F-044

GPGPU を用いたマルチエージェントシミュレーションの視界判定の高効率化
An Efficient LoS Detection Method Using GPGPU on Multi Agent Simulation

関口　大貴†　　古市　昌一†
Taiki Sekiguchi　Masakazu Furuichi

1. はじめに

従来より，視界判定（Line of Sight：LoS）はマルチエージェントシミュレーションにおいて重要な技術であり，特に防衛，訓練などの実世界を模擬するシミュレーションにおいて活用されている．

現在，ほとんどのシミュレーションシステムにおける LoS 判定は，点と点との間での“見えるか見えないか“という判定であるが，複雑な環境下において，大きさや形状を持つユニットとしてモデル化されたエージェントに対して判定を行う場合は“どの程度見えているか”という情報が必要になる．この情報を目標が多数存在する場面で処理する場合，その計算量は膨大なものとなる．

今回は GPU を用い，要求される計算量に対応し，多数の目標に対して高精度な LoS 判定を同時に取得する手法について提案する．

2. 従来の LoS 判定手法

2.1 ステップ法

ステップ法は，視点と注視点からなる 2 点の見通し判定手法である．シミュレーションシステムの多くはこの手法を利用している．

まず，視点から注視点までをメッシュなどを用い分割し，それぞれのメッシュに標高値を格納する．そして，分割単位で判定し，最終的に視点から注視点を標高メッシュにより，見通し判定を行うものである．標高メッシュによる判定は図 1 に示すように簡易な方法であるが，高速である反面，複雑な地形を表せないなど，制約も存在する．

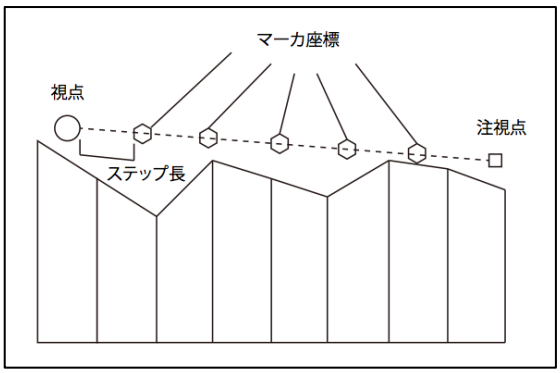

図 1　ステップ法のアルゴリズム

† 日本大学大学院生産工学研究科
Graduate School of Industrial Technology, Nihon University

2.2 GPU を用いた LoS

Salomon らの手法[1]は LoS 判定に GPU を用い，高速化させた手法である． GPU のレンダリング処理を用い，LoS を行うことにより，2 点間の見通し判定を高速化する．地形情報を二次元配列として読み込み，視点から注視点間に線を引く．描画する際，描画しようとしているピクセルが地形 に埋まる場合のみ描画する．描画した結果，ピクセルが描画されていれば False,されていなければ視点間に障害物がないため，True となる．図 2 に Salomon らの GPU を用いた LoS 判定アルゴリズムを示す．

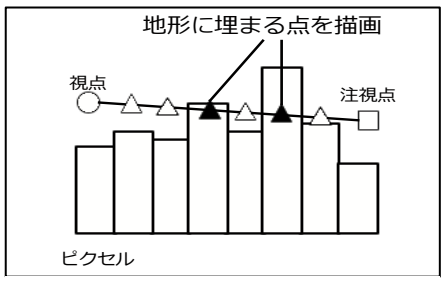

図 2　Salomon らによる GPULoS のアルゴリズム

2.3 面積 LoS

倉本による手法[2]は，GPU によるレンダリングを利用した 1 対面での視界判定手法である．(以下，面積 LoS と呼称する)この手法では，ターゲットのモデルが“どの程度見えているか”という情報が加味されており，主に市街地などの障害物がある状況での視界判定に適している．

まず，ターゲットのみを描画し，ピクセル数を計算する．そして Z バッファをクリアし，地形のみを描画する．次に Z バッファを持ったままターゲットを描画し，ピクセル数を計算する．これにより，ターゲット自身のピクセル数と，障害物に隠れていないピクセル数から，どの程度見えているのかという情報が数値で認識することが可能になる．面積 LoS はマルチエージェントシミュレーションなどの上で模擬される 1 対多数の判定では計算量が多くなる．図 3 に面積 LoS のアルゴリズムを示す．

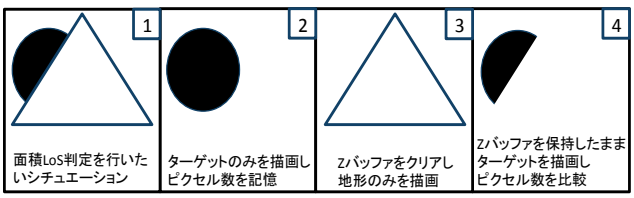

図 3　面積 LoS のアルゴリズム

Copyright © 2016 by
The Institute of Electronics, Information and Communication Engineers and
Information Processing Society of Japan All rights reserved.

3. 提案手法

本提案方式は 1 対 1 の判定を 1 体複数のシチュエーションに最適化し，計算資源として GPU を利用したものである．マルチエージェントシミュレーションでは複数対複数を模擬する必要がある．本来 1 対 1 で行われる LoS 判定を複数のターゲットに対して行う場合，膨大な処理が行われることになる．これは現在の豊富な計算資源をもってしても制約があり，大きな負荷となる．これに対し，本提案方式は 4 章で述べる一括視界判定アルゴリズムを利用することにより 1 対複数の視界判定に最適化し，さらに，要求される計算量に余裕をもって対応するため，GPU を用いた計算を行う．

これらに先述した面積 LoS の“どの程度見えているか”という情報を加え，True,False ではなく実数値で取得する．

これにより，複数のターゲットを一括で“どの程度見えているか”という情報を得ることが可能になる．

4. 一括視界判定アルゴリズム

先述した面積 LoS は対象が複数の場合，ターゲットとの視界判定を 1 対 1 で行いその度に地形を描画し直す必要がある．本提案方式では，視界内にいるターゲットを一括で描画，判定することにより，地形の描画や座標演算などを複数回行うことによる負荷を回避することができる．

図 4 に面積 LoS を用いた複数のターゲットに対する判定方法を示す．図 4 右図のようにターゲット 1 体ごとに判定を行い，その都度地形を含め，描画しているため，計算量が多く負荷が高い．

図 5 に本提案方式を用いた複数のターゲットに対する判定方法を示す．図 5 左図のように視界を区切り，右図のように視界を面として描画するため，一回の視界判定で多数のターゲットへの視界判定を行うことが可能である．例として四方向に視界を区切ったが，さらに分割し，精度を高めることも可能である．

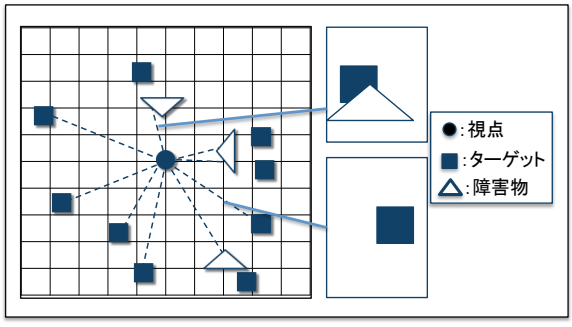

図 4　面積 LoS による判定

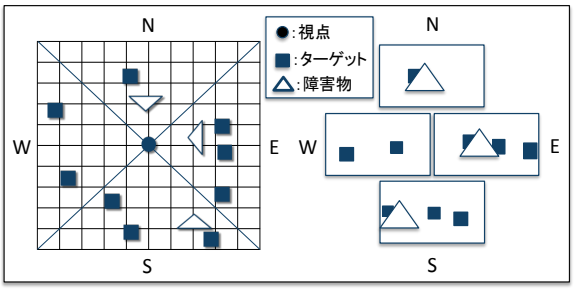

図 5　本提案方式による判定

5. 計算時間の見積もりと評価方法

地形の頂点数に対してターゲットの頂点数が充分に小さい時を想定した．面積 LoS と本提案方式の実行時間の概算は以下のように表現できる．

$$T_1 = NTs(V + P_1) \quad \cdots (1)$$
$$T_2 = 4(Ts + V + NP_2) \quad \cdots (2)$$

この時．
T_1 =面積 LoS の実行時間
T_2 =本提案方式の実行時間
N =ターゲット数
T_s =GPU へのコマンド転送時間
V =頂点シェーダの演算時間
P_1 =面積 LoS のターゲットあたりのピクセル演算時間
P_2 =本提案方式のターゲットあたりのピクセル演算時間
とする．

(2)式の定数 4 は，4 面に対し視界判定を行った場合を想定した．

(1)式を面積 LoS を用いた実行時間，(2)式を本提案方式を用いた実行時間として立式した．

(1)式と(2)式から，本提案方式を用いることによりターゲット数による GPU へのコマンド転送時間．頂点シェーダーの演算時間の増加が解消されることがわかる．このことより，ターゲット数が増加しても実行時間を高速化，安定化させられると考えられる．

面積 LoS は判定ごとに地形オブジェクトとモデルを再描画する必要があるため，毎回頂点演算を実施する必要がある．今回提案した方式では，視界範囲をまとめて描画するため，地形描画時間削減，頂点シェーダーの処理時間削減につながると考える．本提案方式では多数の視界判定を一括で行うため，視界判定を行いたいターゲットの増加とともに，既存方式に比べて有利になると考える．

6. 今後の課題

今回は単純な形状のモデルを用いることを想定し構築を行ったが，提案方式ではあらゆる形状のモデルに対して LoS 判定を行うことができる．さらに人型モデルを使用し，身体の各部位に重み付けをすることで認知のしやすさを変化させることも可能である．吹雪や，砂埃などの複雑な視界環境であっても，レンダリングされた情報を使用するため，有効性があると考える．

現時点では理論およびアルゴリズムが構築できたため，今後．実際の計算機環境において評価を行う予定である．さらに上でも述べたように高効率化，高機能化の余地があるため，研究を進めることが今後の課題である．

参考文献

[1] Brian Salomon et.al，“ACCELERATING LINE OF SIGHT COMPUTATION USING GRAPHICS PROCESSING UNITS”，Proc.of Army Science Conference，2004
[2] 倉本健介，“GPU を用いた見通し判定の高精度化方法について”，FIT2010 第 9 回情報科学技術フォーラム，No421．(2010)

Copyright © 2016 by
The Institute of Electronics, Information and Communication Engineers and
Information Processing Society of Japan All rights reserved.