デカルト座標流体シミュレーターにおける局所高解像度化の試み

首都大学東京 都市基盤環境コース　新谷哲也

ツール（道具）は「目的を実現すること」の重要性はもちろんのこと，その「使い易さ・分かり易さ」が利用される分野，もしくはその分野を超えた領域の発展に寄与することがある．つまり，ツールを操作することに時間と気を大きく取られることがなくなれば，さらなる応用を考える余裕ができるということである．模型ヘリコプターが古くから存在したにもかかわらず近年ドローンが大きく取り上げられ，様々な場面で応用されていることはその良い例である．それ故，我々がツールとして利用している湖や貯水池，沿岸域における流動・水質解析シミュレーターもその「使い易さ・分かり易さ」を追求する意義はある．このことは，シミュレーターを GUI 等でラッピングしてユーザーフレンドリーにするというよりもむしろ，その構造が我々の理解と一致していること，直感に基づいて操作可能なこと，組み替えが可能なように適切な粒度の部品でシミュレーターが構成・分類されていることが重要と考えている．

これまで，著者らは上記目的のために，シミュレーターを組み込み型（例えば int, double 等）から直接積み上げるのではなく，数値流体の分野で理解可能な型（オブジェクト：例えば水域，移流，計算セル等）を定義し，ある意味専用の言語体系（階層構造）を作ってその上でシミュレーターを組み上げる試みを行っている (新谷・中山 2009)．また，ユーザーが自由にオブジェクトを生成・組み換え・修正を行って実行内容を変化させることができるようにスクリプト言語上で利用可能とした (新谷・中山 2015)．これらにより，Couette 流れのような単純な計算から東京湾の流動計算まで無駄なくスケーラブルな対応，既存の GUI やグラフィックライブラリー等とのシームレスな連携，シミュレーター自身を繰り返し計算の内部に入れる計算等が可能になった．

「使い易さ・分かり易さ」という観点から空間の離散化を考えると，デカルト座標格子は直感的で他を圧倒するメリットがある．さらに，過去の膨大な数値スキームの研究資産，地形データを始め多くのデータがデカルト座標で与えられることも魅力的な点である．しかしながら，曲線的な地形境界を粗い計算格子で表現することや部分的に解像度を上げることが苦手であり，この欠点を克服した一般座標格子や非構造格子等が開発されて実務で利用されはじめている．一方，これらの空間離散化手法の問題点として，複雑な形状の水域に対して適切な計算格子を作成することに多くの労力が必要なことが挙げられる．境界に適合した格子であっても大きく歪んだ格子は精度低下・不安定性の原因となる．それ故，デカルト座標格子（より一般的に，境界に無理に沿わさず楽に構築できる構造格子）を合理的に利用して，上記欠点を克服する試みは多くの研究者によってなされてきた．nesting grids や multiple grids，そして tree-based grid（四分木）はその代表的な例である．しかしながら，これら提案されてきた多くの手法はそのアルゴリズムや利用法が複雑になる傾向にあり，著者の主観もあるが，ツールとしての「使い易さ・分かり易さ」が損なわれているように思われる．

そこで本研究では，デカルト座標格子の局所高解像度化 (refinement) を可能な限り「複雑さ」を増大させずに実現することを目的として，非構造格子のデータ構造と四分木の離散化パターンに基づいた新しいアプローチを提案する．まず，単純化するためにいくつかの妥協を行う．(1)refinement は水平方向のみとする（鉛直方向は partial-step を導入した非一様 z 座標）．(2)refinement 前の基本となる水平グリッドは各方向にそれぞれ一様間隔格子とする．(3) 静的な refinement とする．これらの近似の下，**図 1** に示すような refinement 前の基本グリッドを定義する．この基本グリッドの採用によって (i, j) インデックスによる容易な場所の特定（境界条件の指定，データの抽出等）の恩恵が得られる．ここまでは，従来のデカルト座標モデルと全く同様に見えるが，ここで示されている計算セル（のようなもの）は一段階抽象化されていて，計算セル（３次元の場合は鉛直方向にセルが積み重なったコラム）のコンテナ（入れ物）となっている．つまり，このコンテナ一つ一つが水平方向に任意分割可能な３次元デカルト座標系グリッドとなっていると考えて良い（コンテナ内部にさらに

(i, j) インデックスが存在する）．すべてのコンテナに一つずつ計算セル（コラム）が入っていれば従来のデカルト座標モデルと同等となる．また，この図の決して冠水しない陸部分（灰色部分）と水底以下の計算セルはマスクするなどして計算から除外するのではなく，水平・鉛直方向ともにメモリー上からもそぎ落とす．このような対応を行うと $i+1$ や $j-1$ などのインデックスを利用したフラックスの計算が不可能になるが，これらは非構造格子的な対応関係（簡単に言うと，お隣さん pointer を保持するオブジェクト群）を構築することによってフラックスの計算を実現している（新谷・中山 2015）．このことにより，コンテナ内の計算セルの接続，コンテナを跨いでの接続を含め自由な連結関係を実現できる．さらに良いことは，非構造格子では困難であった移流計算における高次補間のための風上点の特定が接続関係により容易に特定できるため，3 次精度の ULTIMATE-QUICKEST 等をそのまま利用できる．また，非構造格子と異なり，連続的に計算セルを番号付けできるため，生成される行列が優対角化される可能性が高い等のメリットも期待できる．

次に簡単に利用方法を説明する．先ほどの基本グリッド情報の他に，**図 2** に示すように各コンテナの 2 方向の分割数情報マトリックスを定義する（テキストファイルで与える）．これらの情報を元に最終的な空間離散化は**図 3** に示すような形になる．従来の四分木モデルでは隣り合う計算セルの分割数の比が 2 倍に固定されていることが多いが，本手法では隣接するセル間の境界面に対して接線方向の分割数が整数倍の関係になっていればよく，法線方向の分割数は基本的に何倍でも構わない．これらの比を大きくすると当然計算精度は低下するが，このことで保存性が失われることはない．本手法のメリットは，全体，もしくは任意の場所の解像度を**図 2** に示す内容のファイルをテキストエディターで修正しながら計算できることろにある．解像度を変化させるためにグリッドを再作成する必要なく，インクリメンタルに修正しながら高精度化を行える．地形データの解像度が refinement の解像度の整数倍であれば，より解像度の高い地形データを採用しながら空間離散化ができる．

本シミュレーターは SWIG を利用してスクリプト型プログラミング言語 Lua のモジュールとして作成してあり，モジュール自身は C++（および CMake, Boost ライブラリー）で作成し，内部は OpenMP によって並列化されている．水面変動や非静水圧の陰的な行列計算には，OpenCL(GPGPU) と OpenMP が利用可能な ViennaCL(http://viennacl.sourceforge.net/) を利用した．また，四分木構造のデータは可視化が面倒であると言われているが，非構造格子とみなし VTK unstructured データ形式 (vtu, http://www.vtk.org/) で出力することで容易に可視化できる．設計方法や数値アルゴリズム等の詳細に関しては，別の機会に紹介したい．

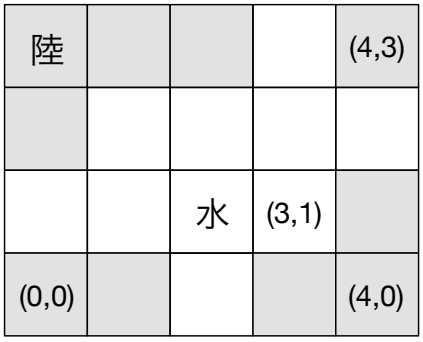

図 1 基本グリッドの概念図

1	1	1	3	1
1	2	1	1	1
1	2	1	5	1
1	1	3	1	1

1	1	1	1	1
1	2	1	1	3
3	3	1	5	1
1	1	1	1	1

図 2 x 方向分割 (左) と y 方向分割（右）情報マトリックス

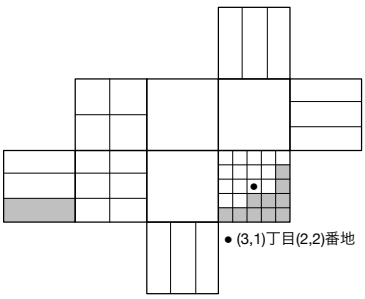

図 3 最終的な計算グリッド

参考文献：新谷・中山 (2009) オブジェクト指向型環境流体モデルの開発と検証，水工学論文集 53,
新谷・中山 (2015) 生物の細胞組織構造を模した流体シミュレーターの開発と検証，水工学論文集 59