粒子系データ可視化ツール
Zindaiji 3

武田隆顕　国立天文台天文シミュレーションプロジェクト

シミュレーションデータ可視化の2つの目的

自分の研究データを素早く見るための可視化
（同じ分野の専門家に見せるための可視化）

レスポンスを速く
物理量を見やすくするため派手派手に
演出は考えない

一般の人に見て理解してもらうための可視化

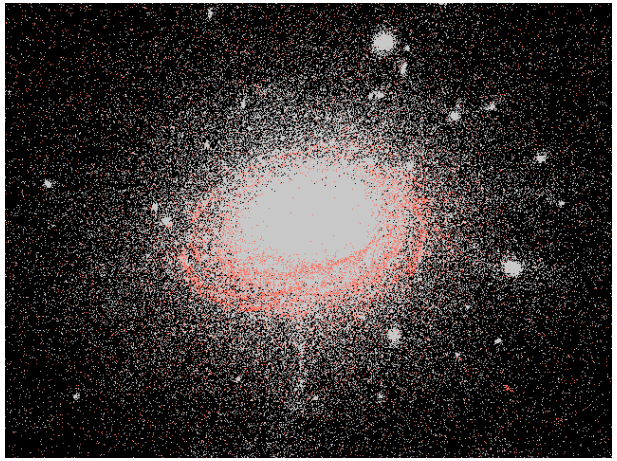
きれいで格好よく
しょぼいとしょぼい印象になる

4D2Uプロジェクト

後者に重きを置いた
一般公開用の映像作成

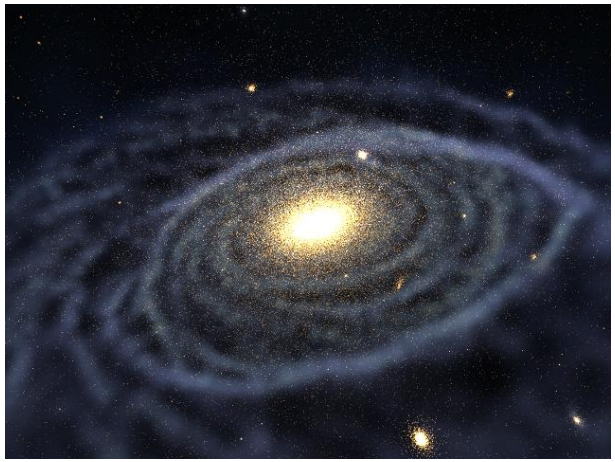
粒子（N体）データ用可視化ツール開発

理想としては…

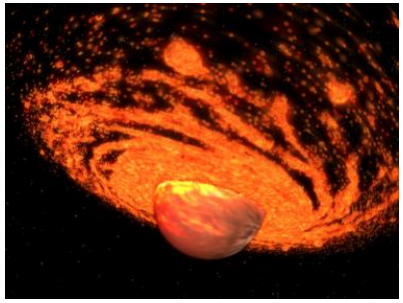


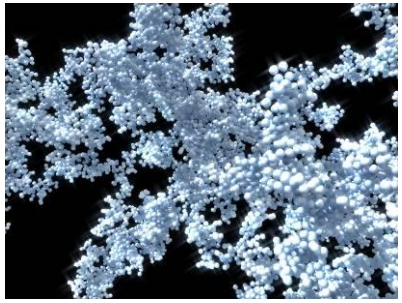
普段は研究用の可視化ツール

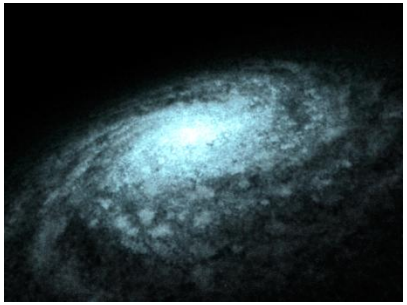
普段隠れている設定を開いて
ぺちぺちと調整



いざというときは華麗なレンダリング

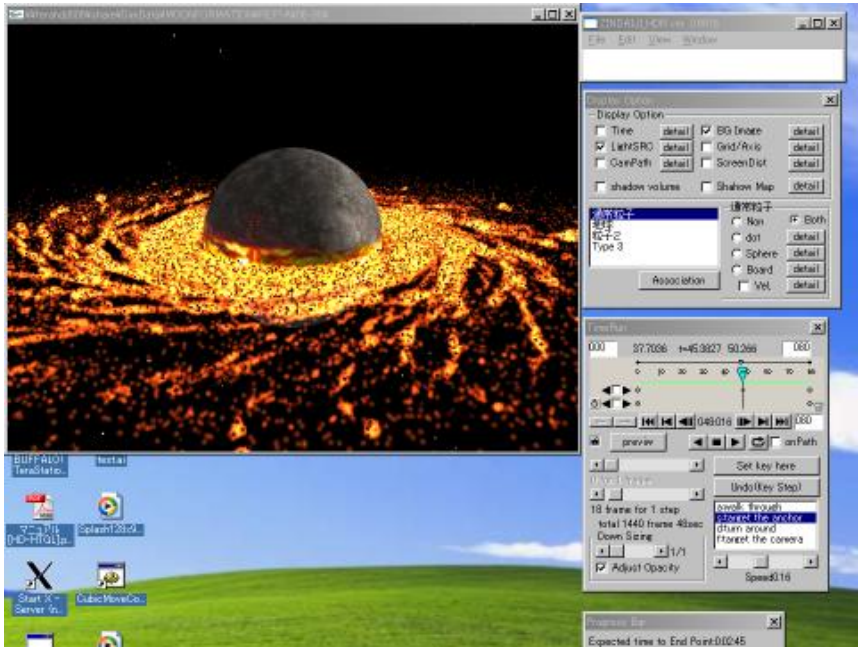






旧ZINDAIJI

２年前に紹介。今まで主力だった。

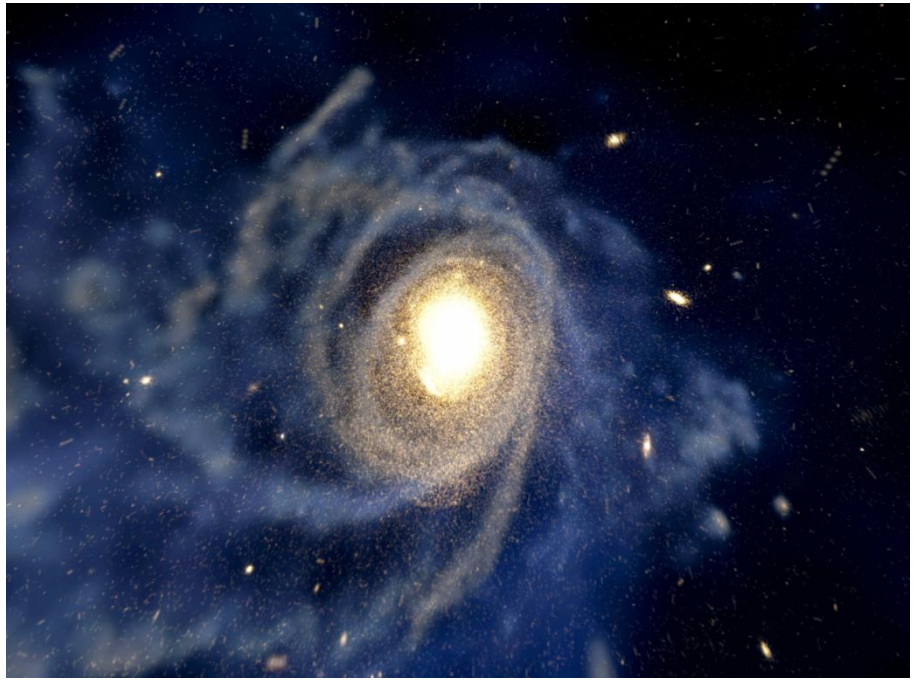


1.GUIで視点移動

2.タイムライン制御

3.補間機能つき

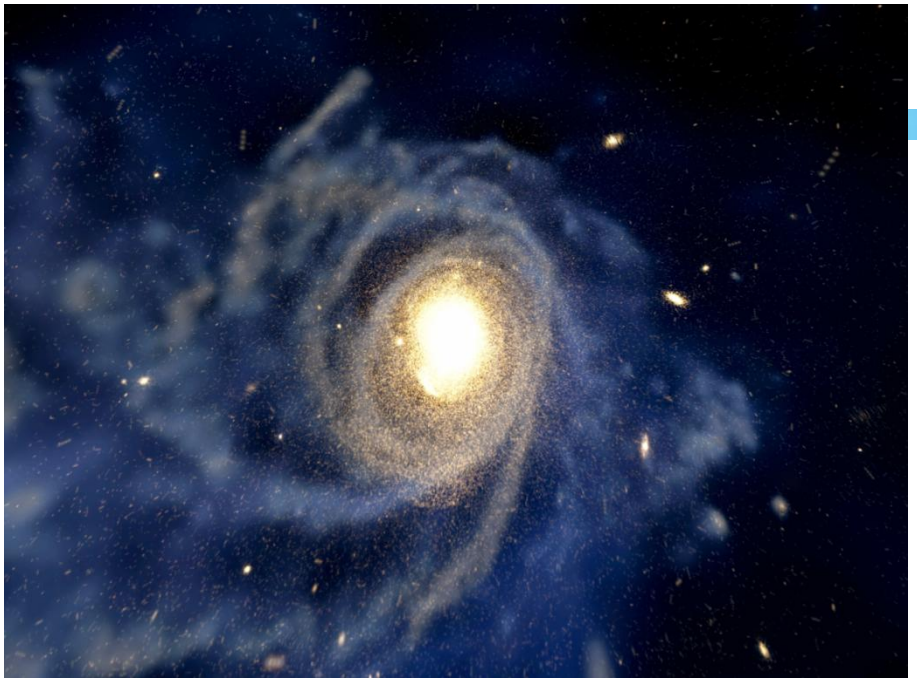
4.凝った絵も出せる



欠点

Windows用の32bitアプリ

GUIがいまいち



メモリ使用量の問題

最低限でも位置速度その他で
一粒子60 byte は必要なので、
100万粒子だと60MB

時間進化を扱うには
せいぜい200万粒子が限界

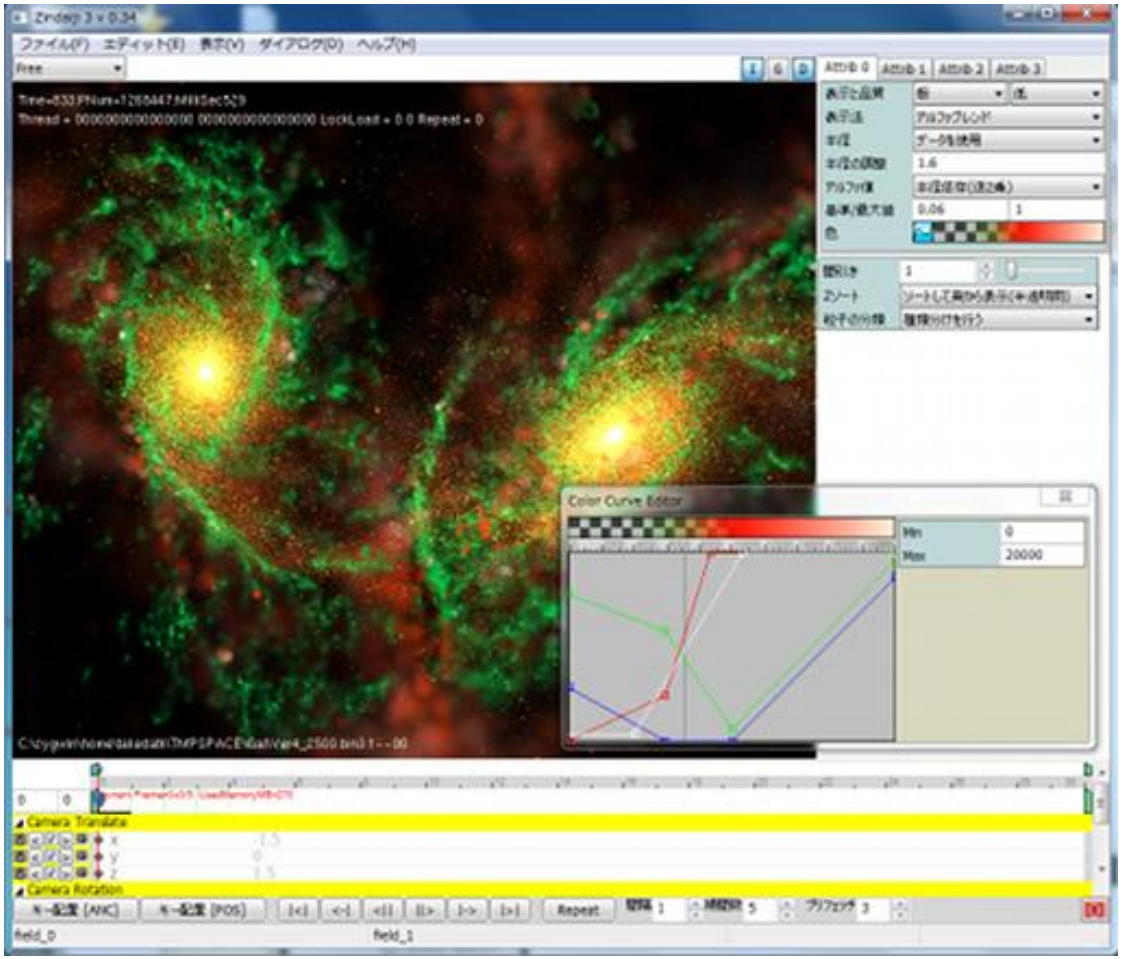
x, y, z
vx, vy, vz
ax, ay, az
jx, jy, jz
radius
type, ID

(ax, ay,az, jx, jy, jx は、補間に必要な変数で、
x,y,z, vx, vy, vzから生成
他の温度などといった情報は、必要ならば追加して保持する)

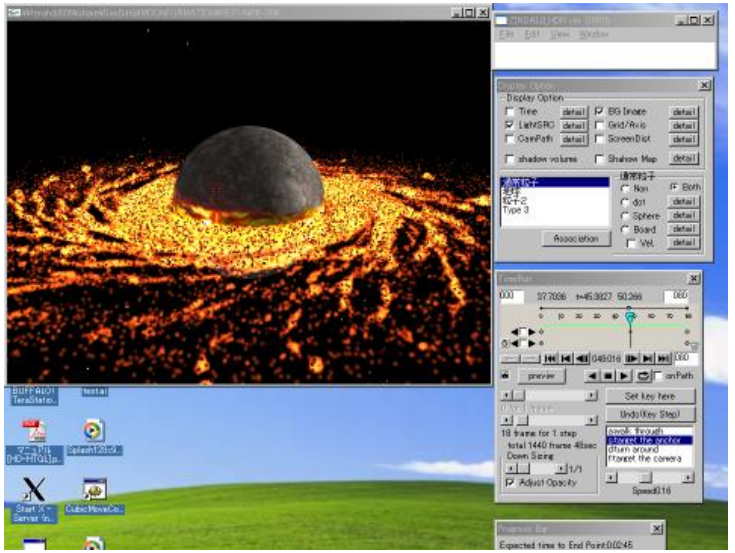
他の情報はグループごとにまとめて保持する。

ZINDAIJI 3

■開発中画面



旧ZINDAIJI



ZINDAIJI 3 の機能向上点

■64bit対応でメモリの制限が大幅に緩和

2000万粒子以上O.K.

■描画部分のアルゴリズム改善

同じ条件で大体倍の速度

■待ち時間の大幅削減

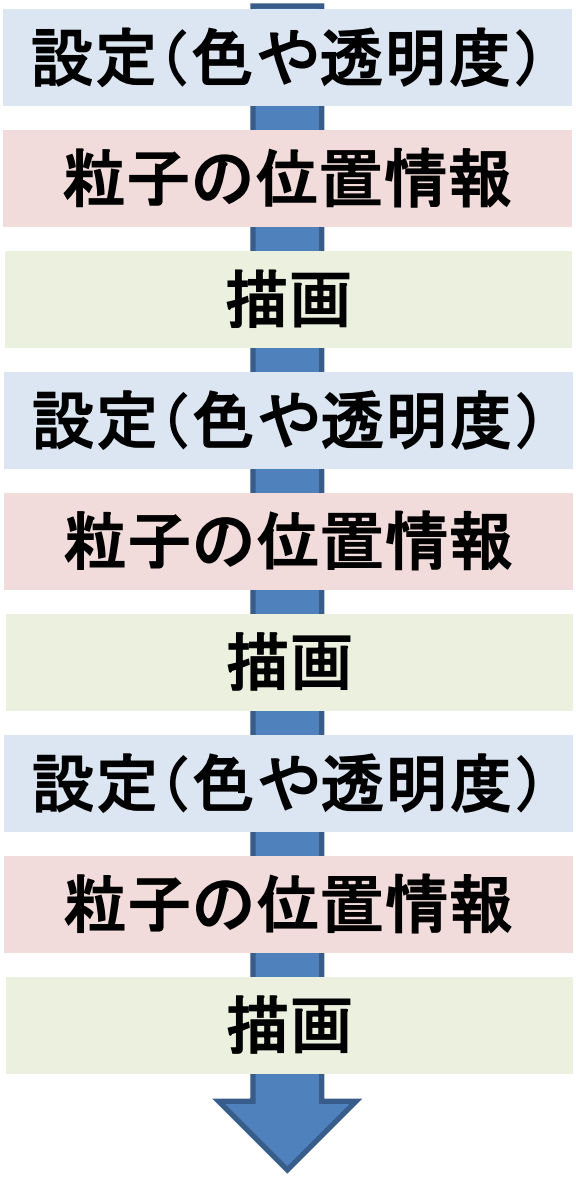
マルチスレッド化

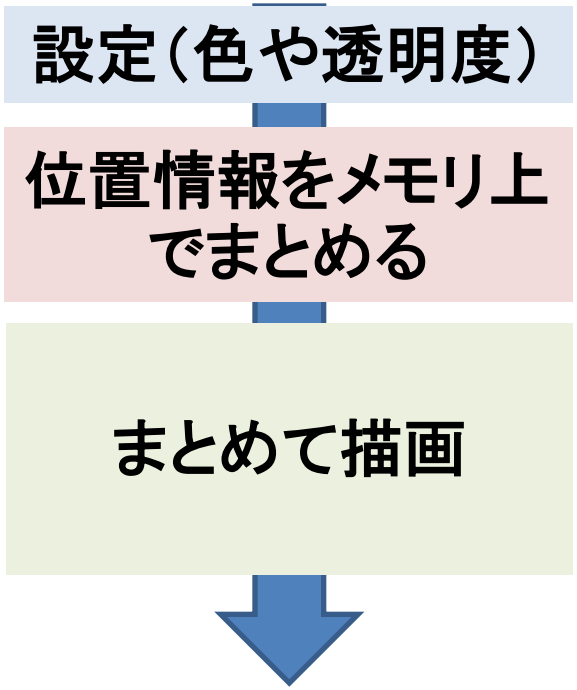
■GUIの大幅改善

■マルチプラットフォーム

Windowx/Linux/MacOSX

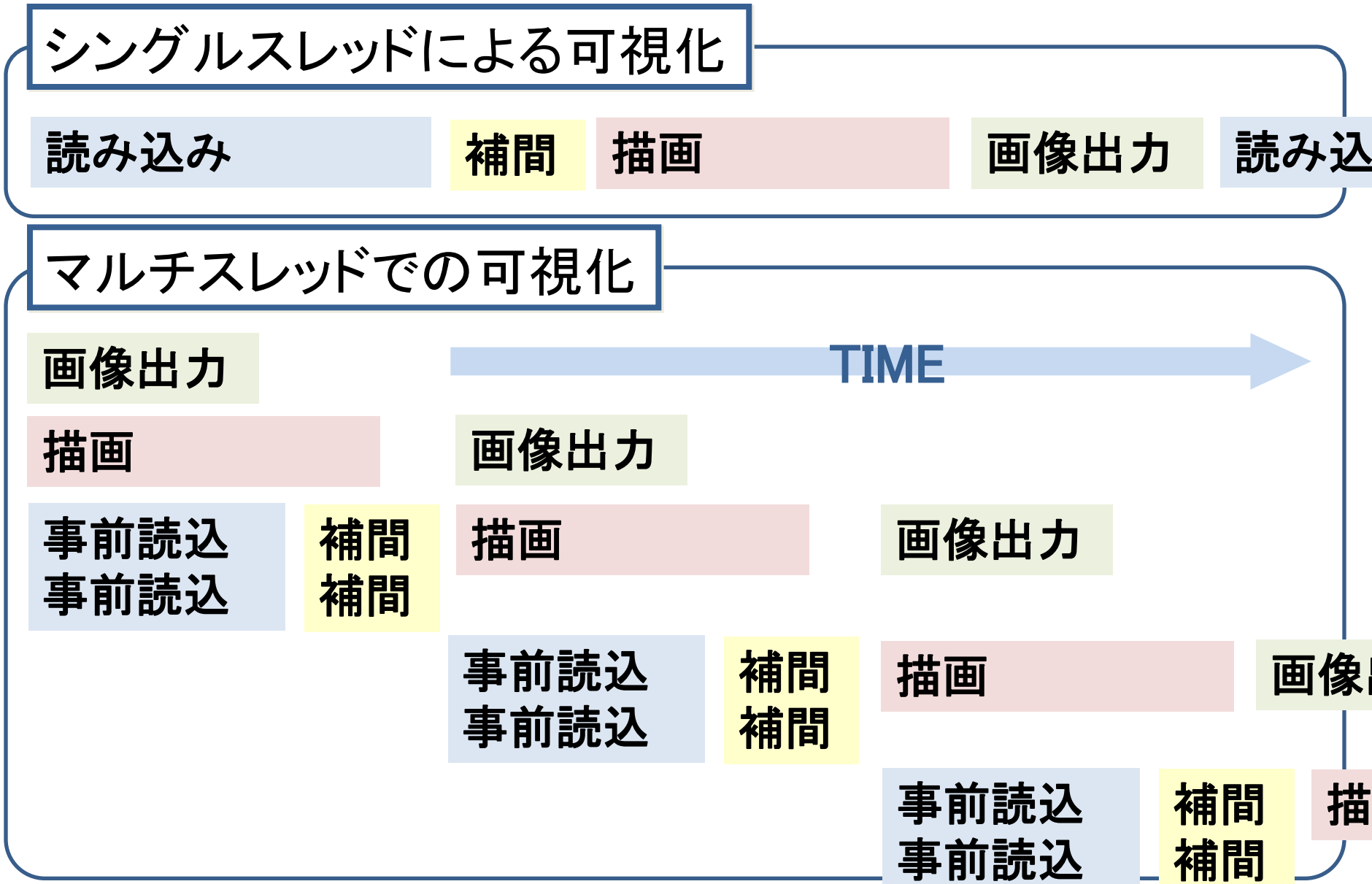
描画アルゴリズム改善





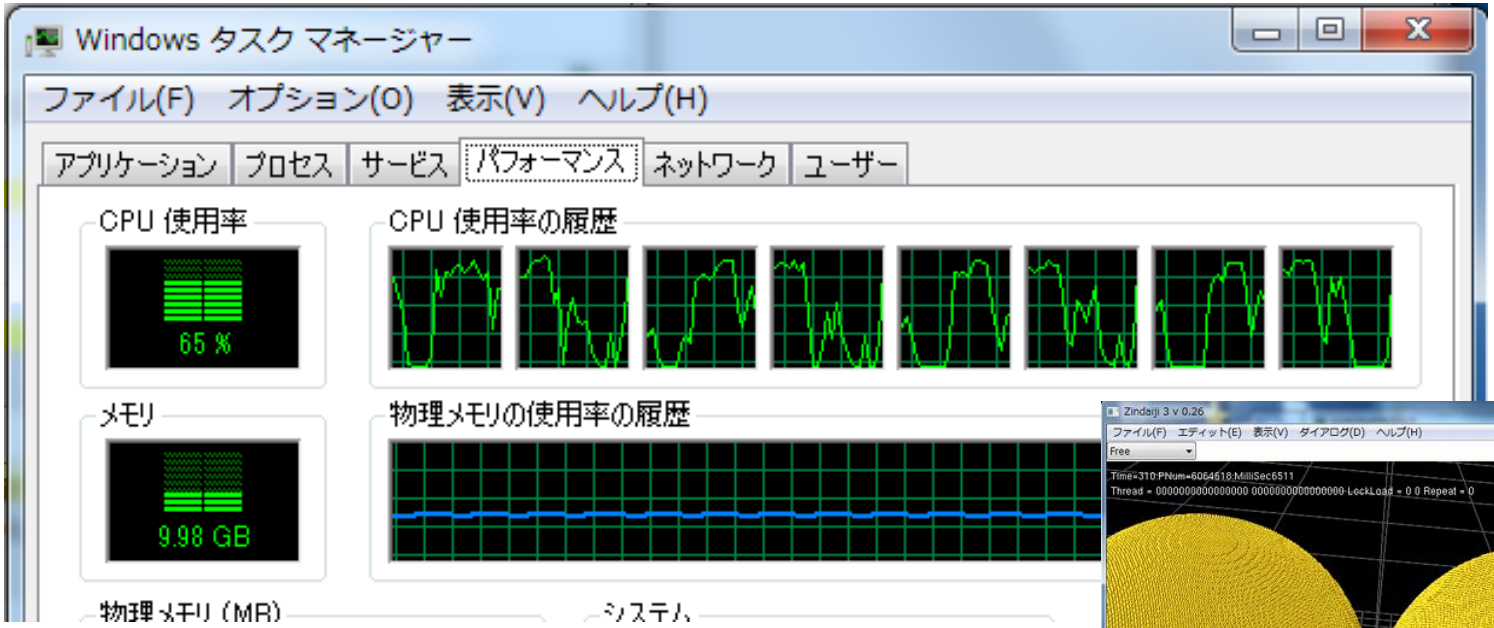
■状況に応じた場合分けで描画の高速化

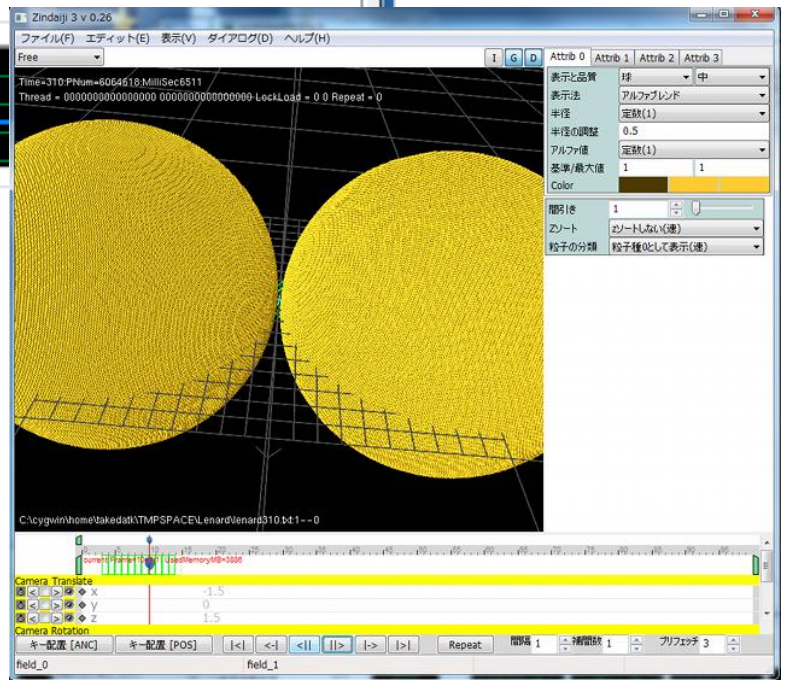
待ち時間の大幅削減


シングルスレッドによる可視化
読み込み
補間
描画
画像出力
読み込
マルチスレッドでの可視化
画像出力
TIME
描画
画像出力
事前読込
事前読込
補間
補間
描画
画像出力
事前読込
事前読込
補間
補間
描画
画像
事前読込
事前読込
補間
補間
描

待ち時間の大幅削減

■600万体の分子動力学データ(ASCII)から作成

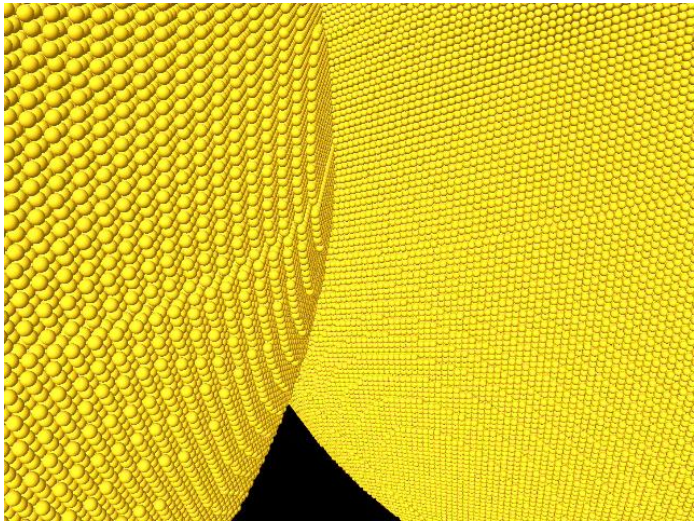




4コア8スレッドのCorei7の
CPU使用率が半分程度

2000万粒子データから映像化

(田中秀和さん(北大低温研)による分子動力学によるダスト衝突)



■600万体のデータ(亀裂が走る様子)

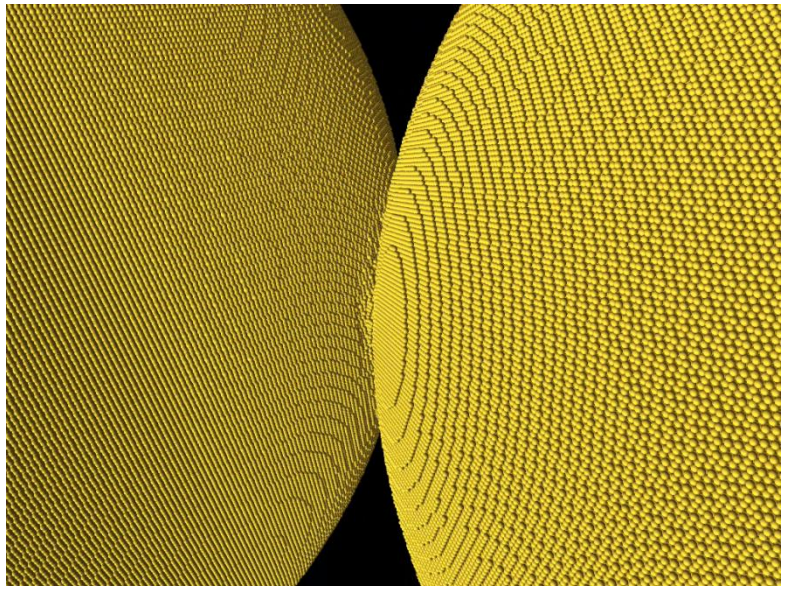
作成環境
Corei7 975(4core-8thread 3.3GHz)
GeForce GT240
(比較サイトスコア10672消費電力69W)

GTX580だと(スコア24554消費電力244W)

■2000万体のデータ

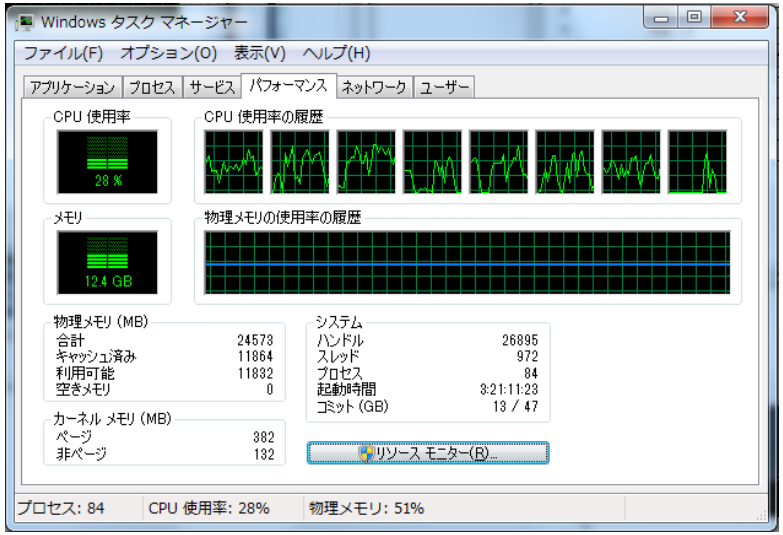
1ステップ481MBのASCIIデータ

描画に12秒(補間5コマで60秒)
読み込み(約30)秒は並列で処理



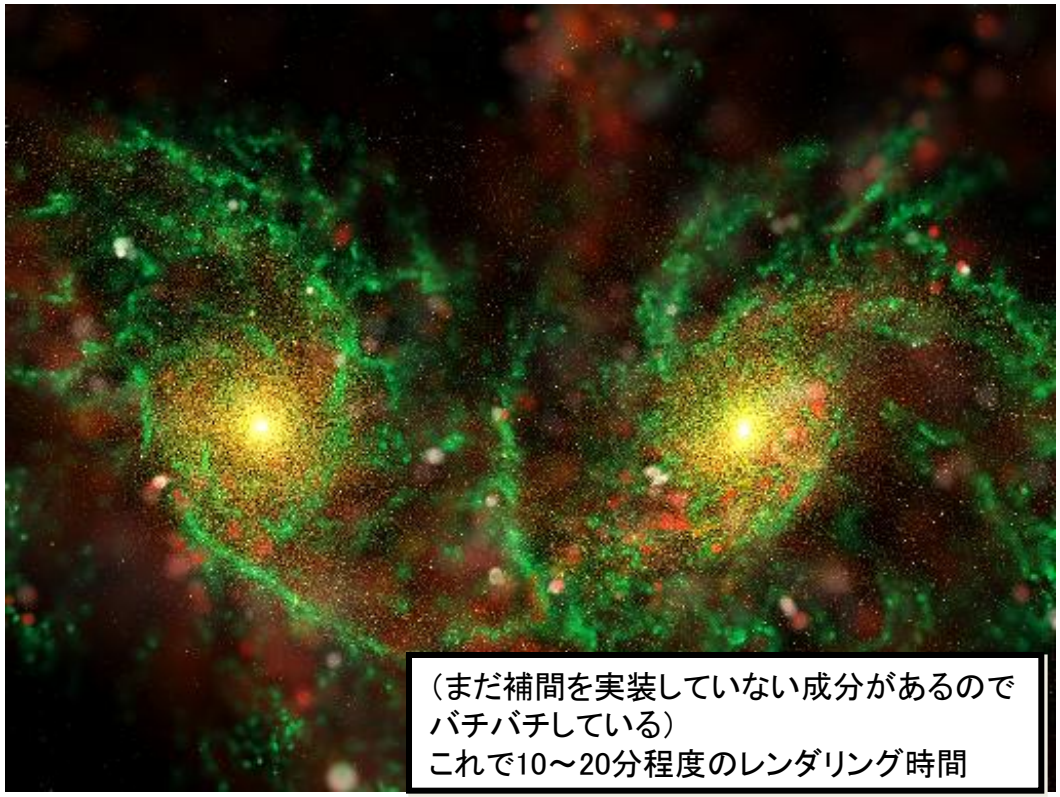
100～150万体の粒子データから映像化

■松井秀徳さんの
100万体強の銀河衝突データ(Binary)から作成



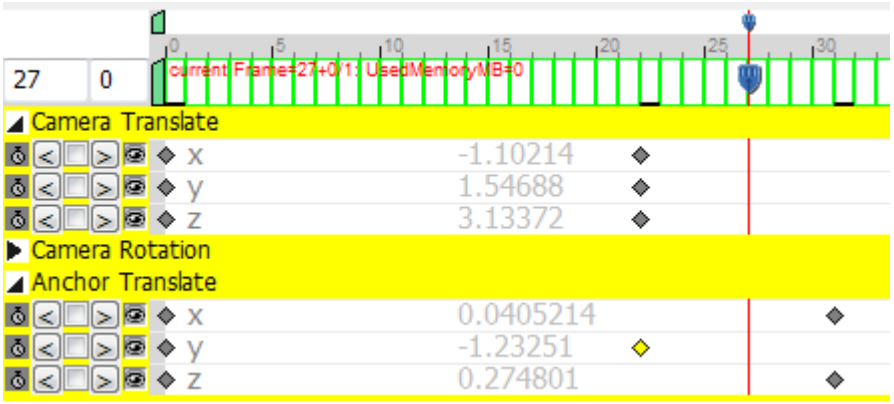
描画に0.25～0.35秒
読み込みや画像圧縮を
並列処理

8コア中の25～35%程度
(2～3コア相当)を使用

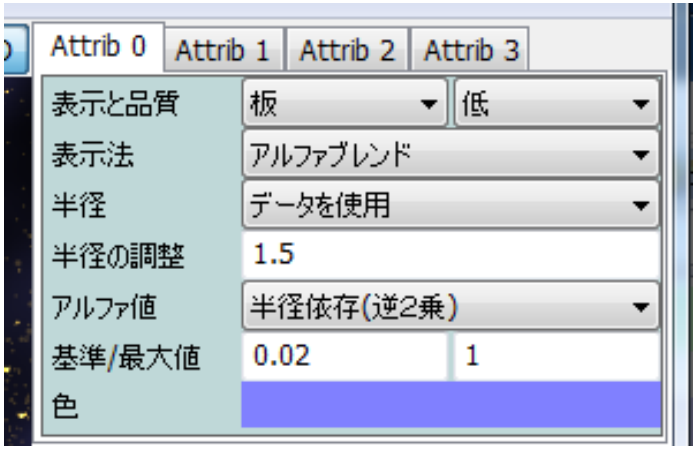


(まだ補間を実装していない成分があるので
バチバチしている)
これで10～20分程度のレンダリング時間

GUI大幅改定



■大きいタイムライン



■設定用パネルを統合

■UNDO機能の充実(行った操作をスタック)

Ctrl+Y

値をOldからNewに変更

キーフレームを移動

キーフレームを作成

オブジェクトを作成

値をNewからOldに変更

キーフレームを(逆に)移動

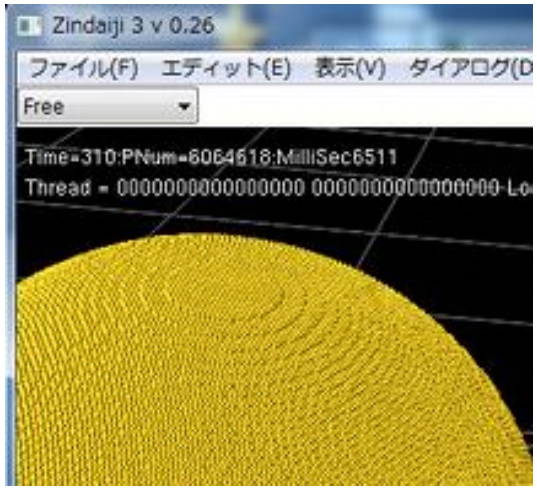
キーフレームを削除

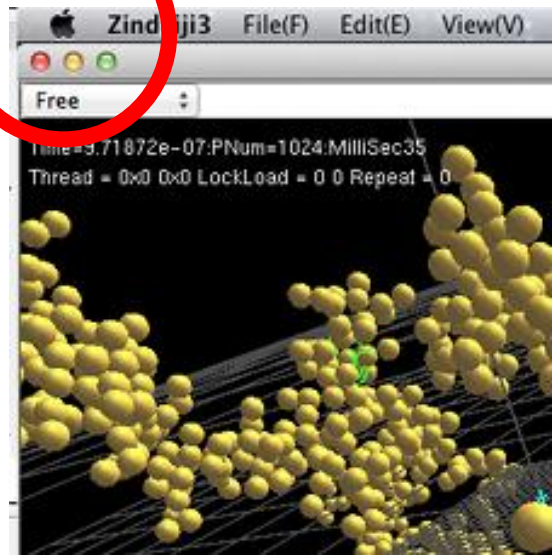
オブジェクトを削除

Ctrl+Z

マルチプラットフォーム

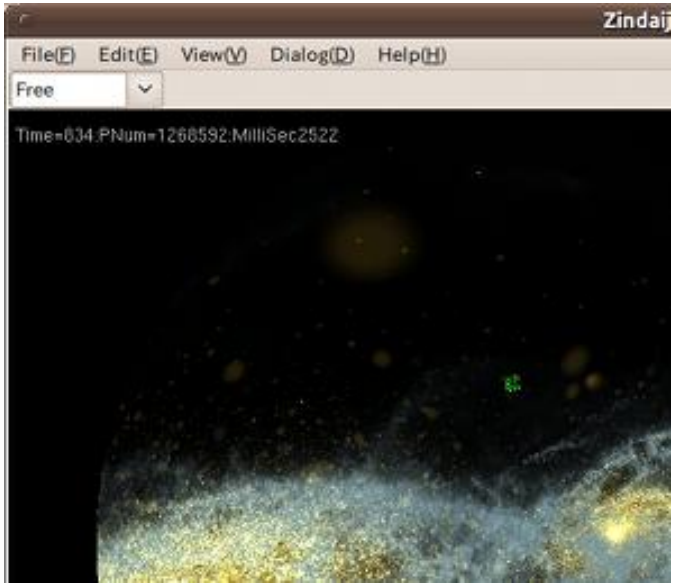
Windows





MacOSX

LINUX



■GUI部分に
マルチプラットフォームの
ライブラリ(wxWidgets)を利用

公開URL

■開発途中でも順次アップデートしながら
公開する方針で

http://th.nao.ac.jp/~takedatk/COMPUTER/ZINDAIJI3/Zindaiji3Top.html

■1/15 ver0.30として公開
…
■2/13 ver0.37として公開

今後の開発

旧Zindaijiにあってver0.30時点で無い機能

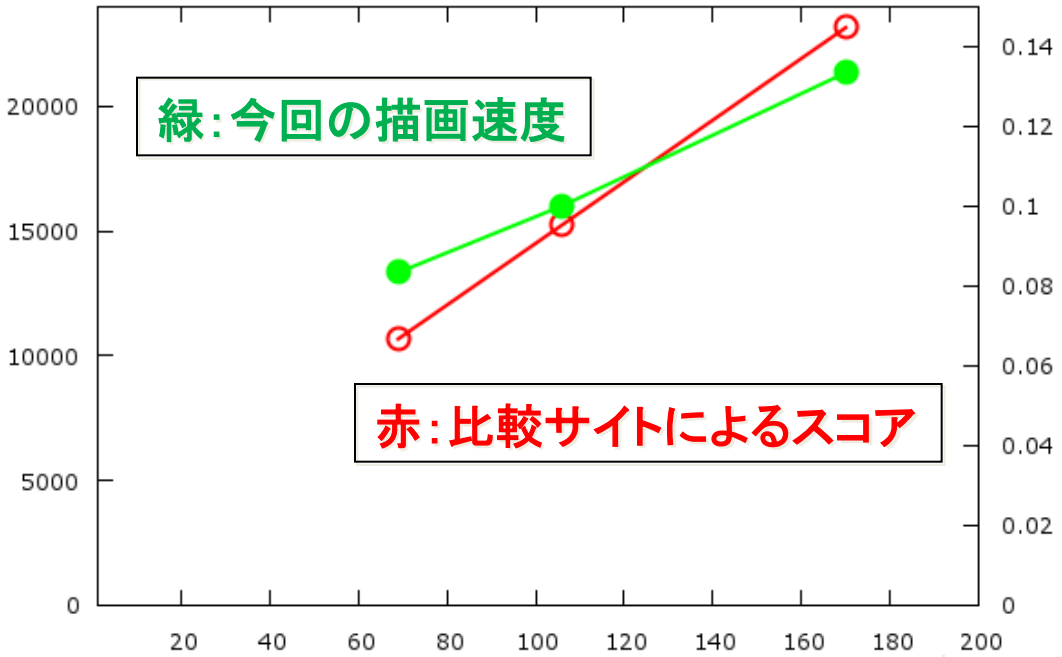
- 温度などに応じた粒子の色(**済**)
- 読み込むデータの文法編集機能(**済**)
- 影の描画や背景の描画、モーションブラー等

追加でつけたい機能

- 周期境界データのコピーなど
- データ以外のオブジェクト表示
- カラーカーブやレジェンドなどFigure用の表示機能
- ネットワークレンダリングで億の単位へ

2000万粒子データから映像化(参考データ)

GeForce GT240
(スコア10672消費電力69W)
12秒
GeForce GTS450
(スコア15237消費電力106W)
10秒
GeForce GTX560
(スコア23138消費電力170W)
7.5秒



(比較サイトによる)消費電力

1ステップ481MBのASCIIデータ

読み込み(約30)秒

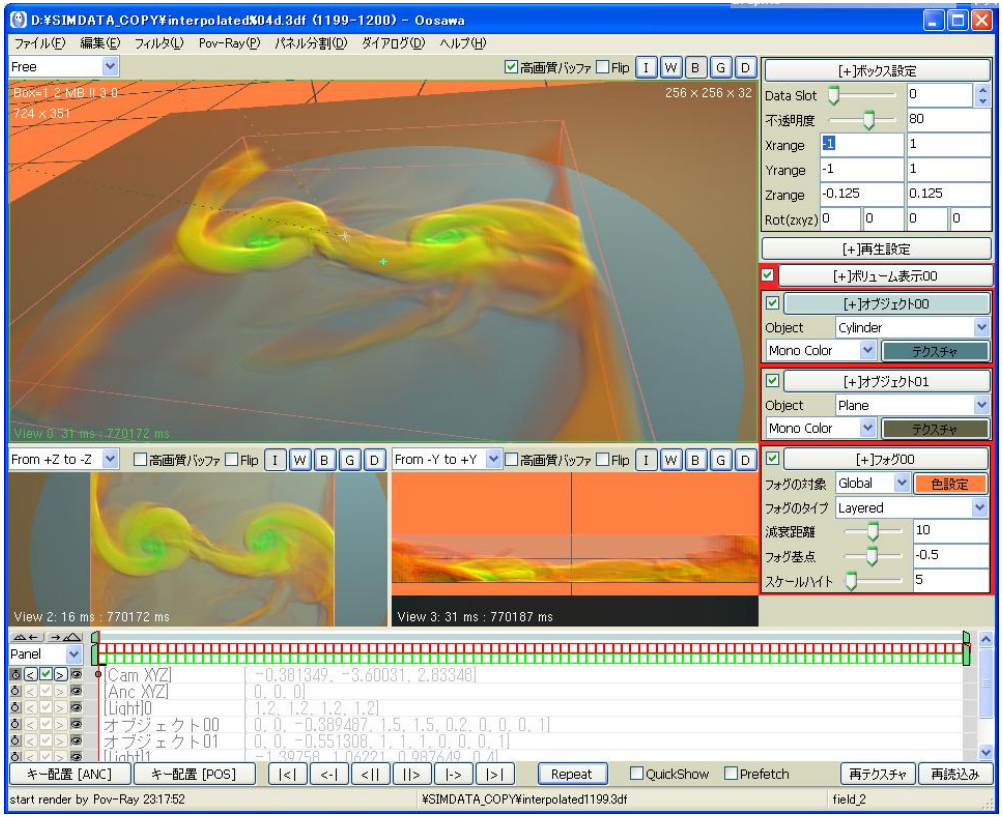
1ステップ231MBのBinaryデータ

読み込み(約3～4)秒

高くて熱いグラフィックボードにしてもあまり効率は良くないかも…？

Oosawa

■時系列ボリュームデータ用の可視化ツール



■GUIの部分や、事前読み込み機能などは
Oosawa開発中に発展させたもの