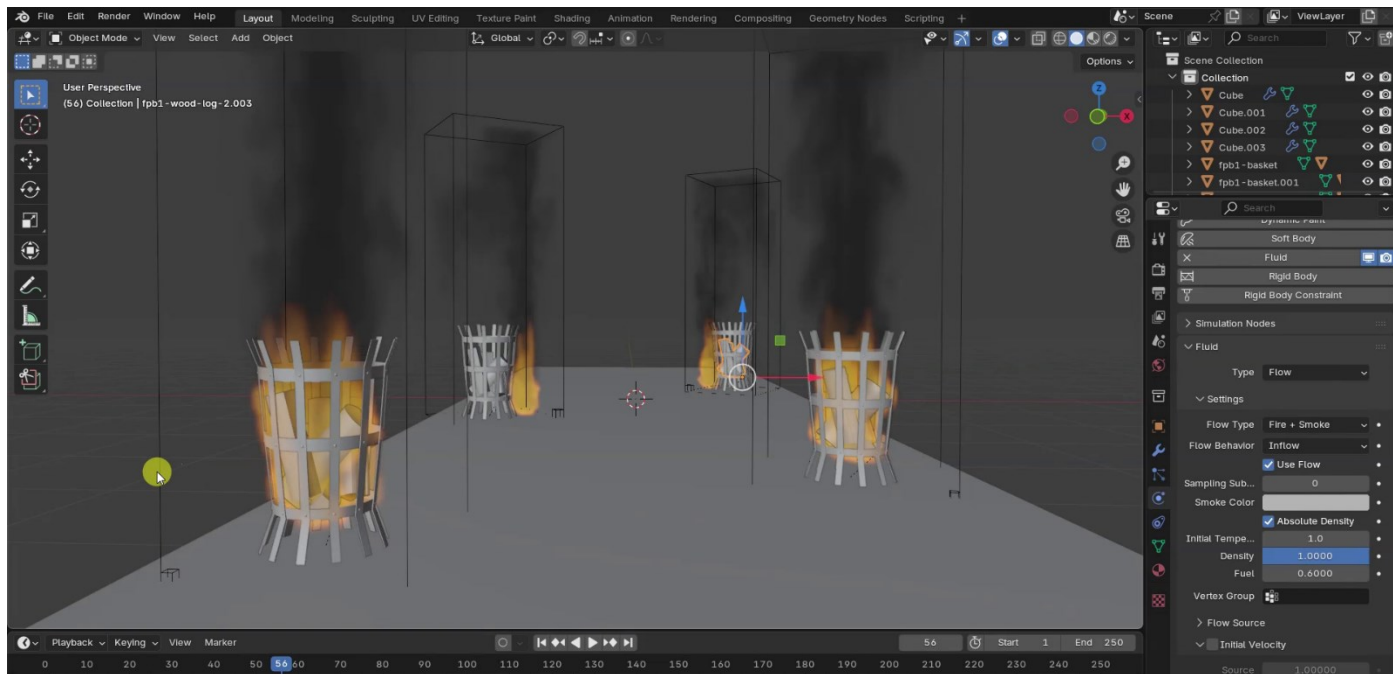
発生する fire の量を減らすには、Inflow オブジェクトの Fuel パラメータを調整します。直感的に、Fuel の値が大きいほど、各フレームで消費される燃料が多くなります。解決策はこの値を下げ、炎が適切な大きさになるまで調整することです。

前回までに説明したように、Baking（物理シミュレーションの計算）に関連するデータは専用のフォルダーに保存されます。そのパスは Domain オブジェクトの Cache セクションに表示されています。Domain を作成する前にファイルを保存してあるので、cache フォルダーは Blender プロジェクトファイルと同じディレクトリに作成されます。

Domain の Cache セクションでは、Domain simulation を計算するフレーム範囲を設定できます。これは必ずしもシーンのアニメーション全体のフレーム範囲と一致させる必要はありません。例えば、短い範囲だけ Bake して fire の見え方を確認したいときや、Fluid オブジェクトがカメラに映らないフレームではシミュレーションを計算する必要がないときに便利です。Cache セクションの他のパラメータを見る前に、Domain、brazier、薪（Inflow オブジェクト）をひとまとまりのセットとしてコピーし、それをシーンに配置した場合を見てみましょう。前に見たように、シーン全体を覆う Domain を1つだけ使うと、良い flame を作るには非常に高い resolution が必要で、しかも無駄な空間が多く残ってしまいます。したがって、複数の brazier を追加する場合は、セット全体を複製するしかありません。

問題は、Timeline で Play を押すと Blender がすべての Domain に同じ cache フォルダーを使ってしまうことです。つまり、すべてが同じ simulation データを共有してしまうのです。もし全シミュレーションで同じパラメータを使いたいのであれば、Domain を少し回転させたり、それぞれ異なる Material を与えたりして、レンダリング時に flame の見た目を変えることで視聴者を「だます」方法もあります。

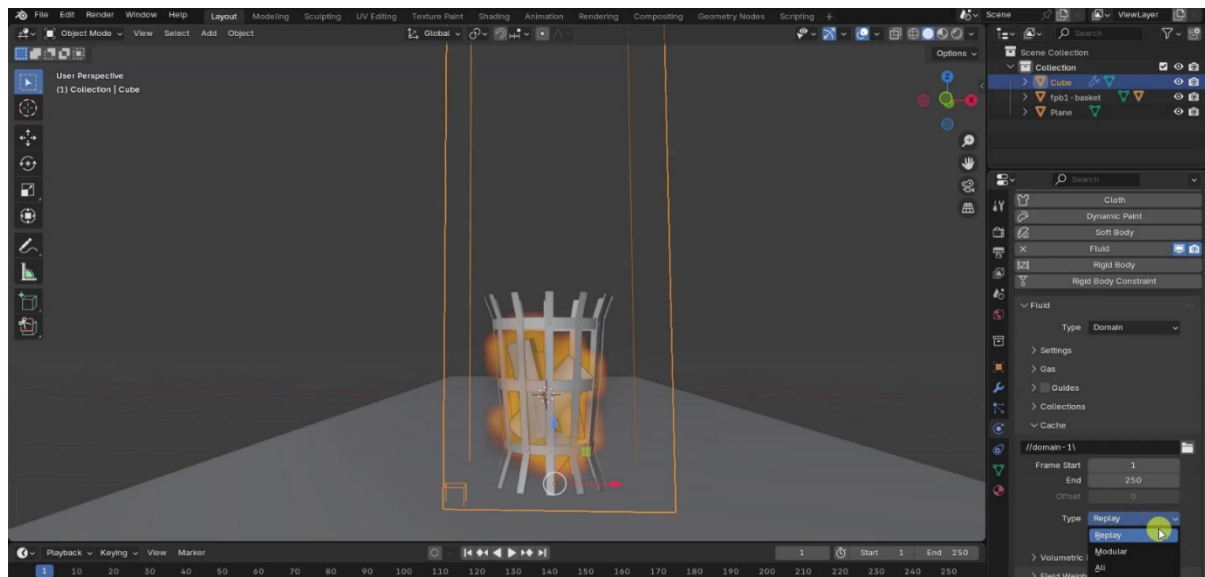
本当の問題は、Domain の resolution や他のパラメータを変更してから再び Bake を実行したときに発生します。例えば、背景にある Domain には高い解像度が必要ないので resolution を下げています。すると Blender は異なる特性を持つオブジェクト間で同じ volumetric data を共有してしまい、完全に混乱してしまうのです。



各 Domain ごとにデータを明確に分けたい場合は、simulation を実行する前に Domain ごとに別々の cache フォルダを指定する必要があります。言うまでもなく、こうすると各 Domain のデータでディスクの使用量は大幅に増えます。どの戦略を採用するかはニーズ次第です。非常に大きな Domain をひとつか複数使って同じ simulation を複製するのか、あるいは Domain ごとに別々の cache を作成するのを選ぶ必要があります。

これまで Blender は、Timeline で Play を押すたびに simulation の Bake を計算していました。しかし、これは唯一のモードではありません。実際には非常に便利なツールを使えるもうひとつの方法があります。ここでは話を簡単にするために、シーン内に Domain を持つ brazier をひとつだけ残し、他を削除しました。ただし注意してください。Domain を削除してもディスク上のデータは削除されません。これは良い点でもあります。なぜなら後でシミュレーションデータを保存して再利用できるからです。ただし今回はそのデータが不要なので、手動で削除します。

Domain の Cache セクションには Type メニューがあり、デフォルトでは Replay に設定されています。これがこれまで使ってきたモードで、frame 1 に戻って Play を押すたびに Blender が新しい Bake を再計算し、以前のデータを上書きします。

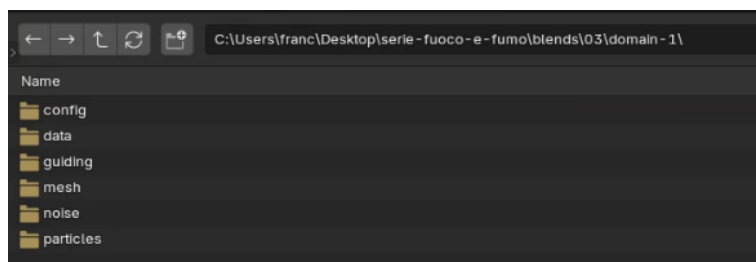


Modular モードでは、simulation の異なる側面を分けて Bake できます。これにより、シミュレーション全体をやり直さずに特定の部分だけを確認したり置き換えたりすることが可能になります。

All モードでは、シミュレーション全体を Bake しますが、その計算は Timeline ではなくバックグラウンドで行われます。Bake を開始するにはパネルの Bake All ボタンをクリックします。Blender は進行状況バーを表示します。計算が完了すると、Timeline で Play を押すとすでに事前計算されているため、はるかにスムーズに結果を確認できます。

simulation の設定を変更するには、まず Free All ボタンをクリックして既存のデータを削除する必要があります。その後パラメータが再び編集可能になり、変更できます。もちろん、その後もう一度 Bake All をクリックして Blender にシミュレーションを再計算させる必要があります。

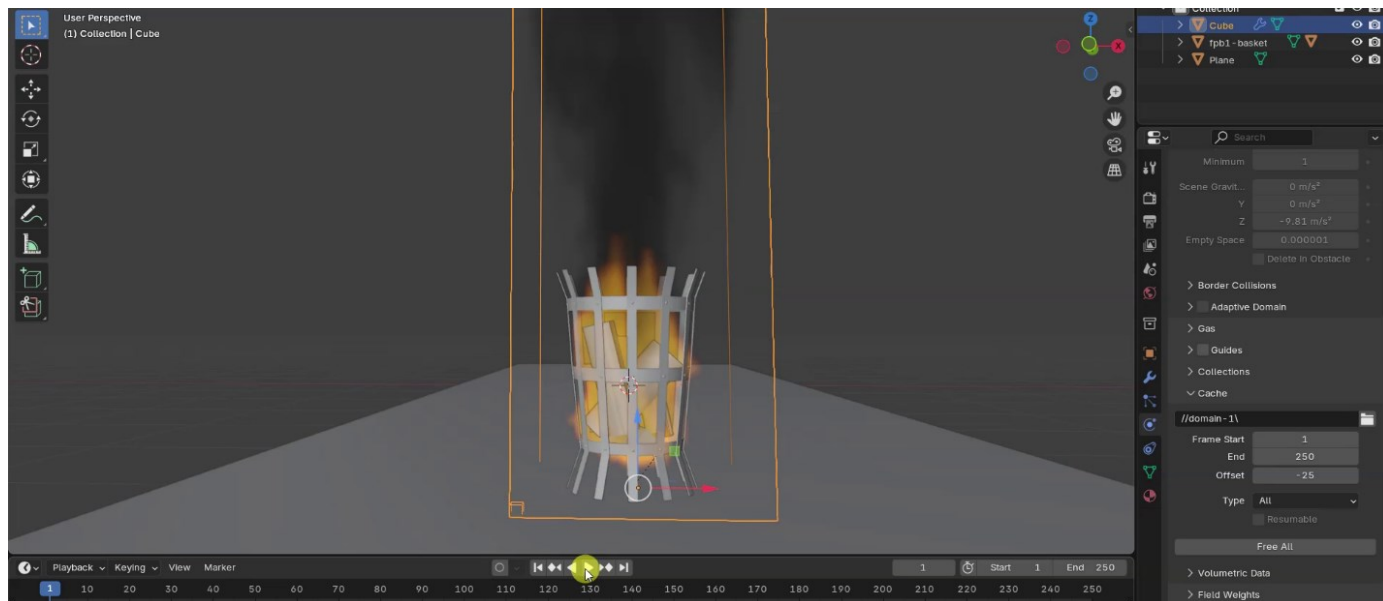
使用中のフォルダーを確認すると、Modular や All モードを使った際に、どのようにモジュールや bake のステップが分けて計算・保存されているかが分かります。



All モードでは Offset パラメータが利用可能になり、すでに気づいているかもしれない問題を解決できます。これまでの fire simulation はすべて、アニメーションの最初のフレームで炎が点火するところから始まっていました。しかしこれは必ずしも望ましいわけではなく、すでに燃えている状態からアニメーションを開始したい場合もあります。

この問題を解決するには、Offset フィールドに負の数を入力します。ツールチップには、これは Bake が早く始まるという意味ではなく、最初のフレームでシミュレーションがその数のフレーム分すでに進んでいるかのように再生されると説明されています。アニメーションの長さは Frame Start と Frame End の間で指定されたフレーム数のままです。この場合、Timeline の最後の25フレームにはシミュレーションが存在し

ません。なぜなら、Offset によって Timeline の1フレーム目でシミュレーションの25フレーム目を再生するように Blender に指示したからです。



これを解決するには、Bake でより多くのフレームを計算し、シミュレーションが Timeline 全体をカバーするようにすればいいのです。そこで Cache セクションの Frame End を 275 に設定し、Free All をクリックしてから Bake All をもう一度実行しました。

Fire Simulations in Blender 4.5 basics | 4/10: Geometry, Inflow, Outflow

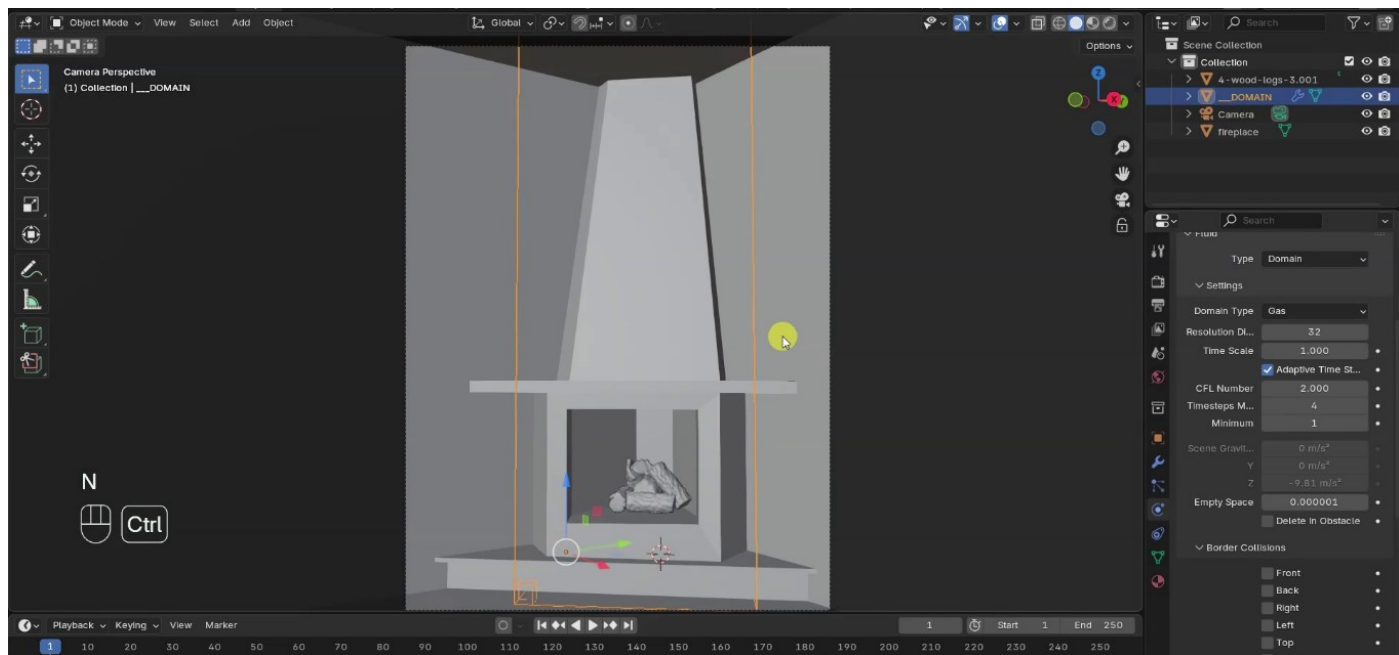
<https://youtu.be/Q8QYFOEueOE>

皆さん、こんにちは！このシリーズの第4回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の物理シミュレーションで使用される3種類の Flow オブジェクトのパラメータを詳しく見ていきます。

今回使用するシーンには、暖炉の中にいくつかの薪が配置されています。これらはひとつの mesh オブジェクトとしてまとまっており、Inflowとして設定します。シミュレーション用の Domain はすでに作成しており、暖炉のボリュームだけを囲んでいます。

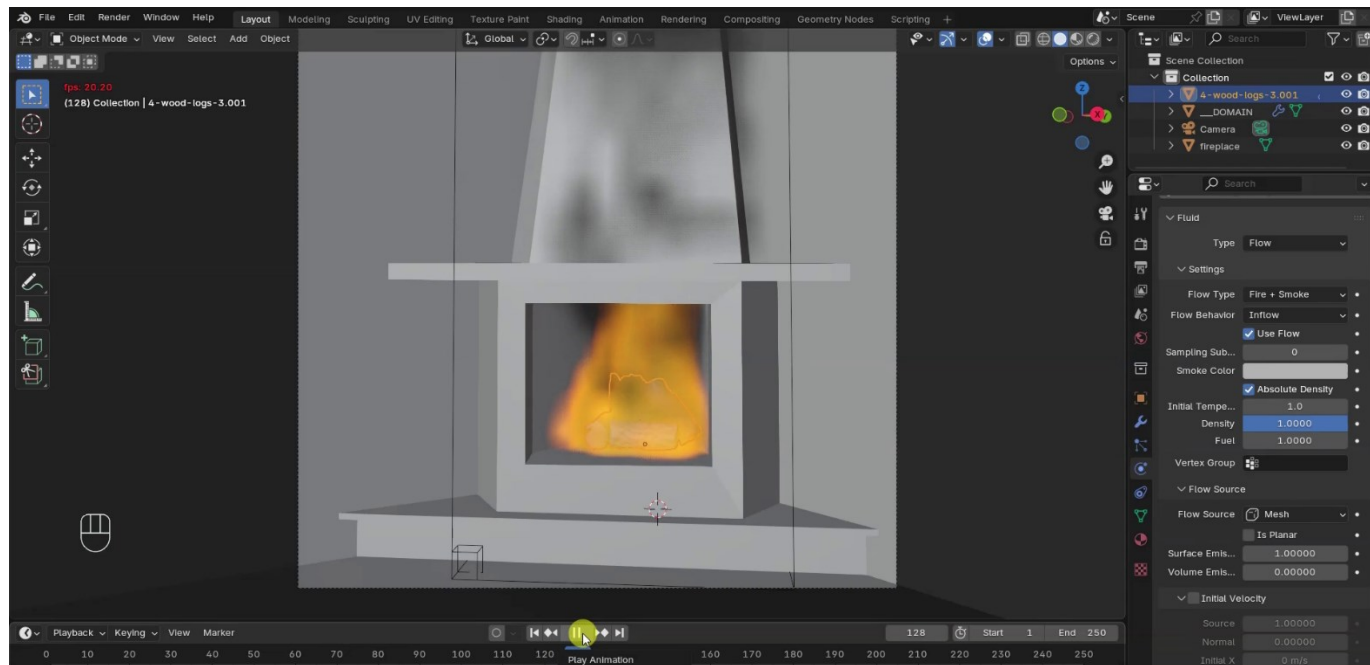
薪に Fluid コンポーネントを追加し、タイプを Inflow に設定します。Fire And Smoke モードには Smoke と Fire の両方のパラメータが含まれているため、両方を確認するためにこの組み合わせを選びます。

後ほど、Smoke のみに関連するパラメータを指摘します。ご覧のとおり、パラメータの数はそれほど多くありません。



デフォルトでは Flow Behavior が Geometry に設定されています。前回までに見たように、このモードでは最初に炎と煙を発生させた後、オブジェクトはそれ以上 fire や smoke を放出しません。爆発や一度きりの煙のパフには便利ですが、今回必要なのは Inflow モードです。いずれにせよ、Use Flow チェックボックスを除けば、Geometry と Inflow は同じパラメータを共有しているため、ここでは Inflow を見ながらすべてを確認していきます。

前回のエピソードで説明したとおり、炎の量を制御するには Fuel パラメータを調整できます。今回の場合、少し下げれば活発だが過剰ではない炎を作れますし、大きく下げれば赤熱した熾火のような表現が可能です。



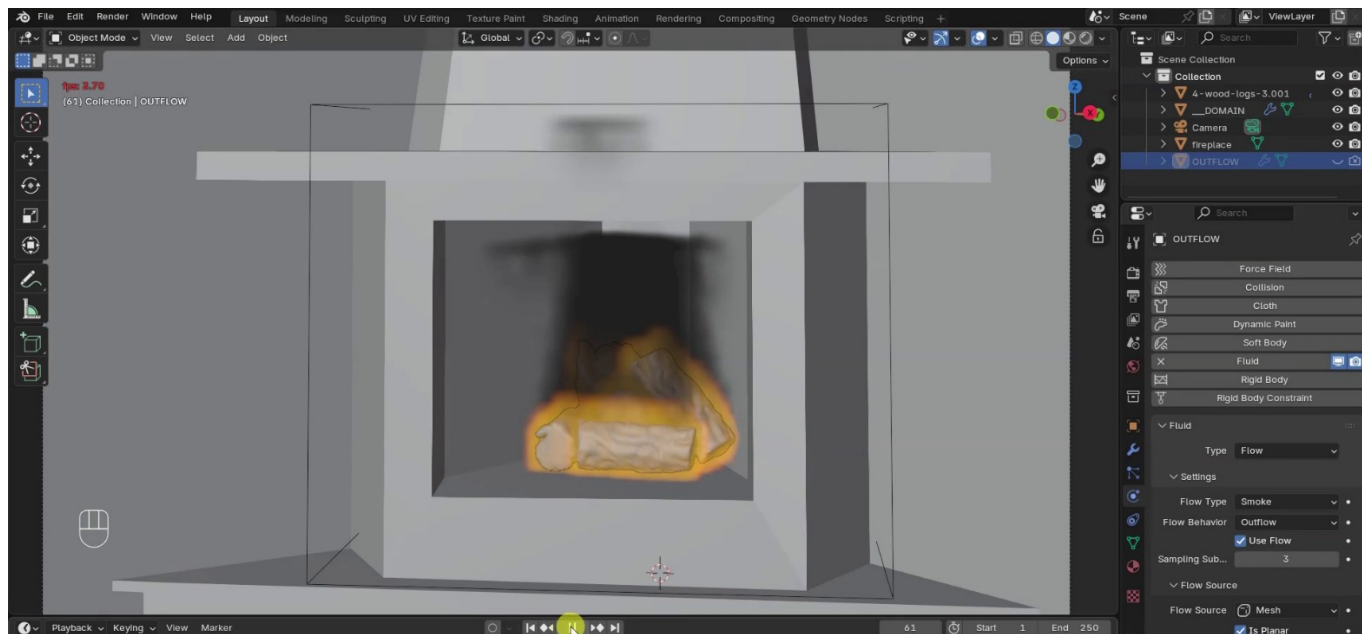
Domain のサイズを少し小さくして解像度を上げました。Inflow パラメータを確認する前に、Outflow オブジェクトを使って暖炉内の煙を消す方法を見ていきましょう。

Outflow オブジェクトは mesh で、fire や smoke を衝突時に消す役割を持ちます。ここでは単純な Plane を追加して Outflow として使い、暖炉の上部、煙突に煙が上がる位置に配置しました。Outflow オブジェクトは Inflow と同様に、レンダリングで表示される必要はありません。そこで 3D Viewport プレビューと最終レンダリングの両方で Outflow オブジェクトの表示を無効にします。今後は Outliner から選択します。

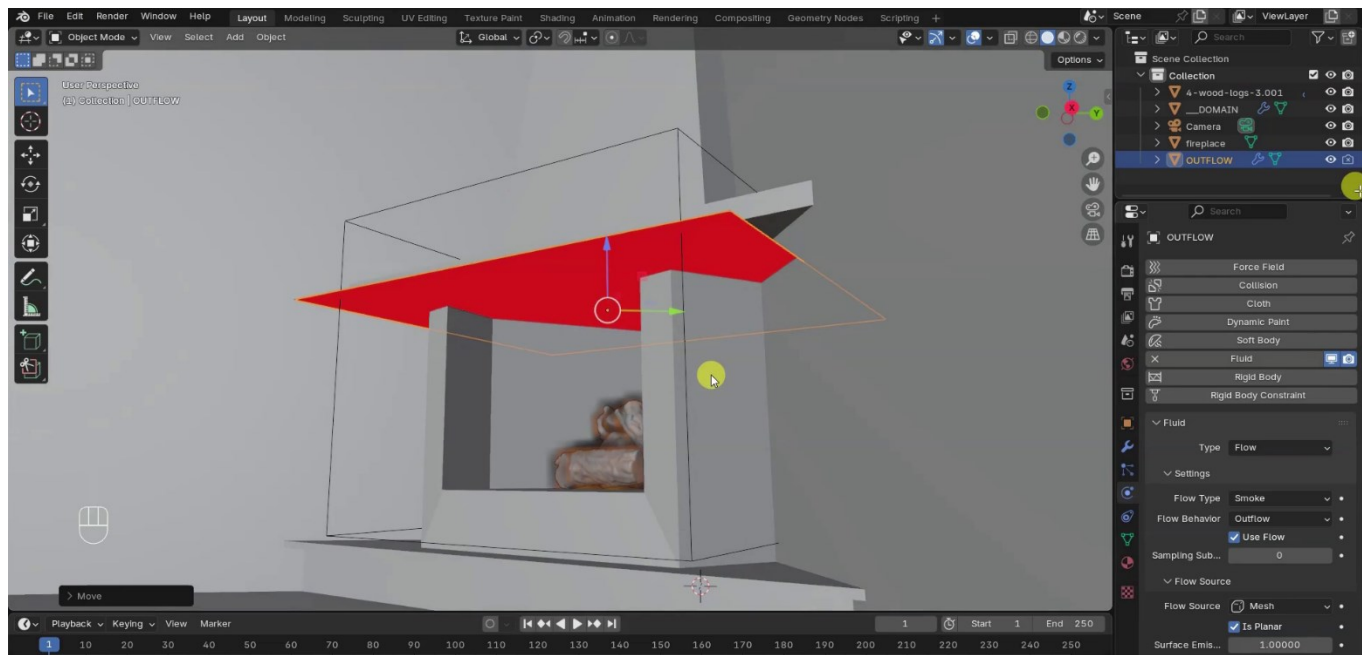
そのオブジェクトに Fluid コンポーネントを追加し、タイプを Smoke Flow、Behavior を Outflow に設定します。このタイプのオブジェクトにはパラメータがほとんどなく、実際には Geometry や Inflow にも存在するものです。簡単に説明します。Is Planar オプションは、オブジェクトにボリウムがない場合、あるいは閉じていない場合に選択します。Plane はまさにそのようなオブジェクトなので、このオプションを有効にします。Use Flow パラメータは Inflow や Outflow を有効または無効にでき、アニメーション可能なので、断続的または不連続に fire や smoke を発生・除去したい場合に便利です。Sampling Substeps パラメータはフレーム間で追加計算されるシミュレーションステップの数を定義し、シミュレーションの品質を向上させます。特に高速なシミュレーションに有効で、3 などの低い値でも明確な違いが見られます。Surface Emission と Volume Emission は fire や smoke の放出や除去の位置をオブジェクトの surface または volume からどの程度ずらすかを制御しますが、この状況では不要です。

Flow Source については後ほど解説します。

シミュレーションを再生すると、最初は問題ないように見えますが、ある時点で煙が少し上に抜け出してしまふのが分かります。Substeps の数を増やしても状況は変わりません。



Plane を 3D View で再び表示させ、視点を試してみると解決策が明らかになります。Plane が小さすぎるため、ある時点で煙が外に抜け出して上昇してしまうのです。唯一の解決策は Plane を適切にリサイズして、もう一度試すことです。



では、Outflow で扱ったものを除いて Inflow オブジェクトのパラメータを詳しく見ていきましょう！ Smoke Color は実際には 3D Viewport プレビューでの煙の色を指しています。レンダリング時の煙の色は別の場所で定義されます。ここで異なる色を割り当てることで、複雑なシーンで煙を見分けやすくしたり、このオブジェクトの Attribute を Shading で利用して Domain Material の色やその他の要素を変化させたりすることができます。

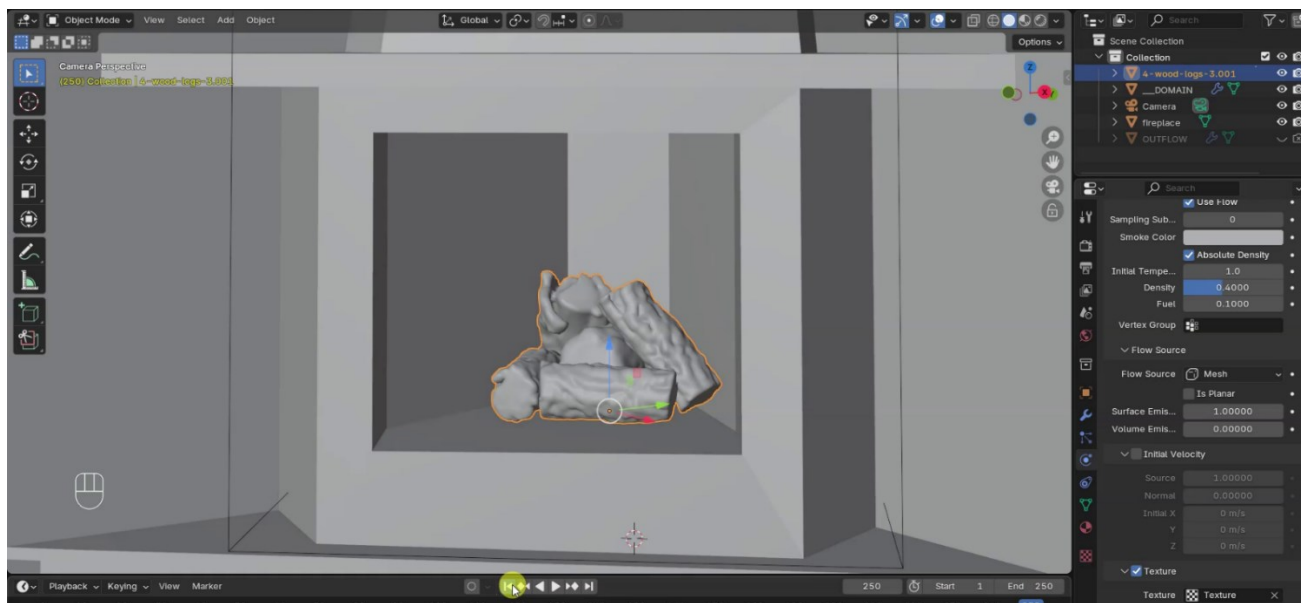
Absolute Density フィールドはデフォルトで有効になっており、通常はそのままにしておくのが良いです。これにより、Domain 内の fire と smoke の密度に一定の上限が設けられます。無効にすると fire と smoke が極端に増え、制御不能の炎のような結果になることもあります。ここでは Outflow があるので影響は抑えられますが、デモとしてこのオブジェクトを一時的に Domain から外して炎が増える様子を見せています。Absolute Density を無効にするのが有効な場合もありますが、多くのケースではそうではないため、再び有効に戻します。

Initial Temperature と Density パラメータは smoke に関連するもので、Fire のみの simulation では使用できません。Density は放出する煙の量を指し、値が高いほど煙のボリュームが大きくなります。今回のシーンでは煙が多すぎるので、この値を下げます。

Initial Temperature パラメータは、煙の温度と周囲の環境温度との差を定義し、ガスが上昇する速さを制御します。ただし、その効果は Domain のパラメータである Buoyancy Heat に依存しているので、別のエピソードで詳しく解説します。

これまで Inflow オブジェクトは表面全体から flame を放出していましたが、Vertex Group で頂点を選択し、その名前を Flow コンポーネントの専用フィールドに指定することで、放出をオブジェクトの特定部分に限定することもできます。

flame の放出にバリエーションを加えるもうひとつのツールが Inflow パネルの下部にある Texture です。ここで Texture を割り当てられ、特に重要なのは Offset フィールドです。これは procedural Texture のどの「スライス」を使うかを定義します。Clouds や Noise などの procedural Texture は2D イメージではなく、実際には 3D volume です。Offset 値を Timeline に沿ってアニメーションさせることで、Texture が縦方向にシフトし、毎回異なるパターンで放出されます。これにより flame の見た目が均一ではなくなります。ただし、この方法は適切な Texture の種類と Offset の変化の組み合わせを見つけるまでに何度も試行が必要になるのが難点です。



Initial Velocity グループのツールを使えば、flame に初期の方向や速度を与えることができます。

フィールド名は直感的なので説明はこれくらいにして、実際の例は別のシーンで後のエピソードにてお見せします。

最後に Flow Source フィールドです。ここでは Mesh と Particles を選択できます。意味は分かりやすいと思います。これまでは Mesh を使い、オブジェクトから fire と smoke を放出してきました。

必要であれば Vertex Group で放出面を限定することもできます。

しかしオブジェクトには particle system を持たせることもでき、場合によっては emitter ではなく particles から fire や smoke を発生させたいこともあります。

その場合は Flow Source を Particles に設定し、もちろん mesh のどの particle system が simulation を駆動するかを指定する必要があります。ご想像のとおり、Flow コンポーネントはジオメトリに対して、または選択したオブジェクトの特定の particle system に対して割り当てることができます。

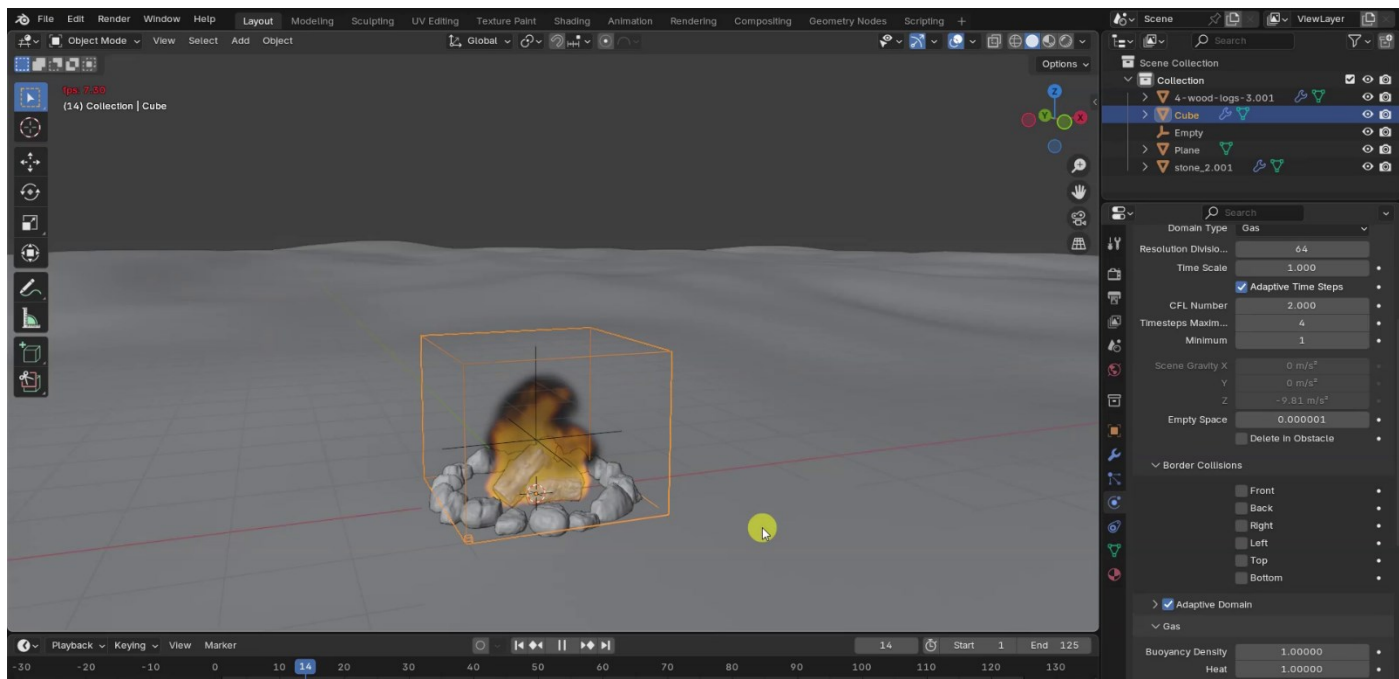
Fire Simulations in Blender 4.5 basics | 5/10: Domain Settings 1/2 (Gas, Fire, Collections)

https://youtu.be/9xM5c_6DY5g

皆さん、こんにちは！このシリーズの第5回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の simulation において、Domain の主要なパラメータのいくつかを見ていきます。具体的には Settings、Gas、Fire、Collections セクションのものです。これらのパラメータの一部を使えば、炎や煙をより生き生きと、均一でない見た目にすることができます。

まず Settings セクションの主要な要素から始めましょう。これまでに Resolution Divisions パラメータを見てきましたが、Domain の下の角に表示される小さな Cube によって示される Voxel のサイズが非常に小さい一方で Domain 自体が大きい場合、simulation を実行するために必要な計算量とメモリが増えることを理解しています。

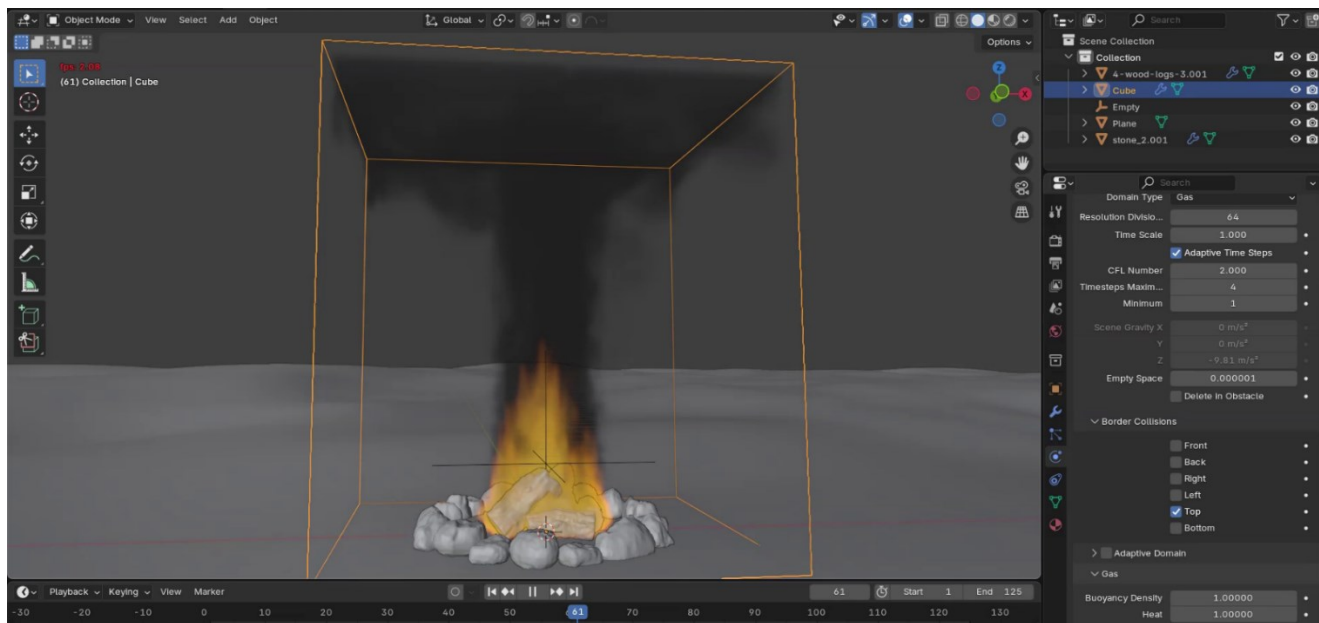
Settings セクションの下部にある Adaptive Domain オプションを有効にすると、Blender は simulation の間に Domain のサイズを動的に変更できます。



ただし、Adaptive を有効にしても Domain の高さはある程度までしか伸びません。その後、煙が Domain の天井に触れると Outflow として扱われ、消え始めます。Gas セクションには煙をより早く消散させる専用のオプションがあり、後ほど確認しますが、まずは Adaptive Domain を無効にして Border Collisions セクションを見てみましょう。

Border Collisions パラメータを有効にすると、バウンディングボックスの側面が障害物として扱われ、炎や煙が反射されます。デフォルトでは無効なので Outflow として扱われ、fire と smoke は境界に触れた瞬間に消えてしまいます。Domain の天井に Border Collision を有効にすると、煙はそこで蓄積し、横に広がりながら端に到達すると消えます。これは側面の Border Collision が無効だからです。

他の側面にも Border Collision を有効にすると、Domain は閉じた箱のように振る舞い、fire と smoke は壁に触れても消えず、内部に蓄積し続けます。



Settings パネルの上部に戻ると、Time Scale と Adaptive Time Steps があります。Time Scale は simulation の進行を遅くしたり速くしたりできます。

Adaptive Time Steps チェックボックスはデフォルトで有効になっており、アルゴリズムがアニメーションの各フレームに必要な中間ステップ (Steps) をいつ追加計算するかを決定します。このチェックボックスの下にある3つのパラメータは、それぞれアルゴリズムが毎回何ステップ計算するかを直接制御します。Maximum と Minimum はその名のとおりですが、CFL Number は少し複雑です。実装の詳細には触れませんが、知っておくべきことは、CFL Number を大きくすると必要な Steps が少なくなり計算時間も短縮されますが、精度が低下するということです。炎や煙が特に速く動く simulation では、CFL Number を低めに設定することで品質を改善するのが良いでしょう。

Gravity パラメータはデフォルトで無効になっています。なぜなら Scene タブのグローバル設定が使用されるからです。もし Scene タブでグローバルの Gravity を無効にすると、Physics 内でこれらのパラメータが再び有効になり、編集できるようになります。

Empty Space は Smoke 要素でのみ機能します。具体的には、このしきい値以下で満たされた Voxel は空と見なされ、simulation で無視されるため、計算時間を節約できます。

Gas セクションに移ると、最初に Buoyancy Density と Heat パラメータがあります。これらは気体に働く上向きの力を表しており、ひとつはガスの密度に基づき、もうひとつは温度に基づきます。これらの値は直接的に温度や密度を表すのではなく、それらによる上向きの力を示しています。

幸い、このパラメータのツールチップは非常に分かりやすく、値を上げればガスがより速く上昇することが説明されています。特に Buoyancy Heat は煙の熱による上昇力です。

煙の温度と周囲環境の初期温度差は、各 Inflow オブジェクトで Initial Temperature パラメータによって定義されます。また、Inflow オブジェクトのパネルには Density パラメータもあり、Domain の Buoyancy Density と関連しています。

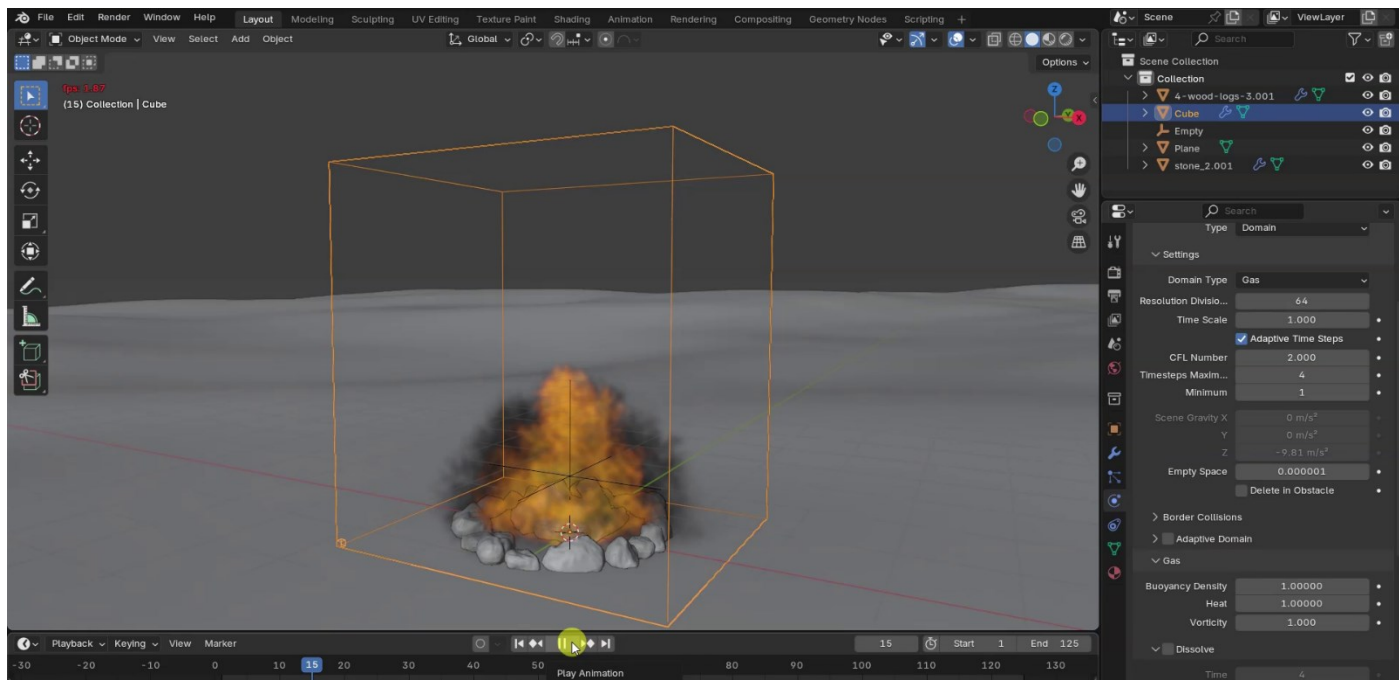
Vorticity は煙の乱流を制御し、間接的に炎にも影響を与えます。

値が高いほど煙が乱されます。ただし注意が必要です。

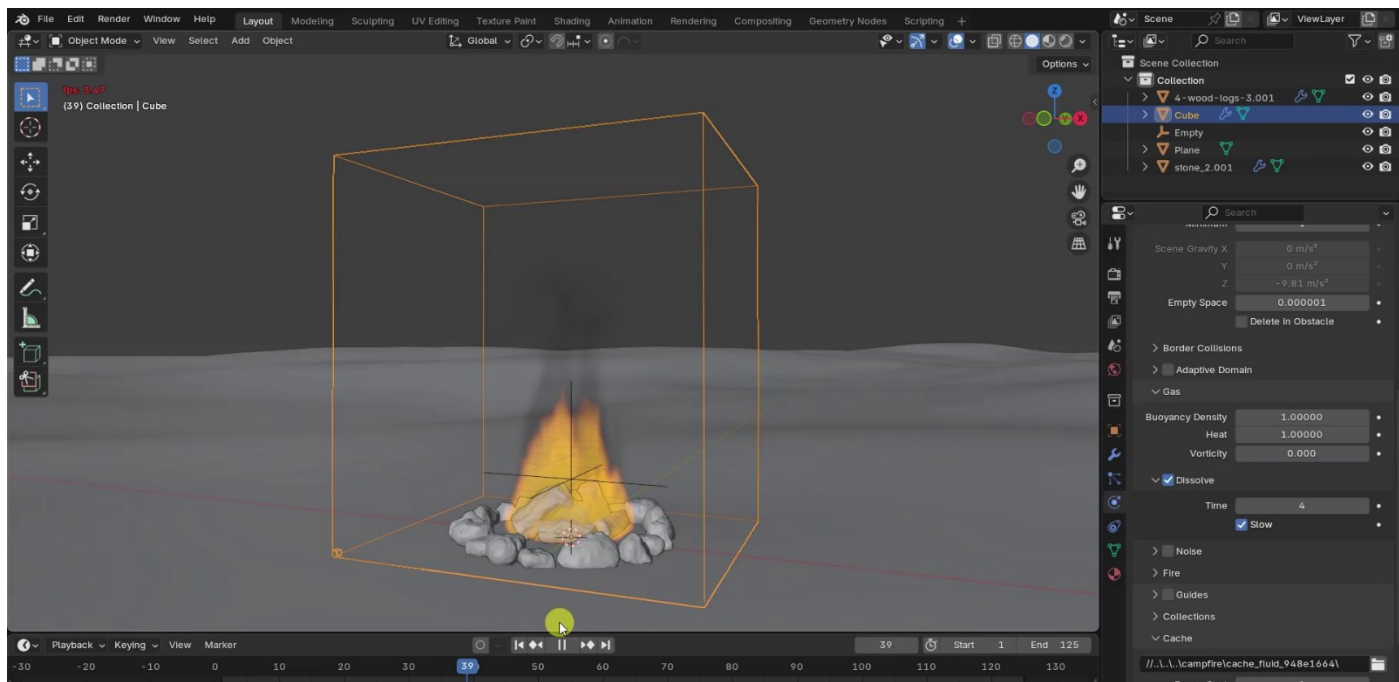
低い値でも非常に強い乱流を生み出すことがあるからです。

画面では Buoyancy Density を変化させると結果がどのように変わるかも示しています。

Buoyancy Density が大きいと煙の上昇が速くなります。

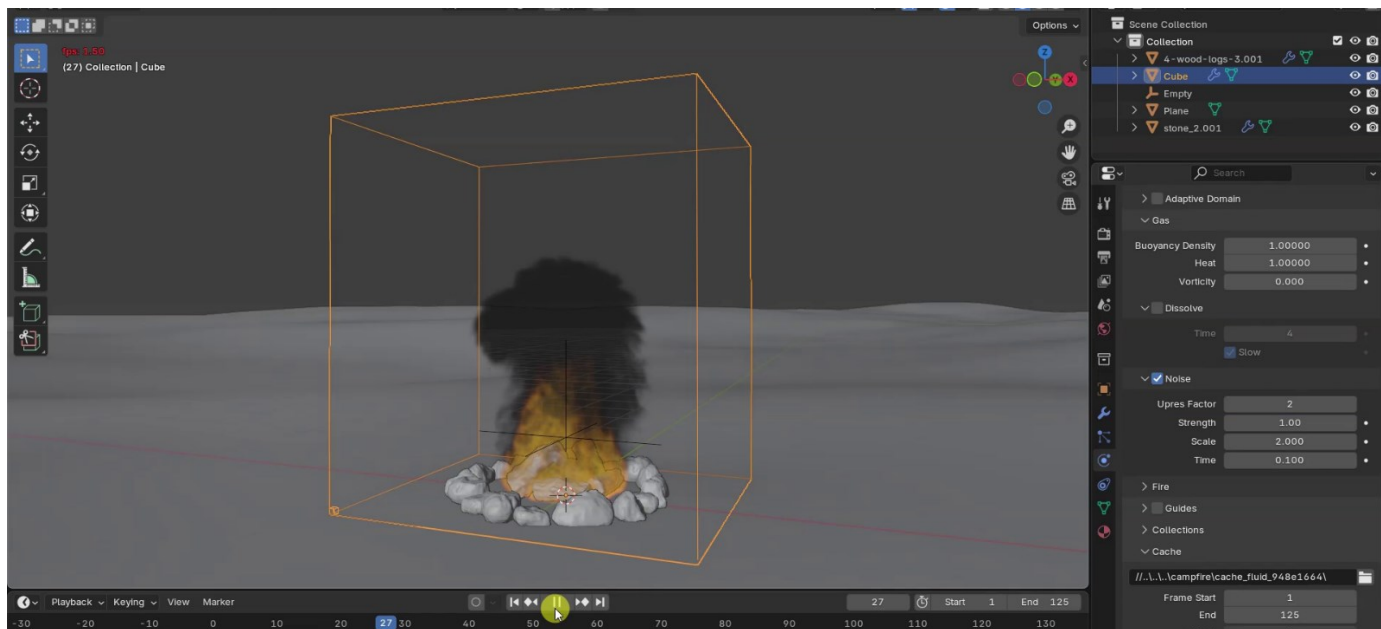


ガスが天井に蓄積するのを防ぎ、一般的により早く消えるようにするには、Dissolveを有効にして Dissolve Time パラメータを調整します。これはフレーム数で測られます。当然、値を小さくすればするほど煙は早く消えます。表示されるフレーム数が減るからです。Dissolve Slow オプションを有効にすると、煙が対数的に消えていきます。つまり最初は速く、時間が経つにつれてゆっくりと消えるようになります。



Gas セクションには Noise と Fire も含まれており、炎に乱れを加えることができます。特に Noise は炎の輪郭に強く影響を与え、多くのディテールを追加するので、たとえば焚き火や暖炉の炎をより乱れて均一でない見た目にするのに便利です。Strength と Scale パラメータは直感的に乱れの強さを制御します。

Upres Factor パラメータは、Domain のベース解像度に追加の細分化を適用してノイズのディテールを作り出すもので、このためシミュレーションは遅くなります。ノイズが加えられる領域は高い解像度で計算されるからです。Fire セクションのパラメータを見ると、Gas セクションと同じように Vorticity パラメータがあることに気づきます。煙と同じように、Vorticity は炎に乱流を与えます。炎にも温度があり、値を高くすれば煙と同様に炎も速く上昇します。この2つのパラメータを組み合わせることで、炎はより高く、より生き生きとした見た目になります。



Collections セクションには Flow と Effector という2つのフィールドがあり、それぞれに Collection を指定できます。これにより、指定された Collection に属する Flow オブジェクトや Effector オブジェクトだけを simulation に含めることができます。

Effector は fire と smoke の経路をそらすことのできるオブジェクトを定義する特別な要素です。

Effector の種類である Collider と Guide については、後のエピソードで解説します。

デフォルトでは Collection は指定されていないため、すべての Flow オブジェクトと Effector オブジェクトが simulation に含まれます。

Fire Simulations in Blender 4.5 basics | 6/10: Domain Settings 2/2 (Weights, Data, Render, Display)

<https://youtu.be/eG3TYUuH6l4>

皆さん、こんにちは！このシリーズの第6回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の simulation において、Domain オブジェクトの設定を仕上げていきます。

具体的には、Field Weights、Volumetric Data、Viewport Display、Render セクションの主要なパラメータを見ていきます。

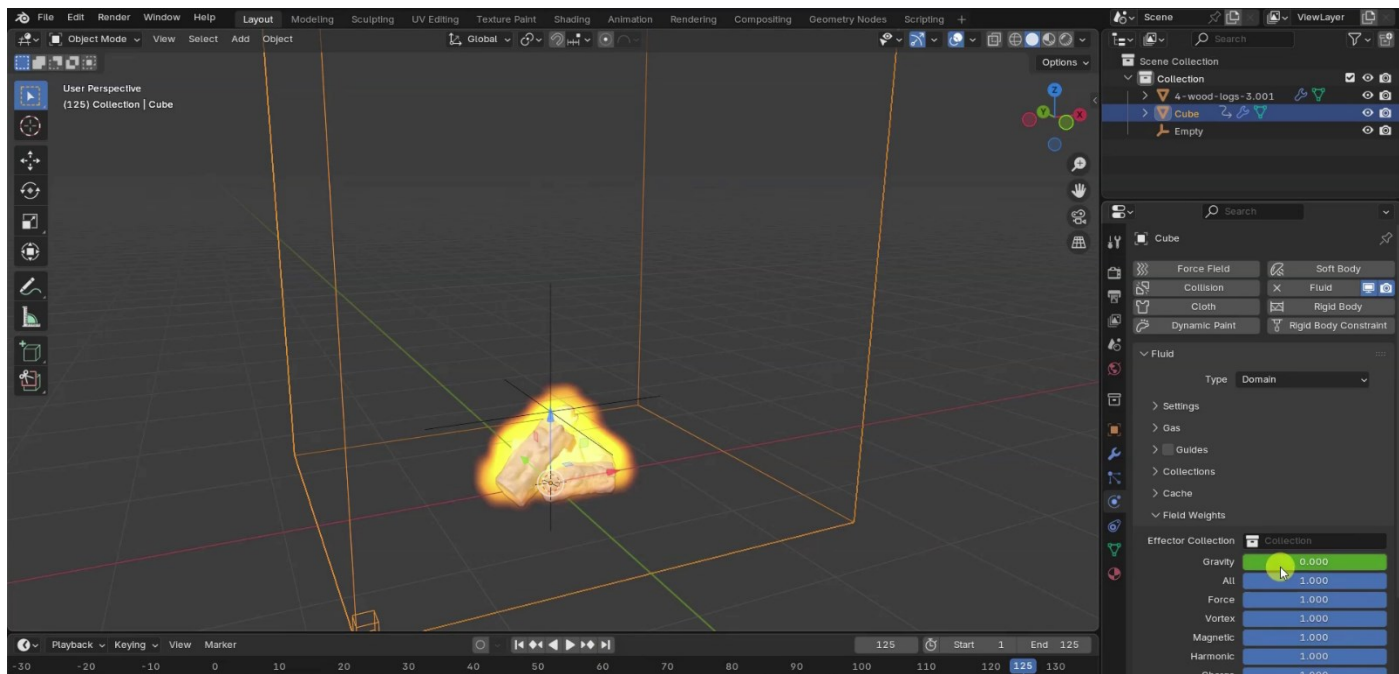
これらの一部は、次回扱う Attribute と呼ばれる Material の性質を理解する助けにもなります。

Field Weights については、simulation が外部の力の影響を受けることができる、ということをまず知っておけば十分です。

ここでは、それら外部の力が simulation 要素にどの程度影響を与えるかを定義できます。

その外部の力のひとつが Gravity で、これはシーンレベルでデフォルトで有効になっており、fire と smoke が上昇する原因になっています。Field Weights セクションの Gravity 値を調整することで、こうしたフィールドがどのように働くかを確認できます。

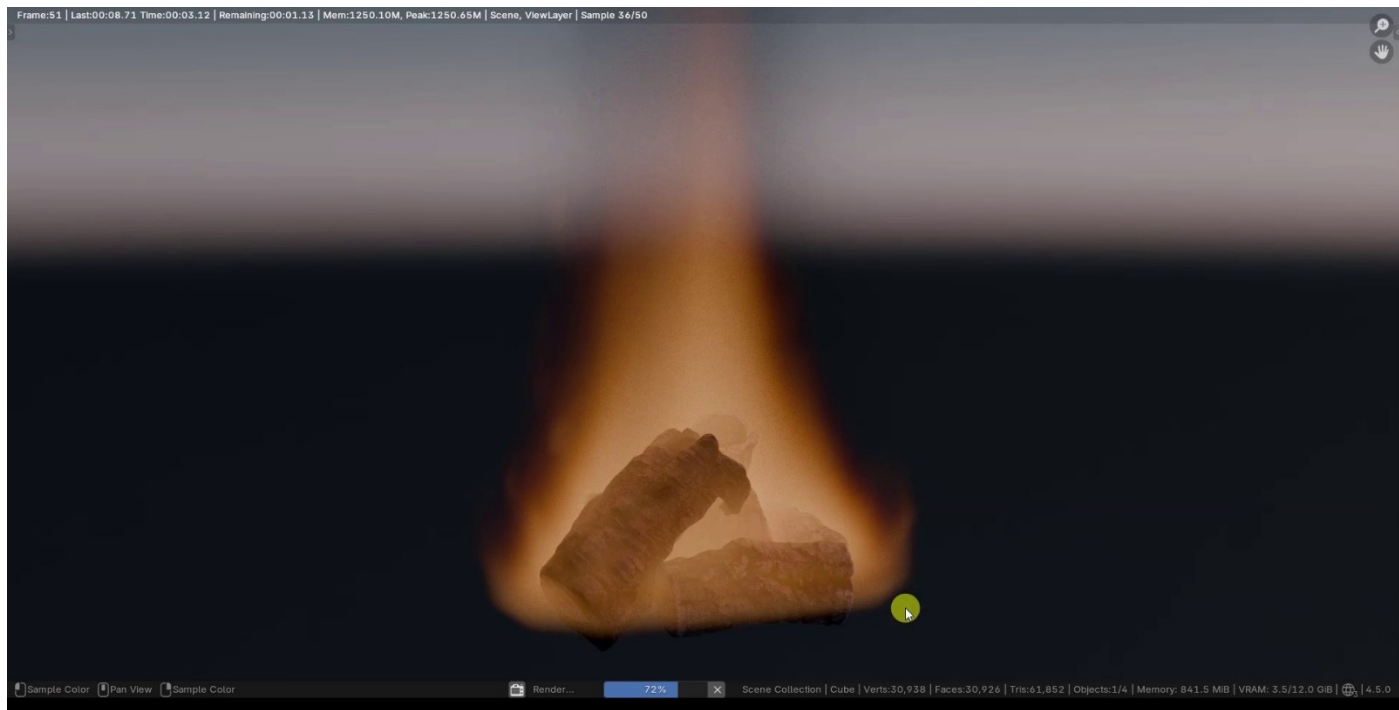
さらに Field Weights のパラメータはアニメーション可能なので、力そのものの設定を変えなくても、このパネルからアニメーション中に影響力を変更することができます。



また All というコントロールもあり、これは simulation に影響を与えるすべての force field に同じ値を適用します。Effector Collection フィールドでは、指定された Collection に属する Effector のみを simulation に含めることができます。デフォルトでは空なので、Domain 内のすべての Effector が simulation に影響します。いずれにせよ、force field と Effector については後のエピソードで詳しく解説するので、ここではこのセクションを閉じます。

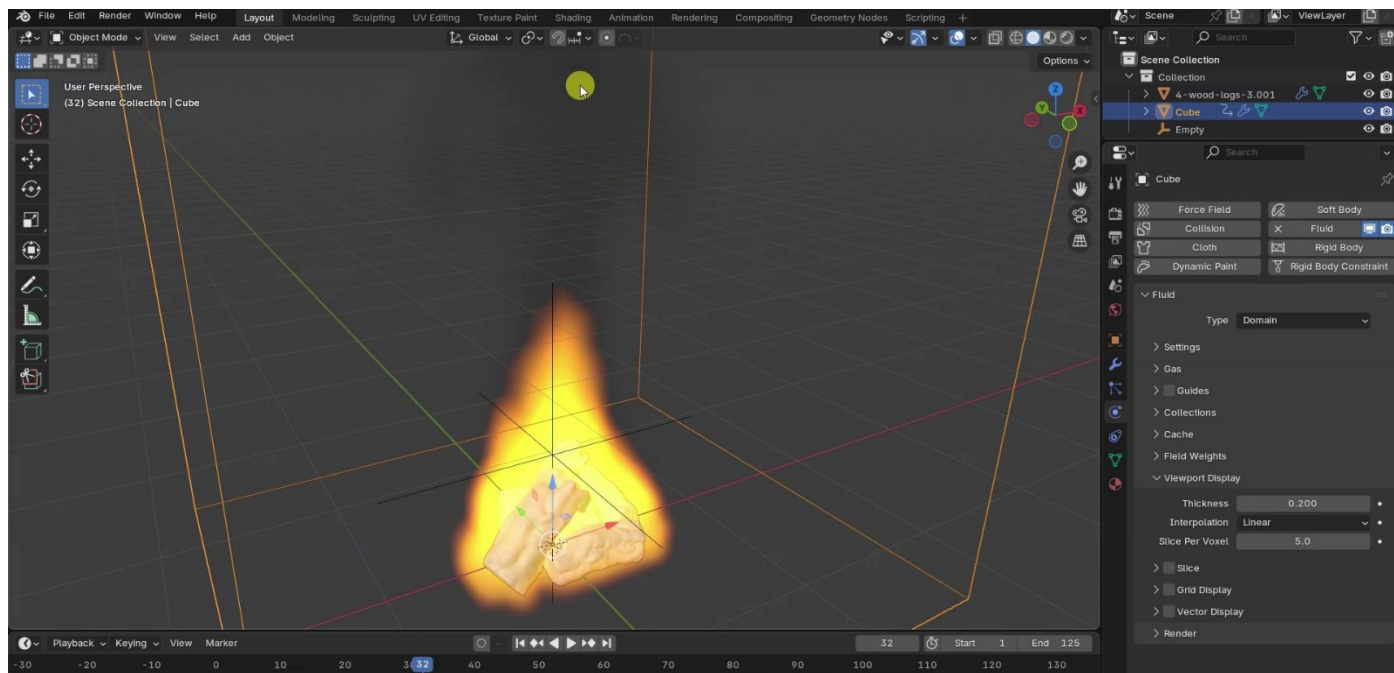
以前 cache について話したときは、どんな情報が保存され、どう再利用できるかに注目しましたが、保存形式については触れませんでした。その設定は Cache パネルの Volumetric Data セクションにあり、ディスク上に Voxel データを保存する際のフォーマットや圧縮方法、精度を指定できます。ディスク保存で最も圧縮率が高いのは Zip ですが、デフォルトはマルチスレッド対応で高速な Blosc です。Half precision は 16 ビットでデータを保存し、Full precision は 32 ビットで保存しますが、場合によっては過剰になることもあります。デフォルト設定は Open VDB、Blosc、Half で、多くの場合この設定で問題ありません。

Render セクションには実際にはひとつのパラメータしかなく、それは Motion Blur に関するものです。Motion Blur は Render タブでプロジェクトレベルで有効化する必要があり、後のレンダリングチュートリアルで見るように、多くの場合とても美しい効果を得ることができます。

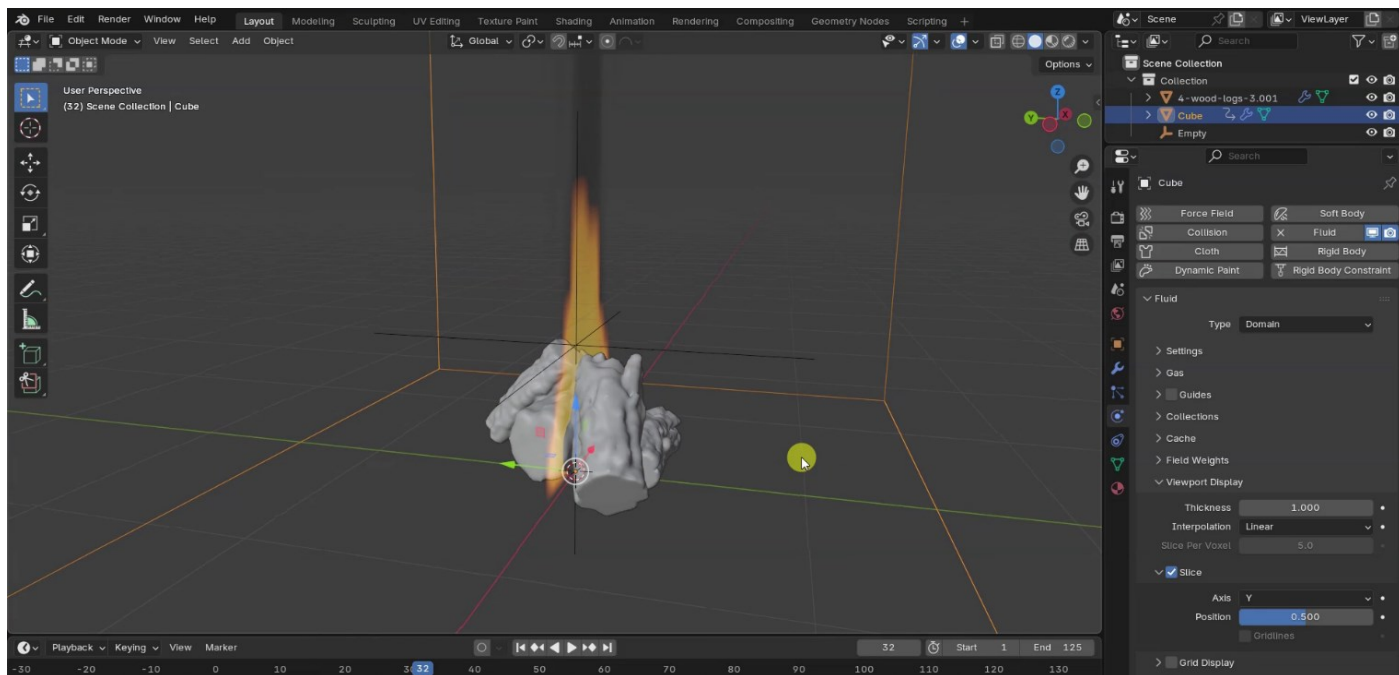


しかし、このチュートリアルのコアは Viewport Display セクションにあります。ここには個々の Voxel に含まれる情報を可視化できる便利なオプションが多くあり、次回のエピソードで fire と smoke の Material を変化させるために使用する Domain Attribute を理解する助けとなります。

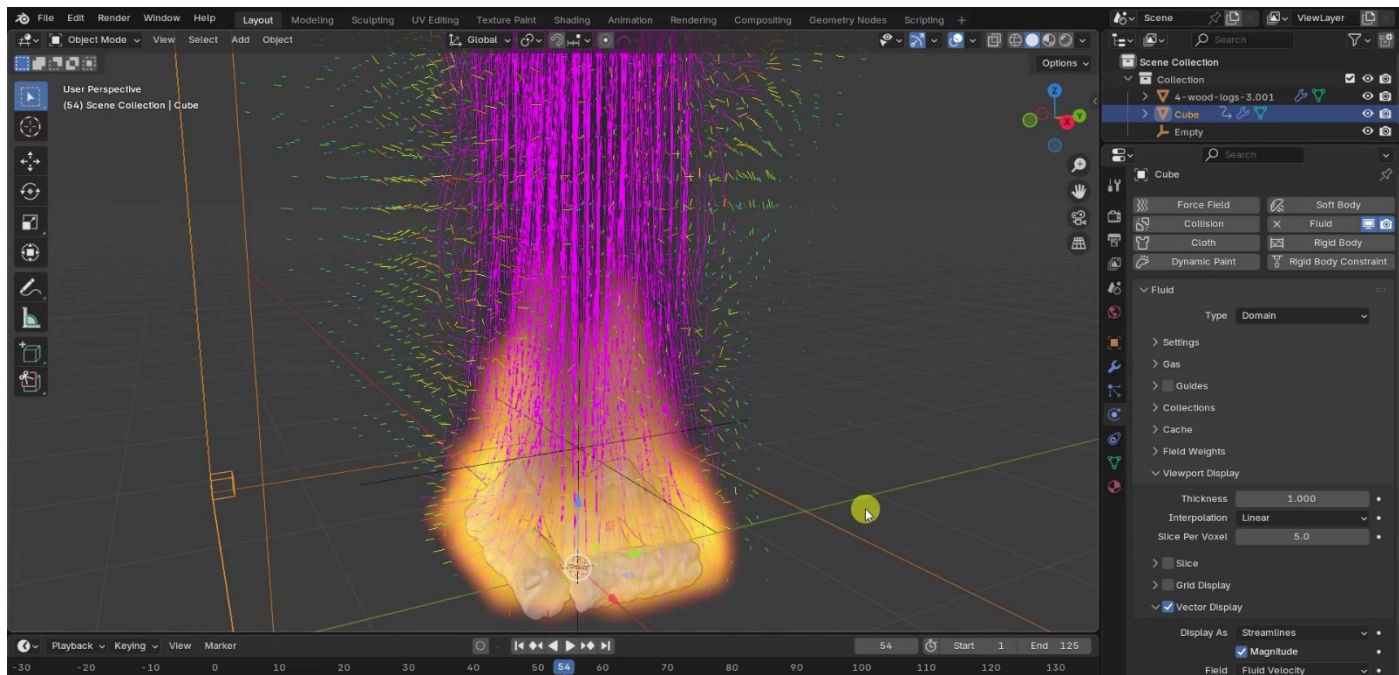
まず知っておくべきことは、3D View では各 Voxel に対して複数のスライスが表示されているということです。ここでは煙の厚さ、補間品質、Voxel ごとのスライス数などのパラメータを指定できます。煙の厚さを増やすと見えやすくなり、その挙動を観察しやすくなります。ただし、このセクションのパラメータは Solid モードの 3D Viewport での見え方にのみ影響し、最終的なレンダリングには影響しないことに注意してください。これは純粋に可視化のためのものです。



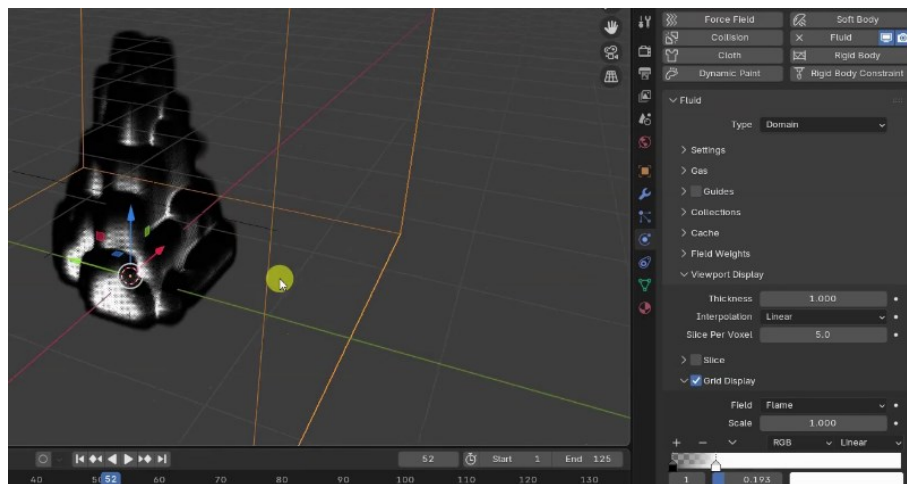
Domain 全体の単一スライスを見たい場合は、Slice オプションを有効にして軸を選択できます。これは Domain をその軸に沿って切断し、断面を表示するようなものです。最初は 3D シーンでカメラを動かすたびに表示が変わります。これはスライス軸がデフォルトで Auto に設定されており、視点に応じて自動的に適応するためです。Position の値は 0 から 1 の範囲で、選択した軸に沿った Domain の大きさに比例します。ここでは Y 軸を使い、Position 値を調整した例をお見せします。



Vector Display では、各 Voxel 内に Velocity、Guides、Forces のベクトルが表示されます。Guides は Force Field と同様に fire や smoke の挙動に影響を与える特別なオブジェクトです。現時点ではシーンに Guides や Forces がないので、流体そのものに固有の Velocity の表示しか見られません。このセクションの他のオプションでは、可視化に使われるグラフィック要素の見た目やサイズを調整して、より読みやすくすることができます。



最後に Grid Display セクションを残しておきました。これは最も興味深いと思うからです。ここではシミュレーションの特性、例えば各 Voxel の熱や密度を可視化できます。ご存じのとおり、Domain は Voxel に分割されており、その解像度は Grid Display に表示されるキューブの分割で明確に見えます。Domain には各 Voxel ごとに温度や密度といった情報が保存されています。Color Ramp を調整することで、これらの特性の値の範囲を全 Voxel にわたって視覚化できます。この種の可視化は特に有用です。なぜなら、これらの特性の一部は Shading セクションで利用できる Attribute として保存されているからです。次回は、それらを Shading で取り出し、例えば温度に基づいて fire や smoke の Voxel の Material を変化させる方法を見ていきます！



Fire Simulations in Blender 4.5 basics | 7/10: Emission, Attributes, Volume Info

<https://youtu.be/AuMfDCnZvwc>

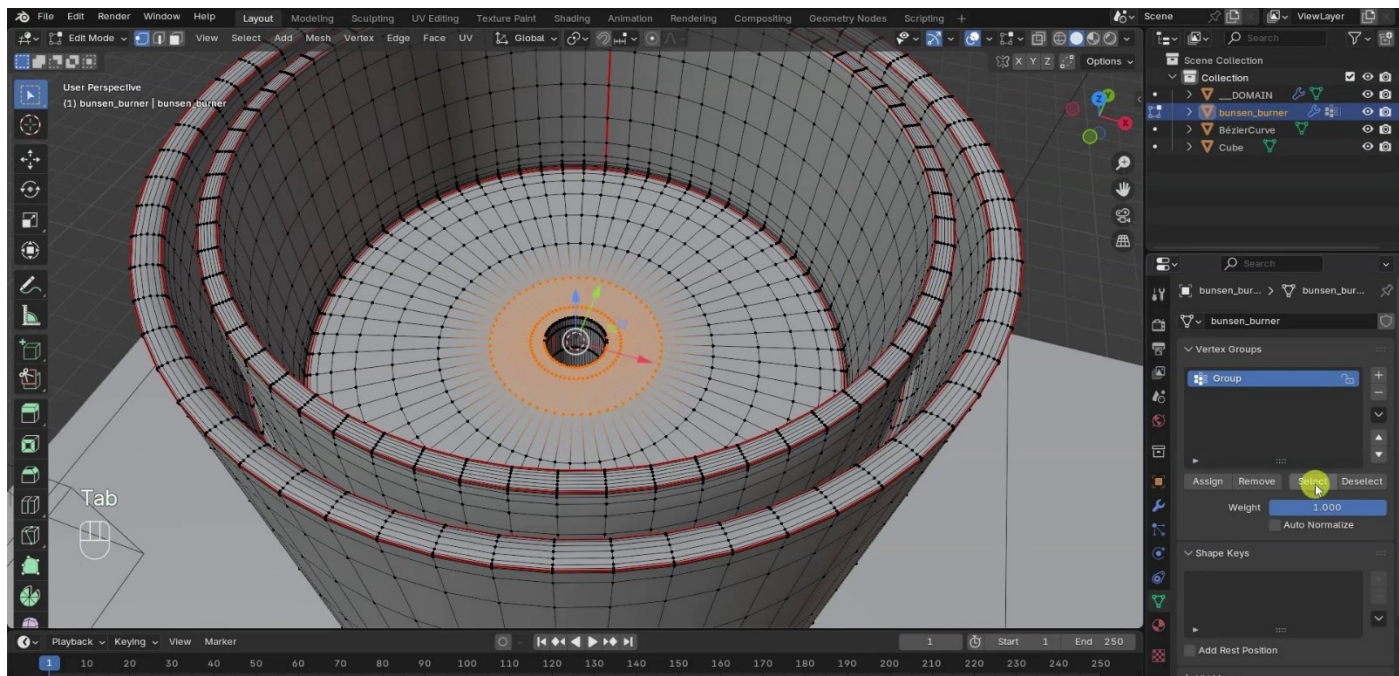
皆さん、こんにちは！このシリーズの第7回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の simulation において、前回解説した Voxel Attribute を利用し、レンダリング用の fire と smoke の Material を変更する方法を見ていきます。

今回の最初の例で使うシーンでは、fire の emission は Inflow Fire に設定した Fluid コンポーネントを持つオブジェクトの Vertex Group から発生しています。

simulation の Domain はバーナーの上部だけを囲んでおり、そこで flame が発生します。

Vertex Group からの emission に関して、もし flame が表示されない場合、その原因は Domain の resolution にあるかもしれません。

解像度が低すぎて Voxel が大きい場合、emission 頂点を含む Voxel が空とみなされ、flame を放出できなくなることがあります。その場合は Domain の resolution を上げてみてください。

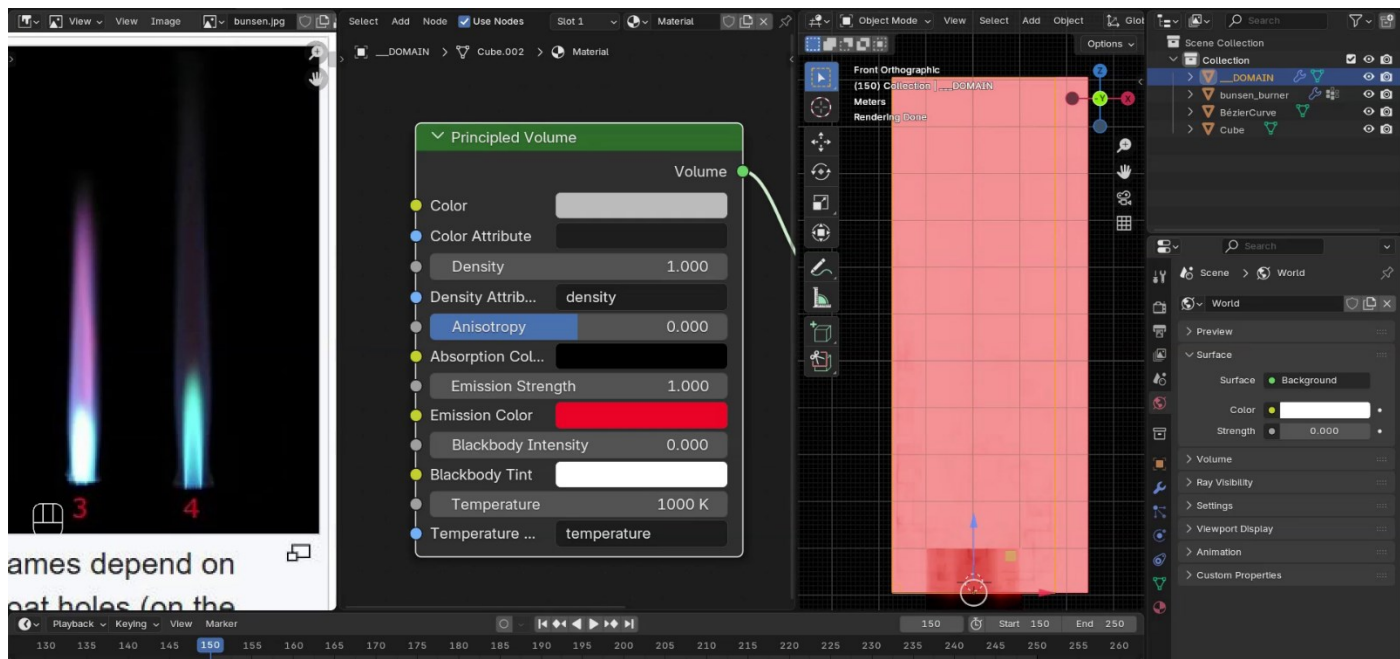


デフォルト設定ではflameの高さは非常に低く、横からはほとんど見えません。しかしflameの見た目を調整する方法はいくつも学んできました。例えば、Inflow オブジェクトの Initial Velocity を操作して炎に上向きの勢いを与えたり、Domainの反応速度を変えて炎を高くしたりできます。今回のバーナーで生成されるflameの種類は乱流や色によって変化します。ほとんど乱流のないflameを得るために、DomainのFireセクションのVorticity値を0に下げました。

Material の定義に進む前の最後の変更として、emission Vertex Group に頂点を追加し、Timeline のアニメーションを simulation の最後の 100 フレームに限定しました。つまり flame がすでに安定してよく発達した状態です。

では、Material の定義に移りましょう！3D 側面ビューを直接 Rendered モードにし、World の Background Strength を 0 に設定したので、最初はプレビューが真っ黒です。左側の Image Editor に参照画像を置き、中央には Domain を選択した Material Editor を配置し、Principled Volume ノードを設定しています。

これまでのチュートリアルで見たように、Emission Strength を上げて Emission Color に均一な色を割り当てても、結果はがっかりするものでした。Blackbody の使い方は学びましたが、ここからはいよいよ Emission モードの使い方を習得していきます。

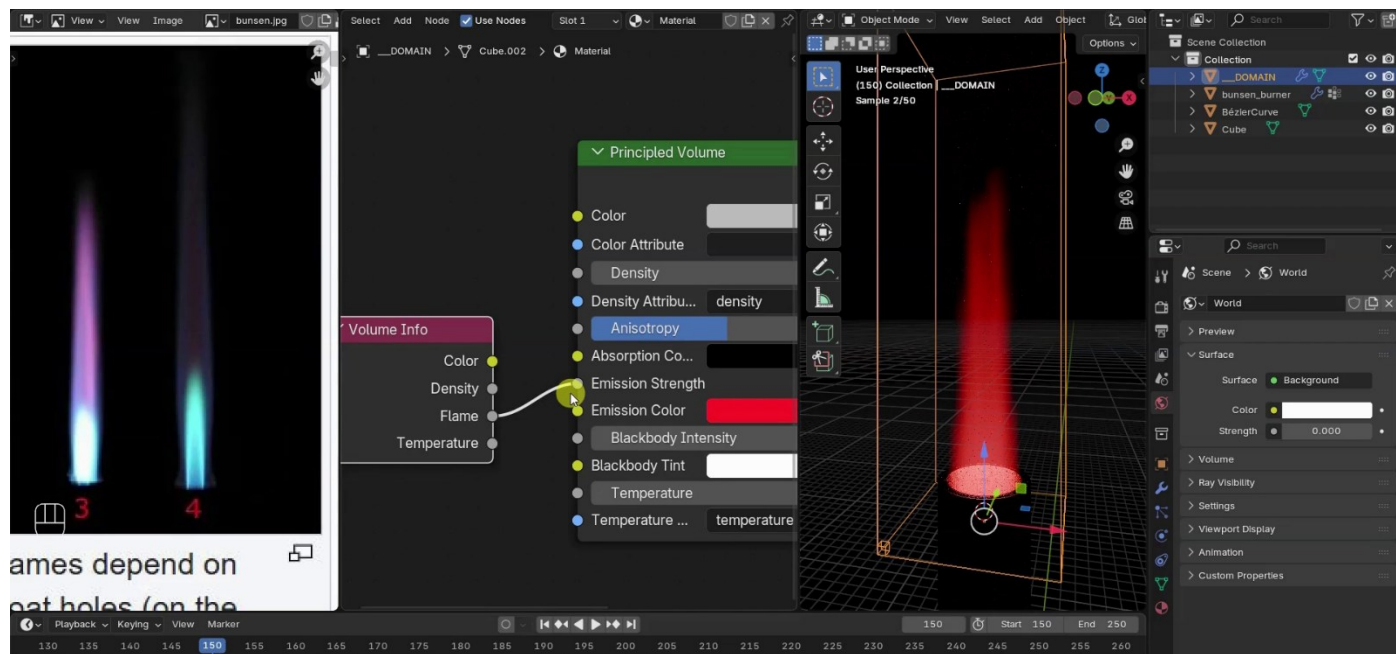


前回のエピソードから分かるように、Fluid シミュレーターは各 Voxel に情報を保存しています。これらのデータは Attribute ノードを使い、フィールドに属性名を入力することで取得できます。

しかし Volumetric Material には Volume Info という専用ノードがあり、これらの属性を出力ソケットとして提供してくれます。

Flame 出力や Temperature 出力を Emission Strength 入力に接続すると、flame の見え方が劇的に変わります！ Flame 出力は Temperature 出力よりも補間された値が良く、Domain の解像度もより明確に反映されます。このため、ここでは Flame を使います。

Flame と Temperature の出力は Emission Color にも利用できます。特に Color Ramp ノードを間に挟むことで、Domain Voxel に含まれる情報に基づいて flame の色をポイントごとに変化させられます。



私は Flame の情報を Color Ramp に接続し、それを Material の Emission Color パラメータに使用しています。ここでの作業は、Color Ramp のストップに色と透明度を設定することです。

そのために Blender の Color Picker を使って参照画像から直接色をサンプリングすることもできます。

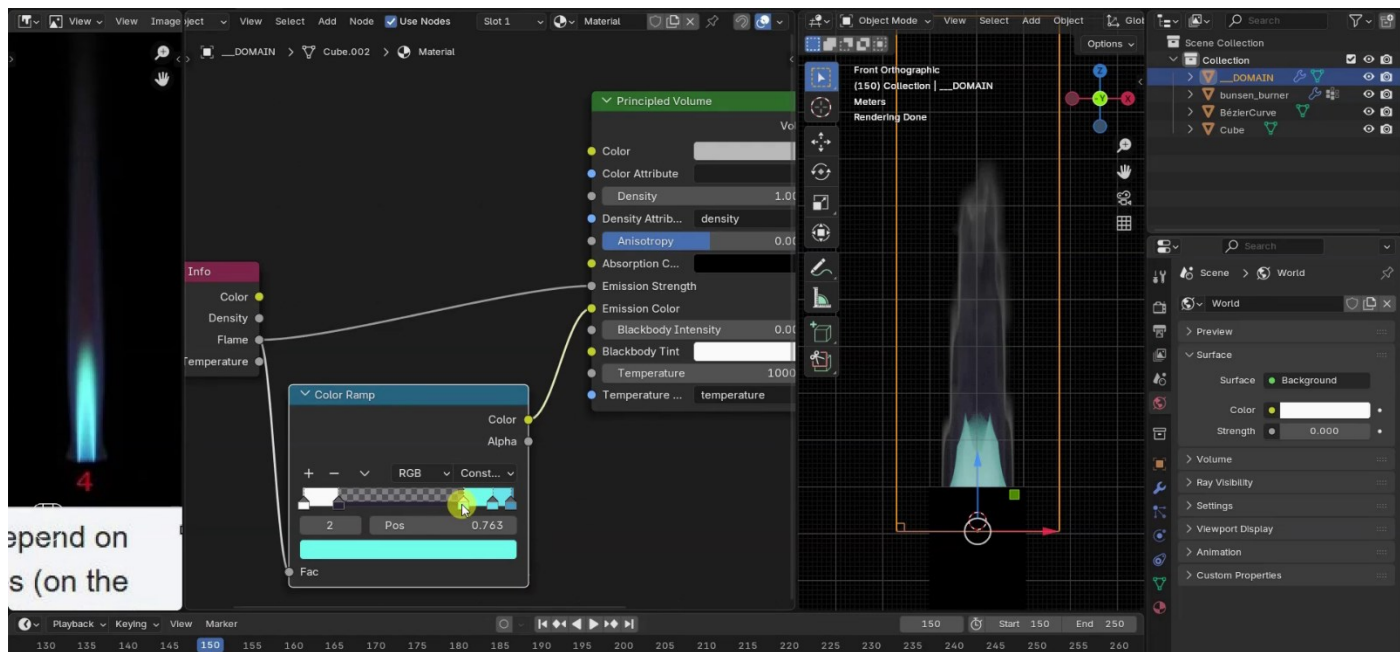
ストップの色には透明度を含めることもでき、これは flame の外側部分で特に有用です。境界より少し内側には半透明の領域があり、その外側がより不透明になるのをよく見かけます。

また、Color Stop 間の補間も慎重に選ぶ必要があります。中でも最も極端なのは Constant で、ひとつの Color Stop から別の Color Stop に突然切り替わります。

現実的ではありませんが、よりアーティスティックな表現には使えます。

今回の Material に関するチュートリアルが短く感じるのは、これまでのチュートリアルで学んだ情報とパラメータを基にしているからです。

この時点ですでに、Domain と Flow オブジェクト両方の Fluid パラメータを通じて flame の形を定義し、さらに Blackbody や Emission と Blender が提供するさまざまなノードを使って flame の見た目を決定できる知識が揃っています。



ここで紹介したセットアップに限定する必要はありません。他の Material ノードも試してみてください。例えば、Math ノードを Multiply や Power に設定して Emission Strength を変更し、Flame から来る情報を加工することができます。さらに、これらのノードのパラメータにはキーフレームを設定できるので、Material を通じて flame の強さをアニメーション化することができます。これなら、simulation の Fluid コンポーネントのフィールドを直接変更する必要はありません。

このチュートリアルを締めくくる前に、ひとつの Domain 内に3つの Inflow オブジェクトを作成し、それぞれ異なる色を持たせた方法をお見せします。結果は完璧ではありませんが、ノードの別の使い方、特に Color Ramp の応用を示す例として取り上げます。

ご存じのとおり、Material は Domain レベルで定義する必要があります。なぜなら、それは Domain の Voxel から生成されるからです。つまり、Domain 内の Inflow オブジェクトに Blackbody ノードを個別に割り当てることはできません。

しかし、Inflow オブジェクトには Smoke Color というフィールドが Inflow パネル内にあります。この色は Volume Info ノードを使って Domain Shader で取得でき、具体的には Color 出力を使用します。



このフィールドを Mix Shader ノードの Factor として使うことで、Voxel ごとに定義された smoke color に応じて異なる Principled Volume ノードを割り当てることができます。

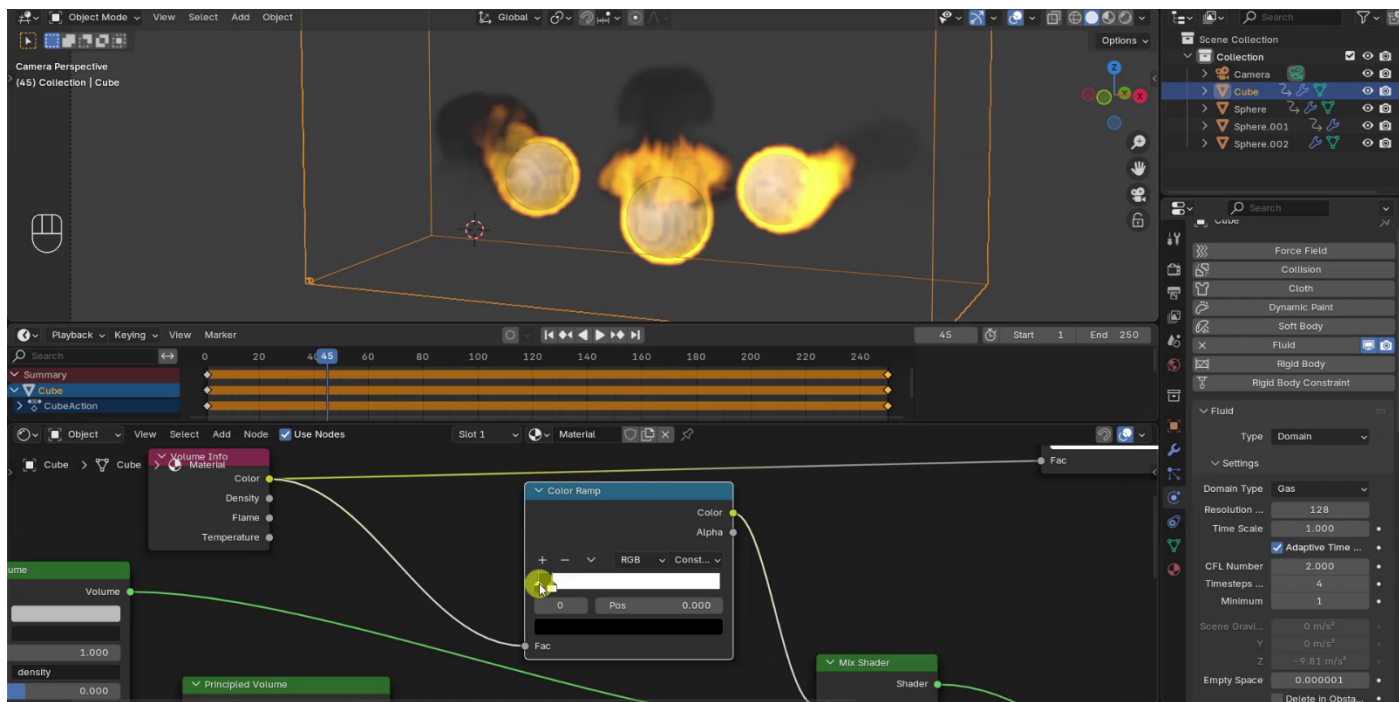
ただし Mix Shader だけでは2つの Shader が単純に 50% ずつ混ざってしまうので、Volume Info から得られる色をはっきり分離するために Color Ramp を使い、その補間方法を Constant に設定します。

ここで、3つの Inflow オブジェクトに白、グレー、黒という3種類の Smoke Color を左から順に割り当てました。最初の Color Ramp では黒とそれ以外の色を分けます。

黒は 0 に対応するので、最初の Mix Shader ノードの1つ目の Shader 入力に送られます。

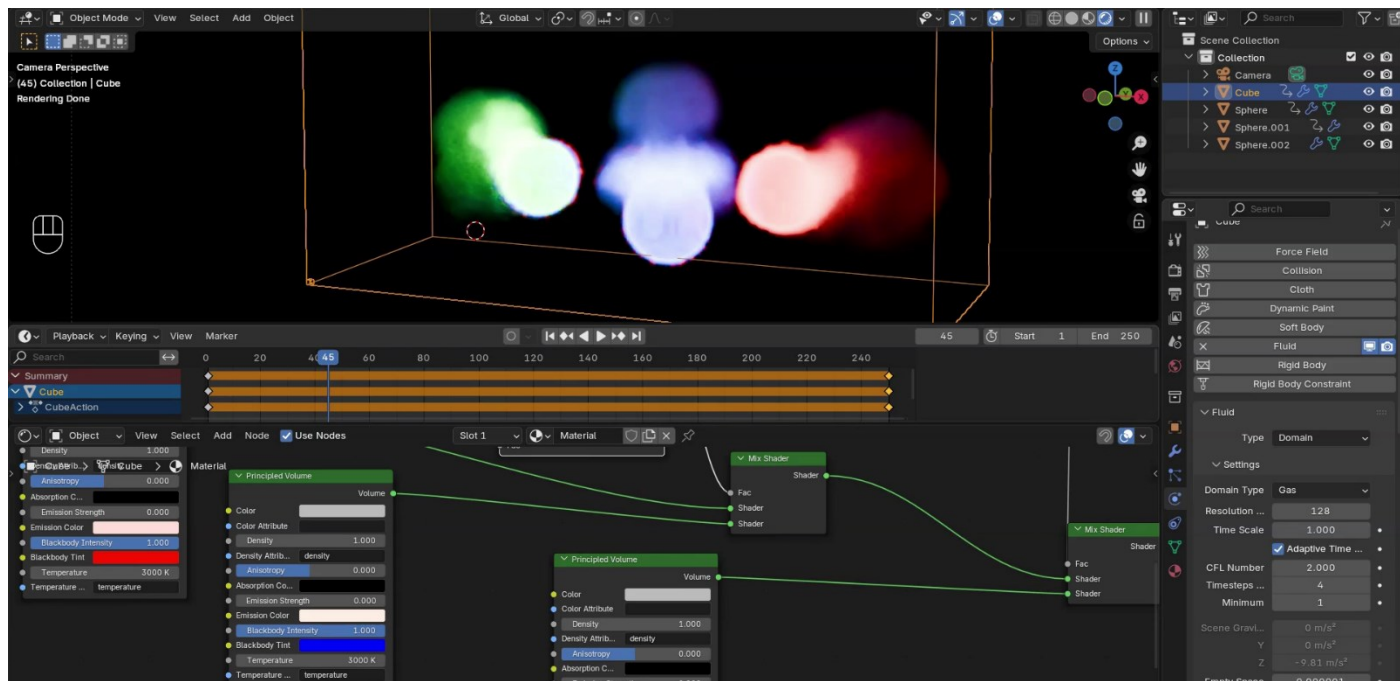
そこには赤い Blackbody Tint を持つ Principled Volume ノードが接続されています。

さらに 3D プレビューで Denoise を有効にし、Rendered ビューで要素の色をよりはっきり確認できるようにしました。



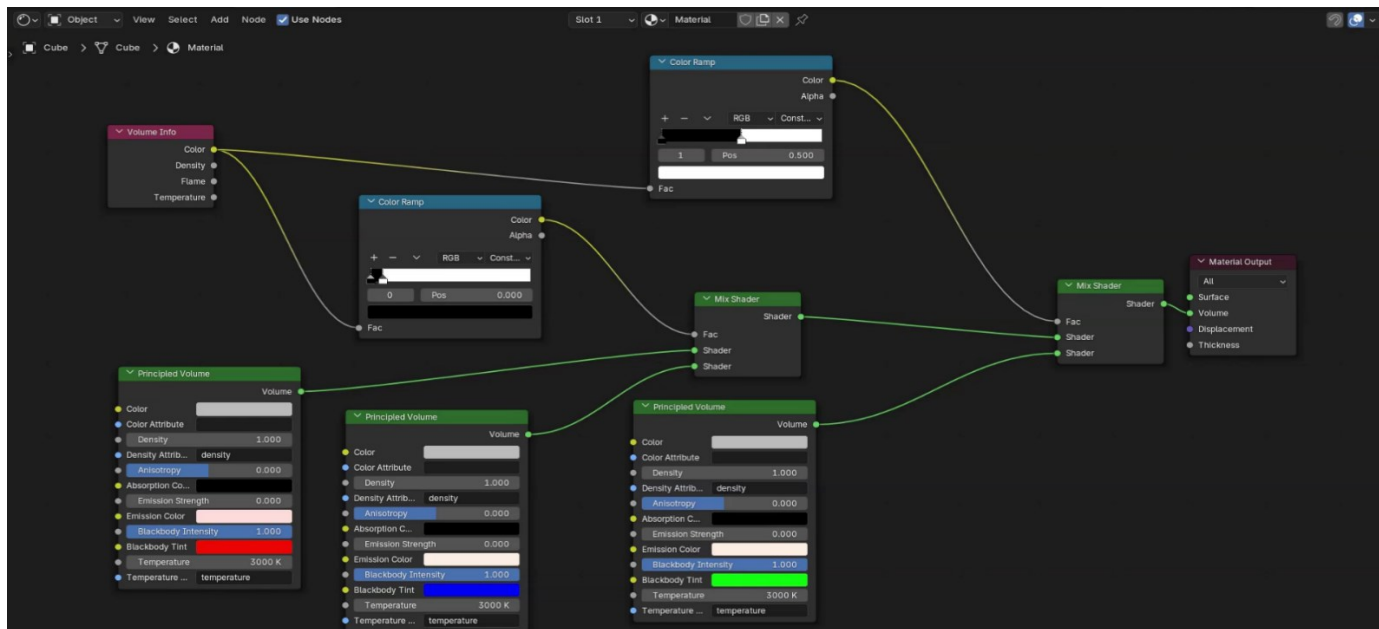
Mix Shader の2つ目の Shader 入力には青い Blackbody Tint を持つ Principled Volume ノードが接続されています。したがって、この時点で smoke color が黒なら赤い Blackbody が表示され、黒以外なら青い Blackbody が表示されます。

次に、もうひとつの Mix Shader ノードを追加します。これは別の Color Ramp を入力として受け取り、これも Constant に設定し、Color Stop をランプの中央に配置します。



この場合、smoke color が黒からグレーの間であれば、2つ目の Mix Shader の1つ目の Shader 入力が使われます。この Shader は先ほどの黒とグレーの区別による結果なので、最終的な色は赤か青になります。

一方、smoke color がグレーから白の間であれば、2つ目の Mix Shader の2つ目の Shader 入力を使用されます。ここには緑の Blackbody Tint を持つ Principled Volume ノードが接続されています。したがって、Smoke Color が白のオブジェクトは緑でレンダリングされます。もちろん、2色や3色の炎の球体を作る方法はこれだけではありませんが、ノードの活用例として紹介しました。前述のとおり分離は完全ではないので、もし改善案や別のセットアップを見つけたら、ぜひコメントで提案してください！



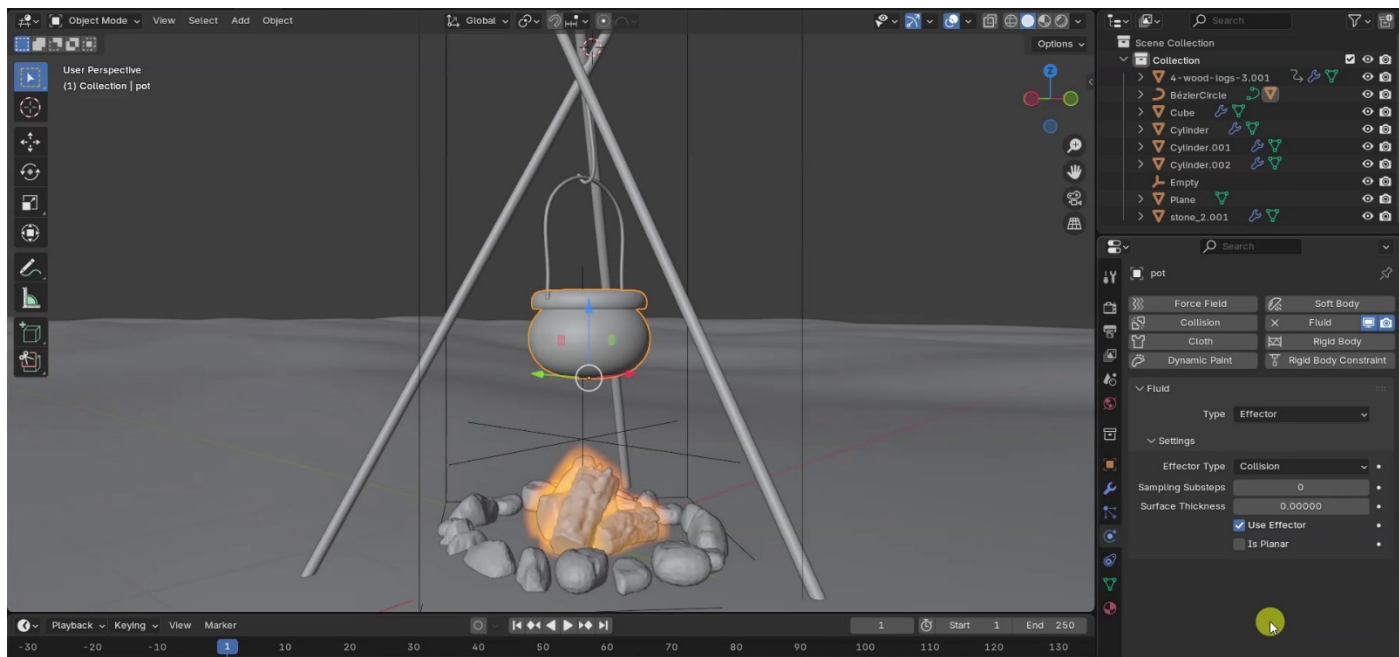
Fire Simulations in Blender 4.5 basics | 8/10: Effectors (Guides, Colliders)

<https://youtu.be/YKyoB-gAuRw>

皆さん、こんにちは！このシリーズの第8回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の simulation において利用できる2種類の Effector オブジェクト、Collision と Guides について見ていきます。

Effector オブジェクトは物理シミュレーションの要素をそらしたり、その流れに影響を与えたりするために使われます。画面に表示しているシーンでは、flame と smoke が鍋をそのまま通り抜けてしまいます。これは、鍋が障害物として検出されていないためです。

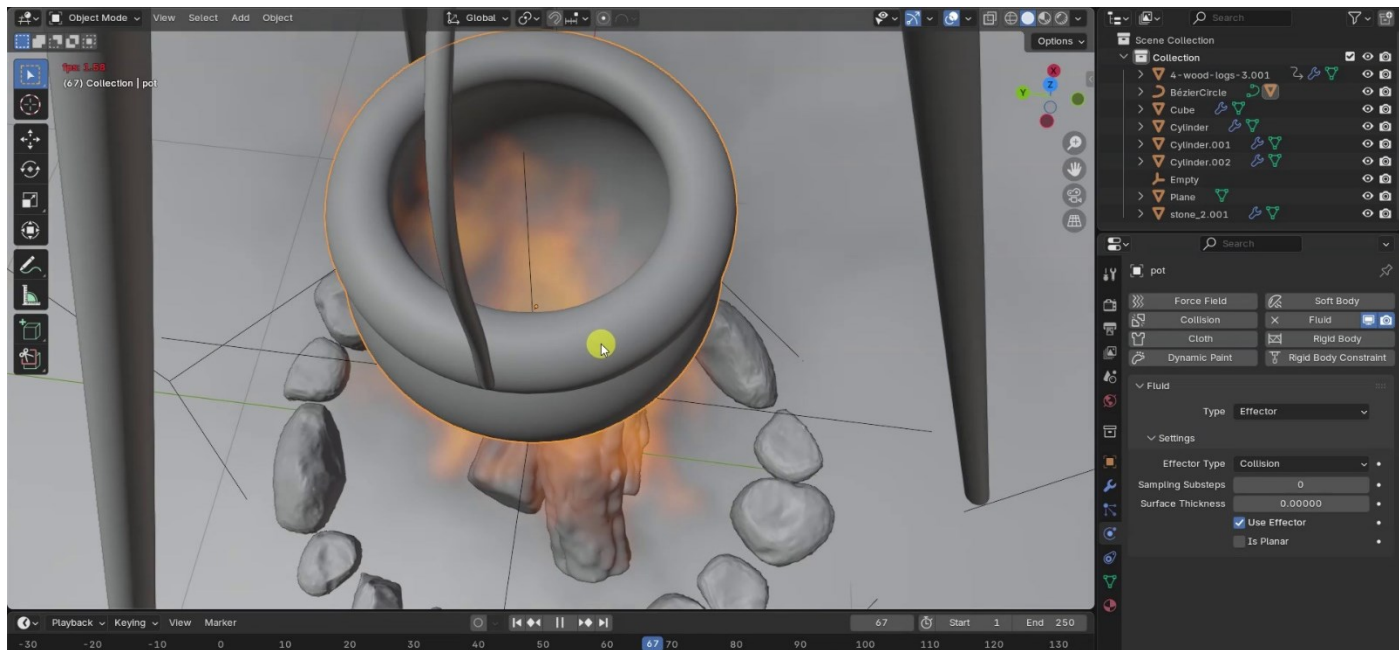
この問題を解決するには、鍋に Fluid コンポーネントを割り当て、タイプを Effector に設定し、専用メニューで Collision を選択する必要があります。



ご覧のとおり、パラメータは非常に少なく、実際にはその多くが Flow オブジェクトで見たものと似ています。例えば Sampling Substeps はフレーム間に追加の計算を行うものでした。また、Use Effector オプションは Use Flow と同様にアニメーション可能で、衝突を任意に有効または無効にできます。さらに Is Planar はボリュームを持たないジオメトリや、内側と外側を定義できない場合に役立ちます。ここで新しいパラ

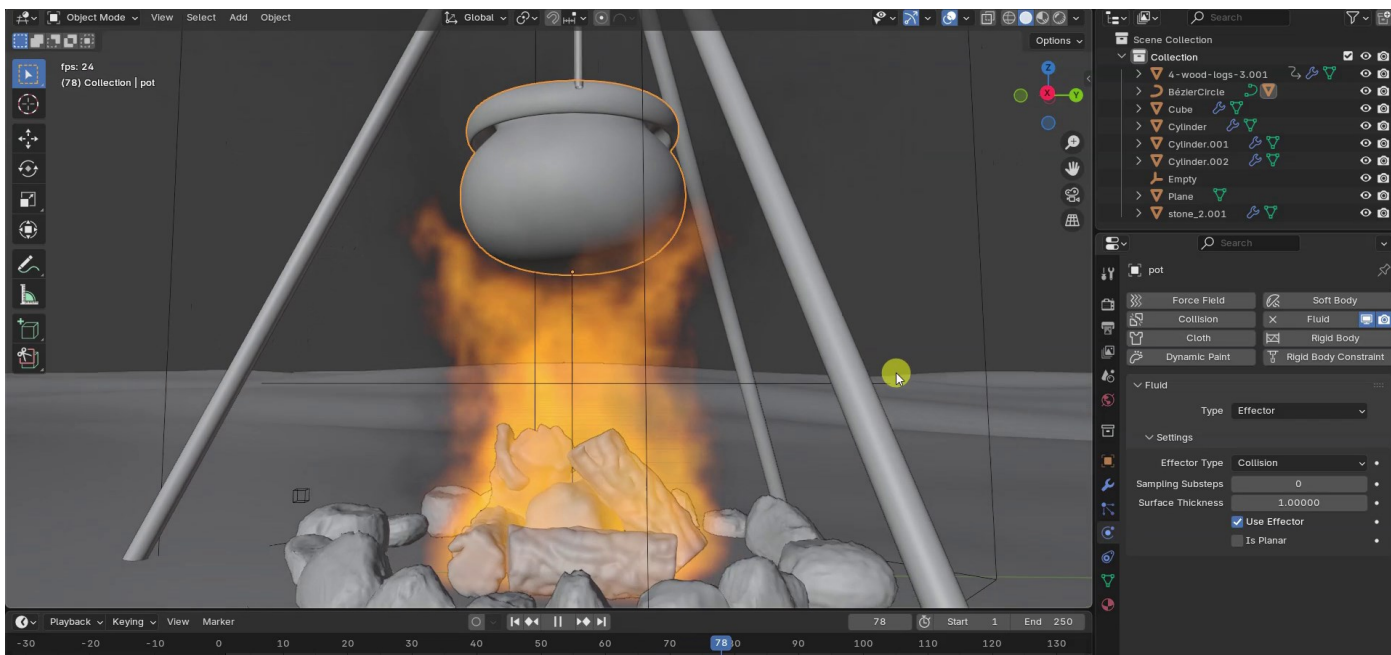
メータは Surface Thickness で、オブジェクトの周囲に外殻のようなを作ります。これは simulation で fire や smoke がオブジェクトの表面に食い込むような問題が起きる場合に便利です。

ただし simulation を再生すると、一部のフレームで flame がまだオブジェクトを通り抜けてしまうことに気づきます。

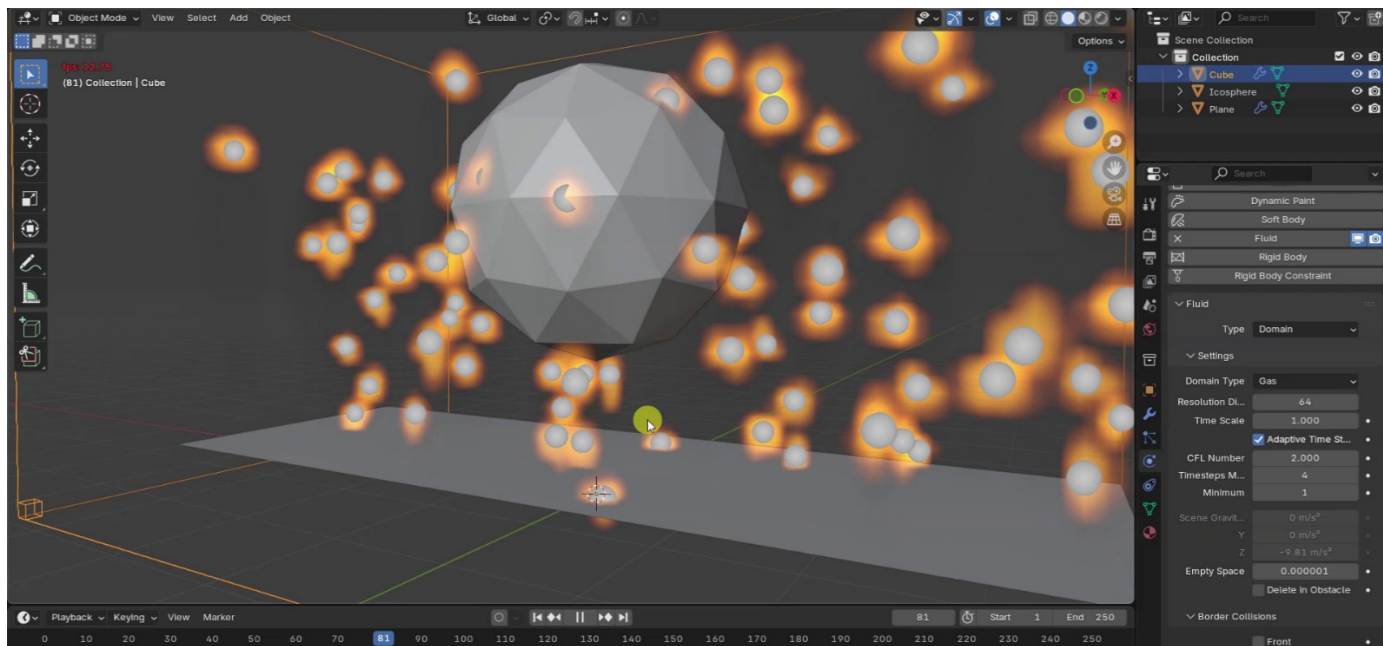


この問題を解決するため、Domain の resolution はそのままにしてテストを繰り返し、Surface Thickness の値を徐々に上げていきました。

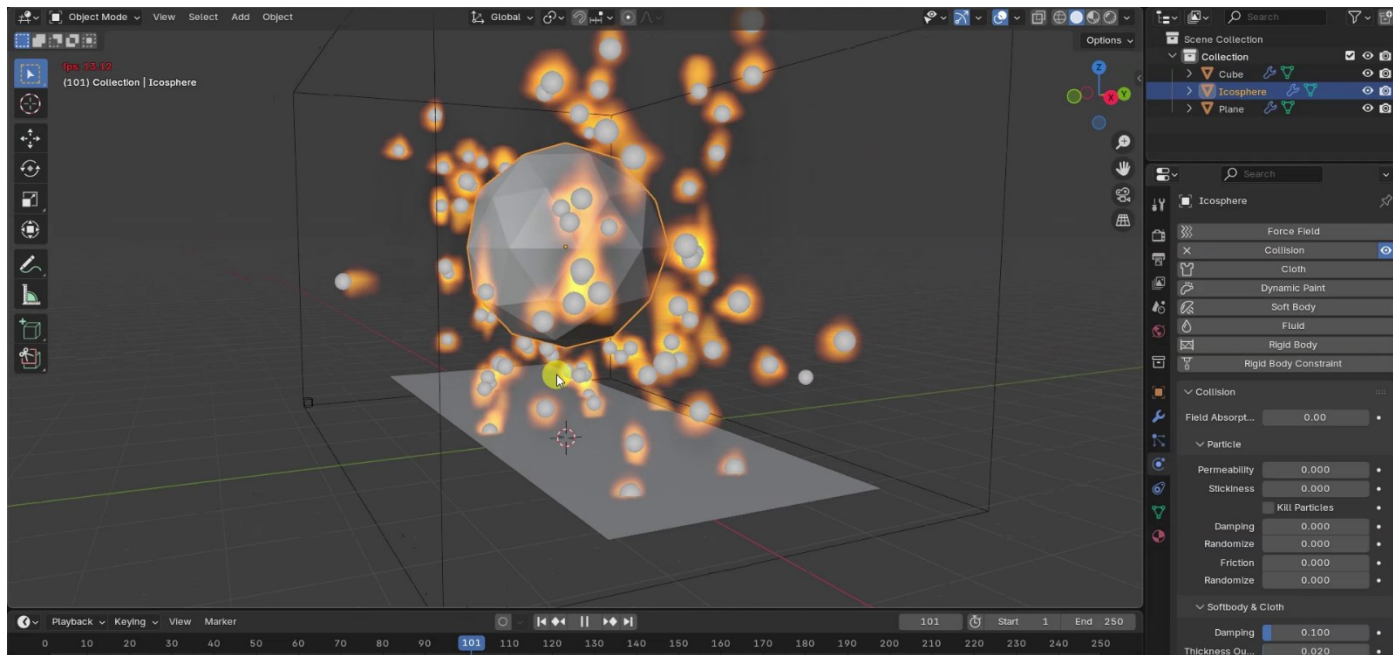
Thickness を 1 に設定したところ、画面に表示している結果が得られました。Substeps 値は変更していません。ツールチップには、この値は主に高速に動く Effector に役立つと説明されているからです。今回のオブジェクトは静止しているので、Thickness だけを調整し、これが最終結果です。



Guide オブジェクトに進む前に、Collision についてもう1点確認しておきましょう。今回のシーンでは、上向きに particle を放出する Plane があります。particle を上昇させるため、particle system 設定で Gravity を無効にしました。この emitter オブジェクトは fire simulation の Inflow オブジェクトでもあります。そこで Flow Source を Particle System に変更し、対応するフィールドに作成した particle system を指定しました。fire particle が system の particle よりも大きくブロック状にならないように Domain の resolution を上げています。ご覧のとおり、particle は Domain 内のオブジェクトと一切相互作用していません。



この場合、fire particle をオブジェクトに当たったときに跳ね返らせるには Collision コンポーネントを追加する必要があります。このコンポーネントは particle を跳ね返しますが、その煙や flame の一部は依然としてオブジェクトの中に入ってしまう。したがって、fire と smoke の simulation を正しく扱うには、Collision Effector タイプの Fluid コンポーネントも追加する必要があります。



今回のエピソードを締めくくるにあたり、Effector オブジェクトの Guide タイプについて見ていきましょう。

これは特別なオブジェクトで、速度を持ち、simulation 内の flame と smoke の動きに影響を与えます。例として、simulation Domain 内に奇妙な軌道を描いて動く sphere を追加しました。この sphere にはまだ Fluid コンポーネントがないので、simulation には何の影響も与えていません。

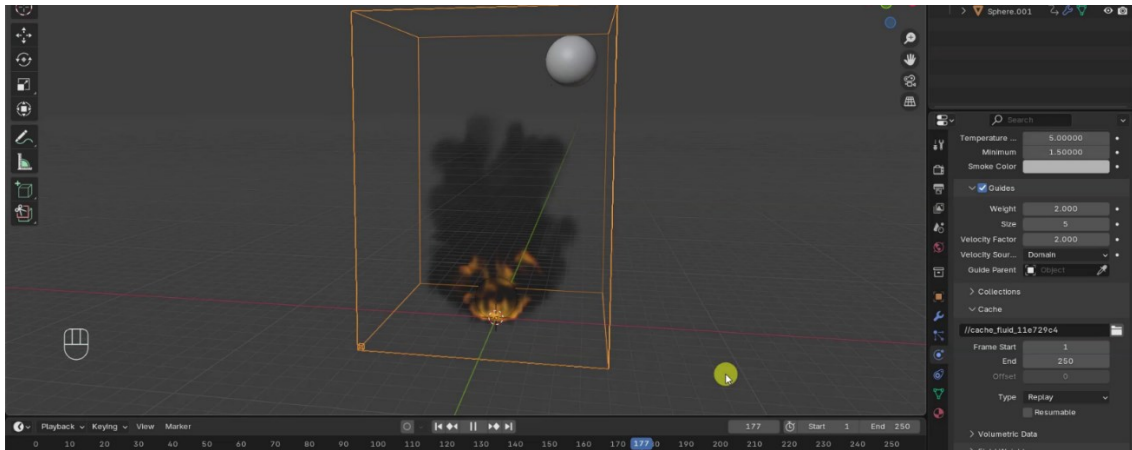
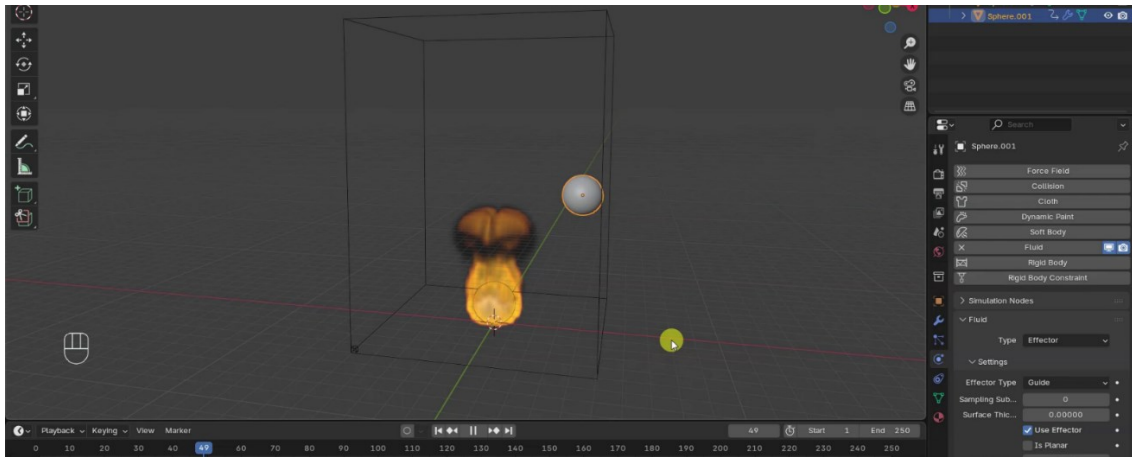
Fluid Effector コンポーネントを追加し、そのタイプを Guide にすると、これまで見たことのあるパラメータに加え、新しいものがいくつか表示されます。

その中でも後ほど特に重要になるのが Velocity Factor です。

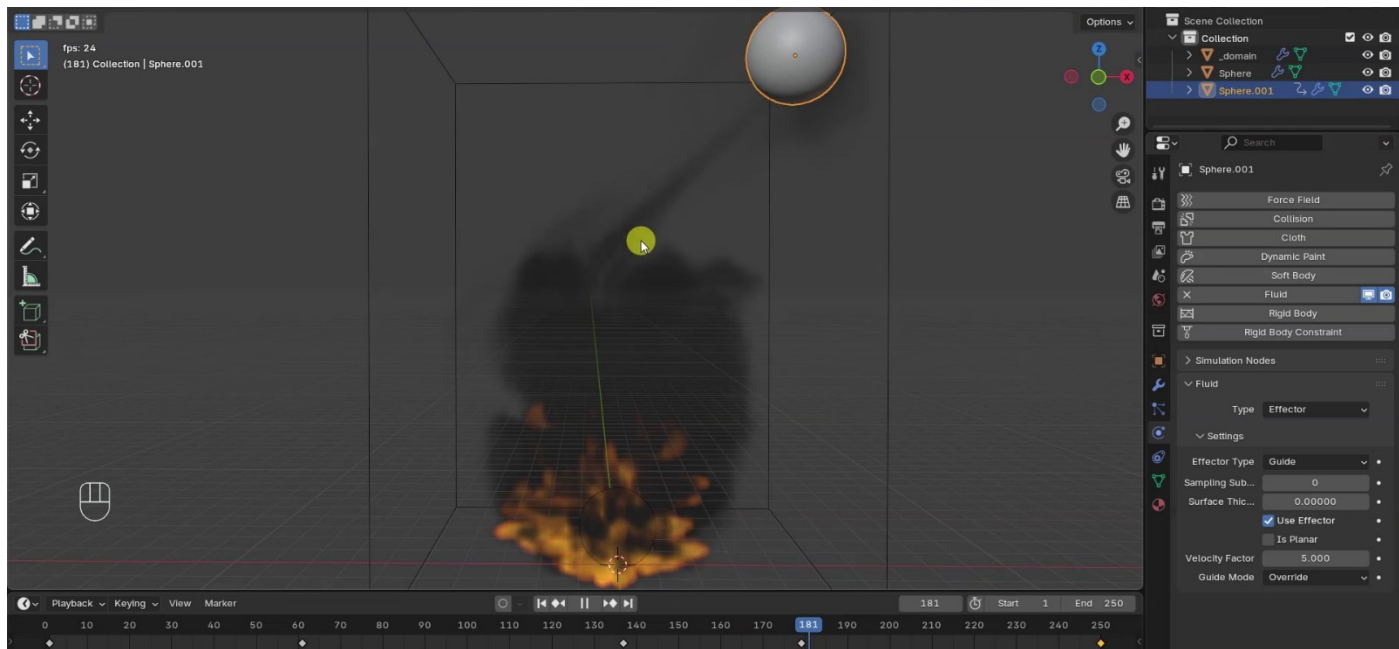
simulation を再生すると、パフォーマンスが低下しますが、オブジェクトが fire や smoke に与える目に見える効果はありません。

これは、計算コストの非常に高い Guides を使用するには、Domain 側でも有効化する必要があるためです。

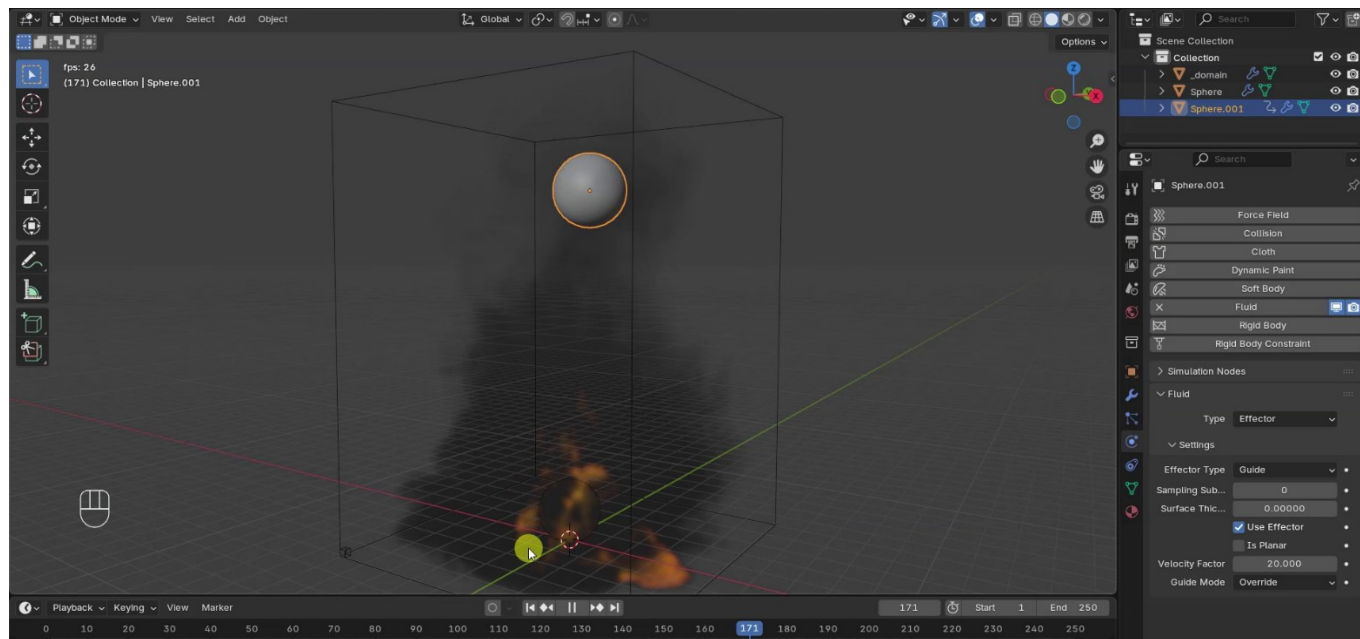
有効化しないと効果が出ません。そこで Domain の Guides セクションを有効にして simulation を再起動しました。計算に時間がかかるので、Blender に処理させ、その結果をお見せします。



結果は Guides を使わなかった場合と比べて明らかに違いがありますが、まだそれほど印象的ではありません。オブジェクトの simulation への影響を強めるために、その Velocity Factor を高い値に設定してテストしてみます。こちらは Velocity Factor を 5 に設定した simulation です。smoke が引きずられていく様子がより明確になりました。



こちらは Velocity Factor を 20 に設定した simulation です。Guide オブジェクトは Collider ではなく、simulation の要素をかき乱し、引きずる特別なタイプであることがよく分かります。



ただし fire と smoke の動きに影響を与える方法はもうひとつあります。それが Force Field の利用で、次回のチュートリアルで解説します。

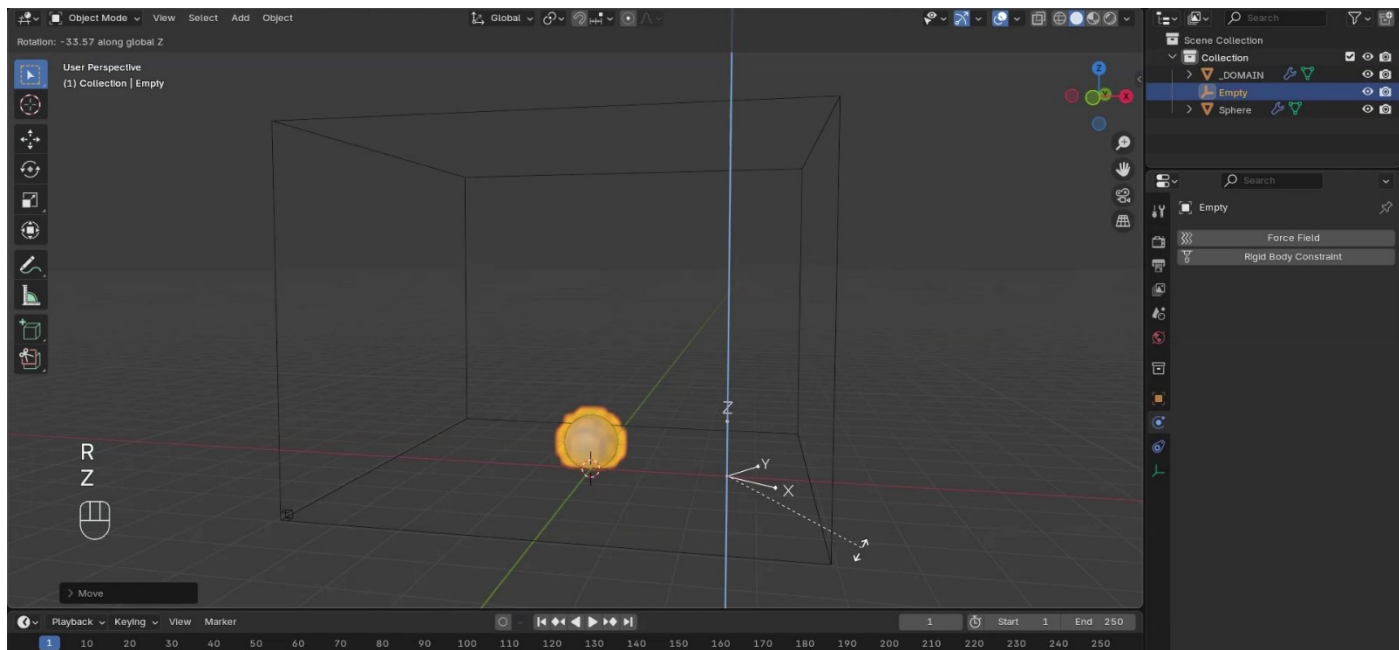
Fire Simulations in Blender 4.5 basics | 9/10: Force Fields (Turbulence, Wind, Curve Guide)

<https://youtu.be/cjX0akeWCc0>

皆さん、こんにちは！このシリーズの第9回チュートリアルでは、Blender 4.5 の Fluid シミュレーターを使った fire と smoke の simulation において、Turbulence や Curve Guide といった Force Field を使い、simulation 要素の挙動や軌道に影響を与える方法を見ていきます。

今回のチュートリアルで使用する開始シーンは非常にシンプルで、Domain と fire と smoke を Inflow モードで放出する sphere だけで構成されています。

シーンに Empty を追加すると、それに Force Field を割り当てて simulation 要素を乱したり誘導したりできます。Force Field は Physics パネルにあり、Mesh オブジェクトの Fluid simulation と同じ場所にあります。ここでは Empty のタイプに Arrows を選びました。一部の Force Field は特定の方向に作用するため、このタイプを使うと視覚的に分かりやすいからです。



その後 Force Field を追加すると、デフォルトでは Force タイプになっています。各 Force Field には、その作用と影響する方向を説明するツールチップが表示されます。Force はデフォルトで Point 形状を持ち、中心から放射状に作用します。しかし simulation を再生しても効果は見られず、Inflow オブジェクトが Force Field の範囲内にあるように見えるにもかかわらず、何も起こりません。

このような場合に有効と思われる手順は次のとおりです。まずプロジェクトファイルを保存し、cache フォルダを指定します。

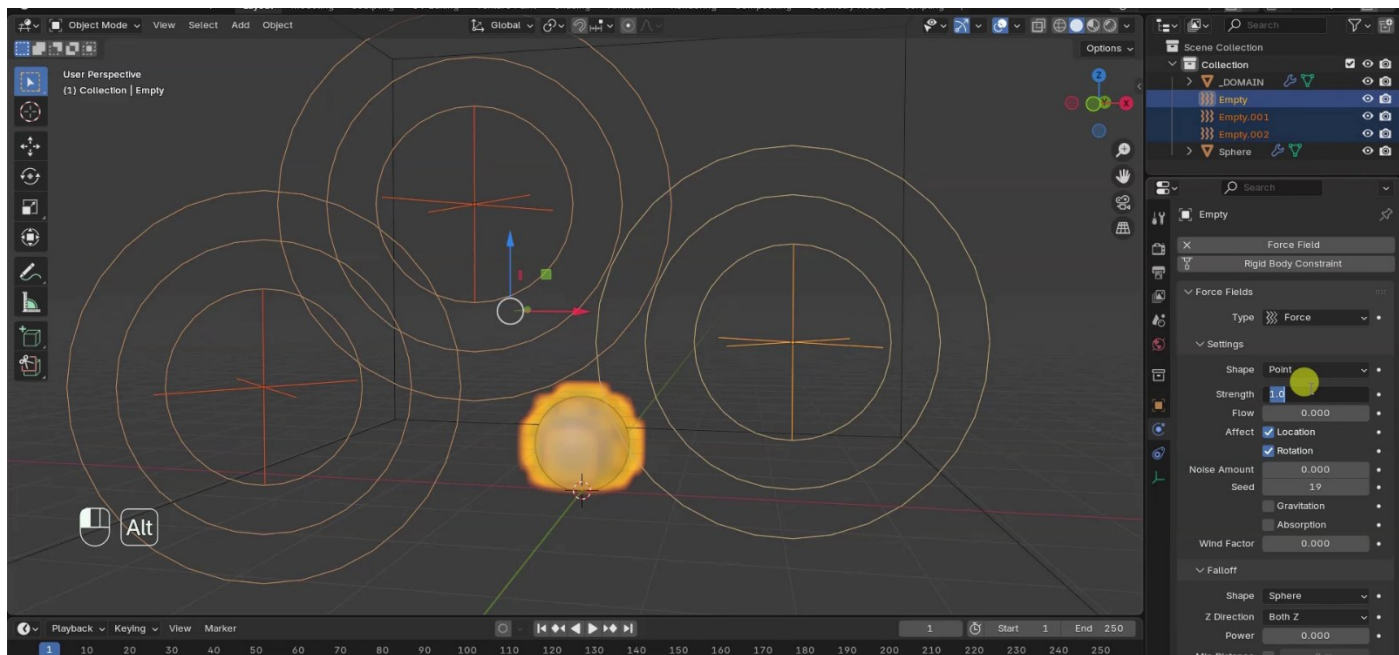
次に Domain の Baking モードを All に切り替え、少なくとも数フレーム分の simulation を Blender に計算させます。

この処理が完了したら Play を押して、simulation が正しく計算されたか確認します。

うまくいっていれば、Domain Cache をクリアし、Baking モードを Replay に戻します。

その後、プロジェクトのアニメーションの長さとして Domain の simulation フレーム範囲を調整し、Timeline の Play をクリックします。simulation が正常に動作するようになったら、以降は Replay モードでさまざまな効果をテストできます。

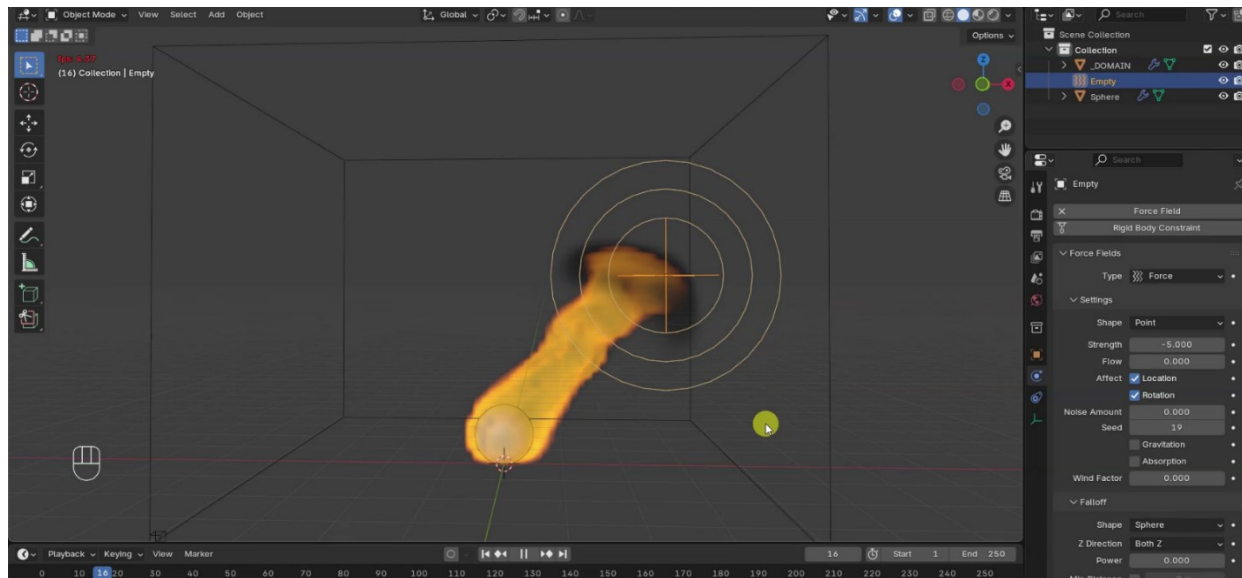
効果の強さを制御するには、Force Field に Strength フィールドが用意されており、これは選択した Force Field に適用され、アニメーションも可能です。補足として、フィールドの値を複数のオブジェクトに同時に変更する場合、ALT キーを押しながらフィールドをクリックして編集すれば可能です。



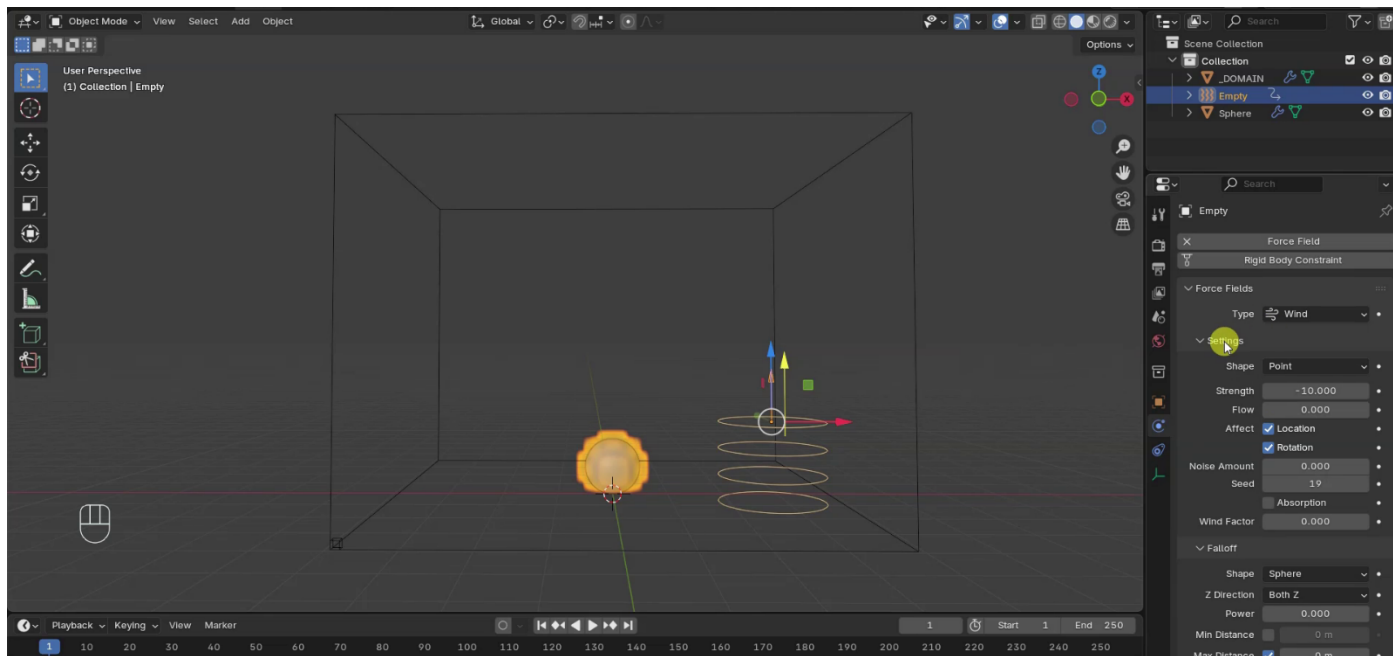
したがって Strength を使えば、個々の Force Field の強さを調整でき、アニメーションで変化させることもできます。しかし Domain に関するチュートリアルで学んだように、シーン内のすべての Force Field、あるいは特定タイプの Force Field の影響力をまとめて制御することも可能です。これらの設定は Domain の Field Weights セクションにあり、多くの場合キーフレームも設定できるため、1つまたはすべての Force Field の効果を時間経過に応じてアニメーション化できます。

Force の Strength フィールドは、他の多くの Force Field と同様に負の値を取ることができます。Force の場合、これによって fire が Force Field に引き寄せられるようになります。画面では、シーン内で Empty の動きをアニメーションさせ、Force に負の値を設定した例を示しています。ご想像のとおり、これは fire と smoke を制御する興味深い方法です。

Blender に用意されているすべての Force Field のパラメータを解説するには、別のミニシリーズが必要になるでしょう。なのでここでは扱いませんが、ひとつ注目すべきセクションがあります。それが Falloff で、Force Field の影響範囲を設定できます。

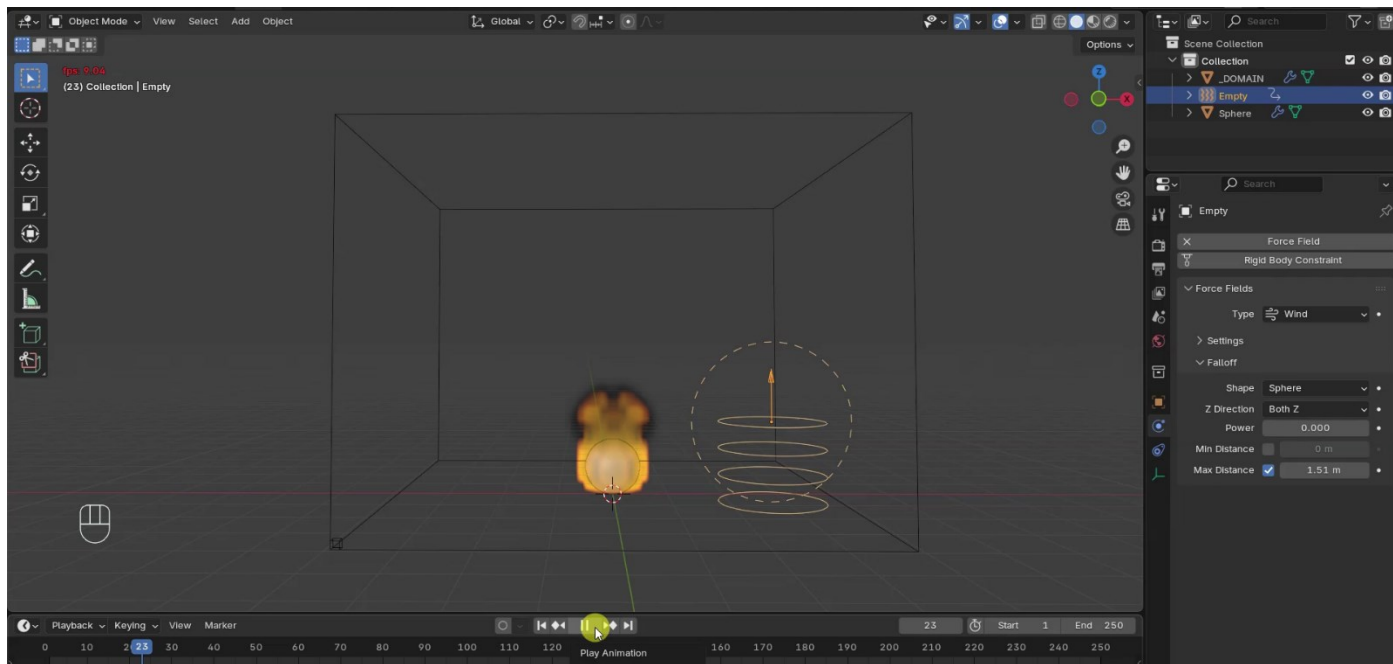


現状では、Force Field は減衰なしで仮想空間全体に影響を与えています。より分かりやすい例を示すために、Force を Wind に変更し、Strength を負のままにしました。Wind は Z 軸方向のみに作用し、Inflow オブジェクトからかなり離れた位置に配置されていますが、それでも Inflow は影響を受けています。



Force Field の影響範囲を制限するには、Falloff セクションの Max Distance フィールドを使用します。このパラメータはデフォルトでは無効です。

有効にすると、Force Field の範囲が 3D Viewport に円で表示されます。他の多くのパラメータと同様に、Max Distance にもキーフレームを設定でき、アニメーションで Force Field の効果を制御する方法のひとつになります。この場合、Strength を負にしたままなので、Wind が flame と smoke を引き寄せています。



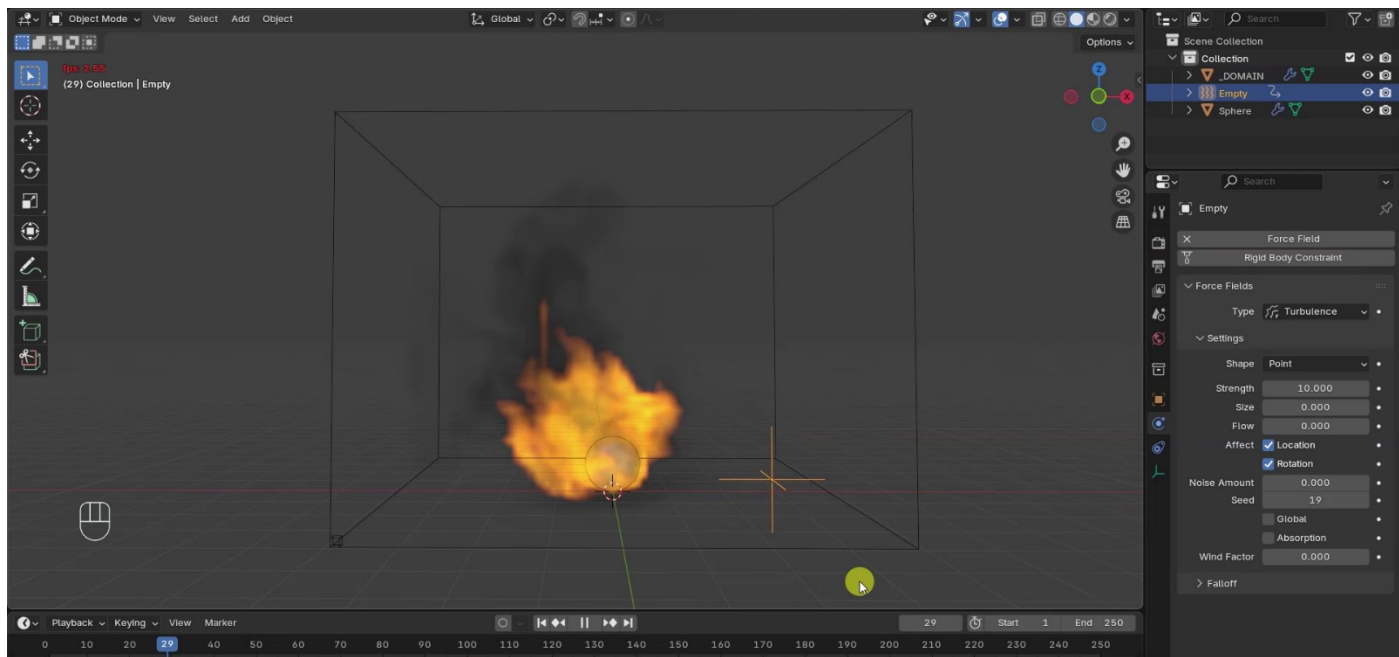
次に Max Distance を無効にし、Force Field を元の方向に戻して Strength を正の値に設定し、他の Force Field をいくつか簡単に確認してみます。

flame に動きを加えるのに非常に便利な Force Field は、もちろん Turbulence です。導入される乱れは非常に強力で、smoke が flame を見えにくくしてしまうため、Domain の Dissolve オプションを使ってこれを抑えています。

Turbulence は Noise パラメータを増やすことでさらにランダムにできます。この Noise は実際には疑似乱数で、一定のパターンに従っています。Noise Seed パラメータを使ってそのパターンを選ぶことができます。もし flame が Vorticity や Texture を調整してもまだ均一に見える場合、小さな Turbulence Force Field がまさに必要なものかもしれません。

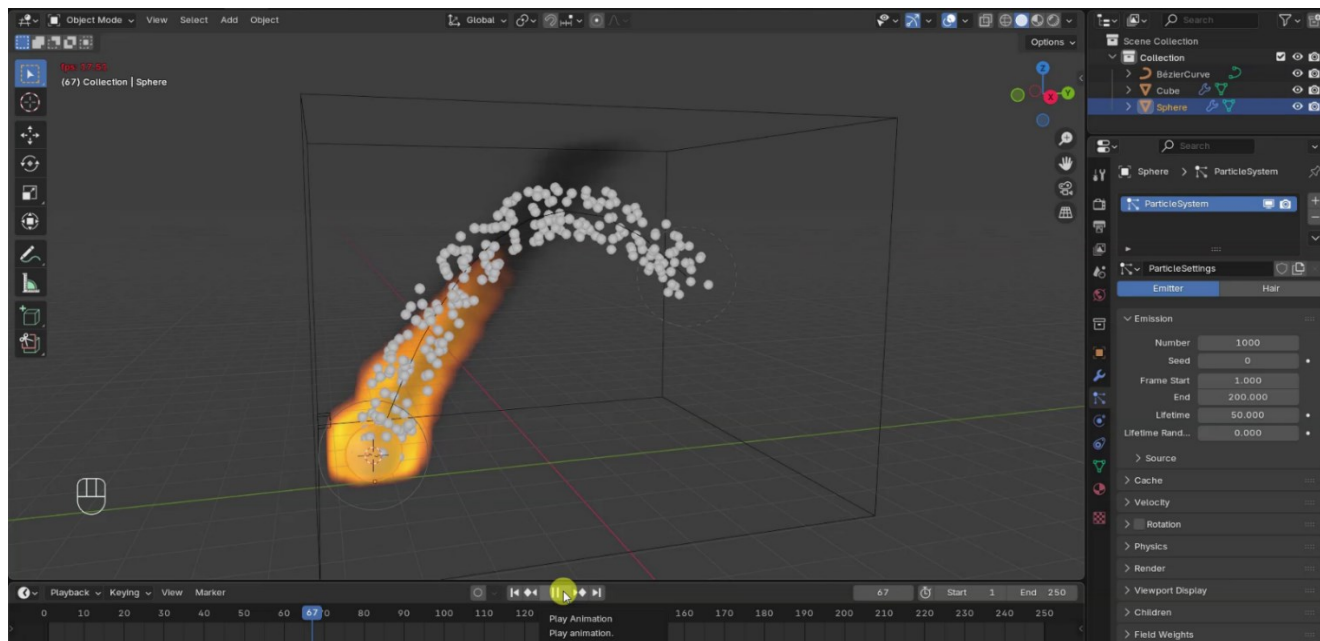
最後に Curve Guide Force Field を見ていきましょう。これは simulation の要素をパスに沿って動かすことができます。この例のために、最初のシーンを一から再現し、新しいファイルにプロジェクトを保存し、Domain 用に新しい cache フォルダを設定しました。

こうすることで、シーンに配置した Domain と Inflow オブジェクトの初期設定は作成時のデフォルトのものになります。ただし、煙で視界が遮られすぎないように、Dissolve を有効にし、デフォルトより多めのフレーム数を設定しました。また、パスのカーブを配置するスペースを確保するために Inflow オブジェクトを横に移動しました。



Bezier カーブを追加し、それに Curve Guide Force Field を割り当てたところ、結果はかなり残念なものでした。カーブや Curve Guide モディファイアに慣れている方なら、カーブの Origin をパスの始点に設定し、最も重要なのはカーブの向きが正しいことを確認するのが良い習慣であるをご存じでしょう。これは Edit Mode の Curve Overlays で確認でき、向きが間違っている場合は Switch Direction で修正できます。simulation を再起動すると、見た目は少し良くなりますが、まだ完全には正しくありません。

Domain で Gravity を無効にしたり、Field Weights 内の Curve Guide 値を上げたりと、さまざまな設定を試しました。もちろん Vorticity や Reaction Speed を下げたり、解像度を上げたりもしました。何かしらの変化はありましたが、十分ではありません。そこで、私が望む結果を得るために普段使っている方法をお見せします。Inflow を持つオブジェクトに Particle System の Emitter タイプを追加し、パーティクルが正しく放出されるように基本パラメータを設定します。Emitter が Force Field の範囲内にあるため、パーティクルはすぐにカーブパスに沿って動きます。



Force Field の範囲は、そのパネル内の Minimum Distance パラメータで設定できます。

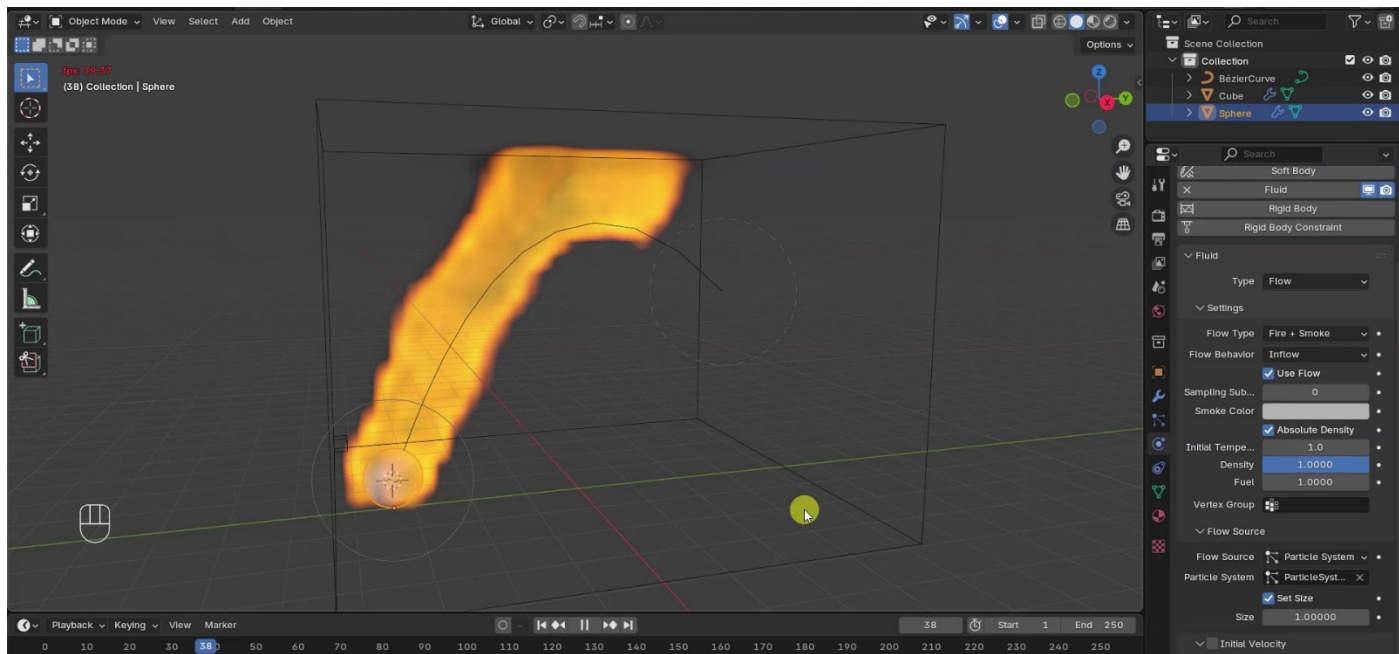
ここで Clumping を 1 に設定すれば、すべてのパーティクルがカーブの終点に収束します。

この時点で、Emitter の Particle System の Render を None に設定します。

これらのパーティクルはレンダリングされるべきではないからです。その後、Inflow パネルに戻ります。

このパネルで Flow Source を Mesh から Particle System に変更し、先ほど作成した Particle System を指定します。こうすることで、カーブに沿って動くパーティクルから fire と smoke が放出されます。

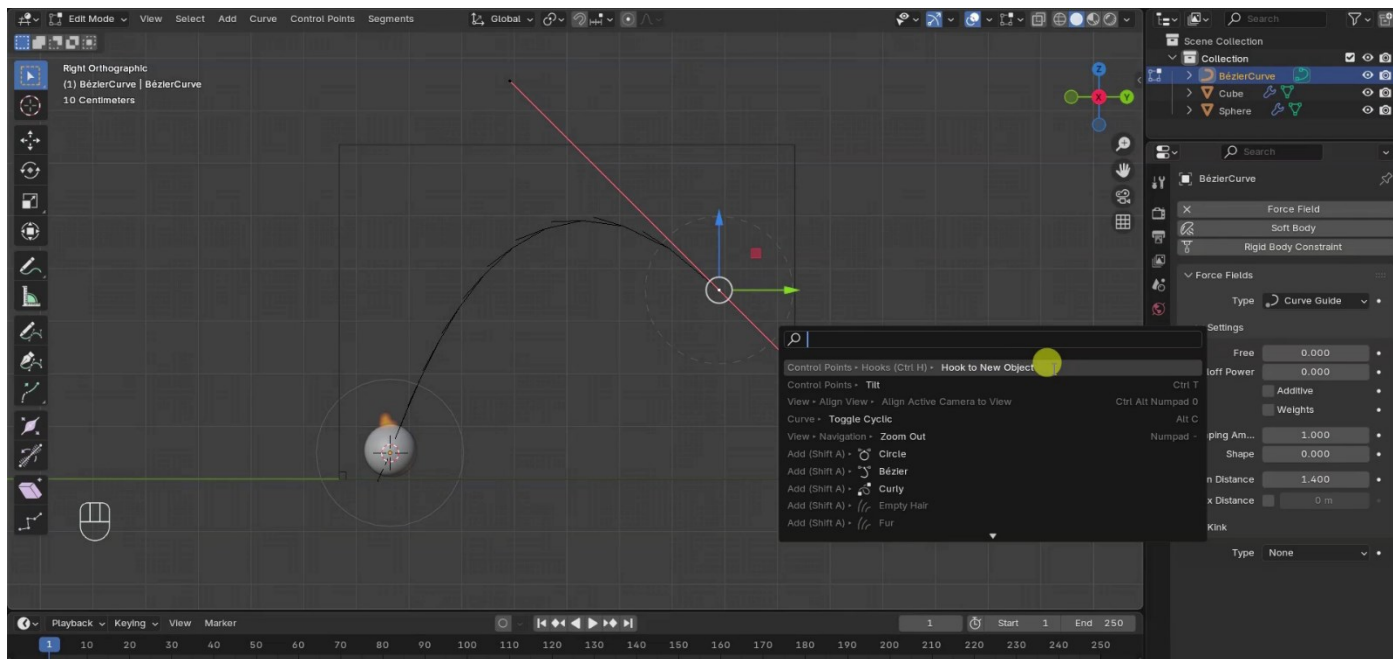
ここから Fuel、Vorticity などのパラメータを調整して、flame に望む見た目を与えることができます。



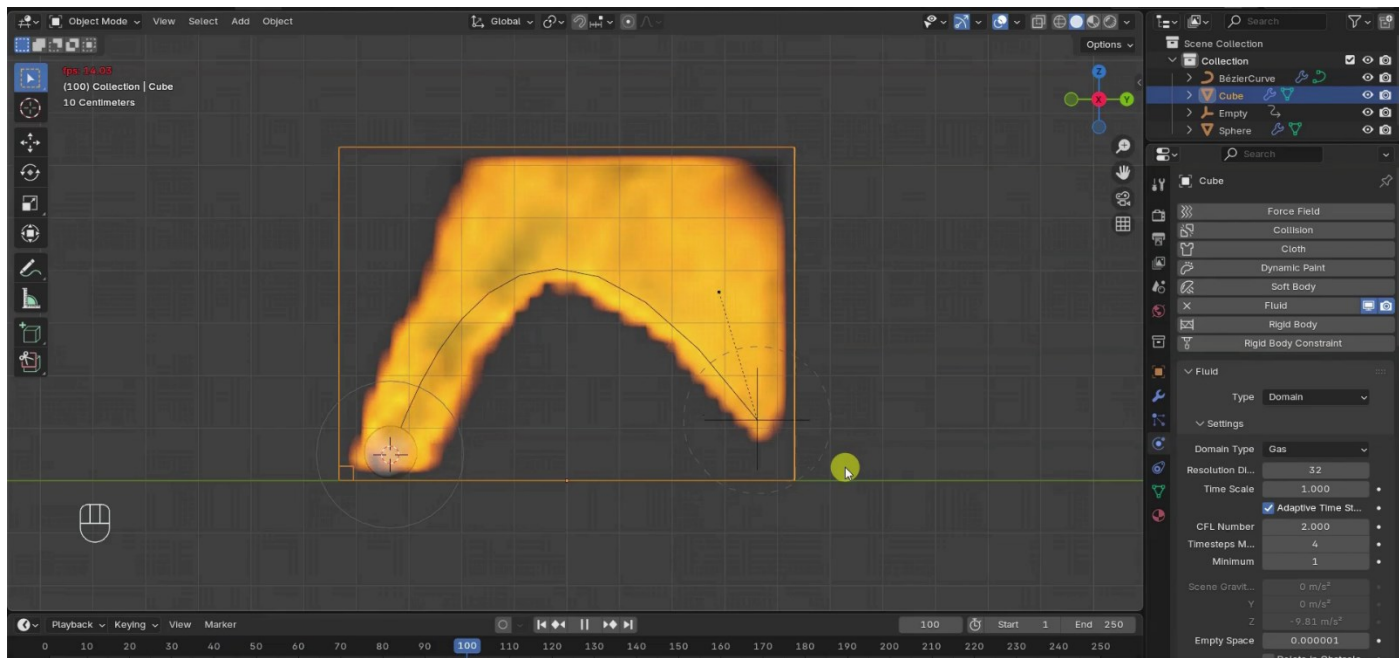
チュートリアルを終える前に、Hooking と呼ばれるテクニックを使ってカーブの制御点を外部オブジェクトにリンクする方法も紹介します。

これにより、カーブの形を簡単にアニメーション化できます。

まずカーブを選択し、Edit Mode に入り、最後の制御点を選びます。次に Hook オペレーターを使用し、Hook To New Object オプションを選択すると、そのための Empty が作成されます。

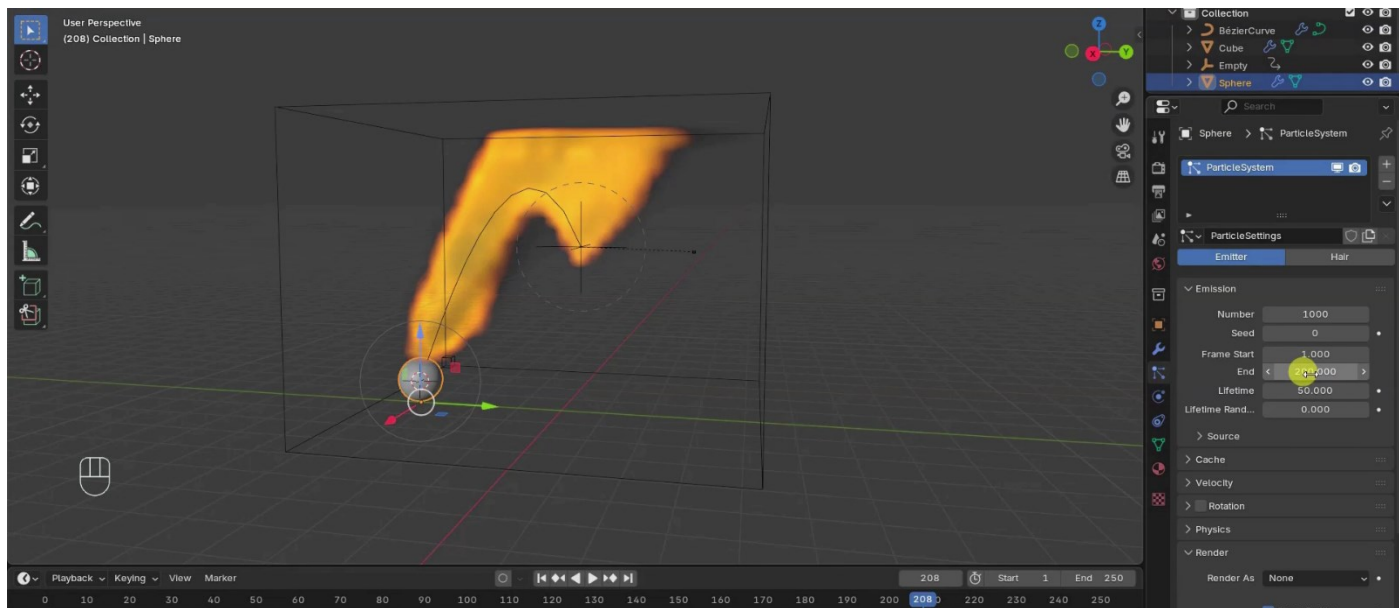


Object Mode では、この Empty を動かしてアニメーション化でき、その動きに応じてカーブと flame が変形します。もちろん同じテクニックを使って他の制御点にも Empty を追加し、より複雑なアニメーションを作成することも可能です。



ご覧のとおり、ある時点でパーティクルの放出が止まります。

これは Frame End と呼ばれるパラメータで、Particle System パネル内にあります。同じパネルにある Lifetime パラメータは、パーティクルがカーブに沿って進む速さを制御します。



最後にもう少し考慮すべき点を挙げます。Emitter オブジェクトのサイズは放出面の大きさを決定します。したがって、パーティクルをより細いカーブに集中させたい場合は、このオブジェクトのサイズを調整してください。また、パーティクルの数は flame カーブがどれほど分割されて見えるかに影響するので、炎を滑らかで連続した見た目にするには数を増やす必要があります。この点については、次回のチュートリアルで扱う Motion Blur を使用するのも効果的です。

Fire Simulations in Blender 4.5 basics | 10/10: Motion Blur, Filmic, Compositing, Eevee

<https://youtu.be/TAUIbsfeWbo>

皆さん、こんにちは！このシリーズの第10回、そして最終回のチュートリアルでは、完全な操作手順ではなく、Motion Blur、Color Management、そしてポストプロダクションでの Node Compositing を使って fire の見た目をレンダリング中やその後に変更するためのヒントや提案を共有します。また、Eevee で fire と smoke をレンダリングする際の設定についても解説します。

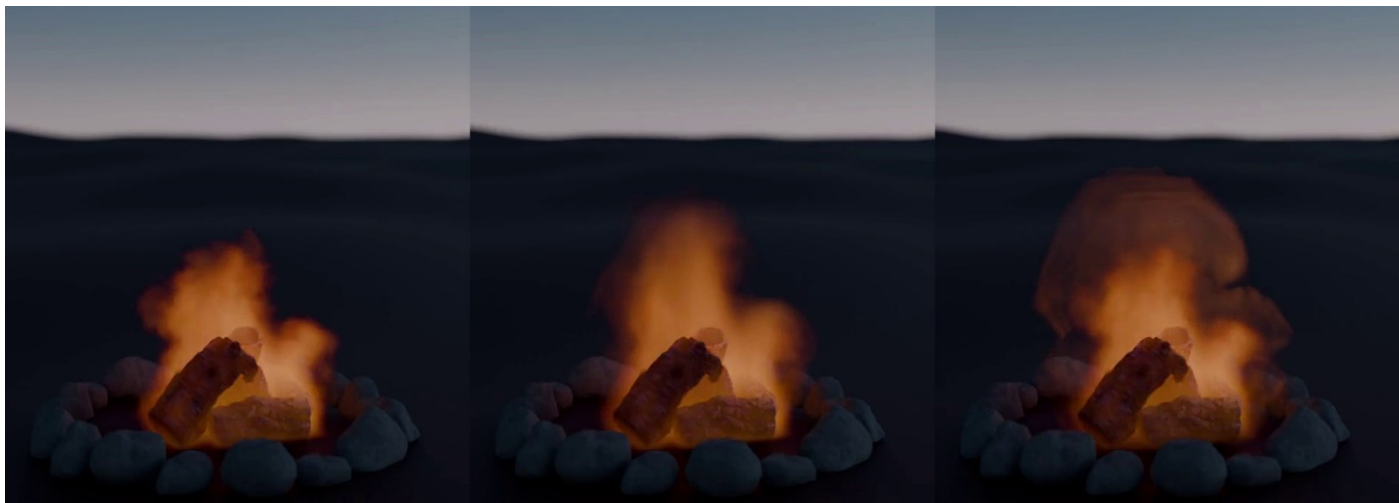
まず、ポストプロダクションではなく、レンダリングそのものに影響を与える設定から始めましょう。それが Render タブで選択する Color Management です。特に Filmic モードは、より彩度の高い flame を生成する傾向があります。High Contrast を選択すると fire がより強調された見た目になります。

画面では比較を示しています。左は AgX でレンダリングしたもの、右は Filmic と High Contrast でレンダリングしたものです。Color Management は fire だけでなく画像全体に影響を与えます。下部の石を見てもそれが分かります。すぐに、Compositing で flame だけを分離して調整する方法を見ていきます。



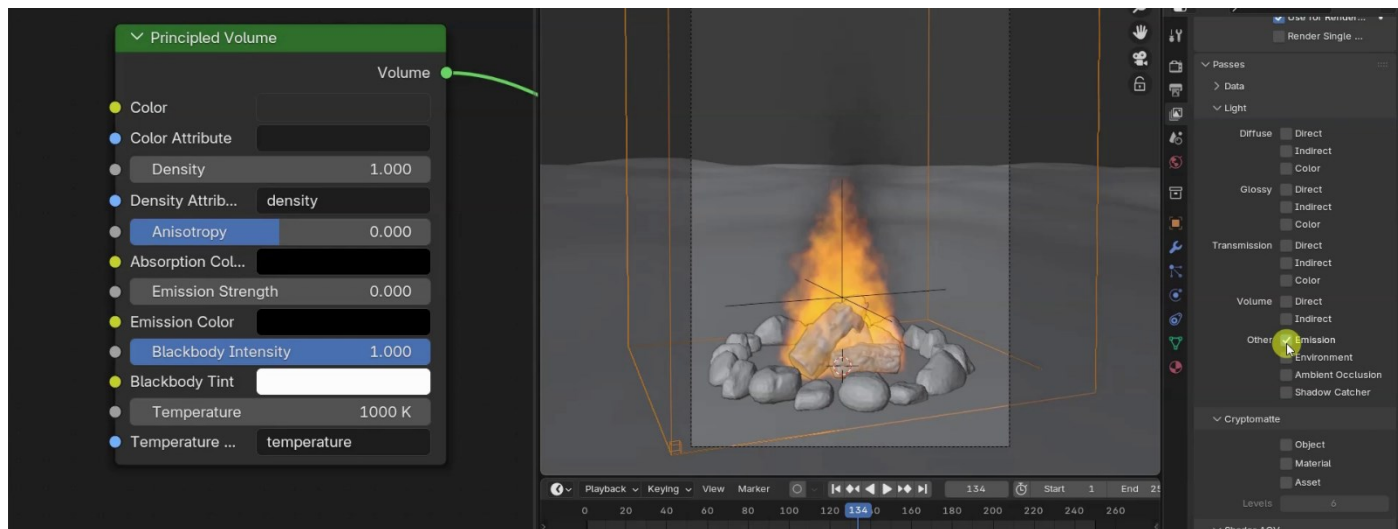
Render タブで有効にできるもうひとつのオプションが Motion Blur です。これは flame がフレームごとに完全に静止することがないため、この種の simulation をレンダリングする際に非常に有効です。以前のエピソードで、Domain に Motion Blurring 用の Velocity Scale に関連するフィールドがあることを説明しましたが、Motion Blur は Render タブのレンダリングレベルでも有効化する必要があります。このタブの Shutter パラメータは、現在のフレームの前後に何フレームを考慮するかを定義します。値を大きくするとモーションブラーは強くなります。

画面では比較を示しています。Motion Blur を無効にしたレンダー、Motion Blur を有効にして Domain の Velocity Scale を 1 に設定したレンダー、そして Velocity Scale を 10 に設定して違いを強調したレンダーです。残念ながら、Shutter や Velocity Scale の正しい値を見つける「魔法の公式」はありません。最適な選択はプロジェクトのニーズによって変わるため、テストが必要です。Motion Blur を有効にして Shutter や Velocity Scale の値を上げると、レンダリング時間が大幅に増加することを覚えておいてください。



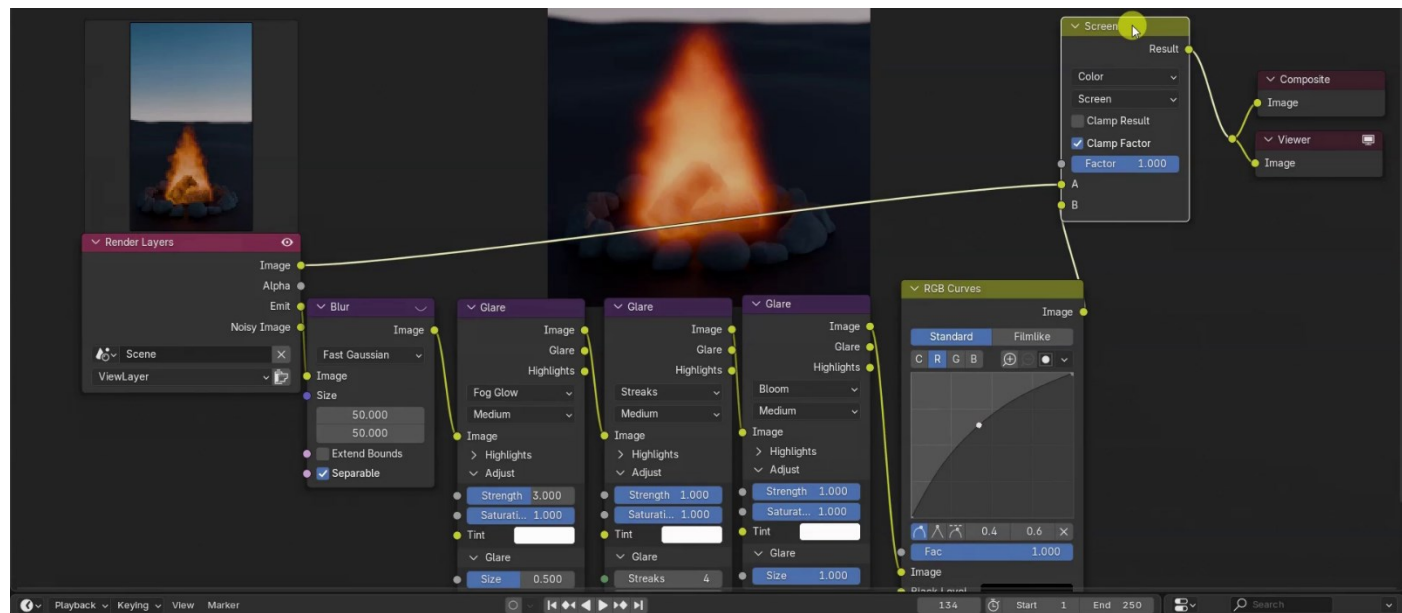
レンダリングにおいて最終的な見た目は、ポストプロダクションでの適切な Node Compositing にも依存します。fire も例外ではなく、この場合は Emission Render Pass を利用できます。これにより Emission や Volumetric Material のように光を放つソースのピクセルを分離してコンポジットできます。そこで、テスト

レンダーを行う前に専用パネルでこの Render Pass を有効化し、Compositing Nodes エディターに移動しました。



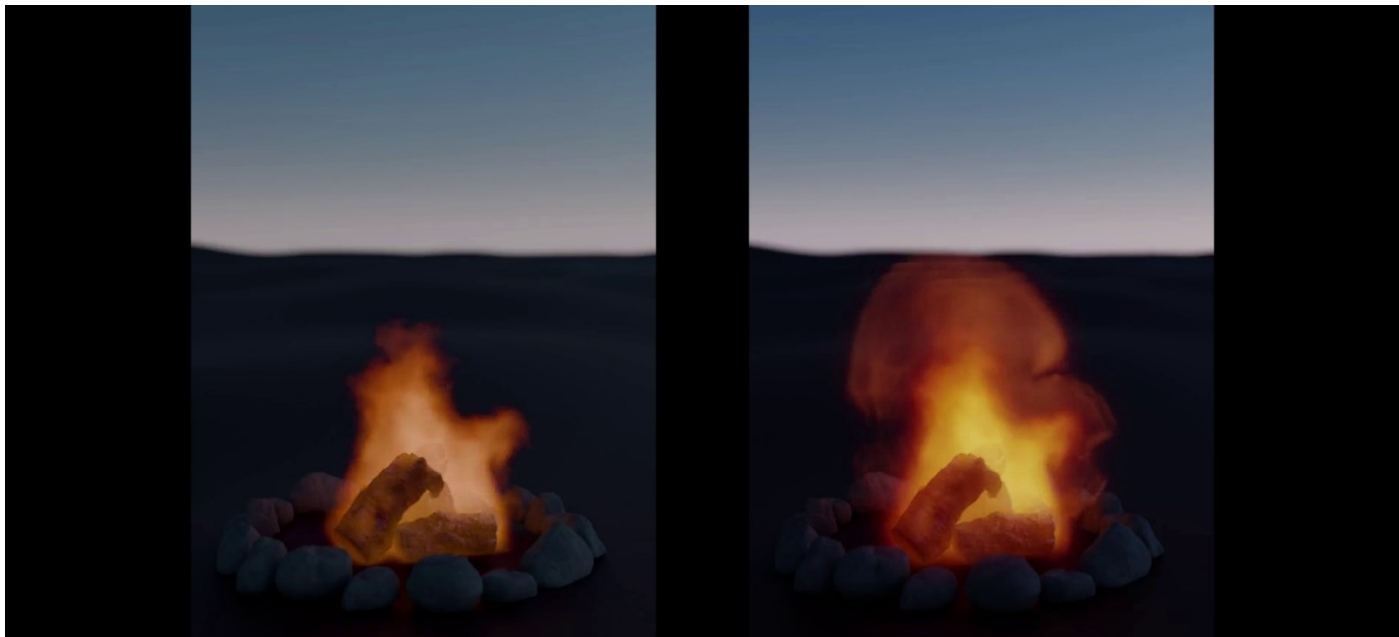
黒背景にカラーで flame を出力する Emission 情報を使って、blur、glow、contrast、あるいは color の改善にさまざまな Compositing ノードを適用できます。これらの処理結果は元のレンダーに戻すことができます。Blender にはそのためのさまざまなブンドモードや係数があります。例えば flame を個別に処理し、blur をかけたり複数の Glare 効果を追加したりした後、その結果を Mix Color ノードを Screen に設定して元のレンダーに重ねています。また、RGB Curves ノードの C チャンネルを使って明るさを調整し、同じノード内で Red チャンネルだけを変更することも試しました。最終結果は少し強すぎるかもしれませんが、

Factor 値を調整すれば修正できます。さらに、Mix Color ノードを選択して M キーを押せば、Mute 機能でノードを有効・無効に切り替えることもできます。



Eevee のレンダリング設定に関するヒントに進む前に、ここまで解説してきたオプションの最終比較をお見せします。左側は Color Management を AgX に設定し、Motion Blur なし、Compositing なしでレンダリングしたものです。右側は同じシーンを Filmic の High Contrast でレンダリングし、Shutter を 1、Velocity

Scale を 10 に設定した Motion Blur を加え、さらに先ほど説明したノード構成で少しポストプロダクションを行ったものです。

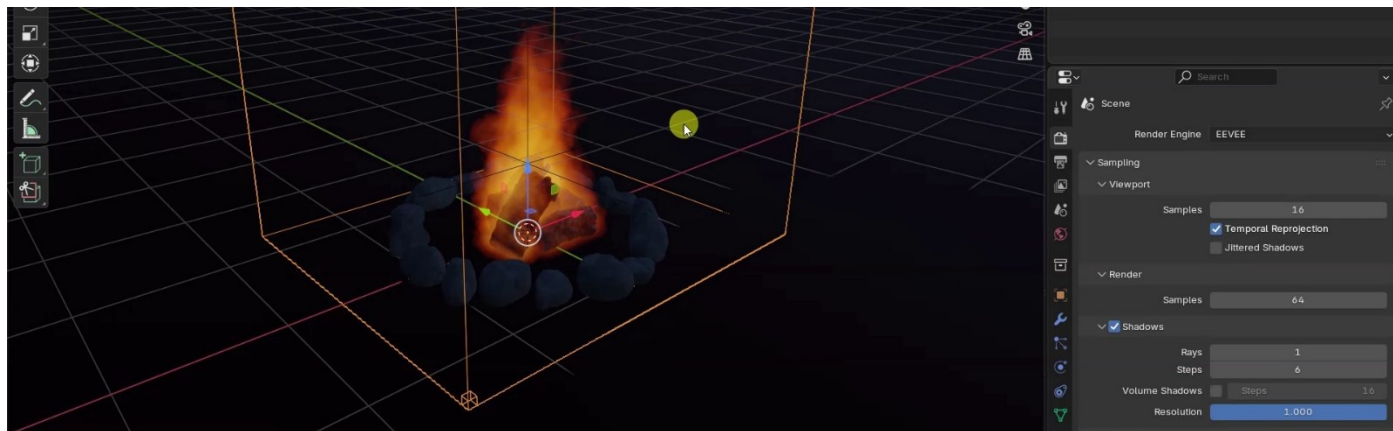


では、Eevee のレンダリング設定について進めましょう。

ここまで私は常に Cycles を使ってレンダリングしてきましたが、Eevee ではリアルタイムレンダリングという特徴から、いくつかの違いが必要となります。

Eevee では fire と smoke の simulation はシーンのスライスとして表現され、それらが半透明のシートのように積み重ねられて表示されます。設定すべきパラメータのひとつは、観察者の位置に対してこれらのスライスの開始点と終了点です。

最終的な品質を左右する設定は主に Render タブの Volumes セクションにあります。



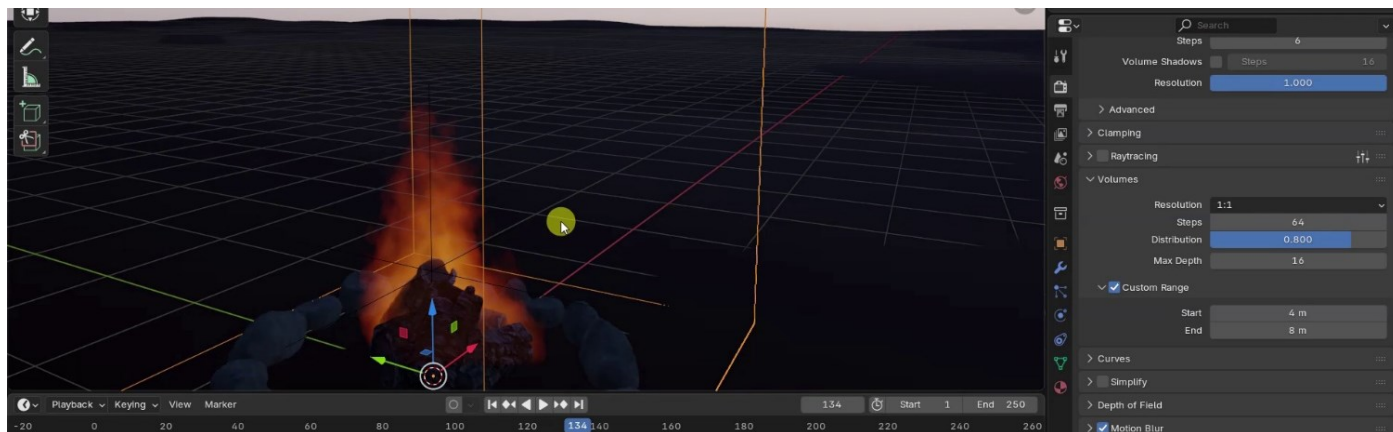
このセクションで最初のパラメータは Resolution です。直感的に、この値が 1 対 1 に近いほど品質は高くなります。比率が低いと品質は下がりますが、レンダリング時間は短くなります。

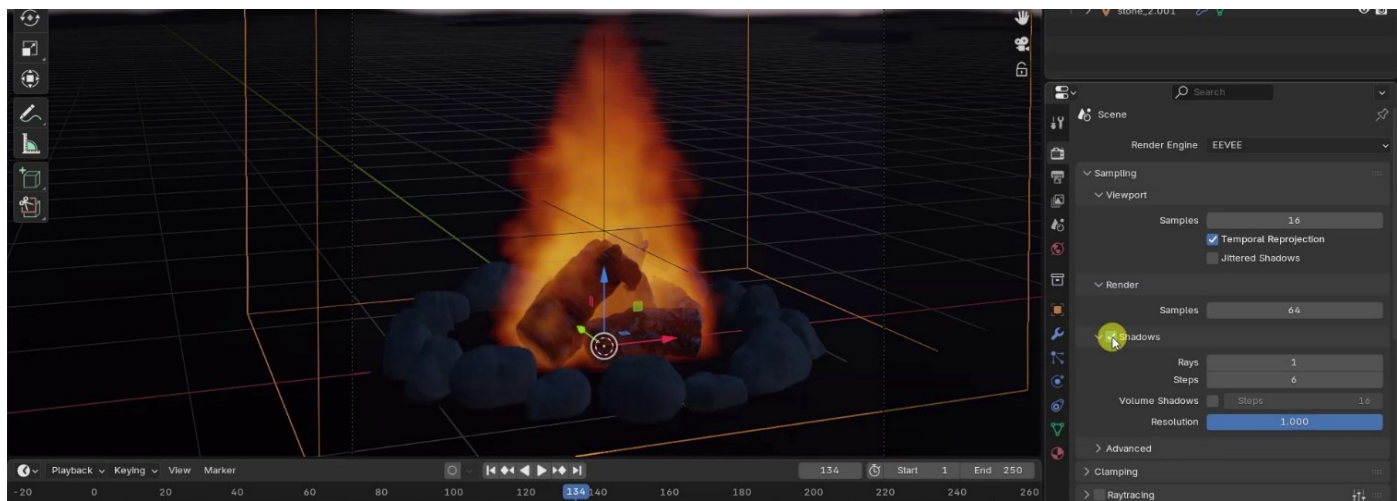
先ほど説明したように、表示領域は Steps と呼ばれるスライスに分割されます。Steps パラメータは、デフォルトで有効になっている Custom Range Start と End のオプションと連動します。Start から End までの空間（観察者の位置から前方）が Steps の数に応じてスライスに分割されます。それぞれのスライスがレン

ダリグされ、合成された結果が最終画像となります。当然、デフォルトの 100 メートルは不要なので、Start と End の値を simulation の Domain に合わせて小さくし、その範囲に Steps を集中させることでより良い結果が得られます。

このセクションで理解すべき最も重要なパラメータはこれらであり、特にシーンを移動するときに simulation が突然消える理由を説明するのに役立ちます！

シャドウとその品質については、Render タブの Sampling Shadows セクションに Volume Shadows チェックボックスがあります。これはデフォルトでは無効ですが、有効にすると Steps の値を設定でき、当然ながら品質に直結します。





これで Blender 4.5 における fire と smoke の Fluid simulation 基礎シリーズの最終チュートリアルは終了です！

このシリーズが皆さんの役に立ったことを願っています。

もし感謝の気持ちを伝えたいなら、チャンネル登録と動画への Like が一番の方法です。

ありがとうございました、それではまた会いましょう！