移動オブジェクトを対象とした時空間データ管理の一手法

学生員 王 軼群* 非会員 野澤 博*
 非会員 土方 嘉徳* 非会員 仲谷 美江*
 正員 西田 正吾*

Spatio-Temporal Data Management for Moving Objects

Yiqun Wang*, Student Member, Hiroshi Nozawa*, Non-member, Yoshinori Hijikata*, Non-member,
 Mie Nakatani*, Non-member, Shogo Nishida*, Member

This paper deals with spatio-temporal indexing method for moving objects. In our research, we propose XAT (eXtended Adaptive Tree) structure, consisting of spatial trees and temporal trees, for fast search for spatio-temporal data. The searching process in XAT structure is divided into two steps. The first step roughly narrows down the potential solutions (moving objects) according to the given searching range. The last step fixes the real solution by checking the object's moving track. We compare XAT structure and 3D structure, one of the conventional methods, by computer simulation. The result shows that XAT structure works faster than 3D structure when there is difference between the spatial search range and temporal search range and the objects' moving areas are small.

キーワード：移動オブジェクト，時空間データ管理，範囲検索，木構造，XAT 構造，3D 管理構造

Keywords: moving object, spatio-temporal data, range search, tree structure, 3D structure, XAT structure

1. はじめに

存在位置を特定可能な機能を備えた携帯端末の普及に伴い、オブジェクトの移動データが入手しやすくなりつつある。そこで、入手した移動データを解析し、その結果を将来への対策に役立てるようなニーズが出てくると考えられる。そのような解析を支援する機能の一つとして、ある時間帯にある場所に存在したオブジェクトを検索する機能が挙げられる。

筆者らは静止オブジェクトを対象として、ある検索範囲（空間範囲と時間範囲）中に含まれるオブジェクトを効率的に検索する方式である AT 構造（Adaptive Tree 構造）を提案してきた⁽¹⁾。一般的にデータを木構造で管理することは高速検索の基礎となるが、検索範囲が広がると多くの中間ノードや葉ノードを探索することになり、検索速度が低下する問題がある。AT 構造は、空間データから作成した木構造（空間木）と、時間データから作成した木構造（時間木）を用意しておき、検索範囲から、空間の検索範囲と時間の検索範囲の大小を比較し、検索範囲の狭い方の木構

造を選択する方式である。

本研究では、AT 構造の考え方を移動オブジェクトの検索に拡張した XAT 構造（eXtended Adaptive Tree 構造）を提案する。一般的に、ある一つの移動オブジェクトの時空間データは、その移動軌跡を一定の時間間隔で区切った移動線分データとして表現される^{(2)~(5)}。最も基本的な検索方式としては、この移動線分データを直接木構造で管理する方式が考えられる⁽⁴⁾。つまり、移動線分データを木構造の葉ノードで管理し、木構造を葉ノードまで探索した後に、その移動線分を持つ移動オブジェクトを検索結果とする方式である。しかし、この方式では、移動オブジェクトの数が増大した時に検索時間が増大するだけでなく、より動きを詳細に追跡しようとした時にも、移動線分データの数が増大し、検索時間が増大する問題がある。

XAT 構造ではこの問題に対し、最初に検索範囲に入りそうなオブジェクトを空間木と時間木から構成される木構造で絞り込み、その後に各オブジェクトの移動線分データが検索範囲に入るか否かのチェックを行なうことで検索時間の高速化を図る。

本稿では、2. で関連研究について述べる。3. で、筆者らが提案する XAT 構造について説明する。4. では、XAT 構造を計算機実験により評価する。5. では、オブジェクトの範囲検索以外の検索要求に対する実験を行うことで、また

* 大阪大学

〒560-8531 豊中市待兼山町 1-3

Osaka University

1-3, Machikaneyama, Toyonaka, Osaka 560-8531

他の時空間データ管理方式に XAT 構造と同様の工夫をして実験を行い、それと XAT 構造を比較してみることで、より詳しく XAT 構造の特性を調べる。最後に、6. でまとめを述べる。

2. 関連研究

時空間データの管理方式には、様々な方法論が提案されている。これらの多くは、多次元データを木構造で管理するための基礎的なアルゴリズムである k-d 木⁽⁶⁾、R-tree⁽⁷⁾、BD 木⁽⁸⁾、MD 木⁽⁹⁾などを応用している。どのような観点から検索速度の向上を目指すかと言う点に着目すると、大きくは次の3種類に分類される。

(1) 時間に対して、検索速度向上を目指す方式

MT(Multi-Tree) 構造とその発展型 (MR-Tree⁽¹⁰⁾、MP Tree⁽¹¹⁾、HR-Tree⁽¹²⁾、Overlapping Linear Quard-Trees⁽¹³⁾)、2+3 R-tree⁽⁴⁾

(2) 空間に対して、検索速度向上を目指す方式

ST(Single-Tree) 構造とその発展型 (RT-Tree)⁽¹⁴⁾

(3) 時間と空間に対して、均一の検索速度を目指す方式

3D(3 Dimensional) 管理構造とその発展型 (3D R-Tree)⁽¹⁵⁾

MT 構造とその発展型は、空間情報のみに基づいた木構造をいくつかの時区間ごとに作成する。現在 (最新) の時間のデータに対して検索を行なうことが多い場合や、過去の検索においても、検索条件の時間範囲が狭い場合に有効な手法である。

ST 構造とその発展型は、空間情報にのみ基づいた木構造を一つだけ作成し、時間情報は葉ノード中にあるオブジェクトのデータの属性として扱う。検索条件において、空間範囲が狭い場合に有効な手法である。

3D 管理構造とその発展型は、位置 (2 次元) と時間 (1 次元) で構成される 3 次元空間を木構造で管理するものである。木構造は、位置 (x,y) と時間 (t) の 3 つの属性を繰り返し順番に用いることで作成される (実用上、高さに相当する z 軸に関するデータは、DEM と呼ばれるオブジェクトの属性値として扱われることが多い⁽¹⁶⁾)。時間と空間、また現在と過去に対して、均一の検索速度を提供するため、分類した 3 種類の手法の中では、最も汎用性の高い手法である。

筆者らが提案する XAT 構造は、3D 管理構造と同じく、時間と空間に対して均一の検索速度を提供することを目指すものである。XAT 構造が 3D 管理構造と異なるのは、(1) 1 つの木構造で時空間データを管理するのではなく、時間データを用いる木構造と空間データを用いる木構造を別々に作成して適応的に切り替える点と、(2) 一度オブジェクト単位で検索候補を絞り込んでから詳細な移動経路をチェックする点にある。本稿では、4 節にて XAT 構造と 3D 管理構造をシミュレーション実験により比較している。

3. XAT 構造

(3・1) 移動オブジェクトデータ 移動オブジェクトとは、時間の経過と共に空間中を移動するオブジェクトのことである (以降、特に必要がない限り単にオブジェクトと呼ぶ)。その移動経路は曲線として表されるが、この曲線を計算機で管理することは難しい。そこで、一般的にオブジェクトの移動経路データの管理においては、移動経路を微小時間で区切ることによって複数の線分のつながりとして考え、その線分群を管理する (図 1)^{(2)~(5)}。本研究においても、オブジェクトの移動経路データを、この移動線分で表すものとする。

(3・2) AT 構造と XAT 構造 AT 構造における基本原理は、データを管理する木構造を、時間情報のみから作成するもの (時間木) と、空間情報のみから作成するもの (空間木) に分けて構築しておき、検索条件の時間範囲 (以下、時間検索範囲) と空間範囲 (以下、空間検索範囲) の割合の大きさから、検索条件の厳しい方の木構造を検索することにある⁽¹⁾。XAT 構造は、この考え方を移動オブジェクトの検索にも適用できるように拡張したものである。

XAT 構造では、検索の高速化のために以下の工夫を行う。

(1) 時間木または空間木から検索結果の候補となるオブジェクトを絞り込んだ後、そのオブジェクトの各移動線分が検索範囲に含まれるか否かを検査する。

(2) 上記検査時には、移動線分データを先頭から逐次的にチェックするのではなく、これらも空間木と時間木 (実際には二分探索木) で管理する。

また、AT 構造の時間木では、本来時間軸に線分 (区間) データとして表されるオブジェクトの存在時間範囲の情報を、二次元平面上の点データに射影し、その点データを管理している。このことも、移動オブジェクトの検索の高速化に貢献すると考えられる。木構造で管理する対象データが一次元直線上の線分データであれば、各中間ノードの分岐条件は範囲となる。すると、異なる中間ノードであっても、その分岐条件の範囲には重なりができることがある。これが二次元平面上の点データであれば、各中間ノードの分岐条件は 2 つのどちらかの軸上での閾値となり、分岐条件に重なりのある中間ノードはできない。この違いは、探索する中間ノード・葉ノード数に影響するため、検索時間にも影響する。AT 構造の時間木におけるこの工夫は、静

図 1 移動物体の移動軌跡の線分化

Fig. 1. Line segments of moving objects.

図2 XAT 検索の流れ

Fig. 2. Flow of search in XAT.

止オブジェクトだけでなく、移動オブジェクトにおいても有効と考えられる。

XAT 構造における処理の流れは図2に示され、大きくはオブジェクトを管理する部分（以下、オブジェクト管理部）と、各オブジェクトごとにその移動線分を管理する部分（以下、線分管理部）に分けられる。オブジェクト管理部と線分管理部はそれぞれ、検索判断ロジック部、空間木、時間木の3つの部分から構成される。これら3つの部分は、オブジェクト管理部においても、線分管理部においても、ほぼ同様の機能を果たす。以降では、オブジェクト管理部における検索判断ロジック部、空間木、時間木を指す時はそれぞれ、「オブジェクト用検索判断ロジック部」、「オブジェクト用空間木」、「オブジェクト用時間木」と呼ぶ。線分管理部における検索判断ロジック部、空間木、時間木を指す時はそれぞれ、「線分用検索判断ロジック部」、「線分用空間木」、「線分用時間木」と呼ぶ。オブジェクト管理部と線分管理部の両方の検索判断ロジック部、空間木、時間木を指す時は、単に「検索判断ロジック部」、「空間木」、「時間木」と呼ぶ。以下の節では、これら3つの部分をオブジェクト管理部、線分管理部に分けて説明する。

〈3・3〉 オブジェクト管理部

〈3・3・1〉 オブジェクト用検索判断ロジック部 オブジェクト用検索判断ロジック部では、全時間範囲に対する

図3 空間平面、時間軸への射影

Fig. 3. Projection on space plane and time axis.

図4 外接長方形

Fig. 4. Projection on space plane and time axis.

時間検索範囲の割合 p_{obj} と、全空間範囲に対する空間検索範囲の割合 q_{obj} を求め、その割合の小さい方の木構造（オブジェクト用空間木またはオブジェクト用時間木）を選択する。具体的には、 $q_{obj} < p_{obj}$ であればオブジェクト用空間木を選択し、 $p_{obj} \leq q_{obj}$ であればオブジェクト用時間木を選択する。

〈3・3・2〉 オブジェクト用空間木 オブジェクト用空間木は、オブジェクトの3次元空間（位置+時間）中における移動経路情報のうち、位置に関する情報のみを使い、オブジェクトを管理する木構造である。3次元空間中の移動線分群から、時間軸を無視して空間平面上に射影してできた線分群をオブジェクト単位でまとめ（図3）、その中心点をk-d木を用いて管理する。k-d木とは、最も基礎的な多次元空間データの管理手法で、多次元空間中の点データ群を、データの個数が半分ずつになるように空間を分割して、それを木構造にしたものである⁽⁶⁾。

木構造の中間ノードにおける分岐条件の設定のために、空間平面上に射影されたオブジェクトの移動軌跡に外接長方形(MBR: minimum bounding rectangle)⁽¹⁶⁾を設定する（図4）。外接長方形は、オブジェクトのすべての線分を含み、なおかつ最小になるように設定する。バケット型管理方式を用いることとし、葉ノード中の全オブジェクトの外接長方形を含む最小の長方形をその葉ノードの外接長方形とする。各中間ノードにおいては、その2つの子ノードの外接長方形を共に含む最小の長方形をその中間ノードの外

図5 オブジェクト用時間木の構築方法

Fig.5. Construction of temporal tree for objects.

接長方形とする。以上の手順を葉ノードから根ノードに向かって再帰的行なうことにより、全てのノードに分岐条件として外接長方形の情報を付加する。

オブジェクト用空間木を検索する方法は、空間検索範囲と現時点のノード P に付加した外接長方形を比較し、重なりがある子ノードをたどっていく。子ノードに重なりがある限り再帰的に領域の比較を繰り返し、到達したすべての葉ノードにあるオブジェクトを獲得する。獲得したオブジェクトにおいて、その存在時間範囲と時間検索範囲とに重なりがあるか否かを検査し、重なりがあればそのオブジェクトを検索結果の候補として出力する（これは、不要にオブジェクトの木構造を探索することを防ぐためである）。

〈3.3.3〉オブジェクト用時間木 オブジェクト用時間木は、オブジェクトの3次元空間（位置+時間）中における移動経路情報のうち、時間に関する情報のみを使い、オブジェクトを管理する木構造である。ここではまず、オブジェクトの開始（発生）・終了（消滅）時間を X・Y 軸にとり、図3のように2次元平面上の点情報に変換する。このことにより時区間データを点情報として、データ構造化することが出来る。このようにデータ変換を行うと、全てのデータは図5に示すように領域 $y \leq x$ ($x \geq 0, y \geq 0$) の三角形の範囲に含まれることになる。なぜなら、開始時間は終了時間より必ず前にあるからである。この三角形の範囲内にある点情報は3分木を用いて管理される（図5）。

オブジェクト用時間木の作り方は、まずデータを終了時間でソーティングし、終了時間の小さい方から全体の4分の1にあたるデータを探し出す。次に、データの開始時間に着目し、開始時間の小さい方から全体データ数の4分の1にあたるデータを探し出す。その後、この2つのデータの平均値 a を用いて時間データ分布を図5のように3分割する。三角形になった領域は以下再帰的に分割していく。長方形領域は k-d 木を用いて2分木で再帰的に分割を繰り返す。

オブジェクト用空間木を検索する方法は、時間検索範囲と現時点のノード P における分岐条件を比較し、重なりがある子ノードをたどっていく。子ノードに重なりがある限り再帰的に領域の比較を繰り返し、到達したすべての葉ノードにあるオブジェクトを獲得する。獲得したオブジェクト

において、その移動範囲（外接長方形）と空間検索範囲とに重なりがあるか否かを検査し、重なりがあればそのオブジェクトを検索結果の候補として出力する（これは、不要にオブジェクトの木構造を探索することを防ぐためである）。

〈3.4〉線分管理部

〈3.4.1〉線分用検索判断ロジック部 線分管理部における検索判断ロジック部では、対象とするオブジェクトの存在時間範囲に対する時間検索範囲の割合 p_{seg} と、同存在空間範囲に対する空間検索範囲の割合 q_{seg} を求め、その割合の小さい方の木構造（線分用空間木または線分用時間木）を選択する。具体的には、 $q_{seg} < p_{seg}$ であれば線分用空間木を選択し、 $p_{seg} \leq q_{seg}$ であれば線分用時間木を選択する。

〈3.4.2〉線分用空間木 線分用空間木は、各オブジェクトの移動線分の3次元空間（位置+時間）中における情報のうち、位置に関する情報のみを使い、そのオブジェクトの移動線分を管理する木構造である。具体的には、3次元空間中の移動線分から、時間軸を無視して空間平面上に射影してできた線分の中心点を k-d 木を用いて管理する。

ここでも、オブジェクト用空間木と同様に外接長方形を設定するが、外接長方形は各線分に対して設定する。バケット型管理方式を用いることとし、葉ノード中の全線分の外接長方形を含む最小の長方形をその葉ノードの外接長方形とする。各中間ノードにおける外接長方形は、オブジェクト管理部の空間木（〈3.3.2〉節）と同様に設定する。

線分用空間木を検索する方法も、オブジェクト用空間木の場合と同様である。ただしこの場合、必ず深さ優先で探索し、葉ノードに到達した時点で、その葉ノード中の各移動線分が検索範囲（時間検索範囲と空間検索範囲）に重なるか否かを検査する。重なっている移動線分が一つでも発見されれば、その時点でこの木の探索を終了し、その移動線分を持つオブジェクトを検索結果として出力する。

〈3.4.3〉線分用時間木 線分用時間木は、各オブジェクトの移動線分の3次元空間（位置+時間）中における情報のうち、時間に関する情報のみを使い、移動線分を管理する木構造である。オブジェクト用時間木と異なり、線分用時間木は二分探索木となる。これは、オブジェクトが特定された時点で、移動線分の時間情報は、時間軸にソートされた状態で重なりなく並んでいるためである。線分用時間木の検索は、必ず深さ優先で探索し、葉ノードに到達した時点で、その葉ノード中の各移動線分が検索範囲（時間検索範囲と空間検索範囲）に重なるか否かを検査する。重なっている移動線分が一つでも発見されれば、その時点でこの木の探索を終了し、その移動線分を持つオブジェクトを検索結果として出力する。

4. XAT 構造の評価

〈4.1〉評価の目的 本節では、計算機上のシミュレーション実験により XAT 構造の評価を行う。本評価の目的は、(1) 最も代表的な既存の時空間データ管理手法である

3D 管理構造と比較を行うことと、(2) XAT 構造における高速化のための工夫が検索時間の短縮に貢献しているの可否かを明らかにすることにある。3節でも述べたように、高速化の工夫点は以下の3点(AT 構造から引き継いだものを1点含む)である。

(1) オブジェクト絞り込み

検索結果となる可能性のあるオブジェクトを絞り込んでから、詳細な動きを検査する。

(2) 木構造の二重化

移動線分データを先頭から逐次的にチェックするのではなく、これらも空間木と時間木で管理する。

(3) 時間情報の点データ化

本来時間軸に線分データとして表されるオブジェクトの存在時間範囲の情報を、点データに射影する。

このうち、工夫(2)に関しては、線形リストよりも木構造の方が高速な検索となることは明らかなので、実験により評価をするまでもない。そこで、工夫(1)と工夫(3)を行わないバージョンの XAT 構造を、それぞれ XAT 構造 (Type 1)、XAT 構造 (Type 2) として実装し(それぞれの手法の詳細は(4・2)で説明する)、これらとオリジナルの XAT 構造を比較する。

〈4・2〉 比較対象手法の詳細 比較対象手法の詳細は次のようになる。

(1) 3D 管理構造

位置(2次元)と時間(1次元)で構成される3次元空間を木構造で管理するものである。具体的には、移動線分を中心点で代表させ、その中心点を k-d 木を用いて管理する(図6)。線分の領域情報は、開始点と終了点の2点を含み、時間軸と空間軸の3つの軸を使って作成できる外接直方体として表現する。木構造の中間ノードでは、この外接直方体のデータを分岐条件として設定している。

(2) XAT 構造 (Type 1)

XAT 構造のようにオブジェクトを絞り込む木構造と移動線分をチェックするための木構造を持つのではなく、1つの時間木と空間木で直接すべてのオブジェクトの移動線分データを管理する。

(3) XAT 構造 (Type 2)

XAT 構造のように、オブジェクトの存在時間情報を、開始時間軸と終了時間軸から構成される2次元平面上に点データとしてマッピングするのではなく、そのまま1次元の時間軸上に表される線分データとして扱う。時間軸上でオブジェクトまたは移動線分の存在時間範囲の中心点の数がちょうど2分ずつされるように木構造を作成した後、各中間ノードに、存在時間の範囲情報を分岐条件として設定している(図7)。

〈4・3〉 標準実験条件 実験用にオブジェクトの移動データを作成した。今回の実験では移動の仕方として、直線運動、ランダム運動、正弦波運動の3種類を作成した。これらの運動を組み合わせれば、現実のほとんどの移動データを表現できると考え、この3種類を選択した。

図6 3D 管理構造

Fig. 6. 3D structure.

図7 XAT 構造 (Type 2) におけるオブジェクト用時間木

Fig. 7. Temporal tree for objects in XAT structure (Type 2).

表1 標準実験条件

Table 1. Parameters for experiments.

オブジェクト数	1000
全時空間範囲	[0,0,0]-[1000,1000,1000]
オブジェクトの移動範囲	全空間範囲の1%(面積)
オブジェクトの存在時間範囲	全時間範囲の20% (200 単位時間)*
線分化の細かさ	1 単位時間おき
検索条件	検索条件 A または検索条件 B のどちらか (検索条件 A) 空間:全空間範囲の10%(面積) 時間:全時間範囲の5% (検索条件 B) 空間:全空間範囲の5%(面積) 時間:全時間範囲の10%

本比較では、いくつかの実験を行うが、表1に標準とする実験条件を示す。この後の実験で実験条件の各種パラメータを変更する場合は、この値を中心として変更する。これらのパラメータのうち、オブジェクトの移動範囲は、そのオブジェクトの移動線分群のうち、最も左、右、上、下にあるものを含む長方形の面積としている。時間検索範囲と空間検索範囲には、2通りの組み合わせ(検索条件 A と検索条件 B)を標準値とする。実験には、Pentium4(1.5GHz)のマシンを使用し、プログラミング言語には C++ を使用

した。また、各葉ノードでは、最大10個のデータを保管可能とした。

〈4・4〉 実験方法 データ構造作成の観点から、XAT構造と3D管理構造を比較する。ここでは、標準実験条件において、データ構造を作成し、作成した中間ノード数と作成した葉ノード数を求める。

次に、データ構造検索の観点から、XAT構造と3D管理構造、XAT構造(Type 1)、XAT構造(Type 2)を比較する。ここでは、検索の際に探索した中間ノード数と葉ノード数の和(以下、探索ノード数)を計算する。実験条件のパラメータのうち、ここでは以下の2つに注目する。

(1) 時間検索範囲と空間検索範囲

(2) オブジェクトの移動範囲と存在時間範囲

(1)は、検索という利用面を考えた場合、最も頻繁に変更のありうるパラメータである。本評価では、これらを変更することで、XAT構造と3D管理構造を比較する。また、XAT構造(Type 1)、XAT構造(Type 2)とも比較し、2つの高速化の工夫が、検索時間の向上に貢献しているのか否かを検証する。(2)は、オブジェクトが持つ特性である。これらを変更して、一度オブジェクト単位で管理するXAT構造と、直接移動線分を管理する3D管理構造を比較し、XAT構造がいかなる条件でも優れた検索速度を実現するのか否かを検証する。

〈4・5〉 実験結果

〈4・5・1〉 データ構造作成 表2に、XAT構造及び3D管理構造の作成した中間ノード数と葉ノード数を示す。この表から、XAT構造より3D管理構造の方がこれらの数が少ないことが分かる。これは、XAT構造では時間木と空間木の両方を作成する必要があるためである。このことから、XAT構造はメモリコストが悪くなることが分かる。

〈4・5・2〉 データ構造検索

(1) 時間検索範囲と空間検索範囲

図8-(a),(b),(c)に、時間検索範囲を10%に固定し、空間検索範囲を1%-5%-10%-15%-20%-40%-60%と変化させたときの探索ノード数を示す。図8-(d),(e),(f)に、空間検索範囲を10%に固定し、時間検索範囲を1%-5%-10%-15%-20%-40%-60%と変化させたときの探索ノード数を示す。(a),(d)は直線運動、(b),(e)はランダム運動、(c),(f)は正弦波運動に対応する。なお、図8中の3D管理構造(Type 3)については、〈5・2〉節で説明する。

時間検索範囲(空間検索範囲)に対し、空間検索範囲(時間検索範囲)が大きくなるにつれて、XAT構造と3D管理構造の探索ノード数の差は大きくなる傾向がある。このことから、特に時間と空間の検索範囲の偏りが大きくなる時に、XAT構造は3D管理構造よりも優れた検索性能を発揮することが検証された。

また、XAT構造と高速化の工夫を行わないXAT構造(Type 1,2)を比較しても、XAT構造はいずれの方式よりも、探索ノード数が短いことが分かる。このことから、XAT構造における高速化の工夫が、実際に検索時間の短縮に貢

献していることが検証された。

(2) オブジェクトの移動範囲と存在時間範囲

図9-(a),(b)に、オブジェクトの移動範囲を、全空間範囲の0.1%-0.5%-1%-5%-10%-20%と変化させたときの探索ノード数を示す。図9-(c),(d)に、オブジェクトの存在時間範囲を、50-100-200-400-600時間単位(全時間範囲の、5%-10%-20%-40%-60%)と変化させたときの探索ノード数を示す。(a),(c)は時間検索範囲5%で空間検索範囲が10%、(b),(d)は時間検索範囲10%で空間検索範囲が5%である。ここでは、直線運動の結果のみ示しているが、ランダム運動や正弦波運動についても、ほぼ同様の結果が得られている。

図9-(a),(b)から、オブジェクトの移動範囲が広い場合は、XAT構造は3D管理構造に比べて、探索ノード数の悪化が急であることが分かる。これは、XAT構造が一度オブジェクト単位で絞り込みを行っているためである。全空間に対してオブジェクトの存在する割合が高くなると、検索範囲が同じでも、その範囲に入るオブジェクトの数が多くなる。すると、XAT構造では移動線分までチェックする必要のあるオブジェクトの数が増えるため、その探索ノード数が悪化するのである。

図9-(c),(d)から、オブジェクトの存在時間範囲が長い場合は、XAT構造と3D管理構造の探索ノード数の変化には大きな差は見られない。XAT構造では、全時間に対するオブジェクトの存在する割合が高くなると、移動線分までチェックする必要のあるオブジェクトの数が増えるため、探索ノード数が悪化する。それに対し、3D管理構造でも、オブジェクトの存在時間が長くなることで、1つのオブジェクト当たりの移動線分数が増えるため、探索ノード数が悪化する。XAT構造と3D管理構造の双方で、探索ノード数が悪化する要因があるため、探索ノード数の変化にも大きな差は見られなかったと言える。

これらのことから、オブジェクトの存在時間が長くなっても、XAT構造における検索時間の悪化の程度は、3D管理構造のそれと大きな差はないことが分かった。しかし、オブジェクトの移動範囲が大きくなった場合は、XAT構造は3D管理構造よりも検索時間が悪化することが分かった。

〈4・6〉 結 論 実験結果からXAT構造と3D管理構造の特性の違いをまとめると、表3ようになる。XAT構造のメモリ消費量は3D管理構造の2倍になるため、メモリ消費にシビアな応用では、3D管理構造を選択する必要がある。逆にメモリ環境に余裕のある応用では、オブジェクトの移動範囲が狭い場合には、3D管理構造よりもXAT構造の方が優れており、オブジェクトの移動範囲が大きい場合には、XAT構造よりも3D管理構造の方が優れていると言える。また、検索において時間検索範囲と空間検索範囲に差が生じることが頻繁にあることが想定される場合には、3D管理構造よりもXAT構造の方が良いと言える。このようにアプリケーションの特性に応じて、XAT構造と3D管理構造を使い分ける必要がある。

XAT構造の特性に適したアプリケーションとして、ある

表2 データ構造作成コスト

Table 2. Cost for data structure creation.

	3D 管理構造	XAT 構造	XAT 構造 (内訳)			
			オブジェクト管理部		線分管理部	
			時間木	空間木	時間木	空間木
中間ノード数	32767	62260	133	127	31000	31000
葉ノード数	32768	64271	143	128	32000	32000

表3 XAT 構造と 3D 管理構造の比較

Table 3. Comparison of XAT structure and 3D data structure.

	3D	XAT
メモリ量	○	×
狭い時空間範囲検索	○	○
時間/空間のどちらが広い検索	×	○
オブジェクトの移動範囲が広い	○	×
オブジェクトの移動範囲が狭い	×	○

(a) 直線運動

(d) 直線運動

(b) ランダム運動

(e) ランダム運動

(c) 正弦波運動

(f) 正弦波運動

図8 検索範囲に対する検索時間

Fig. 8. Search time for search range.

時点で終了したデータ群に対して分析を行うものが挙げられる。具体的な例としては、消防署や警察署、自衛隊、地域防災センターなどにおける、災害・事件発生時の救急活動状況の事後解析が考えられる。つまり、これらの組織において、救急車両や救急隊員の災害発生時の対応として、いつ

どこに誰がいたのかを、後々のために災害や事件が起こった後に、人手で解析するような場合である。この応用では、人間が解析するのに、何度も場所や時間を変えて検索しなおすことが想定される。したがって、データがメモリ上に乗る程度のデータ量であっても、何度も検索しなおすこと

(a) 時間検索範囲: 10%
空間検索範囲: 5%

(c) 時間検索範囲: 10%
空間検索範囲: 5%

(b) 時間検索範囲: 5%
空間検索範囲: 10%

(d) 時間検索範囲: 5%
空間検索範囲: 10%

図9 移動範囲/存在時間範囲に対する検索時間

Fig. 9. Search time for moving range and existing time.

を考えると、一回あたりの検索時間が短い方が良いと言える。その点で、XAT 構造は優れていると考えている。

5. 種々の検索要求と 3D 管理構造改良

〈5・1〉 種々の検索要求に対しての性能比較

〈5・1・1〉 目的 XAT 構造は、移動オブジェクトの範囲検索を目的として設計されているが、もう少し広い視点から見ると、以下の検索要求も時空間データ管理においては重要となってくる。

(1) 移動線分の範囲検索

(2) 移動線分の最近接検索

ここで、XAT 構造と 3D 管理構造の両検索要求に対する性質を見ることにする。ただし、最近接検索とは、時間検索範囲を固定した空間の最近接検索を指すこととする。そこで、XAT 構造と 3D 管理構造の両方を改良し、移動線分を範囲検索するもの (XAT 構造 (Type 3) と 3D 管理構造 (Type 1)) と移動線分を最近接検索するもの (XAT 構造 (Type 4) と 3D 管理構造 (Type 2)) を実装し、それぞれをシミュレーション実験により比較を行う。

〈5・1・2〉 比較対象手法の詳細 比較対象手法の詳細は次のようになる。

(1) XAT 構造 (Type 3)

オリジナルの XAT 構造では、2 段階目の移動線分の木

構造を検索する時には、一つでも解が見つければその時点で検索を終了し、解となる移動線分を持つオブジェクトを検索結果としていた。これに対し、XAT 構造 (Type 3) では、解を見つけても引き続き探索を行い、解となった移動線分を検索結果とする。

(2) 3D 管理構造 (Type 1)

オリジナルの 3D 管理構造では、移動線分検出後に、その移動線分を持つオブジェクトを検索結果としていたが、3D 管理構造 (Type 1) では、移動線分をそのまま検索結果とする。

(3) XAT 構造 (Type 4)

最初に範囲検索同様、木構造を根ノードから葉ノードへ探索する。葉ノードにたどり着いた時点で、検索キーの空間の一点が含まれる空間範囲は分かるが、最も近い移動線分を得るには、その空間範囲に隣接している範囲の移動線分もチェックする必要がある。そこで、到達した葉ノードから木構造を逆探索し、そこからさらに他のノードを探索する。このときの空間検索範囲は、その時点の最近接の移動線分の距離 d から一辺 $2d$ の長さの正方形を設定する⁽¹⁷⁾。

(4) 3D 管理構造 (Type 2) XAT 構造 (Type 4) と同様に、最近接検索を行う。

〈5・1・3〉 結果 検索要求 (1) に対しての実験は、標準実験条件において時間検索範囲と空間検索範囲を変化さ

せることで行った。図 10-(a) に、時間検索範囲を 10% に固定し、空間検索範囲を 1%-5%-10%-15%-20%-40%-60% と変化させたときの探索ノード数を示す。図 10-(b) に、空間検索範囲を 10% に固定し、時間検索範囲を 1%-5%-10%-15%-20%-40%-60% と変化させたときの探索ノード数を示す。ここでは、直線運動の結果のみ示しているが、ランダム運動や正弦波運動についても、ほぼ同様の結果を得ている。これらの図から、XAT 構造 (Type 3) と 3D 管理構造 (Type 1) では、ほぼ同じ探索ノード数を得ていることが分かる。これに対し、4.5.1 節で説明したオブジェクトの検索の時には、時間検索範囲と空間検索範囲のどちらかの範囲が大きい時に、XAT 構造は 3D 管理構造よりも性能が良かった。オブジェクトを検索の対象にしたときに比べて、移動線分を検索の対象としたときに探索ノード数の差が縮まったのは、XAT 構造のオブジェクトの絞り込みの効果がなく、検索範囲中のすべての移動線分を検索したからである。

検索要求 (2) に対しての実験は、標準実験条件において時間検索範囲を、1%, 5%, 10%, 20%, 40%, 60% と変動させて行った。ランダムで開始時刻と空間の中心点の組み合わせを 20 点決め、それらを検索条件として検索を行い、その平均探索ノード数を求めた。その結果を図 11 に示す。まず、3D 管理構造 (Type 2) に着目すると、時間検索範囲が大きくなるほど、探索ノード数が多くなっている。3D 管理構造は中間ノードでは、データを分割するのに、空間の X 軸、空間の Y 軸、時間軸 T の 3 つの軸を順に繰り返して用いている。そのため、時間検索範囲が大きい場合、最初の本構造の探索の時に多くのノードを探索してしまい、探索ノード数が多くなっている。

次に、XAT 構造 (Type 4) に着目すると、空間検索範囲が大きくなるほど、探索ノード数が少なくなっている。探索ノード数が増えない理由は、空間の最近接検索のために空間木を使っているためで、根ノードから探索していく時に、時間検索範囲は考慮していないためである。また、探索ノード数が減る理由は、時間検索範囲が大きくなるほど、葉ノードから木構造を逆探索した後に、そこからさらに探索する回数が少なくなるからである。詳しく説明すると、XAT 構造 (Type 4) は、葉ノードにたどり着いてから、そこにある移動線分の存在時刻を調べて、それが時間検索範囲に入っているか否かを確認している。そして、時間検索範囲に入る移動線分のうち、最も中心点の近くにある移動線分までの距離 d を用いて、逆探索してからの探索に用いる空間検索範囲を設定している。時間検索範囲が小さい場合、ある移動線分が中心点から最も近くにあったとしても、その移動線分の存在時間範囲が時間検索範囲に入っておらず、その次に近くにある移動線分をチェックする確率が高くなる。こうして空間検索範囲が大きくなり、逆探索した後に探索するノード数が多くなるのである。これらのことから、時間検索範囲が小さい場合は、3D 管理構造 (Type 2) の方が検索時間は短い、時間検索範囲が大きくなるほ

(a) 時間検索範囲 : 10%

(b) 空間検索範囲 : 10%

図 10 移動線分の範囲検索

Fig. 10. Range search of moving segments.

図 11 移動線分の最近接検索

Fig. 11. Nearest neighbor search of moving segments.

ど、XAT 構造 (Type 4) の検索時間の方が短くなることが分かる。

〈5.2〉 3D 管理構造の改良 XAT 構造は、AT 構造に対して木構造を二重化することで、移動オブジェクトの検索の高速化を狙ったものであるが、同様の工夫は 3D 管理構造に対しても可能である。ここでは、3D 管理構造を二重化 (オブジェクトを管理する 3D 管理構造と移動線分を管理する 3D 管理構造) して改良したもの (3D 管理構造 (Type 3)) を実装し、これと XAT 構造、オリジナルの 3D 管理構造と比較した。実験は標準実験条件において、時間検索範囲と空間検索範囲を変化させることで行った。その結果を図 8 に示す。3D 管理構造も木構造を二重化する

ことで高速化できることが分かる。3D 管理構造 (Type 3) は、空間検索範囲よりも時間検索範囲の方が大きい場合 (図 8-(d),(e),(f)) には、XAT 構造とほぼ同じ探索ノード数となっている。しかし、時間検索範囲よりも空間検索範囲の方が大きい場合 (図 8-(a),(b),(c)) には、XAT 構造の方が探索ノード数が少なくなっている。このことから、3D 管理構造も木構造を二重化することで、高速化することは可能であるが、それでもなお XAT 構造の方が空間検索範囲が広い場合には、検索時間が早いことが分かった。

6. まとめ

本稿では、移動オブジェクトを対象とした時空間データを高速に検索する手法を提案した。本手法では、オブジェクトを管理する時間木及び空間木と、移動線分を管理する時間木及び空間木を用意しておき、(1) 検索範囲に応じて時間木と空間木を適応的に切り替える、(2) 一度オブジェクト単位で検索結果の候補を絞り込んでから詳細な動きを検査する、ことを行っている。従来方式である 3D 管理方式とシミュレーション実験により比較した。その結果、提案する手法は、メモリコストがかかる欠点はあるが、時間検索範囲と空間検索範囲のどちらか一方が広い場合とオブジェクトの移動範囲が狭い場合は、3D 管理構造より優れていることが確認できた。今後は、逐次データが投入・削除されるような動的データを扱えるように、提案した手法を拡張していきたいと考えている。

(平成 13 年 11 月 28 日受付, 平成 14 年 11 月 20 日再受付)

文 献

- (1) K. Ikemoto, H. Nozawa, A. Naka, N. Saiwaki, and S. Nishida: "A Data Management Structure for Spatio-Temporal Walkthrough", *T. IEE Japan*, Vol.121-C, No.1, pp.142-149 (2001-1) (in Japanese)
池本和生・野澤 博・仲 篤起・才脇直樹・西田正吾:「時空間ウォークスルーのためのデータ管理の方式」, 電学論誌 C, 121, 1, pp.142-149 (2001-1)
- (2) M. Nabil, A.H.H. Nu, and J. Shepherd: "Modelling Moving Objects in Multimedia Databases", *Proc. of the 5th Int. Conf. on Database Syst. for Advanced Appl.*, pp.67-75 (1996)
- (3) M. Erwig, et al.: "Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases", *GeoInformatica*, Vol.3, No.3, pp.269-295 (1996)
- (4) M.A. Nascimento, J.R.O. Silva, and Y. Theodoridis: "Evaluation of Access Structures for Discretely Moving Points", *Proc. of the Intl. Workshop on Spatiotemporal Database Management (STDBM'99)*, pp.171-188 (1999)
- (5) N. Ukai and Y. Masunaga: "A Moving Object Model for the Realization of Spatio-temporal Databases", *IPSJ SIG-Notes DataBase Syst.*, 99-DBS-119, pp.153-158 (1999) (in Japanese)
鶴飼規子・増永良文:「時空間データベース構築のためのムービングオブジェクトモデル」, 情報処理学会研究報告, 99-DBS-119, pp.153-158 (1999)
- (6) J.L. Bentley: "Multi dimensional Binary Search Trees Used for Associative Searching", *Comm. of the ACM*, Vol.18, pp.509-517 (1975)
- (7) A. Guttman: "R-Trees: A Dynamic Index Structure for Spatial Searching", *Proc. of ACM SIGMOD Intl. Symp. on the Management of Data*, pp.45-57 (1984)

- (8) Y. Ohsawa and M. Sakauchi: "On a Multidimensional Data Management Structure with Efficient Dynamic Characteristics", *T. IEICE*, Vol.J66-D, No.10, pp.1193-1200 (1983-10) (in Japanese)
大沢 裕・坂内正夫:「良好な動特性を持つ多次元点データ管理構造の提案」, 電子通信学会論誌 D, J66-D, 10, pp.1193-1200 (1983-10)
- (9) Y. Nakamura, S. Abe, Y. Ohsawa, and M. Sakauchi: "MD-Tree: A Balanced Hierarchical Structure for Multidimensional Data", *T. IEICE*, Vol.J71-D, No.9, pp.1745-1752 (1988-9) (in Japanese)
中村泰明・阿部 茂・大沢 裕・坂内正夫:「多次元データの平衡木による管理—MD木の提案」, 電子通信学会論誌 D, J71-D, 9, pp.1745-1752 (1988-9)
- (10) M.A. Nascimento, J.R.O. Silva, and Y. Theodoridis: "Evaluation of Access Structure for Discretely Moving Points", *Spatio-temporal Database Management 1999*, pp.171-188 (1999)
- (11) T. Teraoka, M. Maruyama, Y. Nakamura, and S. Nishida: "The Multidimensional Persistent Tree: A Spatio-Temporal Data Management Structure Suitable for Spatial Search", *T. IEICE*, Vol.J78-D-II, No.9, pp.1346-1355 (1995-9) (in Japanese)
寺岡昭彦・丸山 稔・中村泰明・西田正吾:「空間検索を効率化した時空間データ管理構造の提案—多次元 Persistent Tree—」, 電子情報通信学会論誌 D-II, K78-D-II, 9, pp.1346-1355 (1995-9)
- (12) M.A. Nascimento and J.R.O. Silva and Y. Theodoridis: "Access Structures for Moving Points", *Technical Report 33*, Time Center, (1998)
- (13) T. Tzouramanis, M. Vassilakopoulos, and Y. Manolopoulos: "Overlapping Linear Quadrees and Spatio-Temporal Query Processing", *The Computer Journal*, pp.325-343 (2000)
- (14) X. Xu, J. Han and W. Lu.: "RT-tree: An Improved R-tree Index Structure for Spatiotemporal Databases", *Proc. of the 4th Intl. Symposium on Spatial Data Handling*, pp.1040-1049 (1990)
- (15) Y. Theodoridis, M. Vazirgiannis, and T. Sellis.: "Spatio-temporal Indexing for Large Multimedia Applications", *Proc. of the 3rd IEEE Conf. on Multimedia Computing and Systems*, pp.441-448 (1996)
- (16) Y. Ohsawa and Y. Nakamura: "Spatial Data Structures for Efficient and Fast Spatial Retrieval", *IPSJ Magazine*, Vol.42, No.10, pp.965-971 (2001-10) (in Japanese)
大沢 裕・中村泰明:「空間データの効率的な管理と高速空間検索のためのデータ構造」, 情報処理学会学誌, 42, 10, pp.965-971 (2001-10)
- (17) M. Sakauchi and Y. Ohsawa: "Image Database", *Shokodo*, (1987) (in Japanese)
坂内正夫・大沢 裕:「画像データベース」, 昭晃堂, (1987)

王 軼 群 (学生員) 平成 13 年 3 月, 大阪大学基礎工学部システム科学科卒業。平成 15 年 3 月, 同大学大学院修士課程修了。現在, 同大学大学院基礎工学研究科システム創成専攻博士課程在籍。データベースの研究に従事。

野 澤 博 (非会員) 平成 10 年 3 月, 大阪大学基礎工学部システム工学科卒業。平成 12 年 3 月, 同大学大学院修士課程修了。データベースの研究に従事。平成 12 年 4 月, (株) 日本総合研究所入社。現在に至る。

土方 嘉徳 (非会員) 平成8年3月, 大阪大学基礎工学部シ

ステム工学科卒業。平成10年3月, 同大学大学院修士課程終了。同年4月, 日本アイ・ビー・エム(株)東京基礎研究所入所。平成14年4月, 大阪大学大学院基礎工学研究科システム人間系専攻リサーチアソシエイト。同年10月, 同助手。現在, 大阪大学大学院基礎工学研究科システム創成専攻助手。博士(工学)。平成15年2月, 計測自動制御学会学術奨励賞受賞。インターネット・アプリケーション技術, データベース, ヒューマンインタフェースの研究に従事。情報処理学会, 電子情報通信学会, 計測自動制御学会, 日本ソフトウェア科学会, ヒューマンインタフェース学会, 日本データベース学会, IEEE各会員。

仲谷 美江 (非会員) 昭和58年3月大阪大学人間科学部卒

業。同年4月三菱電機(株)中央研究所入社。マンマシンインタフェース, グループウェア, 高齢者支援の研究に従事。1999年非営利団体に移り, 高齢者マンション設立に関わる。平成12年より大阪大学リサーチアソシエイト。工学博士。平成11年情報処理学会研究賞受賞。電子情報通信学会, 情報処理学会, 日本認知科学会, 在宅ケア学会な

どの会員。

西田 正吾 (正員) 昭和27年1月5日生。昭和49年3月東

京大学工学部電子工学科卒業。昭和51年3月同大学大学院修士課程修了。同年4月三菱電機(株)入社。同社中央研究所システム基礎研究部研究員, グループマネージャーを経て, 平成7年4月, 大阪大学基礎工学部システム工学科教授。現在, 大阪大学大学院基礎工学研究科システム人間系専攻教授。システム技術, ヒューマンインタフェース

技術, メディア技術の研究に従事。工学博士。昭和59-60年MITメディアラボ客員研究員。61年度, 平成5年度電気学会論文賞, 平成4年度電気学会著作賞, 平成7年度電気学会進歩賞, 平成13年度ヒューマンインタフェース学会論文賞受賞。電気学会, 情報処理学会, IEEEほか会員。IEEE Fellow。著書は, 「ヒューマン・コンピュータ交流技術」(オーム社, 共著), 「メディア工学」(朝倉書店), 「情報メディア工学」(オーム社, 共著)など。