生成都市道路モデルにおいて A スター最短経路探索アルゴリズムの評価

艾宇飛[†1]　川合康央[†1]

概要：頻発する自然災害に対応するために，A スター最短経路探索アルゴリズムを用いて，日本の現実都市を対象とした災害避難シミュレーションシステムを開発した．関連研究により，A スター最短経路探索アルゴリズムは迷路実験において高い効率を示している．しかし，より広範な都市道路においてもその効率を維持できるかどうかは未検証である．本論文では，Unity をツールとして使用し，ランダムに都市道路モデルを生成するスクリプトを作成した．それぞれに A スター最短経路探索アルゴリズムとダイクストラ法を実装して，時間とステップ数の観点から A スター最短経路探索アルゴリズムの評価を行った．

1. はじめに

日本は，特有の地理環境と気候のため，多くの自然災害が発生している．特に，台風や集中豪雨などの気象と地形の起伏が原因で発生する水害によって，低海抜地域では道路が封鎖されることもある．また，河川の氾濫等により交通が麻痺する場合もある．災害が拡大した際には，図 1 に示すように，封鎖された道路が住民の避難に大きな支障をきたすこととなる[1]．このような問題を解決するために，最短経路支援を含む，様々なタイプな避難支援システムが提案された[2, 3].

本研究では，自然災害と人の避難行動を模擬できる災害避難シミュレーションシステムを開発した. 本システムは，より大規模な都市を対象とし，動的な経路調整と複数エージェントの同時避難に対応することで，都市全体における災害発生時の避難をシミュレートすることを目指すこととした．システムの実装にあたっては，専用のツールを使用して，日本の地理情報をインポートし，A*最短経路探索アルゴリズムを実装するとともに，エージェント生成ルールを作成した．本システムの詳細は，本文の第二節で詳しく説明する.

次に，A*最短経路探索アルゴリズムが，より広範な都市道路を対象とした場合でも高い効率を維持できるかを，確認するために評価実験を行った．Unity を使用してランダムな都市道路モデルを生成するスクリプトを作成し，生成された都市道路モデルを複製して，A*最短経路探索アルゴリズムとダイクストラ法をそれぞれ実装した．評価実験の詳細は第三節で, アルゴリズムの評価結果は第四節で示す.

図 1　品川区洪水災害地図

2. 避難システム

災害避難シミュレーションシステムは Unity 2021.3.36f1 を用いて開発し，スクリプト言語には C# を使用した．シミュレーションの説得力と現実性を高めるため，実際の都市データを活用することとした.

地理情報の出典は以下の通りである：
・土地起伏・都市建物情報：U.S. Geological Survey
・道路情報：国土交通省

2.1 都市モデル

図 2 は，Cesium を用いて土地や建物データをインポートしたスクリーンショットである．Cesium は Analytical Graphics, Inc.社が開発したオープンソースのプロジェクトであり，高精度かつ高効率な仮想地球儀を提供する．Cesium を利用することで，高品質な地形や 3D モデルをインポートし，研究の実験対象として活用できる[4].

図 3 は，PLATEAU を用いて道路レイヤーをインポートしたスクリーンショットである．PLATEAU は国土交通省

†1 文教大学大学院情報学研究科

が主導する，日本全国の 3D 都市モデルの整備・オープンデータ化プロジェクトである．PLATEAU を使用することで，編集可能な Unity オブジェクトとして日本の地理データをインポートでき，他のアルゴリズムやシステムの実装を容易にする[5].

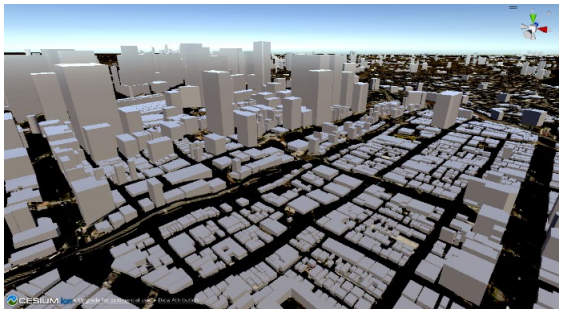

図 2　Cesium からの建物情報

図 3　PLATEAU からの道路レイヤー

2.2　エージェント設定

人間を模擬するため，赤い四角形の「エージェント」プレハブを設定した．エージェントは，道路レイヤー上で自由に移動できるものとした．エージェントには，「身長」「最大移動速度」「回転速度」などの属性が設定されており，これらを調整することで，よりリアルに人間の特徴を表現できる．また，数値を変更するとシステム内に即時に反映される．

避難時には，住民が道路沿いに移動すると仮定し，個体間の差異は考慮しない．しかし，実際の災害発生時には住民の位置が常に異なるため，エージェント生成スクリプトを定義した．このスクリプトによって，仮想都市空間内のすべてのエージェントが道路上に配置され，その後道路沿いに避難する過程をシミュレートできるようになっている．

2.3　実装したアルゴリズム

シミュレーションシステムでは，A* 最短経路探索アルゴリズム（A-Star Pathfinding algorithm）を採用した[6]．以下，代表的な他のアルゴリズムとの比較を行う．

深さ優先探索アルゴリズム（Depth-First Search algorithm）は，その原理がシンプルであり汎用性が高く，多くの場面で使用されるアルゴリズムの一つである[7]．既存研究では，深さ優先探索アルゴリズムと A*アルゴリズムの比較が行われているが[8]，結果 A*アルゴリズムが時間効率の面で優れていることが示唆されている．

ダイクストラ法（Dijkstra's algorithm）は，グラフ理論における単一始点最短経路問題を解くための，最良優先探索アルゴリズムである．このアルゴリズムは，辺の重みが非負数の場合に適用され，その効率性から広く使用されている[9]．既存研究においても，ダイクストラ法と A*アルゴリズムの性能比較が行われ，それぞれの特性や適用場面が報告されている[7]．特に小規模な迷路実験の結果，A*アルゴリズムはダイクストラ法と比較して，ステップ数が著しく少ないことが示されている[10].

そのため，本シミュレーションシステムでは A*最短経路探索アルゴリズムをコアとして実装し，エージェントの避難行動に適用している．

図 4　道路上にランダムに生成されたエージェント

2.4　システム作成

3D 都市モデル，エージェント，最短経路探索アルゴリズムを組み合わせることで，図 5 に示すような避難シミュレーションシステムを実装した．全てのエージェントは正常に動作し，避難行動を行うことが可能であった．さらに，本システムを基盤として，避難所のカバレッジ率や各避難所の利用率計算など，多くの機能を将来的に拡張することができる．

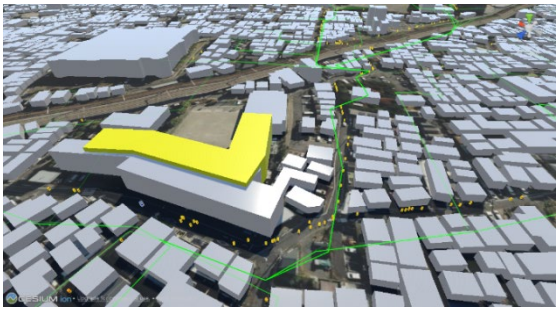

図 5　避難するエージェント

3.　評価実験システム

都市道路モデル生成システムは Unity 2021.3.36f1 を用いて開発し，スクリプト言語には C# を使用している．

3.1 道路生成

「ノード」というプレハブを作成した．道路を生成する際には，このプレハブを次々と生成して道路を構成する．各ノードには，「前ノード」「起点までの距離」「終点までの距離」の 3 つのパラメータが設定されている．また，作成したスクリプトによって，ノードが訪問された場合には青色に変わり，最短経路として認定された場合には黄色に変わるようにした．

道路を生成する前に，「MapSize」というパラメータに基づいて背景となる正方形の基盤を生成する．この基盤は道路の背景として使用される．

基盤が生成された後，設定されたパラメータ「HorizontalNum」と「VerticalNum」を上限としてノードを繰り返し生成し，横方向と縦方向の道路を構成する．観察をしやすくするため，生成時には各ノードが自身の座標を「Name」属性として設定する．また，各隣接する 2 つのノードには一定の確率で自らを消去する処理を行い，さらに隣接ノードが 1 つしかないノードもすべて消去する．この方法によって，不規則な小道を再現できるようにした．

ここまでで，道路の生成はおおむね完了となる．システムはまずランダムにノードを起点として選択し，次に起点以外のランダムなノードを終点として選択する．図 6 に生成された道路を示す．

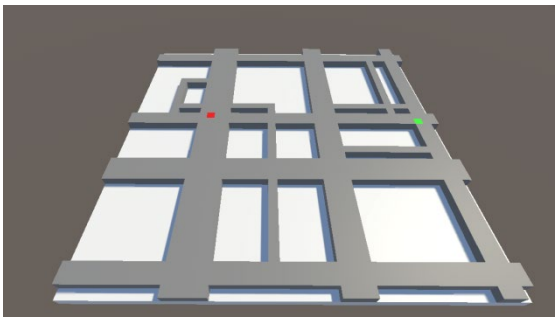

図 6　生成した都市道路

3.2 A スター最短経路探索アルゴリズムの実装

A*最短経路探索アルゴリズムは柔軟性が高く，Cost（コスト）に基づいて次のノードを選択する．本実験では，A*最短経路探索アルゴリズムの Cost の算出にユークリッド距離を用いる．この距離計算を適切に最適化することで，アルゴリズムの効率をより高めることができる．

スクリプト内では，Unity の Vector3.Distance() 関数を使用してユークリッド距離を簡単に計算し，その結果をノードの「終点までの距離」パラメータに記録している．

起点位置から，スクリプトは周囲の 4 つの隣接ノードを探索する．探索したノードを「未探索ノード」集合に追加し，その集合の中から「終点までの距離」値が最小のノードを次のステップとして選択する．このプロセスを繰り返し，終点ノードに到着するまで探索を続ける．

探索が完了した後，終点ノードから「前ノード」属性をさかのぼることで最短経路を取得し，その経路をマーキングして可視化する．

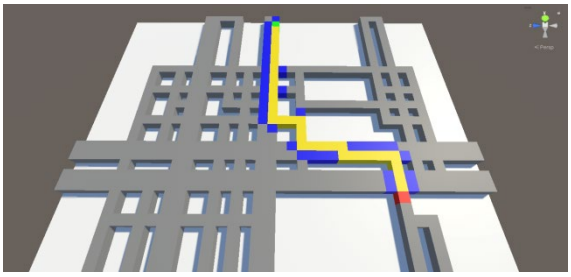

図 7　実装した A スター最短経路探索アルゴリズム

3.3 ダイクストラ法の実装

ダイクストラ法は，最短経路を確実に求められるアルゴリズムであり，多くの分野で用いられている．本システムでは，各ノードが生成されるとき，起点から当該ノードまでの距離を計算し，「起点までの距離」パラメータに記録している．

起点位置から，スクリプトは周囲の 4 つの隣接ノードを探索し，探索したノードを「未探索ノード」集合に追加する．そのうえで，集合内のノードの中から「起点までの距離」値が最小のノードを次のステップとして選択する．もし同じ値のノードが複数存在する場合は，インデックスが小さいノードが優先的に選択される．この方法によって，ダイクストラ法は道路を十分に探索できる．このプロセスを繰り返し，終点ノードに到達するまで探索を続ける．

探索が完了した後は，終点ノードから「前ノード」属性をさかのぼることで最短経路を取得し，経路をマーキングして可視化する．

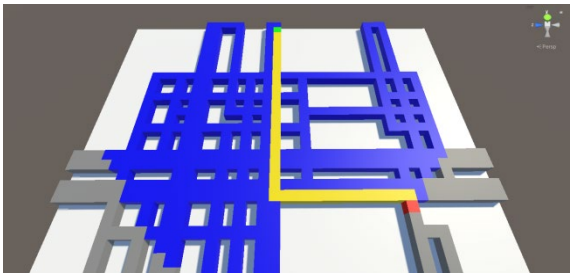

図 8　実装したダイクストラ法

3.4 実験からの出力

ノード探索のプロセスでは，一定の停止時間を挿入することで，より直観的に結果を確認･比較できるようにした．

今回の実験では，以下の 5 つのパラメータを出力する．時間は秒単位，ステップ数は「探索したノード数」単位である．

（AStar：A スター最短経路探索アルゴリズム）
（Dijkstra：ダイクストラ法）

MazeNum → 実験番号
Star：Step → A-Star 完成までステップ数
Star：Time → A-Star 完成まで時間
Dijkstra：Step → Dijkstra 完成までステップ数 Dijkstra：Time → Dijkstra 完成まで時間

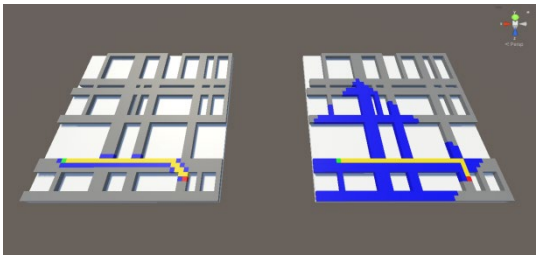

図 9 対比実験進行中の様子

4. A スター最短経路探索アルゴリズム評価

合計で 50 回の比較実験を実施し，得られたデータに基づいて「TimeEfficiency（時間効率）」と「StepEfficiency（ステップ効率）」の 2 種類の指標を計算した．

時間効率の算出では，ダイクストラ法の処理時間を A*最短経路探索アルゴリズムの処理時間で割ることで，A*最短経路探索アルゴリズムがどの程度時間効率を向上できるかを示している．

一方，ステップ効率の算出では，最短経路を保証できるダイクストラ法のステップ数を分母とし，A*最短経路探索アルゴリズムのステップ数を分子とすることで，A*最短経路探索アルゴリズムがどの程度ステップ数の面で効率を下げるかを示している．

得られたデータは箱ひげ図として可視化し，Y 軸の差が大きいため，時間効率とステップ効率を分けて展示した．図 10 に示すように，A*最短経路探索アルゴリズムの時間効率の向上は顕著である．50 回の実験のうち，最大で 2400% を超える時間効率が確認され，最低でも 200% 以上の向上が見られた．つまり，最も低いケースでもダイクストラ法に対して 2 倍以上の速度向上を示していることになる．Y 軸の平均値は約 1100% であり，実験によく利用されるダイクストラ法と比較しても，A* 最短経路探索アルゴリズムが広範な都市道路モデル上で顕著な効率向上を示したことがわかる．

ステップ効率については，図 11 に示すように，50 回の実験の中で 8 回の異常値が見られ，残りの 42 回では A*最短経路探索アルゴリズムがダイクストラ法と同様に最短経路を見つけ出したことを示している．8 回の異常値のうち 6 回分の平均値は約 90% 付近であり，A*最短経路探索アルゴリズムが最短経路を見つけられない場合でもステップ数の偏差は 10% を超えないことを示した．最低値は 65% であり，最悪の場合は全体の 35% 程度の追加移動が発生することがわかった．

データ処理後に，異常値が発生した実験番号を確認したところ，移動距離が最適ではない場合でも時間効率の向上は，約 1000% 程度であることがわかった．これにより，A*最短経路探索アルゴリズムの優れた時間効率は，ある程度ステップ効率における欠点を補う可能性があることが示唆される．

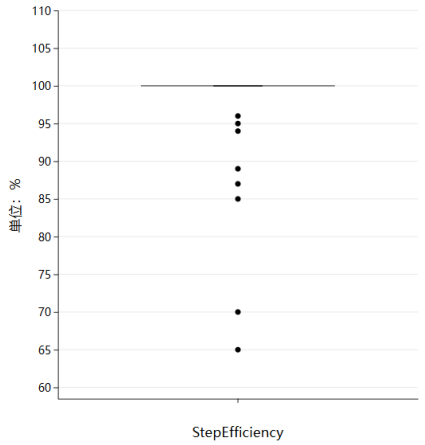

図 10 時間効率

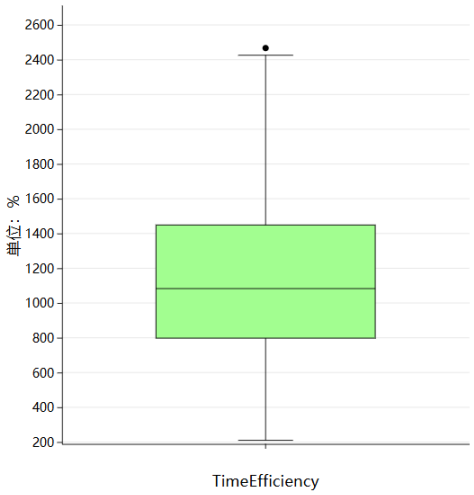

図 11 ステップ効率

5. まとめ

本研究では，A*最短経路探索アルゴリズムが，より広範な都市道路モデル上でどの程度効率的に動作するかを評価するために，ランダム生成スクリプトを作成した．このスクリプトを用いて A*最短経路探索アルゴリズムと，研究で広く用いられているダイクストラ法を実装し，比較検証を行った．

実験の結果，広範な都市モデルを対象とした場合，A*最短経路探索アルゴリズムは時間効率の面で平均 10 倍以上の改善を示し，極めて顕著な優位性を持つことが分かった．

一方，ステップ数の評価においては，A*最短経路探索アルゴリズムが基本的に最短経路を見つけられるものの，まれに最短経路ではない場合が発生し，その際の経路距離の誤差は平均 10% 以内にとどまった．この問題は，本研究で用いたコスト計算手法が単純なユークリッド距離ベースであったためであり，コスト計算の最適化によって解決可能である．

総合的に見て，本研究の実験結果から，A* 最短経路探索アルゴリズムは広範な都市道路モデル上で高い効率を示し，優位性を保持していると考えられる．

謝辞 本研究は JSPS 科研費 JP 23K11728 及び文教大学情報学部共同研究費，文教大学大学院情報学研究科共同研究費の助成を受けたものです．

参考文献

[1] 及川康, 児玉真, 片田敏孝. 水害進展過程における住民対応行動の形成に関する研究. 土木学会論文集, 2005, vol.786, p.89-101.
[2] 渡邊博之, 成田祐一, 大山勝徳, 加瀬澤正, 武内惇, 竹中豊文. モバイル端末を活用した災害時最短避難経路提示システムの開発. 情報処理学会論文誌, 2012, vol.53, no.7, p.1768-1773 .
[3] 柿本竜治, 山田文彦, 田尻亮司, 原田翔太. リスクコミュニケーションを通じた実践的水害避難訓練に基づく避難行動シミュレータの構築. 土木計画学研究・論文集, 2009, vol.26, no.1, p.113-122.
[4] 川合康央, 池田岳史, 益岡了. 大規模 3 次元都市空間モデルを用いた都市情報ビジュアライゼーションシステムの提案. 日本デザイン学会研究発表大会概要集, 日本デザイン学会第 70 回研究発表大会, 2023, p.448-449.
[5] 稲垣誠, 川合康央. PLATEAU を使用した横浜市津波シミュレーションシステムの開発. 第 84 回全国大会講演論文集, 2022, p.911-912.
[6] Yan, Y.. Research on the A Star Algorithm for Finding Shortest Path. Highlights in Science, Engineering and Technology, 2023, vol.46, p.154-161.
[7] 長井歩, 今井浩. df-pn アルゴリズムの詰将棋を解くプログラムへの応用. 情報処理学会論文誌, 2002, vol.43, no.6, p.1769-1777.
[8] Liu, X., Gong, D.. A comparative study of A-star algorithms for search and rescue in perfect maze. 2011 international conference on electric information and control engineering, 2011, p.24-27.
[9] Sari, I. P., Fahroza, M. F., Mufit, M. I., Qathrunad, I. F. Implementation of Dijkstra's Algorithm to Determine the Shortest Route in a City. Journal of Computer Science, Information Technology and Telecommunication Engineering, 2021, vol.2, no.1, p.134-138.
[10] Zikky, M.. Review of A*(A star) navigation mesh pathfinding as the alternative of artificial intelligent for ghosts agent on the Pacman game. EMITTER International journal of engineering technology, 2016, vol.4, no.1, p.141-149.