計算工学講演会論文集　Vol.28 (2023年5月)　　計算工学会

ゲームエンジン用可視化フレームワーク VisAssetsの開発

Development of VisAssets: a Visualization Framework for the Game Engine

川原慎太郎[1)]，宮地英生[2)]，伊藤貴之[3)]，田中覚[4)]，樫山和男[5)]
Shintaro Kawahara, Hideo Miyachi, Takayuki Itoh, Satoshi Tanaka and Kazuo Kashiyama

1) 博(工) 海洋研究開発機構（〒236-0001 神奈川県横浜市金沢区昭和町3173-25, E-mail: kawahara@jamstec.go.jp）
2) 博(工) 東京都市大学 メディア情報学部 教授（〒224-8551 横浜市都筑区牛久保西3-3-1, E-mail: miyachi@tcu.ac.jp）
3) 博(工) お茶の水女子大学 基幹研究院 教授（〒112-8610 東京都文京区大塚2-1-1, E-mail: itot@is.ocha.ac.jp）
4) 理博 立命館大学 情報理工学部 教授（〒525-8577 滋賀県草津市野路東1-1-1, E-mail: stanaka@is.ritsumei.ac.jp）
5) 工博 中央大学 理工学部 教授（〒112-8551 東京都文京区春日1-13-27, E-mail: kaz@civil.chuo-u.ac.jp）

We are developing an open-source visualization framework named VisAssets that runs on the game engine Unity. One of the features of this framework is that visualization applications can be developed simply by connecting small modularized visualization elements. VisAssets was targeted at structured grid data. However, it is now being extended to unstructured grid data, particle data, and data for information visualization. This paper presents an overview of VisAssets and its development status.

Key Words: *Visualization, Framework, Game Engine*

1.　はじめに

ゲームエンジンUnityで動作する，オープンソースの可視化フレームワークVisAssets[1]の開発を進めている．モジュール化された可視化の小要素を階層的に接続するだけで，可視化アプリケーションを開発できるのが本フレームワークの特徴である．当初は構造格子データを対象として開発を開始したが，現在はそれ以外のデータ形式にも適用範囲を広げつつある．本稿ではVisAssetsの概要と，現在の開発状況について報告する．

2.　可視化フレームワークVisAssets

安価なヘッドマウントディスプレイ（HMD）の登場により，CAVE[2]のような大型かつ高額な機器を用いなくても没入感の高い立体映像を提示できるようになった．CAVE, HMDともに，その上で動作するアプリケーションの開発は旧来のプログラミング言語でも可能ではあるが，開発効率やメンテナンス性の面から「ゲームエンジン」と呼ばれるソフトウェアを用いるのが現在では主流となっている．計算科学分野においても，シミュレーション等により得られたデータの可視化結果をHMDで提示する事例が増えており，その多くでゲームエンジンが用いられている．これは，3-Dモデルデータ化された可視化結果をゲームエンジンにインポートするだけでHMDに表示できるためであるが，この場合可視化条件を変更する度にモデルデータの差し替えが必要となる．一方，対象となるデータの読み込みから可視化アルゴリズムの適用までの全工程をUnity上で行う場合，それらの機能はユーザ自身がプログラミングにより実装しなければならない．

ゲームエンジンへの可視化機能の実装の例としては，VTK[3]のUnity対応版であるActiViz for Unity[4]があるが，内部でVTKのWindowsDLLを利用する仕組みのため，AndroidベースのOSを搭載する多くのスタンドアロン型HMDでは動作しない．Windowsでの利用においても，VTKの機能を利用するためにはユーザ自身によるプログラミングが必要な点も導入のハードルとなっている．

これに対し，VisAssetsではデータ読み込み，フィルタリング，マッピングの三種に大別される可視化フローの小要素をモジュールとして提供している．これらをUnityのビジュアルエディタ上で適切に接続することで，プログラミングレスで可視化アプリケーションを作成できる．また，特定のプラットフォームに依存する機能を用いていないため，スタンドアロンHMDの他，スマートフォンでも動作する．図-1はVisAssetsによる可視化の様子を示したものである．本図下部にあるアイコン群がVisAssetsにおける可視化モジュールであり，これらを適切な親子関係で接続することで可視化フローを構築できる．

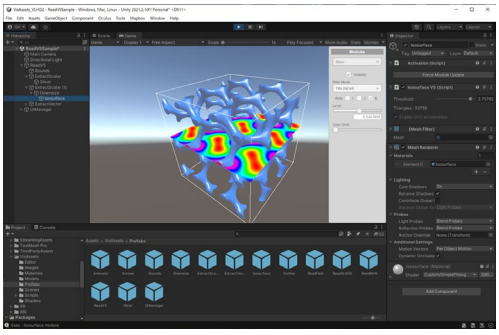

図-1 VisAssetsによる可視化の様子

3. さまざまなデータ形式への対応の試み

VisAssetsの公開パッケージに含まれるサンプルモジュールは，いずれも構造格子データを対象として設計されたものである．読み込みモジュールは指定したデータを内部データクラス内の構造格子用変数に格納し，フィルタリングおよびマッピングモジュールはそのデータクラスに対して可視化アルゴリズムを適用する．構造格子以外の入力データに対しても，この内部データクラスの拡張により柔軟に対応することができる．VisAssetsの核となるのは親子関係にある可視化モジュール間の接続可否判定と状態遷移であり，これは入力データの形式に依らず機能する基本的な動作原理であるためである．

ここでは，構造格子以外のデータ形式への対応例として，全く異なるデータ構造を持つ非構造格子，点群，情報可視化データの三種を対象とし，それらをVisAssetsの枠組み内で扱うための試みについて紹介する．内部データクラス内でのデータ保持方法の規定が最も重要な工程となるため，現時点ではいずれもテスト実装段階である．

(1) 非構造格子データ

既存のデータ形式（VTK, AVS/Express）を参考に，非構造格子データを保持できるよう内部データクラスを拡張し，AVS/Express用非構造格子データの読み込みと外形線の表示を行うモジュールをテスト実装した（図-2）．本図では格子点番号に応じた色を設定している．

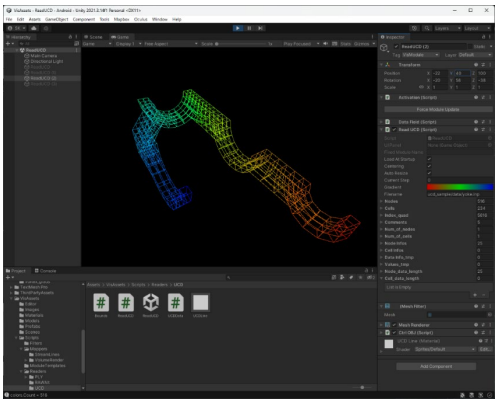

図-2 非構造格子データの可視化例

(2) 点群データ

点群データを保持できるよう内部データクラスを拡張し，PLY形式データの読み込みと表示を行うモジュールをテスト実装した（図-3）．本図では500万点超の点群表示の他，同じ領域の構造格子データを既実装の同データ用モジュールを，地図表示にはMapBox[5]を用いている．このように，他のオブジェクトとの同時利用が容易にできるのもゲームエンジンを用いる利点の一つである．

他のデータ形式においても同様ではあるが，描画対象（この場合は点群）が大量となる場合は描画負荷が非常に高く，PCでの実行時には問題がなくても，スタンドアロン型HMDでの実行時には点群数の相当な削減が必要であった．

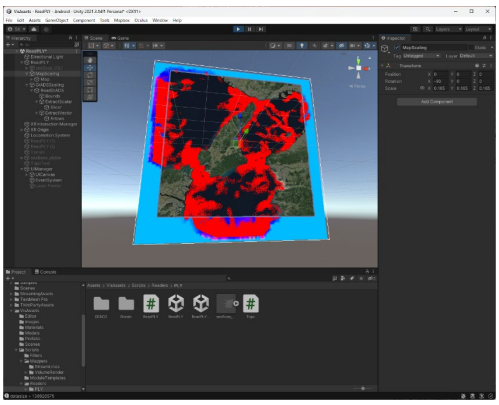

図-3 点群データの可視化例

(3) 情報可視化データ

情報可視化手法である平安京ビュー[6]のデータの読み込みと表示を行うモジュールをテスト実装した．専用のデータ形式であるため，内部データクラスの構造や描画ルーチンの実装においては，Javaで記述されたオリジナル版の平安京ビューのソースコードをほぼそのままC#で移植することで対応し，オリジナル版の平安京ビューの機能をほぼ実装することができた．

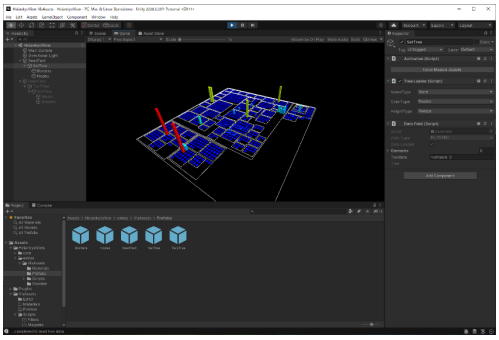

図-4 「平安京ビュー」データの可視化例

4. 構造格子用モジュールの開発

構造格子データ用として，ボリュームレンダリングモジュールを開発した（図-5）．伝達関数の変更用ユーザインタフェースの実装が今後の課題である．

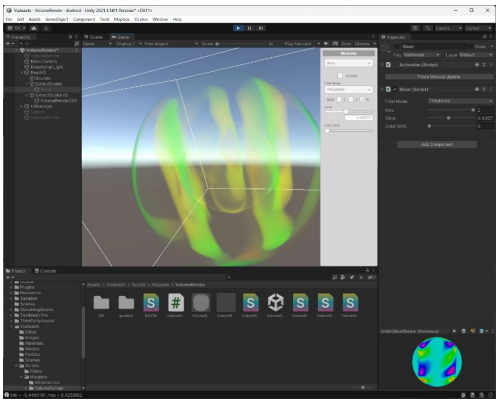

図-5 ボリュームレンダリングモジュール

5. まとめ

本稿では，Unity用可視化フレームワークVisAssetsの概要を紹介するとともに，構造格子以外のデータ形式への

対応の試みなど，その開発状況について述べた．VisAssetsはgithubにて公開しており[7]，誰でも自由に利用することができる．今回紹介したモジュール群についても今後のバージョンアップでの公開を予定している．新たなモジュールの作成を容易にする仕組みも提供しており，利用者自身による開発へのフィードバックにも期待したい．

謝辞

本稿に関する研究の一部はJSPS科研費21K11916およびJP22K12054の助成を受けたものです．

参考文献

[1] 宮地英生，川原慎太郎：ゲームエンジンを用いたVR 可視化フレームワークの開発，日本シミュレーション学会論文誌, Vol.12, No.2, pp.59-67, 2020.

[2] Cruz-Neira, C. et al: Surround-Screen Projection-based Virtual Reality: The Design and Implementation of the CAVE, Proc. of SIGGRAPH'93, pp.135-142, 1993.

[3] VTK: https://vtk.org/（最終アクセス日：2023年4月6日）

[4] Activiz for Unity: https://www.kitware.eu/activiz/（最終アクセス日：2023年4月6日）

[5] Mapbox: https://www.mapbox.com/unity（最終アクセス日：2023年4月6日）

[6] 伊藤貴之，山口裕美，小山田耕二： 長方形の入れ子構造による階層型データ可視化手法の計算時間および画面占有面積の改善，可視化情報学会論文集, Vol.26, No.6, pp.51-61, 2006.

[7] VisAssets: https://github.com/kawaharas/VisAssets（最終アクセス日：2023年4月6日）