

経路探索における部分グラフを用いた 経路グラフ最適化に関する研究

小坂 大樹¹ 阿部 雅樹^{†1} 渡辺 大地^{†1}

概要：経路探索は、ある地点からある地点への移動経路を算出するものである。カーナビやロボット AI、ゲーム AI などの分野で用いる。経路探索を行うには、マップ状態を表すグラフデータが必要である。しかし、マップの大きさなどによってグラフデータの量が増大すると、経路探索に時間が掛かってしまう。ゲーム分野においてはリアルタイムに経路探索を行うため、限られた時間で探索を行う必要がある。そのため、探索時間短縮のために、経路探索アルゴリズムであるダイクストラ法や A* アルゴリズムなどの改良が行われてきた。また、経路探索を行う際のグラフのデータを階層化により削減することで探索時間の短縮を図る、上位層グラフなどがある。しかし、マップによっては階層化に適していないグラフがある。そこで、本研究では上位層グラフの適応が難しいマップにおいて、グラフデータの削減を行うことを目的とする。グラフのエッジに対して、削除を行うための優先度を設定し、その優先度にしたがってエッジを削除することで、グラフデータの削減を図る。

1. はじめに

経路探索とは、ある地点からある地点への移動経路を算出するものである。カーナビゲーションシステム [1] や、ロボット AI [2]、ゲーム AI [3]、交通シミュレーションや、災害時の経路シミュレーションなど多くの分野で用いる。図 1 は、ナビゲーションアプリで八王子駅から東京工科大学へのルート検索を行った様子を示したものである。

経路探索を行う際のデータとして、グラフを用いる。グラフにおけるノードを各地点、エッジを各地点同士を結ぶ道として扱い、それらのデータを用いて経路探索を行う。

経路探索の基本となるアルゴリズムの 1 つに、Dijkstra [4] が提唱したダイクストラ法がある。このアルゴリズムは、エッジが非負のコストを持つグラフにおいてノード間の最適な経路を算出するものである。カーナビゲーションシステムや、電車の乗換案内 [5] にも利用されており、運賃や所要時間をコストとすることで、適切な経路を探索する。

しかし、経路探索を行うマップの拡大やコストなどの情報の追加に伴いグラフデータが大きくなると、ダイクストラ法をそのまま適用しただけでは、探索に多くの時間を要してしまう。カーナビゲーションシステムや電車の乗換案内においても、グラフデータが膨大であるため、実際はダ



図 1 八王子駅から東京工科大学へのルート検索の様子

イクストラ法などを改良したアルゴリズムを適用していることが多い。ゲーム分野においては、キャラクターの移動経路を求める為にリアルタイムに経路探索を行う。そのため、限られた時間内に経路探索の処理を完了する必要がある。さらに、オープンワールドといった広大なマップ上をキャラクターが自由に移動できるゲームが増えているため、探索時間の短縮が特に重要となっている。

¹ 東京工科大学大学院バイオ・情報メディア研究科
IPSJ, Chiyoda, Tokyo 101-0062, Japan

^{†1} 現在、東京工科大学メディア学部
Presently with Tokyo

探索時間を短縮するために多くの研究が行われている。2点間最短経路問題を解くためにPeterら[6]は、A*アルゴリズムを提案した。このアルゴリズムは、ダイクストラ法の探索時における冗長な部分を省略するためのアルゴリズムである。ゲーム分野においてキャラクターの移動を行うための経路探索などで用いることが多い。基本的な流れはダイクストラ法と同じである。A*アルゴリズムでは、そこからヒューリスティック関数というものを設定する。ヒューリスティック関数とは、あらかじめ判明している情報を元に追加で設定するコストのことである。多くの場合、目的地となるノードからのユークリッド距離を各ノードに設定し、冗長な探索を省略している。これにより、探索量・時間の短縮を図っている。

また、Stenz[7]が提唱したD*アルゴリズムは、ノードが変化した際に、関連のあるノードのみを更新するアルゴリズムである。もともと存在した道が塞がるなど、マップが変化した際にダイクストラ法やA*アルゴリズムは一から探索をし直す必要があるが、D*アルゴリズムでは、一部のノードのみ更新すれば良いため、マップ変化に応じた再探索の時間を短縮することができる。

一方で、経路探索のアルゴリズムを改良するのではなく、経路探索を行うためのグラフデータを削減することで、探索量・時間の短縮を図る手法がある。グラフデータを削減するための手法の一つとして、上位層グラフがある。上位層グラフとは、階層構造を持つグラフのことである。上位層グラフでは、グラフデータにおいて位置的に近いノード群をひとつのエリアとして扱う。その後、各エリアを一つのノードとして扱うことでグラフに対して階層化を行う。図2は、上位層グラフの一例を示したものである。このよ

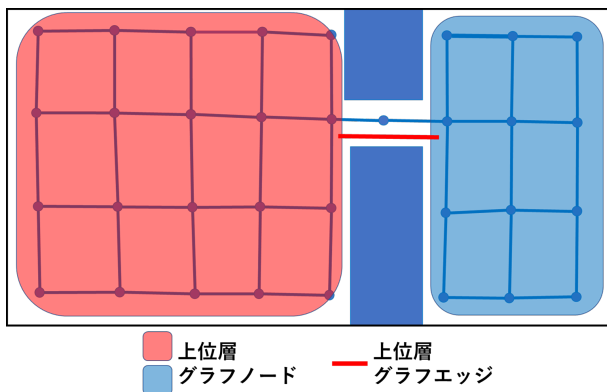


図2 上位層グラフの一例

うな上位層グラフを用いることで、探索時におけるグラフデータの削減を行うことができ、計算時間や計算量の削減に繋がる。しかし、上位層グラフでは対応できない場合がある。上位層グラフは、グラフのノードに対してエリア分けを行う必要がある。その為、グランド・セフト・オート[8]における街中や、Ark:Survival Evolved[9]における

森の中といった、建物や木などの障害物が点在しているようなマップでは、上位層グラフを作成するためのエリア分けが難しい場合がある。

そこで、本研究では上位層グラフの適応が難しいマップにおいて、グラフデータの削減を行うことを目的とする。グラフのエッジに対して、削除するための優先度を設定し、その優先度に応じたエッジ削除を行うことで、グラフデータの削減を図る。

もとのグラフで求まる最短経路と、あるエッジを削除した際に求まる最短経路にどれだけ差が出るかを、各エッジに対して求めた。また、すべてのノードの組み合わせに対して最短経路を求め、各エッジを通過する回数を求めた。これらを行った結果、各エッジを削除した際のエッジ削除前との変化を比較したものを優先度設定の理論最適解とした。しかしこの方法だと、本研究で想定する大規模なマップにおいては、大幅な処理時間が掛かってしまう。そこで、理論最適解に近い結果をより短い処理時間で算出するために、エッジの内積と距離を用いた優先度の設定方法について提案する。

簡易な経路グラフに対し検証を実施したところ、調整係数を適切に調整することにより、理論最適解に近い結果を本手法により算出することができた。しかしながら、経路グラフの種類によってはあまり良好とは言えない場合もあり、任意の経路グラフで良好な結果を得るには手法の改善が必要と考える。

2. 提案手法

2.1 表記定義

本研究におけるグラフ各部の数式表現は、基本的には文献[10]に準拠するものとし、以下のように定める。

- 本論文では、経路探索に用いるグラフのことを経路グラフと呼称する。
- 経路グラフにおける頂点のことをノード、辺のことをエッジと呼称する。図3で、ノード及びエッジを示す。
- エッジ e_n がノード P_i とノード P_j を結んでいるとき、 P_i と P_j は隣接するという。
- エッジ e_n は P_i, P_j 間の距離 $d(P_i, P_j)$ をコストとして持つ。
- 経路 W は、順に伝っているノードの列と定義し、経由するノードを要素として、

$$W = \{P_0, P_1, \dots, P_n\} \quad (1)$$

と表記する。

- 図4では、図3において式(2)となる経路を示したものである。

$$W = \{P_0, P_3, P_5\} \quad (2)$$

- 式(2)の経路 W の長さ $|W|$ は以下の通りである。

$$|W| = d(P_0, P_3) + d(P_3, P_5) \quad (3)$$

- P_i を始点, P_j を終点とする経路の内、長さが最短となる経路を最短経路とし、その長さを

$$D(P_i, P_j) \quad (4)$$

と表記する。

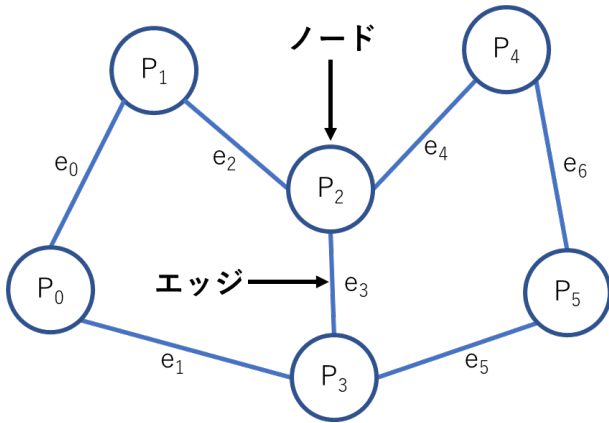


図 3 ノードとエッジ

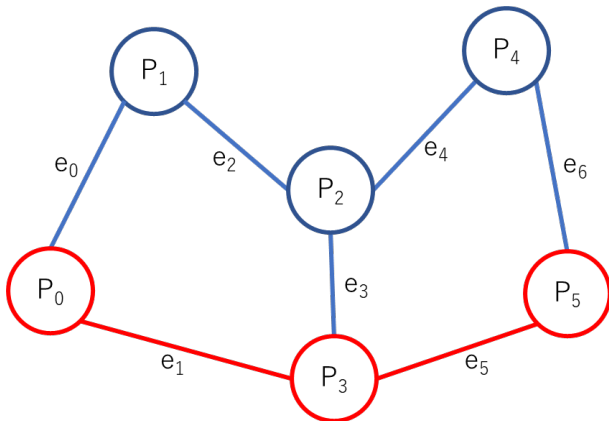


図 4 経路の一例

2.2 提案手法

本研究では、削除優先度を設定するために3つの方法を行った。

2.2.1 項と 2.2.2 項では、すべてのノードの組み合わせに対して最短経路を求めることで優先度の設定を行った。それらの結果を踏まえて 2.2.3 項では、エッジの内積と距離を用いた、より短い処理時間で優先度を設定するため手法を提案する。

2.2.1 最短経路の長さによる優先度設定

本項では、削除優先度を行うために以下の手順を行う。

- (1) 経路グラフのエッジをひとつ削除する
- (2) 2つのノード P_i, P_j の最短経路の長さ $D(P_i, P_j)$ を求める
- (3) 手順 2. をすべてのノードの組み合わせに対して行い、その合計値を算出する
- (4) 手順 1-3 をすべてのエッジに対して行う
- (5) 手順 3. で求めた値の正規化を行う

正規化を行った値が大きいほど削除優先度を低く、値が小さいほど削除優先度を高く設定する。

2.2.2 各エッジの通過回数による優先度設定

本項では、削除優先度を行うために以下の手順を行う。

- (1) 2つのノード P_i, P_j の最短経路を求める
- (2) 求めた最短経路がどのエッジを通るか求める
- (3) 手順 1-2 をすべてのノードの組み合わせに対して行い、各エッジを通った回数を求める
- (4) 手順 3. で求めた値の正規化を行う

正規化を行った値が大きいほど削除優先度を低く、値が小さいほど削除優先度を高く設定する。

2.2.3 エッジの内積と距離を用いた優先度設定

2.2.1 項と 2.2.2 項における優先度設定では、ノードとエッジの数の増加に伴う、処理時間の増加量が多い。本研究では大規模なマップを想定しているため、ノードとエッジ数の増加に伴う、処理時間の増加を減らしたい。そこで本項では、エッジの内積と距離を用いた優先度設定を行う。まず経路グラフの中心地点となる O を任意に設定する。中心地点となる O を設定した様子を図 5 に示す。

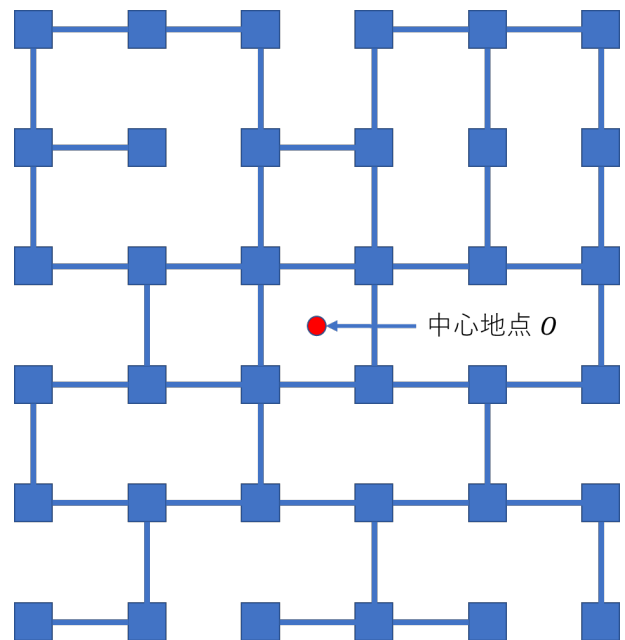


図 5 中心地点 O を設定した様子

また、エッジの midpoint M_i と単位方向ベクトル V_i を求め

る。次に、エッジ e_i の中点 M_i から任意の距離 T_1 を閾値とし、その距離内にあるエッジ e_i 以外のエッジ e_j に対して、以下の計算を行う。

$$I_i = \begin{cases} \frac{|V_i \cdot V_j|}{d(M_i, M_j)} & d(M_i, M_j) \geq 1 \wedge |V_i \cdot V_j| \geq T_2 \text{ のとき} \\ |V_i \cdot V_j| & d(M_i, M_j) < 1 \wedge |V_i \cdot V_j| \geq T_2 \text{ のとき} \\ 0 & |V_i \cdot V_j| < T_2 \text{ のとき} \end{cases} \quad (5)$$

ここで T_2 は任意に設定する閾値とする。任意の距離内にあるエッジが複数ある場合は、それぞれのエッジに対して式 (5) から I_i の値を求めその合計値を I_i の値とする。閾値である距離 T_1 により、周囲のエッジを求めた様子を図 6 で示す。

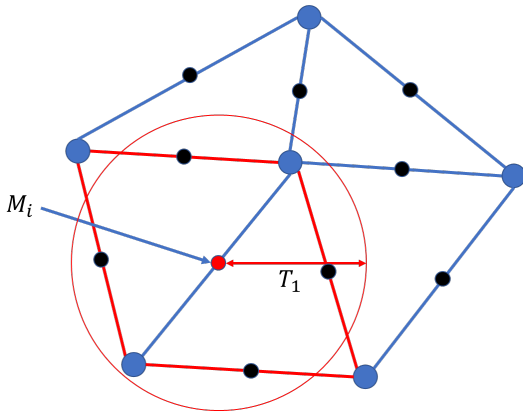


図 6 周囲のエッジを求めた様子

すべてのエッジに対して I_i の値を算出したら、各エッジの優先度 P_i を求める。式 (6) を示す。

$$P_i = \alpha I_i + \beta d(O, M_i) \quad (6)$$

ここで α, β は調整係数とする。 α, β は、マップ状況に応じて任意の数値に設定し調整を行う。また、閾値である T_1, T_2 の値も α, β と同様に、マップ状況に応じて任意の数値に設定し調整を行う。式 (6) で求めた P_i の値が大きいほど削除優先度を低く、小さいほど削除優先度を高く設定する。

3. 検証・考察

3.1 検証

本章では、2.2.1 項、2.2.2 項、2.2.3 項による手法をそれぞれ手法 1、手法 2、手法 3 と呼称する。各手法を用いて、経路グラフのエッジに対して優先度の設定を行った。設定した優先度となる数値が高いほど削除優先度が低く、数値が低いほど削除優先度が高い。そこで、設定した優先度に対して正規化を行い、エッジに対して疑似カラーを設定した。エッジが赤いほど削除優先度が低く、エッジが青いほ

ど削除優先度が高くなる。エッジに対して疑似カラーを設定した様子を図 7 に示す。

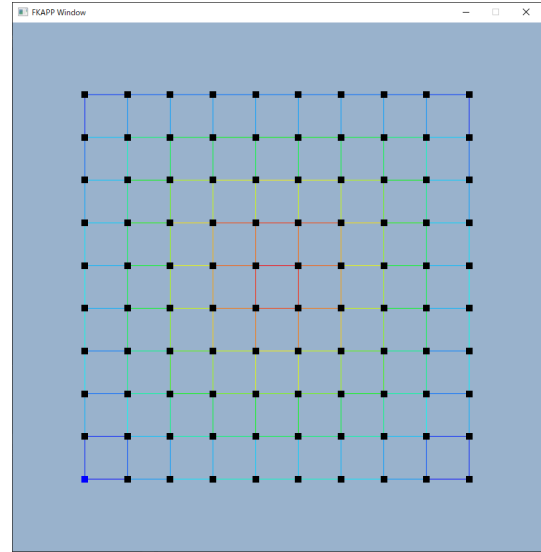


図 7 エッジに対して疑似カラーを設定した様子

なお、手法 3 では係数や閾値の調整により、手法 1 を用いた場合の疑似カラーに近づけている。

3.1.1 マップ 1

図 8 は、手法 1 を用いて優先度を設定した様子を示すものである。

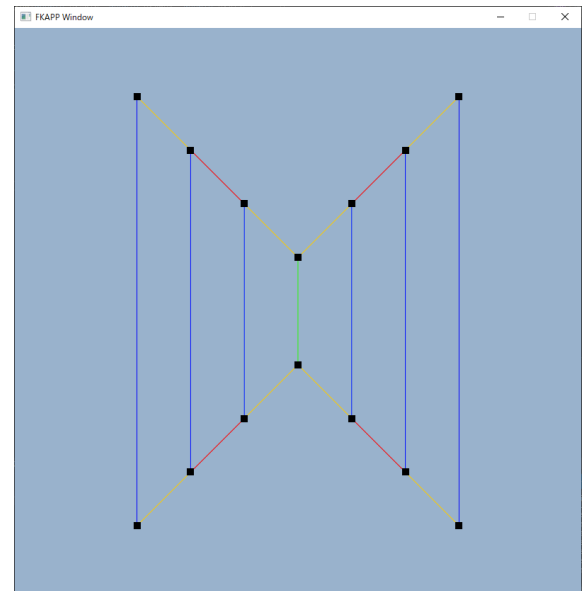


図 8 マップ 1-手法 1

赤く表示したエッジを削除すると、削除したエッジの両端のノード間での移動の際、本来は必要のなかった上下の移動が発生し、最短距離の増加量が多い。一方、青く表示したエッジを削除しても、赤く表示したエッジと比べて、最短距離の増加量が少ない。

図 9 は、手法 2 を用いて優先度を設定した様子を示すも

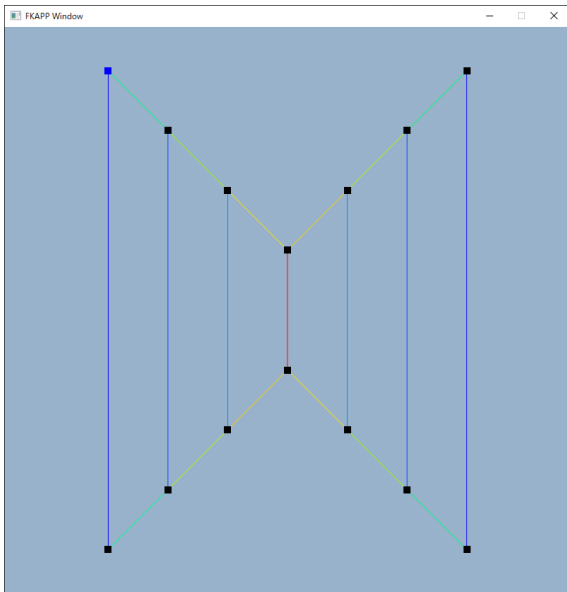


図 9 マップ 1-手法 2

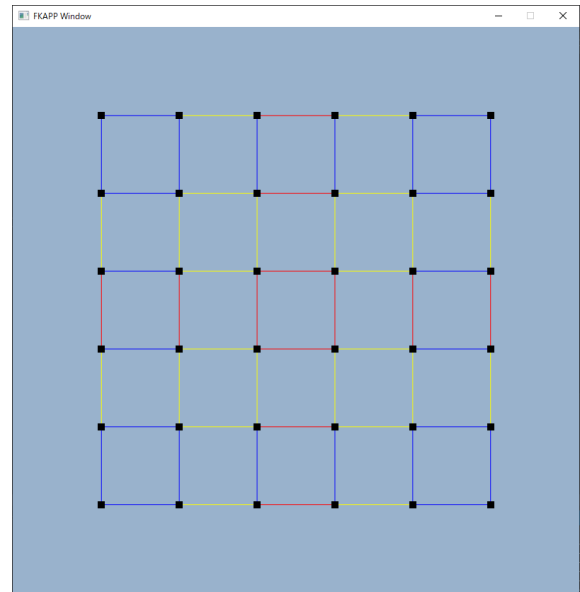


図 11 マップ 2-手法 1

のである。

マップ 1 では、マップの左上と右下、右上と左下での移動の際、最短経路はマップ中央を経由するため、マップの中心に近いエッジ程赤くなっている。

図 10 は、手法 3 を用いて優先度を設定した様子を示すものである。

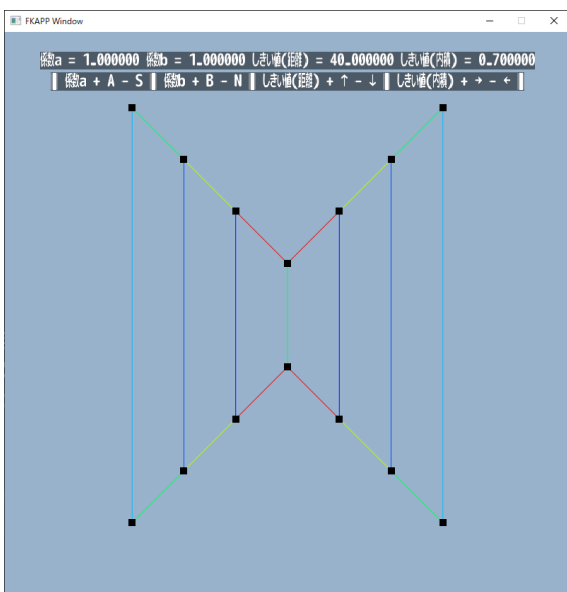


図 10 マップ 1-手法 3

マップ 1 の場合、係数や閾値を調整することで、手法 1 の結果に近い疑似カラーを出力することができた。

3.1.2 マップ 2

図 11 は、手法 1 を用いて優先度を設定した様子を示すものである。

格子状のマップにおいては、同じ列または行にあるエッジをみたとき、中央に近いほど赤くなっている。これは、

中央のエッジ程通る回数が増えるからである。マップ 2 においては、各エッジのコストは同じであるため、通る回数が多いエッジ程、削除した場合の最短距離の増加量が大きくなる。

図 12 は、手法 2 を用いて優先度を設定した様子を示すものである。

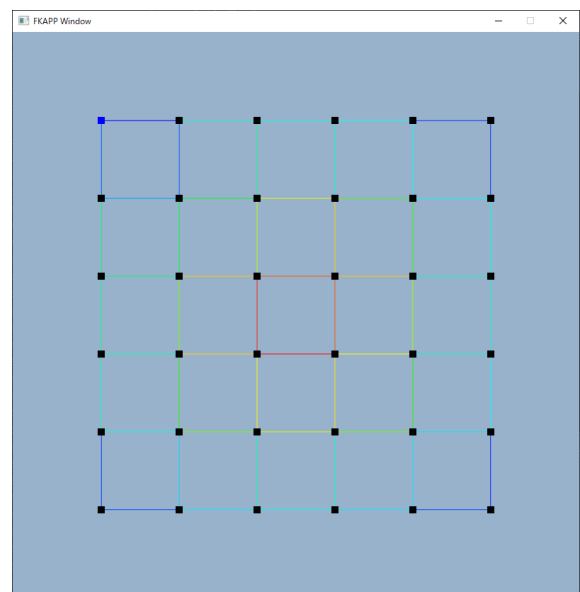


図 12 マップ 2-手法 2

格子状のマップにおいては、マップ中央に近づくほど赤くなっている。中央の 4 つのエッジをみたとき、わずかに色に違いがある。これは、最短経路の候補が複数ある場合、選ぶエッジに偏りが出ているからである。

図 13 は、手法 3 を用いて優先度を設定した様子を示すものである。

マップ 2 の場合、係数や閾値を調整することで、手法 1

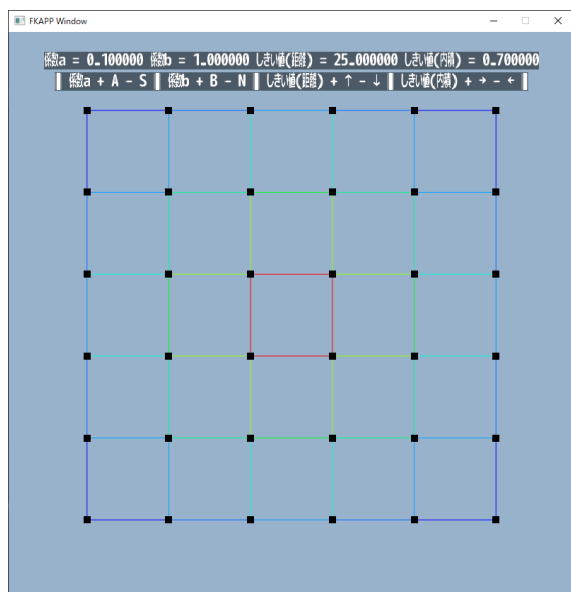


図 13 マップ 2-手法 3

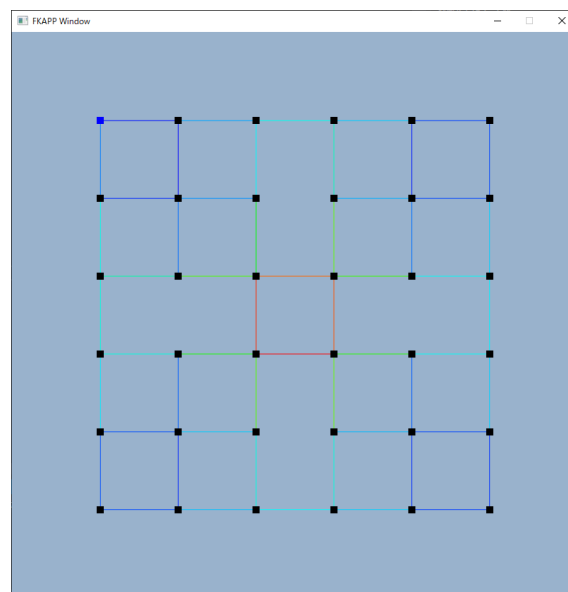


図 15 マップ 3-手法 2

の結果に近い疑似カラーを出力することができた。

3.1.3 マップ 3

図 14 は、手法 1 を用いて優先度を設定した様子を示すものである。

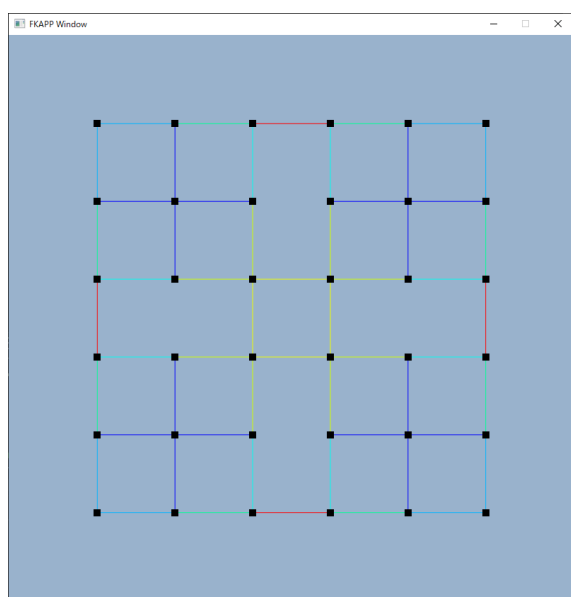


図 14 マップ 3-手法 1

赤いエッジを削除すると、中央付近のエッジを経由する必要があるため、他のエッジと比べると最短距離の増加量が多い。赤いエッジを通らない場合、中央付近のエッジを通る回数が多い。そのため中央付近のエッジを削除すると、最短距離の増加量がやや大きい。

図 15 は、手法 2 を用いて優先度を設定した様子を示すものである。

手法 1 では赤くなっていたエッジが、手法 2 では青くなっている。これは、このエッジを削除したときの最短距

離の増加量は大きい、通る回数は少ないからである。

図 16 は、手法 3 を用いて優先度を設定した様子を示すものである。

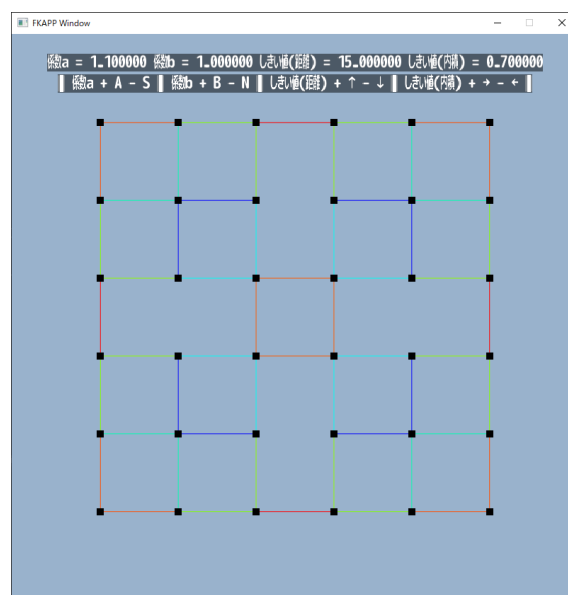


図 16 マップ 3-手法 3

マップ 3 の場合、マップ 1 やマップ 2 と比較すると、係数や閾値を調整しても手法 1 の結果に近い疑似カラーは出力できなかった。

3.1.4 マップ 4

図 17 は、手法 1 を用いて優先度を設定した様子を示すものである。

マップ 4 では横方向のエッジに比べて斜め方向のエッジのコストが小さいため、斜め方向のエッジが全体的に赤くなる傾向にある。

図 18 は、手法 2 を用いて優先度を設定した様子を示す

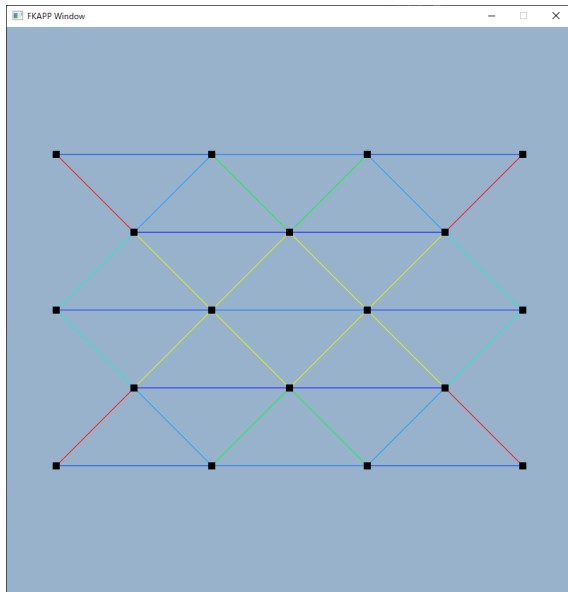


図 17 マップ 4-手法 1

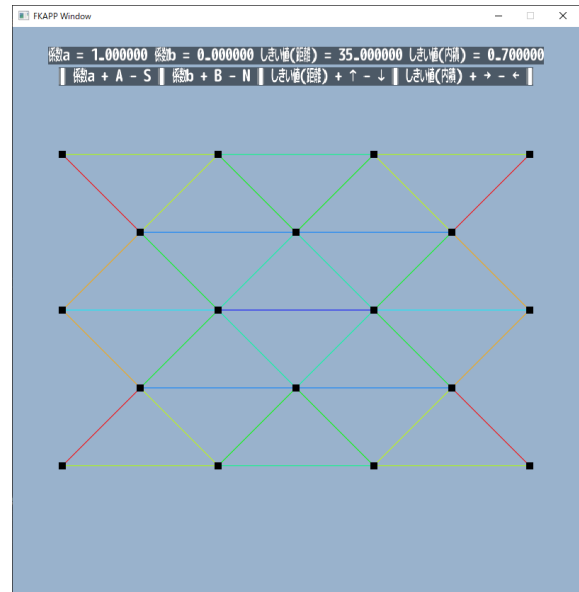


図 19 マップ 4-手法 3

ものである。

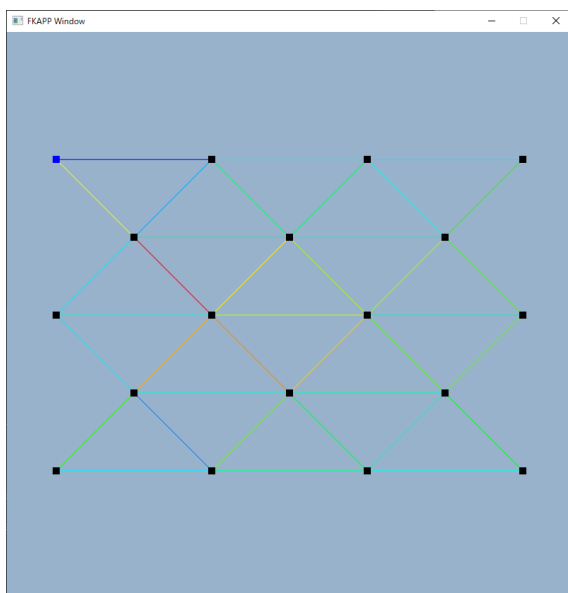


図 18 マップ 4-手法 2

マップの外側にいくほど青くなる傾向がある。また、マップ 2 と同様に最短経路の候補が複数あるため、結果に偏りがでている。

図 19 は、手法 3 を用いて優先度を設定した様子を示すものである。

マップ 4 の場合、係数や閾値を調整することで、手法 1 の結果に近い疑似カラーを出力することができた。

3.2 考察

手法 1 では、経路グラフ全体でみたときに削除しても最短距離の増加量が小さいエッジを抽出することができた。また手法 2 では、通る回数の少ないエッジを抽出すること

ができた。しかし、手法 2 では通る回数は少なくとも、最短距離の増加量が大きいエッジを抽出してしまう場合がある。また、手法 1 と手法 2 ではマップサイズの拡大に伴う処理時間の増加量が大きい。そこで、手法 3 では、係数や閾値を調整することで、手法 1 に近い結果をより少ない処理時間で求めることを図った。検証の結果、いくつかのマップにおいては、係数や閾値の調整により手法 1 に近い結果を得ることができた。しかし、マップ 3 においては、他のマップと比べると手法 1 に近い結果を得ることはできなかった。マップ 2 とマップ 3 を比較した場合、マップの中心地点の設定が上手くいってないのではないかと考えた。マップ 2 とマップ 3 はノード配置は同じだが、エッジの配置が少し異なっている。しかし、マップの中心地点はマップ 2 とマップ 3 で同じ地点に設定している。手法 3 では、経路グラフ全体の中心地点を一つだけ設定しているが、マップの形状によっては適切な中心地点を設定することが難しいことが考えられる。

4. まとめ

本論文では、経路探索の処理時間短縮の為に、経路グラフデータの削減を行うための手法を提案した。経路グラフにおけるエッジに対して削除を行うための優先度を設定することができた。しかし、現状では提案手法の適用を想定している大規模なマップではなく、簡単なマップでの検証のみになっている。提案手法の処理時間の計測ができていない。今後は、大規模な経路グラフによる評価、提案手法による優先度設定の処理時間の計測等、細かい検証を行う必要があると考える。

参考文献

- [1] 独立行政法人工業所有権情報・研修館：平成 16 年度 特許流通支援チャートカーナビ経路探索技術, <https://www.inpit.go.jp/blob/katsuyo/pdf/chart/fdenki22.pdf>. 参照: 2021.11.20.
- [2] 藤本敬介, 守屋俊夫, 中山泰一: 車輪移動ロボットの経路探索の高速化のためのマップリンク法, 日本ロボット学会誌, Vol. 33, No. 8, pp. 630–641 (2015).
- [3] 三宅陽一郎: デジタルゲームにおける人工知能技術の応用の現在, 人工知能, Vol. 30, No. 1, pp. 45–64 (2015).
- [4] E.W.Dijkstra: A note on two problems in connexion with graphs, *Numer. Math.*, Vol. 1, pp. 269–271 (1959).
- [5] 岩通ソフトシステム株式会社: ダイクストラ法アルゴリズム, <https://www.iwass.co.jp/column/column-03.html>. 参照: 2021.11.20.
- [6] Hart, P. E., Nilsson, N. J. and Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths, *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 2, pp. 100–107 (1968).
- [7] Stentz, A.: Optimal and Efficient Path Planning for Partially Known Environments, *IEEE International Conference on Robotics and Automation* (1994).
- [8] Rockstar Games, I.: Grand Theft Auto V, <https://www.rockstargames.com/V/jp>. 参照: 2021.11.23.
- [9] CHUNSOFT, S.: ARK: Survival Evolved — スパイク・チュンソフト, <https://www.spike-chunsoft.co.jp/ark/>. 参照: 2021.11.23.
- [10] 石村園子: やさしく学べる離散数学, 共立出版株式会社 (2007).