

## ParaView可視化初中級演習

大嶋拓也(新潟大学)

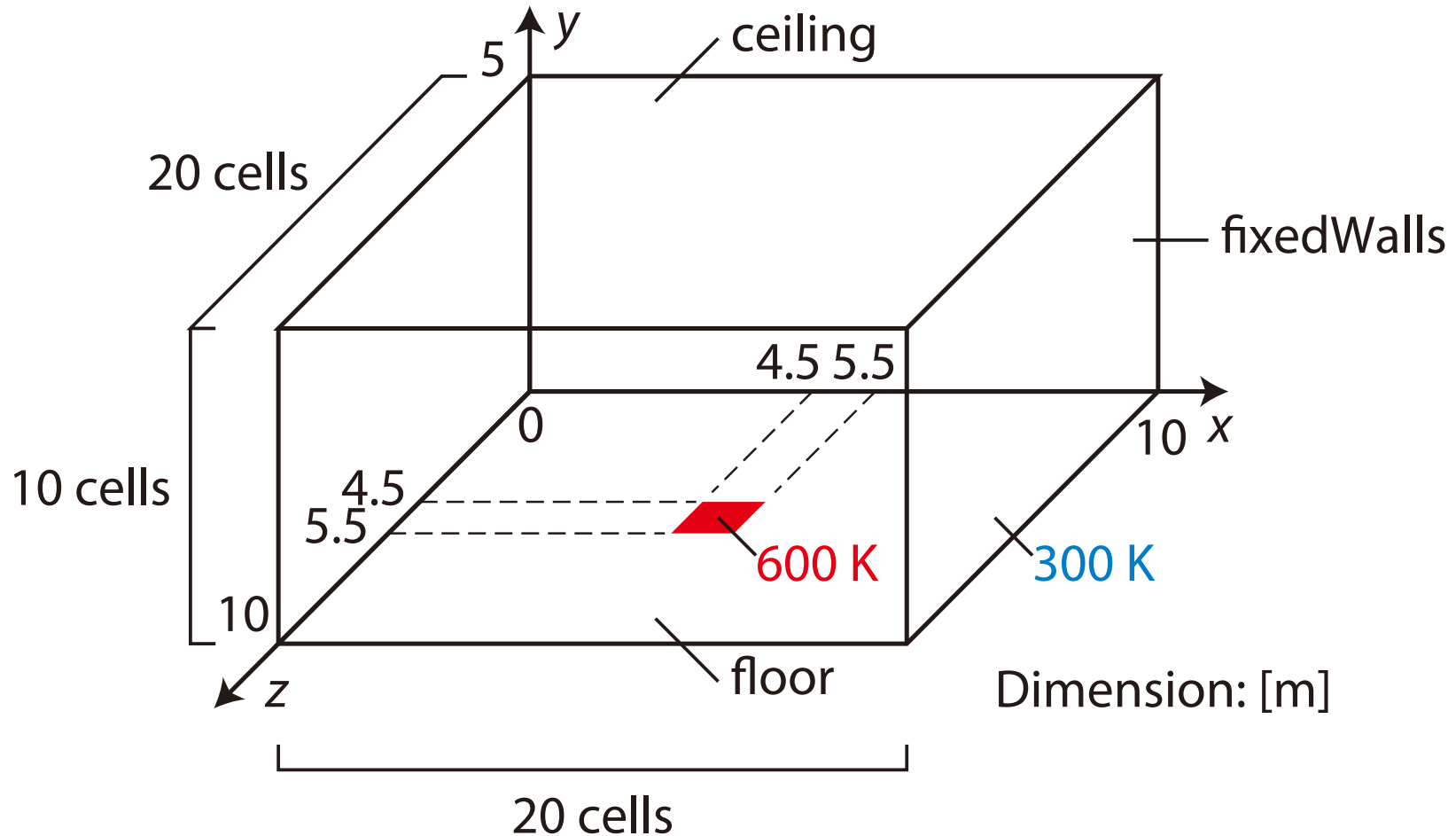
2010年5月14日 第1回オープンCAE講習会

OPENFOAM<sup>(R)</sup> is a registered trade mark of OpenCFD Limited, the producer of the OpenFOAM software and owner of the OPENFOAM<sup>(R)</sup> and OpenCFD<sup>(R)</sup> trade marks. This offering is not approved or endorsed by OpenCFD Limited.

- 本日使用するケースの説明
- OpenFOAMクイックスタート
  - ParaViewのサンプルデータ作成のため
- ParaView可視化演習: フィルタの使い方
  - カスタムフィルタの作成・適用
  - 任意シードを用いたStream Tracer
  - パーティクルトレース
  - マルチビュー

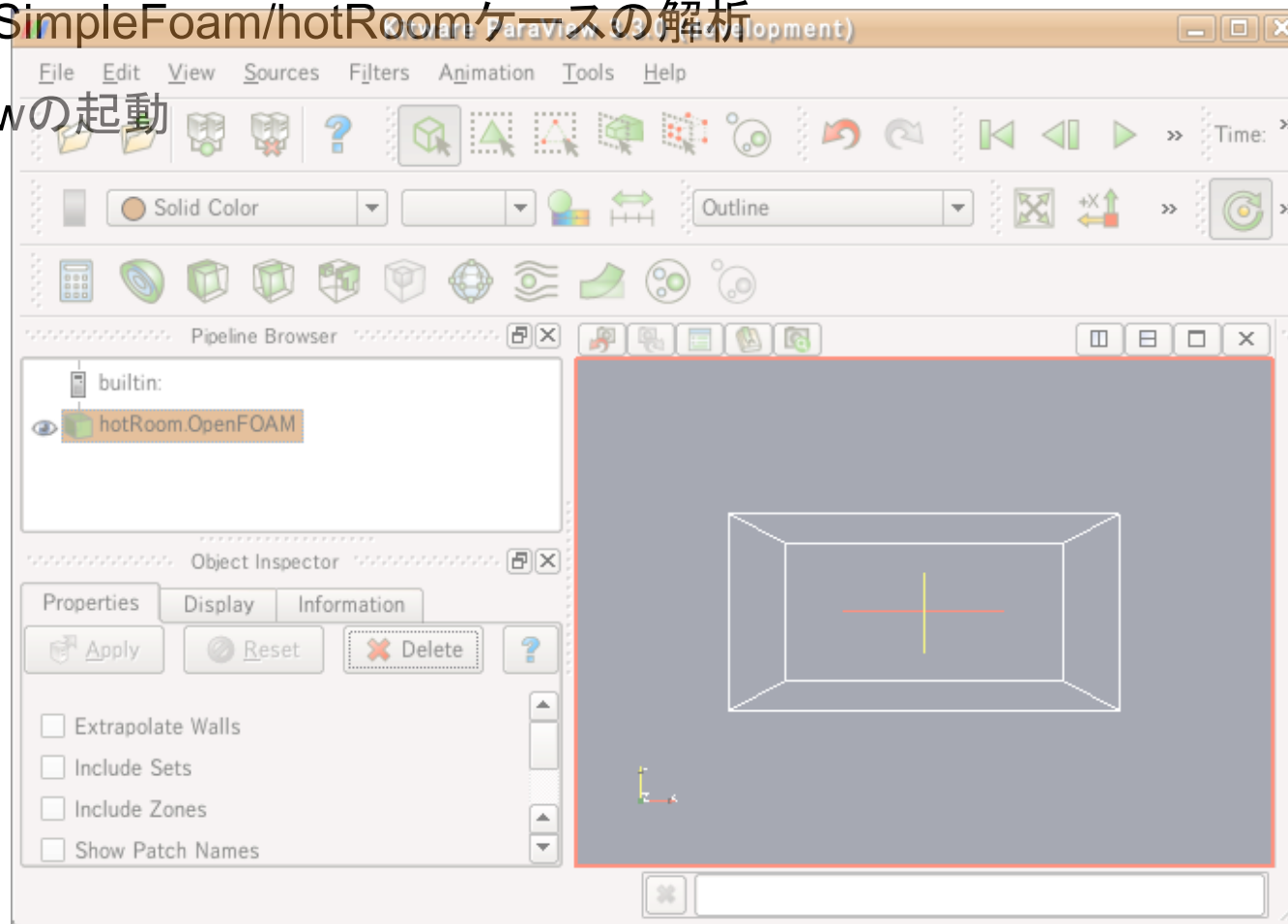
# 本日のケース

- OpenFOAMのbuoyantSimpleFoam/hotRoomケース
- 室内の発熱による対流



# OpenFOAMクイックスタート

- buoyantSimpleFoam/hotRoomケースの解析
- ParaViewの起動



# OpenFOAMクイックスタート(1)

- まず最初に、VMwareの画面を最大にしてください。
  - VMware Player: 画面右上の最大化アイコン
  - VMware Fusion: control + command + return
- OpenFOAMのオペレーションは、コマンド入力が基本



# OpenFOAMクイックスタート(2)

- OpenFOAMの実行ディレクトリを作成

```
mkdir -p $FOAM_RUN
```

- 実行ディレクトリに移動

```
run
```

- チュートリアルデータを実行ディレクトリにコピー

```
cp -a $FOAM_TUTORIALS .
```

- 今回使用するチュートリアルのディレクトリに移動

```
cd tutorials/buoyantSimpleFoam/hotRoom
```

- チュートリアルケースの実行

```
./Allrun
```

以下がまとめて実行される:

- setHotRoomアプリケーション(初期値設定ユーティリティ)のコンパイル
- blockMeshによるメッシュ作成
- setHotRoomの実行
- buoyantSimpleFoam (ソルバ)の実行

- ログからの残差履歴の切り出し

```
foamLog log.buoyantSimpleFoam
```

- Gnuplot (グラフ描画ソフトウェア)の起動

```
gnuplot
```

- 残差履歴のプロット (折れ線で、縦軸を対数スケールで、logs/pd\_0をプロット)

```
set st d l  
set log y  
plot "logs/pd_0"
```

- ◆ 残差の収束を確認する
- ◆ "logs/pd\_0"に、", "で続けて他の変数の履歴も指定可能

- Gnuplotの終了

```
exit
```



# OpenFOAMクイックスタート(5): ParaViewの起動

- ParaViewにデータの場所を指示するための、空のファイルを作成

```
touch hotRoom.OpenFOAM
```

ParaViewは拡張子でデータ形式を判別するのに対し、OpenFOAMのデータファイルは全て拡張子が無いため、この操作が必要になる

- ParaViewの起動と同時に、データを開く

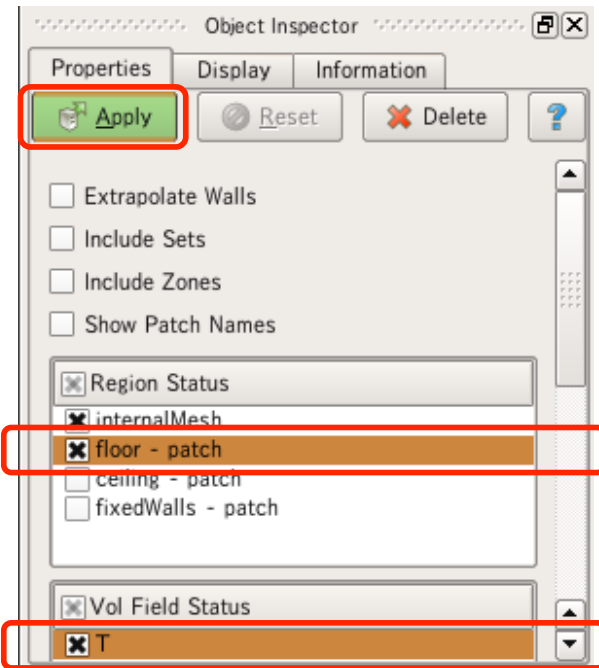
```
paraview --data=hotRoom.OpenFOAM
```

`paraview --data=ファイル名` で、起動と同時にデータを開くことができる

OpenFOAMの`paraFoam`コマンドは、以上を自動的に行うためのもの

# OpenFOAMクイックスタート(6): 読み込むデータの選択

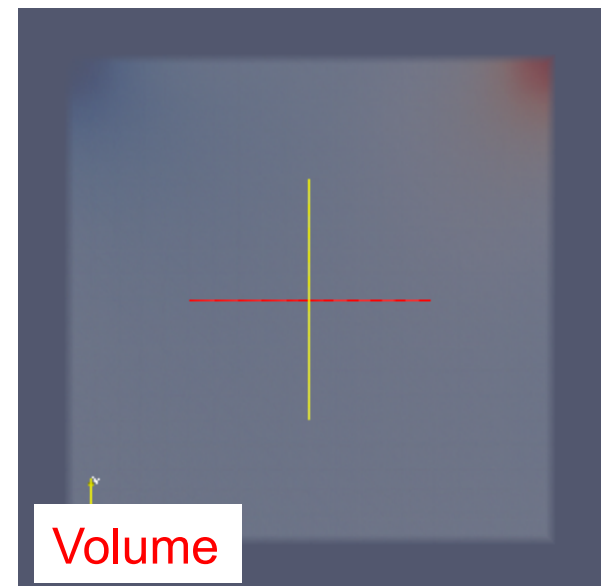
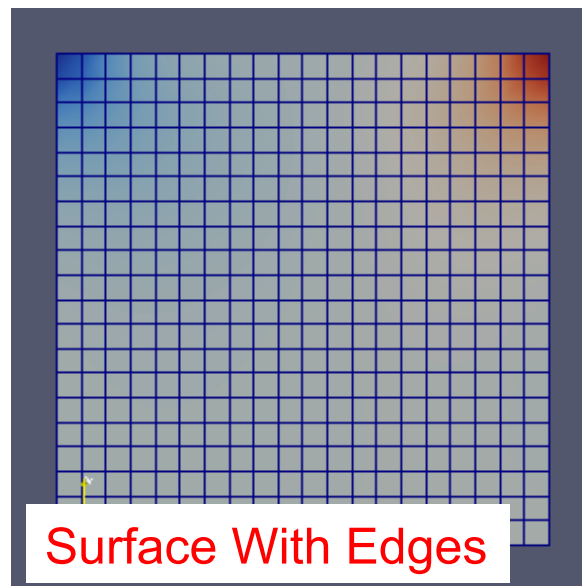
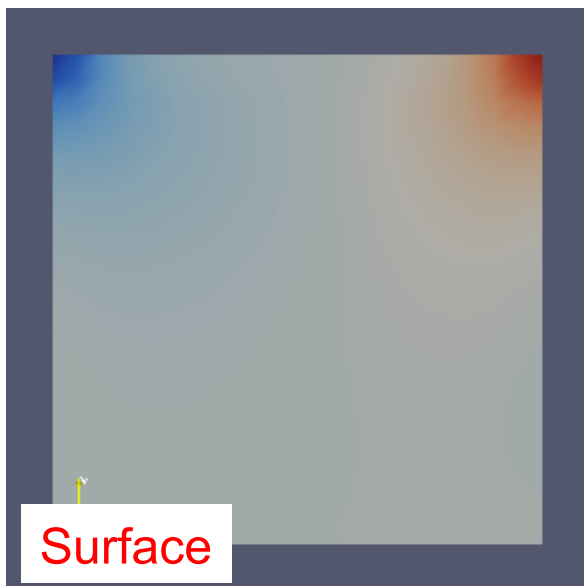
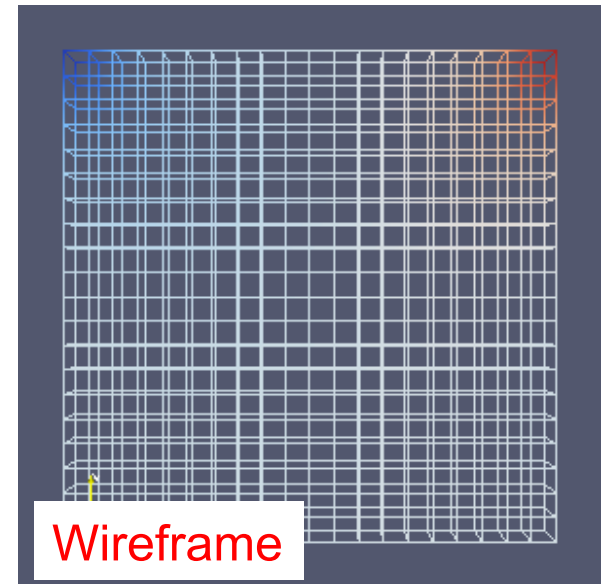
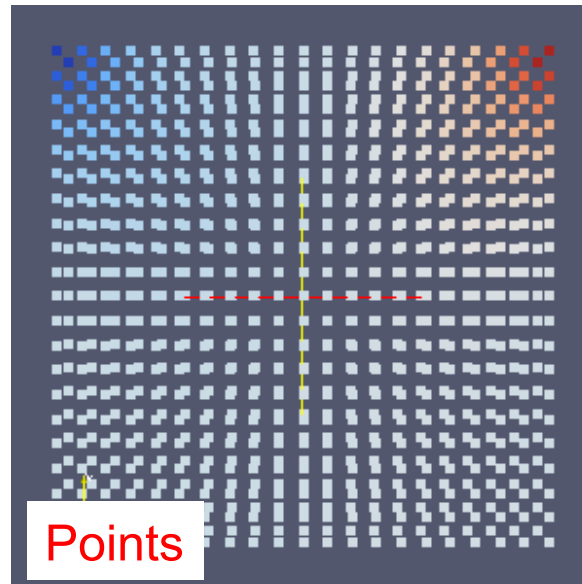
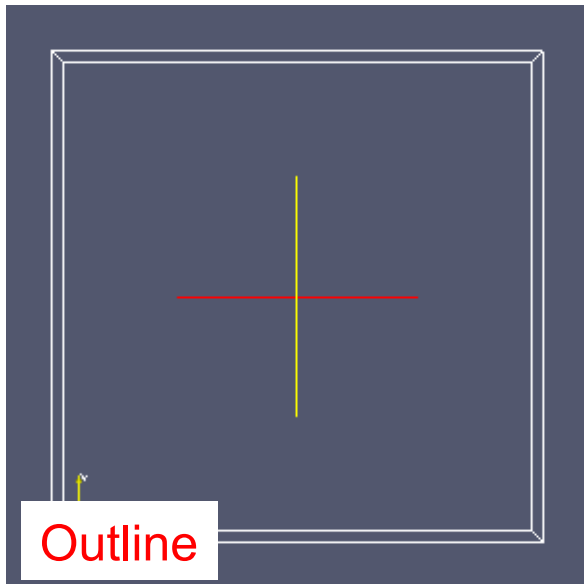
- データを読み込む際に、Object Inspector下、Region Statusの“floor”、Vol Field Statusの“T”をチェックする → “Apply”



- 最後の時刻 (Time: 1000) まで進める

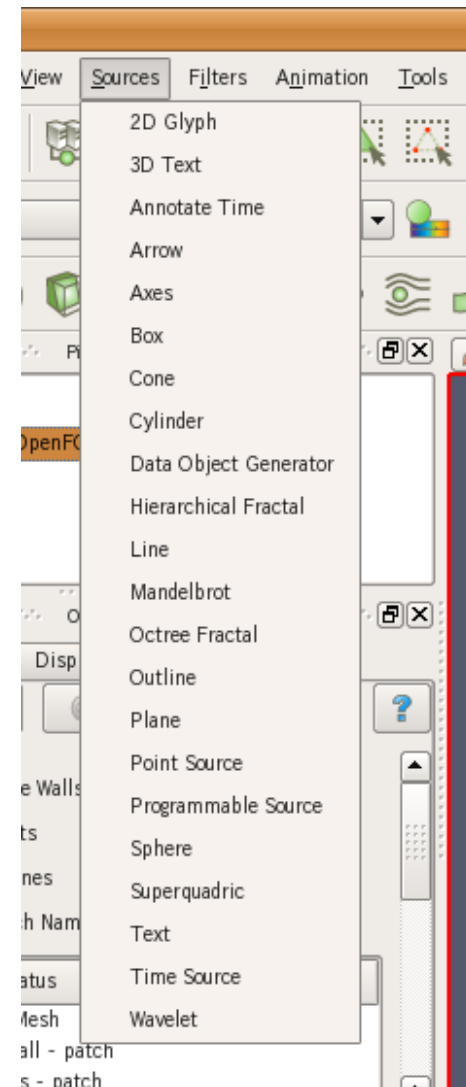


# ParaView復習: レプレゼンテーション



## Sourcesメニュー

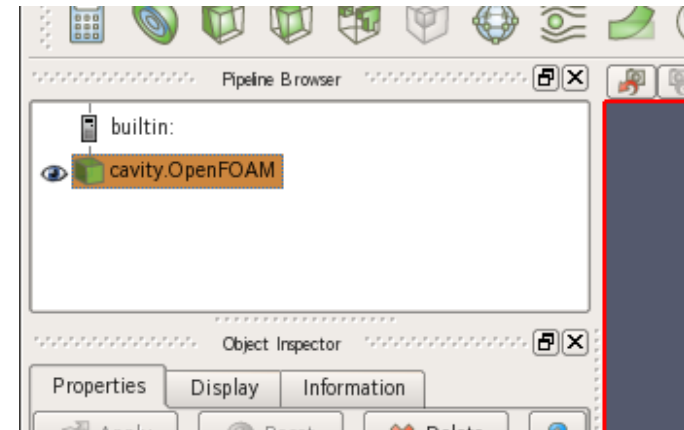
- ParaViewの中で、簡単な形状の作成が可能
  - Annotate Time (解析上の時刻の表示)
  - Box (直方体)
  - Cylinder (円筒)
  - Line (線分)
  - Plane (面)
  - Sphere (球)
  - ...
- フィルタへの入力としても使用できる



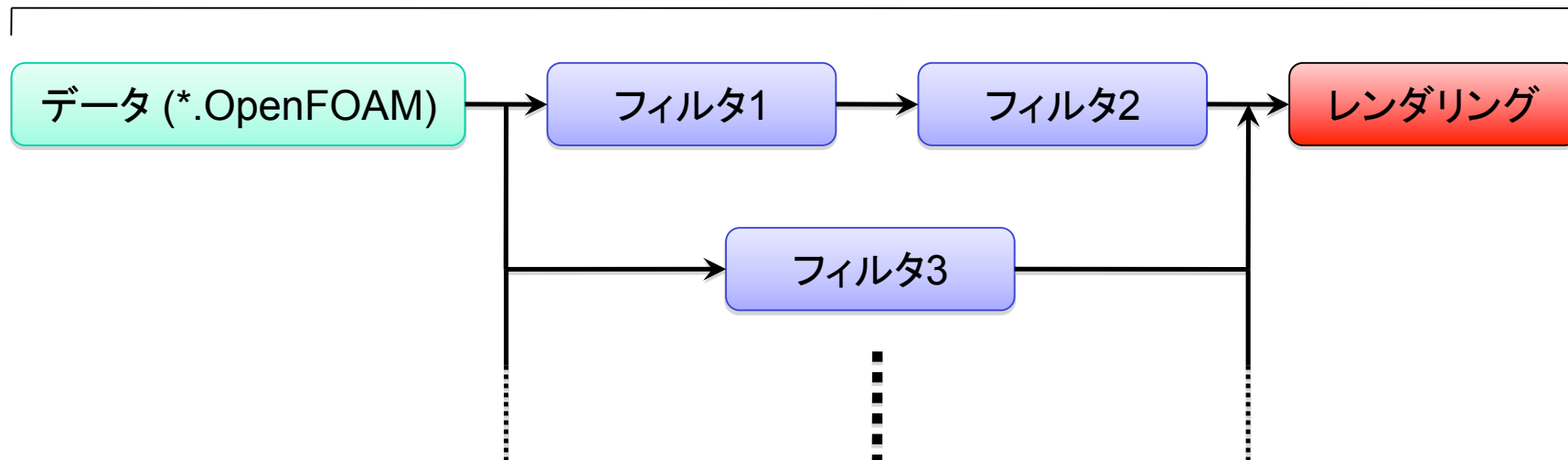
## Filtersメニュー

### •Filtersメニュー + Filtersツールバー

- 数十種類のフィルタ
- データ・形状の加工・抽出が可能
- 複数のフィルタを組合せられる(パイプライン)
- Pipeline Browserに、パイプラインの状態表示

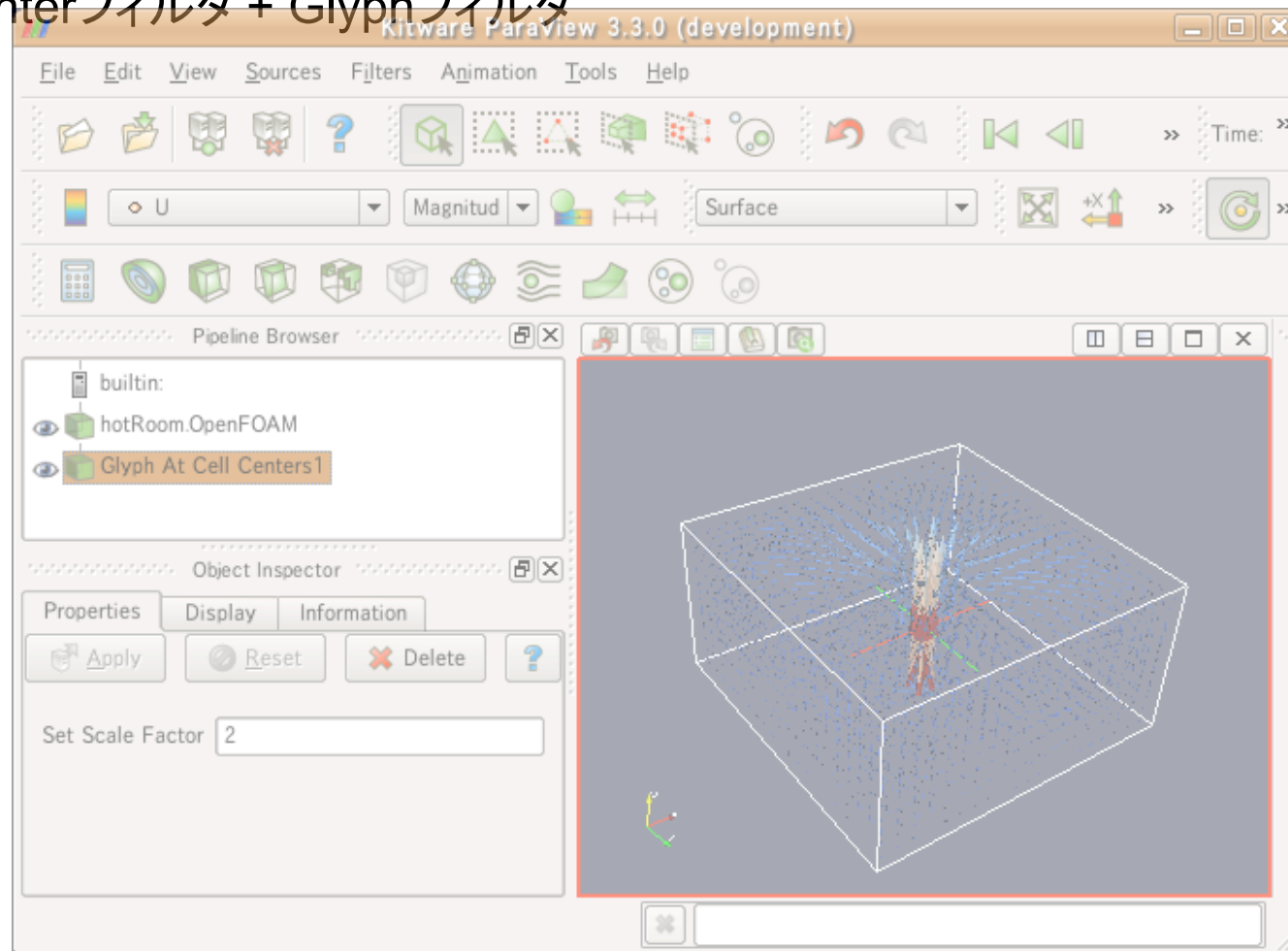


### パイプライン



# カスタム・フィルタの作成・適用

- Cell Centerフィルタ + Glyphフィルタ



## ポイント

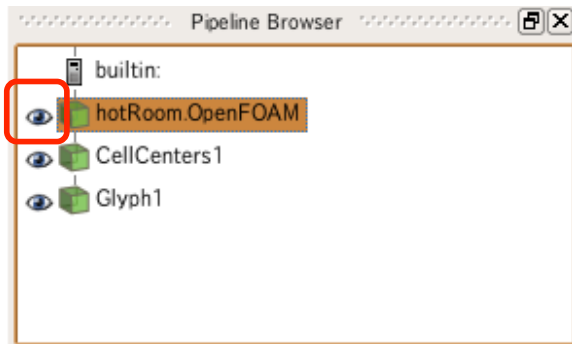
- カスタム・フィルタ: 頻繁に使用するフィルタの組合せ (パイプライン) を、新たなフィルタとして登録できる
- パイプラインを作成・選択し、“Tools”→“Create Custom Filter...” で作成
- 作成されたフィルタは、“Filters...” メニューに追加される
- カスタム・フィルタ実行の度に変更したいプロパティ項目を、作成中に指定できる

## 例題

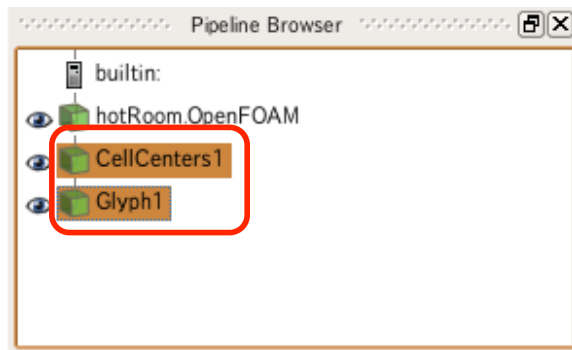
- Cell Centerフィルタ (セル値をセル中心点に与える) + Glyphフィルタ

# カスタム・フィルタ: パイプラインの作成

- hotRoom.OpenFOAMに対し、“Filters...” → “Cell Centers”、Glyphフィルタを直列に適用する
- プロパティは、全てデフォルトのまま可
- 適用後、hotRoom.OpenFOAMをVisibleにする (領域を判りやすくするため)



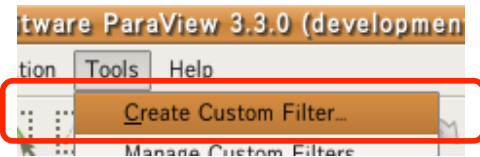
- Pipeline Browserの“CellCenters1”、“Glyph1”を選択する (Shift+クリック)





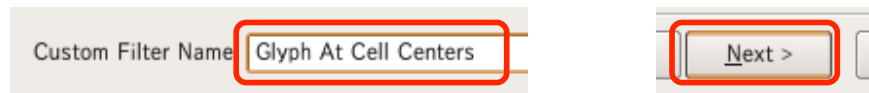
# カスタム・フィルタ: カスタム・フィルタの作成

- “Tools” メニュー→ “Create Custom Filter...” を選択

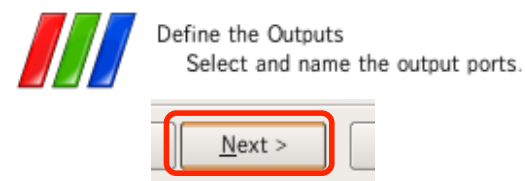
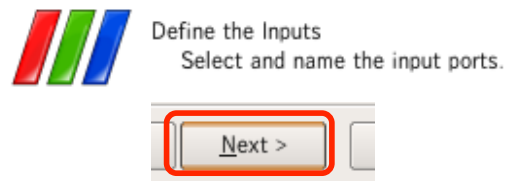


- または、Pipeline Browserの選択上で右クリック→“Create Custom Filter...”

- カスタム・フィルタ名を入力: “Glyph At Cell Centers” → “Next”

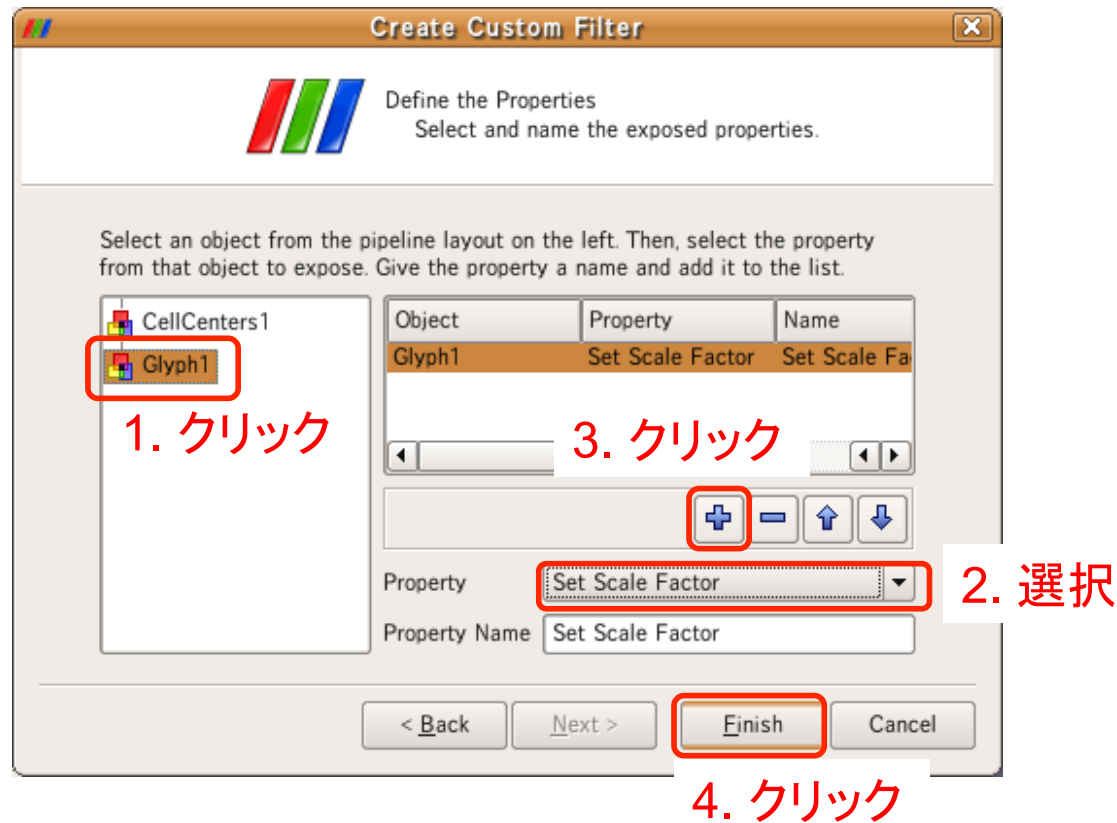


- “Define the Inputs”、“Define the Outputs” の画面は、ともに “Next”

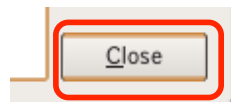


- “Define the Outputs” は、途中のフィルタ出力も利用したいときに使う。“Define the Inputs” は、後の演習問題で。

- ここでは、GlyphフィルタのSet Scale Factorプロパティを可変にする

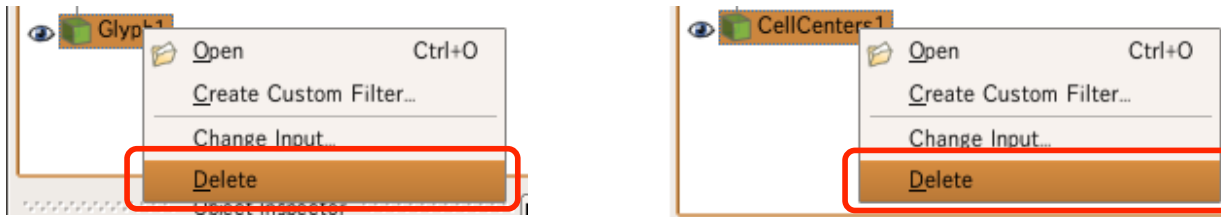


- Custom Filter Managerでは、“Close” をクリック

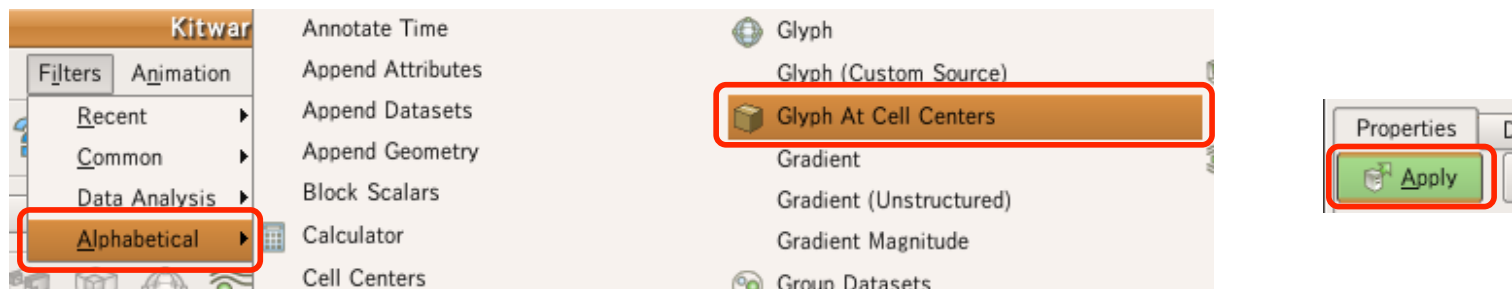


# カスタム・フィルタ: フィルタの使用

- 一旦、Pipeline Browser上のフィルタを削除する
- “Glyph1”、“CellCenters1” の順で、それぞれ選択、右クリック→ “Delete”



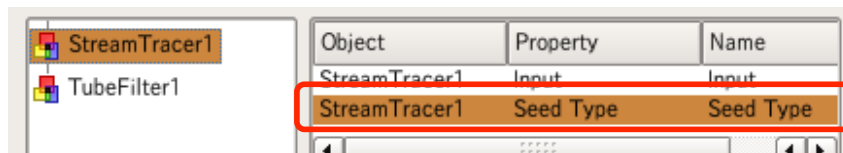
- ParaView最新版では、複数選択+まとめてDeleteが可能
- “Filters...”メニューに追加された “Glyph At Cell Centers” を選択 → “Apply”



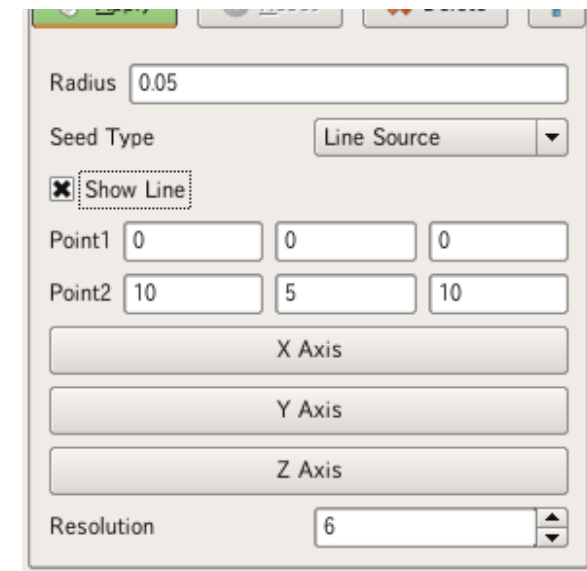
- カスタム・フィルタ作成時にSet Scale Factorプロパティを可変にしたので、このプロパティはカスタム・フィルタ使用時に設定可能。適宜変更してみてください。



- Pipeline Browser上のGlyph At Cell Centers1を削除し、以下のパイプラインの  
カスタムフィルタを作成してみてください。
  - Stream Tracer  → Integration Direction: “FORWARD”、Integrator  
Type: “Runge-Kutta 4”、Seed Type: “Line Source” → “Apply”
  - “Filters” → “Tube”、Radius: 0.05 → “Apply”
- カスタム・フィルタ名: “Tubed Stream Tracer”
- Define the Inputs: 以下のとおり



- Define the Outputs: デフォルトのまま
- Define the Properties: TubeFilter1のRadiusを追加
- 作成したフィルタを適用してみて、右図のプロパティ  
画面になればOK。
- デフォルト値が、パイプライン作成時に設定した値と  
なっていることに注意。(Seed Type、Radius)



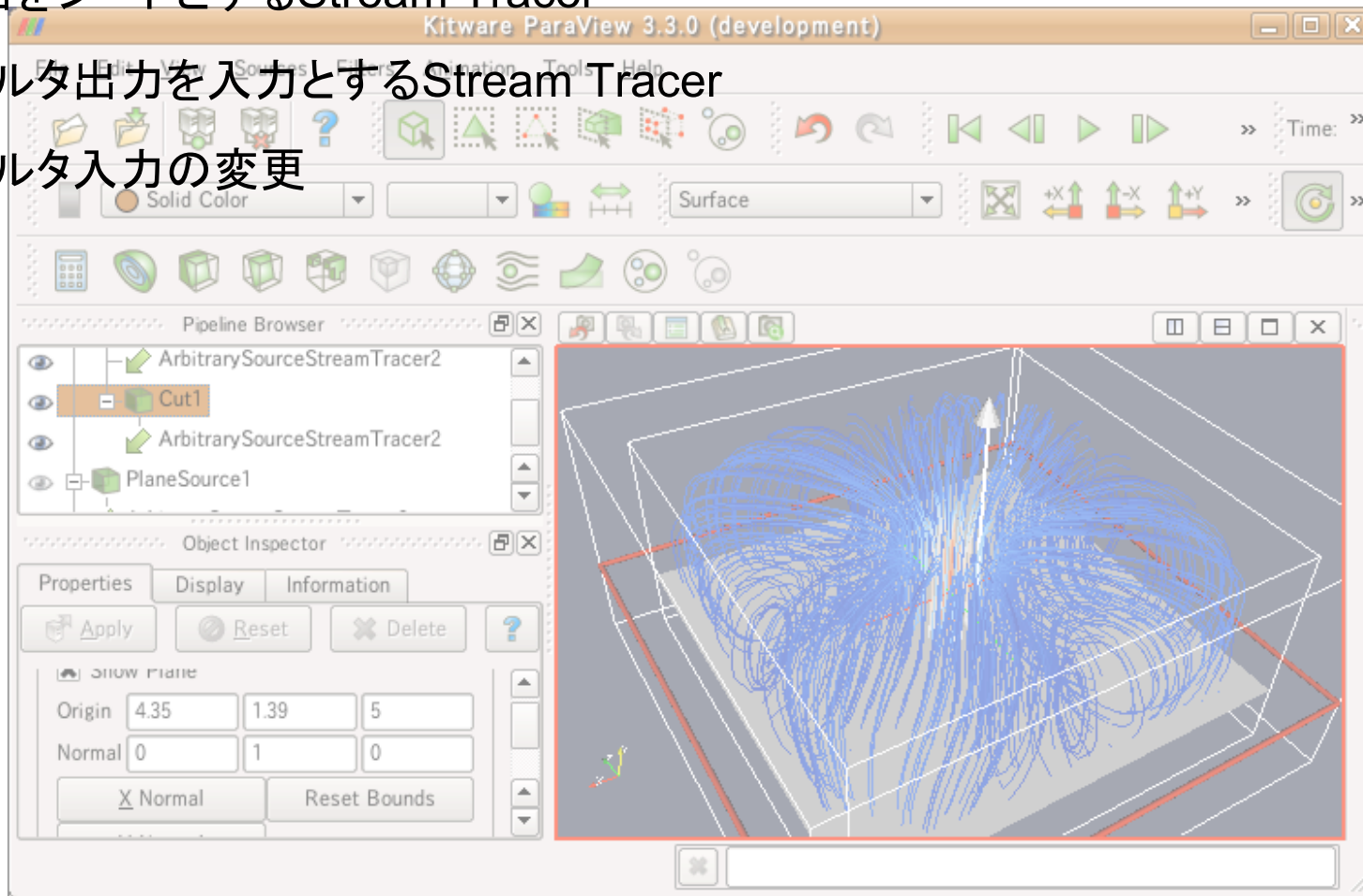
# カスタム・フィルタ: フィルタの使用終了

---

- Pipeline Browser上のhotRoom.OpenFOAM以外のフィルタを全て削除して下さい。
- これで、カスタム・フィルタの演習は終了です。

# 任意シードを用いたStream Tracer

- 平面をシードとするStream Tracer
- フィルタ出力を入力とするStream Tracer
- フィルタ入力の変更



## ポイント

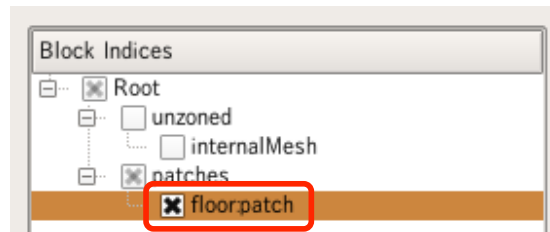
- ツールバーにもアイコンで用意されているStream Tracerは、点群、または線状のシードしか使えない
- それ以外のシードを使いたいときは、Stream Tracer (Custom Source)を使用する
- シードには、Sourcesメニューの各アイテムや、フィルタの出力を使用可能
- フィルタ入力のInputには流れ場のデータを、Sourceにはシードとするソースを指定する
- フィルタ入力の変更は、“Change Input...” で可能

## 例題

- Planeソース + Stream Tracer (Custom Source)
- Sliceフィルタ + Stream Tracer (Custom Source)

# 任意シードStream Tracer: 準備

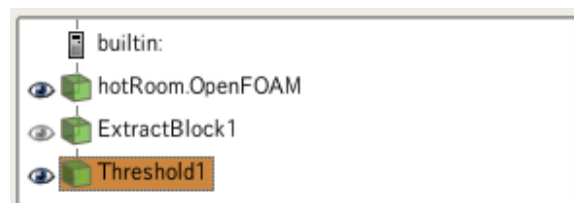
- 発熱面による流れの形成をわかりやすくするため、発熱面(600 K)を抽出する
- “Filters” → “Extract Block” フィルタによって、floor境界面を抽出



- さらにThresholdフィルタによって、 $T = 600$  Kのfaceを抽出、Tで色付け



- Pipeline BrowserのhotRoom.OpenFOAMをVisibleにする

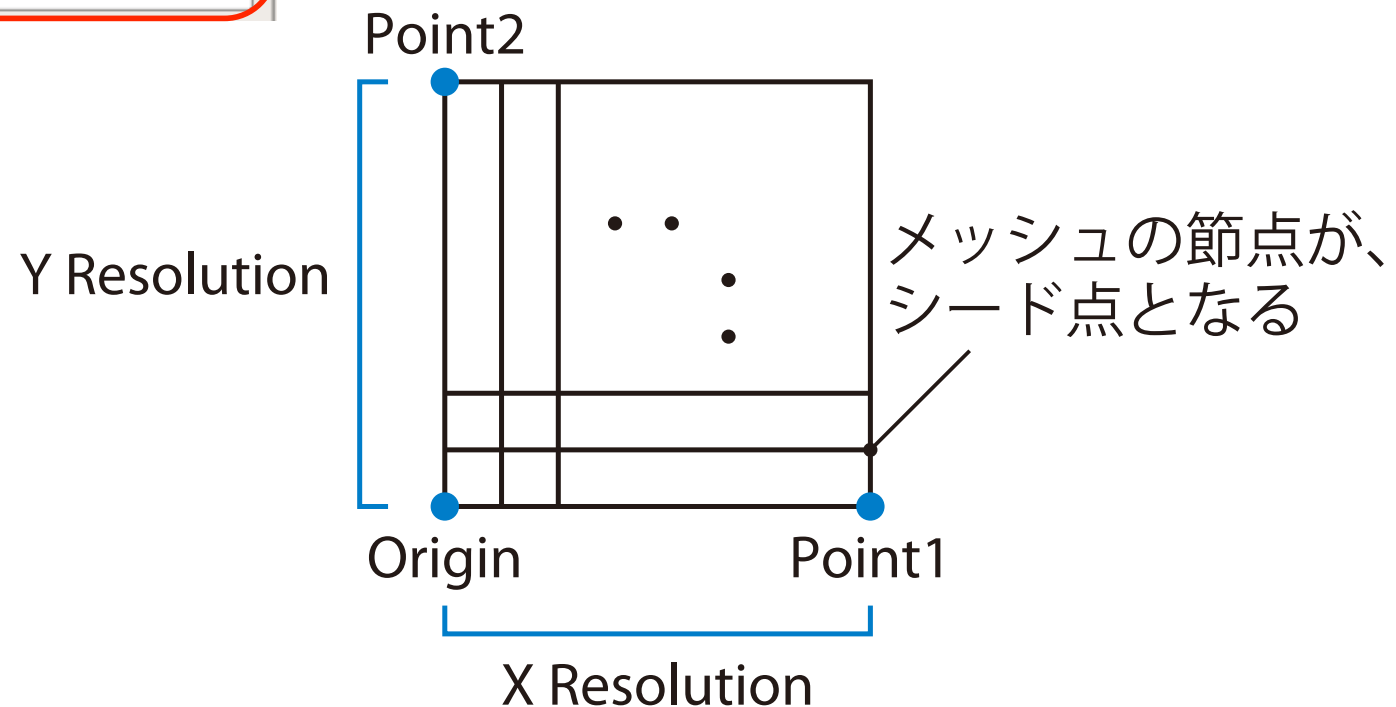




# 任意シードStream Tracer: ソースの作成

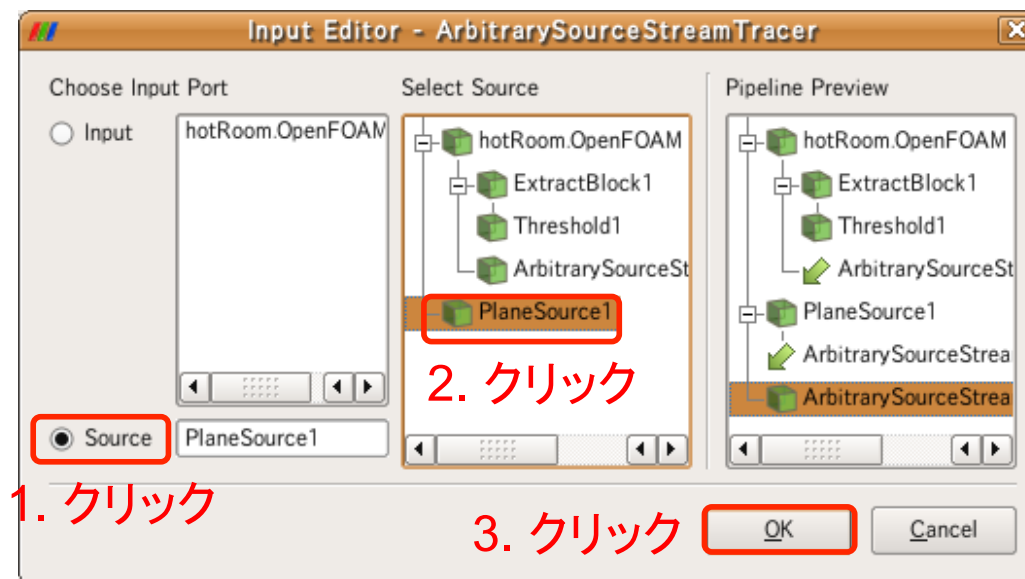
- “Sources” → “Plane” を選択。Origin = (3, 1, 3)、Point1 = (7, 1, 3)、Point2 = (3, 1, 7)、X Resolution = 20、Y Resolution = 20を設定、RepresentationをWireframeに

Origin	3	1	3
Point1	7	1	3
Point2	3	1	7
X Resolution	20		
Y Resolution	20		



# 任意シードStream Tracer: フィルタの実行


- Pipeline Browser上で、hotRoom.OpenFOAMを選択
- “Filters” → “Stream Tracer (Custom Source)” を選択
- Input Editorでは、SourceにPlaneSource1を選択

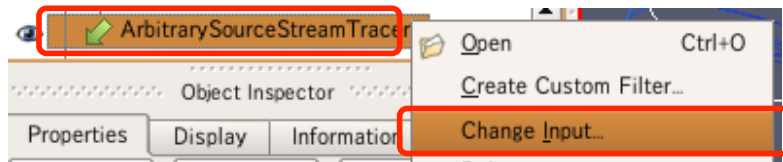


- 流線を見やすくするために、Integration Direction (流線を描く方向): “FORWARD”、Integrator Type (積分の精度): “Runge-Kutta 4”を選択、最後に “Apply”



# 任意シードStream Tracer: フィルタをシードにする

- シードを、Sliceフィルタの出力に変更してみる
- Pipeline Browser上で、hotRoom.OpenFOAMを選択
- Sliceフィルタを選択、“Apply” 
- Pipeline Browser上で、“ArbitrarySourceStreamTracer1” を選択、右クリック→“Change Input...” を選択



- Sourceをクリック、Select Sourceリストの“Cut1” を選択、“OK”

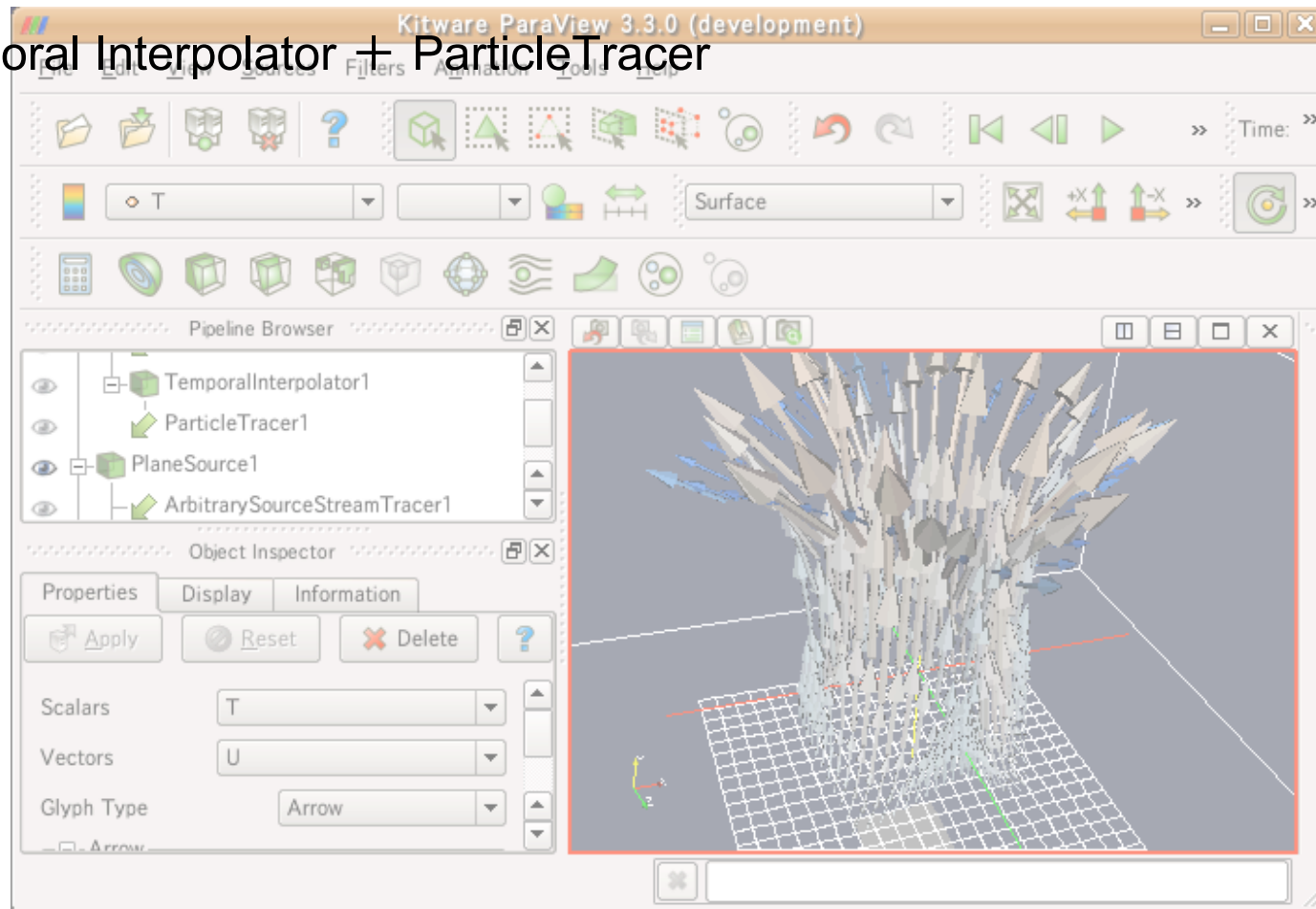


- Pipeline Browser上で“Cut1” を選択し、Slice面を動かすと、それに伴って流線も変わります。

- Pipeline Browser上のArbitrarySourceStreamTracer1とPlaneSource1を削除して、もう一度同じパイプラインを作成してみてください。
  - Plane Source: Origin = (3, 1, 3)、Point1 = (7, 1, 3)、Point2 = (3, 1, 7)、X Resolution = 20、Y Resolution = 20
  - Stream Tracer (Custom Source): Integration Direction: “FORWARD”、Integrator Type: “Runge-Kutta 4”
- Plane Sourceの位置、メッシュ密度(10×10、40×40など)を変えてみてください。

# パーティクルトレース

- Temporal Interpolatorの使用
- Temporal Interpolator + ParticleTracer



## ポイント

- ParaViewは、パーティクルトレースのアニメーションも可能。
- 時間方向のデータ量不足を補うため、Temporal Interpolatorフィルタを使用する。
- フィルタ名はParticleTracer。
- 任意シードStream Tracerと同様、流れ場データとシードとするソースを指定する。
- ParticleTracerの出力を、さらに他のフィルタの入力にできる。

## 例題

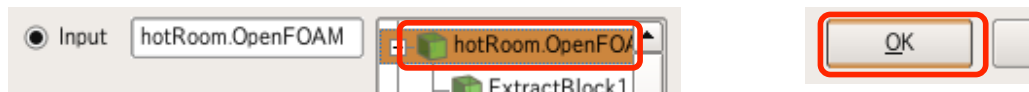
- Temporal Interpolator + Planeソース + ParticleTracer
- Temporal Interpolator + Planeソース + ParticleTracer + Glyph

# パーティクルトレース: Temporal Interpolatorの実行準備

NIIGATA UNIVERSITY



- 元の流れ場データは時刻100毎に保存されているため、そのままでは時間方向のデータ密度が粗い。そのため、Temporal Interpolatorフィルタによって時間方向に補間を行って、時刻1ごとのデータにする。
- ただし、その前にTemporal Interpolatorの問題回避のため、Extract Blockを削除する必要がある。
- Pipeline Browser上でThreshold1を選択、右クリック → “Change Input...” を選択
- Select Sourceリストで hotRoom.OpenFOAMを選択、“OK”



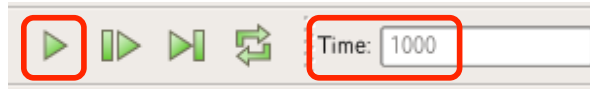
- Pipeline Browser上でExtractBlock1を選択、右クリック → “Delete” を選択

# パーティクルトレース: Temporal Interpolatorの実行

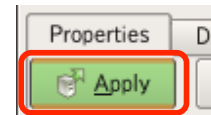
NIIGATA UNIVERSITY



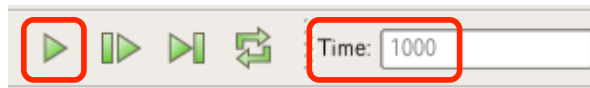
- ここでアニメーション再生を行い、Timeが100ステップごとに増えることを確認



- Pipeline Browser上で、hotRoom.OpenFOAMを選択
- “Filters” → “Temporal Interpolator” を選択
- Discrete Time Step Intervalを1にする → “Apply”



- 再度アニメーション再生を行い、Timeが1ステップごとに増えることを確認



- Pipeline Browser上で、“ArbitrarySourceStreamTracer1”、“Cut1” のVisibleをオフにする

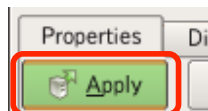


# パーティクルトレース: ParticleTracerの実行

- ここでは、ソースとしてPlaneSource1を再利用する。
- Cut1は使えない (ParaView最新版では改善)
- “Filters” → “ParticleTracer” を選択、以下のようにする



- “Apply”



- アニメーション再生をお試し下さい。



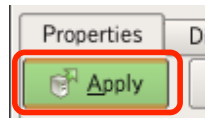
- ParticleTracerは、ソースを2つ使うことができます。ParticleTracer1を削除して、次を行ってみてください。
- “Sources” → “Cylinder”、Resolution: 50、Height: 1、Radius: 3、Center: (5, 1, 5) → “Apply”
- Pipeline Browser上のTemporalInterpolator1を選択後、“Filters” → “ParticleTracer” を選択、Input EditorのSourceにPlaneSource1、Source2にCylinderSource1を選択
- ParticleTracer2のプロパティで、“Enable Source2” をオン／オフして、アニメーション再生を比べてみてください。

# パーティクルトレース: ParticleTracerへのGlyphの追加

NIIGATA UNIVERSITY



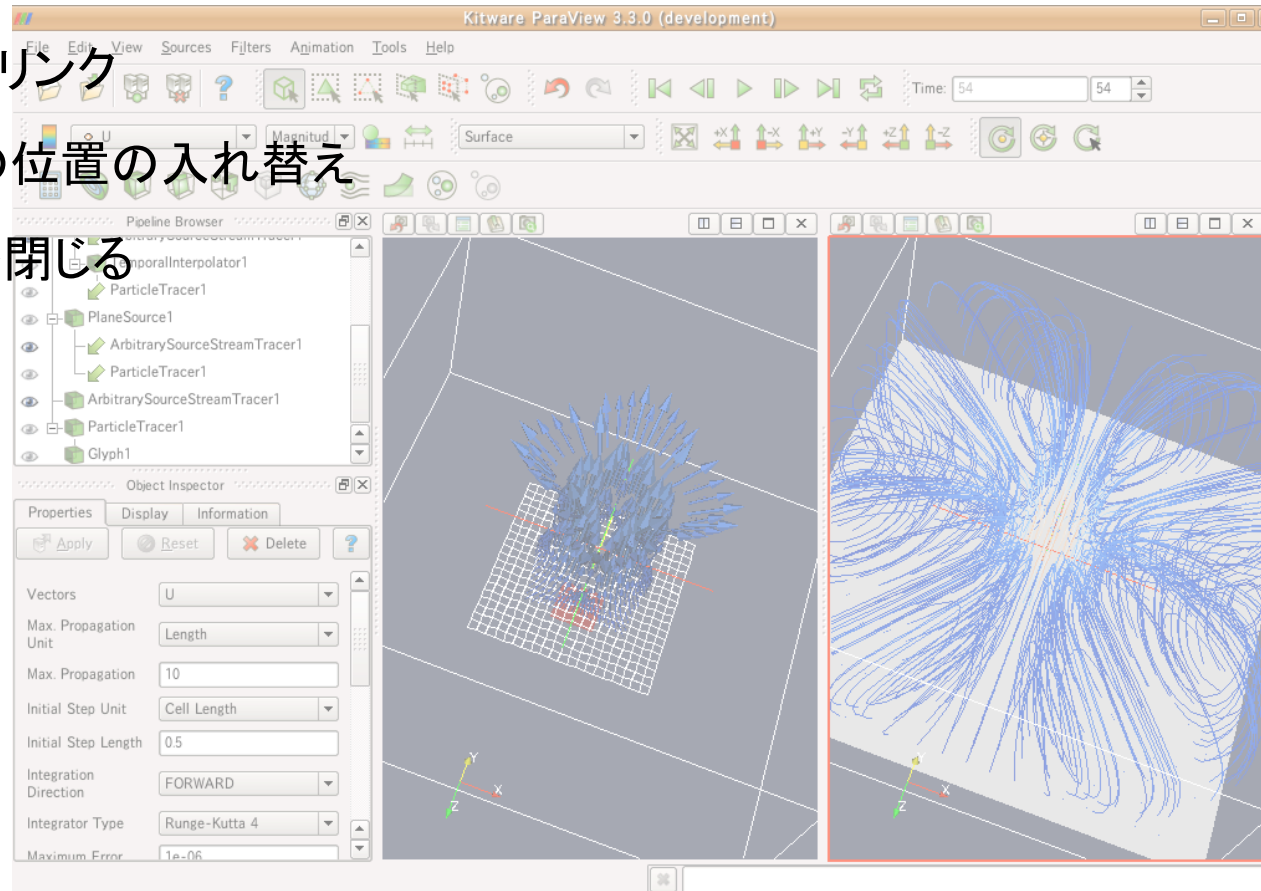
- ParticleTracerの結果を、他のフィルタへの入力にすることも可能。
- Pipeline BrowserでParticleTracer2を選択
- Glyphをクリック、“Apply”



- 定期的にソースにシードが注入されるようにすることもできる。
- Pipeline BrowserでParticleTracer2を選択
- Force ReInjection Every NStepsを100に設定、“Apply”

# マルチビュー

- 複数のビューを、並べて見る
- カメラのリンク
- ビューの位置の入れ替え
- ビューを閉じる

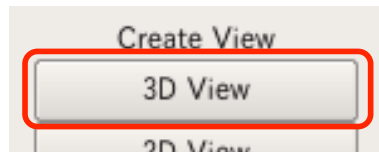


# マルチビュー: ビューの分割

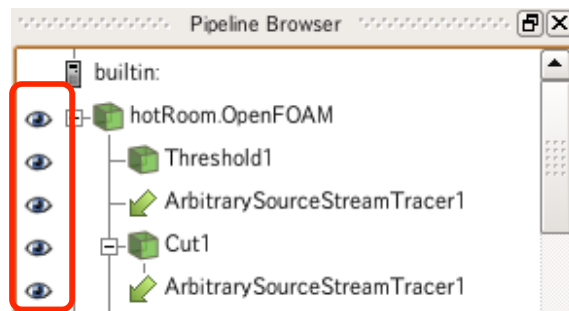
- ParaViewでは、ビュー画面を分割して、複数の可視化ビューを並べて見ることができる。
- ビュー画面右上のアイコンをクリック



- “3D View” をクリック

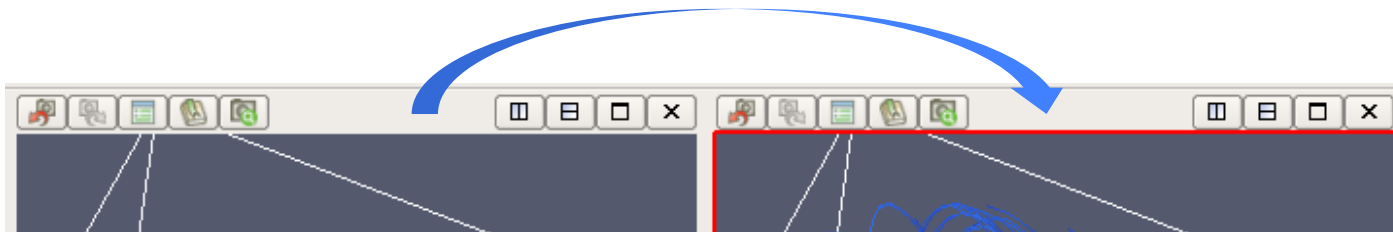


- 空白のビューが開くので、表示したいものをPipeline BrowserでVisibleにする。



- 全体が赤(最新版では青)枠で囲まれているビューに、Visibleの設定が作用する。

- ビュー間で視点を連動させることができる。
  - “Tools” → “Add Camera Link...” を選択
  - または、ビュー上のマウス右クリックで “Link Camera...” を選択可
  - 現在のビューと視点をリンクしたいビューをマウスクリックで選択
- 
- ビューの位置を入れ替えることができる。
  - ビューの上端をドラッグする。



- ビューを閉じるには、ビュー右上端の「×」アイコンをクリックする。

## 今回の講習内容

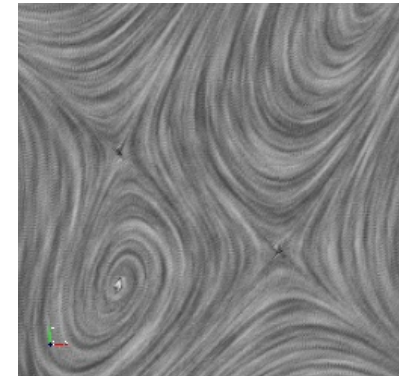
- OpenFOAMクイックスタート
- ParaView可視化演習: フィルタの使い方
  - カスタムフィルタの作成・適用
  - 任意シードを用いたStream Tracer
  - パーティクルトレース
  - マルチビュー

## 今後の計画 (希望次第)

- Pythonスクリプティング (DEXCSが対応次第)
- サーバ・クライアント・モードの使用方法
- ParaViewのカスタマイズ・プラグイン作成方法 (自作readerの作成)
- etc, etc...

## ParaView 3.8

- 昨日現在、3.8 RC(リリース候補)2までリリース。次は正式版と思われる
- GPUベースのボリュームレンダリング
- LIC (line integral convolution): 右図
- Manta (ソフトウェアレイトレーシング)
- フルスクリーン・モード
- “Find Data” (データが指定した範囲にあるセルを探す)
- Parallel OpenFOAM native readerが標準に
  - OpenFOAM 1.5フォーマットまでサポート (1.6独特の機能を使わなければOK)
  - 一部のクライアント側機能(Reload、Watch)省略





## ParaView 3.8以降

- ポリヘドラ(任意多面体)のネイティブ実装
- OpenFOAMのiglooメッシュをClip、左: 従来、右: ポリヘドラ

