DEIM2021 A21–4

TIN 上での空間的スカイライン問合せ

笠井　雄太†　　杉浦　健人†　　石川　佳治†

† 名古屋大学大学院情報学研究科　〒 464–8603 愛知県名古屋市千種区不老町

Email: {kasai,sugiura}@db.is.i.nagoya-u.ac.jp, ishikawa@i.nagoya-u.ac.jp

あらまし　空間的スカイライン問合せは，多次元空間上におけるデータ点及び問合せ点の集合が与えられた際に他のデータ点に空間的に支配されないデータ点の集合を検索する問合せであり，多次元空間上の距離に基づきスカイライン点を列挙できる．しかし，既存の空間的スカイライン問合せはユークリッド距離に基づくため，地理空間を対象とした際に実際の移動距離と大きな差が生じうる．そこで，本研究ではオブジェクトの表面を表す際によく用いられる TIN（triangulated irregular network）上での空間的スカイライン問合せを検討する．TIN を用いたより正確な移動距離に基づく空間的スカイライン問合せを定義し，TIN 上での移動距離に関する索引や枝刈り手法を用いた効率的な解法を提案する．

キーワード　空間的スカイライン問合せ，TIN（triangulated irregular network），GIS（geographic information system）

1 はじめに

1.1 背　　景

意思決定の支援には**スカイライン問合せ**（*skyline query*）が有用である．スカイライン問合せとは，データの集合が与えられた際に他のデータに支配されない集合を検索する問合せである [1]．例えば海の近くにある安いホテルや現在地から近い評価の高い食事店など，対象データが複数の評価指標を持つ際に有効であり，スカイライン問合せを解くことで意思決定における適切な候補を列挙できる [2]．

スカイライン問合せを地理空間上に拡張した研究として，Sharifzadeh らは**空間的スカイライン問合せ**（*spatial skyline query*）を提案した [3]．空間的スカイライン問合せでは，空間上に存在する複数人からの距離を評価指標としてスカイライン問合せを定義しており，例えば集合場所の決定などに応用できる．しかし，この研究では空間上の移動距離をユークリッド距離で計算しているため，3 次元空間における高低差の激しい土地や山を跨ぐような広範囲な地域への問合せにおいて計算上で使用する距離と実際の移動距離との差が大きいという問題がある．

一方，複雑な 3 次元オブジェクトを表現するためのデータモデルとして，*TIN*（*triangulated irregular network*） [4] がよく用いられる．TIN は 3 次元の頂点と辺の集合からなるグラフデータの一種であり，三角形のみのネットワークで構成される．3 次元オブジェクトを表現するデータとしては他に点群データなどがあるが，TIN は辺の情報を用いて物体の面を柔軟かつ効率的に表現する．つまり，TIN を用いることで，3 次元オブジェクト上の移動についてより正確な距離を計算できる．

1.2 目的と貢献

そこで，本稿では TIN 空間上における空間的スカイライン問合せを提案する．距離の計算にユークリッド距離ではなく TIN 空間上の表面距離を用いることで，より正確な移動距離に基づいた 3 次元空間上での空間的スカイライン問合せを定義する．一方で，TIN 空間上の距離計算のコストは大きいため，TIN 空間上での移動距離に関する索引 [5] や枝刈り手法を用いた効率的な解法を提案する．

本稿の貢献を以下に示す．

- ユークリッド距離よりも正確な距離に基づいた TIN 空間上での空間的スカイライン問合せの提案
- 索引や枝刈りを用いた TIN 空間上の空間的スカイライン問合せの高速化
- 評価実験による提案手法の有用性の確認

提案手法の計算量について問合せ点の範囲に基づいた $O(|P'||Q|N'^2 + |P'|^2|Q|)$ を導出し，素朴な手法 $\Theta(|P||Q|N^2 + |P|^2|Q|)$ よりも高速に問題を解けることを示した．

1.3 構　　成

本稿の構成は次の通りである．まず 2 章で本稿の関連研究を述べる．次に3 章で TIN の概要と TIN 空間上の空間的スカイラインの問題定義を述べる．続いて 4 章では空間的スカイラインの解法と TIN へ適用する際の問題を述べる．提案する TIN 空間上の空間的スカイラインを求める方法は，5 章で索引と枝刈りを用いた手法を詳細に述べる．6 章では提案した解法の速度を評価実験により確認し，最後に本研究の結論を7 章で述べる．

2 関連研究

本章では，まずスカイライン問合せ及び関連した問合せ手法について述べる．その後，本稿で使用する TIN 空間上の最近傍点を求める索引について説明する．

2.1 スカイライン問合せ

スカイライン問合せは，Börzsöny ら [1] によって提案された，大きなデータの中から，他のデータに支配されない集合を検索

する問合せである．D 次元のデータをもつ $data_a$ 及び $data_b$ に関して $data_a$ が $data_b$ を**支配する**ことを $data_a \prec_Q data_b$ と表記し，次の式 (1) で定める．

$$data_a \prec_Q data_b \equiv \forall i \in [1, D], data_a(i) < data_b(i) \quad (1)$$

$data$ が全てのデータ中のどの点にも支配されない場合，$data$ をスカイライン点と呼ぶ．全てのデータに対するスカイライン点の集合がスカイライン問合せの解である．問合せの解は，支配されたデータよりも必ず優れたデータのみで構成されているため，優れたデータのみを抽出できる利点がある．

表 1　食事店までの距離及び支払金額の例

店名	距離（m）	支払金額
A	100	1500
B	150	1000
C	200	4000
D	250	800
E	300	5000

例えば，現在地からの距離及び食事店で支払う金額をもとに食事店を絞り込む際，スカイライン問合せが有用となる．表 1は食事店それぞれに対する距離及び支払金額をまとめたものである．距離については近いほうが良い，支払金額は低いほど良いと定義したときスカイライン問合せの解は，どの店にも支配されない店 A，店 B，店 D となる．

スカイライン問合せの効率的な解法には，Papadias ら [6] による分枝限定法による手法や，Chomicki ら [7] のデータの順序整列による支配関係の確認回数削減の手法が存在する．しかし，これらの研究は空間データのために考案されていないため，データオブジェクト間の空間的な関係を考慮していない．

2.2　TIN 空間の距離計算

TIN を構成する頂点数を N として，二点間の距離 D_s を求めるためのアルゴリズムには Chen [8] らによる Chen-and-Han アルゴリズムがある．最短経路を計算するために，与えられた TIN の面の展開図のパターンを探索する．時間計算量は $\Theta(N^2)$，空間計算量は $O(N)$ [9] である．改善手法として，Kapoor ら [10] による $O(N \log^2 N)$ の手法が発表されているが，アルゴリズムの詳細やその主張にはいくつかの問題があることが Schreiber ら [11] によって指摘されている．そのため，本稿では正確な距離を計算できる手法として Chen らによる $\Theta(N^2)$ の手法を TIN 空間上の表面距離の計算時間とする．

2.3　TIN 空間上の索引

本稿で使用する TIN 空間上の索引には，Shahabi ら [5] によって提案された k 近傍点を発見する *TSI*（*tight surface index*）及び *LSI*（*loose surface index*）が存在する．最近傍点を発見する方法の一つにボロノイ図が存在するが，TIN 空間上のボロノイ図の構築時間は $\Theta(|P|N^3)$ であるため，頂点数が大きい TIN に対して現実的な時間で計算が行えない問題がある．代替として TSI 及び LSI は，時間計算量 $\Theta(N^2 \log N)$，空間計算量 $O(N)$ で

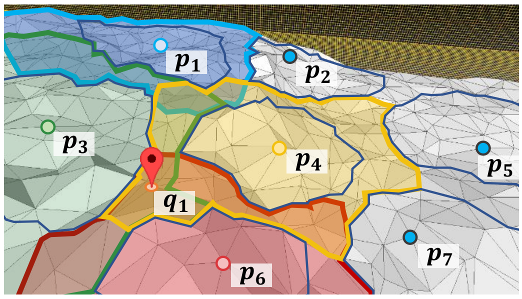


図 1　候補点集合 P に対応する tight surface index 及び loose surface index の例

索引を事前に構築し，最近傍点を $O((N/|P|)^2 + \log |P|)$ 時間で求める．すなわち，TIN 空間上での最近傍点を効率的に求める．

2.3.1　TSI の定義と性質

TSI は TIN の表現可能な面上全ての点集合 T に含まれる候補点 $p \in P \subset T$ に対して式 (2) で定義される．

$$TSI(p) \equiv \{r \mid r \in T \wedge \forall p' \in P \setminus \{p\}, D_n(p,r) < D_e(p',r)\} \quad (2)$$

中心が点 $p, p' \in P$ であるボロノイセルの境界線をなす点の集合は $D_s(p,m) = D_s(p',m)$ を満たす点 m の集合であるため，式 (6) より TSI はボロノイセルよりも狭い範囲を表現している．すなわち，$TSI(p)$ が q を含む場合は q の最近傍点が p となる．

2.3.2　LSI の定義と性質

また TSI はその定義から，TIN で表現される面上全てを覆えない場合があるため，式 (3) で定義される LSI を使用し最近傍点を発見する．

$$LSI(p) \equiv \{r \mid r \in T \wedge \forall p' \in P \setminus \{p\}, D_e(p,r) < D_n(p',r)\} \quad (3)$$

$LSI(p)$ は p のボロノイセルよりも広い範囲を表現し，p が各問合せ点 $q \in Q$ の最近傍点になりうるかを判定する．つまり，問合せ点 q が $LSI(p)$ に含まれるとき，p は q の最近傍点である可能性がある．問合せ点が $LSI(p)$ に含まれない場合，候補点 p は最近傍点になりえない．不要な候補を除外できるため，一度だけ TIN 空間上の表面距離を計算し最近傍点を発見する．

各問合せ点 q は必ず 1 つ以上の LSI に含まれる．例として，図 1に p_1，p_3，p_4，p_6 それぞれの色に対応する LSI の領域と，TSI に含まれない問合せ点 q_1 を示した．候補点 p_3，p_4，p_6 に対応する LSI は q_1 を含む一方，その他の候補点に対応する LSI は q_1 を含まず，多くの不要な候補を除外できている．

3　準　　備

本章では，TIN，TIN 空間上の空間的スカイライン問合せの定義及び，TIN 空間の距離と不等式について述べる．本稿で使用する記号とその説明を表 2に示した．

3.1　TIN

3 次元空間の**点群データ**は LiDAR やカメラ，レーザーを介して得られる．点の情報を元に 3 次元空間上の地形やオブジェクトをより正確に表現するために，一定の規則に従って領域全体

表 2　本稿で使用する記号一覧

記号	説明
T	TIN の表現可能な面全てを表す点の集合
N	TIN を構成する点の総数
N'	ネットワーク距離に基づく TIN 削減後の TIN の点の総数
P	TIN 空間上の候補点の集合
P'	最小包囲矩形による削減後の候補点の集合
Q	TIN 空間上の問合せ点の集合
S	スカイライン問合せの解（$S \subseteq P$）
S'	スカイライン問合せの途中解（$S' \subseteq P$）
$D_e(s,t)$	ユークリッド空間上の 2 点 s,t 間の距離
$D_s(s,t)$	TIN 空間上の 2 点 s,t 間の表面距離
$D_n(s,t)$	TIN 空間上の 2 点 s,t 間のネットワーク距離
$VC(p_i)$	候補点 p_i に対応するボロノイセル
$TSI(p_i)$	候補点 p_i に対応する tight surface index [5]
$LSI(p_i)$	候補点 p_i に対応する loose surface index [5]

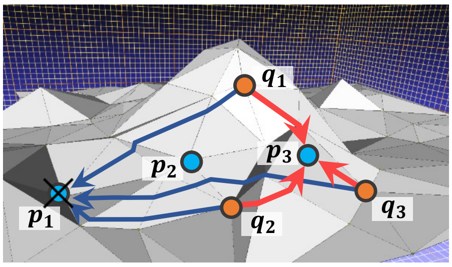


図 2　TIN 空間上の空間的スカイライン問合せで $p_3 \prec_Q p_1$ となる例

を覆う三角形の集合を生成した結果が TIN である [4]．TIN が表現する地形は，水平面上の点に標高を割り当てた連続関数のグラフであり，三角形で構成される面の集合からなる．各三角形は，隣接する三角形と頂点及び辺を共有する．

TIN は点を繋ぐ辺の情報を用いて物体の面を柔軟かつ効率的に表現する．面の表現により，TIN 空間上の距離は，3 次元の点のみで構成された空間よりも正確な距離を表現可能である．

3.2　問 題 定 義

$P = \{p_1, \ldots, p_m\}$ を TIN 空間上の候補点の集合，$Q = \{q_1, \ldots, q_n\}$ を TIN 空間上の問合せ点の集合，$D_s : \mathbf{R}^3 \times \mathbf{R}^3 \rightarrow \mathbf{R}$ を TIN 表面上の距離関数と定義する．Q に関して $p_a \in P$ が $p_b \in P$ を *TIN* **空間上で空間的に支配する**（*spatially dominate on TIN*）ことを $p_a \prec_Q p_b$ と表記し，次の式 (4) で定める．

$$p_a \prec_Q p_b \equiv \forall q \in Q, D_s(p_a, q) < D_s(p_b, q) \quad (4)$$

p が P 中のどの点にも支配されない場合，p を**スカイライン点**と呼ぶ．Q に対するスカイライン点の集合が TIN 空間上の空間的スカイライン問合せの解 $S \subseteq P$ である．例えば，図 2の TIN 空間上の空間的スカイライン問合せを考える．各問合せ点 $q \in \{q_1, q_2, q_3\}$ からの距離を計算すると全ての問合せ点において $D_s(p_3, q) < D_s(p_1, q)$ であるため，p_3 は p_1 を空間的に支配する（$p_3 \prec_Q p_1$）．つまり，p_1 はスカイライン点ではない．一方，p_2 と p_3 は問合せ点 q_2 と q_3 について互いに優位性があるため（$D_s(p_2, q_2) < D_s(p_3, q_2)$ かつ $D_s(p_3, q_3) < D_s(p_2, q_3)$），それぞれ空間的に支配されていない．したがって，この問合せの解は $S = \{p_2, p_3\}$ となる．

TIN 空間上の表面距離を求めるためには，2.2 節で述べた Chen-and-Han アルゴリズムを使用し，$\Theta(N^2)$ の計算時間が必要である．これを元に式 (4) を用いて素朴な手法でスカイライン点を判定する時間計算量は $\Theta(|P||Q|N^2 + |P|^2|Q|)$ となる．

3.3　TIN 空間上の距離と上界下界

TIN 空間上の空間的スカイライン問合せの問題の一つは，表面距離の計算に必要な時間計算量の大きさである．そこで，より高速に計算できるユークリッド・ネットワーク距離を使用する．なお，本稿におけるネットワーク距離とは TIN を構成する三角形の辺上を経路とした距離とする．問合せ点集合 Q から TIN 空間上の任意の頂点へのネットワーク距離は $\Theta(|Q|N \log N)$ で事前計算できるため，処理時には $\Theta(1)$ で使用できる．

ユークリッド距離を D_e，表面距離を D_s，ネットワーク距離を D_n で表すとき，以下の関係が成り立つ [5]．

$$D_e(s,t) \leqq D_s(s,t) \leqq D_n(s,t) \quad (5)$$

この式から，点 $p, p' \in P$ と問合せ点 $q \in Q$ について次の関係が成り立つ．

$$D_n(p,q) < D_e(p',q) \Rightarrow D_s(p,q) < D_s(p',q) \quad (6)$$

つまり，式 (4) の空間的な支配関係の一部はユークリッド距離とネットワーク距離から求められる．

$$\forall q \in Q, D_n(p,q) < D_e(p',q) \Rightarrow p \prec_Q p' \quad (7)$$

一方，p, p' と問合せ点 $q \in Q$ について $D_s(p,q) < D_s(p',q)$ となる場合，次の関係が成り立つ．

$$D_s(p,q) < D_s(p',q) \Rightarrow D_e(p,q) < D_n(p',q) \quad (8)$$

式 (8) から，$D_s(p,q) < D_s(p',q)$ を満たす必要条件は $D_e(p,q) < D_n(p',q)$ である．したがって $D_s(p,q) < D_s(p',q)$ であるかどうかを調べる際，$D_e(p,q) < D_n(p',q)$ を満たさない場合は D_s の計算なしに不等式を満たさないことが分かる．

4　空間的スカイライン問合せ

TIN 空間上での空間的スカイライン問合せの解法を提案する準備として，本章では Sharifzadeh らによって提案された空間的スカイラインの解法 [3] について簡潔に述べ，それらを TIN 空間へ適用する際の問題点を説明する．

4.1　最小包囲矩形の利用

候補点集合 P から途中解 S' が見つけられた後，新たな候補点 p_{new} がスカイライン点となりえない場合，即座に計算を終了することを目的とする．式 (9) は既知の解が p_{new} を支配し，

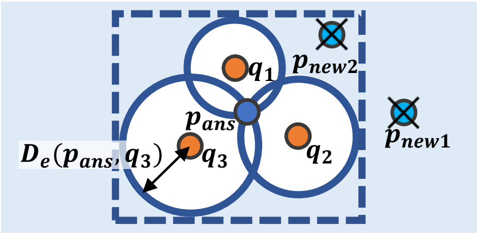


図 3　3 次元空間で p_{ans} に支配される領域

p_{new} はスカイライン点ではないことを判定する式である．

$$\exists p_j \in S', \forall q_i \in Q, D_e(p_j, q_i) < D_e(p_{new}, q_i) \tag{9}$$

これに対し Sharifzadeh らは，各問合せ点 $q \in Q$ を中心とした半径 $D_e(p_j, q)$ の円の領域内に p_{new} が含まれないとき p_{new} が p_j によって支配されると導いた．つまり，式 (9) の支配関係の判定を，ある領域に対する空間的な交差関係の判定へと帰着させた．p_{new} が支配されるかどうかの判定には通常 $\Theta(|P||Q|)$ の時間計算量が必要であるが，円の和集合に対する最小包囲矩形に点 p_{new} が含まれるかの判定は $\Theta(1)$ で実行できるため，空間的スカイライン問合せの解の効率的な検出を実現している．

例として，図 3に途中解 $p_{ans} \in S'$ によって支配されない領域（白塗りの領域）及びその最小包囲矩形（点線の矩形）を示す．p_{new1} は矩形に含まれないため $p_{ans} \prec_Q p_{new1}$ と判定でき，厳密な距離計算を行うことなく効率的に枝刈りできる．一方で，p_{new2} のように矩形の中には含まれるが円の和集合には含まれない（p_{ans} によって支配される）点も存在する．このように最小包囲矩形を用いた判定は偽陽性をもつため，p_{new2} のような点は定義に従い支配関係を判定する必要がある．

4.2　ボロノイ図及び凸包の利用

ボロノイ図とは与えられたオブジェクトの集合 P のそれぞれに対して，距離関数 d に従い最も近い領域に分割した図である [12]．式 (10) で定義される領域 $VC(p)$ を，p のボロノイセルと呼ぶ．

$$VC(p) \equiv \{r \mid r \in T \land \forall p' \in P \setminus \{p\}, d(p, r) < d(p', r)\} \tag{10}$$

［定理 1］　候補点 p に対応するボロノイセル $VC(p)$ に少なくとも 1 つ以上の問合せ点が含まれる場合，p はスカイライン点となる．

定理 1は文献 [3] で証明された．候補点集合に対応するボロノイセルを利用し，スカイライン点を判定する．

また，Sharifzadeh らは問合せ点集合 Q の凸包の内部に含まれる候補点は全てスカイライン点であることを示した．更に先のボロノイ図を利用し，ボロノイセルと凸包が交差するならばそのような $VC(p)$ をなす p はスカイライン点であることを示し，計算の簡略化を行った [3].

4.3　TIN を用いる際の問題

これらの計算量の削減はユークリッド空間における距離の使用を前提としており，TIN 空間上の表面距離では成立しない．つまり，既存手法を単純に TIN 空間へと拡張することは難しい．これは，手法で使用される凸包を表現するためには TIN 空間が線型結合を満たす必要がある一方で，TIN の距離空間が線型結合を満たしているかが明らかではないためである．

また，一部の手法を TIN 空間上に適用した場合に計算時間の問題がある．例えば支配関係を簡略化するために必要な二等分線は TIN 空間上に定義可能である一方，構築する際の時間計算量は $O(N^4)$ である．ボロノイ図についても同様に，構築するための時間計算量は $\Theta(|P|N^3)$ [8] である．これらは単純に TIN 空間上の空間的スカイライン問合せを解くよりも計算量が大きい．

したがって，本稿では現実的な計算時間で TIN 空間上の空間的スカイラインを解くことを目指す．先に述べた計算時間が大きい問題は，TIN 空間上の表面距離 D_s の計算に毎回 $\Theta(N^2)$ の時間計算量が必要であることに起因している．すなわち，高速化のためには表面距離の計算回数を削減する必要がある．

5　計算時間の削減手法

本章では TIN 空間上の空間的スカイラインを求める方法について述べる．提案手法では，素朴な方法に対して処理を追加し改善を行う．表面距離の計算回数を削減する改善手法を説明し比較を行うために，式 1 をもとに愚直にスカイラインを求める手法を Algorithm 1，提案手法を Algorithm 2に示した．Algorithm 1に示した素朴な手法の計算量は $\Theta(|P||Q|N^2 + |P|^2|Q|)$ である．一方，提案手法では先の最小包囲矩形によって絞り込まれた候補点を P'，ネットワーク距離をもとに削減した TIN の頂点数を N' としたとき，全体の計算量は $O(|P'||Q|N'^2 + |P'|^2|Q|)$ となる．提案手法による改善点は本章の各節で詳細に述べる．

5.1　TSI と LSI を用いた初期解の生成

まず，Algorithm 2の 3行目から 5行目で行う，初期解の生成について述べる．TSI 及び LSI を使用し，最近傍点に基づくスカイライン点を発見することを目的とする．定理 1及び文献 [5] で証明された定理 2より，各問合せ点に対する最近傍点を候補点の中から発見できれば，その候補点がスカイライン点である．

［定理 2］　候補点集合 P 及びその TSI が与えられたとする．各候補点 $p \in P$ について，その TSI 中の任意の点 $\forall r \in TSI(p)$ に対する TIN 空間上における P 中の最近傍点は p である．

Shahabi らによって提案された TSI 及び LSI では，TIN 空間上の最近傍点を効率的に計算する．Algorithm 2の 4行目では，スカイライン点を求める関数 `NearestNeighborSearch` を使用し，その内部で TSI 及び LSI を使用し最近傍点を発見する．

$|Q|$ 個の問合せ点に対する最近傍点は合計で 1 個以上 $|Q|$ 個以下存在する．したがって，TSI 及び LSI を用いて最近傍点を発見し，1 個以上 $|Q|$ 個以下のスカイライン点が明らかになる．これら発見済みのスカイライン点の集合を初期解とする．

問合せ点に対応するスカイライン点を発見するための計算量について述べる．関数 `NearestNeighborSearch` では，TSI 及び LSI を用いて最近傍点に基づくスカイライン点を発見する．

Algorithm 1 TIN 空間上で空間的スカイライン問合せを解く素朴なアルゴリズム

```
Input: P, Q, TSI, LSI
Output: S
function NaiveSSQonTIN(P, Q, TSI, LSI)
    S′ ← ∅
    for all p ∈ P do
        p_is_dominated ← false
        for all p′ ∈ S′ do
            if p′ ≺_Q p then                    ▷ 途中解 p′ が p を支配
                p_is_dominated ← true
            end if
        end for
        if p_is_dominated is false then         ▷ p がスカイライン点
            S′ ← S′ ∪ {p}
            p に支配される候補点が存在するならば S′ から削除
        end if
    end for
    return S ← S′
end function
```

TSI を用いた判定には，ある問合せ点が実際に TSI に含まれるかを確認するために $O(N/|P| + \log|P|)$ [5] または，式 (2) を用いた不等式による $O(|P|)$ 時間で判定ができる．つまり，1 個の問合せ点に対して $O(\min(N/|P| + \log|P|, |P|))$ で最近傍点を計算できる．LSI を用いた判定は，$O((N/|P|)^2 + \log|P|)$ で求められる [5]．関数 `NearestNeighborSearch` は $|Q|$ 回呼び出されるため，TSI と LSI を使用してスカイライン点を発見する場合の時間計算量は $O(|Q|(\min(N/|P|, |P|) + (N/|P|)^2 + \log|P|))$ となる．

5.2 判定式の簡略化

本節では，5.1 節で述べた初期解，及び判定中の途中解をもとに，特定の条件を満たす場合は D_s の計算なしに候補点がスカイライン点ではないと示すことを目的とする．TIN 空間上の空間的スカイラインを解く素朴な手法では，表面距離 D_s の計算が全体の時間計算量を支配しているが，式 (5) に示した表面距離に対する上界下界を使用し，判定式を簡略化できる．示した判定式は5.3 節で使用する．

以降，スカイライン点であるか未確認の候補点 p_{new} に対する判定について述べる．次の式 (11) を満たす場合，p_{new} はスカイライン点ではない．

$$\exists p_j \in S', \forall q_i \in Q, D_s(p_j, q_i) < D_s(p_{new}, q_i) \tag{11}$$

ここで，式 (11) が必ず満たされる，すなわち必ず p_{new} がスカイライン点ではないと判定するためには式 (12) を満たす必要がある．式 (12) は式 (11) に式 (6) を組み合わせることで導かれる．

$$\exists p_j \in S', \forall q_i \in Q, D_n(p_j, q_i) < D_e(p_{new}, q_i) \tag{12}$$

式 (11) が満たされる可能性があると判定するためには，式 (13) を満たす必要がある．式 (13) は式 (11) に式 (8) を組み合わせることで導かれる．

$$\exists p_j \in S', \forall q_i \in Q, D_e(p_j, q_i) < D_n(p_{new}, q_i) \tag{13}$$

Algorithm 2 TIN 空間上で空間的スカイライン問合せを解くアルゴリズム

```
Input: P, Q, TSI, LSI
Output: S
 1: function SSQonTIN(P, Q, TSI, LSI)
 2:     S′ ← ∅
 3:     for all q ∈ Q do
 4:         S′ ← S′ ∪ {NearestNeighborSearch(q, TSI, LSI)}
 5:     end for
 6:     bounding_box ← BoundingBoxByNetworkDist(S′, Q)
 7:     P_candidate ← RangeSearchOnRTree(bounding_box)
 8:     for all p ∈ P_candidate do
 9:         if p is out of bounding_box then
10:             continue
11:         end if
12:         p_is_dominated ← false
13:         for all p′ ∈ S′ do
14:             if p′ ≺_Q p then              ▷ 式 (12)–(13) を優先的に使用
15:                 p_is_dominated ← true
16:             end if
17:         end for
18:         if p_is_dominated is false then
19:             S′ ← S′ ∪ {p}
20:             bounding_box.update(p)        ▷ 新たな矩形を管理
21:             p に支配される候補点が存在するならば S′ から削除  ▷
                5.2 節に基づく簡略化した判定式を優先的に使用
22:         end if
23:     end for
24:     return S ← S′
25: end function
```

式 (11) が必ず満たされないと判定するためには，式 (13) の余事象となるかを確認すれば良い．よって式 (13) が満たされない場合，式 (11) は必ず満たされない．同様の方針により，p_{new} がスカイライン点である判定，及び p_{new} が途中解の一部を支配する判定についても，表面距離の計算を行わない式を導ける．

式 (12)，(13) はネットワーク距離及びユークリッド距離のみで判定可能である．つまり，p_{new} に対してそれぞれの式を満たすかどうかを全て $\Theta(|P||Q|)$ で判定できる．スカイライン点となる可能性のみがある場合は，5.4 節で述べる判定を行う．

5.3 最小包囲矩形による不要な候補点の削除

次に，5.2 節で導出した式をもとに，Algorithm 2の 6行目の最小包囲矩形の生成及び 7行目の最小包囲矩形の使用法について述べる．まず，5.1 節で発見した初期解と式 (12) をもとに，最小包囲矩形を生成する．その後，支配関係が不明な候補点全体から明らかにスカイライン点となりえない点を，最小包囲矩形によって計算の候補から外すことを目的とする．

5.3.1 最小包囲矩形の生成と利用

Algorithm 2の 6行目の最小包囲矩形の生成を行う関数 `BoundingBoxByNetworkDist` について述べる．p_{new} がスカイライン点ではないことを判定する式 (12) を最小包囲矩形による判定に置き換えるため，図 4に示す作図を行う．例として

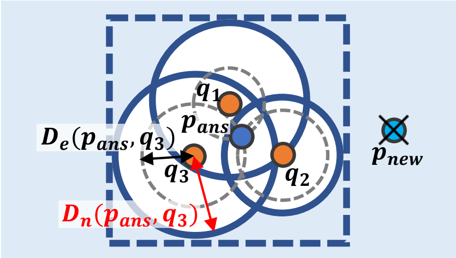


図 4　TIN 空間で p_{ans} が支配する領域と最小包囲矩形の例

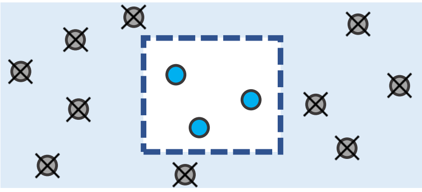


図 5　最小包囲矩形による候補点削減の例

図 4に橙色で示した問合せ点 $q_i \in Q$ を中心とし，スカイライン点 p_{ans} までのネットワーク距離 $D_n(p_{ans}, q_i)$ を半径とした円を青色で記した．更に，複数の円に対する最小包囲矩形を青い点線で記した．既存手法で円の半径は D_e であったが，D_n としても問合せ点や候補点の位置は変化しないため図形的な意味が変化せず，同様に判定できる．矩形が点を含むかどうかの判定は空間的スカイライン問合せと同様に $\Theta(1)$ で計算できる．

最小包囲矩形を生成するためのスカイライン点は，5.1 節で述べた，TIN 空間の最近傍点に基づくスカイライン点を使用する．最近傍点に基づくスカイライン点ではなく，素朴な方法で得られたスカイライン点をもとに最小包囲矩形を生成する場合，矩形が大きくなる可能性がある．一方，最近傍点に基づくスカイライン点は，候補点集合 P に含まれるどの候補点からも支配されない．すなわち，より小さい最小包囲矩形を生成可能であり，多くの候補点をスカイライン点ではないと判定できる．

5.3.2　最小包囲矩形と空間索引の連携

TIN 及び候補点のデータが RAM よりも大きい場合，DBMS の使用が考えられる．Algorithm 2の 7行目で示した関数 `RangeSearchOnRTree` では，候補点に対して R-tree [13] に代表される空間索引を事前に設定しておく．最小包囲矩形を DBMS に渡し範囲問合せを行うことで必要なデータにのみアクセスし，不要な候補点を削減できる．

削減の例として，図 5に最小包囲矩形を使用して削減された候補点を灰色，削減後に残った候補点を水色で示した．空間索引を使用しない場合は，全ての候補点が最小包囲矩形に含まれるかを逐次判定して候補点を削減する必要がある．一方，空間索引を使用する場合，最小包囲矩形に含まれる候補点のみを即座に取得でき，より高速に候補点を削減できる．これにより，削減後の候補点を P' として，空間索引を使用しない場合の時間計算量 $\Theta(|P|)$ から，$\Theta(|P'| + \log|P|)$ へと改善される．

5.4　枝刈り処理後の候補点の判定

5.2 節で導出した式をもとに，Algorithm 2の 13行目から 17行目の候補点がスカイライン点であるかの判定について述べる．最小包囲矩形による枝刈りは，4.1 節の 3 次元空間での空間的スカイラインと同じく偽陽性があるため，枝刈り後の候補点には5.2 節で導いた D_s を計算しない判定を適用する．判定後，支配関係が明らかになる場合は処理を終了し，他の候補点に対して枝刈り及び支配関係の判定を繰り返し行う．支配関係が明らかにならない場合は，5.2 節の式をもとに，ユークリッド距離とネットワーク距離を使用した判定を利用することで，表面距離の計算回数を削減しながら判定できる．処理に必要な時間計算量は，問合せ全体で $O(|P'||Q|N^2)$ である．

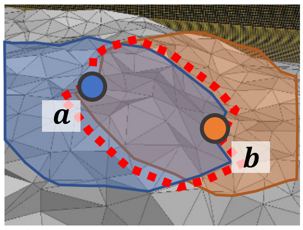


図 6　ネットワーク距離に基づく TIN 削減の例

5.5　スカイライン点発見後の処理

候補点がスカイライン点と判明した後，Algorithm 2の 18行目から 22行目の更新処理について述べる．まず，現在確認している候補点 p_{new} を途中解 S' に追加する．その後，枝刈りに使用している最小包囲矩形を新たに作成し管理する．最後に，途中解のうち p_{new} に支配される点を削除する．

削除の際は5.2 節の式をもとに，ユークリッド距離とネットワーク距離を使用した判定を利用することで，表面距離の計算回数を削減しながら判定できる．処理に必要な時間計算量は，5.4 節の判定と同じく $O(|P'||Q|N^2)$ である．

5.6　表面距離計算時の TIN の削減

表面距離を求める際，式 (5) を用いることで不要な TIN を削減し，更に計算時間を短縮できる．すなわち，2 点間の表面距離 $D_s(a, b)$ を計算する場合はその上界であるネットワーク距離 $D_n(a, b)$ 以下に位置する TIN のみを切り出すことで表面距離の計算を速く行うことが可能である．より詳細には，$D_n(a, x) \leqq D_n(a, b)$ かつ $D_n(b, x) \leqq D_n(a, b)$ をみたす点 x のみを使用した TIN で表面距離の計算を行えば良い．削減後の TIN の頂点数を N' とすると，表面距離の計算は $\Theta(N^2)$ から $O(N + N'^2)$ に短縮される．

図 6にネットワーク距離に基づく TIN 削減の例を示した．点 a を中心として $D_n(a, x) \leq D_n(a, b)$ をみたす点 x をもつ領域を青色，点 b を中心として $D_n(b, x) \leq D_n(a, b)$ をみたす点 x をもつ領域を橙色で示した．表面距離の計算前に 2 つの領域の共通範囲である，赤い点線で囲まれた領域に含まれる TIN のみを取り出すことで，必要最低限の TIN のみで表面距離を計算可能である．TIN の削減はネットワーク距離の大きさに依存するが，2 点間の距離が小さければ小さいほど計算時間が速くなる利点が存在する．

5.7 全体の計算時間

提案手法全体の計算量について述べる．5.1 節の TSI 及び LSI を用いた最近傍点の基づくスカイライン点の発見には $O(|Q|(\min(N/|P|,|P|)+(N/|P|)^2+\log|P|))$, 5.3 節の発見されたスカイライン点をもとに最小包囲矩形を作成するために $\Theta(|P'||Q|)$, 最小包囲矩形を用いた候補点の削減に $\Theta(|P'|+\log|P|)$，5.4 節で述べた削減された候補点をユークリッド距離とネットワーク距離及び表面距離を用いて判定し $O(|P'||Q|N^2)$，判定のループは $\Theta(|P'|^2|Q|)$ 時間必要である．よって全体の計算量の上界は $O(|P'||Q|N^2+|P'|^2|Q|)$ となる．更に5.6 節の手法を取り入れTIN の削減を行い，計算量の上界を $O(|P'||Q|N'^2+|P'|^2|Q|)$ とし，提案手法全体の計算時間を削減できる．

6 実　　験

提案手法の性能を評価するために実験を行った．データセットを特徴づけるパラメータが与える影響について，提案手法の計算量と一致するかを確認するために，(a) 最も計算時間を要する D_s の計算回数，(b) 全体の実行時間，を測定した．

6.1 実験環境と問合せのパラメータ

表 3　評価実験で用いたパラメータの一覧

パラメータ	設定値（既定値）
候補点	100, 200, 300, 400
問合せ点	5, 10, 15, 20
TIN の面積に対する 問合せ点の最小包囲矩形の面積の割合 [%]	1, 2, 3, 4
TIN の頂点数	1000, 2000, 3000, 4000

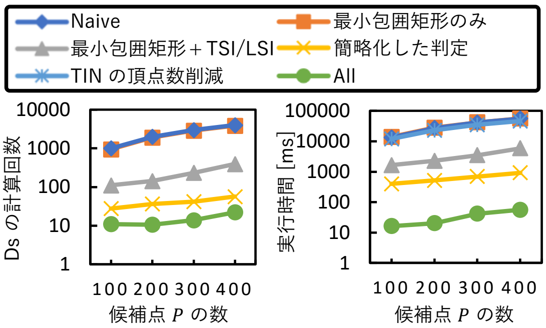
実験で用いたパラメータを表 3に示した．各実験では 1 つのパラメータが与える影響を確認するため，それ以外のパラメータは既定値を使用し，TIN，候補点及び問合せ点をプログラムの入力とした．また，TIN データは GitHub で公開した TIN 生成を行うプログラム[1]によって作成した．表面距離の計算には，計算幾何学のライブラリとして公開されている CGAL[2]の実装を使用した．実験環境の OS は Ubuntu 20.04.2 LTS，CPU は Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz，RAM は 32GB を使用し，実装は C++，コンパイラは gcc(9.3.0)，コンパイル時のオプションは O3 を指定した．TIN の立体的な地形による計算のばらつきを考慮するため，パラメータに基づく候補点と問合せ点を乱数により 30 組作成し，これらを入力とした場合の計算回数と実行時間の平均値を実験の結果とした．

6.2 それぞれの手法がもたらす実行時間への影響

5 章で説明した計算時間を減らすための手法のうち，以降の節で述べる 4 つについて，それぞれが計算時間にどの程度影響を及ぼすかを明らかにする．表 3に示した 4 つのパラメータについて表面距離 D_s の計算回数（a）及び実行時間（b）の変化を測定し，結果を図 7–10に示した．

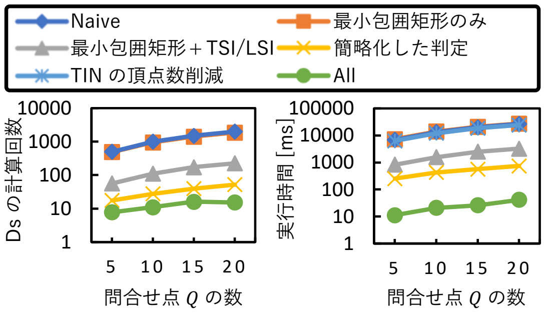
1：https://github.com/Yang-33/SpatialSkylineQueries-on-TIN

2：https://github.com/CGAL/cgal/blob/287836254f9541aa2b8faedbe08a7304d6097b7f/Surface_mesh_shortest_path/include/CGAL/Surface_mesh_shortest_path/Surface_mesh_shortest_path.h



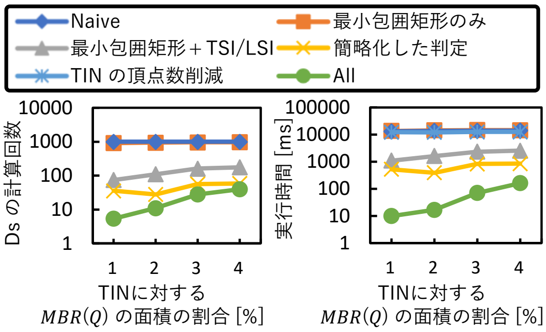
(a) 候補点の数に対する Dsの計算回数の変化　(b) 候補点の数に対する 実行時間の変化

図 7　候補点の数の変化に対する比較



(a) 問合せ点の数に対する Dsの計算回数の変化　(b) 問合せ点の数に対する 実行時間の変化

図 8　問合せ点の数の変化に対する比較

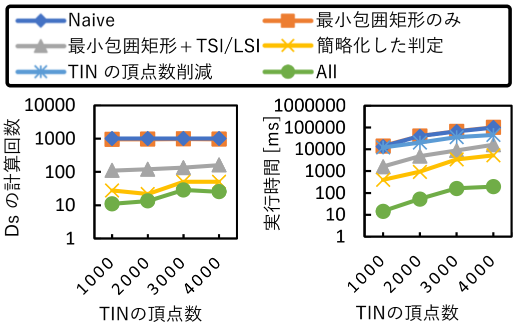


(a) 問合せ点の範囲に対する Dsの計算回数の変化　(b) 問合せ点の範囲に対する 実行時間の変化

図 9　問合せ点の位置の範囲の変化に対する比較

6.2.1 初期解に基づかない最小包囲矩形による候補点削減実行時間への影響

まず初めに，素朴な方法でスカイライン点が発見された後に，5.3 節で説明した最小包囲矩形を作成し，候補点を削減した場合の評価を行う．図 7–10の橙色の点が結果に対応する．削減された候補点の数は 5 %未満であり，大きな計算時間の短縮はされなかった．これは，5.3.1 項で述べたように，適当な順序で得られたスカイライン点をもとに生成した最小包囲矩形は大きな範囲となるため，削減できる候補点が僅かであるからと考えられる．つまり，単に最小包囲矩形を使用しただけでは効果的



(a) TINの頂点数に対する Dsの計算回数の変化　(b) TINの頂点数に対する 実行時間の変化

図 10　TIN の頂点数の変化に対する比較

な候補点の削減は行われないといえる．

6.2.2　TSI 及び LSI により得られる初期解に基づく最小包囲矩形を用いた場合の実行時間への影響

次に，TSI 及び LSI で発見したスカイライン点をもとに，5.3 節で説明した最小包囲矩形を作成し，候補点を削減した場合の評価を行う．5.3.1 項で述べたように，TSI 及び LSI で発見したスカイライン点をもとに矩形を作成するとより小さい最小包囲矩形を生成可能である．図 7–10の灰色の点が結果に対応する．初期解をもとに作成した最小包囲矩形により候補点を削減した場合，候補点の数はいずれも 10 %未満まで削減された．また，図 9が示すように問合せ点の範囲が増加するにつれて実行時間は増加する．これは矩形の大きさが変化するため，削減される候補点の数が増加したからと説明できる．したがって，問合せの範囲が全体の計算時間に影響をもたらす．

6.2.3　判定式の簡略化を用いた際の実行時間への影響

続いて，素朴な方法に対して，支配関係の判定時に表面距離の上界下界の式 (5) を用いて表面距離の比較を簡略化し，計算回数を削減する手法の評価を行う．表面距離の上界下界で比較が成立しない場合，正確な表面距離を計算した．図 7–10の黄色の点が結果に対応する．素朴な方法に対してこの手法を適用した場合，97 %以上の計算で簡略化した判定が行われ，表面距離の計算を行う必要がないことが示された．

6.2.4　TIN の削減による実行時間への影響

最後に，5.6 節で説明したネットワーク距離に基づく TIN の頂点数の削減を素朴な方法に適用した場合の評価を行う．図 7–10の水色の点が結果に対応する．素朴な方法と表面距離の計算回数は同じであるため，各図の（a）への記載は省略した．結果，いずれのパラメータを変化させても，計算時間が 20 %程短縮された．特に，TIN の頂点数を増やした場合，図 10（b）が示すように計算時間は短縮された．これは，表面距離の計算時間が $\Theta(N^2)$ であるため，削減後の TIN の割合の二乗分計算時間が短縮された結果と説明できる．

6.3　素朴な手法と提案手法全体の実行時間の比較

素朴な手法と，提案手法を全て使用した場合の比較を行った．図 7–10の緑色の点が結果に対応する．全て規定値の場合，提案手法は素朴な手法と比較し，実行時間を最も支配する D_s の計算が 1000 回から 10 回と 100 倍，実行時間は 17614 ms から 13 ms と 1354 倍短縮された．最小包囲矩形の利用，スカイライン点の簡略化した判定，及び TIN の頂点数削減はそれぞれ独立した実行時間の削減方法である．このため，比較した他のどの場合よりも，表面距離の計算回数及び実行時間が小さい．

7　結　　論

本稿では，ユークリッド距離に基づく空間的スカイライン問合せに対して，より正確な移動距離に基づいた TIN 空間上での空間的スカイライン問合せを提案し，問合せを実行する方法及び改善点を提案した．TIN の距離空間では既存研究の効率的な手法を直接適用できないため，複数の手法を素朴な解法に追加し効率的な実行を目指した．更に，提案手法の計算量について問合せ点の範囲に基づいた $O(|P'||Q|N'^2 + |P'|^2|Q|)$ を導出し，素朴な手法 $\Theta(|P||Q|N^2 + |P|^2|Q|)$ よりも高速に問題を解けることを示した．特に，頂点数の大きい TIN 及び候補点が与えられ，問合せ点が一部の領域にのみに存在する場合，提案手法は素朴な手法よりも高速に問合せの解を求められることを確認した．

謝　　辞

本研究は JSPS 科研費（JP16H01722，JP19K21530，JP20K19804）の助成,，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）及び地球環境情報プラットフォーム構築推進プログラム（DIAS）の委託業務による．

文　　献

[1] S. Börzsöny, D. Kossmann, and K. Stocker, “The skyline operator,” in *Proc. ICDE*, pp. 421–430, 2001.
[2] C. Kalyvas and T. Tzouramanis, “A survey of skyline query processing,” *CoRR*, vol. abs/1704.01788, 2017.
[3] M. Sharifzadeh and C. Shahabi, “The spatial skyline queries,” *Proc. VLDB*, pp. 751–762, 2006.
[4] T. Peucker, “The triangular irregular network,” in *Proc. of the Digital Terrain Models (DTM) Symposium, American Society of Photogrammetry*, pp. 516–540, 1978.
[5] C. Shahabi, L. Tang, and S. Xing, “Indexing land surface for efficient kNN query,” *Proc. VLDB*, vol. 1, no. 1, pp. 1020–1031, 2008.
[6] D. Papadias, Y. Tao, G. Fu, and B. Seeger, “An optimal and progressive algorithm for skyline queries,” in *Proc. ACM SIGMOD*, pp. 467–478, 2003.
[7] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang, “Skyline with presorting,” in *Proc. ICDE*, pp. 717–719, 2003.
[8] J. Chen and Y. Han, “Shortest paths on a polyhedron,” in *Proc. Annual Symposium on Computational Geometry (SoCG)*, pp. 360–369, 1990.
[9] S.-Q. Xin and G.-J. Wang, “Improving Chen and Han’s algorithm on the discrete geodesic problem,” *ACM Trans. Graph.*, vol. 28, no. 4, pp. 104:1–104:8, 2009.
[10] S. Kapoor, “Efficient computation of geodesic shortest paths,” in *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pp. 770–779, 1999.
[11] Y. Schreiber and M. Sharir, “An optimal-time algorithm for shortest paths on a convex polytope in three dimensions,” in *Twentieth Anniversary Volume:*, pp. 1–80, Springer, 2009.
[12] M. De Berg, M. Van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry*. Springer, 1997.
[13] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, “The R*-tree: an efficient and robust access method for points and rectangles,” in *Proc. ICDE*, pp. 322–331, 1990.