空間情報のインクリメンタルな視覚化のための検索手法とセンサデータベースへの応用

平成15年度

白石 陽

目 次

図 目 次

表 目 次

第1章

序論

1.1 はじめに

モバイルコンピューティングやインターネットの利用形態の多様化により，携帯電話を用いた周辺のショッピングや飲食店に関する情報提供サービス，PHS の位置検出機構や GPS を利用したナビゲーションシステム，インターネットを介した地図提供システム，地図に基づくコンテンツ提供システムなど，多くの位置情報システムが出現している．本論文では，位置に関連した様々な情報を管理し，それらの情報を位置に基づいて処理した結果をユーザに提供するシステムのことを “位置情報システム” と呼ぶ．位置情報システムでは，たとえば，ユーザの現在地や地図上の地点を，住所，駅名，経緯度座標で指定することにより，その位置に関連した様々な情報を取得することができ，空間的な領域を指定することにより，指定した領域の地図やその領域に関連した情報を取得することができる．これらの地理的な位置に関連付けられる情報のことを “空間情報” と呼ぶ．

空間情報の中で，位置情報システムが様々な情報を統合し，活用する上で，デジタル化された地図情報 (地図データ) は非常に有用である．地図データの具体例としては，行政界データ，道路データ，鉄道データなどが挙げられる．地理情報システム (Geographic Information System: GIS) の分野では，これらの地図データに加えて，標高データ，土地利用データ，地価データなども扱われる．これらのデータに共通しているのは，幾何的な形状を表す情報と属性情報から構成されることであり，このような空間情報を，本論文では “空間データ” と呼ぶ．さらに，国内外で空間データの相互利用や標準化を目指した活動が展開されている [明野 他 97a, 明野 他 97b, NSDIPA, DPC, OGC, TC211]．たとえば，G-XML プロジェクト [DPC] は，GIS コンテンツの相互流通の実現を目指した活動を行っており，GIS データを記述するための XML 仕様を公開している．この仕様に基づき，GIS データを XML 形式のテキストデータとして記述することができる．また，国土地理院 [GSI] からは情報統合の基盤となる地図として「数値地図 2500(空間データ基盤)」が発行されている．今後，ネットワークを介して，多種多様な空間データが配信されるようになれば，

より便利な位置情報システムやサービスが実現されていくと考える．本論文では，ネットワーク上の空間データを管理するサーバから，多種多様な空間データを取得し，利用できる環境を想定する．

一方，近年，気象情報や交通情報のような環境の状態を表すセンサデータをオンラインで公開するシステムやサービス [国土交通省 b, 日本気象協会, 環境省, IWATE] が多数出現しており，今後，さらに多種多様なセンサデータをネットワークを介して利用できる環境が整備されていくと考える．これらのサービスでは，地理的に分散したセンサから時々刻々と提供されるセンサデータを時系列としてデータベースに蓄積しているため，最新のデータだけでなく，過去のデータを閲覧することが可能である．本論文では，このような時系列のセンサデータを保持するデータベースのことを，“センサデータベース” と呼び，アプリケーションが，ネットワーク上の複数のセンサデータベースから異種の時系列のセンサデータを取得し，利用できる環境を想定する．ロボット工学，計測工学，制御工学の分野では，センサデータと言った場合，頻繁に更新されるリアルタイムのデータを指すことが多いが，本研究では，データベース中に時系列として蓄積されたセンサデータを対象とする．

センサデータは，ユーザの生活している環境の状態を表しているため，ネットワークを介して多種多様なセンサデータを利用できれば，モバイル情報システムや高度交通システムなどの位置情報システムにとって有用な情報となる．たとえば，天気予報や渋滞情報，観光地の混雑状況などを複合的に利用したナビゲーションシステム，気象情報や大気の状態に基づく生活環境のモニタリングを行うシステムなどの応用が考えられる．このようなセンサデータを利用するシステムにとっては，最新のデータや過去のある時刻のデータだけでなく，指定した時区間の時系列データも有用であり，センサデータが時間属性を持つという点を考慮することが重要である．さらに，センサデータは，地理的な位置に関連付けられるため，“空間情報” の一種と考えることができる．位置情報システムでの利用という観点からは，センサデータの「空間情報」としての側面も重要である．本論文では，センサデータを組み合わせて，あるいは，他の情報と関連付けて処理を行うことを “センサデータの統合” と呼ぶが，地理的な位置情報に基づいて，収集したセンサデータを地図データや他のセンサデータと統合することができれば，より効果的な情報提供が実現できると考える．

1.2　本研究の目的

本論文では，次の 2 種類の空間情報を取り上げる．

- 空間データ：
  幾何的な形状情報と属性情報から構成される．
- センサデータ：

一般に，センサデータは，センサの出力値と更新時刻から構成されるが，本論文では，センサの設置されている位置についての情報も含むものとする．

ネットワーク上の空間データサーバに対して，クライアントがデータを要求する場合，検索結果を受信するまでの応答時間が問題となる．空間データは，形状や属性など多種多様な情報から構成されるが，空間データサーバは大量の空間データを管理しているため，サーバ側でのデータ探索コストの増大に伴い，応答時間が増大する可能性がある．同様に，ネットワーク上の複数のセンサデータベースからセンサデータを収集し，統合処理を行う場合にも，サーバ側のデータ探索にかかるコストやクライアント側のデータ統合処理の計算コストなどが増大し，統合結果を提示するまでの応答時間が増大することが予想される．結果として，ユーザの待ち時間が増大し，空間データやセンサデータなどの空間情報をインタラクティブに閲覧する上での問題となる．

人工知能の分野では，実時間問題解決の枠組みとして，“anytime algorithm” が提案されている [Dean *et al.* 88, Boddy *et al.* 89, Boddy *et al.* 94]．Anytime algorithm は，問題解決の処理品質が単調に増加する，すなわち処理結果を逐次的に提供できるという性質を持つ．本研究では，anytime algorithm の持つ逐次性に着目し，空間データやセンサデータなどの空間情報を視覚化する際の応答時間の増大という問題に対して，検索および統合の途中結果をインクリメンタルに提供するアプローチを取る．最終的な結果を待つことなく，ユーザの理解しやすい形で，検索や統合処理の途中結果を提示できれば，処理途中でのインタラクティブな操作やデータ要求が可能となり，応答性の良い視覚化システムを実現できると考える．

本研究の目的は，多種多様な空間情報をネットワークを介してインクリメンタルに視覚化するための検索手法を開発することである．具体的には，次のそれぞれの問題について，検索結果である空間情報をインクリメンタルに提供する手法を開発することを目的とする．

(A) ネットワークを介した空間データの検索
ネットワーク上の空間データを管理するサーバに対して，データを要求し，検索結果として空間データを取得する問題を扱う．

(B) ネットワーク上のセンサデータベースに対する検索
ネットワーク上の複数のセンサデータベースから時系列のセンサデータを収集するための検索手法および収集したセンサデータを統合する手法を開発する．

ネットワーク上のデータベースからの空間情報の取得の方法，およびネットワーク上に分散した複数のデータベースから収集した空間情報の統合のための手法を開発することで，ネットワークを介した空間情報の視覚化のための枠組みが構築できると考える．

1.3　本研究のアプローチ

空間情報のインクリメンタルな視覚化を実現するためには，(A) 空間データ検索，(B) センサデータベース検索のそれぞれについて，次の点を考慮する必要がある．

ネットワークを介した空間データの検索

ネットワークを介した空間データの検索は，(1) クライアントから空間データを管理するサーバへのクエリの送信，(2) サーバでのデータ探索処理，(3) サーバからクライアントへの探索結果の送信という手順から成る．クライアントが空間データをインクリメンタルに表示するためには，空間データサーバ内で保持されている探索の中間結果を，外部のクライアントに検索の途中結果として提供することが必要となる．

また，検索の途中結果をユーザの理解しやすい意味のあるまとまりとして提供するためには，検索結果をどのように分割し，どのような順序で提供するかが重要である．すなわち，空間データサーバが，どのような順序でデータ探索を行い，その結果を送信するかが問題となる．

さらに，一般に，空間データ探索のコストは大きいため，探索処理の途中で，ユーザに対して何らかの情報を提供するためには，全探索の結果を分割し，順序付けるのではなく，探索過程そのものを制御する必要がある．

ネットワーク上のセンサデータベースに対する検索

ネットワーク上に分散した複数のセンサデータベースからセンサデータを収集する場合には，まず，アプリケーションの要求するセンサデータを提供できるセンサデータベースを探し出す必要がある．収集したセンサデータを “空間情報” として統合するためには，センサ (つまり，センサデータ) の地理的な分布範囲を考慮して，検索システムが地理的な位置や範囲を利用した検索 (位置指向検索) をサポートする必要がある．

複数のデータベースから収集したセンサデータを統合する場合には，位置情報を利用して，異種のセンサデータ同士あるいはセンサデータと空間データとを関連付けること，すなわち，センサデータの “空間情報” の側面に注目した統合処理方法が有用である．たとえば，地理的に分布した大量のセンサデータを閲覧する際には，個々のセンサデータの値を見るよりも，位置情報に基づき，複数の関連するセンサデータを加工し，抽象度の高い情報として提供することで，全体的な傾向が把握しやすくなると考える．さらに，ユーザの注目する場所の環境の状態を把握するためには，指定した単種類のセンサデータを視覚化するだけでなく，複数の異なる種類のセンサデータを統合し，その結果を表示することが効果的であると考える．

これらの条件を満たすことで，ネットワーク上の複数のセンサデータベースに蓄積されたセンサデータを，“空間情報” として検索し，統合する枠組みが構築できると考える．その上で，センサデータをインクリメンタルに視覚化するためには，要素技術として，センサデータベース検索の結果をインクリメンタルに提供する手法と，その検索結果をインクリメンタルに統合する手法を開発する必要がある．

以上の点を踏まえて，本論文では，具体的に次の手法を提案する．

(i) 領域分割に基づく空間データ検索手法 (3 章)

(ii) センサデータベース検索のためのインクリメンタルなデータ提供手法 (4 章)

(iii) 空間集約に基づくセンサデータ視覚化手法 (5 章)

(iv) インクリメンタルなセンサデータ統合手法 (6 章)

領域分割に基づく空間データ検索手法

まず，ネットワークを介した空間データの検索手法として，領域分割に基づく手法を提案する．本検索手法は，クエリに指定した問合せ領域を分割して，空間インデックスを用いて管理されている空間データを分割領域ごとに探索し，その結果をインクリメンタルに提供する．これにより，クライアントは “領域” という意味のある単位で検索の途中結果を受信することができる．さらに，ユーザが領域分割のためのパラメータを指定することにより，ユーザの関心のある領域の情報を優先的に取得し，閲覧することが可能となる．この領域分割に基づくアプローチは，本研究における空間情報の視覚化のための基本となる考え方である．

センサデータベース検索のためのインクリメンタルなデータ提供手法

次に，この領域分割に基づくアプローチを，センサデータベースのための検索手法に適用する．提案手法は，仲介エージェントと各データベースのラッパーとなるセンサデータサーバから構成され，各サーバの管理するセンサデータベース中のセンサデータを “空間情報” として扱うために位置指向検索をサポートする．各データサーバは，センサデータを時系列として保持するだけでなく，各センサの位置情報も管理する．センサデータサーバは，検索結果として，時系列データと位置情報を返すことができ，クライアントは，検索結果に含まれている位置情報を用いて様々な統合処理を行うことが可能となる．

領域分割に基づくインクリメンタルなデータ提供は，位置情報に基づくセンサデータ統合を行う場合に有用であると考えられる．一方で，時間情報を考慮していないため，時系列データの集約などの時系列データ処理やセンサデータの時間変化の閲覧に対応する枠組みとしては不十分である．そこで，位置情報や領域だけでな

く，時間情報を考慮したインクリメンタルなデータ提供が効果的であると考え，領域分割に加えて，時区間分割を導入する．

仲介エージェントは，クエリに指定した領域と時区間を分割し，その順序に従って，各データサーバへのクエリの送信と各サーバからの検索結果の転送を制御する．これにより，各サーバからのセンサデータを，時区間ごと，領域ごとに同期させながらインクリメンタルに提供することが可能になる．

空間集約に基づくセンサデータの視覚化手法

さらに，複数のセンサデータベースに対する検索結果であるセンサデータを視覚化する仕組みとして，空間集約処理に基づく手法を提案する．空間集約は，地理的な関係性を利用して空間データを加工する手法であり，地理情報システム (GIS) の様々な処理に用いられている．空間集約を利用することで，ネットワークを介して取得した種類の異なるセンサデータに対して，単なる表示レベルの重ね合わせではなく，データレベルの統合，さらには意味的な統合を行うための枠組みが提供できると考える．本論文では，GIS の分野で提案されている空間データ統合，空間補間，オーバーレイ処理といった基本的な空間集約処理に基づいて統合処理を行う視覚化システムを構築する．

そして，空間集約に基づくセンサデータ視覚化システムにおいて，ユーザの移動履歴に基づくセンサデータの閲覧の機能を導入することで，閲覧性の向上を図る．ユーザの移動空間や生活空間を如実に表す移動履歴を利用することで，センサデータベースに対する直接的なデータ要求が可能となる．移動履歴に基づくデータ要求と空間集約に基づく検索結果の加工によって，効果的なセンサデータ視覚化システムが実現できる．

インクリメンタルなセンサデータ統合手法

最後に，複数のセンサデータベースから収集したセンサデータをインクリメンタルに統合する方法として，空間集約に基づく手法を提案する．地理情報システムで利用されている空間集約のアルゴリズムは，インクリメンタルに提供されるセンサデータを処理する枠組みとはなっていないため，改良が必要となる．具体的には，空間データ統合，空間補間，オーバーレイ処理のそれぞれに対して，インクリメンタルにセンサデータを統合する手法を考案する．提案手法は，領域分割に基づいて提供されるセンサデータに対して，分割領域ごとに空間集約処理を行い，その集約結果をインクリメンタルに表示することができる．

1.4 本論文の構成

以下，2章で，関連研究として，空間情報の視覚化のための要素技術，センサデータの検索と統合のための要素技術，および，処理結果をインクリメンタルに提供する研究について述べる．3章では，インクリメンタルに検索結果を提供するための領域分割に基づく空間データ検索手法を提案する．4章では，3章で提案した領域分割に基づく手法をセンサデータベース検索に適用し，インクリメンタルなセンサデータ提供方式を提案する．5章で，空間集約に基づくセンサデータの視覚化について述べ，その有用性について議論したのち，6章で，センサデータベースから取得したセンサデータをインクリメンタルに統合する手法について述べる．7章で，本論文での提案手法についての議論を行い，今後の課題を述べる．最後に，8章で，本研究の結論を述べる．

第2章

関連研究

本章では，まず，空間情報を視覚化するための要素技術として，空間データベースや地理情報システムの分野における空間データの管理，検索，統合手法について述べる．次に，本研究で想定している異種センサネットワーク環境について述べ，センサデータベースのための検索と統合に関連する研究事例を挙げる．最後に，本研究の特徴であるインクリメンタルなアプローチとして，“anytime algorithm” と，その関連研究について述べる．

2.1 空間情報処理技術

2.1.1 位置情報に基づく統合

パーソナルナビゲーションシステム，Intelligent Transport System (ITS)，オンライン地図サービスなど様々な位置情報システムに関する研究開発が行われている．これらの位置情報システムに共通するのは，地理的な位置情報に基づき様々な情報を統合している点である．

位置情報に基づく統合に関する研究としては，Web 上の様々なデータリソースのデータを位置関連情報に基づき収集し，検索するための手法 [横路 他 97] が提案されており，Web 上の試験サービスとして運用されていた．このサービスでは，住所などの位置情報を入力することでその場所に関連した情報を掲載しているホームページを検索できる．さらに，Web を用いた地理情報システムに基づくコンテンツ提供システム [Kyoto, 平松 他 00a, 平松 他 00b, 相良 他 00] も提案されており，経緯度情報に基づき様々なコンテンツを地図上に表示することができる．

2.1.2 情報統合のための空間データ

地図を描画するためのデジタル化された地図情報 (地図データ) や地理情報システム (GIS) で利用されるデータ (GIS データ) は，行政区域，河川，鉄道，道路などを示

す図形データ (幾何データ) と，名称など各図形に関連した情報によって記述される．地理情報システムの分野では，地図データや GIS データなど，幾何データと属性データによって表現されるデータのことを “空間データ” と呼ぶが，幾何データのことを “フィーチャ”，非幾何属性のことを単に “属性” と呼ぶことも多い[1]．空間データを記述するための基本的な幾何データは，次の 3 つである [野上 他 01, 岸野 他 00, 張 01]．

- 点データ (point data)
  座標によって表現される幾何学的な位置．
- 線データ (line data)
  点の順序系列によって表現される線分 (line segment) やアーク (arc) など．
- 面データ (area data)
  線データによって表現されるポリゴン (polygon) など．

このような点データを基本要素とした空間データモデルのことをベクタモデルと呼ぶ．

位置情報システムが，他の関連情報を統合利用し，より便利なサービスを実現するためには，地図データなどの空間データは必要不可欠な情報である．しかし，様々なベンダによって提供される地図データは独自の規格に沿って記述されているため互換性がなく，相互に利用することが難しい．それに対し，地理情報システム (GIS) や空間データベースの分野では，地図データや GIS データの相互運用や標準化を目指した動きが国内外で見られる [明野 他 97a, 明野 他 97b, NSDIPA, DPC, OGC, TC211]．

日本では，省庁が中心となった国土空間データ基盤推進協議会 [NSDIPA] が存在し，国土地理院 [GSI] からは「数値地図 2500(空間データ基盤)」と呼ばれる地図データが CD-ROM として配布されている [JMC]．数値地図 2500 は，2500 分の 1 の縮尺の基盤データであり，様々な地図情報がテキストデータとして表現されている．たとえば，街区，道路，鉄道などを示す地図データが，アークやポリゴンの集合として表現されている．また，国土数値情報ダウンロードサービス [国土交通省 a] という Web サイトでは，国土に関わる様々な空間データをダウンロードして利用できる．国土数値情報は，点データ，線データ，面データ，メッシュデータ，表データの 5 つの形式に分類されている．たとえば，面データとしては行政界・海岸線データ，線データとしては鉄道データや道路データ，点データとしては地価公示点データや公共施設データ，メッシュデータとしては自然地形データや土地利用データなどが挙げられる．

さらに，G-XML プロジェクト [DPC] は，GIS コンテンツの相互流通の実現を目指した活動を行っており，GIS データを記述するための XML 仕様が公開されている．この仕様に基づき，GIS データを XML 形式のテキストデータとして記述する

[1]空間データベースの分野では，空間データとして，主に幾何データを扱っている．

ことができ，点，線，ポリゴンなどの基本図形を表現することができる．これらのテキストとして表現されたベクトル形式の空間データは，情報統合を行う際の基盤となるデータである．

以下，本論文では，幾何データとしての空間データに着目し，点 (point)，線 (line)，多角形 (polygon) などの各幾何データ要素を空間データオブジェクトと呼ぶ．空間データは，空間データオブジェクトの集合であり，空間データベースシステムにより管理される空間データオブジェクトの集合や検索結果に含まれる空間データオブジェクトの集合を指す．

2.1.3 空間データの検索と管理

空間問合せ

空間データを検索するための問合せを，空間問合せ (spatial query) と呼ぶ．代表的な空間問合せを次に示す．

- 点位置決定の問合せ (point location query)
  点位置決定問題 [浅野 他 00] とも呼ばれ，指定した点と重なる空間データオブジェクトを探し出す問合せである．

- 範囲問合せ (range query)
  指定した範囲に含まれる空間データオブジェクトを探し出す問合せである．範囲としては，円や矩形などが指定されるが，矩形による範囲問合せは window query と呼ばれる．

- 近傍問合せ (nearest neighbor query)
  指定した点に最も近い空間データオブジェクトを探し出す問合せである．

- k 近傍問合せ (k-th nearest neighbor query)
  指定した点に最も近い k 個の空間データオブジェクトを探し出す問合せである．近傍問合せとともに，2 次元の地図データに限らず，多次元データに対して利用されることが多い．

地理情報システムでは，地図表示や空間データ分析のために，ある領域内に存在する空間データを探し出す範囲問合せがよく利用される．本論文においても，ある領域内の異種のセンサデータと空間データを要求し，統合するために，範囲問合せを対象とする．

空間インデックス

幾何属性を持つ空間データを効率的に管理し，検索するための索引構造を空間インデックスと呼び，これまで様々な空間インデックスが提案されている [岸野 他 00, Rigaux *et al.* 01, 浅野 他 00, Gaede *et al.* 98]．代表的な空間インデックスとして，R-tree[Guttman 84] が挙げられるが，空間データベースの分野では，R-tree を利用した研究や拡張に関する研究が多数行われている [Beckmann *et al.* 90, Sellis *et al.* 87, Roussopoulos *et al.* 95, Papadopoulos *et al.* 97]．R-tree は，B-tree のような平衡木であり，すべての葉ノード (leaf node) は，同じ高さに存在する (図 2.1)．

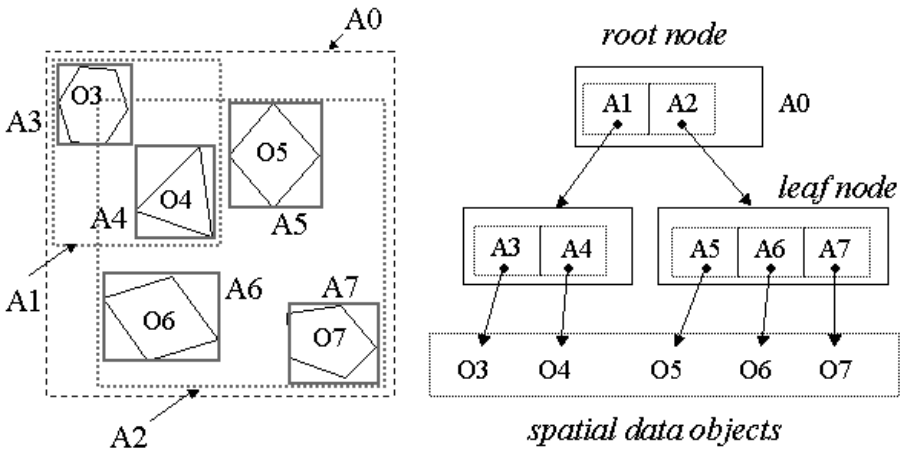

図 2.1: R-tree の構造

R-tree の各ノードは，複数のエントリを持つ．葉でないノードのエントリは $< rect, child\text{-}pointer >$，葉ノードのエントリは $< rect, record\text{-}pointer >$ と表現できる．$rect$ は最小包囲矩形 (Minimum Bounding Rectangle：MBR) を表す．葉ノードのエントリを e，その MBR を $e.rect$ とする時，$e.rect$ は，e が指し示すレコード中の空間データオブジェクト o を囲む最小の矩形領域として計算される (図 2.2)．すなわち，空間データオブジェクト o の幾何形状を，$o.geom$ とする時，$o.geom \subseteq e.rect$ が成り立つ．

各ノードの MBR は，そのノードのすべてのエントリ (すなわち，子ノード) の MBR を覆う矩形領域として計算される (図 2.3)．

R-tree は一般に n 次元の空間データを扱うことができるが，本論文のように地図データなどの 2 次元の空間データを対象とする場合，MBR は，2 点の座標 (x_1, y_1)，(x_2, y_2) によって定まる矩形として表現される．

図 2.1 において，A3, A4, A5, A6, A7 はそれぞれ，空間データオブジェクト O3, O4, O5, O6, O7 に対する包囲矩形 (MBR) を表し，葉ノードのエントリの属性となる．A1 は，A3, A4 を覆う MBR であり，A2 は，A5, A6, A7 に対する MBR で

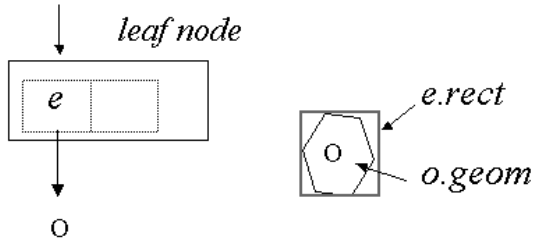

図 2.2: エントリの包囲矩形とオブジェクトの幾何形状の関係

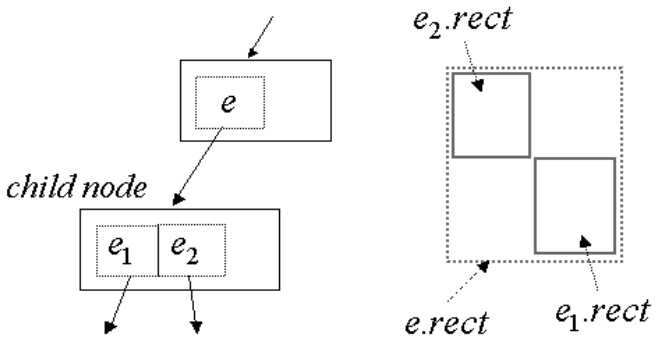

図 2.3: R-tree における最小包囲矩形 (MBR) の定義

ある．A0 は，A1 および A2 を覆う MBR であり，図 2.1 においては，根ノードの属性となる．このように，MBR を階層的に定義していくことにより，図 2.1 の右図に示すような木構造のインデックスが構築される．

R-tree は，動的な平衡木であり，B-tree と同様に，データオブジェクトの追加 (insertion)，削除 (deletion) に対して，葉ノードを同じ高さに保つことができる．平衡な木構造を保つために，R-tree では，各ノードのエントリ数について制限を加える 2 つのパラメータ (m, M) が存在する．M はノードが含むことのできるエントリの最大数，m はノードが含まなければならない最低限のエントリ数を表し，$m \leq M/2$ の関係を満たさなければならない．オブジェクトを追加する際には，木構造インデックス中の各ノードの MBR とオブジェクトとの包含関係を調べることにより，そのオブジェクトと重なり合う MBR を属性として持つ葉ノードが選択される．この時，エントリの最大数 M を越える場合には，$M+1$ のエントリが 2 つのグループに分割され，新たにノードが生成される．新しく生成されたノードを親ノードのエントリとして追加する際，親ノードのエントリ数によっては，その親ノードも分割されることになる．この分割は，葉ノードから根ノードに向けて伝播していき，結果的に，索引木の分割 (splitting) が行われることになる．

R-tree では，空間範囲問合せに対して，MBR に基づく探索が行われる．問合せ範囲 R_{query} と重なり合う空間データオブジェクトを探し出す場合には，木構造インデックス中のノードが属性として持つ包囲矩形 (MBR) との包含関係を調べる．もし重なり合う場合には，そのノードのエントリが指し示す子ノードの MBR との包含関係を調べ，その処理を繰り返すことで探索を進めていく (図 2.4)．

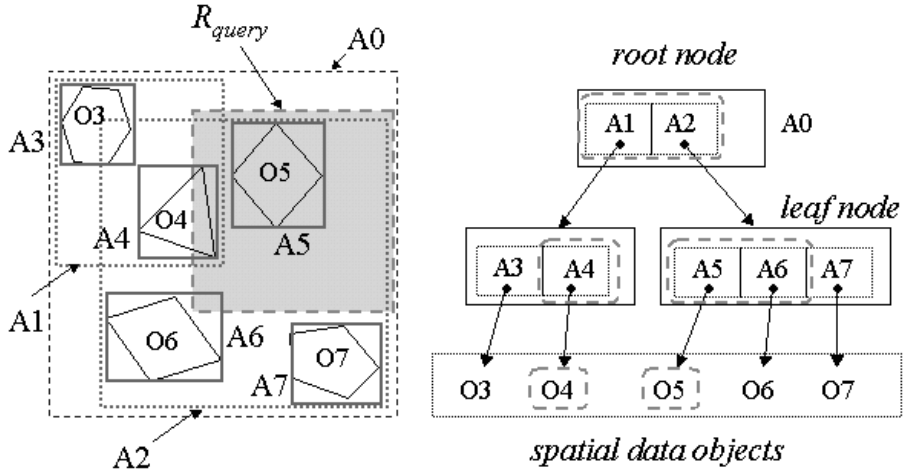

図 2.4: R-tree における探索処理

また，空間インデックスに基づく探索では “filter&refinement” 戦略 [Orenstein 89, Brinkoff *et al.* 93] が，よく利用されている．“filter&refinement” 戦略では，空間デー

タオブジェクト (o) が問合せ領域に重なるかどうかを比較する際に，まず，そのオブジェクトの MBR($e.rect$) と比較する．そして，問合せ領域と MBR が重なり合う場合に，そのオブジェクトの幾何形状 ($o.geom$) の比較を行うことで，そのオブジェクトへのアクセスを行う (図 2.2)．オブジェクトの幾何形状の参照はディスクアクセスを伴うが，MBR との比較によってアクセスすべき空間データオブジェクトを絞り込むことができるので，アクセスコストを減少させることができるという点で効果的である．

図 2.4 の例では，まず，根ノードのエントリの MBR(A1，A2) と問合せ領域 R_{query} との包含関係を調べる．A1，A2 ともに，R_{query} と重なり合うので，それらのエントリが指し示す子ノードのエントリについて調べる．葉ノードのエントリの MBR の中では，A4，A5，A6 が R_{query} と重なり合い，O4，O5，O6 が探索結果の候補となる．最終的に，オブジェクトの幾何形状との比較を行うことによって，O6 は除外され，O4，O5 が探索結果となる．

2.1.4 Spatial join

空間データベースの分野では，空間結合 (spatial join) に関する研究が多数行われている [Rigaux *et al.* 01, Huang *et al.* 97, Lo *et al.* 94, Patel *et al.* 96, Lo *et al.* 96, Koudas *et al.* 97]．一般に spatial join は，異なる空間データオブジェクトの集合に含まれる要素を，交差や包含などの空間的な関係 (spatial relation) に基づいて関連付ける技術である．すなわち，空間データオブジェクトの集合 R，S の要素 $r \in R$，$s \in S$ について，r と s の空間的な関係を調べる．たとえば，R を道路の形状データ，S を河川の形状データとする場合には，r，s はともに，線データオブジェクトとなり，r と s の交差関係を調べることにより，道路と河川が交わる場所を抽出することができる．

Spatial join では，2 つの空間データオブジェクトの集合要素の総当たり的な比較を避けるために，まず，何らかの手段を用いて比較対象となる空間データオブジェクトを絞り込むための処理（filter 処理）が行われ，候補として残った空間データオブジェクトについて空間的な関係が調べられる．この filter 処理の手法として，空間インデックスに基づく手法 [Huang *et al.* 97, Lo *et al.* 94] や区画分割に基づく手法 [Patel *et al.* 96, Lo *et al.* 96, Koudas *et al.* 97] などが提案されている．空間インデックスに基づく手法では，filter 処理に，空間インデックスの各ノードの最小包囲矩形（MBR）を利用する．MBR は，その下位ノードや空間データオブジェクトを包含する領域を表しているため，各データ集合に対する空間インデックスの MBR を比較しながら，各インデックスに対する探索を同期的に進めることで，比較対象である空間データオブジェクトを絞り込むことができる．一方，区画分割に基づく手法では，データの分布する空間を複数の区画（partition）に分割し，各区画単位で 2 つのデータ集合の要素である空間データオブジェクト同士を比較するこ

とで，filter 処理を行っている．本論文で提案する手法では，センサデータ統合のための filter 処理を，受信したセンサデータを分割領域ごとに処理することで実現しているとも考えられ，その観点では，spatial join に関する研究と類似していると言える．

しかしながら，既存の spatial join の手法の多くは，統合処理の途中結果を提供することを考慮していない．途中結果を逐次的に表示することは可能であるが，filter 処理のためのデータ構造に依存することになる．たとえば，区画分割に基づく手法 [Patel *et al.* 96, Lo *et al.* 96, Koudas *et al.* 97] では，区画単位で表示され，文献 [Huang *et al.* 97, Lo *et al.* 94] の方法では，空間インデックスの構造に依存した順序で表示されると考えられる．したがって，これらの手法では，必ずしも，ユーザの理解しやすい順序で統合結果を提示できるとは限らない．それに対して，本論文で提案するセンサデータの検索／統合手法は，ユーザに対する視覚的な効果を考慮した区画分割を行っている点で既存の spatial join 手法と大きく異なる．提案手法は，途中結果に意味を持たせるために，ユーザが指定した領域分割のパラメータに基づいて統合結果を表示できるため，ユーザに対する視覚的効果も大きく，情報提示技術として有用であると考える．

さらに，spatial join の多くは，ローカルディスク中の 2 つのデータセットを対象としており，ネットワークを介して，リモートホスト中の空間データベースから，逐次的に提供されるデータを処理する枠組みとして利用するためには，枠組みの拡張が必要である．

2.1.5　地理情報システム

地理情報システム (Geographic Information System: GIS) の分野では，地図データや地理データなどの空間データを管理，検索，加工，統合，表示するための技術に関する様々な手法が考案され，GIS ソフトウエア上の機能として実装されているものも多い [Rigaux *et al.* 01, Longley *et al.* 01, 伊理 99, 張 01, 野上 他 01, 村井 02, 小方 他 98]．代表的な地理情報処理のための手法を以下に挙げる．

- 空間データ統合 (spatial data integration)
  包含，交差，近接などの空間的な関係を比較し，複数の空間データを統合する方法のことを，本論文では，空間データ統合と呼ぶ．空間データ統合は，空間検索や空間分析を行う上で不可欠であり，地理情報処理を支える基本的な手法になっている．

- 空間補間 (空間内挿，spatial interpolation)
  空間補間は，「地理データは場所に関連し，近いデータほど関連性が高い」[Longley *et al.* 01] という性質を利用して，ある地点の地理データを周辺の地理データから推論するものである．空間補間の手法としては，逆距離加

重法 (Inverse Distance Weighted: IDW)，スプライン，クリギングなど様々な手法が提案されている [Longley *et al.* 01, McCoy *et al.* 01, 伊理 99, 張 01, Bailey *et al.* 95]．空間補間の手法は，点として分布している離散的な地理データを面的な連続性のあるデータへと変換できるため，様々な地理情報処理に利用されている．

- オーバーレイ処理 (overlay analysis)
  オーバーレイと呼ばれる空間分析手法では，複数の空間データを重ね合わせて，空間的関係に基づいて新たな空間データを生成したり，空間データの属性値を計算するといった処理が行われる [Longley *et al.* 01, 張 01, 野上 他 01].

これらの地理情報処理の基本となる手法は，地理的な位置や関係性に基づいて，空間データを抽象度の高い表現に変換できる，という特徴があるため，本論文では，これらの手法を総称して “空間集約 (spatial aggregation)” と呼ぶ．データベースの分野では，“集約 (aggregation)” という言葉がよく使われるが，“空間集約” は，空間データの集約処理のことを指す．

空間集約処理のセンサデータの視覚化への適用

既存のセンサデータ閲覧システム [日本気象協会, 環境省, 国土交通省 b] では，位置情報に基づいて，各センサデータをシンボルとして地図上に配置することはできるが，空間データとの地理的な関係性を調べる機能を装備していない．また，各センサデータの分布は異なるため，点データのまま，種類の異なるセンサデータを関連付けるのは容易ではない．さらに，ユーザが知りたいのは，ノードの分布状況ではなく，全体的なデータの分布傾向や指定した地点に関する環境情報である．大量のセンサデータを閲覧する場合には，これらの点データを整理し，ユーザの理解しやすい情報に変換することが有用である．大縮尺での閲覧を考えた場合には，センサノードの分布密度によっては指定した地点あるいは表示範囲内にセンサデータが存在しない場合も考えられるため，データを補間するための技術が必要である．

空間集約の手法によって，表示レベルの重ね合わせではなく，データレベルでの統合が可能となるため，空間集約の手法をセンサデータの視覚化に適用することは効果的であると考えられる．空間データ統合によって，複数多種のセンサデータを，行政区ポリゴンなどの空間データオブジェクトに関連付けることで，その空間データオブジェクトを介して，それらのセンサデータを統合することが可能である．空間補間の手法を用いることにより，点分布として表現されるセンサデータを，面的な分布を持つ連続性のあるデータに変換することが可能である．種類の異なるセンサデータを，空間補間によって同じ粒度のメッシュデータに変換し，オーバーレイ処理を施すことによって，同じ位置のセルの値を比較することが可能となる．セルは，メッシュの構成要素である各区画を表す．

2.2　センサデータの検索と統合

2.2.1　異種センサネットワーク環境

現在，気象情報や交通情報を提供するための Web 情報サービスが多数存在し，様々な種類のセンサデータをオンラインで利用するための環境が整備されつつある [環境省, 国土交通省 b, 日本気象協会, IWATE]．たとえば,「大気汚染物質広域監視システム」[環境省] では，NO，NO_2，SO_2，Ox などの大気汚染物質に関する測定結果が公開されており,「川の防災情報」[国土交通省 b] では，河川に関する水位，水質，雨量などの情報を閲覧できるようになっている．図 2.5 に，これらのセンサノードの分布状況を示す.

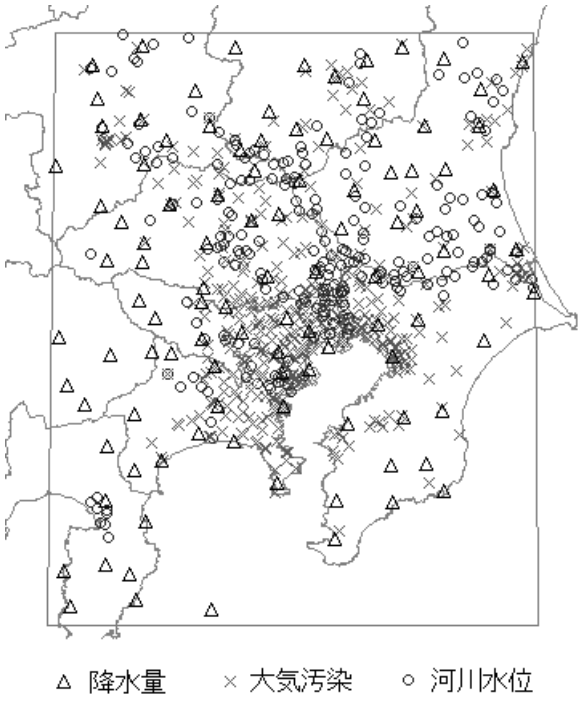

図 2.5: 異種センサネットワーク環境

図 2.5 の矩形領域には，411 個の大気汚染観測ノード [環境省]，305 個の河川水位観測ノード [国土交通省 b]，105 個のアメダスの降水量観測ノードが含まれており，密度や分布の異なるセンサノードが混在していることがわかる．なお，大気汚染観測ノードと河川水位観測ノードに関しては，各データ提供サイトで測定値とは別に観測点情報が公開されており，その中に，観測点の位置情報として住所が記載されている．本論文では，これらのノードの分布を表示するために，CSV アドレスマッチングサービス [CSIS] を利用して，住所情報から各観測点に対する経度緯度座標を求めている.

本論文では，それぞれのセンサノードの集合を 1 つのデータリソースと考え，“センサネットワーク” と呼ぶ．そして，図 2.5 のように，異種のセンサデータを提供するセンサネットワークが混在する環境（“異種センサネットワーク環境”）を想定する．異種センサネットワーク環境では，各センサネットワークの設置されている領域が，地理的に離れて存在する場合もあれば，重なり合うこともある．また，あるセンサネットワークの構成ノード数が多い場合には，データの管理や検索という点で，地域ごとに別々のセンサネットワークとして構成することも考えうる．

また，前述のセンサデータ提供サイト [環境省, 国土交通省 b] では，最新のセンサデータだけでなく，過去のある時点のセンサデータを参照することが可能である．たとえば,「大気汚染物質広域監視システム」では，測定値が 1 時間ごとに更新され，過去 7 日分の測定値を閲覧することができる．すなわち，これらのサイトでは，センサによって観測されたデータを時系列として保持していることになる．本研究では，このような時系列のセンサデータを蓄積したデータベースのことを，“センサデータベース” と呼ぶ．異種センサネットワーク環境においても，各センサネットワークから提供されるセンサデータが，対応するデータベースに時系列として蓄積されるものとする．ユーザは，ネットワーク上に分散した複数のセンサデータベースにアクセスすることで多種のセンサデータを利用できる．

2.2.2 センサデータの検索／統合のための要素技術

ネットワーク上に分散したセンサデータベースからの異種のセンサデータを統合する問題は，センサフュージョン [山崎 他 92] というよりも，むしろ，異種の情報源からのデータを統合する問題に近い．また，各センサデータベースより取得されるセンサデータは，時間データ (temporal data) あるいは時系列 (time series) として表現される．

異種情報の統合

データベースの分野では，異種データベースやマルチデータベースに関する研究として，異種情報統合の問題が扱われている．データアクセスのためのインタフェースや検索結果の表現が統一されていないなどのデータベースの異種性に対処するために，mediator と wrapper から成るアーキテクチャがよく利用され，“連邦データベースシステム (federated database systems)” と呼ばれている [Wiederhold 92, Roth *et al.* 97, 吉野 他 98]．Wrapper は，異種のデータリソースごとに存在し，mediator からの問合せをデータリソースが解釈できる形式に変換する役目を果たす．mediator は，wrapper の選択 (すなわちリソースの選択) や複数の wrapper に対する問合せ最適化などの処理を行い，アプリケーションと wrapper の間で仲介を行う．

一方，マルチエージェントシステムの分野では，“連邦アーキテクチャ(federation architecture)” が提案されている [Genesereth *et al.* 94]．連邦アーキテクチャでは，ファシリテータ (facilitator) というエージェントが，異なるドメイン上のエージェント間の協調を促進する．ファシリテータは，同一ドメイン内のエージェントの提供するサービスを他のドメインのエージェントに公開する役目を果たす．他にも，マルチエージェントシステムの分野では，broker agent，directory agent，middle-agent といった様々なファシリテータと同じ役目を果たすエージェントが報告されている [Decker *et al.* 96, Decker *et al.* 97, Bradshaw 97, 高橋 他 98]．

ここでは，以上のようなデータベースの分野やマルチエージェントシステムの分野で提案されている，仲介エージェント (mediator) とラッパエージェント (wrapper) から構成されるシステムを総称して，“連邦エージェントシステム” と呼ぶ．連邦エージェントシステムでは，各データリソースに対応した wrapper が，そのデータリソースに関する記述 (情報源記述) を仲介エージェントに登録し，仲介エージェントはその情報を各ラッパエージェントの属性情報として保持する．仲介エージェントは，クライアントからの検索要求に対して，ラッパエージェントに関する属性情報を調べ，該当するラッパエージェントを探し出す．すなわち，連邦エージェントシステムでは，各データリソースがどのようなデータを提供するか，すなわち情報源をどのように記述するかが重要である [Levy *et al.* 96]．

文献 [宗像 他 00] では，センサデータなどの周期的に発生する複数のデータ列から，鮮度と同期度を考慮してデータを選択する方式を提案している．周期の異なる周期データを提供する異種の情報源に対する wrapper と，これら複数の wrapper に対して，連続的な問合せや逐次的な問合せを発行する mediator を想定している．データ列の周期が異なるという情報源の異種性に着目しており，センサデータを対象としている点で興味深いが，本論文では，センサデータの種類やセンサの設置場所が異なるという異種性に注目する．

また，時間的に変化するデータを提供する動的な情報源を対象とした情報収集のための mediator に関する研究も行われている [北村 他 02] が，時系列データではなく，常に最新のデータを対象としている点，位置情報に基づく統合を目的としていない点で本論文のアプローチとは異なる．

Temporal join/aggregation

時間データベース（temporal database）の分野では，時間属性を持ったデータ（temporal data）のための結合演算（temporal join）や集約演算（temporal aggregation）に関する研究が多数行われている．Temporal join に関する研究の中で，本論文の提案手法と関連性が高いものとして，区画分割に基づく手法があげられる [Soo *et al.* 94, Son *et al.* 96]．これらの手法では，時間軸を partition と呼ばれる区画に分割し，各データセットから取り出した temporal data を partition に割り当て

る．そして，各 partition ごとに，別々のデータセットに属するデータ同士に対して結合演算を行うことで，全体の処理の効率化を図っている．また，temporal aggregation に関しても，同様に，区画分割に基づく手法が提案されている [Moon *et al.* 00]．この手法は，時間軸を bucket と呼ばれる時区間に分割し，temporal data を 複数の bucket に関連付け，各 bucket ごとに処理を行う．

これらの temporal join/aggregation に関する手法と，本論文における時区間分割に基づくセンサデータの提供／統合手法は，統合対象である temporal data をネットワークを介して取得するという違いはあるが，複数の時間データベースから取得したデータに対して，分割時区間ごとに統合処理を行っている点で，類似している．しかしながら，既存の temporal join/aggregation の手法の多くは，結合演算や集約演算の効率化を目的としており，各 partition ごとの計算結果または全体の計算結果を算出することに重点が置かれている．本論文では，むしろ，各時区間ごとの処理や計算結果の出力だけではなく，各時区間に対する統合処理の結果を，どのように表示していくかという統合結果の提供順序に着目している．提案手法では，ユーザにとって直感的で理解しやすい順序で，統合の途中結果を出力できる．したがって，既存の temporal join/aggregation 技術に比べて，提案手法は，時間属性を持つセンサデータの効果的なプレゼンテーションを実現するための統合技術として有用であると考える．

2.3 処理結果のインクリメンタルな提供

2.3.1 Anytime algorithm

通常，リアルタイムシステムの分野では，“実時間性” といった場合には,「ある決められた時間 (デッドライン) までに処理を終了させられるか」を問題にしていることが多い．しかしながら，一般に分散システムでは，ネットワークやマシンの負荷などの影響を受けるために，通信コストを予測し，実時間性を保証することが非常に難しい．

処理時間が制限される状況での問題解決の手法として “anytime algorithm” が人工知能の分野で提案され，研究が進められている [Dean *et al.* 88, Boddy *et al.* 89, Boddy *et al.* 94, Zilberstein *et al.* 93, Zilberstein *et al.* 96, Zilberstein *et al.* 99]．Anytime algorithm は，問題解決に割り当てられる時間が増加するにつれて解の品質が単調に増加する性質を持つ手法であり，処理途中に割り込まれた時にも何らかの結果を返すことができる．Anytime algorithm は，実時間性を持つ問題解決の枠組みとして提案されており，具体的な実装は問題領域に大きく依存する．Anytime algorithm における解品質の向上は，“performance profile” と呼ばれる時間関数として表現される．Performnace profile は，時間と品質のトレードオフを示す．従来の問題解決アルゴリズムでは，すべての処理が終了してから解を出力するため，

解品質は図 2.6(a) に示すように変化する．それに対して，anytime algorithm では，図 2.6(b) に示すように解品質は時間とともに単調に増加する．定められた時点ごとに処理の中間結果を保持することによって，段階的に解品質を向上させることもできる (図 2.6(c))．

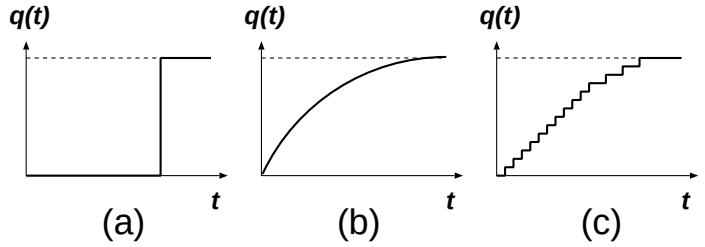

図 2.6: 処理品質の時間変化

他方，リアルタイムシステムの分野では，“インプリサイス計算 (imprecise computation)” [Lin *et al.* 87, Bettati *et al.* 92] という anytime algorithm とよく似た概念が提唱されており，いくつかのシステムに応用されている．文献 [Chen *et al.* 97, Huang *et al.* 95] では，静止画／動画などの画像データ転送のためのアルゴリズムが提案されている．しかしながら，画像データの特性を活かした転送方法であるため，XML などのテキストで表現された空間データやセンサデータの転送に適用することは難しい．さらに，画像データの送受信プロセスのみに着目しているため，本研究で想定している空間データやセンサデータを管理するデータベースの探索に適用することはできない．

リアルタイムデータベースの分野で “approximate query processing” という手法が提案されている [Vrbsky *et al.* 93, Vrbsky 97]．この手法は，imprecise computation に基づいており，時間が制限される状況において関係データベースに対する問合せの結果出力を逐次的に行うことを目的としている．文献 [Vrbsky 97] では，センサデータのような時間属性を持つデータ（temporal data）を扱えるモデルへの拡張が行われており，逐次的な結果出力の方法について述べているが，その出力結果の処理方法までは言及していない．それに対して，本論文では，空間的かつ時間的なセンサデータ統合のための枠組みとして逐次的なデータ提供方式を提案している．領域分割と時区間分割のパラメータを指定することで，どのように検索結果を分割するかだけでなく，位置情報と時間情報を考慮して，センサデータ統合の結果を，どのように提示するかを制御することができる．また，本論文では，異種センサネットワーク環境を想定しているため，関係データベースシステム内の問合せ処理ではなく，連邦データベースの構成要素である仲介エージェントとセンサデータサーバとの連携方法に焦点を当てている点も異なる．

2.3.2 途中結果を利用するアプローチ

処理の途中結果を逐次的に提供するアプローチは，ユーザへの情報提供という点で効果的であり，データベースの分野でも，いくつかの研究事例が見られる [Hellerstein *et al.* 97, Shanmugasundaram *et al.* 00]．“online aggregation” [Hellerstein *et al.* 97] では，集約演算の対象となるデータが逐次的に提供され，受信データを利用した処理の結果が，集約の途中結果として信頼度とともに提示される．CONTROL Project[CONTROL] では，online aggregation に基づく様々なアプリケーションが構築されている．たとえば，各州ごとの統計データを利用した集約処理の途中結果が，各州に対応するポリゴンデータの表示に反映され，Web におけるインターレース GIF やプログレッシブ JPEG のように，表示情報が時間とともに段階的に詳細化されていく．文献 [Shanmugasundaram *et al.* 00] では Web 検索の部分的な結果を生成できる問合せ処理システムについて述べている．しかし，これらの研究は，いずれも，関係データベースに対する問合せや関係データに対する集約演算を対象としており，空間データベースやセンサデータを対象としたものではない．関係データベースにおける集約演算は，テーブル (表) 操作に基づくものであるが，空間データに対する集約演算は，2.1 節で述べたように，地理情報システム，空間結合 (spatial join)，計算幾何学の分野で提案されている空間的な関係性に基づく手法を利用する．空間集約に基づくセンサデータ統合を対象とした場合，むしろ，センサデータの空間データの側面に着目した処理方法が効果的であると考える．

インクリメンタルに処理結果を提供するその他の方法としては，領域に着目する以外にも，地図データの階層性に着目する方法や地図データの詳細度に着目する手法も考えられる．地理情報システムや地図ソフトウェアでは，道路，鉄道，行政界などの地図データが別々のレイヤとして管理されており，文献 [Bertolotto *et al.* 99, Bertolotto *et al.* 01] では，ベクタ形式の地図データをレイヤの順序で提供する手法を提案している．

文献 [Hjaltason *et al.* 95, Hjaltason *et al.* 99] では，空間データオブジェクトの順序付け (ranking) のための最近傍探索アルゴリズムを提案している．この手法では，基準点からの距離に基づいてオブジェクトが順序付けられ，探索結果として，基準点に近い空間データオブジェクトから順番に出力される．この手法は，インクリメンタルに結果を出力できるという点で，本論文における空間データ検索のための手法と類似しているが，本研究の目的は，応答性の良い空間情報の閲覧システムを実現することにあり，文献 [Hjaltason *et al.* 95, Hjaltason *et al.* 99] では，単一のキューを用いた厳密な順序付けを行うのに対して，提案手法は，領域ごとに複数のキューを用いて，領域単位での順序付けを行う点が大きく異なる．分割パラメータを指定することで，基準点からの距離だけでなく，各軸方向への情報提供の順序を制御できる点，また，ネットワーク上の分散したデータベースからのインクリメ

ンタルな情報提供を実現している点で，提案手法が有用であると考える．

2.4　まとめ

本章では，まず，空間情報の視覚化のための要素技術として，空間データベースや地理情報システム分野における空間データの検索および統合の手法について説明した．Spatial join の分野で提案されている区画分割の手法は，領域分割に基づく提案手法と類似している．しかし，その多くはネットワークを介した空間データの統合を扱っておらず，途中結果を出力することを目的としていないため，提案手法のように，ユーザの要求に応じた順序で統合結果を出力することはできない．

本研究では，センサデータを時系列として蓄積した “センサデータベース” がネットワーク上に複数存在する環境を想定し，アプリケーションは，このデータベースにアクセスすることで多種多様な時系列のセンサデータにアクセスできるものとする．異種センサネットワーク環境におけるセンサデータベース検索は，異種情報源からのデータ検索と同様の問題を含むため，本研究では，検索システムを連邦データベースシステムとして構成する．さらに，センサデータを “空間情報” として扱うためには，検索システムが位置指向検索をサポートすることが不可欠である．そして，収集したセンサデータの統合結果を，直感的で理解しやすい情報としてユーザに提示するために，地理的な関係性に基づく “空間集約” の手法を利用する．Web や地理情報システムの分野では，位置に基づき様々な情報を管理し，位置指向検索に対して関連情報を提供するシステムが多数存在しているが，ネットワーク上に分散したデータベースから収集した異種のセンサデータの統合や，それらのセンサデータを統合するための空間集約処理を扱っているわけではない．

ネットワークを介して，空間データやセンサデータを検索し，統合する際には，応答時間の増大という問題が生じる．本論文では，人工知能の分野で提案されている “anytime algorithm” の逐次性に着目し，空間データ検索とセンサデータベース検索に適用し，それぞれ 3 章と 4 章で述べる．Anytime algorithm は，処理品質が単調に増加する性質を持ち，処理結果を逐次的に出力するという特徴を持つが，適用する問題に応じ具体的な手順を考案する必要があり，どのように品質を定義するか，どのように中間結果を保持するかが問題となる．空間データ検索に対しては，空間データの収集率を処理品質として定義し，代表的な空間インデックスである R-tree に基づく探索アルゴリズムを考案する．また，センサデータベース検索に対しては，分散したサーバからのデータ収集という観点から仲介エージェントの機能に着目する．本研究では，空間情報の視覚化のための anytime algorithm に基づく検索手法を提案するが，このことは，人工知能における問題解決の枠組みとしてではなく，情報提示技術への応用を示すことにもなると考える．

第3章

領域分割に基づく空間データ検索手法

本章では，ネットワークを介した空間データ検索に対して，検索結果をインクリメンタルに提供するための領域分割に基づく手法を提案する．提案手法は，問合せ領域を分割し，その分割領域ごとに検索の途中結果をクライアントに提供する．この領域分割のアプローチは，本研究における空間情報のインクリメンタルな視覚化を実現するための基本的な考え方であり，4 章および 6 章において，この考え方をセンサデータベースに応用する．

3.1 Anytime algorithm に基づく検索手法

本論文では，空間データ検索に対する応答の実時間性を考慮するために，与えられたデッドラインまでに処理を完全に終了させるのではなく，前述の “anytime algorithm” の考えに基づき，空間データ検索に対する検索結果の品質を時間とともに向上させることとする．本章では，時刻 t における検索結果の品質 $q(t)$ を，クライアントにおける空間データオブジェクトの収集率として式 (3.1) のように定義する．

$$q(t) = \frac{N(t)}{N_{all}} \tag{3.1}$$

$N(t)$ は，t までにクライアントが収集したデータオブジェクトの個数である．N_{all} は，検索結果に含まれるデータオブジェクトの総数である．

検索結果を一括して転送する場合，検索結果の品質は 2.3.1 節の図 2.6(a) に示すように変化し，ユーザは，ある時刻まで何の結果も得ることができない．それに対して，本研究では，ユーザが検索の中間結果を得られるように，図 2.6(b)，(c) に示すように検索結果品質を変化させる．Web の検索システムなどでも，同様に検索結果品質は変化するが，ユーザの嗜好やアクセス数などの情報に基づき検索結果

に対して順序付けが行われており，空間データの検索においても，データ転送の順序を決定する必要があると考える．

本研究では，次の方針に基づき，空間検索に対するデータ提供手法を設計する．

- 通常，空間データベースシステムでは，探索の中間結果はシステムの内部では保持されるが，外部のクライアントに提供されるように設計されていない．本論文では，式 (3.1) に定義した検索結果の品質 $q(t)$ を単調に向上させるために，空間データを提供するシステムが，空間データ検索に対する探索の中間結果を保持し，その結果をクライアントに逐次的に転送する．これにより，検索結果である空間データが，クライアントに対してインクリメンタルに提供される．

- ユーザへの地図情報の提供のためにユーザの注目する領域の空間データを優先的に転送したり，クライアントでの空間情報処理のために領域ごとに空間データを転送するなど，ユーザにとって意味のあるまとまりとして，空間データを逐次的に提供することは有用であると考えられる．本研究では，空間データ検索の問合せ領域を考慮して転送データの順序付けを行う．

- 通常，データオブジェクトの取得はディスクアクセスを伴うので，空間データ検索においてはデータ探索コストが大きく，クライアントが大量のデータを要求する場合には，その探索コストはさらに大きくなることが予想される．そこで，全探索終了後に検索結果を順序付けるのではなく，データ探索そのものを制御する．

ネットワークを介して，空間データを利用した情報統合や解析など様々な空間情報処理をクライアントが行う場合には，必ずしも処理の開始時点にすべてのデータが揃っている必要はない．部分的な検索の途中結果を利用して処理を開始することも可能である．クライアントが逐次的に提供されるデータに基づき地図表示やデータ解析などの処理をするためには，それらの処理をしやすい意味のあるまとまりで提供されることが必要である．すなわち，領域を考慮したインクリメンタルなデータ提供が，クライアントでのインタラクティブかつ柔軟な処理を行う上で有用であると考える．

3.2　空間データ検索のためのインクリメンタルなデータ提供手法

本手法は，役割の異なる 2 つの計算モジュールから構成される．本章では，データ探索のための計算モジュールを探索プロセス，検索結果を転送するための計算モ

ジュールをデータ転送プロセスと呼ぶ．探索プロセスは，クエリに対してデータ探索を実行し，探索の途中結果をバッファに蓄えていく．一方，転送プロセスは，このバッファを随時参照し，中間結果を逐次的に転送する．これらのプロセスを並行に動作させ，探索結果をバッファを用いて共有させることにより，探索プロセスの動作中に，探索の中間結果をクライアントに転送することができる．

また，クライアントに転送される空間データオブジェクトの順序付けは，探索プロセスの途中結果の出力順序に依存するが，本手法は，クエリに指定された問合せ領域を分割し，その分割領域ごとに探索を実行することで対応する．クライアントは，この領域分割によって順序付けられた空間データを受信することになる．

さらに，本研究で前提としているのは，位置情報システムのための，空間データ基盤や G-XML などのテキストとして表現された空間データである．そこで，本論文では，クライアントに提供する空間データを XML（eXtensible Markup Language）により表現するものとする．

次に本手法におけるデータ探索のためのアルゴリズムを述べ，最後に検索結果の転送方法を述べる．

3.2.1 空間範囲問合せ

本章では，次の空間範囲問合せ (spatial range query) を扱う．

(a) 矩形問合せ Q_{rect}
$R_{query,rect} = \langle x_{min}, y_{min}, x_{max}, y_{max} \rangle$

(b) 距離に基づく問合せ Q_{circle}
$R_{query,circle} = \langle x_{center}, y_{center}, radius \rangle$

Q_{rect} と Q_{circle} は，2 次元の空間データオブジェクトを要求するクエリであり，その問合せ領域 (R_{query}) の表現が異なる．Q_{rect} は，2 つの頂点 (x_{min}, y_{min})，(x_{max}, y_{max}) によって定まる矩形を問合せ領域とし，Q_{circle} は，点 (x_{center}, y_{center}) を中心とした半径 $radius$ の円を問合せ領域とする．

ここで，領域 R_a と R_b に対して，次の 2 つの関数を定義する．

- $OVERLAP(R_a, R_b)$
  $R_a \cap R_b \neq \phi$ が成り立つ時，$true$ を返す．この時，R_a と R_b は重なり合う．
- $CONTAIN(R_a, R_b)$
  $R_a \supseteq R_b$ が成り立つ時，$true$ を返す．この時，R_a は R_b を含む．

本研究で扱う範囲問合せは，問合せ領域 R_{query} と重なるすべての空間データオブジェクトを収集するものとする．したがって，R_{query} を持つクエリに対する検索結果は，次の条件を満たすデータオブジェクト o の集合として表現される．

$$OVERLAP(R_{query}, o.geom) = true \tag{3.2}$$

$geom$ は，空間データオブジェクト o の幾何形状である．

3.2.2　分割領域リストの生成

本手法は，クライアントから送られたクエリに対して，まず，その問合せ領域 R_{query} を，互いに重なり合わない領域に分割する．

$$R_{query} = r_1 \cup r_2 \cup ... \cup r_k = \bigcup_{i=1}^{k} r_i \tag{3.3}$$

$$\forall i, j \quad r_i \cap r_j = \phi \quad (i \neq j) \tag{3.4}$$

r_i は分割によって生成された領域であり，k は分割数である．そして，これらの分割領域を順序付け，リスト $region\_list = \langle r_1, r_2..., r_k \rangle$ として保持する．この領域の分割および分割領域の順序付けは実装依存とし，領域分割については上記の条件を満たせばよい．たとえば，図 3.1 に示すように，$R_{query,circle}$ を半径方向に分割してもよいし，$R_{query,rect}$ をグリッドに分割してもよい．$R_{query,circle}$ の場合，各分割領域は 2 つの円に囲まれた領域として表現され，その分割領域を円の中心からの距離により順序付けることができる．本実験で用いた分割ポリシーについては，3.3.3 節で述べる．

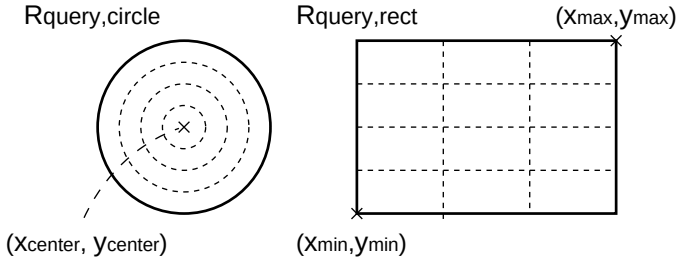

図 3.1: 問合せ領域の分割の例

3.2.3　領域分割に基づく探索アルゴリズム

探索プロセスは，生成された分割領域のリスト $region\_list$ の順序に従い，各分割領域ごとに空間インデックスを利用して探索を行う．たとえば，図 3.2 に示すよ

うに，問合せ領域 $R_{query,rect}$ を 2 つの矩形 (R_i, R_{i-1}) で囲まれた領域 r_i に分割し，ある基準点からの距離に基づいて分割領域 r_i を順序付け，その順序に基づいて探索を行う．

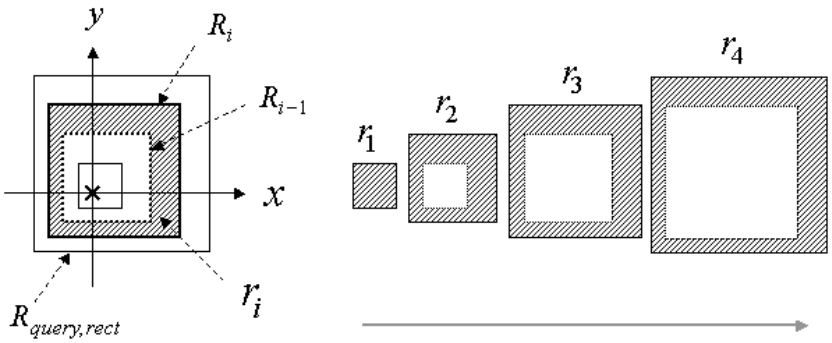

図 3.2: 問合せ領域の分割と分割領域の順序付け

この時，データ探索済みの領域は時間とともに拡大していくことになる．本論文では，拡大領域 R_i を，式 (3.5)，式 (3.6) のように定義することで，分割領域 r_i $(1 \leq i \leq k)$ を決定する．

$$R_i = \bigcup_{j=1}^{i} r_j = \begin{cases} R_{i-1} \cup r_i & (i > 1 \text{ の時}) \\ r_i & (i = 1 \text{ の時}) \end{cases} \tag{3.5}$$

$$R_{i-1} \subset R_i \tag{3.6}$$

式 (3.4)，式 (3.5) より $R_{i-1} \cap r_i = \phi$ が成り立つので，分割領域 r_i は拡大領域の差分領域として，式 (3.7) のように表現できる．

$$r_i = \begin{cases} R_i - R_{i-1} & (i > 1 \text{ の時}) \\ R_i & (i = 1 \text{ の時}) \end{cases} \tag{3.7}$$

一般に，木構造の探索アルゴリズムは，解候補となるノードをキューに挿入し，キューから取り出したノードを展開するという手順を繰り返すことで，探索を進める [Russel *et al.* 95]．本論文では，通常の空間インデックス探索の場合 (図 2.4) と同じように，各分割領域とノードの包囲矩形 MBR とを比較し，それらの領域が重なり合う場合に，キューを用いてノードを展開していく (図 3.3)．

図 3.3 の例では，図 2.3 中のエントリ e をキューから取り出し，そのエントリの MBR($e.rect$) と分割領域 r_i との比較を行い，e の指し示すノードのエントリ ($e1$,

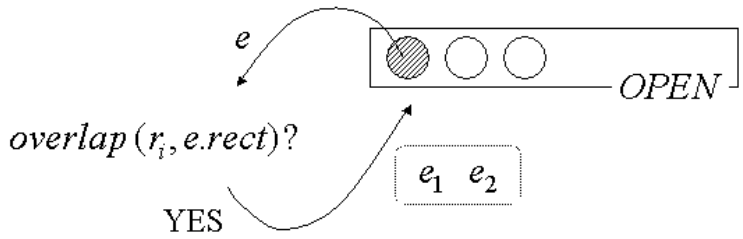

図 3.3: R-tree の探索過程におけるノードの展開 (取り出した要素が葉ノードのエントリでない場合)

$e2$) を新たな要素として，キューに挿入している．なお，本章で提案する探索アルゴリズムでは，新しい要素をキューの先頭に挿入するものとする．

また，空間データオブジェクトへアクセスする場合には，2.1.3 節で述べた “filter&refinement” 戦略に従う (図 3.4)．

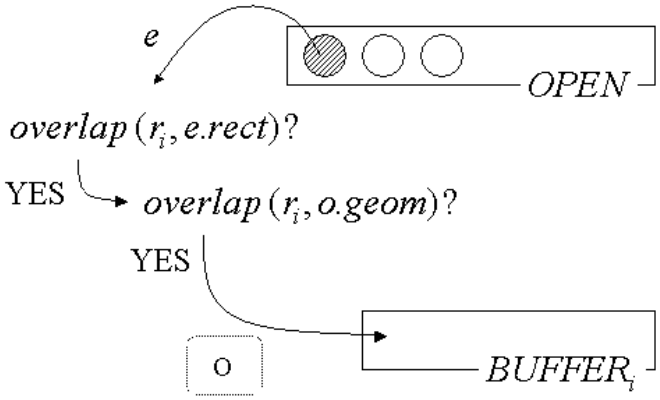

図 3.4: “filter&refinement” 戦略に基づく条件の比較 (取り出した要素が葉ノードのエントリである場合)

図 3.4 では，キューからエントリ e を取り出し，その MBR($e.rect$) が分割領域 r_i と重なり合う場合に，e が指し示す空間データオブジェクト o の幾何形状 $o.geom$ と r_i との比較を行う．$o.geom$ が r_i と重なり合う場合に，探索結果を保持するバッファに挿入する．

次に，分割領域 r_i $(1 \leq i \leq k)$ に対する基本的な処理手順を図 3.5 に示す．$OPEN$ は探索のためのキューを，$BUFFER_i$ は r_i に対する探索結果を保持するためのバッファを表す．キュー $OPEN$ の要素は，R-tree におけるノードのエントリである．各ノードは複数のエントリを持ち，MBR と，子ノードあるいは空間データオブジェクトへのポインタを属性として持つ．そのエントリ e が属性として持つ $e.rect$ は，

そのエントリが指し示すノードあるいは空間データオブジェクトの包囲矩形である (図 2.3). エントリが空間データオブジェクト o を指し示す場合には，$e.rect$ は，そのオブジェクトの包囲矩形を表す (図 2.2).

```
L1  OPEN に，R-tree の根ノードのすべてのエントリを挿入
L2  while OPEN ≠ φ do
L3    OPEN から要素 e を取り出す
L4    if e が葉ノードのエントリである then
L5      if OVERLAP(r_i, e.rect) = true then
L6        if OVERLAP(r_i, o.geom) = true then
L7          o を BUFFER_i に挿入
L8        endif
L9      endif
L10   else
L11     if OVERLAP(r_i, e.rect) = true then
L12       e の指し示すノードのすべてのエントリを OPEN へ挿入
L13     endif
L14   endif
L15 end
```

図 3.5: 分割領域 (r_i) に対する探索アルゴリズム

要素 e が葉ノードのエントリでない時 (図 3.3) は，その要素の包囲矩形 $e.rect$ と分割領域 r_i との包含関係を調べる (L11)．それに対して，要素 e が葉ノードのエントリである時 (図 3.4) は，データレコード中のある空間データオブジェクト o へのポインタを含むので，“filter&refinement” 戦略に従い，そのオブジェクトの包囲矩形 $e.rect$ との比較を行ってから，そのオブジェクトの幾何属性 $o.geom$ と分割領域 r_i との包含関係を調べる (L4〜L9).

ただし，分割領域にまたがるオブジェクトが存在する時に，各分割領域 r_i ごとにそのオブジェクトとの包含関係を調べると，それぞれの分割領域ごとに同一のオブジェクトが通知される．したがって，ある分割領域に対するオブジェクトを探索する時，既に通知されたオブジェクトを除外する必要がある．そこで，$OPEN$ から取り出した要素 e が葉ノードのエントリでない場合 (L11〜L13) および e が葉ノードのエントリである場合 (L4〜L9) において，各分割領域ではなく，拡大領域との比較を行う．以下，(1) e が葉ノードのエントリでない場合，(2) e が葉ノードのエントリである場合に分けて，条件判定の方法を述べる．

(1) e が葉ノードのエントリでない場合 – 包囲矩形との比較

キューから取り出した要素 e が葉ノードのエントリでない場合は，図 3.5 の L11 で，式 (3.8) の左辺の真偽値を調べるために右辺を評価する．

$$\begin{aligned} OVERLAP(r_i, e.rect) &\leftrightarrow \begin{cases} C_1 \wedge C_2 & (i > 1 \text{ の時}) \\ C_1 & (i = 1 \text{ の時}) \end{cases} \\ C_1 &= OVERLAP(R_i, e.rect) \\ C_2 &= \neg CONTAIN(R_{i-1}, e.rect) \end{aligned} \tag{3.8}$$

式 (3.8) の右辺では，拡大領域 R_i および R_{i-1} との比較が行われる．R-tree では，各ノードの包囲矩形はすべての子ノードの領域を覆う領域として定義される．したがって，$CONTAIN(R_{i-1}, e.rect) = true$ を満たす矩形領域 $rect$ に含まれるすべての空間データオブジェクトは，R_{i-1} に含まれているため，r_{i-1} に対する探索時に既に通知されていることになる．よって，R_{i-1} に含まれるノードを展開する必要はなく，R_i と重なり，R_{i-1} に含まれないものを探索の候補とすればよい．

(2) e が葉ノードのエントリである場合 – 空間データオブジェクトとの比較

e が葉ノードのエントリである場合，すなわち e が空間データオブジェクト o へのポインタを持つ場合には，まず，式 (3.8) を利用して，e について拡大領域との比較を行う (図 3.5 の L5)．この条件を満たさないオブジェクトは，包囲矩形が R_{i-1} に含まれるので，探索結果の候補から除外できる．そして，条件を満たす o について，図 3.5 の L6 で，式 (3.9) の右辺を評価し $true$ を返す場合に，o を探索結果として通知する．

$$\begin{aligned} OVERLAP(r_i, o.geom) &\leftrightarrow \begin{cases} C_3 \wedge C_4 & (i > 1 \text{ の時}) \\ C_3 & (i = 1 \text{ の時}) \end{cases} \\ C_3 &= OVERLAP(R_i, o.geom) \\ C_4 &= \neg OVERLAP(R_{i-1}, o.geom) \end{aligned} \tag{3.9}$$

空間データオブジェクト o は最初に重なり合った分割領域に対して通知されるので，そのオブジェクトが既に通知されているかどうかは，R_{i-1} と重なり合うかどうかを調べることにより判定できる．

提案手法に基づく探索手順の例

たとえば，図 2.1 に示した R-tree に対して，図 3.6，図 3.7 のように，問合せ領域 $R_{query,rect}$ を矩形状の拡大領域 R_1，R_2，R_3 を用いて分割して，探索を行う場合

を考える．各分割領域は，$r_1 = R_1$，$r_2 = R_2 - R_1$，$r_3 = R_3 - R_2$ と定義される．図 3.6 は分割領域 r_1 に対する探索手順，図 3.7 は r_2 に対する探索手順を示す．

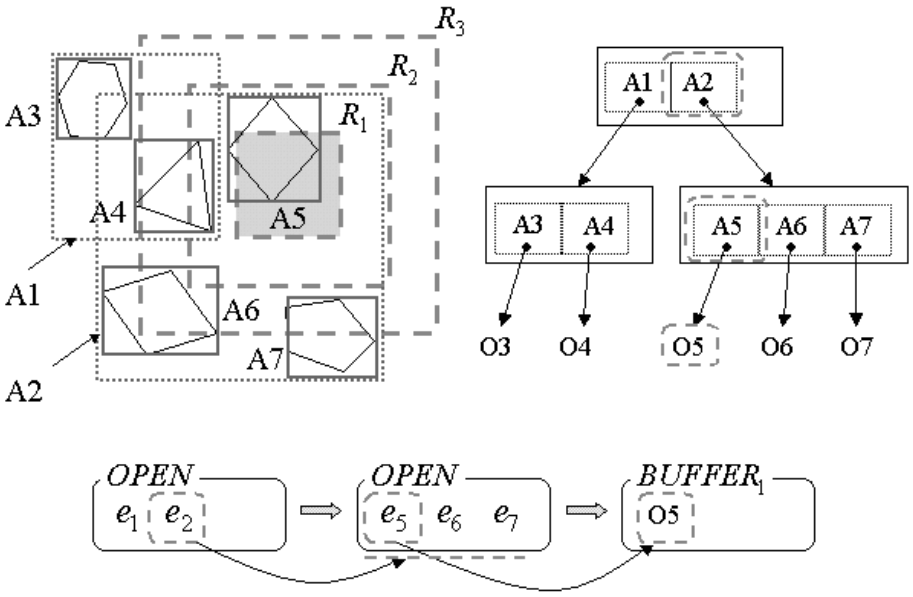

図 3.6: 提案手法に基づく探索手順の例 – 分割領域 r_1 に対する探索

図 3.6 では，まず，$OPEN$ 中のエントリ (e_1, e_2) について，$e_2.rect$ が R_1 と重なり合うため，e_2 が指し示す子ノードのエントリ (e_5, e_6, e_7) を $OPEN$ に挿入する．$e_5.rect$ が R_1 と重なり合い，e_5 が指し示すオブジェクト o について，$o.geom$ が R_1 と重なり合うため，o が r_1 に対する探索結果となり，$BUFFER_1$ に挿入される．

図 3.7 では，e_1，e_2 の MBR について，分割領域 r_2 との包含関係を調べるために，式 (3.8) に従って，R_1 および R_2 との比較が行われる．$e_1.rect$，$e_2.rect$ がともに条件を満たすため，e_3, e_4 および e_5, e_6, e_7 が $OPEN$ に展開される．これらの $OPEN$ 中のエントリについて，$e_3.rect, e_7.rect$ は R_2 と重なり合わず，$e_6.rect$ は R_2 と重なり合うが，$o_6.geom$ が R_2 と重なり合わない．e_5 については，式 (3.9) の C_3 の条件は満たすが，C_4 の条件を満たさない．r_2 に対する探索結果としては，e_4 に対するオブジェクト o_4 のみが $BUFFER_2$ に挿入される．

3.2.4 アルゴリズムの改良

前述のアルゴリズムは，各分割領域それぞれに対して探索を根ノードから開始するため，探索済みのノードや空間データオブジェクトを何度も調べることになる．さらに，空間データオブジェクトの通知の重複を避けるために，分割領域と重なるオブジェクトの中から既に通知したものを除外する処理を行う．空間データオ

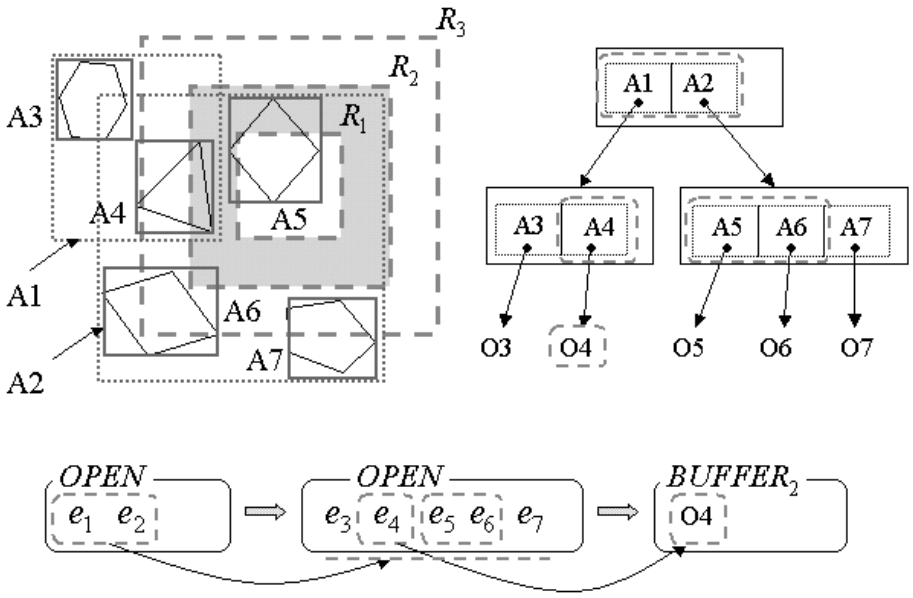

図 3.7: 提案手法に基づく探索手順の例 – 分割領域 r_2 に対する探索

ブジェクトとの包含関係を調べる場合には，データレコードへのアクセスを伴う上に，MBR との包含関係を調べる場合より比較コストも大きいため，前述のアルゴリズムは探索のオーバヘッドが大きいと考えられる．

そこで，探索のために単一のキューを用いるのではなく，各分割領域 r_i ごとに探索のためのキュー $OPEN_i$ を作成し，これらの複数のキューを用いて探索を行う (図 3.8，図 3.9)．図 3.6，図 3.7 と同様の例について，図 3.8 は分割領域 r_1 に対する改良アルゴリズムに基づく探索手順，図 3.9 は r_2 に対する探索手順を示す．図 3.6，図 3.7 と同様に，r_1 に対する探索結果は $BUFFER_1$ に，r_2 に対する探索結果は $BUFFER_2$ に，それぞれ格納される．ただし，図 3.6，図 3.7 では，探索のために単一のキュー ($OPEN$) を用いているのに対して，図 3.8，図 3.9 では，分割領域 r_1，r_2 それぞれに対してキュー $OPEN_1$，$OPEN_2$ を用いている点が異なる．

$OPEN_i$ から取り出した要素が分割領域 r_i と重なり合う限りは，$OPEN_i$ を用いて要素を展開していく．重なり合わない場合には，その要素を次の分割領域 r_{i+1} に対する探索の候補として $OPEN_{i+1}$ に挿入する．r_i に対する探索が終了した後，次の領域 r_{i+1} に対する探索は，あらためて根ノードから開始するのではなく，$OPEN_{i+1}$ に既に含まれている要素を用いて開始する．

図 3.8 では，分割領域 r_1 に対する探索の過程で，e_1, e_6, e_7 の包囲矩形が R_1 と重なり合わないため，次の分割領域 r_2 に対するキュー $OPEN_2$ に挿入される．図 3.9 では，$OPEN_2$ に格納されたこれらの要素を用いて，r_2 に対する探索が開始される．さらに，r_2 に対する探索過程で $OPEN_3$ に格納された要素 e_3, e_6, e_7 を用いて，

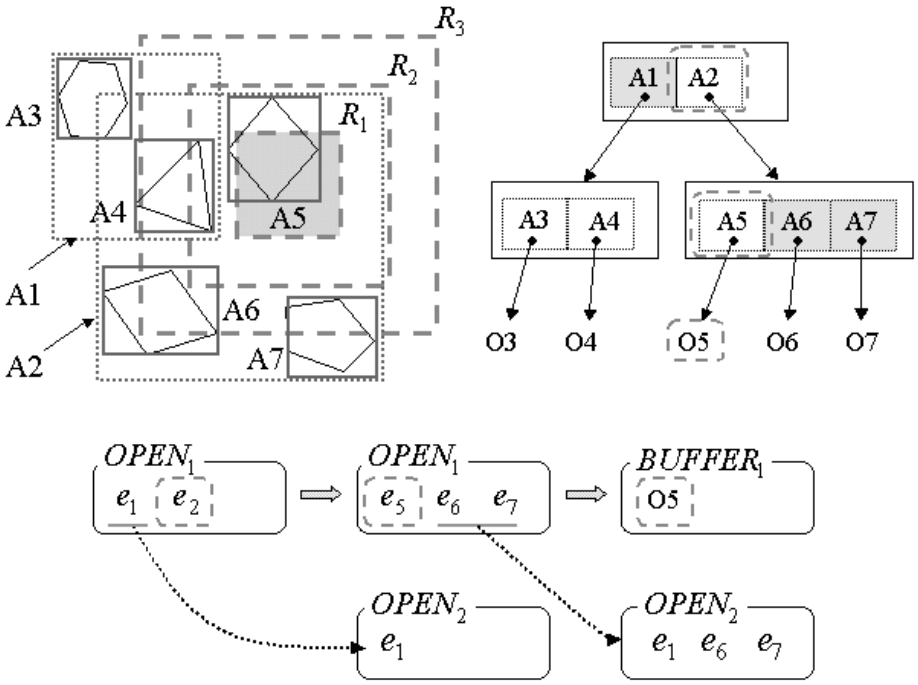

図 3.8: 拡張アルゴリズムに基づく探索手順の例 – 分割領域 r_1 に対する探索

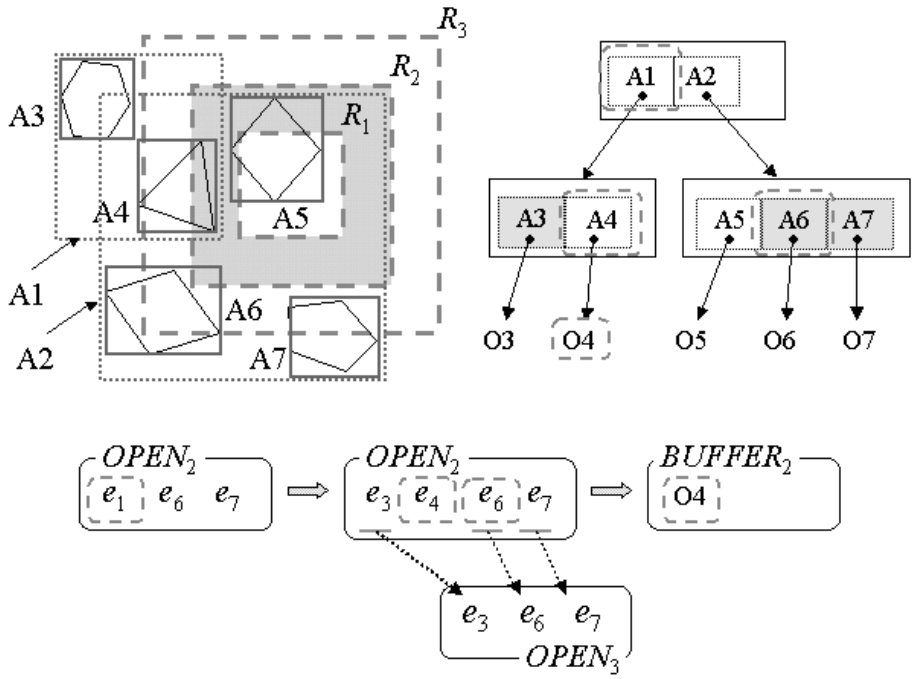

図 3.9: 拡張アルゴリズムに基づく探索手順の例 – 分割領域 r_2 に対する探索

r_3 に対する探索が開始される.

図 3.10 に，拡張したアルゴリズムの詳細を示す.

```
L1   region_list = ⟨r_1, r_2..., r_k⟩ を生成
L2   r_i に対して，OPEN_i を生成
L3   r_i に対して，BUFFER_i を生成
L4   OPEN_1 に R-tree の根ノードのエントリを挿入
L5   for region_list の順に，r_i をチェック do
L6     while OPEN_i ≠ φ do
L7       OPEN_i より要素 e を取り出す
L8       if e が葉ノードのエントリである then
L9         if OVERLAP(r_i, e.rect) = true then
L10          if OVERLAP(r_i, o.geom) = true then
L11            o を BUFFER_i に挿入
L12          else
L13            e を OPEN_{i+1} に挿入
L14          endif
L15        else
L16          e を OPEN_{i+1} に挿入
L17        endif
L18      else
L19        if OVERLAP(r_i, e.rect) = true then
L20          e の指し示すノードのすべてのエントリを OPEN_i に挿入
L21        else
L22          e を OPEN_{i+1} に挿入
L23        endif
L24      endif
L25    end
L26  end
```

図 3.10: 空間データ探索アルゴリズム

この拡張アルゴリズムでは，検索結果として既に通知されたオブジェクトが探索の候補としてキューに蓄えられることはないので，分割領域 r_i とキューから取り出した要素 e を比較する時 (L9，L19)，拡大領域 R_i との比較を行えばよい (R_{i-1} との比較は，3.2.3 節で述べたように，既に通知済みのデータオブジェクトを除外するために行われる)．e の包囲矩形 $e.rect$ との比較を行う場合 (L9，L19) には，$OVERLAP(R_i, e.rect) = true$ が成り立つかどうか，空間データオブジェクト o と

の比較を行う場合 (L10) には，$OVERLAP(R_i, o.geom) = true$ が成り立つかどうかを調べる．

3.2.5 検索結果の転送

データ探索プロセスは，分割領域 r_i に対する探索結果をバッファ ($BUFFER_i$) に蓄えていく．データ転送プロセスは，探索の終了した分割領域から順に，バッファ中の探索結果を参照し，逐次的にパケットとしてクライアントに転送する．探索プロセスと転送プロセスは並行に動作しているので，転送プロセスは，すべての探索の終了を待たずに，検索の途中結果をクライアントに転送することが可能である．さらに，各パケットに含むことのできるデータオブジェクトの最大数 (max_item) をパラメータとして指定することも可能である．この場合，max_item に指定した個数のデータオブジェクトを順次バッファから取り出して転送する．したがって，指定するパラメータによって，次の 3 通りの分割が考えられる．

- 領域分割のパラメータのみを指定する場合
  分割領域の順序に従って，検索結果が逐次的に提供される．
- 領域分割のパラメータとオブジェクト数 (max_item) を指定する場合
  分割領域の順序に従って，検索結果が提供され，各分割領域に対する検索結果は，max_item によって分割され，提供される．
- オブジェクト数 (max_item) のみを指定する場合
  領域分割は行われず，max_item に基づいて検索結果が分割され，提供される．

また転送プロセスは，各パケットに対して XML によるタグ付けを行う．実験で用いた XML 表現については，3.3.4 節で述べる．

3.3 実装

3.3.1 空間データ提供システム

提案手法に基づく空間データ提供システムを，Java 言語 (JDK1.2) を用いて実装した．

本システムは，TCP コネクションを介して空間データを提供するサーバとして動作し，探索プロセスとデータ転送プロセスは，それぞれスレッド (java.lang.Thread) として実装した．これらのスレッドは並行に動作し，バッファを用いて検索結果を共有する．探索用のスレッドは中間結果を保持するためのバッファを持ち，転送用のスレッドはこのバッファから送信すべきデータを取得する．

3.3.2　データ構造

本実験では，サンプルデータとして，国土地理院 [GSI] 発行の「数値地図 2500(空間データ基盤)」[JMC] を使用した．この数値地図には，街区，行政区，道路情報など様々な地図データが含まれ，図葉と呼ばれる矩形領域ごとに，それぞれのデータがファイルとして保存されている．各図葉は，縦 1500 メートル (m)，横 2000 メートル (m) の矩形領域である．座標表現は，緯度経度ではなく，平面直角座標系[1][JMC 98, 村井 02] で示され，単位は メートル (m) である．たとえば，平面直角座標 IX 系は，東経 139 度 50 分，北緯 36 度 0 分で表現される点を原点，横方向 (左 → 右) を x 軸，縦方向 (下 → 上) を y 軸とする座標系である．本実験で用いた数値地図 2500(空間データ基盤)「神奈川-3」「神奈川-4」[2]の地図データは，平面直角座標 IX 系 での座標で表現されている．

この数値地図 2500(空間データ基盤) に収録されているデータファイルから，図葉ごとに，線データおよびポリゴンデータを含むデータファイルをそれぞれ新たに生成し，空間データベースとして構成する．そして，この空間データベースの空間インデックスとして R-tree[Guttman 84] を実装した．R-tree では，葉ノードのエントリがデータレコードを特定するための id 情報を含むので，この id 情報としてファイル名とファイルポインタをインデックスの構築時に記録する．ファイルポインタはファイルの先頭からのバイト数である．データファイルへのアクセスは，このファイルポインタを引数として，java.io.RandomAccessFile クラスの seek() メソッドを用いて行う．なお，この空間インデックスはデータの種類ごとに作成するものとする．

3.3.3　問合せ領域の分割

式 (3.7) のように，各分割領域 r_i は拡大領域 R_i を用いて表現できるので，問合せ領域をどのように分割するかは，拡大領域をどのように表現するかによって決定される．本論文では，拡大領域 R_i の計算式を定義することによって，領域の分割方法を定める．具体的には，図 3.11 に示すように，ある基準点と領域分割の幅に基づき問合せ領域を分割するものとする．

矩形問合せ Q_{rect} については，その問合せ領域を次のパラメータに基づき分割する (図 3.11 (a))．

- 領域分割の基準点：(x_{base}, y_{base})
- x 軸方向の分割幅：$dx_+(\geq 0), dx_-(\leq 0)$
- y 軸方向の分割幅：$dy_+(\geq 0), dy_-(\leq 0)$

[1]日本には，19 個の平面直角座標系が存在する．
[2]日本測地系から世界測地系への移行に伴い，現在の刊行区分とは異なっている．

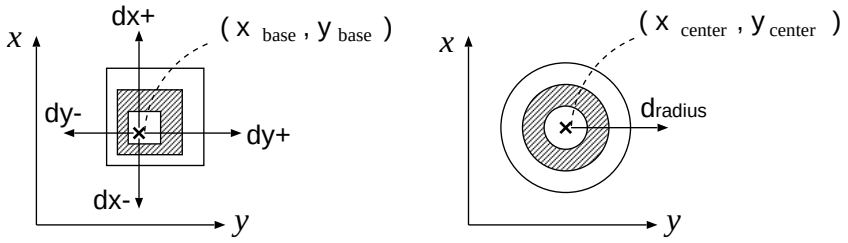

図 3.11: 基準点と分割幅を指定した時の問合せ領域の分割：(a) 矩形問合せ Q_{rect}，(b) 距離に基づく範囲問合せ Q_{circle}

これらの分割パラメータを利用して，拡大領域 $R_i(1 \leq i \leq k)$ を式 (3.10) に示すように計算する．

$$R_i = \begin{array}{ll} \langle max(x_{base} + dx_- \times i, x_{min}), & max(y_{base} + dy_- \times i, y_{min}), \\ min(x_{base} + dx_+ \times i, x_{max}), & min(y_{base} + dy_+ \times i, y_{max})\rangle \end{array} \tag{3.10}$$

$min(a, b)$ は，a と b とを比較し最小値を返す関数であり，$max(a, b)$ は，最大値を返す関数である．

一方，距離に基づく範囲問合せ Q_{circle} については，半径方向の分割幅 (d_{radius}) を用いて，その問合せ領域を分割する (図 3.11 (b))．領域分割の基準点は円の中心 (x_{center}, y_{center}) とし，拡大領域 R_i を式 (3.11) に示すように計算する．

$$R_i = \begin{cases} \langle x_{center}, y_{center}, d_{radius} \times i\rangle & 1 \leq i < k \text{ の時} \\ \langle x_{center}, y_{center}, radius\rangle & i = k \text{ の時} \end{cases} \tag{3.11}$$

これらの式によって計算される拡大領域 R_i は，式 (3.5)，(3.6) に示す条件を満たし，ある点を基準としてデータ収集領域を拡大させていくことができる．

なお本研究では，これらの領域分割のためのパラメータは，クライアントがサーバに送信するクエリに検索範囲とともに指定されるものとする．すなわち，ユーザの要求に応じた領域分割を行うためのパラメータがクライアント側で指定され，サーバは，その分割パラメータに基づき領域分割を行い，その探索結果を逐次的にクライアントに提供する．たとえば，ユーザの周辺の地図データをユーザに近い位置から表示する場合には，式 (3.11) に示す分割パラメータを利用し，基準点としてユーザの位置を指定すればよい．また，ユーザの移動あるいは地図のスクロールといった動作に合わせて地図データを取得する場合には，式 (3.10) に示す分割パラメータを利用し，移動方向に応じた分割幅の指定が可能である．

3.3.4　検索結果のタグ付け

図 3.12 に，本実装で用いた XML 表現のための DTD (Document Type Definition) を示す．線データおよびポリゴンデータは，それぞれ <**line**> タグおよび <**polygon**> タグによって記述される．図 3.13 に，図 3.12 に示した DTD に基づく空間データの記述例を示す．

```
<!ELEMENT spatial-data (line|polygon)*>
<!ELEMENT line (id?,((x,y),(x,y)))>
<!ELEMENT polygon (id?,((x,y),(x,y),(x,y)+))>
<!ELEMNET x (#PCDATA)>
<!ELEMENT y (#PCDATA)>
```

図 3.12: 空間データ表現のための DTD

```
<spatial-data>
  <line>
    <x>-50997.8</x><y>-16011.8</y>
    <x>-50996.8</x><y>-16008.8</y>
  </line>
  <line>...</line>
  ...
</spatial-data>
```

図 3.13: XML による空間データの記述例

3.3.5　地図描画アプリケーション

3.3.1 節の空間データ提供システムをサーバとするクライアントとして，地図描画プログラムを作成した．この地図描画クライアントは，起動時にサーバとの TCP コネクションを確立し，そのコネクションを介してサーバに対してデータ要求を行い，サーバからの検索結果を受信する．描画クライアントは，XML によって表現された検索結果から空間データを抽出し，そのデータに基づき地図の描画を行う．なお，XML パーサとしては，SAX (Simple API for XML) を利用した．

3.4 実験および評価

3.4.1 実験環境

空間データ提供システムと地図描画クライアントはそれぞれ，表 3.1 に示すホスト (Sun Ultra-10 ワークステーション) 上で動作させた．また，各ホストは 100 Mbps の Ethernet で接続されている．

表 3.1: 測定環境

	CPU	メモリ
サーバ	440 MHz	512 MB
クライアント	333 MHz	128 MB

また,「数値地図 2500(空間データ基盤)」[JMC] より作成した表 3.2 のデータセットを実験に用いた．

表 3.2: 実験に用いたデータセット

	種類	オブジェクト数	ポイント数
$D_{line,1}$	線データ	26745	53490
$D_{pol,1}$	ポリゴンデータ	2452	115959

$D_{line,1}$ は 4 個の図葉から構成され，そのインデックスが覆う矩形領域 R_{index} は，$\langle -52500, -18000, -49500, -14000 \rangle$ である．$D_{pol,1}$ は 140 個の図葉から構成され，その R_{index} は $\langle -63000, -36000, -39000, -2000 \rangle$ である．図 3.14 に，$D_{line,1}$，$D_{pol,1}$ の覆う領域を示す．以降の実験では，問合せ領域，分割幅，分割の基準点の単位はメートル (m) とする．

なお，R-tree のパラメータについては，$m = 10$，$M = 20$ とした．

3.4.2 処理時間の測定

地図描画クライアントが，領域分割のパラメータを指定せずに，空間データサーバに対してデータを要求してから地図の描画を終えるまでの応答時間 (t_{wait}) と，その内訳について調べる．t_{wait} には，データ探索時間 (t_{search})，検索結果の XML タグ付けにかかる時間 (t_{tag})，受信した XML データの解析時間 (t_{parse}) と地図の描画にかかる時間 (t_{draw}) が含まれる．残りの時間 (t_{etc}) には，データ転送時間が含まれる．これらの処理時間の測定は，本手法の有効性を議論していく上で必要な基本的

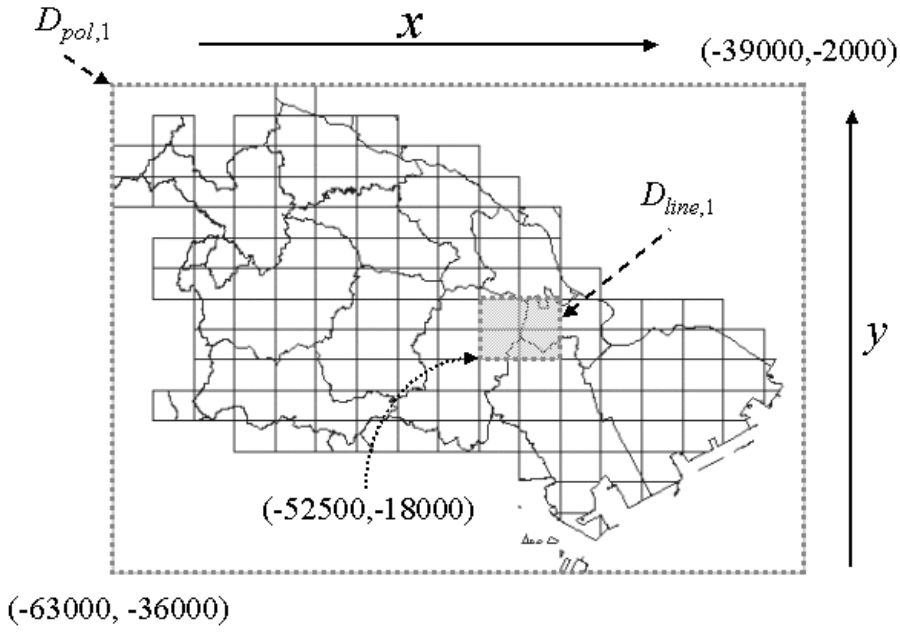

図 3.14: 実験に用いたデータセットの覆う領域

評価である．なお，時間の測定には，java.lang.System.currentTimeMillis() メソッドを利用した．

本実験では，クライアントが距離に基づくクエリ Q_{circle} をデータサーバに対して依頼した時の処理時間を調べた．サーバが $D_{line,1}$ に含まれる線データを管理する時の測定結果を図 3.15 に，$D_{pol,1}$ に含まれるポリゴンデータを管理する時の結果を図 3.16 に示す．

線データおよびポリゴンデータの場合ともに，問合せ領域の半径の増加に伴い，収集するデータオブジェクト数が多くなるため，クエリに対する応答時間は増加する．応答時間の中では探索時間 (t_{search}) の占める割合が最も大きい．これはデータ探索そのものを制御することが重要であり，すべての探索を終了してから検索結果を分割して逐次的に提供することが効果的でないことを意味する．また，t_{parse} や t_{tag} などの XML データの処理時間の割合も小さくなく，その処理時間は，収集したデータオブジェクト数に依存する．すなわち，分割領域が大きくなると，それに伴い検索結果に含まれるオブジェクト数が多くなるので，その検索結果に対する XML データの処理時間は増加することになる．

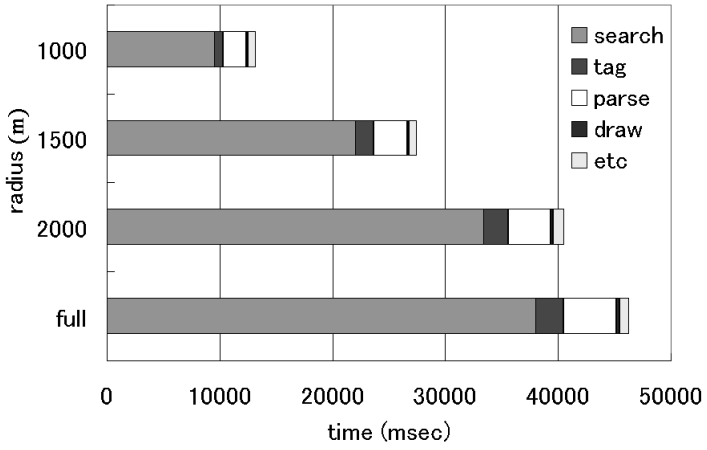

図 3.15: $D_{line,1}$ を管理するサーバにデータを要求した時の処理時間の割合

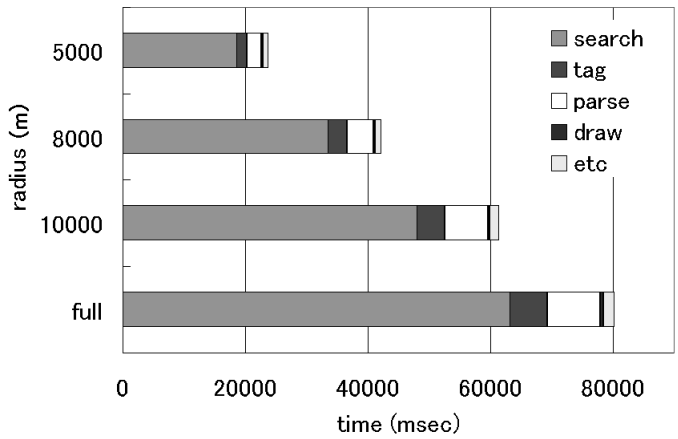

図 3.16: $D_{pol,1}$ を管理するサーバにデータを要求した時の処理時間の割合

3.4.3　検索結果の品質の時間変化

検索結果を分割し逐次的に転送した時の検索結果の品質の変化を調べる．本実験では，$D_{line,1}$ を管理するサーバに対して，クライアントが問合せ領域 $R_{query,rect} = \langle -52500, -17500, -49500, -14500 \rangle$ を持つ矩形問合せ Q_{rect} を送信する．そして，クエリを送信した時刻から，各受信パケットに基づき地図の描画を終了した時刻を測定し，式 (3.1) より，その時の収集データオブジェクト数を用いて検索結果の品質 $q(t)$ を計算した．

まず，領域分割に基づく探索を実行した場合の結果を図 3.17 に示す．領域分割のパラメータについては，$(x_{base}, y_{base}) = (-51000, -16000)$，$dx_+ = dy_+ = dx, dx_- = dy_- = -dx$ として，dx を変化させた．

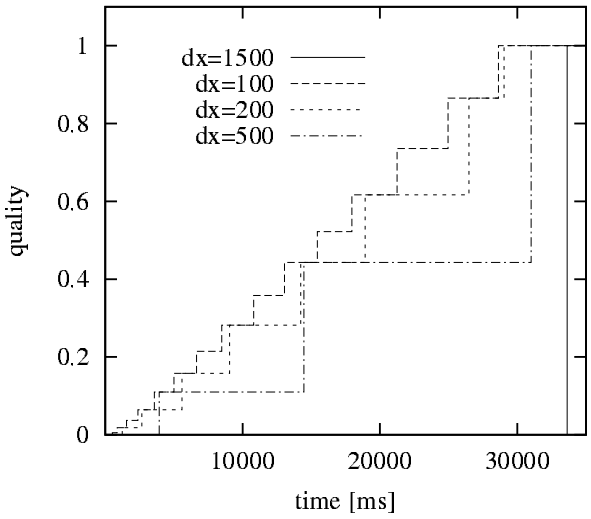

図 3.17: 地図描画クライアントにおける検索結果品質 $q(t)$ の時間変化 (領域分割を用いた場合)

図 3.17 より，検索結果品質 $q(t)$ が時間とともに増加し，領域分割をせず検索結果を一括転送した時 $(dx = 1500)$ より早い時刻に何らかの結果を取得できていることがわかる．分割幅 dx が小さいほど，分割数は多くなり，各分割領域は小さくなる．それに伴い，最初に探索する分割領域の大きさが小さくなるため，最初の応答時間は短くなる．

図 3.17 では，領域分割を行わない場合 $(dx = 1500)$ と比べて，領域分割を行った場合 $(dx = \{100, 200, 500\})$ の全応答時間が短く，分割幅 (dx) が小さいほど，その応答時間が短くなる傾向が見られる．

ここで，サーバ側におけるデータ探索結果の品質 $q_{search}(t)$ を，式 (3.12) のように定義する．

$$q_{search}(t) = \frac{N_{search}(t)}{N_{search,all}} \tag{3.12}$$

$N_{search}(t)$ は，時刻 t までにサーバが収集したデータオブジェクトの個数である．$N_{search,all}$ は，サーバにおける全探索の結果に含まれるデータオブジェクトの総数である．各分割領域に対する探索が終了した時刻 (t) と，それまでの探索結果に含まれるデータオブジェクト数の総和を記録することによって，$q_{search}(t)$ は計算できる．

図 3.17 は，クライアント側での各検索結果受信時の地図描画が終わる時刻に基づいて，式 (3.1) より計算されており，ユーザに対する応答時間の変化を示している．この時のサーバ側での探索結果の品質 $q_{search}(t)$ の時間変化を示す．図 3.17 は，3.4.2 節で言及した各処理時間が影響するが，図 3.18 はサーバでの探索時間のみが影響する．

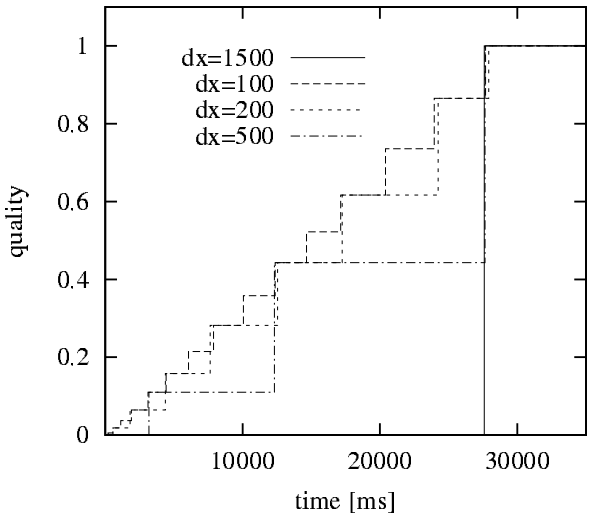

図 3.18: 空間データサーバにおける探索結果の品質 $q_{search}(t)$ の時間変化

図 3.18 では，分割幅 (dx) が小さいほど最初の探索結果が出力される時間は短いが，すべての探索を終了する時間に，ほとんど差は見られない．すなわち，図 3.17 と図 3.18 より，分割幅が大きいほど，最後の領域の探索を終了してから，その領域に対する空間データに基づいて描画を終了するまでの時間が長くなっている傾向が見られる．分割幅が大きいほど，最後に転送されるパケットに含まれるデータオブジェクト数が大きくなるので，そのデータの XML 処理にかかる時間は大きくなる．全応答時間は最後に転送されるデータオブジェクトの数に依存するので，本実験においては，図 3.18 に見られるように，探索のオーバヘッドが小さいために，

分割幅が小さいほど全応答時間が短くなる．この探索のオーバヘッドについては，3.4.5 節で述べる．

さらに，領域分割を用いた場合 (図 3.17) との比較として，次のそれぞれの場合について調べる．

- 領域分割を行わずに，オブジェクト数 (max_item) に基づく分割を行う場合
- 領域分割とオブジェクト数による分割を併用する場合

オブジェクト数のみに基づいて分割を行う場合

サーバが領域分割を行わずに探索を行い，指定したオブジェクト数 max_item (図中では，**max**) ごとに，探索結果をバッファから取り出し，取り出した結果をパケットとして順番に転送した場合の検索結果品質 $q(t)$ の変化を図 3.19 に示す．

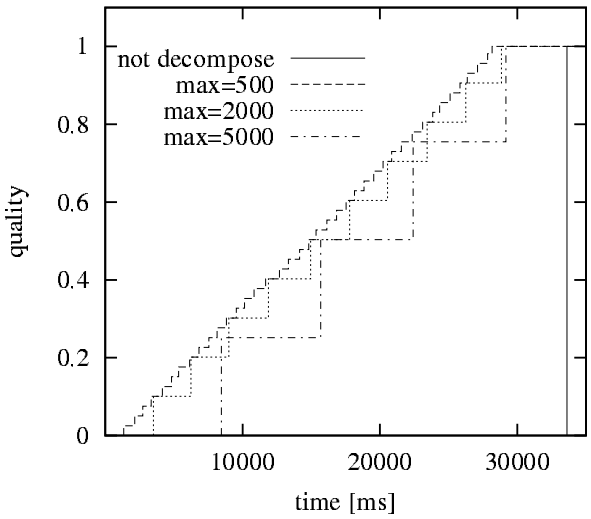

図 3.19: クライアントにおける検索結果品質 $q(t)$ の時間変化 (領域分割を行わずに，オブジェクト数のみを変化させた場合)

図 3.19 より，領域分割を行った場合と同様に，max_item に基づくパケット分割によって，検索結果品質が時間とともに増加し，max_item が小さいほど，最初の結果を取得する時刻は早く，パケットの受信間隔は短くなり，検索結果の品質はなめらかに変化する．

領域分割を用いた場合 (図 3.17) は，各分割領域に対する探索の終了後に，その探索結果が転送される．ただし，その探索の終了とともに，次の分割領域に対する探索は開始され，探索結果のタグ付けや転送処理は次の領域に対する探索と並行に実行される．

それに対して，max_item に基づくパケット分割を用いた場合 (図 3.19) は，探索実行中に結果を随時転送できる．その際の全応答時間は転送データのサイズ，すなわち max_item の大きさに依存するので，図 3.19 では，max_item の値が小さいほど，全応答時間が短くなる傾向が見られる．当然のことながら，パケット分割のみを用いる場合は，領域分割による探索のオーバヘッドを伴わないので，領域分割に基づく探索を行う場合よりも，全応答時間を短くすることができる．

領域分割とオブジェクト数による分割を併用する場合

領域分割を行う場合でも，max_item を指定することによって，各分割領域に対する探索結果を更に細かく分割することができる．そこで，サーバに $Q_{circle}(radius = 1500)$ を送信し，dx= {100, 500, 1500} を用いて領域分割を行い，**max**= {5000, 2000, 500} に基づきパケット分割を行った場合の全応答時間を図 3.20 に示す．$dx = 1500$ の時は領域分割は行わず，**max**=30000 の時は，パケット分割は行われない．

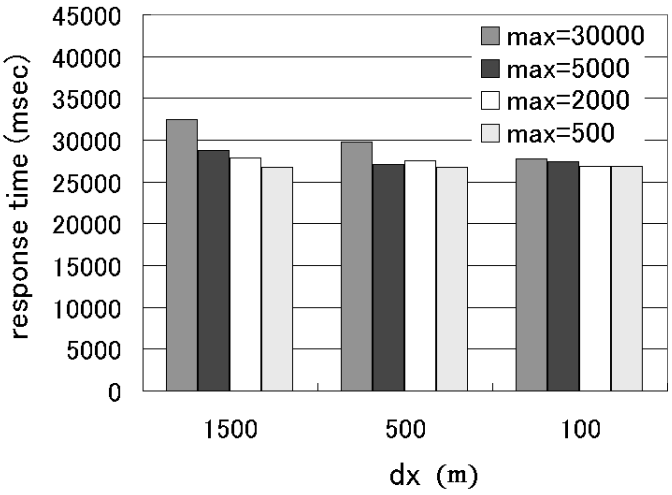

図 3.20: 領域分割とオブジェクト数による分割を併用した場合の応答時間

図 3.20 より，領域分割を行った場合 ($dx = \{100,500\}$) においても，max_item を指定することによって全応答時間が減少することがわかる．ただし，分割幅が小さい場合には，その領域分割の効果により応答時間が小さくなっているため，max_item の効果は小さいと考えられる．実際に，各分割領域に対する結果に含まれるデータオブジェクトの数が max_item より小さい場合には，max_item に基づく検索結果の分割は行われない．また，max_item をさらに小さくした場合には，サーバからクライアントに転送されるパケットの個数が非常に多くなり，通信のオーバヘッドが大きくなると考えられる．

3.4.4　地図の描画

3.4.3 節で述べたように，領域分割に基づく探索を行った場合 (図 3.17) とパケット分割を用いた場合 (図 3.19) ともに，検索結果品質は同様の変化を示す．しかし，パケット分割を用いた場合と，領域分割を用いた場合とでは，検索結果品質の変化が同じであっても，そのデータ収集領域に差があり，クライアント側での地図の描画という点で大きく異なる．本節では，その地図描画について述べる．

本実験においても $D_{line,1}$ に含まれる線データを対象とした．問合せ領域 $R_{query,circle} = \langle -51000, -16000, 1500 \rangle$ に対して，領域分割 ($d_{radius} = 100$) を行った場合の，描画クライアントにおける途中結果の出力を図 3.21 と図 3.22 に，全データ取得後の出力を図 3.23 に示す．

図 3.21: 領域分割を用いた場合の途中結果に基づく地図の描画 (時刻 t = 1556 ミリ秒)

図 3.22: 領域分割を用いた場合の途中結果に基づく地図の描画 (時刻 t = 8484 ミリ秒)

図 3.21，図 3.22，図 3.23 より，円 ($R_{query,circle}$) の中心を基準として，時間とともにデータ収集領域が拡大していく様子が観察できる．これに対して，領域分割を

図 3.23: 全データ取得後に描画された地図 (時刻 t = 28622 ミリ秒)

行わず，オブジェクト数に基づき検索結果を分割し，逐次的に転送した場合の出力を図 3.24 に示す.

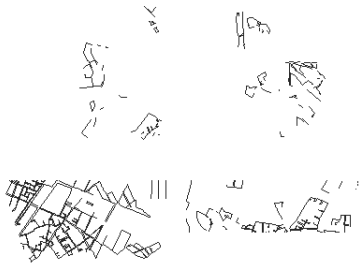

図 3.24: オブジェクト数に基づく分割を行った場合の途中結果に基づく地図描画

いずれの場合も，データが到着する度にインクリメンタルに地図情報が描画され，ユーザは検索結果を一括転送する場合より早い時刻に，地図の断片を見ることができる．しかし，オブジェクト数に基づいて検索結果を分割する場合には，ユーザの視点からすればランダムに地図データが描画される (実際には，その描画順序は空間インデックスの構造と探索ポリシーに依存する)．それに対して，領域分割に基づく探索を行った場合には，ユーザの指定した点を中心にして放射状に地図が描画されていく.

さらに，$D_{pol,1}$(ポリゴンデータ) を管理するサーバに対して，クライアントが，問合せ領域として $R_{query,rect} = \langle -54000, -29000, -45000, -19000 \rangle$ を指定した時の描画結果を 図 3.25 に示す．図 3.25 では，$R_{query,rect}$ と重なり合うすべてのポリゴンデータが表示されている.

図 3.25 と同様の問合せ領域に対して，領域分割の基準点 $(-54000, -29000)$ と

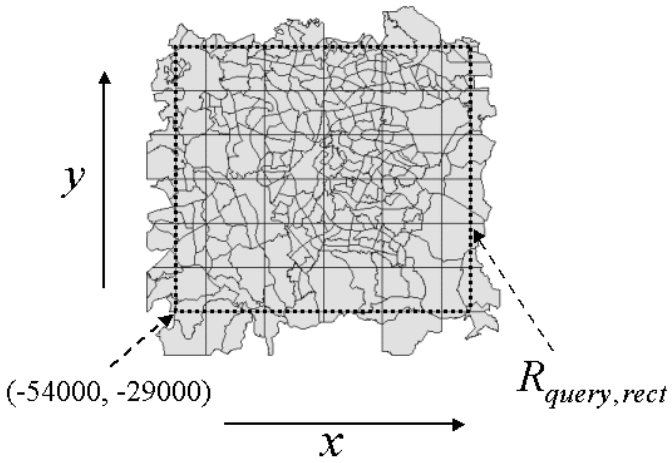

図 3.25: 問合せ領域と検索結果であるポリゴンデータ

分割幅 (dx_+,dy_+) を指定した時の検索の途中結果に基づく描画例を図 3.26 の左図, 右図に示す.

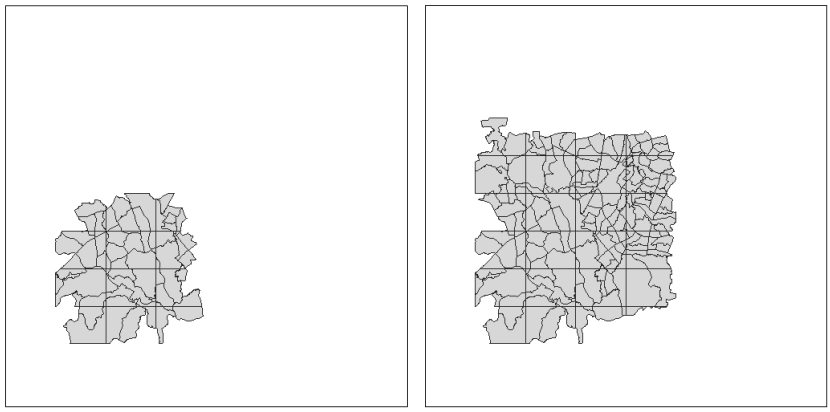

図 3.26: 途中結果に基づくポリゴンデータの描画 (時間とともに左図から右図へデータ収集領域が拡大)

この実験においても，図 3.26 左図 → 図 3.26 右図 → 図 3.25 に示すように，問合せ領域の左下頂点 $(-54000, -29000)$ を基準にして，データ収集領域が拡大していく様子が観察された．分割幅として dx_+ のみを指定すれば x 軸の正方向に領域を拡大させていくことができる．このように，条件 (3.5)，(3.6) を満たす分割ポリシー (3.11)，(3.10) を用いて，様々な領域分割と順序付けが可能であり，ユーザの関心のある領域内の空間データを優先的に処理したい場合には，領域分割に基づく方法が有効であると考えられる．

空間データ検索に対する途中結果が領域ごとに提供されれば，その取得した情報に基づいてインクリメンタルに空間情報処理を行うことができ，さらに，その途中結果に基づきインタラクティブに別の空間データを要求することも可能である．本手法による領域を考慮した空間データの逐次的な提供は，リアルタイムかつ柔軟な空間情報処理を行う上で有用であると考えられる．

また，本実験では，サーバから提供される空間データに対して，クライアントはXML データの文字列解析を行い，その抽出結果に基づいて地図の描画を行っているが，その XML データの処理時間は小さくなく，それが応答時間に影響していることがわかる．つまり，受信したデータに対する処理時間が大きい場合，探索終了後にすべての結果を提供すると，そのデータ処理を終了するまでの時間が非常に長くなる可能性がある．GIS アプリケーションにおいて，情報統合やマイニングといった処理時間の大きい空間情報処理を逐次的に行う場合には，本手法によるインクリメンタルなデータ提供が有効であると考える．

3.4.5 領域分割によるオーバヘッド

分割によるオーバヘッドについて議論するために，領域分割に基づく探索を実行した時のオーバヘッドについて考察する．この節では，表 3.3 に示すデータを用いて実験を行う．

表 3.3: 実験に用いたデータセット

	種類	オブジェクト数	図葉数
$D_{line,2}$	線データ	29125	4
$D_{pol,1}$	ポリゴンデータ	2452	140
$D_{pol,2}$	ポリゴンデータ	43613	140

まず，線データ $D_{line,2}$ を管理するサーバに対して，$Q_{circle}(radius = 1500)$ を送信し，$d_{radius} = \{20, 50, 100, 200, 500\}$ を用いて領域分割を行った場合の，全応答時間および探索コスト (総探索時間および総ディスクアクセス回数) の測定結果を，それぞれ図 3.27，図 3.28 に示す．ここでは，3.2.4 節で述べた拡張した探索アルゴリズム (**extend**) を，探索のために複数のキューを用いない拡張前の探索アルゴリズム (**basic**) と比較する．$d_{radius} = 1500$ は，領域分割を行わない場合に相当する．図図 3.28 の **time** は，総探索時間 (単位：msec) を示す．**access** は，総ディスクアクセス回数を示し，分割を行わない場合のディスクアクセス回数に対する比率 (rate) として表す．

図 3.28 より，総探索時間がディスクアクセス回数に依存し，ディスクアクセス回数が多くなるにつれて総探索時間が増加することがわかる．ディスクアクセス回数

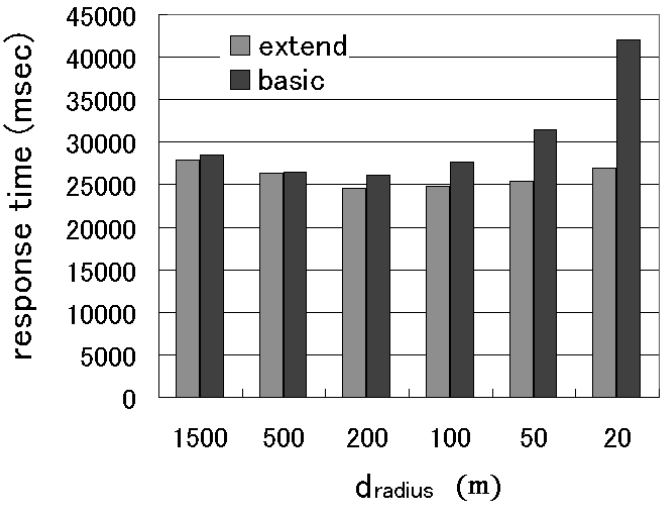

図 3.27: $D_{line,2}$ を管理するサーバにデータを要求した時の全応答時間

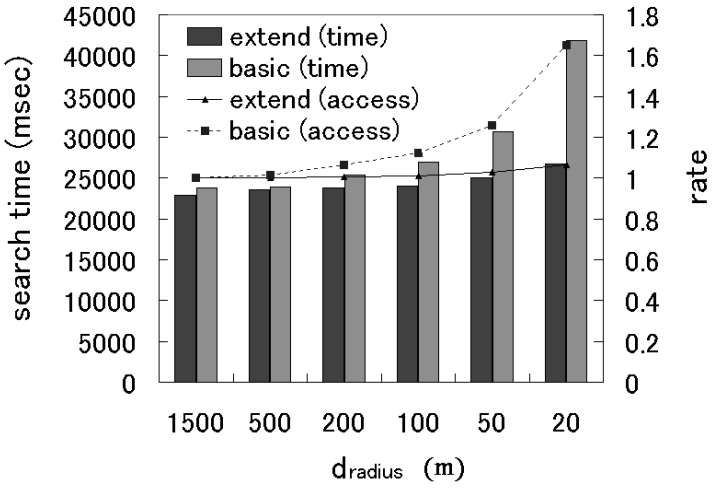

図 3.28: $D_{line,2}$ を管理するサーバにデータを要求した時の探索コスト (総探索時間および総ディスクアクセス回数)

は **basic** より **extend** を用いた場合が小さいことがわかる．これは，**extend** では，探索時に複数のキューを用いて探索を行うことにより，既に探索した領域に含まれる空間データオブジェクトを調べる必要がないためと考える．さらに，**extend** の場合は，領域分割によるディスクアクセス回数の増加の度合は小さく，総探索時間の増加も小さい．

図 3.27 と図 3.28 より，探索時間 (search time) の増加の度合に伴い応答時間 (response time) が変化していることがわかる．また，**extend** を用いた場合の応答時間は，分割幅が大きくなるにつれて，減少した後，増加する傾向が見られるが，その増加の割合は非常に小さい．応答時間が減少する理由は，応答時間が最後のパケットに含まれるオブジェクト数に依存するためであるが，応答時間の増加については探索コストの増加率が影響していると考えられる．

次に，表 3.3 に示した $D_{pol,1}$ および $D_{pol,2}$ に対して，**extend** を用いた場合について，それぞれ応答時間と探索コスト (探索時間およびディスクアクセス回数) を調べた．サーバに対して，$Q_{circle}(radius = 2000)$ を送信した時の実験結果を，図 3.29 と図 3.30 に示す．図中の **response** と **search** はそれぞれ，全応答時間 (単位：msec) と総探索時間 (単位：msec) を示す．**access** は，総ディスクアクセス回数を示し，分割しない場合のディスクアクセス回数に対する比率 (rate) として表す．

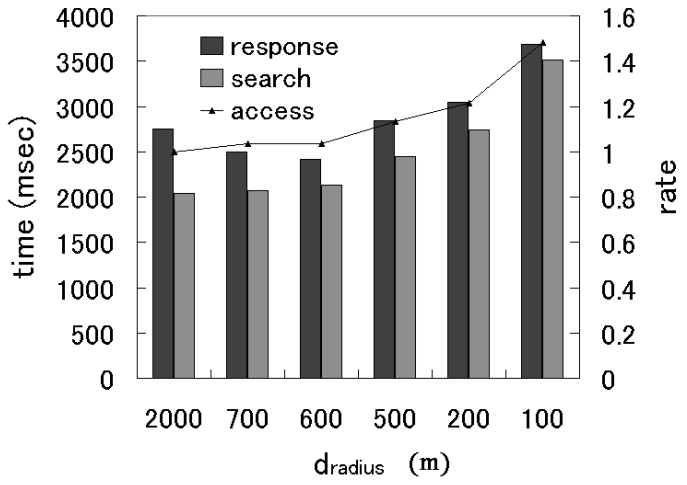

図 3.29: $D_{pol,1}$ を管理するサーバにデータを要求した時の全応答時間，総探索時間および総ディスクアクセス回数 (**extend** を用いた場合)

図 3.28 と同様に，分割幅 d_{radius} が小さくなる，すなわち分割数が多くなるにつれて，総ディスクアクセス回数は増加し，それに伴い総探索時間が増加することがわかる．全応答時間についても，領域分割によって一度は減少するが，さらに分割数が多くなると増加する傾向が見られる．その増加率は，図 3.27 より，図 3.29 ／図 3.30 の方が大きい．図 3.28，図 3.29 および図 3.30 より，応答時間の増加は総

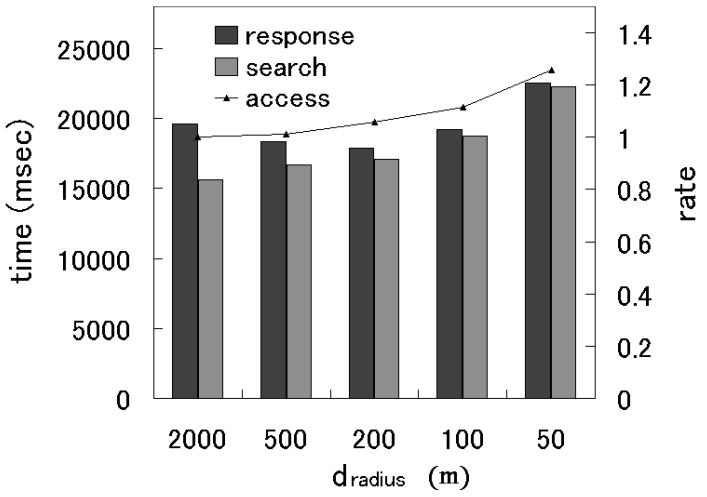

図 3.30: $D_{pol,2}$ を管理するサーバにデータを要求した時の全応答時間，総探索時間および総ディスクアクセス回数 (**extend** を用いた場合)

ディスクアクセス回数の増加の割合が影響していると考える．図 3.28 では，ディスクアクセス回数の増加率が非常に小さいのに対して，図 3.29 と図 3.30 では，ディスクアクセス回数の増加率が大きい．結果として，領域分割を行った時の総探索時間が分割を行わない時の全応答時間を上回る場合が生じる．

$D_{line,2}$，$D_{pol,1}$ および $D_{pol,2}$ に対する各結果に対して，ディスクアクセス回数の増加率が異なるのは，分割幅 (d_{radius}) に対する空間データオブジェクトの大きさが影響していると考える．そこで，各データセットに含まれる空間データオブジェクトの包囲矩形 (MBR) の大きさについて調べた．MBR は，各データオブジェクトの近似図形であるので，各データオブジェクトの大きさを反映していると考える．MBR の大きさを表す指標として，各 MBR の対角線および外周の長さを計算し，平均値を算出した．各データセットに対する平均値を表 3.4 に示す．

表 3.4: MBR の大きさ (平均値)

	対角線の長さ (m)	外周の長さ (m)
$D_{line,2}$	16.4	41.2
$D_{pol,1}$	661.9	1806.4
$D_{pol,2}$	131.1	359.5

図 3.29 および図 3.30 より，表 3.4 に示した MBR の対角線の長さと比べて，分割幅が小さい場合に応答時間は増加する傾向が見られ，MBR の対角線が長い $D_{pol,1}$(図 3.29) の方が探索時間および応答時間の増加率が大きいことがわかる．それに対し

て，$D_{line,2}$(図 3.27, 図 3.28) では MBR の対角線が短いので，探索のオーバヘッドが小さく，それに伴い応答時間の増加率も小さいことがわかる．

以上の結果より，領域分割に基づく探索では，空間データオブジェクトに対する分割幅の大きさが，探索のオーバヘッドに大きく影響することがわかる．オブジェクトの大きさが分割幅より大きい場合には，そのオブジェクトが複数の分割領域と重なる可能性が高く，ディスクアクセス回数の増加をもたらすものと考える．逆に，オブジェクトの大きさが分割幅より小さい場合には，オブジェクトが複数の領域にまたがる可能性は低くなるため，ディスクアクセス回数の増加率は低いと考える．

さらに，クエリに指定された検索範囲の大きさを変化させた場合の探索コストについて測定した．$D_{pol,1}$ および $D_{pol,2}$ を管理するサーバに対して，$radius$ をパラメータとするクエリ Q_{circle} を送信した時の測定結果を図 3.31，図 3.32 に示す．図 3.31，図 3.32 において，$d_{radius} = radius$ の場合は，領域分割を行わない場合を示す．また，各クエリに指定された検索範囲の大きさを表 3.5 に示す．$D_{pol,1}$ と $D_{pol,2}$ は 140 個の図葉から構成されるデータセット (図 3.14) であるので，問合せ範囲の大きさは，これらのデータセットが覆う全領域に対する面積比として計算できる．

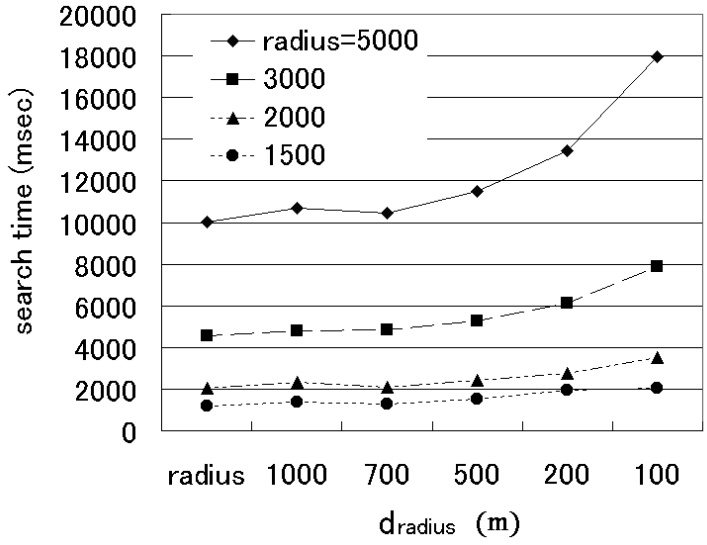

図 3.31: $D_{pol,1}$ を管理するサーバにデータを要求した時の探索時間

図 3.31，図 3.32 では，問合せ領域 ($radius$) が大きいほど探索時間は大きく，各クエリに対する探索時間は，分割幅 (d_{radius}) が小さくなるにつれて大きくなる．そして，探索時間の増加率は，図 3.29，図 3.30 で見たように，空間データオブジェクトの大きさに影響されることがわかる．分割幅がデータオブジェクトと比較して大きい場合には，問合せ範囲の大きさに関わらず，探索コストの増加率が非常に小さい．一方で，分割幅がデータオブジェクトより小さい場合には，問合せ範囲が大

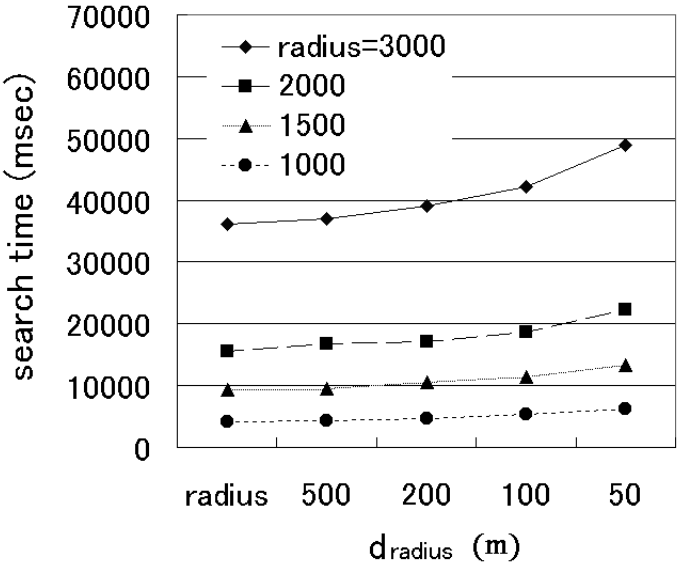

図 3.32: $D_{pol,2}$ を管理するサーバにデータを要求した時の探索時間

表 3.5: 問合せ範囲の大きさ

radius (m)	範囲面積 (%)
5000	18.6
3000	6.7
2000	2.9
1500	1.6
1000	0.7

きいほど分割数が多くなり，さらに，探索のオーバヘッドが大きいため，探索コストの増加率は高くなる．

本節では，全応答時間と探索時間という観点から，領域分割によるオーバヘッドについて考察した．本手法による領域分割は，探索のオーバヘッドをもたらすが，3.4.3 節，3.4.4 節で述べた利点を踏まえれば，十分に有用であると考える．

3.5 まとめ

本章では，ネットワークを介した空間データ検索のための領域分割に基づくインクリメンタルな空間データ提供手法について述べた．本手法は，空間範囲問合せに対して，問合せ領域の分割を行い，各分割領域ごとに探索結果を保持し，その探索結果を中間結果としてクライアントにインクリメンタルに提供する．地図描画アプリケーションを用いた実験の結果，本手法は領域分割に基づく探索と探索結果の逐次提供によってユーザに対して効果的に地図を描画できることを確認した．この領域分割に基づくインクリメンタルなデータ提供は，地図表示だけでなく，位置情報に基づく統合処理や空間情報処理を行うアプリケーションに対しても有用であると考える．

第4章

センサデータベース検索のためのインクリメンタルなデータ提供手法

本章では，まず，異種のセンサデータを提供する複数のデータベースが存在する環境において，3章で提案した領域分割に基づく空間データ検索手法を拡張し，インクリメンタルにセンサデータを提供する方式を提案する．本方式は，仲介エージェントとセンサデータベースのラッパーとなるセンサデータサーバから構成され，クエリに指定した問合せ領域と時区間を分割し，その分割に基づく順序で検索結果をインクリメンタルに提供する．

4.1 Anytime algorithm に基づくセンサデータ検索手法

ネットワーク上の複数のセンサデータベースから時系列のセンサデータを収集し，その統合結果を表示する場合に，応答時間の増大が予想される．位置情報システムでのセンサデータの利用を想定すると，空間データ検索の場合と同様に，領域や位置情報を考慮することによって効果的なデータ提供が期待できる．たとえば，観光中の移動ユーザにとって，近隣の交通機関の運行状況，道路の渋滞情報，観光地の混雑状況に関するセンサデータは，自分の位置，目的地あるいは移動経路に近い領域から優先的に取得したい情報である．また，統合結果がユーザの興味のある範囲から順番に表示されれば，ユーザは，その表示結果に基づいて，検索範囲の拡大や新たな領域に対するデータ要求を行うことが可能である．途中結果が “領域” という意味のあるまとまりで提供されることにより，インタラクティブにセンサデータを閲覧することができると考える．

本章では，インクリメンタルな検索結果の提供という点に着目し，3章で提案した領域分割に基づく手法を，次の方針に従って，センサデータベース検索に適用

する．

- 異種センサネットワーク環境では，単一のデータベースからデータを取得するのではなく，ネットワーク上の複数のデータベースからデータを収集できる検索システムが必要である．データ参照方法や検索結果提供方式の違いなどのデータベースの異種性を吸収するために，本章では，検索システムを，2.2.2 節で説明した仲介エージェント（mediator）とラッパー（wrapper）から構成される連邦データベースシステムとして構成する．さらに，設置領域やノード分布の異なるセンサネットワークからのセンサデータに対するシームレスな検索を実現するために位置指向検索をサポートさせる．その上で，領域分割の概念を分散データベース検索に拡張する．
- 各データベースは，空間データではなく，位置属性を持つ時系列データを管理する．センサデータのような時系列データに対して統合処理を行い，その結果を表示する場合には，時間順序が重要になると考えられる．そこで，空間情報だけでなく，時間情報も考慮して検索結果を分割する．具体的には，領域分割に加えて，時区間分割を導入し，これらの分割を指定するパラメータを導入することで，センサデータ統合の結果の提示順序を制御する．

4.2　センサデータ検索システム

4.2.1　システム構成

センサデータ検索システムの構成を図 4.1 に示す．

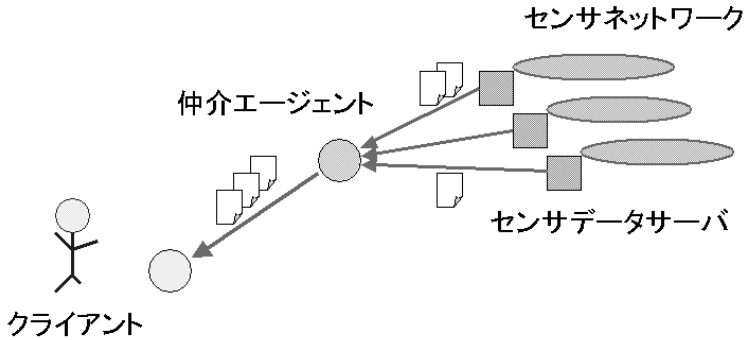

図 4.1: センサデータ検索システムの構成

センサデータサーバ

センサデータサーバは，各センサデータベースのラッパーとして機能する．センサネットワークはデータの種類ごとに構成され，各センサネットワークごとにセンサデータベースが存在するものとする．センサデータサーバは，対応するセンサネットワークの各センサノードの位置情報を保持するためのテーブル（location table）と，各ノードから提供されるセンサデータを時系列として管理するための時系列データテーブルを持つ．時系列データテーブルは，表 4.1 に示す属性を持つ．

表 4.1: センサデータサーバの持つ時系列データテーブル

フィールド名	説明
$NODE\_ID$	センサノードの識別子
$DATA\_TYPE$	ノードの提供するデータの種類
$VALUE$	センサの観測値
$UPDATE\_TIME$	センサデータの更新時刻

仲介エージェント

仲介エージェントは，クライアントからのクエリを適切なサーバに転送し，各サーバからの検索結果をクライアントに転送する役割を持つ．仲介エージェントは，各センサデータサーバに対するセンサネットワークに関する情報を管理し，この情報に基づいて，クエリに指定された条件を満たすデータを提供しうるセンサデータサーバを探し出す．

センサネットワークに関する情報として，次の 2 つの属性情報が，仲介エージェントによって保持される．

- :data-type 属性
  センサデータサーバが提供するデータの種類を示す．

- :area 属性
  センサデータサーバが管理するセンサネットワークの設置場所に関する情報（以下，センサネットワークの位置情報と呼ぶ）を表す．本論文では，センサネットワークの位置情報（SNR: Sensor Network Region）を，各センサネットワークを構成するすべてのセンサノードを含む領域として定義する．

本検索システムにおいては，センサノードの位置を 2 次元座標 (x, y) で表現し，SNR を，式 (4.1) のように，点 (x_{min}, y_{min}) と点 (x_{max}, y_{max}) によって定まる矩形領域として表現する（図 4.2）．

$$SNR = \langle x_{min}, y_{min}, x_{max}, y_{max} \rangle \tag{4.1}$$

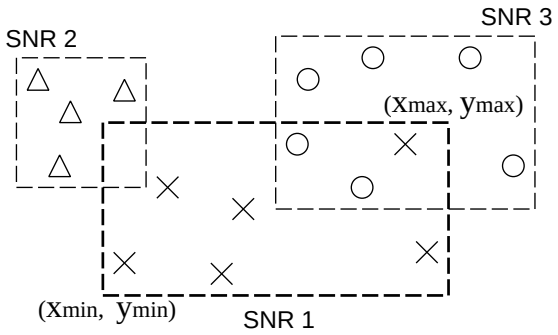

図 4.2: センサネットワークの位置情報

4.2.2　クエリの表現と分割

クライアントは，センサデータを収集するために，式 (4.2) で表現されるクエリを仲介エージェントに送信する．

$$Q \quad = \quad \langle A, DT, TI, SC, TC \rangle \tag{4.2}$$

- A は，問合せ領域を表す．2 次元の矩形や円などの領域表現が記述される．
- DT は，クライアントが要求するセンサデータの種類から成る集合である．
- $TI = [t_{min}, t_{max}]$ は，下限値（t_{min}）および 上限値（t_{max}） によって定まる時区間表現である．
- SC（Spatial Constraints：空間制約）は，問合せ領域（A）を分割するためのパラメータである．
- TC（Temporal Constraints：時間制約）は，時区間（TI）を分割するためのパラメータである．

問合せ領域 A および時区間 TI を指定することにより，ある領域の任意の時区間の時系列センサデータを取得できる．

空間制約（SC）によって分割された領域を順序付け，時間制約（TC）によって分割された時区間を順序付けることで，検索結果の転送順序が決定される．TC と SC を指定することで，クエリ Q はサブクエリに分割される．

$$Q = \langle q_{1,1}, q_{1,2}, ..., q_{2,1}, ... \rangle \tag{4.3}$$
$$q_{l,i} = \langle a_{l,i}, DT, TS_l \rangle \tag{4.4}$$

$a_{l,i}$ はサブクエリ $q_{l,i}$ に対する問合せ領域であり，TS_l は $q_{l,i}$ に対する時区間パラメータである．

まず，問合せ領域の分割に関しては，基本的に，3 章で空間データ検索に利用した分割方法に従う．

$a_{l,i}$ は，式 (4.6) の関係を満たす 2 つの領域表現（$a_{l,i}^{upper}$，$a_{l,i}^{lower}$）の差分領域として，式 (4.5) のように表現される（図 4.3：左図）．

$$a_{l,i} = \begin{cases} a_{l,i}^{upper} - a_{l,i}^{lower} & (i > 1) \\ a_{l,i}^{upper} & (i = 1) \end{cases} \tag{4.5}$$

$$a_{l,i}^{lower} \subset a_{l,i}^{upper} \ , \ a_{l,i}^{lower} = a_{l,i-1}^{upper} \tag{4.6}$$

問合せ領域の分割は，その領域表現に依存するが，センサデータ検索に対しても空間データ検索の場合と同様の以下の 2 種類の分割方法を用いる．

- 問合せ領域が円である時，円の中心を基準点として半径に沿って，その領域を分割する．円の中心座標と半径方向の分割幅を空間制約として指定する．
- 問合せ領域が矩形である時，分割の基準点と各座標軸方向の拡大幅を指定する（図 4.3：右図）．

空間制約（SC）として，分割の基準点とデータ収集領域の拡大方向を指定することで，ユーザの注目している領域から優先的にデータを取得することが可能となる．

次に，時区間分割についても，式 (4.7)，式 (4.8) に示すように，2 つの時区間（TS_l^{upper}, TS_l^{lower}）の差分として分割時区間 TS_l を定義する（図 4.4）．

$$TS_l = \begin{cases} TS_l^{upper} - TS_l^{lower} & (l > 1) \\ TS_l^{upper} & (l = 1) \end{cases} \tag{4.7}$$

$$TS_l^{lower} \subset TS_l^{upper} \ , \ TS_l^{lower} = TS_{l-1}^{upper} \tag{4.8}$$

時間制約（TC）として，図 4.4 に示すように，分割の基準点（t_{base}）と各時間軸方向の拡大幅（t_+，t_-）を指定し，式 (4.9) を用いて時区間 TS_l^{upper} を計算する．

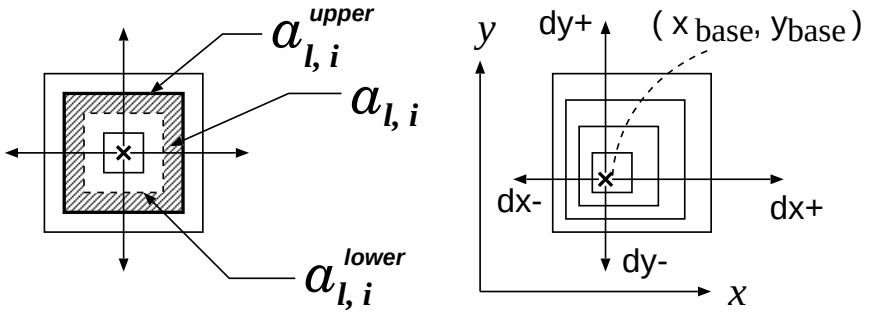

図 4.3: 問合せ領域の分割と空間制約

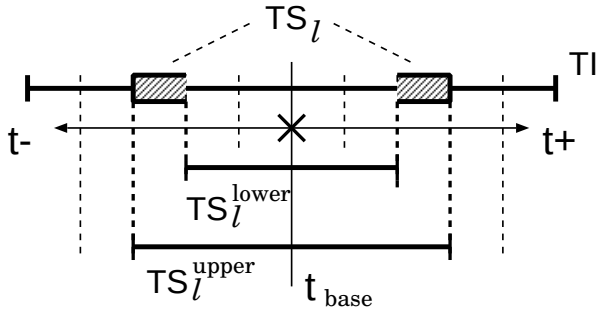

図 4.4: 時間制約に基づく区間分割

$$TS_l^{upper} = [t_{base} + t_- \times l, t_{base} + t_+ \times l] \quad (t_+ \geq 0, t_- \leq 0) \tag{4.9}$$

時区間分割を用いる場合には，優先的に情報を取得したい時刻を t_{base} として指定し，情報の提供順序を決定するために，拡大幅 t_+, t_- を指定する．たとえば，基準時刻 t_{base} として，$TI.t_{min}$ を指定し，拡大幅として，$t_+(>0), t_- = 0$ を指定することで，時刻の古いデータから新しいデータという時間順序で検索結果を提示していくことができる．$t_-(<0), t_+ = 0$ を指定した場合には，現在から過去にさかのぼった逆方向の情報提示が可能である．また，基準時刻 t_{base} が必ずしもクエリに指定した時区間 TI の始点や終点とは限らないので，そのような場合には，t_+ と t_- の両方を指定する必要がある．この場合，ある時間帯を中心とした両方向への情報提供になる．

なお，本論文では，センサデータを閲覧する際の，時間情報を考慮することが重要であることを踏まえ，空間制約（SC）と時間制約（TC）の両方が指定された場合には，検索結果は基本的に時区間分割に基づく順序で提供され，各時区間の中で領域分割が利用されるものとする．これにより，センサデータの統合結果の空間的な分布を，時間順序に従って閲覧することが可能となる．空間制約（SC）を指定しない場合には，時区間分割に基づくデータ提供が行われ，時間制約（TC）を指定しない場合には，領域分割のみが行われる．

4.2.3 空間制約と時間制約を考慮した検索結果の品質の定義

本論文では，空間制約（SC）と時間制約（TC）を考慮して，センサデータ検索の結果品質 $q_{tc,sc}(t)$ を式 (4.10) のように定義する．

$$q_{tc,sc}(t) = \sum_{l=1}^{m} \sum_{i=1}^{j} \frac{N_{l,i}(t)}{N_{all}} \tag{4.10}$$

$N_{l,i}(t)$ は時刻 t における分割領域 $a_{l,i}$ に対する収集データ数である．N_{all} はそのクエリに対して収集される総データ数である．m, j に関して，TS_m は，時区間制約を満たして検索結果が収集されている最新の時区間を示し，$a_{l,j}$ は，TS_l に対して，空間制約を満たして検索結果が収集されている最新（最大）の分割領域を示す．つまり，$TS_l(1 \leq l \leq m-1)$ の検索結果の収集が完了し，TS_m について $a_{m,i}(1 \leq i \leq j-1)$ の検索結果の収集が完了していることを示す．$q_{tc,sc}$ は，時間制約と空間制約を満たしている領域のデータ収集率によって計算され，それらの制約に基づいて検索結果が転送される場合には増加するが，そうでない場合には変化しない．

検索結果品質 $q_{tc,sc}$ を向上させるためには，時間制約と空間制約を満たすように，センサデータの位置情報と時刻情報を考慮して，各サーバからの検索結果を同期させて転送する必要がある．

4.2.4　仲介エージェントによる検索結果の転送制御

本節では，$q_{tc,sc}$ を向上させるために，仲介エージェントがサーバへのクエリの送信とサーバからの検索結果の転送を制御する方式（本論文では，buf 方式と呼ぶ）について述べる．

処理手順

検索システムは，クライアントからのクエリ Q に対して，次の手順に従って動作する．

(1) 仲介エージェントは，時間制約（TC）と空間制約（SC）に基づいて，Q をサブクエリ $q_{l,i}$ に分割し，サブクエリのリストを作成する．リスト中のサブクエリは，時区間分割に従って順序付けられ，各時区間ごとに領域分割に従って順序付けられる．この時，検索結果を保持するためのバッファも同様の順序を持つように作成する．

(2) 仲介エージェントは，$queue\_size$ 個のサブクエリを，該当するセンサデータサーバにまとめて送信する．$queue\_size$ は，仲介エージェントのバッファリングのためのパラメータである．

(3) 仲介エージェントは，各サーバからの検索結果を各時区間 TS_l に対する各領域 $a_{l,i}$ ごとにバッファに保持し，バッファの順序に従って検索結果をクライアントに転送する．$a_{l,i}$ に対する検索結果の転送を完了した後に，$a_{l,i+1}$ の検索結果の転送を開始し，サブクエリ $q_{l,i+queue\_size}$ を該当するサーバに送信する．k を分割領域数とする時，$i = k$ の場合には，次の時区間（TS_{l+1}）の分割領域 $a_{l+1,1}$ の検索結果を転送する．$(i + queue\_size) = k$ の場合には，サブクエリ $q_{l+1,1}$ を送信する．

(4) サブクエリ $q_{l,i}$ を受信したサーバは，各サブクエリに対して探索を行い，その探索結果を分割時区間と分割領域の順序に従って逐次的に仲介エージェントに転送する．

時間制約と空間制約を考慮したクエリの実行と検索結果転送の制御により，複数のセンサデータサーバからの検索結果を，時区間ごと，領域ごとに同期させてクライアントが収集することが可能となる．さらに，$queue\_size$ を指定することにより，

サブクエリの先行要求と検索結果のバッファリングが行われ，複数のセンサデータサーバからのデータ収集を効率的に行うことができる．

センサデータサーバの選択

手順 (2) で，センサデータサーバを選択するために，仲介エージェントは，まず，$q_{l,i}$ に指定されたデータの種類（DT）と各センサデータサーバの提供するデータの種類（:data-type）を比較し，指定されたデータを提供できるセンサデータサーバを探し出す．そして，そのセンサデータサーバに対応するセンサネットワーク s の位置情報 SNR_s と $q_{l,i}$ の問合せ領域 $a_{l,i}$ とが重なっているかどうか，すなわち，$a_{l,i} \cap SNR_s \neq \phi$ が成り立つかどうか調べる．$C_1 \equiv (a_{l,i}^{upper} \cap SNR_s \neq \phi)$ および $C_2 \equiv (SNR_s \subseteq a_{l,i}^{lower})$ という 2 つの条件を評価し，C_1 が真，C_2 が偽である場合に，センサネットワーク s からのセンサデータを提供するサーバが選択される．

センサデータサーバによるデータ探索

手順 (4) で，サブクエリ $q_{l,i}$ を受信したセンサデータサーバは，location table を参照し，分割領域（$a_{l,i}$）に含まれるセンサノードを探し出し，分割領域ごとにノード識別子（$NODE\_ID$）のリスト（ID_LIST）を生成する．センサノード v の位置 p_v について，条件（$p_v \subseteq a_{l,i}$），すなわち，式 (4.11) の条件が成り立つ時，ノード v を ID_LIST に加える．

$$(p_v \subseteq a_{l,i}^{upper}) \ \wedge \ \neg(p_v \subseteq a_{l,i}^{lower}) \tag{4.11}$$

そして，センサデータサーバは，ID_LIST の要素をキーにして，時系列データテーブルを探索し，クエリの条件を満たすセンサデータを探し出す．テーブル中のセンサデータの更新時刻 t について，時刻 t が時区間 TS_l に含まれるかどうかを判断するために，式 (4.12) の条件が成り立つかどうかを調べる．

$$(t \subseteq TS_l^{upper}) \ \wedge \ \neg(t \subseteq TS_l^{lower}) \tag{4.12}$$

センサデータサーバによる探索結果は，時間属性を持つデータ（temporal data）の系列，すなわち時系列データ（time series data）として表現され，XML 文字列として仲介エージェントに転送される．仲介エージェントは，各センサデータサーバからの検索結果を，別々のパケットとしてクライアントエージェントに転送する．検索結果の XML 表現のための DTD を図 4.5 に示す．

各検索結果は，<**sensor-data**> タグによって囲まれ，<**data-item**> タグの中に各ノードに対する時系列データ（<**time-series**>）が記述される．<**temporal-item**> は，**id** に指定されたセンサノードの観測値 (**value**) と更新時刻 (**time**) から

```
<!ELEMENT sensor-data (data-item)*>
<!ELEMENT data-item
      (id,type,location,time-series)>
<!ELEMENT id (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT location (x,y)>
<!ELEMNET x (#PCDATA)>
<!ELEMENT y (#PCDATA)>
<!ELEMENT time-series (temporal-item)+>
<!ELEMENT temporal-item (value,time)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT time (#PCDATA)>
```

図 4.5: センサデータの XML 表現のための DTD

成る．<**location**> タグによって各センサノードの位置情報が記述され，この位置情報に基づいてクライアントでの統合処理が行われる．

4.3　実験および評価

4.3.1　実装

本検索機構を Java 言語（JDK1.2）を用いて実装した．各エージェントは複数のスレッドによって実装され，検索結果の転送には TCP のコネクションを利用する．センサデータサーバは，起動時に読み込んだセンサノードの位置情報から location table を生成する．センサデータ検索時には，あらかじめ生成した時系列データを蓄積したファイルを参照する．検索システムを構成するエージェントは，表 4.2 に示すホスト（Sun Ultra WorkStation，Solaris OS）上で動作させた．CA はクライアント，MA は仲介エージェント，SA はセンサデータサーバである．

表 4.2: 測定環境

	CPU	メモリ	エージェント
host 1	440 MHz	1024 MB	CA
host 2	333 MHz	384 MB	MA, SA
host 3	900 MHz (×2)	2 GB	SA

実験用のサンプルデータとして，気象庁提供の気象データ CD-ROM「アメダス

観測年報（2000 年）」（気象業務支援センター発行）に収録されている時別値データを利用した．この気象データ CD-ROM には，気温，降水量，風速などに関する 1 年分の時系列データが，月ごとに，観測点別にファイルに保存されている．各データファイルは 1 時間単位で順番に並んだデータレコードから構成される．

4.3.2，4.3.4 節の実験では，アメダスの観測点をセンサノードとし，気温データ（temp）あるいは降水量データ（prec）を提供するセンサネットワークを想定する．センサデータサーバは，センサネットワークごとに時系列データを管理し，アメダスの観測点情報に記録されている経緯度座標に基づき location table を生成する．

なお，以降の実験では，2 点の座標で表現される矩形領域を問合せ範囲としてクエリに指定し，その矩形領域を，基準点の座標 (x_{base}, y_{base}) と各軸方向の分割幅 (dx_+, dy_+, dx_-, dy_-) によって表現される空間制約（SC）を用いて分割する（図 4.3）．問合せ領域と空間制約は，経緯度座標系で指定し，単位は，度とする．また，クエリ（Q）に指定した時区間の長さを $t_{q,len}$ と表現する．

4.3.2 気象データを用いた実験

アメダスの気温ノードを，北海道，東北，関東，中部，近畿，中国，九州地方という 7 つのグループに分ける．各グループに対応するセンサデータサーバによって，各グループに含まれるセンサノードの観測データが時系列として管理され，7 つのセンサデータサーバから日本全土の気温データが提供されるものとする．各サーバの管理する領域（SNR）と気温ノード分布を図 4.6 に示す．

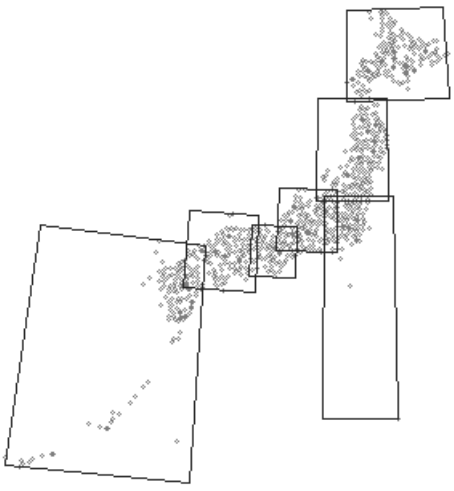

図 4.6: 複数のサーバに管理されたセンサノードの分布と SNR

さらに，図 4.6 のセンサノードの分布領域 (SNR) と，各 SNR に含まれるセン

サノード数を表 4.3 に示す．表 4.3 の SNR は，経緯度座標系を用いて，⟨ 西端の経度座標 , 南端の緯度座標 , 東端の経度座標 , 北端の緯度座標 ⟩ と表現する．

表 4.3: センサノードの分布領域 SNR とセンサノード数

サーバ	グループ	SNR	ノード数
SA_1	北海道	$\langle 139.561, 41.421, 145.763, 45.518\rangle$	160
SA_2	東北	$\langle 137.865, 36.931, 141.968, 41.525\rangle$	173
SA_3	関東	$\langle 138.391, 27.090, 142.188, 37.121\rangle$	90
SA_4	中部	$\langle 135.786, 34.600, 139.095, 37.443\rangle$	108
SA_5	近畿	$\langle 134.378, 33.448, 136.828, 35.735\rangle$	72
SA_6	中国	$\langle 130.928, 32.720, 134.745, 36.201\rangle$	116
SA_7	九州	$\langle 123.010, 24.058, 131.925, 34.695\rangle$	123

検索結果品質の変化

まず，表 4.3 に示す複数のセンサデータサーバに，仲介エージェントを介して，クエリを送信し，空間制約を指定した時の検索結果品質の変化について調べる．検索結果品質の変化は，クライアントエージェントが検索結果を受信した時刻（t_{wait}）を記録し，各時刻に対して 式 (4.10) で定義した $q_{tc,sc}$ を計算することによって求めることができる．

日本全土の気温分布を表示するためのクエリ（$A_{jp} = \langle 123, 24, 146, 46\rangle$，$DT = \{temp\}, t_{q,len} = 20, SC = \{(x_{base}, y_{base}) = (123, 24),\ dx_+ = dy_+ = 1\}$）を送信した時の検索結果品質（$q_{tc,sc}$）の変化を図 4.7 に示す．図 4.7 の横軸は t_{wait}，縦軸は $q_{tc,sc}$ である．

図 4.7 の buf は，4.2.4 節で述べた提案手法の検索応答を示す．図中の w は $queue\_size$ である．図 4.7 より，$w = 1, 2, 12$ いずれの場合も，検索結果品質が時間とともに増加していることがわかる．$w = 1$ の場合には，a_{i+1} に対するサブクエリの送信を，a_i に対するすべての結果を受信するまで待つため，その待ち時間によるオーバヘッドが大きく，総応答時間が長くなっている．$w = 2, 12$ の場合は，各サーバにサブクエリを先行要求することで，サブクエリの実行開始時刻を早めることができるため，同期のための待ち時間を短縮し，総応答時間を短くすることができる．

ここで，図 4.8 に各分割領域 a_i に含まれるセンサノード数をサーバごとに示す．この実験では，図 4.6 を覆う問合せ領域 (A_{jp}) の左下 (南西) 端点を分割の基準としているため，番号の小さい分割領域ほど九州地方に近い領域となっている．

図 4.8 では，領域 r_9 の九州地方に含まれるセンサノードの個数や領域 r_{11} の中国

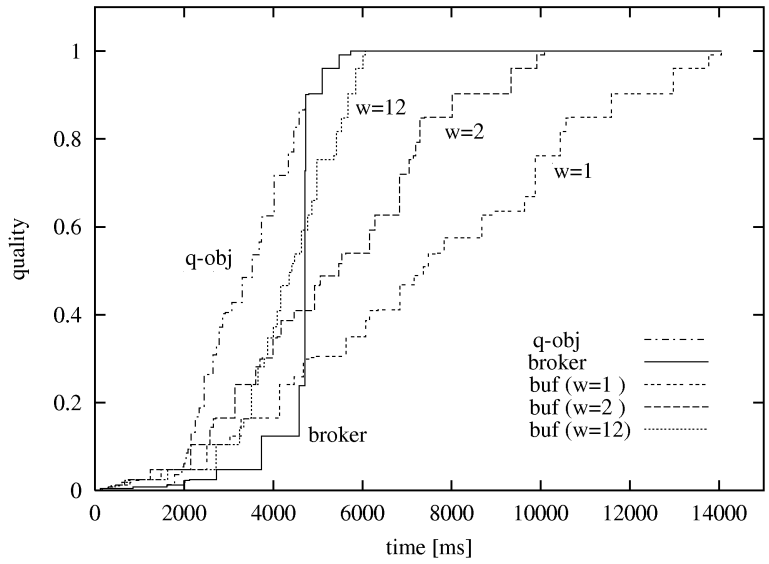

図 4.7: 複数のサーバに気温データを要求した時の検索結果品質の変化

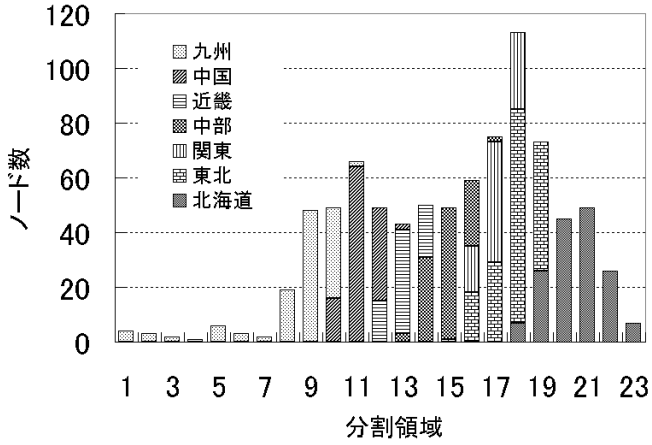

図 4.8: 各分割領域に含まれるセンサノード数

地方に含まれるセンサノードの個数が多いことがわかる．$w = 12$ の場合には，これらのセンサノード数が多く，探索コストの大きい領域のデータを先行要求しているため，$w = 2$ の場合よりも総応答時間が短縮されている．

図 4.7 の broker は，各サーバが独立にクエリの分割を行い，仲介エージェントがサーバからのサブクエリの実行結果をそのままクライアントに転送する方式である．図 4.7 の q-obj は，broker 方式を用いた時の単なるデータアイテムの収集率を示し，q_{obj} と broker 方式の $q_{tc,sc}$ の差は，空間制約を満たしていないデータアイテムの数に相当する．図 4.7 では，broker 方式を用いた場合に，データ収集の後半のある時刻に結果品質が急激に上昇している．このことは，空間制約を満たしていないデータアイテムが未処理のままクライアントで保持され，その時刻に大量のデータの処理を開始する必要があることを示す．一方，データ収集の初期の段階では，broker 方式 と比較して buf 方式の検索結果品質が高く，これは初期の段階で収集される空間制約を満たしたデータアイテム数が broker 方式より buf 方式が多いことを示す．したがって，クライアントが空間制約に指定した順序で領域ごとに逐次的な統合処理を行いたい場合には，buf 方式を用いると受信データを効率良く処理できることがわかる．

図 4.7 では，buf 方式より，broker 方式の総応答時間が短くなっている．本節の実験では，受信したセンサデータを，位置情報に基づいて地図上に描画するという単純な統合処理を行っているため，クライアント側での統合処理にかかる時間は小さい．それに対して，5 章，6 章で取り上げる空間集約のようなコストの大きい処理をクライアントが行う場合には，前述の議論から，総応答時間が大きくなると考えられる．この統合処理コストの影響については，6.3.1 節，6.3.2 節で考察している．

位置情報に基づくセンサデータ統合

次に，位置情報に基づくセンサデータ統合の一例として，時系列データの集約演算の結果をメッシュ統合する場合を取り上げる．この例では，クライアントは，検索結果である時系列データを取得すると，センサノードごとに平均値，最大値，最小値などを算出するための集約演算を実行する．そして，各ノードに対する時系列データの集約演算の結果をメッシュの目（セル）ごとに統合し，その結果を表示する．

ここでは，日本全土の平均気温分布を統合結果として表示するために，前述の実験と同様のクエリ（$A_{jp} = \langle 123, 24, 146, 46 \rangle$，$SC = \{(x_{base}, y_{base}) = (123, 24),\ dx_+ = dy_+ = 1\}$）を，クライアントが送信するものとする．図 4.9 は，収集した気温データの集約結果である平均気温について，これらの集約結果の平均値を各セルごとに算出し，メッシュデータとして描画したものである．図 4.10 は，buf 方式を用いた場合の途中結果に基づくメッシュ統合の例を示す．図 4.9，図 4.10 は，メッシュ統

合の結果だけでなく，クエリ領域の分割の様子を示す．メッシュは，第 1 次地域区画[1][JMC 98] に相当する．

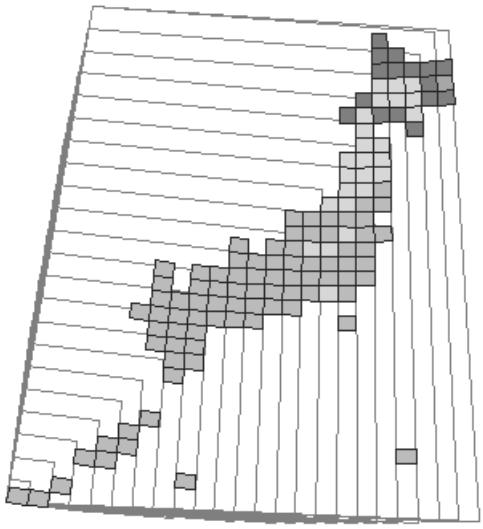

図 4.9: 平均気温のメッシュ統合例（buf 方式）

提案手法（buf 方式）を用いた場合，図 4.10 → 図 4.9 のように，基準点 $(123, 24)$ に近い領域から順番に逐次的なメッシュ統合を実現できる．これは，各分割領域に対する検索結果を受信する度に，その領域と関連するセルのデータ収集の完了を判別できるためである．一方，各サーバがそれぞれ独立に返答する broker 方式では，各サーバからの検索結果は順番に取得できるが，複数のサーバからの検索結果を領域ごとに同期させながら収集することができない．したがって，検索結果受信時に，データ収集を完了した領域を判別できないため，途中結果に基づく逐次統合ができず，全データの収集を完了するまで統合結果を表示することができない．結果として，各ノードに対する時系列の長さによっては，ユーザに対する応答時間の増大をもたらす．

4.3.3 異種センサデータ統合

本節では，異種センサネットワーク環境を想定した環境での実験を行う．具体的には，国土数値情報 [国土交通省 a] の「鉄道データ (平成 7 年度)」から抽出した駅ノード，同じく「道路データ (平成 7 年度)」から抽出した交差点ノードを，利用客数や交通量などを出力するセンサノードと仮定する．本実験で想定する異

[1]数値地図情報整備のための標準地域メッシュの 1 つである．1 次メッシュとも呼ばれ，全国の地域を 1 度ごとの経線と 3 分の 2 度 (40 分) ごとの緯線によって，縦横に分割したものである．

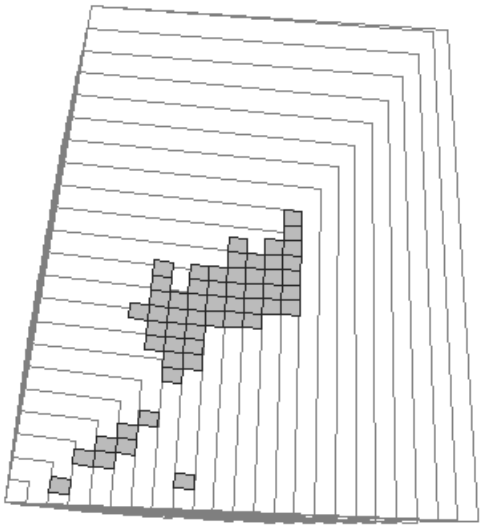

図 4.10: 途中結果に基づく平均気温のメッシュ統合例（buf 方式）

種センサネットワーク環境を構成するセンサノード数を表 4.4 に示す．$A_{tokyo} = \langle 139.5, 35.49, 139.9, 35.89 \rangle$ である．

表 4.4: センサノード数

問合せ領域	駅ノード	交差点ノード	降水量ノード
A_{tokyo}	672	1009	8

本節の実験では，4.3.2 節の実験例よりも，ノード数が多く，密度の高い異種のセンサネットワークが互いに重なり合う状況を想定している．図 4.11 に，A_{tokyo} に含まれるセンサノードの分布を示す．駅ノードは 1 個のサーバによって，交差点ノードについては県別に 7 個のサーバによって管理されているものとする．ここでは，これら合計 8 個のサーバに対して，東京近郊（A_{tokyo}）のセンサデータを要求し，空間制約（$SC = \{(x_{base}, y_{base}) = (139.7, 35.69),\ dx_+ = dy_+ = 0.05,\ dx_- = dy_- = -0.05\}$）を指定した場合について調べた．図 4.11 は最終的な結果であるノード分布，図 4.12 は途中結果に基づくノード分布を示す．

図 4.11 と 図 4.12 は，同一の問合せ領域（A_{tokyo}）に関する表示結果であり，図 4.12 → 図 4.11 のように，基準点である A_{tokyo} の中心 $(139.7, 35.69)$ に近い領域から，逐次的にノード分布が表示されることがわかる．

一方，図 4.13 に，broker 方式を用いた場合の同一のクエリに対する途中結果に基づくノード分布を示す．

この例では，各センサデータサーバのノードの分布は分割領域ごとに同期して表

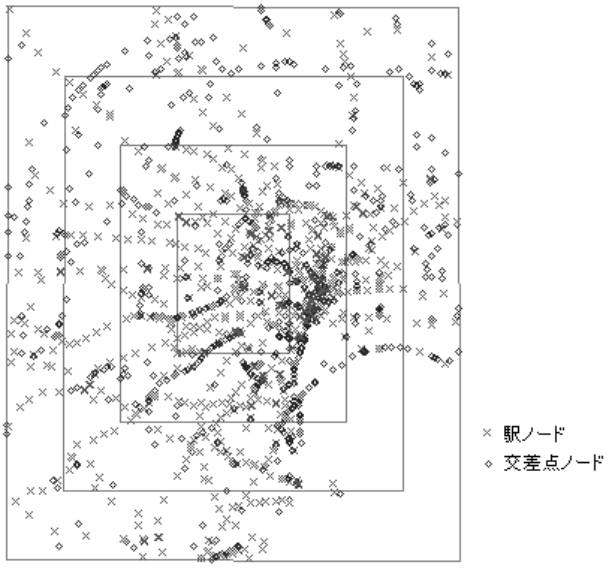

図 4.11: 異種センサノードの分布の表示（駅ノード，交差点ノード）

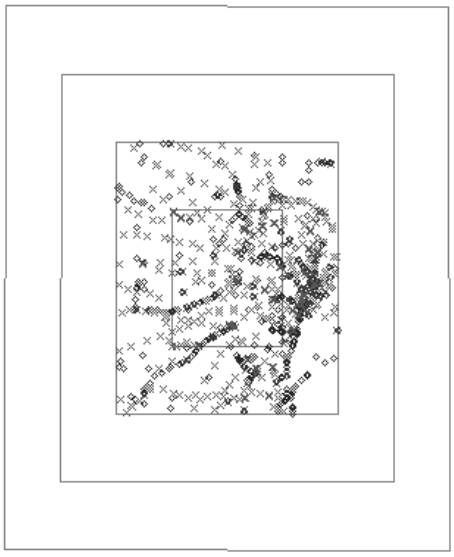

図 4.12: buf 方式を用いた場合の異種センサノードの分布の逐次表示

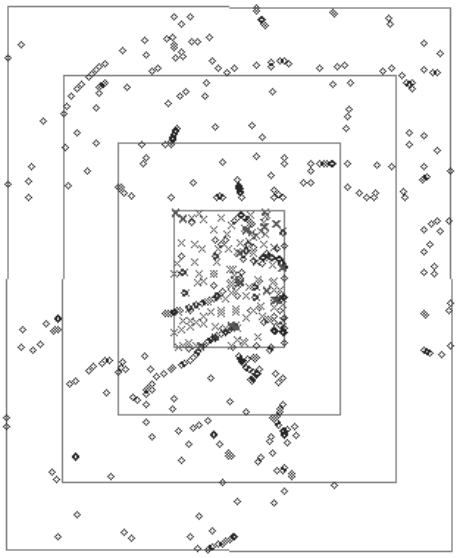

図 4.13: broker 方式を用いた場合の異種センサノードの分布の逐次表示

示されず，各サーバごとに基準点に近い領域から表示される．これは，各サーバの管理するセンサノードの分布や密度，さらにはサーバでの領域分割や検索応答が異なり，それらのサーバがそれぞれ検索結果を返すためである．結果として，ユーザの要求した順序で検索結果を収集できるとは限らない．

以上より，領域分割に基づく本手法は，配置（ノード分布）やノード密度の異なるセンサネットワークが混在する環境においても，空間制約を満たすように複数のサーバからのデータを領域ごとに同期させながら収集することができることがわかる．このことは，位置情報に基づく異種センサデータ統合をインクリメンタルに行う上で有用であると考えられる．

4.3.4　時区間分割に基づく時系列データ統合

本節では，時区間分割に基づく時系列データ統合の例を扱う．実験データとして，アメダス気象データ CD-ROM に収録されている降水量データを用いた．

本節の実験では，時間制約（TC）として，順方向の分割幅（t_+）を指定し，クエリ（Q）に指定した時区間（TI）の下限値（$TI.t_{min}$）を基準時刻（t_{base}）とする．時間制約の単位は，時とする．

時系列集約結果のインクリメンタルな結合

$A_{tp} = \langle 138.35803, 34.53578, 141.06116, 37.58538 \rangle$ の範囲の 120 時間分（$TI = [t_{base}, t_{base} + 120]$）の降水量データを要求し，時間制約として $TC = \{t_+ = 12\}$ を指定した時の時系列データ統合の結果を図 4.14，図 4.15，図 4.16 に示す．ここでは，時系列データ統合として，積算降水量の分布の表示を扱う．クライアントは，各時区間のデータを受信するごとに，各時区間内の各ノードの降水量データを積算し，それまでの積算値に加算し，その加算結果を表示していく．この時，各分割時区間 TS_i に関して，$TS_i^{upper} = [t_{base}, t_{base} + 12 \times i]$ と表現する．たとえば，$i = 5$ の時，$TS_5 = (TS_5^{upper} - TS_5^{lower}) = ([t_{base}, t_{base} + 60] - [t_{base}, t_{base} + 48])$ となる．

図 4.14，図 4.15，図 4.16 は，それぞれ TS_1，TS_5，TS_{10} に対する途中結果に基づく積算降水量分布を示し，図 4.16 は最終的な 120 時間分（すなわち 5 日分）の積算降水量分布を示すことになる．なお，図中の棒グラフは各ノードの積算降水量を示し，積算値を高さとして表現し，階級ごとに色分けしている

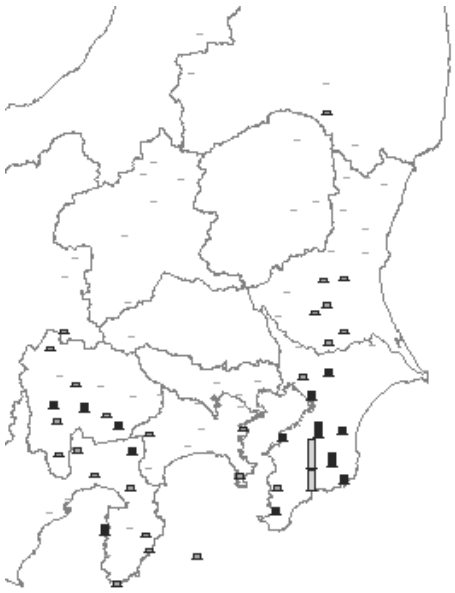

図 4.14: 途中結果に基づく積算降水量分布（TS_1 受信時）

時区間分割を用いない場合には，図 4.16 が結果として提示されるのみであるが，120 時間分のデータを収集するため，その統合結果を表示するまでの応答時間が増大し，その間，何の情報も提供されない．それに対して，時区間分割を用いた場合には，時系列集約の途中結果を図 4.14 → 図 4.15 → 図 4.16 のように，インクリメンタルに提示することができる．さらに，“時区間” という意味のあるまとまりで，時間順序に従って，統合結果を表示しているので，時系列データの集約の過程を観察することもできる．たとえば，図 4.16 より，5 日間の雨の降水状況として，図の

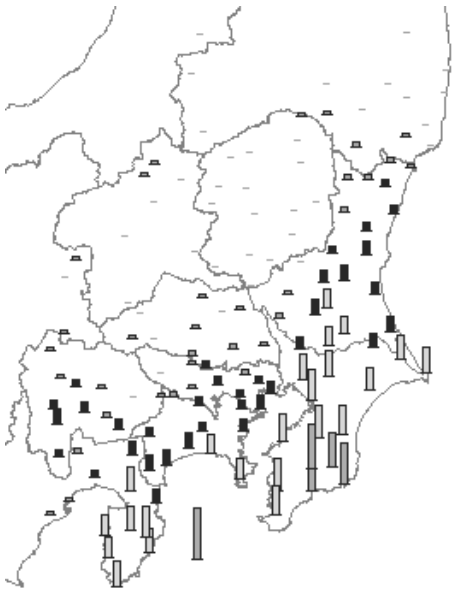

図 4.15: 途中結果に基づく積算降水量分布（TS_5 受信時）

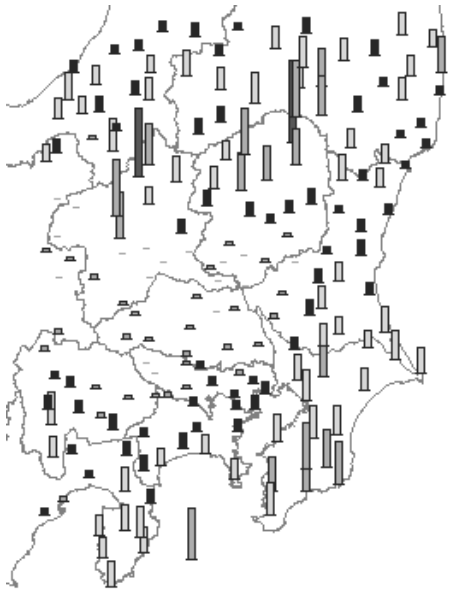

図 4.16: 積算降水量の分布（TS_{10} 受信後）

下部（南部）と上部（北部）で降水量が多いことがわかるが，図 4.14 と図 4.15 より，最初の 60 時間では南部の降水量が多く，後半の 60 時間で北部の降水量が多いことがわかる．

時系列データ収集時の応答時間が増大する場合に，検索結果を分割する基準として，領域分割を用いるか，時区間分割を用いるか，あるいは，その両方を用いるかを決定することは効果的なセンサデータの閲覧を実現する上で重要である．本節で扱ったような途中結果の時間変化も含めて閲覧したい場合には，指定した全時区間の集約結果が領域ごとに表示される領域分割ではなく，時系列集約の途中結果を空間的な分布として表示できる時区間分割を用いることが適していると考えられる．

時系列集約結果の時間変化

前節の実験が各時区間ごとの集約結果を結合していたのに対して，本節の実験では，日本全土（A_{jp}）の 60 時間分の降水量データを要求し，時間制約として $TC = \{t_+ = 6\}$ を指定し，各時区間ごとの積算降水量の分布を表示させる．時区間 TS_8，TS_9，TS_{10} に対する積算降水量の分布を図 4.17，図 4.18，図 4.19 に示す．

図 4.17: 時区間 TS_8 に対する積算降水量分布

この例では，時区間ごとにインクリメンタルに提供される検索結果に基づいて，それぞれの時区間での降水量分布を表示していく．したがって，図 4.17 → 図 4.18 → 図 4.19 のように，クエリに指定した時区間（TI）の降水量の変化を時間順序に従って観察することができる．図 4.17 までは降雨地域が九州地方から東の方向に向かって徐々に拡大していく傾向が見られたが，その後，図 4.18，図 4.19 に示すよ

図 4.18: 時区間 TS_9 に対する積算降水量分布

図 4.19: 時区間 TS_{10} に対する積算降水量分布

うに，降雨地域が減少していく，すなわち指定した時区間の最後の 18 時間で，雨が上がっていく様子が観察される．このように，時区間分割に基づいて提供されるセンサデータを利用して，時間順序に従って，ある範囲のセンサデータの変化だけでなく，降雨地域の移動や拡大状況などセンサデータの分布範囲が変化する様子も含めて観察することができる．したがって，時区間分割に基づく本手法は，センサデータ統合の結果をダイナミックに提示し，効果的なセンサデータプレゼンテーションを実現する上で有用であると考えられる．

時区間分割に基づくプレゼンテーション

前節までの例では，時間制約として t_+ を指定した場合を取り上げている．「時系列集約結果のインクリメンタルな結合」の例では，各時区間の統合処理の完了ごとに，それまでの集約結果（積算降水量）のみを保存することで，図に示したような逐次的な表示を実現できる．「時系列集約結果の時間変化」の例では，各時区間ごとに，その都度，集約結果を表示できるため，計算結果を保持する必要がない．逆に，各時区間に対する統合結果を保持しておくことで，受信したデータをすべて保持しなくても，すべての統合結果を表示した後で，ある時区間に対する統合結果を再度表示することができる．クライアントが，これらの途中結果を保持するのに十分な記憶容量を持っていれば，データサーバに対して，新たにデータ要求をしなくても，集約結果を再生することができる．さらに，センサデータを受信するごとに，ある時間幅（window）のセンサデータを保持し，その window に含まれる時系列データを統合していくことで，sliding window による時系列集約処理も実現可能である．たとえば，1 時間おきに過去 6 時間の降水量分布を連続して閲覧したい場合には，クライアント側で，すべての受信データを保持しなくても，window を 1 時間単位で移動させながら，window に含まれる最新の 6 時間分のデータを常に保持するようにすればよい．

t_- を指定することにより，現在に近い時刻から逆順序に過去にさかのぼって，時区間ごとの集約結果をアニメーションとして表示することができる．t_- を指定した場合は，通常の時間軸と逆方向の情報提供になるが，t_+ の場合と同様に，各時区間ごとに，集約結果を記憶すれば，任意の時区間に対する処理結果を参照することが可能である．

$t_+(>0), t_-(<0)$ の両方を指定した場合には，基準時刻 t_{base} を中心とした時間軸の両方向に広がる逐次結果表示となる．たとえば，１日の最高気温や平均気温の分布を表示したい場合に，時区間分割のパラメータとして，$t_{base} = 12$ 時 ，$t_+ = 3$ 時間，$t_- = -3$ 時間 を指定する場合を考えてみる．この場合，正午を中心として，$[12-3*i, 12+3*i](i = 1, 2, 3, 4)$ という時区間単位で，統合の途中結果が表示される．一般に，気温は，夜間より昼間の方が高いため，最高気温分布を知りたい場合には，データ収集の初期の段階で，最終的な結果を閲覧できる可能性がある．

平均気温分布を知りたい場合には，途中結果の変化の度合いから，気温変化の度合いを把握することができると考えられる．変化が少なければ，昼夜の気温差が小さいことになる．そのほか，台風接近時の降水量や風速の変化，通勤時間の前後での列車の混雑度や道路の渋滞状況を知りたい場合にも，t_+，t_- の両方を指定することが有用であると考えられる．

以上より，時区間分割に基づくデータ提供という枠組みの中で，受信データや途中結果の処理方法，時区間分割のパラメータの指定方法などを変えることで，多様なアプリケーション記述が可能であることがわかる．

センサデータストリーム技術との比較

データベースの分野では，近年，センサデータベースやセンサデータストリームに関する研究が盛んに行われている [Bonnet *et al.* 01, Madden *et al.* 02, Aurora]．特に，センサからのストリームデータを対象とした研究が盛んであり，環境モニタリングなどのリアルタイムアプリケーションを構築する上で重要である．本章で提案した手法は，複数のデータリソースからのセンサデータを，時間情報と空間情報を考慮して，クライアントが処理しやすいように整列化している点で，センサデータストリームに関する研究との類似点も見られる．空間制約や時間制約を満たす順序でセンサデータが提供されれば，データの受信ごとに逐次的に統合処理ができるが，そうでない場合には，複数のサーバからのデータを保持するためのオーバヘッドが大きくなることもある．

また，領域分割と時区間分割のパラメータを指定することにより，センサデータ統合の結果を表示する順序を制御できるという点では，マルチメディアデータのストリーミングや QoS 制御のための手法との類似性を指摘することもできる．提案手法は，センサデータの持つ空間属性と時間属性を扱っているという点が特徴的であるが，ネットワーク環境における品質制御手法を参考にしながら，より効果的な分散センサデータのプレゼンテーションを実現したいと考えている．

4.4　まとめ

本章では，センサデータベースの検索の手法として，インクリメンタルに検索結果であるセンサデータを提供する方式を提案した．位置情報と時間情報の両方を考慮した逐次的なデータ提供は，時系列のセンサデータを収集し，統合する際の応答時間の増大に対処するだけでなく，効果的なセンサデータの閲覧を実現する上での情報提示（プレゼンテーション）技術として有用であることが，実験を通して示唆された．領域分割に基づくデータ提供は，位置情報に基づいて逐次的にセンサデータ統合を行う場合に，時区間分割に基づくデータ提供は，時間順序を考慮して時系

列データ統合の結果を表示する場合に，それぞれ効果的である．また，領域分割と時区間分割に基づくアプローチは，クライアントが時間順に領域ごとに逐次的に統合処理をできるというだけでなく，ユーザがパラメータを指定することによって，その処理結果の提示順序を制御できるという点で視覚的な効果がある．

第5章

空間集約に基づくセンサデータの視覚化手法

本章では，4章で述べたセンサデータ検索システムから取得したセンサデータを加工して表示する方法，すなわち，センサデータを視覚化する方法について述べる．まず，受信したセンサデータを加工するための空間集約処理について述べ，実験を通して，センサデータの視覚化における空間集約処理の有用性について議論する．さらに，空間集約に基づくセンサデータ視覚化システムにおいて，移動履歴に基づいてセンサデータを閲覧する方法を提案する．

5.1 センサデータ視覚化システム

5.1.1 システム構成

センサデータ視覚化システムの構成を 図5.1に示す．クライアントは，4章で提案したセンサデータ検索システムからセンサデータを受信し，3章で提案した空間データサーバから空間データを取得し，その統合結果を表示する．

本章では，クエリ Q を次のように表現する．

$$Q = \langle A, DT, TI, DC \rangle \tag{5.1}$$

問合せ領域 A，データの種類 DT，時区間 TI については，4章と同様であるが，これらのクエリパラメータに加えて，DT に指定した各センサデータに対する条件式を $DC = \{dc_1, dc_2, ..., dc_n\}$ に記述する．

たとえば,「気温が 24 °C 以上」という条件を指定したい場合には，$\{\text{temp} \geq 24\}$ のように記述する．なお，空間制約 (SC) および時間制約 (TC) については，本章の議論では取り上げないため，ここでは省略する．

クライアントは，仲介エージェントに，クエリ Q を送信し，Q に対する検索結

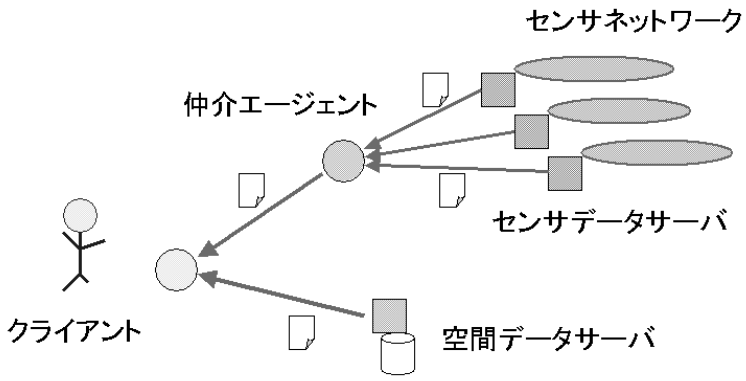

図 5.1: センサデータ視覚化システムの構成

果を受信する．そして，受信したセンサデータに対して，空間集約処理を行い，その結果を表示する．次節より，具体的な空間集約処理の手順について述べる．

5.1.2　空間集約処理の手順

以下に，空間集約処理の説明に用いる記号表現を挙げる．

- p：空間データオブジェクト
  p に対する幾何形状を $p.geom$，その包囲矩形を $p.rect$ とする．
- o：センサデータオブジェクト
  o の位置属性 $o.loc$ とする．$o.loc$ は，センサノードの位置座標を示し，点データとして表現される．
- c：メッシュの構成要素であるセル
  c の形状である矩形を $c.rect$ と表現する．

空間データ統合

本論文では，統合対象となる空間データとして，行政界データなどのポリゴンデータを考える．問合せ領域 A と重なり合う，すなわち，$p.geom \cap A \neq \phi$ を満たすポリゴンデータオブジェクト p を統合対象とする．

空間データ統合では，センサデータ要求のためのクエリを送信する前に，統合対象となるポリゴンデータを，ネットワークを介して空間データサーバから取得する．

そして，受信した各センサデータオブジェクト o について，$o.loc \in p.geom$ を満たすポリゴンデータオブジェクト p に関連付けて処理を行う．

空間補間に基づくメッシュ統合

本論文では，2.1.5 節で述べた空間補間の手法として， 逆距離加重法 (Inverse Distance Weighted，以下，IDW と呼ぶ) [Longley *et al.* 01, McCoy *et al.* 01] を採用する．本章では，センサデータの視覚化における空間補間の適用の有用性を示すことを目的としており，その補間法の 1 つとして IDW を選択する．IDW は非常に簡単なアルゴリズムではあるが，クリギングなど他の補間方法と比べると処理コストが小さく，実用的な方法であり，空間補間の有用性を示すには十分な手法である．

IDW では，ある地点 x の値 $z(x)$ を，近接する複数の点 i の値 z_i を利用し，式 (5.2) を用いて計算する．

$$z(x) = \frac{\sum_i w_i z_i}{\sum_i w_i} \tag{5.2}$$

w_i は，z_i に対する重みであり，点 x と 点 i の間の距離 d_i を用いて，式 (5.3) のように定義される．

$$w_i = 1/d_i^2 \tag{5.3}$$

なお，指定した地点 x に，点データ z_i が存在する場合，すなわち，$d_i = 0$ となる場合には，式 (5.4) を用いて計算を行う．

$$z(x) = z_i \tag{5.4}$$

問合せ領域に対してメッシュを生成し，メッシュを構成する各セルの中心点に関するデータを周囲のセンサデータで補間し，この処理をすべてのセルに対して行うことで，IDW に基づくメッシュ統合を実現できる (図 5.2)．

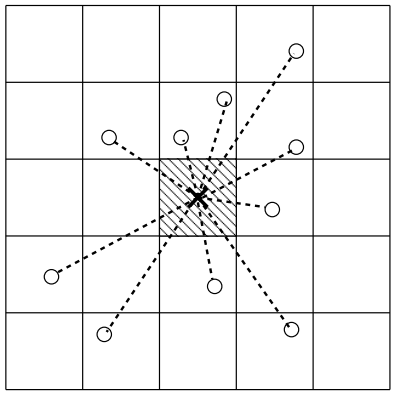

図 5.2: IDW に基づくメッシュ統合

IDW では，空間補間に利用する近接点の数を制限するためのパラメータが必要となる．たとえば，計算に最低限必要な点の個数や点探索の範囲を限定する半径をパラメータとして指定することで，補間処理に利用する近接点の数を制限する．IDW では原理的に，式 (5.3) の重み付けの定義からわかるように，遠くのデータは近くのデータより計算結果に対する影響力が小さいため，これらのパラメータを指定し，利用する近接点データ数を制限しても，十分に空間補間を行うことができる．

本論文では，計算に利用する周囲のセルを決めるパラメータとして max_cell を導入する．空間補間の対象となるセル内部の点に加えて，max_cell 個外側のセルに含まれる点を計算に利用する．$max\_cell = 1$ の時は，対象とするセルと外側の 8 個のセル，$max\_cell = 2$ の時は，さらに外側の 16 個のセルを加えた合計 25 個のセルを計算対象とする．

また，セル c について，これらの周辺セルを含む矩形領域を $cover\_rect(c)$ と表現する．補間したいセル (c) の 2 つの頂点の座標を (x_{min}, y_{min})，(x_{max}, y_{max}) ，セルのサイズを $(size_x, size_y)$ とする時，$cover\_rect(c)$ は，2 点の座標 $(x_{min} - size_x \times max\_cell, y_{min} - size_y \times max\_cell)$，$(x_{max} + size_x \times max\_cell, y_{max} + size_y \times max\_cell)$ で定まる矩形領域として定義できる．

メッシュの構成要素である セル c の値は，受信した検索結果の中で，$o.loc \in cover\_rect(c)$ を満たすセンサデータオブジェクト o の値を利用して，補間処理が行われる．$max\_cell \geq 1$ の場合は，各セルに対する $cover\_rect$ が重なるようになり，あるセルに含まれるセンサデータオブジェクトの値が，周辺の複数のセルの補間に利用されることになる．

オーバーレイ処理に基づくセンサデータ統合

本論文では，空間補間に基づいて生成されたメッシュデータに対して，オーバーレイ処理を適用することで，異種センサデータ統合を実現する．

まず，DT に指定したすべてのセンサデータ dt_l に対するメッシュデータ ML^l と，統合結果を保持するためのメッシュデータ OL を生成する．この時，各メッシュデータの分割とセルサイズは同一とする．

メッシュを構成するセルは，2 次元の位置を使って表現できるため，g を x 軸方向のセルの位置，h を y 軸方向のセルの位置とする時，各メッシュは，セル $c(g, h)$ の集合として定義される．

$$ML^l = \bigcup_{g=1,h=1}^{g=m,h=n} c^l(g, h) \tag{5.5}$$

$$OL = \bigcup_{g=1,h=1}^{g=m,h=n} c^+(g, h) \tag{5.6}$$

$c^l(g,h)$ は ML^l を構成するセル，$c^+(g,h)$ は OL を構成するセルを表す．m は x 軸方向のセルの数，n は y 軸方向のセルの数を示す．

センサデータ検索システムからセンサデータオブジェクトを受信した時には，そのセンサデータオブジェクトの種類 dt_l を判別し，前節「空間補間に基づくメッシュ統合」と同様の手順に従って，ML^l の要素である各セルの値を計算する．そして，$c^+(g,h) \in OL$ の属性値を，対応する $c^l(g,h) \in ML^l$ の補間結果を用いて計算し，統合結果として表示する．

一方，空間データ統合の場合にも，統合対象が，同一粒度の空間データオブジェクトであれば，同様の処理が可能である．たとえば，あるポリゴンデータに含まれる複数種類のセンサデータを，その種類別に，ポリゴンデータの属性として計算することで，そのポリゴンデータをキーとして，それらの結果を比較／統合することができる．

5.1.3 実験結果

4 章と同様に，センサデータとして気象データ CD-ROM「アメダス観測年報 (2000 年)」に収録されている気温データおよび降水量データを利用して実験を行った．

空間データ統合に基づく視覚化

まず，気温データを管理する単一のセンサデータサーバに対して，仲介エージェントを介して，クエリを送信し，ポリゴンデータと関連付ける例を取り上げる．クエリに対する検索結果には，各センサデータオブジェクトに対する時系列データが含まれるが，まず，各時系列に対する処理 (たとえば，平均値の算出) を行う．さらに，各ポリゴンに含まれるセンサデータオブジェクトの時系列処理結果に対して集計処理を行い，そのポリゴンの属性とする．

図 5.3 に，$A = \langle 123, 24, 146, 46 \rangle$ の範囲の 24 時間分のセンサデータを要求し，受信結果に対して，空間データ統合を適用した時の表示結果を示す．この例では，統合対象である空間データとして，国土数値情報 [国土交通省 a] の「行政界・海岸線データ (平成 11 年度)」より抽出した都道府県レベルのポリゴンデータを利用している．また，各センサデータオブジェクトごとに，24 時間の平均気温を算出し，各ポリゴンごとに平均気温の平均値を計算し，その計算結果に基づいて，ポリゴンを色分けして表示している．

図 5.4 は，問合せ領域として，$A = \langle 135.65, 35.57, 137.65, 37.57 \rangle$ を指定した時の，平均気温の分布を示す．この例では，国土数値情報 [国土交通省 a] の「行政界・海岸線データ」に含まれる市町村区レベルのポリゴンデータを利用している．

図 5.3: 空間データ統合に基づく気温分布 (都道府県レベルのポリゴンデータを利用)

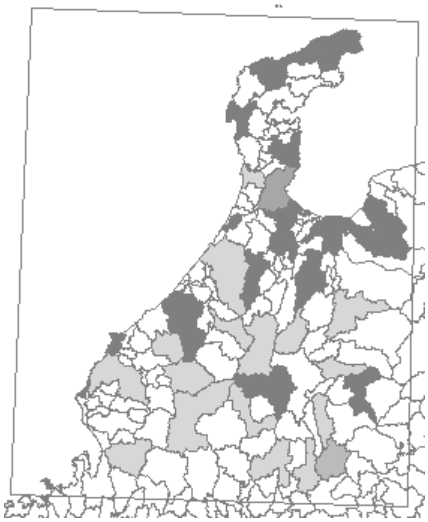

図 5.4: 空間データ統合に基づく気温分布 (市町村区レベルのポリゴンデータを利用)

メッシュ統合に基づく視覚化

同様に，時系列の気温データを取得し，各センサデータオブジェクトごとに平均気温を求める．そして，その結果を，IDW に基づいて処理を行い，メッシュ分布として表示する．

問合せ領域として，$A = \langle 138.52538, 34.516792, 141.06485, 37.109722 \rangle$ を指定した時の，図 5.5 は，IDW に基づくメッシュ統合の結果を示し，図 5.6 は，この領域内のセンサノードの分布を示す．

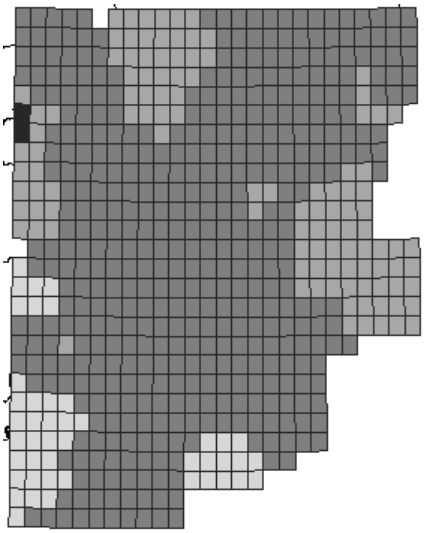

図 5.5: IDW に基づくメッシュ統合による気温分布の表示

図 5.5 では，セルの大きさは 0.1 度四方であるが，図 5.6 に示すように，センサノードの分布が非常に疎であるため，補間処理に必要なデータを確保するために，$max\_cell = 2$ として，25 個のセルに含まれるデータを計算対象としている．

一方，図 5.7 は，図 5.5 と同一のメッシュに対して，4.3.2 節で述べた単純なメッシュ統合を行った場合の結果である．

この手法では，各セルに含まれるセンサデータのみを用いて，それらの平均値を計算し，そのセルの値としているため，図 5.7 からわかるように，値を持たないセルが多数出現することになる．センサデータの分布密度が表示範囲に対して小さく，メッシュのセルサイズが小さいほど，この傾向は強くなる．それに対して，本論文で実装した IDW に基づくメッシュ統合の手法は，図 5.5 に示すように，離散的に分布したセンサデータを連続的な面分布として捉えることができるので，全体の傾向を把握する上で効果的であると考える．

空間補間によって求められたセンサデータの値は，あくまで推定値なので，その場所にセンサが設置されている場合と比較すれば，信頼性は高いと言えない．しかし，全体的な傾向を把握するという点では有用であり，ユーザ自身が経験した事実

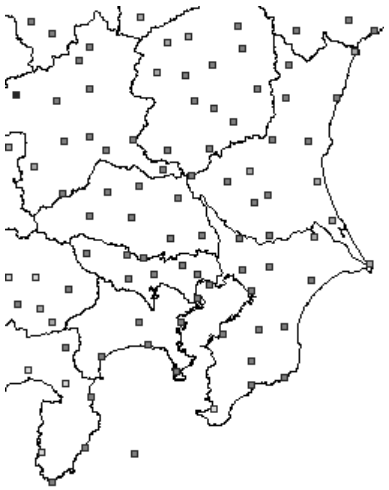

図 5.6: センサノードの分布

図 5.7: 単純なメッシュ統合による気温分布の表示

を裏付けられるような情報が提供できるのであれば，それが推定値であっても十分に意味があると考える．今後，様々なセンサデータリソースが利用できるようになり，センサデータの分布密度が高くなっていけば，IDW の計算結果は近隣のセンサデータ数に依存するので，信頼性は向上していくと考えられる．

オーバーレイ処理に基づく視覚化

気温データと降水量データに対して，オーバーレイ処理を適用する場合について調べる．問合せ領域として $A = \langle 136.74, 34.46, 141.11, 38.03 \rangle$ をクエリに指定し，気温データを管理するサーバと降水量データを管理するサーバから，24 時間分の時系列データを取得した時の，センサデータ統合の結果を図 5.8 に示す．気温データについては 24 時間の平均気温，降水量データについては 24 時間の積算降水量を計算し，その計算結果を利用して，各セルの値を IDW に基づいて計算している．さらに，各セルごとに，気温データについて $\{AVE(temp) \geq 25\}$，降水量データについて $\{SUM(prec) \geq 10\}$ という条件を指定している．したがって，図 5.8 は，「平均気温が 25 °C 以上で，降水量が 10mm 以上の地域を表示せよ.」というクエリに対する結果を示していることになる．

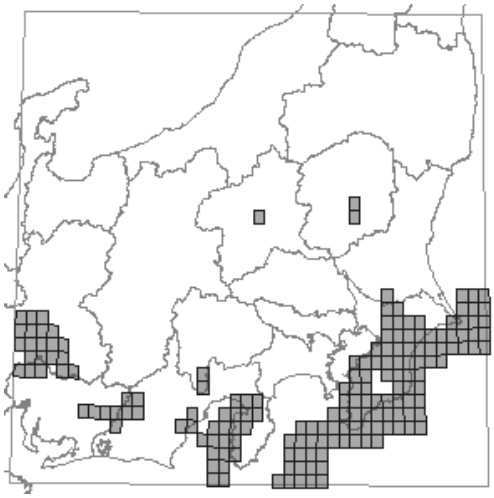

図 5.8: オーバーレイ処理に基づくセンサデータ統合の結果

また，図 5.8 を表示する際の，気温データおよび降水量データに対するメッシュ統合の結果を，図 5.9 に示す．

図 5.9 左図は，気温データに対して $AVE(temp) \geq 25$ という条件を指定しており,「平均気温が 25 °C 以上の場所は，どこか？」というクエリに対する結果を示している．図 5.9 右図は，降水量データに対して $SUM(prec) \geq 10$ という条件を指定した時の結果であり,「10mm 以上の雨が降った場所は，どこか？」というクエ

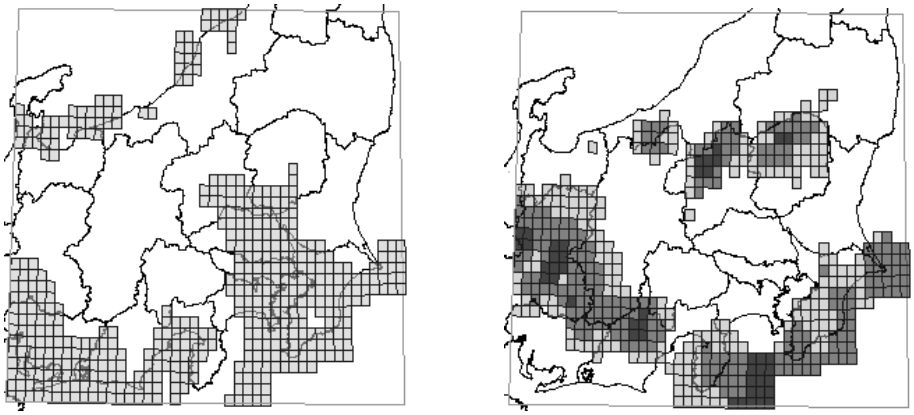

図 5.9: 条件検索の結果 (左図：平均気温分布，右図： 積算降水量分布)

リに対する結果を示している．

図 5.9 と，それらのオーバーレイ処理の結果である図 5.8 を比較することで，気温と降水量に関する条件をともに満たしている地域が結果として出力されていることがわかる．たとえば，図 5.9 右図より，降水量の多い地域が南北にあることがわかるが，図 5.8 より，気温データとの統合の結果，南部の地域で「気温が高く，降水量が多い」ことがわかる．

オーバーレイ処理に基づくセンサデータ統合を行う場合には，クライアント側で空間補間の結果に対して条件検索を適用することに意味がある．仮に，サーバ側で条件検索を行う場合には，クライアント側の空間補間に必要なセンサデータ (点データ) も，サーバ側で取り除かれてしまうために，図 5.8 に示す結果は得られないと考えられる．

オーバーレイなどの空間集約処理に基づいて，センサデータを領域を持つ表現に変換することで，センサデータ同士だけでなく，他の空間データとの包含関係や交差関係を調べやすくなるため，様々なセンサデータ統合が実現できると考えられる．たとえば，国土数値情報 [国土交通省 a] には，標高データや各種統計データなどのメッシュ形式で表現された空間データも含まれるが，これらと同じ粒度のメッシュデータにセンサデータを変換した上で，オーバーレイ処理を適用することで，直接的な関連付けが可能である．これにより,「標高が高く，気温が低い場所は，どこか？」,「都市部で大気汚染度が高い場所は，どこか？」といったクエリに対応することができると考える．

5.2 移動履歴を利用したセンサデータ閲覧手法

本節では，ユーザの移動履歴を利用したセンサデータの閲覧方法について述べる．移動履歴は，ユーザの移動空間や生活空間を表す個人的かつ具体的なデータであり，ユーザの滞在，通過した場所や時刻を如実に表しているため，ユーザ自身の行動や経験に関連した環境情報を取得する際に，非常に有用であると考える．5.1節で述べた空間集約に基づくセンサデータ視覚化システムにおいて，ユーザの移動履歴データを活用することで，より直接的かつ直感的なセンサデータ要求を実現する．また，本節では，センサデータをユーザの移動空間や生活空間と関連付けること，さらには，他の空間情報と関連付けることを “センサデータマッピング” と呼ぶ．デジタル写真や日記，Web 文書の中には，地理的な位置情報と関連付けられるものも多いが，そういった空間情報とのマッピングについても扱う．

5.2.1 移動履歴データの利用

本論文では，ユーザの移動履歴 ($MovingPath$) を式 (5.7) のように表現する．

$$MovingPath = \langle Node_1, ..., Node_k \rangle \tag{5.7}$$

$$Node_i = \{L_i, t_i\} \ \ (1 \leq i \leq k) \tag{5.8}$$

k は，移動履歴を構成する記録点の数である．$Node_i$ は履歴の i 番目の要素であり，L_i は時刻 t_i におけるユーザの位置であり，本論文では，緯度経度座標で表現されるものとする．

重要な点は，移動履歴データが時間属性を持ち，場所に関する情報とともに記録されていることである．ユーザ自身が滞在，通過した場所に関する環境情報を知るためにセンサデータを取得したい時，場所と時刻の両方を覚えている必要があるが，必ずしも記憶しているとは限らない．地図上で場所はわかるが，滞在していた時間がわからない，あるいは，通過した時刻はわかるが，その場所を指定することができない，という状況が発生すると考えられる．そのような場合でも，移動履歴を調べることによって，場所から時刻，あるいは時刻から場所を絞り込むことができる．

5.2.2 センサデータ閲覧システムの構成

図 5.10 に，センサデータ閲覧システムの構成とマッピング処理の概略を示す．センサデータ閲覧システムは，2 つの処理コンポーネントから構成される．

クエリ生成コンポーネント (query component) は，まず，ユーザの入力に基づいて移動履歴を選択する (*select moving path*)．次に，選択された移動履歴に基づい

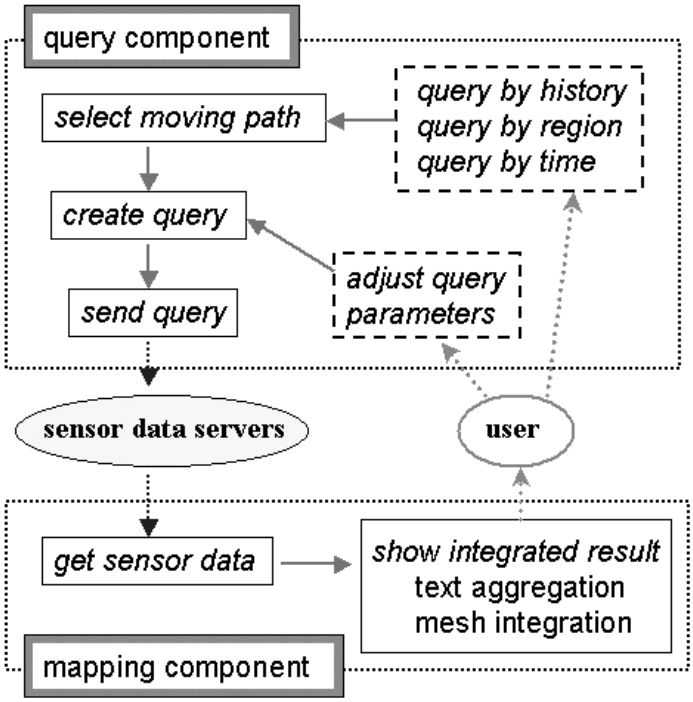

図 5.10: センサデータマッピングのための処理手順

てセンサデータを取得するためのクエリを生成する (*create query*)．生成されたクエリはセンサデータ検索システムに送信される (*send query*)．マッピングコンポーネント (mapping component) は，センサデータ検索システムから取得したセンサデータについて統合処理を行い，その結果をユーザに提示する．ユーザは，その表示結果に基づいて，移動履歴の選択，あるいは先に生成されたクエリパラメータの調整 (*adjust query parameters*) を行う．この一連の流れによって，センサデータの閲覧を対話的に進めることができる．

5.2.3　クエリ生成コンポーネント

移動履歴データの選択

移動履歴データ ($MovingPath$) から，2 つのノード ($Node_p, Node_q$) を選択することで，部分履歴 ($MovingPath'$) が定まる．

$$MovingPath' = \langle Node_p, ..., Node_q \rangle \quad (1 \leq p,\ \ q \leq k,\ \ p \leq q) \tag{5.9}$$

そして，$MovingPath'$ のすべての要素の位置 L'_i $(p \leq i \leq q)$ を覆う最小包囲矩形を $R_{cover} =< x_{min}, y_{min}, x_{max}, y_{max} >$ と表現する (図 5.11)．同様に，すべての要素の時刻 t'_i を含む時区間を $TI_{cover} = [TI_{cover}.min, TI_{cover}.max]$ と表現する．

部分履歴 $MovingPath'$ について，$Node_p = Node_q$ が成り立つ場合には，選択履歴は点として表現される．逆に，始点と終点に同じノードを選択すれば，その地点に対するクエリを生成することができる．

本論文では，移動履歴の選択，すなわち，部分履歴の始点 $(Node_p)$ と終点 $(Node_q)$ を選択するために，次のいずれかの方法を用いる．

- **query-by-history**
  最も基本的な方法であり，履歴中から直接，$Node_p, Node_q$ を選択する．

- **query-by-region**
  領域 (R_{select}) を指定することで，R_{select} に含まれるノードの中で，最も古い時刻のノードを $Node_p$，最も新しい時刻のノードを $Node_q$ とする．この時，部分履歴の要素 $Node'_i$ の位置 L'_i について，$L'_i \in R_{select}$ が成り立つ．また，$R_{cover} \neq R_{select}$ であり，必ずしも同じ領域とはならない．

- **query-by-time**
  時区間 (TI_{select}) を指定することで，TI_{select} に含まれる時刻を持つ移動履歴中のノードを選択する． この時，$TI_{select}.min \leq t'_i \leq TI_{select}.max$ が成り立ち， $TI_{select}.min \leq TI_{cover}.min$, $TI_{cover}.max \leq TI_{select}.max$ である．

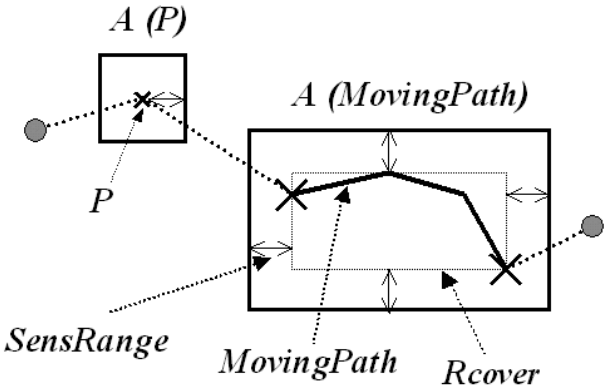

図 5.11: 移動履歴に基づくクエリの生成：$A(MovingPath)$…選択された部分履歴に対する問合せ領域，$A(P)$…選択地点に対する問合せ領域)

クエリパラメータの決定

センサデータを要求するためには，次の 3 つのクエリパラメータを決定する必要がある．

- データの種類：DT
- クエリ領域：A
- 時区間：TI

DT については，何らかの方法で指定するとして，ここでは，A と TI の決定方法について述べる．

A は，R_{cover} と クエリ領域の幅を決定するパラメータ $SensRange$ によって，式 (5.10) のように表現される (図 5.11)．

$$\begin{aligned} A &= \langle x_{min} - SensRange, y_{min} - SensRange, \\ &\quad x_{max} + SensRange, y_{max} + SensRange \rangle \end{aligned} \tag{5.10}$$

なお，本節の実験では，$SensRange$ は，ユーザによってあらかじめ入力されるものとするが，図 5.10 に示したセンサデータ閲覧の過程で，インタラクティブに変更することも可能である．

$MovingPath'$ が点 $P = (x', y')$ として表現される場合は，式 (5.10) において，x_{min} と x_{max}，y_{min} と y_{max} がそれぞれ等しい場合に相当する．したがって，問合せ領域 $A(P)$ は，P の座標と $SensRange$ のみから，式 (5.11) のように計算できる (図 5.11)．

$$\begin{aligned} A(P) &= \langle x' - SensRange, y' - SensRange, \\ &\quad x' + SensRange, y' + SensRange \rangle \end{aligned} \tag{5.11}$$

TI を決定する際には，指定した DT のセンサデータの更新間隔を考慮して，$TI_{cover} \subseteq TI$ を満たす時区間 TI を指定する．この時，$TI.min \leq TI_{cover}.min$, $TI_{cover}.max \leq TI.max$ が成り立つ．

クエリの送信と制御

クエリ生成コンポーネントは，クエリ $Q = \langle DT, A, TI \rangle$ をセンサデータ検索システムに送信する．検索結果であるセンサデータは，マッピングコンポーネントにより加工される．ただし，地点 P に対する問合せ領域 $A(P)$ を指定した時に，その範囲内にセンサデータが存在しない場合には，さらに，$SensRange$ の分だけク

エリ領域を広げて，再度クエリを送信し，センサデータが得られるまで，この処理を繰り返す．この領域拡大に基づくセンサデータ要求によって，最初に指定した $SensRange$ の値が小さい場合にも，柔軟に対応することができる．

クエリパラメータの調整

移動履歴に基づくクエリパラメータの生成では，移動履歴中の時間と位置の依存関係を利用して，クエリ領域と時区間パラメータを決定している．ただし，この枠組みだけでは，移動履歴のある地点における過去のセンサデータを参照することや，同じ時刻の他の地点のセンサデータを参照するといった派生的なクエリに対応することができない．そこで，移動履歴に基づいて生成されたクエリパラメータを，次のいずれかの方法で調整できるものとする．

- クエリ領域を固定したまま，時区間パラメータを変化させる．これにより，ある地点もしくは範囲に対する異なる時刻や時間帯のセンサデータを要求することができる．
- 時区間パラメータを固定したまま，クエリ領域を変更する．これにより，ある時刻もしくは時区間に対する異なる地点や領域のセンサデータを要求することができる．

5.2.4 マッピングコンポーネント

マッピングコンポーネントは，式 (5.12) のように表現されたセンサデータを受信する．

$$SensResult = \bigcup_{m=1} o_m \tag{5.12}$$

センサデータオブジェクト o_m は，センサノードごとに，$\{type, loc, series\}$ の組で表現される．$o_m.type$ はセンサノード m が出力するセンサデータの種類，$o_m.loc$ はノード m の位置である．検索結果は問合せ領域 A に含まれるため，$\forall m\ o_m.loc \in A$ が成り立つ．$o_m.loc$ もまた，緯度経度座標で表現されているものとする．$o_m.series$ は，ノード m に対する時系列データであり，式 (5.13) のように表現される．

$$o_m.series = \bigcup_{n=1} TemporalItem_{m,n} \tag{5.13}$$

$TemporalItem_{m,n} = \{value_{m,n}, update\_time_{m,n}\}$ と表現する．$value_{m,n}$ は，時刻 $update\_time_{m,n}$ におけるセンサノード m の出力値である．また，$\forall m, n\ \ TI.min \leq update\_time_{m,n} \leq TI.max$ が成り立つ．

受信したセンサデータの種類 ($o_m.type$) を判別することによって，時系列データ $o_m.series$ の処理が決定される．各時刻の観測値である気温データなどは，$o_m.series$ に含まれる各データの平均値が計算される．一方，過去 1 時間の合計量を示すような降水量データの場合は，各時刻に対する値を積算していく．そして，このような時間集約処理を行った後で，地理的関連性を利用した処理を行う．

空間補間に基づく表示情報の生成

マッピングコンポーネントは，履歴中のある地点 (P) に対するクエリを送信した時には，矩形領域 $A(P)$ に含まれるすべてのセンサデータを利用して，$A(P)$ の中心点 ($= P$) の値を補間する．補間の方法としては，IDW (Inverse Distance Weighted) を利用する．

選択された移動履歴に対するクエリを送信した時には，5.1.2 節の「空間補間に基づくメッシュ統合」の手順と同様に，A をメッシュに分割し，IDW に基づいて各セルの値を計算し，メッシュ分布を生成する．

空間データ統合に基づく位置情報抽出

前節の地点 (P) に関するセンサデータを求めた時，P に関する位置は，緯度経度座標で表現されている．しかし，その計算結果を位置情報とともに，テキストデータとして提示する場合には，緯度経度よりわかりやすい位置表現が必要となる．本論文では，移動履歴と各種空間データとの地理的関係性を調べることにより，位置情報を抽出する．対象とする空間データがポリゴンデータの場合には，(i) 点位置決定問題，(ii) 線分交差問題 という地理情報処理の基本アルゴリズムを利用することになる [伊理 99].

5.2.5　実装

プロトタイプシステムの実装

実験に用いたプロトタイプシステムについて述べる．実験システムは，Java 言語を用いて実装した．システムのユーザインタフェースの一部を図 5.12 に示す．

地図表示部 (map area) は，行政区や鉄道などの地図データ上に，移動履歴 (moving path) やクエリ領域 (query region) を重ね合わせて表示するだけでなく，センサデータ統合の結果を分布として表示する．センサデータより生成されたテキスト形式の環境情報は，時刻や位置情報とともに，テキスト情報表示部 (text information area) に表示される．センサデータ要求のためのクエリを生成する際には，履歴表示リスト (log point list) での直接的な始点／終点ノードの選択 (**query-by-history** に対

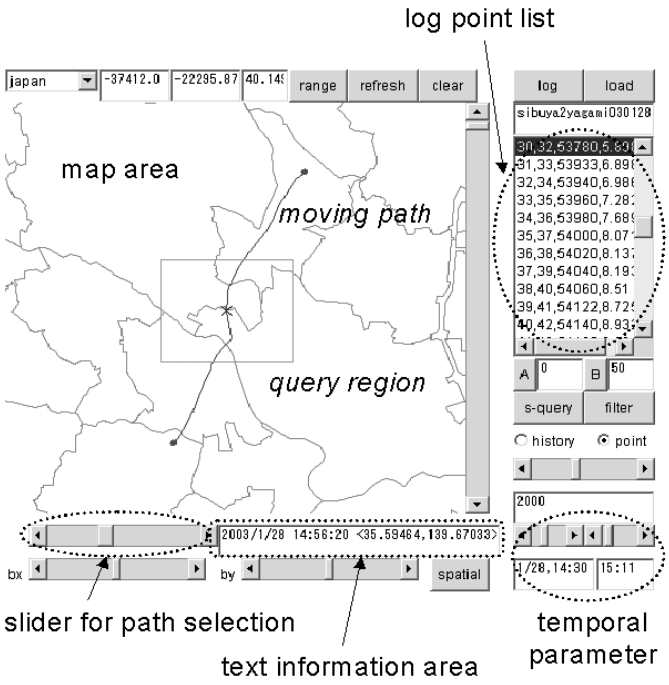

図 5.12: 実験システムの GUI イメージ

応)，地図上でのマウスドラッグによる領域の選択 (**query-by-region**)，時刻指定部 (temporal parameter) での時区間指定 (**query-by-time**) のいずれかを行うことによって，移動履歴データの一部が選択される．また，履歴選択スライダー (slider for path selection) により，移動履歴に沿ったセンサデータの要求と統合結果の閲覧が可能である．

データセット

実験で利用した各種データについて述べる．

- 移動履歴データ
  図 5.13 に示す次の 2 種類の移動履歴データを実験に利用した．
  - (A) 東京近郊での移動履歴データ： GPS を利用して測位
  - (B) より広範囲にわたる移動履歴データ：
    履歴データの生成にジオコーディングを利用

(A) は，ハンディ GPS レシーバー (PCQ-HGR3S，SONY 社製) を利用して，移動中の緯度経度座標を時間情報とともに連続的に記録したものである．移動ログの記

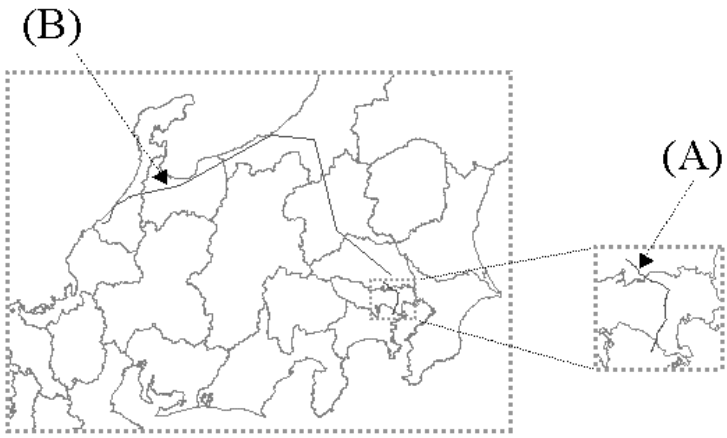

図 5.13: 実験に利用した移動履歴データ

録間隔は，20 秒としており，本実験で用いた履歴データは，185 個の記録点から構成されている．

(B) は，鉄道などを利用して移動した時の履歴データを，GPS を用いず，鉄道の時刻表情報などに基づいて生成したものである．具体的には，時刻表から停車駅の名前と停車時刻を抽出し，駅の住所情報に CSV アドレスマッチングサービス [CSIS] を利用して緯度経度情報を付加することで，電車を利用している際の移動履歴を生成した．このような住所などの位置情報を，地図に直接対応付けられる座標値に変換することを，一般に，ジオコーディングと呼ぶ．鉄道を利用していない部分の履歴については，地図ソフトウェアの地図表示部で，マウスポインタを該当箇所に移動させ，その時に表示される座標値 (緯度経度) を利用した．このデータは実測値ではないが，実際の移動行動を反映しており，滞在地点や通過地点の時刻と位置が記録されているため，(A) と同様に，移動履歴データとして利用することが可能である．(A) との大きな違いは，移動履歴の記録時刻の間隔が一定でない点と，より広い範囲にまたがった履歴データであるという点である．実験で用いたデータは，24 個の記録点を含む．

- センサデータ

移動履歴と対応付けるセンサデータとして,「アメダス観測年報 (2000 年)」を利用する．センサデータは 2000 年の観測値であるが，同月日，同時刻の観測データを移動履歴と対応付けるものとする．ただし，ユーザの移動時の観測データが入手できれば，同様の結果が得られると考える．

- 空間データ

地図表示や位置情報抽出に用いる空間データとして，国土数値情報 [国土交通省 a] を利用する．具体的には，国土数値情報に含まれる「行政界・海岸線データ (平成 11 年度)」,「鉄道データ (平成 7 年度)」,「道路データ (平成 7 年度)」を利用している．

5.2.6 実験結果

単純選択に基づくセンサデータマッピング

まず，移動履歴データ (A) を用いた時の実験結果についてまとめる．図 5.14 に対象とする移動履歴 (moving path) と周辺の気温ノード (temperature node) の分布を示す．履歴データは，図の左上の地点から下方向に向かう移動を示す．(具体的には，埼玉県から東京都を通過し神奈川県まで移動した時の履歴を表している．) 図 5.14 の各矩形は，それぞれクエリ領域を示しており，外側の矩形は全移動履歴に対するクエリ領域を表し，履歴中の 3 つの地点に対するクエリ領域がその内側に含まれている．この例では，$SensRange = 10\text{km}$ としており，各地点に対するクエリ領域の大きさは，約 20km 四方となっている．

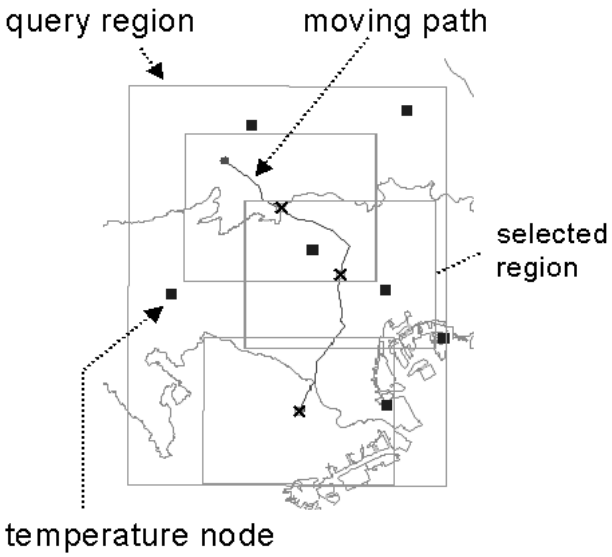

図 5.14: クエリ領域と気温ノード分布

本実験では，**query-by-history** によって，履歴中の任意の地点に関する気温データを要求する場合について調べた．図 5.15 に，図 5.14 の選択領域 (selected region) を問合せ領域とするクエリを送信した時の結果を示す．

図 5.15 では，図に示すクエリ領域 (selected region) に含まれる気温データを利用して，指定した地点の気温データを空間補間に基づいて計算している．そして，そ

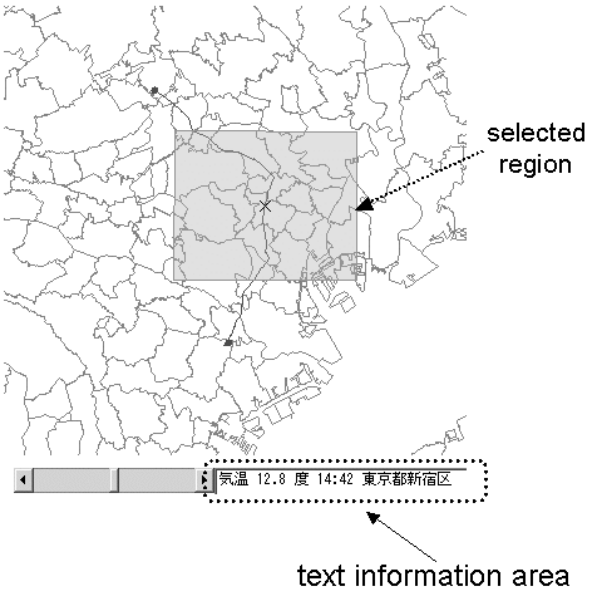

図 5.15: 空間補間に基づく集約結果とテキスト情報の生成

の集約結果を，テキスト情報として，時刻情報および位置情報とともに，テキスト情報表示部 (text information area) に表示している．図 5.14 の各地点のクエリに対して生成されたテキスト情報を移動履歴の時刻順に並べたものを，次に示す．

気温 12.7 度 14:20 埼玉県和光市
気温 12.8 度 14:42 東京都新宿区
気温 12.3 度 15:20 神奈川県横浜市港北区

通常，GPS を用いて測位した場合は，各記録点の位置は，緯度経度で表現される．しかし，地図上で対応関係を把握できる場合を除いて，数値情報として緯度経度座標を表示されても，その位置関係を把握することは難しい．そこで，本実験では，移動履歴と関連する行政区ポリゴンを選択し，その属性情報である行政区名を調べることで，指定した地点を含む行政区の名前を位置情報として出力している．本実験では，国土数値情報の市町村レベルの行政区ポリゴンデータを利用し，履歴データとの交差関係を調べた．その結果を，図 5.16 に示す．

本実験では，行政区ポリゴンデータを利用した空間データ統合に基づいて，履歴中のある点に関する位置情報を抽出しているが，図 5.14 の移動履歴は，鉄道での移動時の記録を含んでいるため，当然のことながら，鉄道データとの地理的関連性が高い．したがって，たとえば，国土数値情報の鉄道データに含まれる駅に関する属性情報を利用すれば，位置情報として，記録点に近い駅の名前を出力することも可能であると考えられる．

図 5.16: 移動履歴に基づく行政区ポリゴンの選択

本実験システムでは，地図表示部とテキスト情報表示部の両方に空間補間に基づく計算結果を表示しているが，実際には，テキスト情報表示部に表示されている結果のみをユーザに提供することも可能である．たとえば，携帯端末での利用を考えた場合には，地図を表示するのに十分な画面が確保できず，地図上でのセンサデータ分布表示が困難な場合があるが，そのような場合に，テキスト形式での環境情報の提供が効果的であると考える．

領域選択に基づくセンサデータマッピング

移動履歴データ (B) を用いて，気温データをマッピングする場合について調べる．図 5.17 に，実験で用いる履歴データと，全履歴データに対するクエリ領域 ($SensRange = 50\text{km}$) に含まれる気温ノードの分布を示す．

図 5.18 は，図 5.17 の全移動履歴データに対するメッシュ統合の結果を表す．図中のセルの濃淡は，気温の違いを示しており，図の右に示した凡例 (単位は，°C) に従う．凡例によると，図 5.18 の左下から右上にかけて，気温の低い地域が存在することがわかる．この例では，履歴中の記録点のすべての時刻を含む 15:00 から 23:00 までの観測データを利用して，各ノードごとに平均値を求め，その結果を利用して空間補間を行っている．したがって，図 5.18 は，移動履歴に関連する時系列のセンサデータを時空間的に集約した結果であると言え，全体的な傾向を表していると考えられる．この場合，空間的なデータの分布状況を把握するだけでなく，時系列データの傾向を予想することもできる．たとえば，平均気温の低い地域は，短い時間帯で見た場合にも気温が低く，ある時刻に大量に雨が降っていれば，集約

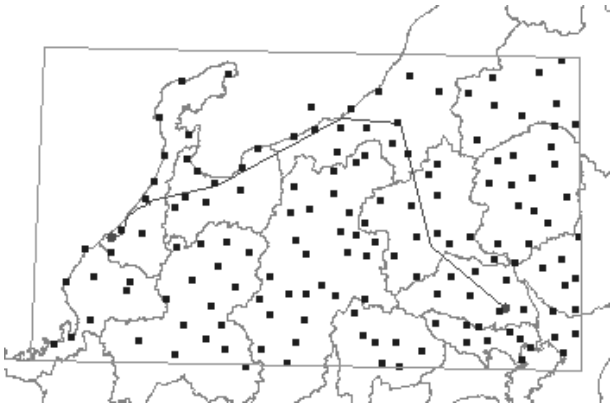

図 5.17: 移動履歴と気温ノードの分布

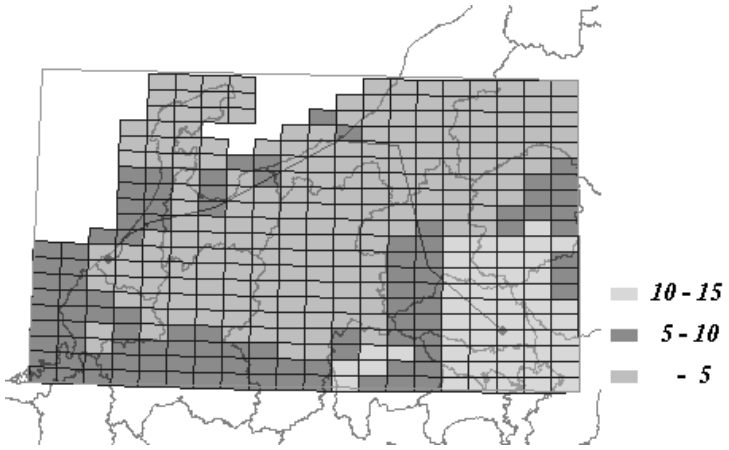

図 5.18: メッシュ統合に基づくセンサデータ分布表示

結果である積算降水量データも多くなると考えられる．

図 5.19 は，**query-by-region** によって図 5.17 に表示されている移動履歴のある区間を選択し，その部分履歴に対するデータを要求した時のメッシュ統合の結果を示している．この図では，気温データの凡例 (単位は，°C) を図の左上に配置している．

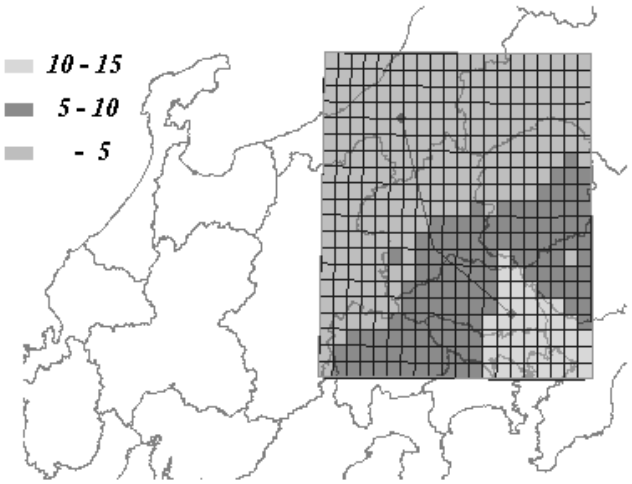

図 5.19: 選択領域のメッシュ統合に基づくセンサデータ分布表示

図 5.19 では，図 5.18 より狭い範囲のクエリ領域を 20×20 のメッシュに分割しているため，より細かな分布傾向が示されている．ただし，**query-by-region** に基づくセンサデータマッピングは，領域だけでなく，結果として時区間も絞り込んでいくため，図 5.19 は，図 5.18 より短い時区間 (20:00〜23:00) に含まれるセンサデータを利用した結果となっている．図 5.18 は昼間から夜にかけての平均値であるが，図 5.19 は，夜間の平均値であるため，全体的に気温が低く，図 5.18 と比較すると，10 °C 以下の領域が広がっていることがわかる．そして，**query-by-region** を繰り返すことで，最終的には，図 5.20 に示すように，集約結果として，ある地点のセンサデータを獲得することができる．図 5.20 は，図 5.19 に表示されている移動履歴のある点を選び，その地点に関するセンサデータを要求した時の結果を示している．この図では，より短い時区間 (21:00〜22:00) のデータを利用している．

対話的なセンサデータマッピング

query-by-region は，基本的に，ユーザが滞在，通過した移動空間に関する環境情報を知りたい時，場所に関する情報は記憶しているが，その場所に居た時刻を記憶していない場合に効果を発揮する．さらに，図 5.18 → 図 5.19 → 図 5.20 に示したように，センサデータの分布状況を閲覧し，全体的な傾向を把握しながら，そ

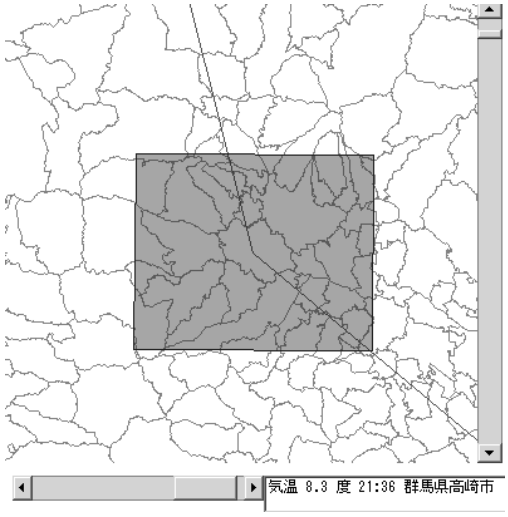

図 5.20: 選択領域に関するセンサデータのテキスト表示

のデータ分布に基づいて領域選択を繰り返すことで，より具体的に，ある地点に関する環境情報を取得することが可能である．たとえば，鉄道で移動中に車窓の外で雪が積もっていることや雨が降っていることだけを記憶しているが，それが，いつどこで観察された現象なのかを記憶していない場合に，**query-by-region** による対話的なセンサデータ閲覧により，その場所と時刻を特定することができる．つまり，移動履歴に基づくセンサデータ統合の結果から場所と時間を推論するという，通常のセンサデータの検索とは逆方向の情報提供ができることを示唆している．

また，センサデータを閲覧していく過程で，提示された統合結果から，履歴中の別の地点や別の時刻のセンサデータに関心を持つ場合もあると考えられる．たとえば，目的地にたどり着いた時に出発地の気温を知りたい場合や，ある地点でのそれまでの気象情報の変化を知りたい場合もあると考えられる．ある場所を訪れた時に，雪が積もっていることが観察されれば，前日までの降雪量を調べることで，その事実を再確認することができる．このような要求に対しては，5.2.3 節で述べたクエリパラメータの調整を行ってからクエリを発行することで対応できる．この機能を利用することで，クエリ領域と時区間を独立に指定することが可能である．

たとえば，図 5.21 は，図 5.18 を求めるために生成したクエリパラメータの中で，時区間パラメータのみを変更し，より部分的な時区間 15:00〜16:00 に対して統合処理を行った結果を示している．

凡例によると，図 5.21 において気温の低い地域が，全体的な傾向を示す図 5.18 においても気温の低いことが確認できる．この例では，広範囲にわたるセンサデータ分布に対して，時区間パラメータを変更した場合を示しているが，履歴中のある 1 つの地点に関して，別の時刻のセンサデータを参照することも可能である．

移動履歴に基づいてクエリを生成し，その上でパラメータの調整を対話的に進め

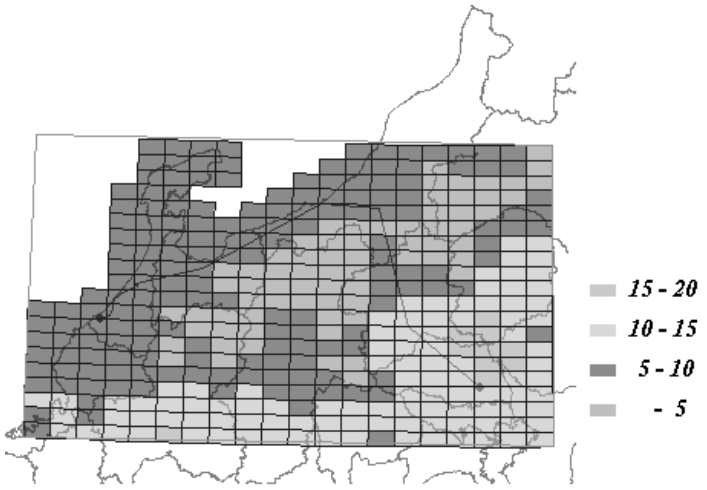

図 5.21: 時区間パラメータを変更した時のメッシュ統合に基づくセンサデータ分布

ていくことは，何も手がかりのない状態からクエリパラメータを模索していくことに比べて，ユーザの負担を少なくできると考えられるので，柔軟なセンサデータの閲覧を実現する上でも有用であると考える．

5.2.7 デジタル写真に対するセンサデータマッピング

5.2.5 節のプロトタイプシステムを拡張し，デジタルカメラで撮影した写真 (デジタル写真) に対して，関連するセンサデータをマッピングするサンプルアプリケーションを作成した．そのユーザインタフェースの一部を図 5.22 に示す．

このアプリケーションでは，図 5.12 のプロトタイプシステムと同様に，移動履歴データを読み込み，その結果を履歴表示リスト (log point list) に表示し，軌跡を地図上に表示することができる．図 5.22 では，図 5.17 と同じ移動履歴を扱っている．さらに，写真リスト表示部 (photo list) に表示されている写真を選択すると，その画像が写真画像表示部 (photo area) に表示され，その撮影時刻情報が写真関連情報表示部 (photo info) に表示される (図 5.22)．

各写真データは，その撮影時刻に基づいて，最も近い記録時刻を持つ移動履歴の 1 つの要素 (地点) と関連付けられる．これにより，移動履歴を選択すると，その構成要素と関連するすべての写真がリストとして photo list に表示される．逆に，photo list 中の写真要素を選択すると，log point list 中の 1 つの要素が定まる．すなわち，photo area に表示されている写真に対する履歴要素が選択される．図 5.22 では，その選択された履歴要素 (selected point) の位置情報 (経緯度座標) がテキスト情報表示部に表示されている．

図 5.23 は，図 5.22 に表示されている写真データの撮影時の気温を求めるために，

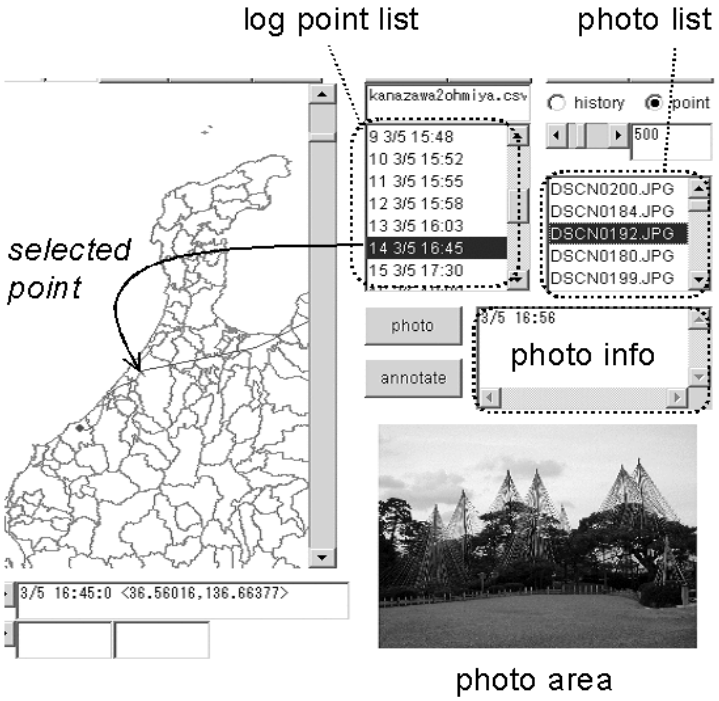

図 5.22: サンプルアプリケーションの GUI イメージ

対応する移動履歴中の要素に基づいてクエリを生成し，そのクエリに対する検索結果を利用して統合処理を行った結果を示している．

図 5.23 のテキスト情報表示部 (text info) に表示されているように，指定した地点の気温データが求められていることがわかる．さらに，図 5.23 の [annotate] ボタンを押すことで，統合結果である気温データが写真データにマッピングされる．その結果として，写真関連情報表示部 (photo info) には，図 5.22 で示した撮影時刻情報に加えて，選択地点の経緯度座標と統合結果である気温データの値が表示される．この実験例は，移動履歴とデジタル写真群を関連付けた上で，提案手法を用いることで，写真データに対するセンサデータマッピングが実現できていることを示している．

ただし，センサデータをマッピングするために利用した履歴要素の座標は，その要素の記録時刻と写真の撮影時刻が近いだけであって，厳密に言えば，写真を取った場所に対応しているわけではない．この時，写真を撮影したことや撮影場所を訪れたことは，ユーザの記憶 (あるいは行動記録) として残されているはずなので，センサデータ統合の結果を写真データに実際にマッピングするかどうかは，ユーザの判断に委ねられる．もし，選択地点と写真の時刻の間隔や，選択地点に滞在していた時間の長さを考えて，ユーザが適切であると判断できる範囲内であれば，[annotate] ボタンによって，センサデータと写真データのマッピングが行われる．

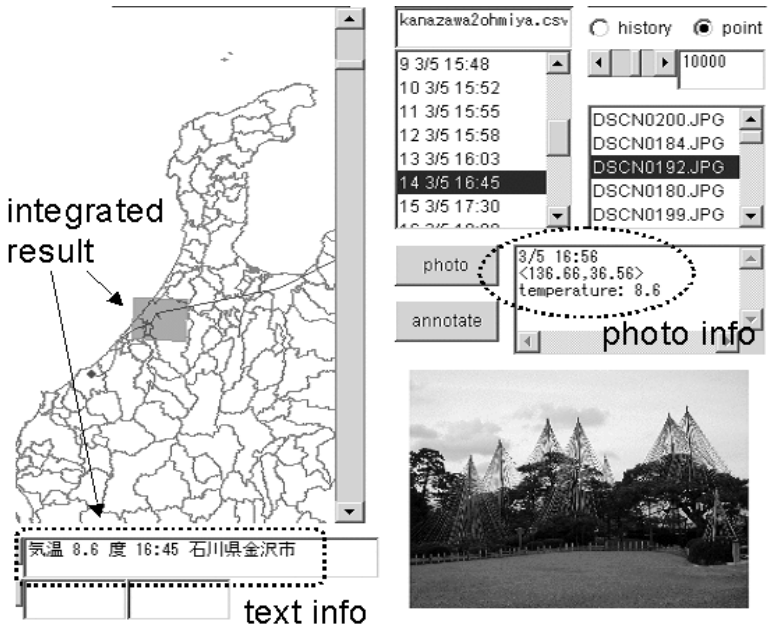

図 5.23: 写真データへのマッピング

移動履歴データ (A) のように，GPS を用いて短い間隔で記録された移動履歴データが利用できれば，十分に近い場所と時間での集約結果の利用が可能である．

また，このアプリケーションでは，写真データを扱っているが，日記やブックマークなどのテキストデータやビデオカメラで撮影した映像データでも，移動履歴中の要素と関連付けることができれば，同様のマッピングが可能である．さらに，写真は，行動日記などとともに管理されることも考えられ，写真に対するセンサデータマッピングを対話的に進めた結果から，日記に記述する情報が得られる場合もあると考えられる．提案手法によって，地理的な位置と関連付けられる写真や行動記録などの個人的な空間情報に対して，その時，その場所のセンサデータを説明情報として付加することが可能である．写真データを一種の空間コンテンツと考えると，写真データをセンサデータと関連付けることで，時間情報や位置情報の他に，センサデータを利用したコンテンツ検索が実現できると考えられる．

5.3 まとめ

本章では，まず，空間集約処理に基づくセンサデータの視覚化手法について述べた．具体的に，センサデータ視覚化システムの機能として，空間データ統合，空間補間，オーバーレイ処理に基づく集約手法を実装した．この空間集約に基づく視覚化手法は，複数のセンサデータサーバから収集した異種のセンサデータを統合する

ための枠組みを提供しており，ネットワークを介して収集した様々なセンサデータを組み合わせ，その統合の結果を効果的に視覚化することができる．

さらに本章では，空間集約に基づくセンサデータ視覚化システムにおいて，ユーザの移動履歴に基づいて，センサデータを閲覧するための枠組みを提案した．ユーザの移動履歴を利用することにより，直接的かつ直感的なセンサデータ要求が可能となり，空間集約処理を利用して提示情報を生成することにより，インタラクティブなセンサデータの閲覧が可能となる．移動履歴と空間集約を利用したセンサデータ閲覧の枠組みは，ある地点に関するセンサデータを求める上で有用であるだけでなく，対話的に閲覧を進めていく過程で，移動空間の状態を把握し，生活空間の理解を深める上でも効果的であると考える．

第6章

インクリメンタルなセンサデータ統合手法

本章では，4章のセンサデータ検索システムから，領域ごとに提供されるセンサデータをインクリメンタルに統合する手法を提案する．すなわち，クライアントが，受信したセンサデータをどのように加工し，その結果をどのようにユーザに提示するかという問題を扱う．具体的には，センサデータを統合する方法として，5章で述べた空間集約処理を取り上げ，空間データ統合，空間補間に基づくメッシュ統合，オーバーレイ処理に基づくセンサデータ統合のそれぞれについて，インクリメンタルにセンサデータを処理する方法を示す．これにより，領域分割に基づいて提供されるセンサデータに対して，領域ごとに空間集約処理が行われ，その処理結果がインクリメンタルに表示される．

6.1 センサデータのインクリメンタルな空間集約

4章で述べたセンサデータ検索手法は，クエリに指定した問合せ領域を分割し，分割領域ごとに検索結果を逐次的に提供する．これにより，クライアントは，すべての処理が終了する前に何らかの検索結果を受信することができるため，受信した途中結果を利用して統合処理を開始し，その結果をインクリメンタルに提示することも可能である．さらに，“領域” という意味のあるまとまりで，ユーザの理解しやすい形で途中結果を提示することができれば，処理途中での意思決定や状況判断に役立てることができ，インタラクティブなセンサデータの視覚化システムを構築する上で有用であると考えられる．

しかしながら，5章で述べた空間集約の手法では，ポリゴンデータや補間対象となるセルなど，空間的な範囲を持つデータを扱う．たとえば，領域ごとに提供されるセンサデータと，ポリゴンデータとの統合を考えると，ポリゴンデータが複数の領域にまたがる場合には，ある領域のセンサデータを収集しただけでは，その領域と重なるポリゴンデータの統合処理を終了させることができない．また，その統合

処理の完了を判定する機能や，それまでの途中結果を保持する機能なども必要になる．4 章で提案しているのはデータ提供技術であるため，実際に，空間集約に基づくインクリメンタルな情報提供を実現するためには，センサデータを受信するクライアント側の処理技術を検討することが不可欠である．本章では，次の地理情報処理を支える基本的な統合方法それぞれに対して，インクリメンタルな処理アルゴリズムを考案する．

- 空間データ統合に基づく手法 (6.2.3 節)
- 空間補間に基づく手法 (6.2.4 節)
- オーバーレイ処理に基づく手法 (6.2.5 節)

6.2.3 節で，空間データ統合のアルゴリズムを提案し，6.2.4 節にて，その手順の一部を拡張することでメッシュ統合を実現する．さらに，6.2.5 節で，オーバーレイ処理に基づく枠組みへと拡張を行っている．

6.2　インクリメンタルな空間集約アルゴリズム

6.2.1　クエリの表現

本章では，次のクエリ Q を想定する．

$$Q \quad = \quad \langle A, DT, TI, SC, DC \rangle \tag{6.1}$$

A は問合せ領域，DT はセンサデータの種類，TI は時区間，SC は空間制約である．DC は，センサデータの値に対する条件である．

なお，本章では，時間制約 (TC) は指定しない場合を想定する．したがって，4 章のクエリにおいて，時区間を分割しない場合に相当するため，問合せ領域 A に対する分割領域 a_i は，次のように定義される．

$$\begin{aligned} A \quad &= \quad a_1 \cup a_2 \cup ... \cup a_k \\ &\quad\quad \forall i, j \quad a_i \cap a_j = \phi \quad (i \neq j) \end{aligned} \tag{6.2}$$

a_i は，分割された領域を示す．k は分割数である．

分割領域 a_i は，4 章の式 (4.5)，(4.6) と同じように，a_i^{upper} と a_i^{lower} の差分領域として定義される．

6.2.2 インクリメンタルな統合処理のためのデータ構造

空間集約をインクリメンタルに行うためのデータ構造として，以下の3つのリストを導入する．

- $RESULT\_LIST_i$
  a_i に対して受信したセンサデータオブジェクトを保持する．$RESULT\_LIST_i$ の要素を $o_{i,k}$ とする時，$o_{i,k}.loc \in a_i$ が成り立つ．

- $OVERLAP\_LIST_i$
  $OVERLAP\_LIST_i$ は，a_i と重なり合う空間データオブジェクトの集合を表し，$OVERLAP\_LIST_i$ の要素 $p_{i,j}$ について，$p_{i,j}.geom \cap a_i \neq \phi$ が成り立つ．$p_{i,j}$ は a_i に対するセンサデータオブジェクトの統合対象となりうる．

- $LAST\_OVERLAP_i$
  a_i のセンサデータを収集した時に，空間データオブジェクト p に含まれるすべてのセンサデータの収集が完了する場合に，p を $LAST\_OVERLAP_i$ の要素とする．すなわち，$LAST\_OVERLAP_i$ は，a_i のセンサデータ受信時に，集約処理を完了させることのできる空間データオブジェクトの集合を示す．p は，$p.geom \cap a_i \neq \phi$ が成り立つ最大の領域に関連付けられる．なお，$LAST\_OVERLAP_i \subseteq OVERLAP\_LIST_i$ である．

$OVERLAP\_LIST_i$ および $LAST\_OVERLAP_i$ について，図6.1の例で説明する．図中の $P1, P2, P3, P4$ はポリゴンデータオブジェクト，$O1, O2, O3, O4, O5$ はセンサデータオブジェクトである．

図6.1において，$P1.geom \cap a_i \neq \phi$，$P2.geom \cap a_i \neq \phi$ が成り立つので，$P1$，$P2$ は，ともに，$OVERLAP\_LIST_i$ の要素となる．しかし，a_i のデータ受信時に，集約処理を完了することができるのは，$P1$ のみであるため，$P1 \in LAST\_OVERLAP_i$ である．領域 a_{i+1} については，$P2, P3 \in LAST\_OVERLAP_{i+1}$ となる．

6.2.3 センサデータとポリゴンデータの統合

次の処理手順に従って，空間データ統合を行う．

1 統合データの準備
図5.1の空間データサーバから，空間データオブジェクト p を取得する．

2 集約処理のためのデータ構造の作成
問合せ領域を分割し，各分割領域 a_i に対して，$RESULT\_LIST_i$，$OVERLAP\_LIST_i$，$LAST\_OVERLAP_i$ を作成する．さらに，この段階で，$OVERLAP\_LIST_i$ と $LAST\_OVERLAP_i$ の要素を計算する．

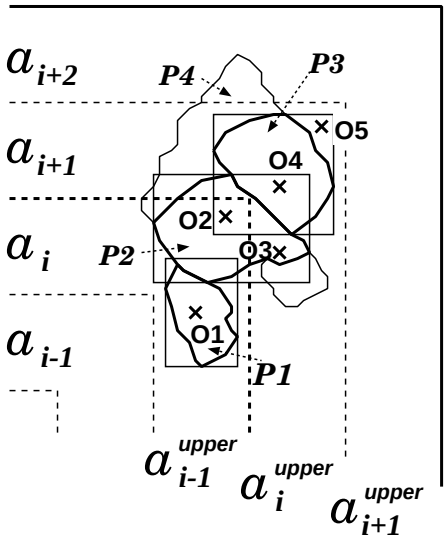

図 6.1: 領域分割に基づくポリゴンデータとの統合例

3 クエリ Q を仲介エージェントに送信する

4 仲介エージェントから検索結果を受信し，処理を行う

最終結果を受信するまで，検索結果を受信するごとに，4.1 と 4.2 の処理を順番に繰り返す．ここで，検索結果は，分割領域ごとに，a_1 から順番に受信され，対応する $RESULT\_LIST_i$ に格納される．

4.1 中間結果の計算

検索結果を受信する度に，$RESULT\_LIST_i$ の要素であるすべてのセンサデータオブジェクト $o_{i,k}$ について，$OVERLAP\_LIST_i$ の要素である空間データオブジェクト $p_{i,j}$ との比較を行う．2.1.3 節で説明した filter&refinement 戦略に従い，$o_{i,k}.loc \in p_{i,j}.rect$ という条件を調べ，条件を満たす $p_{i,j}$ について，$o_{i,k}.loc \in p_{i,j}.geom$ が成り立つ場合に，$p_{i,j}$ の属性として，$o_{i,k}$ の値を用いて中間結果を計算する．

4.2 集約処理の完了

この段階で，a_i に対するすべてのサーバからの結果が受信されていれば，$LAST\_OVERLAP_i$ の要素であるすべての空間データオブジェクト $p_{i,j}$ について，保持されている中間結果を用いて集約結果を計算し，その結果を表示する．この時，集約結果が，センサデータ dt_l に対する条件 dc_l を満たしているかどうかについても評価する．

$OVERLAP\_LIST_i$ は，センサデータオブジェクトを関連付けられるポリゴンデータオブジェクトを探し出す際の探索コストを軽減するためのデータ構造であ

り，$LAST\_OVERLAP_i$ は，ポリゴンデータオブジェクトの集約処理の完了を判定するためのデータ構造である．

6.2.4 空間補間に基づくメッシュ統合

次に，インクリメンタルなメッシュ統合の手順について示す．メッシュ統合は，5.1.2 節で述べたように IDW に基づいて行われる．セル c を空間データオブジェクト p と置き換えることで，メッシュ統合の各ステップを記述できる．ここでは，6.2.3 節の [1]，[4.1]，[4.2] に対応する処理が異なるため，それらのステップについて述べる．

1′ 統合データの準備
セルの大きさを指定し，問合せ領域 A に対するメッシュデータを生成する．セル c は，メッシュデータを構成する空間データオブジェクトと言える．

4.1′ 中間結果の計算
センサデータオブジェクト $o_{i,k}$ $(\in RESULT\_LIST_i)$ について，$o_{i,k}.loc \in c_{i,j}.rect$ が成り立つ時，$c_{i,j}$ の属性として，$o_{i,k}$ ごとに，観測値 (もしくは時系列集約値) と位置情報 $(o_{i,k}.loc)$ を保持する．

4.2′ 集約処理の完了
a_m $(m = 1, 2, ..., i)$ に対する $LAST\_OVERLAP_m$ の要素 $c_{m,n}$ について，$cover\_rect(c_{m,n}) \subseteq a_i^{upper}$ を評価する．この条件が成り立つ時，式 (5.2)，(5.3) に基づき，$cover\_rect(c_{m,n})$ に含まれるセンサデータを用いて，セル $c_{m,n}$ の値を計算する．その値が，センサデータ dt_l に対する条件式 dc_l を満たす場合に，最終的な出力結果として表示する．

IDW に基づくメッシュ統合では，セル c に含まれるセンサデータオブジェクトだけでなく，$cover\_rect(c)$ に含まれるすべてのセンサデータオブジェクトの収集を完了するまでは，集約処理を完了させることができない (図 6.2)．

6.2.5 オーバーレイ処理に基づく異種センサデータ統合

5.1.2 節で述べたように，オーバーレイ処理を実現するためには，センサデータの種類ごとに空間補間に基づくメッシュ統合を適用する必要がある．したがって，オーバーレイ処理に基づくインクリメンタルなメッシュ統合処理は，基本的には，6.2.4 節で述べた手順と同様に行われる．

オーバーレイ処理に基づくメッシュ統合では，DT に指定したすべてのセンサデータ dt_l に対するメッシュデータ ML^l と，統合結果を格納するメッシュデータ OL が必要となる．そこで，まず，ML^l および OL を生成する．

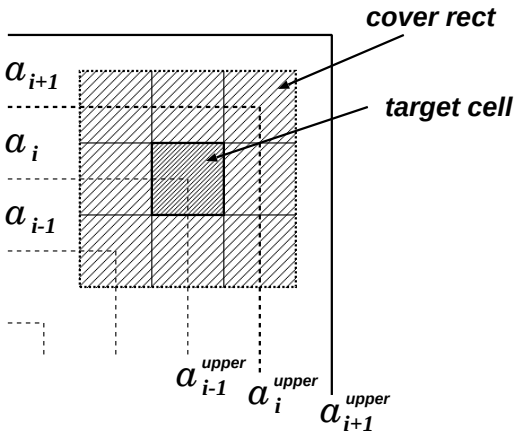

図 6.2: セルの集約計算のための領域判定

次に，領域 a_i のデータを受信した時，ML^l に対しては，6.2.4 節の [4.1′] と同じ手順で，中間結果の保持を行い，[4.2′] と同じ手順で，集約処理の完了判定を行う．この処理を，すべての ML^l について適用する．ただし，OL の計算については，ML^l に対する処理とは異なり，a_i のデータを受信時に，[4.1′] の中間結果の保持は行わず，[4.2′] で，ML^l とは異なる条件で，統合結果の計算を行う．6.2.4 節の [4.2′] に対する OL の処理として，OL の構成要素であるセル $c_{m,n}(1 \leq m \leq i)$ に対して，次の処理が行われる．

4.2 ″ 集約処理の完了

$LAST\_OVERLAP_m$ の要素である $c_{m,n}$ に対応する ML^l $(l = 1, 2, ...)$ の要素 $c^l_{m,n}$ について，集約処理が完了しているかどうか調べる．すべてのセンサデータ dt_l に対して，$c^l_{m,n}$ の集約処理が完了している場合に，OL の要素 $c_{m,n}$ の統合処理を行い，その結果を表示する．

6.3　実験および考察

本節では，Java 言語により実装したセンサデータ視覚化システムを用いた実験の結果についてまとめる．

本章の実験でも，センサデータとして「アメダス観測年報 (2000 年)」を利用する．空間データとして国土数値情報 [国土交通省 a] の「行政界・海岸線データ (平成 11 年度)」から抽出したポリゴンデータを利用する．

本章の実験では，センサデータとしてアメダスの気温データおよび降水量データを利用する．センサデータは，種類ごとに，センサデータサーバによって時系列データとして管理されている．

なお，以降の実験でクエリ Q のパラメータとして指定する問合せ領域 A および空間制約 SC の単位は，度である．

6.3.1 センサデータとポリゴンデータのインクリメンタルな統合

本節の実験では，空間データとして国土数値情報より抽出した都道府県レベルのポリゴンデータを利用し，各ポリゴンに含まれる気温データの平均値を統合結果として表示する．また，本節の実験で想定するクエリの問合せ領域を図 6.3 に示す．

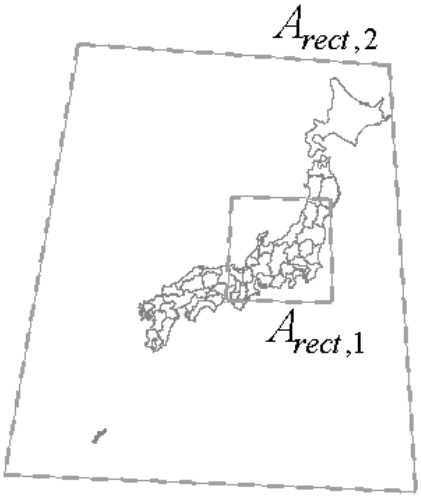

図 6.3: 空間データ統合の実験における問合せ領域

$A_{rect,1} = \langle 134.92, 33.88, 141.41, 39.26\rangle$，$A_{rect,2} = \langle 123, 24, 146, 46\rangle$ であり，$A_{rect,1}$ は，$A_{rect,2}$ の部分領域となっている．まず，表示効果について調べるために，$A_{rect,1}$ に対する実験を行った上で，応答時間が大きい場合の有用性について議論するために，$A_{rect,2}$ に対する実験を行った．

統合結果のインクリメンタルな表示

問合せ領域 $A_{rect,1}$，空間制約 $SC = \{(x_{base}, y_{base}) = (134.92, 33.88), dx_+ = dy_+ = 0.5\}$ をクエリに指定した時の，空間データ統合に基づく気温分布を，問合せ領域の分割の様子とともに，図 6.4，図 6.5 に示す．

図 6.4 は途中結果を，図 6.5 は最終的な結果を示す．図 6.4，図 6.5 では，問合せ領域 ($A_{rect,1}$) が破線で囲まれた矩形として表現され，集約処理の完了した都道府県ポリゴンが色付けられた領域として表示されている．図 6.4→ 図 6.5 より，指定した空間制約に基づいて問合せ領域が分割され，基準点 (問合せ領域の左下端点) に

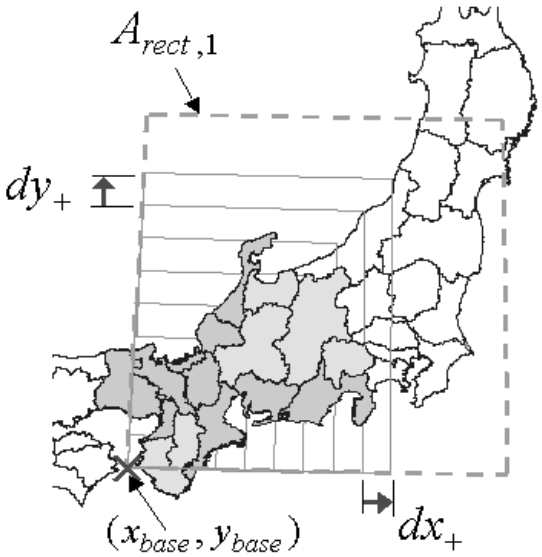

図 6.4: 空間データ統合に基づく気温分布の途中結果

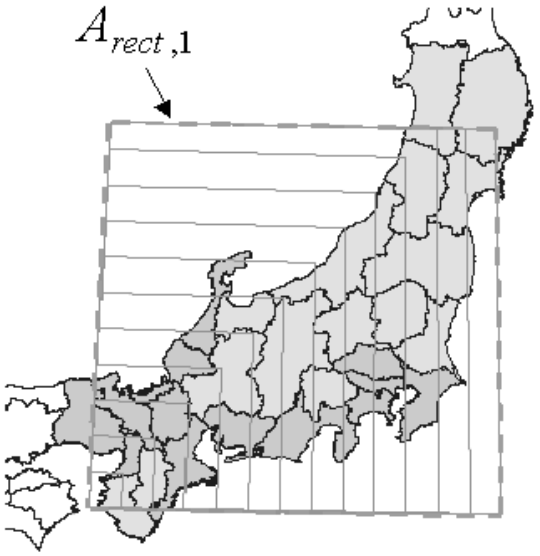

図 6.5: 空間データ統合に基づく気温分布

近い領域から右上方向 (つまり，北東方向) に向かって，インクリメンタルに統合結果が表示されていることがわかる．また，図 6.4，図 6.5 では，集約処理の完了したポリゴンデータの表示だけでなく，データ収集の完了した範囲も矩形領域として表示しており，データ収集の完了した領域とデータ統合の完了した領域の関係を視覚的に理解できる．図 6.4 では，データ収集の完了している領域と重なっていながら，集約処理の結果は表示されていないポリゴンが存在するが，これは，それらのポリゴンの集約処理を完了させるために必要なセンサデータがすべて揃っていないためである．ここで，ポリゴンの集約処理を完了させるために必要なセンサデータとは，クエリ領域に含まれ，なおかつ，そのポリゴンに含まれるものを指す．なお，本節の実験では，問合せ領域と重なり合うすべてのポリゴンデータを統合対象としているが，図 6.5 の問合せ領域 ($A_{rect,1}$) の輪郭線と交差するポリゴンについては，$A_{rect,1}$ に含まれるセンサデータのみを用いて集約処理を行い，最終的な結果を表示している．

総応答時間の変化

ここで，クライアントが，クエリを送信してから，受信したすべてのセンサデータに対する集約処理が完了するまでの時間を，総応答時間と呼ぶ．図 6.6 は，図 6.3 の問合せ領域 $A_{rect,2}$ に対して，空間制約 $SC = \{(x_{base}, y_{base}) = (123, 24), dx_+ = dy_+ = 2.0\}$ を指定し，行政区ポリゴンデータとの統合処理を行った時の総応答時間を示す．分割幅 d は，dx_+, dy_+ の値を示し，d = 100 は，分割をしない場合に相当する．len は，時系列長を示し，単位は，時間 (hour) である．

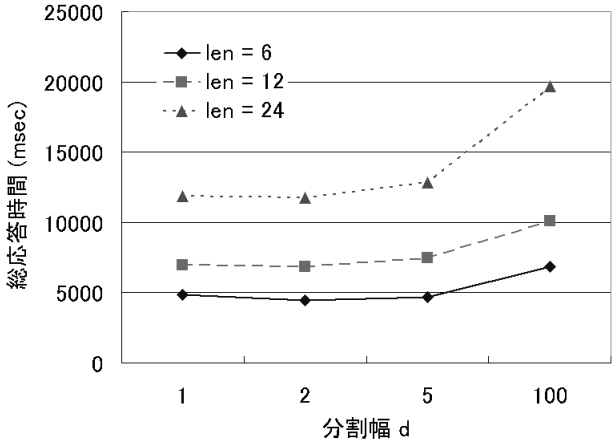

図 6.6: 空間データ統合時の総応答時間

なお，処理時間の測定は，UNIX 環境 (Sun WorkStation, Solaris OS) で行った．各ホストの性能を表 6.1 に示す．

表 6.1: 測定環境

	CPU	メモリ	
host 1	440 MHz	1024 MB	サーバ
host 2	333 MHz	384 MB	仲介エージェント
host 3	900 MHz (×2)	2 GB	クライアント

図 6.6 より，分割しない場合 (d = 100) に比べて，領域分割を行った場合 (d = $\{1, 2, 5\}$) の方が，総応答時間が小さくなっており，時系列長が大きいほど，その傾向が強いことがわかる．

領域分割によって総応答時間が短縮される理由について調べるために，まず，クライアント側のデータ収集率の変化を測定した．図 6.7 は，$A_{rect,2}$ の領域の 24 時間分の気温データに対して，空間データ統合を行った時の，データ収集率の変化を示している．縦軸は，時刻 t までに収集されたセンサデータオブジェクトの数が，最終的に収集されるセンサデータオブジェクトの数に占める割合を示し，横軸は応答時間を示す．

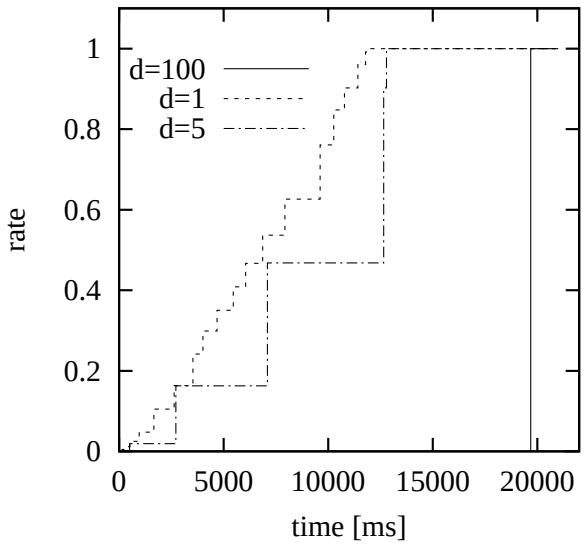

図 6.7: 空間データ統合時のデータ収集率の変化

さらに，この時のサーバでのデータ探索率を図 6.8 に示す．データ探索率は，時刻 t までの探索結果に含まれるセンサデータオブジェクトの数が，全探索結果に含まれるセンサデータオブジェクトの数に占める割合と定義できる．

3.4.3 節では，空間データを管理するサーバに対してクライアントがクエリを送信した時の，検索結果品質の変化 (図 3.17) と探索結果品質 (図 3.18) の変化につい

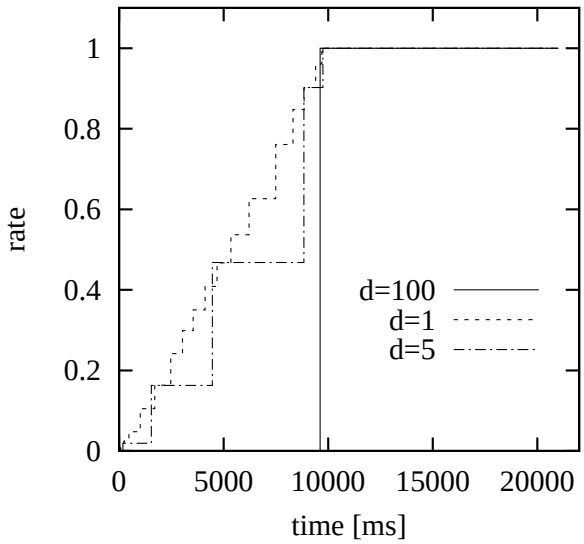

図 6.8: 空間データ統合時のデータ探索率の変化

て考察している．空間データ統合時のデータ収集率 (図 6.7) およびデータ探索率 (図 6.8) についても同様の傾向が見られる．ただし，クライアントからサーバへのクエリの送信と，サーバからクライアントへの検索結果の転送が，仲介エージェントを経由して行われている点は異なる．

領域分割を行った場合には，図 6.8 のように，サーバでのデータ探索率が向上するだけでなく，それに対応して，図 6.7 のように，クライアントでのデータ収集率も向上していくことになる．図 6.7，図 6.8 より，分割しない場合 ($d = 100$) は，全探索終了後に，その探索結果がクライアントに転送され，最終的な結果が出力される時刻まで何の結果も出力されない．一方で，領域分割を行った場合 ($d = 1, 5$) は，探索が終了した領域から順番に，細かな探索結果が転送されるので，総応答時間だけでなく最初の応答時間も短くなり，領域ごとに提供されるセンサデータに対して，逐次的に空間データ統合が行われる．この時の空間データ統合の結果は，図 6.4→ 図 6.5 のようにインクリメンタルに表示され，提案手法が，集約結果の逐次提供を実現できるだけでなく，総応答時間に比べて十分短い時間で意味のある情報を提供できていることがわかる．

次に，図 6.7，図 6.8 の実験時の種々の処理時間について測定した．ネットワークを介した空間集約処理においては，サーバ側でのデータ探索に加えて，クライアント側のデータ統合処理が重要となる．本節の実験では，探索にかかる時間 (search)，統合処理にかかる時間 (integration) に加えて，提案システムがセンサデータを XML 形式で転送しているため，サーバ側の XML タグ付け処理にかかる時間 (tag) とクライアント側の XML 解析処理にかかる時間 (parse) についても測定を行った．各

処理時間と総応答時間 (response) を，図 6.9 に示す．各処理時間は，各分割領域に対する処理時間の総和として示されている．

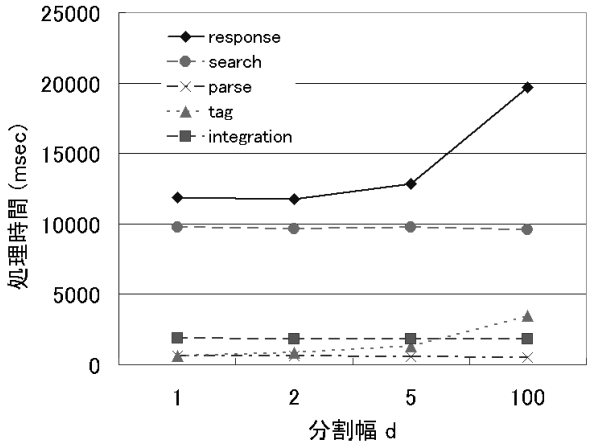

図 6.9: 空間データ統合時の処理時間 (len = 24)

領域分割を行った場合 (d = 1, 2, 5) には，各分割領域に対する探索結果をクライアントが受信する度に統合処理を行っていくため，総応答時間は，最後に受信した検索結果の統合処理時間に依存することになる．それに対して，分割を行わない場合 (d = 100) には，全探索の結果を受信後，図 6.9 に示す時間で，統合処理が行われる．全探索の結果に対する統合処理を，全探索終了後に行うことになるため，サーバの探索時間だけでなく統合処理時間が大きい場合には，総応答時間が大きくなることになる．時系列長が大きい場合には，全探索結果のタグ付けにかかる時間も大きくなるため，総応答時間を上昇させる理由の 1 つとなる．

本章では，領域分割に基づくセンサデータ統合手法を提案しているとも言えるが，ネットワークを介した空間集約処理を扱っているという点だけでなく，ユーザへの情報提供順序を考慮して分割領域を生成し，その上で，$LAST\_OVERLAP$ を導入している点が，既存の spatial join と大きく異なる点であると考える．

6.3.2　空間補間に基づくインクリメンタルなメッシュ統合

問合せ領域 $A_{rect,3} = \langle 136.74, 34.46, 141.11, 38.03 \rangle$ に対する降水量分布を調べる．ここでは，24 時間分の時系列データを要求し，6.2.4 節の手順に従って，その積算値をメッシュ分布として表示する．分割の基準点を $A_{rect,3}$ の左下（南西）端点として，分割幅を $dx_+ = dy_+ = 0.3$ とした時の (積算) 降水量分布を図 6.10，図 6.11 に示す．メッシュの各セルの大きさは 0.1 度四方であり，$max\_cell = 2$ としている．

図 6.10 の左右の両図と図 6.11 左図は途中結果を，図 6.11 右図は最終的な結果を

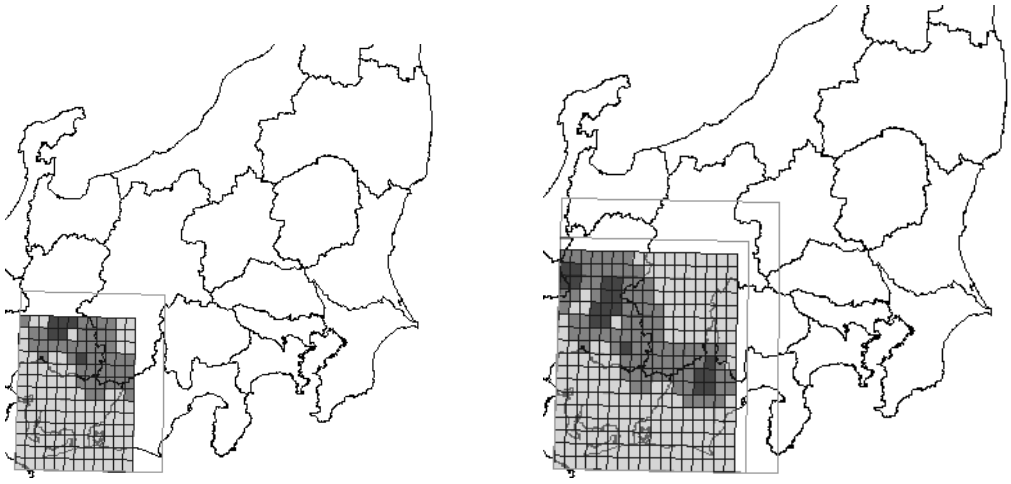

図 6.10: IDW に基づくメッシュ統合による降水量分布の途中結果 (左図から右図へメッシュ統合完了領域が拡大)

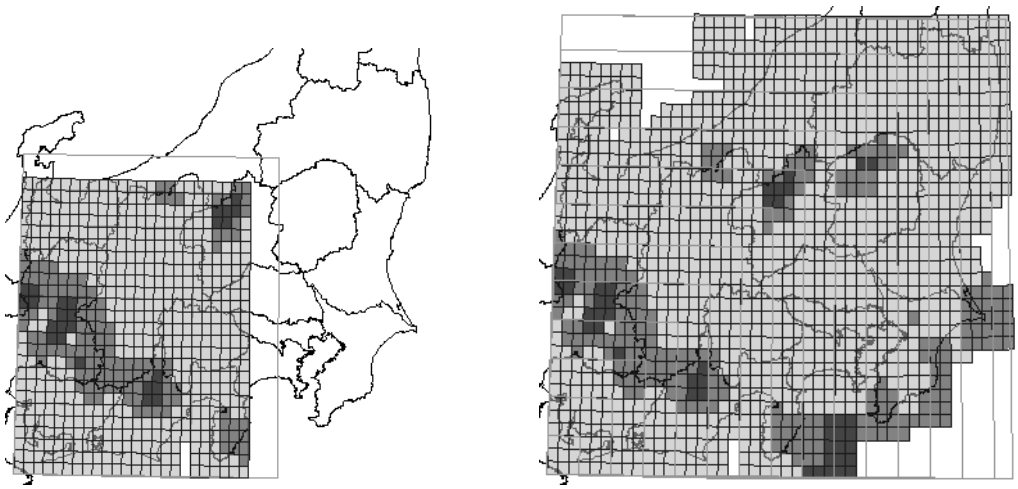

図 6.11: メッシュ統合による降水量分布の表示 (左図：途中結果，右図：最終結果)

示し，これらの図では，統合処理の完了したセルが色付けられて表示されている．前節の空間データ統合の場合と同様に，図 6.10 左図 → 図 6.10 右図 → 図 6.11 左図 → 図 6.11 右図に示すように，基準点に近い場所からデータ統合の完了した領域が時間とともに拡大し，インクリメンタルなメッシュ統合が実現できていることがわかる．メッシュ統合についても，前節の図 6.7 と同様にデータ収集率は変化し，領域分割を行った場合は，すべてのセンサデータに対する集約処理を完了する前に，“領域” という意味のある単位で何らかの統合結果を得ることができる．

図 6.12 に，$A_{rect,3}$ の領域の 24 時間分の降水量データに対して，メッシュ統合を行った時の総応答時間を示す．図中の size は，メッシュを構成するセルの一辺の長さ (単位は度) である．

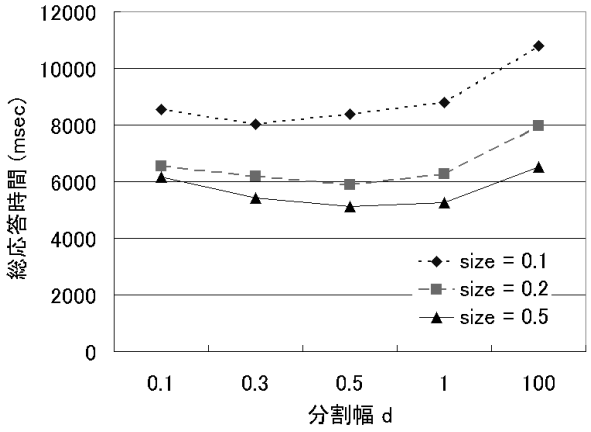

図 6.12: メッシュ統合時の総応答時間

図 6.12 より，セルの大きさ (size) が小さいほど，総応答時間が大きいことがわかる．同じ分割パラメータであれば，各分割領域に対する探索時間は同じであるはずなので，統合処理にかかる時間が影響していると考えられる．図 6.13 に，図 6.12 に対応するクライアントにおける統合処理時間を示す．さらに，表 6.2 に，$A_{rect,3}$ に対して生成されたメッシュを構成するセルの個数を示す．

表 6.2: メッシュを構成するセルの個数

セルの大きさ (size)	セルの個数
0.5	72
0.2	396
0.1	1584

図 6.13，表 6.2 より，セルの大きさ (size) が小さいほど，同一のクエリ領域に対して生成されるセルの個数が多く，結果として，統合対象となるオブジェクト数

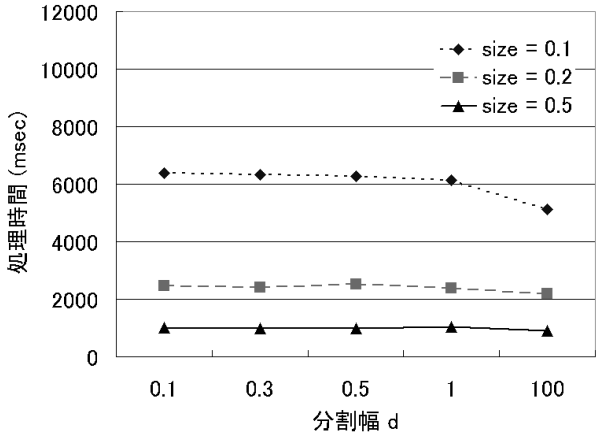

図 6.13: メッシュ統合時の統合処理にかかる時間

が増加するため，統合処理にかかる時間が大きくなっていると考えられる．なお，6.2.4 節で述べた手順の中の，4.1′ の「中間結果の計算」と，4.2′ の「集約処理の完了」に対応する処理時間を調べると，図 6.13 のほとんどが,「集約処理の完了」にあてられており，このステップは，空間補間 (IDW) に基づく計算を行っている部分に相当する．

図 6.12 では，図 6.6 と同様に，分割を行うことによって，総応答時間が短くなる傾向が見られる．セルの大きさが小さい場合，すなわち，統合対象であるオブジェクト数が多く，統合処理コストが大きい場合にも，領域分割を行うことによって，総応答時間を短縮できることがわかる．ただし，size = 0.5 の場合など，分割幅 d が小さくなるにつれて，総応答時間が増加していく傾向も見られる．これは，セルの大きさに比べて，分割幅が小さ過ぎ，複数の領域にまたがるセルが増加するためと考えられる．したがって，メッシュ統合を行う場合には，メッシュの粒度を考慮して，分割幅を指定することが必要である．

6.3.3 オーバーレイ処理に基づくインクリメンタルなセンサデータ統合

本節では，5.1.3 節と同様の統合例を取り上げる．問合せ領域 $A = \langle 136.74, 34.46, 141.11, 38.03 \rangle$ に対して，前項のクエリと同様の空間制約を指定した時のセンサデータ統合の途中結果を 図 6.14 に示す．この時の最終的な統合結果は，図 5.8 である．

図 6.14 左図 → 図 6.14 右図 → 図 5.8 より，オーバーレイ処理に基づくセンサデータ統合の結果をインクリメンタルに表示できていることがわかる．単種類のセンサデータではなく，複数種類のセンサデータに対する統合結果がインクリメンタルに表示され，各データに指定した条件をともに満たしている地域が結果として出

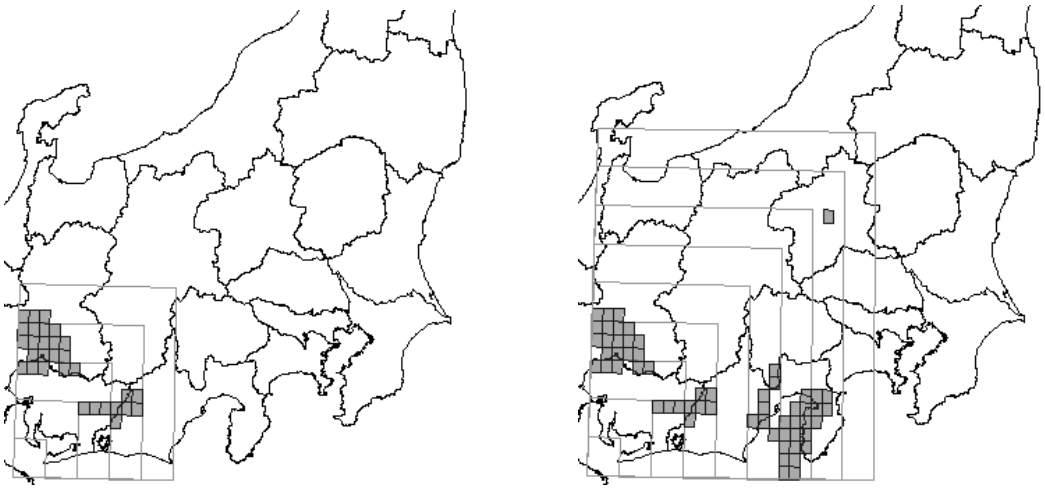

図 6.14: オーバーレイ処理に基づくセンサデータ統合の途中結果

力されている点が重要である．

さらに，オーバーレイ処理によって，センサデータのメッシュ統合結果と，メッシュ形式の空間データをインクリメンタルに統合することが可能である．国土数値情報 [国土交通省 a] には様々なメッシュ形式の空間データが含まれているが，本節では「気候値メッシュ(昭和 62 年度)」に含まれている標高データを統合対象として取り上げる．気候値メッシュは 3 次メッシュ[1]として表現されているが，ここでは，2 次メッシュの標高データに加工して利用する．3 次メッシュの各セルの標高データを用いて，対応する 2 次メッシュのセルの平均標高を算出した．図 6.15 の左図は 3 次メッシュの標高データを示す．図 6.15 の右図は，$height \geq 500$ という条件を満たす 2 次メッシュの標高データを示し，標高が 500 m 以上の地域を表している．色の濃いセルほど，標高が高いことを示す．

図 6.14 と同様の問合せ領域に対して，48 時間分の降水量データを要求し，標高データと統合した時の結果を図 6.16 左図に示す．図 6.16 右図は $SUM(prec) \geq 10$ という条件を満たす領域を表している．図 6.16 左図は「(平均) 標高が 500 m 以上で，2 日間の (積算) 降水量が 10 mm 以上の地域はどこか？」というクエリに対する結果を表している．

この実験では，分割の基準点を問合せ領域の左下端点，分割幅を $dx_+ = dy_+ = 0.5$ としており，基準点に近い領域から，センサデータと標高データとの統合結果がインクリメンタルに表示され，最終的に，図 6.16 左図のメッシュ分布が表示される．図 6.15 右図および図 6.16 右図より，両方の条件を満たす領域が，統合結果として，図 6.16 左図に統合結果として表示されていることがわかる．

[1]第 1 次地域区画 (1 次メッシュ) を縦横に 8 等分して作成したものを第 2 次地域区画 (2 次メッシュ) と呼ぶ．さらに，第 2 次地域区画の縦横それぞれを 10 等分したものを第 3 次地域区画 (3 次メッシュ) と呼ぶ．

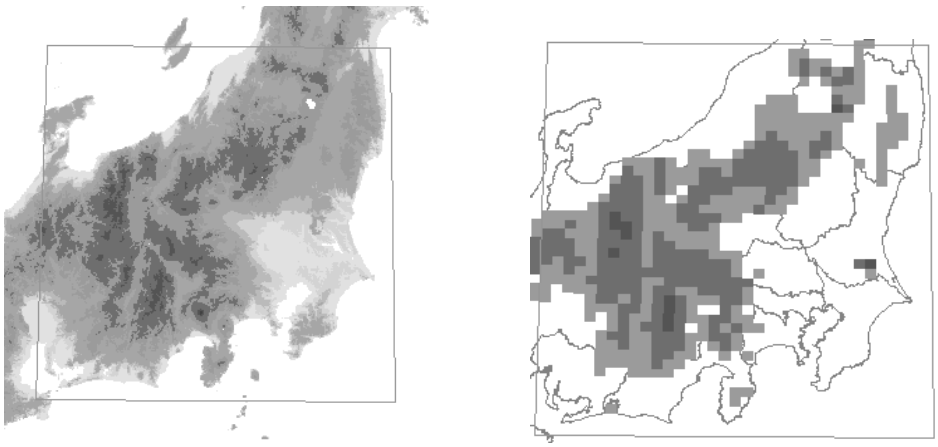

図 6.15: 標高データのメッシュ表示 (左図：3 次メッシュ，右図：2 次メッシュ)

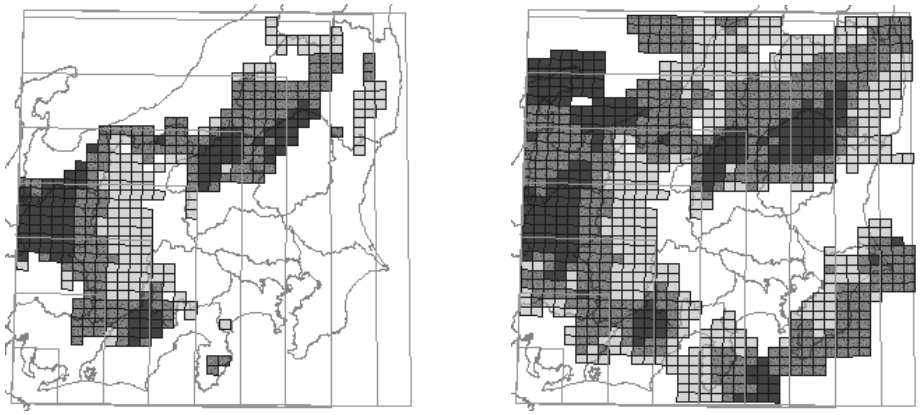

図 6.16: オーバーレイ処理に基づくメッシュデータの統合 (左図：オーバーレイの結果，右図：降水量分布)

4 章では，異種のセンサデータを領域ごとに同期させて提供する枠組みを提案しているが，実際に，複数の種類の異なるセンサデータを関連付ける枠組みを提示しているわけではない．本節の議論より，提案手法が，空間補間に基づいてメッシュ統合を行い，その結果を利用してオーバーレイ処理を行うことで，複数のサーバから提供される異種のセンサデータをインクリメンタルに統合する枠組みを提供できることがわかる．したがって，異種のセンサデータ統合の結果を，ユーザに対して効果的に視覚化する上で，空間集約と領域分割に基づく本手法が有用であると考える．

6.4　まとめ

本章では，センサデータの視覚化のためのインクリメンタルな空間集約手法を提案した．提案手法によって，空間データ統合や空間補間により単種類のセンサデータをインクリメンタルに視覚化するだけでなく，オーバーレイ処理により種類の異なるセンサデータを関連付け，その結果をインクリメンタルに表示できる．また，本章では，3 章や 4 章の実験で扱った例と比較すると，中間結果の計算や集約の完了処理を含む統合処理にかかる時間が大きい場合を扱っている．このような統合処理時間の大きい場合において，領域分割に基づくアプローチが，集約処理の途中に “領域” という意味のあるまとまりで部分的な結果を閲覧できるという点だけでなく，応答時間の短縮という面でも効果的であることが実験により示唆された．

さらに，本章では，ネットワーク上に分散したセンサデータベースから取得した時系列データを統合する問題を取り上げているが，“空間情報” としてのセンサデータの視覚化手法を提案している．その観点では，多種多様な空間情報をネットワークを介して視覚化する 1 つの枠組みを示しているとも言える．

第7章

考察および今後の課題

7.1 Anytime algorithm に基づくセンサデータの収集

4 章で述べたセンサデータ検索方式は，応答時間の増大に対処しているため，大量の時系列データを取得する場合に有効である．4 章の実験では，区間を指定することで時系列データを取得しているが，サンプリングによる時系列データの取得や，より複雑な時系列データに対する検索においても，サーバでのデータアクセスコストが大きくなることが予想され，提案方式が有効に機能すると考える．さらに，時系列データが長くなる場合だけでなく，問合せ領域が広い，あるいはセンサの分布密度が高いなど，問合せ領域に含まれるセンサノード数が大きい場合にも有用であると考えられる．

また，本研究では，ネットワーク上のデータベースに蓄積されたセンサデータを対象としている．一方で，モバイルコンピューティングの分野では，ネットワークに接続された複数の分散センサノードが存在する環境を対象とした研究も盛んに行われており，今後，これらのセンサノードにネットワークを介して直接アクセスすることで，リアルタイムのセンサデータを利用できるようになる可能性もある．さらに，気象情報などの時間的に変化する情報を，Web サイト単位ではなく，Web ページ単位で提供することも考えられる．このような広域ネットワーク上の分散ノードからセンサデータを収集する場合にも，データ収集コストの増大が考えられるため，今後，提案手法の適用および拡張を検討していきたいと考えている．

7.2 検索結果の品質に関する考察

7.2.1 応答性を評価する指標としての検索結果の品質

本論文では，検索結果の品質をデータオブジェクトの収集率に基づいて計算しているため，サーバから検索結果がインクリメンタルに提供されれば，当然のことながら，品質は単調に増加していく．3 章の式 (3.1) の定義では，空間データサーバ

側が検索結果を領域の順序に従って転送するかどうかに関わらず，検索結果の品質は単調に増加していく．4 章の式 (4.10) の定義では，複数のサーバからの検索結果を制約を満たしながら収集できれば検索結果の品質は向上していくことになるが，制約を満たさなくても品質が低下することはない．

ただし，図 7.1 に示すような品質変化のグラフを分析することによって，検索／視覚化システムのユーザに対する応答性を考察することができる．ここで，サーバが領域分割に基づいて検索結果を提供する場合の品質 $q(t)$ は，図 7.1 に示すように変化すると考えられ，(iii) が分割を行わない場合，(ii) が分割を行った場合，(i) がさらに細かな分割を行った場合に相当する．

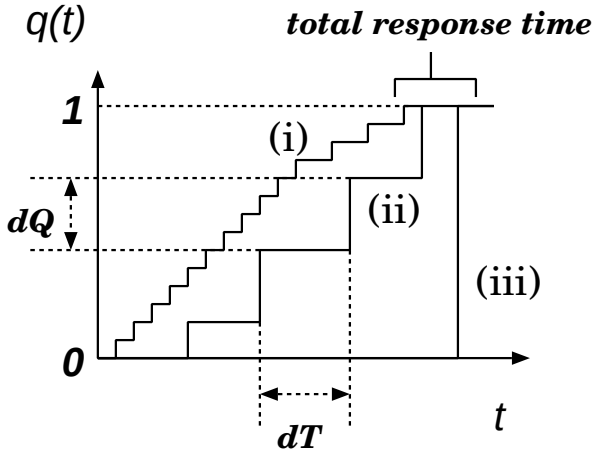

図 7.1: 評価指標としての検索結果品質の変化

3 章，4 章，6 章で述べてきたように，クライアントが検索結果を逐次的に受信する度に，検索結果の品質は時間とともに増加し，クエリを送信してから $q(t)$ が 1 に到達するまでの時間が総応答時間 (total response time) となる．

図 7.1 の (ii) のグラフにおいて，dT は，ある途中結果を受信してから，次の途中結果を受信するまでの時間を示しており，この間の品質は変化しない．図 7.1 の (ii) のグラフより，(i) のグラフの方が dT が短くなるが，dT が長いほど，ユーザに対して次の統合結果を提示するまでの時間が長くなり，dT が短いほど，提示情報を頻繁に更新することになる．したがって，ユーザへの検索結果の提示という観点では，必ずしも，(i) のように，処理品質をなめらかに変化させるよりも，(ii) のように，ある程度の時間間隔を持って，表示情報を更新していく方が効果的な情報提供ができる可能性がある．言い換えれば，領域分割のパラメータ (分割幅) を調整することで，ユーザの要求に応じたタイミング (時間間隔) で，空間情報の検索／統合の結果を制御できる可能性があり，応答性の良いユーザインタフェースシステム構築のための要素技術として提案手法が有用であると考えられる．

また，図 7.1 の (ii) のグラフにおいて，dQ は，品質の変化の差を示すが，この差が大きいほど，ユーザに一度に提示される情報が多く，5 章，6 章のような空間集約などのコストの大きな統合処理を行う場合には，統合結果を表示するまでの時間が長くなる．統合処理の時間は，クライアントの処理性能に依存するが，携帯端末のような資源の制約される環境での空間情報の閲覧を考えた場合には，dQ が小さくなるように分割幅を指定することで，受信結果を効率的に処理できると考えられる．

以上より，本論文で導入したデータ収集率に基づく品質定義は，単純ではあるが，実装した視覚化システムの応答性を評価する指標の 1 つと捉えることができ，本論文において提案手法の有用性を議論する上で，十分な役割を果たしていると考えられる．

7.2.2 新たな品質定義の導入

本論文における検索結果品質の定義では，処理品質が単調に増加するというだけでなく，グラフの勾配もほぼ一定であり，たとえば，図 7.2 の (b) のような変化を示している．

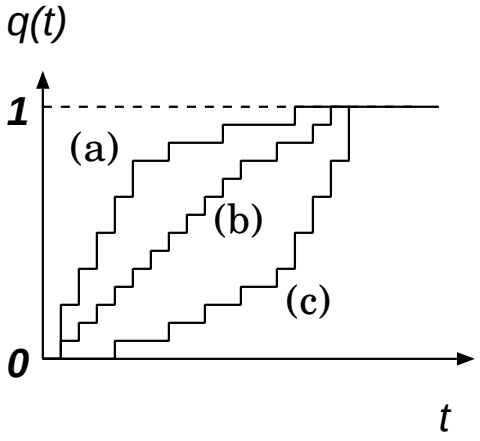

図 7.2: 処理品質の増加傾向の違い

4.3.2 節の図 4.7 で示したように，空間制約や時間制約を満たさないデータオブジェクトが収集される場合 (broker 方式) には，時間とともに制約を満たさないオブジェクトが蓄積されていくため，(c) のグラフのように品質が変化する．(c) のグラフは，データ収集の前半での品質が低く，後半に急激に品質が上昇する．それに対して，(a) のグラフは，データ収集の初期の段階での品質が高く，後半の変化はゆるやかである．Anytime algorithm の観点では，処理途中に割り込まれても何ら

かの結果を返せることが重要であり，ある時点での処理品質が高い (a) の変化を示すことが望ましい．

3 章の実験では，収集したすべての空間データオブジェクトに一様の重みを与えている．4 章の実験では，制約を満たしているセンサデータオブジェクトには一様の重みを与えているが，制約を満たしていないオブジェクトの個数は品質の計算に反映されていない．しかし，空間情報の閲覧を考えた場合には，たとえば，ユーザの指定した地点や領域に近いオブジェクトに対して，距離に応じて重み付けをすることが考えられる．指定した地点に近い分割領域に含まれるデータオブジェクトには高い重みを与え，遠い分割領域ほど低い重みを与えることで，分割領域ごとの重みを反映した検索結果の品質の計算ができる．この場合，空間制約を満たしたデータオブジェクトが収集されれば，処理品質は (a) のグラフのように変化し，その変化から，ユーザの要求に応じたデータ提供ができていることが予想できる．複数のサーバからの検索結果を利用して，オーバーレイなどの位置情報に基づく統合処理を行う場合には，同一領域のデータが収集されているほど望ましい状態であるので，距離に応じた重み付けによって，処理品質が (a) のように変化すると考えられる．

また，空間制約を満たさないデータオブジェクトについても，クライアント側での統合処理のための中間結果の計算や何らかの提示情報の生成に利用できる可能性もある．たとえば，検索範囲全体のおおまかなデータ分布を知りたい場合には，検索の全範囲にわたるデータ収集をしながら，空間制約を満たすデータオブジェクトを優先的に取得するというアプローチも考えられる．この場合には，空間制約を満たしていないデータオブジェクトの個数を品質の計算に利用した方が良いと考えられる．4 章の式 (4.10) では，空間制約を満たしているデータオブジェクト数のみから品質の計算を行っているが，空間制約を満たしている領域に高い重みを与えることで，空間制約を満たしていない領域のデータオブジェクト数も品質の計算に利用できる．

センサデータを対象とした場合には，空間制約や時間制約を満たしているか，ということだけでなく，収集されたセンサデータの鮮度や時間的な一貫性も重要である．最新のセンサデータを利用するアプリケーションであれば，鮮度の良い，すなわち，更新時刻の新しいセンサデータの収集が要求される．また，ある時刻のスナップショットを閲覧したい場合には，複数のサーバから収集したセンサデータの更新時刻が，ほぼ同時刻であることが望ましい．このような場合には，たとえば，ユーザの指定した地点に近い領域については鮮度の高さを優先する，あるいは，各分割領域に含まれるデータについては更新時刻が近くなるように収集を行うことなどが考えられる．今後の課題として，これらの要素を反映した品質の定義を行い，図 7.2 の (a) のグラフのように処理品質を増加させることのできる手法を考案することが考えられる．

7.3 空間補間の方法の選択

2.1.5 節で述べたように，空間補間の方法としては，Inverse Distance Weighted (IDW)，スプライン，クリギング，最近隣法，ボロノイ図など様々な手法が提案されている [Longley *et al.* 01, McCoy *et al.* 01, 村井 02, 伊理 99, 間瀬 他 01, 浅野 他 00]．本論文では，IDW を空間補間の手法として利用し，メッシュ統合を行っている．その理由としては，以下の点が挙げられる．

- IDW は，非常に単純なアルゴリズムであるが，全体的な傾向の分布を把握するためには，十分効果的な手法である．
- 本論文では，領域分割に基づくインクリメンタルな空間集約手法を提案している．IDW は，局所性に基づく計算手法であるため，領域ごとに中間結果を計算し，集約処理の完了を判定する提案アルゴリズム (6.2 節) との適合性が高いと考えられる．
- 5 章では，センサデータの視覚化における空間集約の有用性について議論している．その観点では，空間補間の中で，どの手法が効果的であるか，という以前に，空間補間そのものをセンサデータの視覚化に適用することが有用であることを示すことが重要である．IDW に基づくメッシュ統合は，単種類のセンサデータを集約する場合だけでなく，複数種類のセンサデータに対して IDW を適用し，その結果をオーバーレイする場合にも効果的であることが 5 章の実験結果より示唆されている．

一方で，5.2.7 節で取り上げたような写真アプリケーションでは，ある地点のセンサデータを補間するために，より高度な補間手法が適していると考える．たとえば，クリギングは，地球統計学に基づく手法 [間瀬 他 01] であるが，信頼性の高い補間処理が期待できる．しかしながら，処理コストは IDW に比べて大きく，応答の早さが要求される場合には，適さないと考えられる．したがって，全体的な傾向を把握したい場合や応答性が要求される場合には，IDW を利用し，ある地点のセンサデータを求めたい場合や精度の高い分布が必要な場合にはクリギングを利用するというように，目的に応じて補間方法を切り替えられる枠組みが必要になると考えられる．

また，センサデータの中には，風速／風向といった方向を持つものも含まれる．各軸方向の成分ごとに IDW を適用することによって，提案手法と同様の手順で，方向を持つセンサデータの補間が可能である．降水量などの点データの集約結果であるメッシュ分布と，風速／風向データの補間結果を重ね合わせることで，より効果的な情報提供ができると考えられる．ただし，点データの補間と同様に，アプリケーションの目的に合った補間法であるかどうかは検討する必要がある．

7.4　センサ情報統合サービスの実現

本論文では，空間データ検索手法（3 章），センサデータベース検索手法（4 章）およびセンサデータ統合手法（5 章，6 章）を提案している．これらの要素技術によって，ネットワーク上に分散したデータリソースからの異種のセンサデータと，ネットワークを介して提供される位置情報統合のための空間データをクライアントが統合することが可能となる．そのクライアントとして，計算資源の豊富なデスクトップコンピュータ上の環境モニタリングシステムや地理情報システムなどのアプリケーションが考えられる．

それに対して，計算資源の限られた携帯端末上のアプリケーションを想定した場合には，そのアプリケーション上で統合処理を行うだけでなく，それらのアプリケーションに統合処理の結果を提供するデータ統合システムが必要になると考えられる．今後は，このような情報統合の結果を提供する位置情報サービスや Web サービスとして，本研究で構築したセンサデータの閲覧／視覚化システムを，発展させていきたいと考えている．この場合でも，データ統合システムが各リソースからセンサデータを収集し，統合のための空間データを取得し，統合処理を行い，その結果をインクリメンタルにクライアントに提供することで，応答性の良い情報提供サービスが実現できると考える．

第8章

結論

本論文では，ネットワークを介して空間情報をインクリメンタルに視覚化するための検索手法を提案した．まず，代表的な空間情報である空間データをネットワークを介して取得する問題に対して，検索結果をインクリメンタルに提供するための領域分割に基づく検索手法を提案し，本論文における空間情報の視覚化のための基本的な考え方を示した．次に，ネットワーク上に分散したデータベースに蓄積されたセンサデータを「空間情報」と考え，それらのセンサデータベースから収集したセンサデータを視覚化する際の検索および統合という問題に対して，領域分割に基づくアプローチを適用した．具体的に，領域分割と時区間分割に基づくセンサデータ検索手法，および検索結果をインクリメンタルに統合する手法として領域分割に基づくインクリメンタルな空間集約手法を提案した．これらの手法によって，センサデータベース検索において，効果的な情報提供が実現できることを実験的に示した．このセンサデータの検索と統合の枠組みは，単に，インタラクティブにセンサデータを閲覧するための要素技術というだけではなく，ネットワーク上に分散したデータベースから多種多様な空間情報を収集し，統合し，視覚化するための要素技術と考えることもできる．

空間データ検索のためのインクリメンタルなデータ提供

空間データ検索のための “anytime algorithm” に基づく手法を提案した．Anytime algorithm は処理品質を単調に増加させる性質を持つが，本論文では，データ収集率を検索結果の品質と定義し，検索結果をインクリメンタルに提供することで品質を時間とともに増加させる．本手法は，問合せ領域を分割し，空間インデックスを利用して，分割領域ごとにデータ探索を行い，その結果をインクリメンタルにクライアントに転送する．提案手法を空間データサーバの機能として実装し，地図描画アプリケーションを用いた実験を行った結果，領域分割のためのパラメータを指定することにより，ユーザの関心のある領域の情報を優先的に取得し，効果的に空間データを表示できることを確認した．領域分割によるインクリメンタルなデータ提

供は，地図表示だけでなく，より複雑で高度な空間情報処理に対しても有用であると考える．

センサデータベースの検索のためのインクリメンタルなデータ提供

領域分割に基づく検索手法を，センサデータベース検索に適用し，インクリメンタルなセンサデータ提供方式を提案した．本方式は，仲介エージェントとセンサデータベースのラッパーとなるセンサデータサーバから構成され，これらのエージェントの連携により，クエリの実行と検索結果の転送を制御する．検索結果はユーザの指定する領域分割および時区間分割のためのパラメータに基づいて分割され，インクリメンタルに提供される．本方式により，複数のサーバからのセンサデータを，領域ごとに同期させて収集できるだけでなく，時間順序を考慮して効果的に提示することが可能となる．

インクリメンタルなセンサデータの統合

空間集約に基づくセンサデータ視覚化システムを構築し，ユーザの移動履歴を利用したセンサデータ閲覧の方法を示した．これにより，ユーザにとって理解しやすい提示情報の生成と，センサデータ提供システムに対する直接的かつ直感的なデータ要求が可能となる．その上で，クライアント側で，センサデータベース検索の結果であるセンサデータを統合する手法として，インクリメンタルな空間集約手法を提案した．具体的に，空間データ統合，空間補間に基づくメッシュ統合，オーバーレイ処理に基づく異種センサデータ統合に対して，受信したセンサデータをインクリメンタルに処理する方法を示した．これにより，クライアントは，逐次的に提供されるセンサデータを利用して統合処理を行い，統合の途中結果をユーザに提示することができる．実験の結果，領域分割に基づく検索手法と統合手法によって，“領域”という意味のあるまとまりでセンサデータ統合の結果を提示できることが示された．また，提案手法がインタラクティブなセンサデータ閲覧システムを実現する上で有用であることが示唆された．

本論文で提案した anytime algorithm に基づくセンサデータベース検索手法は，現在ユビキタスコンピューティングやデータベースの分野で注目され，今後ますます重要になると考えられるセンサデータの検索と統合という問題に対して有用であると考える．また，本論文で示した空間情報の視覚化のための検索と統合の枠組みによって，多種多様な空間情報のリアルタイムかつインタラクティブな空間情報の閲覧が実現できると考える．

謝辞

本研究の機会を与えて下さり，また研究を進めるにあたり有益な御助言と御指導を頂いた慶應義塾大学理工学部教授 安西祐一郎 先生に心より感謝致します．“anytime algorithm” への着想という安西先生のアドバイスがなければ，また，空間情報に関する研究に対する先生の御理解がなければ，本論文の完成には至らなかったことと思います．安西先生には，本当に，長い間，温かく見守って頂き，様々な経験を通して，研究することの楽しさと厳しさを教えて頂きました．

慶應義塾大学理工学部教授 大野義夫 先生には，本論文の審査を御快諾頂き，論文に関して多くの貴重な御意見を頂きました．慶應義塾大学理工学部専任講師 高田眞吾 先生には，論文の細部にわたるコメントを頂き，さらに博士論文の手続きを進める上で，的確なアドバイスを頂くなど，大変お世話になりました．また，應義塾大学理工学部教授 櫻井彰人 先生には，お忙しい中，本論文の審査をお引き受け頂き，大変有益なコメントを頂きました．ここに謹んで感謝致します．

慶應義塾大学理工学部教授 天野英晴 先生，慶應義塾大学理工学部専任講師 山崎信行 先生には，博士課程在籍中および博士論文審査の手続きを進めるに際して，数々の御助言と励ましの言葉を頂きました．

慶應義塾大学理工学部専任講師 今井倫太 先生には，研究内容のみならず，研究の進め方について，機会あるごとにアドバイスを頂き，参考にさせて頂きました．大変感謝しております．

慶應義塾大学理工学部教授 寺岡文男 先生，慶應義塾大学理工学部助教授 山本喜一 先生，慶應義塾大学理工学部専任講師 遠山元道 先生には，博士課程に在籍している頃から，研究に関する議論をして頂き，数々の有益なコメントを頂きました．また，慶應義塾大学理工学部教授 川嶋弘尚 先生には，本研究を進める上で貴重な御意見を頂きました．心より感謝致します．

東京大学空間情報科学研究センター助教授 有川正俊 先生には，ゼミに参加させて頂き，研究に関する議論を通して，空間情報システムという観点から様々な有益な御助言と示唆に富む御指摘を頂きました．大変感謝しております．また，相良毅氏（現在，東京大学 生産技術研究所 戦略情報融合国際研究センター助手），田中浩也 氏（現在，京都大学 情報学研究科 COE 研究員），東京大学大学院生 藤田秀之氏，藤森史生氏，野秋浩三氏からは，ゼミでの議論を通して，数々の示唆に富むコメントを頂くことができました．ここに，感謝致します．

そして，川島英之氏，梅澤猛氏，大村廉氏をはじめとする安西・山崎・今井研究室の現役学生の皆様，嶋田総太郎氏，桜井利彦氏，宮島麻美さん，米山華子さん，鈴木英之氏，白井俊紀氏ほか，安西研究室の卒業生である後輩の皆様からは多くの御助力と御意見を頂きました．また，中内靖氏，矢向高弘氏，大隅智春氏をはじめ安西研究室の諸先輩方，研究室秘書の永坂弘子さん，元研究室秘書の続橋諒さんには，何度も励ましの言葉を頂きました．ここに，心より感謝致します．

また，博士課程をともに過ごした同期の仲間である河野通宗氏，山本純氏，同期の友人である根岸征史氏，平松薫氏には，機会あるごとに激励の言葉を頂きました．ここに感謝致します．

最後に，永きにわたり勉強と研究の機会を与えてくれるとともに，健康面，精神面において支えてくれた両親に心から感謝致します．

平成 16 年 2 月
白石 陽

参考文献

[Aurora] The Aurora Project. http://www.cs.brown.edu/research/aurora.

[Bailey *et al.* 95] Trevor C. Bailey, and Anthony C. Gatrell. *Interactive Spatial Data Analysis*. Prentice Hall, 1995.

[Beckmann *et al.* 90] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. In *Proceedings of ACM SIGMOD Conference*, pp. 322–331, 1990.

[Bertolotto *et al.* 99] Michela Bertolotto, and Max J. Egehofer. Progressive Vector Transmission. In *Proceedings of 7th ACM Symposium on Advances in Geographic Information Systems*, 1999.

[Bertolotto *et al.* 01] Michela Bertolotto, and Max J. Egenhofer. Progressive Transmission of Vector Map Data over the World Wide Web. *GeoInformatica*, Vol. 5, No. 4, pp. 345–373, 2001.

[Bettati *et al.* 92] Riccardo Bettati, Nicholas S. Bowen, and Jen-Yao Chung. Checkpointing Imprecise Computation. In *Proceedings of the IEEE Workshop on Imprecise and Approximate Computation*, pp. 45–49, 1992.

[Boddy *et al.* 89] Mark Boddy, and Thomas Dean. Solving Time-Dependent Planning Problems. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pp. 979–984, 1989.

[Boddy *et al.* 94] Mark Boddy, and Thomas Dean. Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence*, Vol. 67, pp. 245–286, 1994.

[Bonnet *et al.* 01] Philippe Bonnet, Johannes Gehrke, and Praveen Seshadri. Towards Sensor Database Systems. In *Proceedings of 2nd International Conference on Mobile Data Management*, pp. 3–14, 2001.

[Bradshaw 97] Jeffrey M. Bradshaw. *Software Agents*. The AAAI PRESS/The MIT PRESS, 1997.

[Brinkoff *et al.* 93] Thomas Brinkoff, Hans-Peter Kriegel, and Ralf Schneider. Comparison of Approximations of Complex Objects Used for Approximation-based Query Processing in Spatial Database Systems. In *Proceedings of 9th International Conference on Data Engineering*, pp. 40–49, 1993.

[Chen *et al.* 97] Xiao Chen, and Albert Mo Kim Cheng. An Imprecise Algorithm for Real-Time Compressed Image and Video Transmission. In *Proceedings of 6th Conference on Computer Communications and Networks*, pp. 390–397. IEEE, 1997.

[CONTROL] CONTROL Projects. http://control.cs.berkeley.edu/projects.html.

[Dean *et al.* 88] Thomas Dean, and Mark Boddy. An Analysis of Time-Dependent Planning. In *Proceedings of the 6th National Conference on Artificial Intelligence*, pp. 49–54, 1988.

[Decker *et al.* 96] Keith Decker, Mike Williamson, and Katia Sycara. Matchmaking and Brokering. In *Proceedings of 2nd International Conference on Multi-Agent Systems (ICMAS-96)*, 1996.

[Decker *et al.* 97] Keith Decker, Katia Sycara, and Mike Williamson. Middle-Agents for the Internet. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI-97)*, pp. 578–583, 1997.

[Gaede *et al.* 98] Volker Gaede, and Oliver Günther. Multidimensional Access Methods. *ACM Computing Surveys*, Vol. 30, Issue 2, pp. 170–231, 1998.

[Genesereth *et al.* 94] Michael R. Genesereth, and Steven P. Ketchpel. Software agents. *Communications of the ACM*, Vol. 37, No. 7, 1994.

[Guttman 84] Antonin Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *Proceedings of ACM SIGMOD Conference*, pp. 47–57, 1984.

[Hellerstein *et al.* 97] Joseph M. Hellerstein, Peter J. Haas, and Helen J. Wang. Online Aggregation. In *Proceedings of ACM SIGMOD Conference*, pp. 171–182, 1997.

[Hjaltason *et al.* 95] G. R. Hjaltason, and H. Samet. Ranking in Spatial Databases. In *Proceedings of 4th international Symposium on Advances in Spatial Databases, SSD '95, (LNCS-951)*, pp. 83–95, 1995.

[Hjaltason *et al.* 99] G. R. Hjaltason, and H. Samet. Distance Browsing in Spatial Databases. *ACM Transactions on Database Systems*, Vol. 24, No. 2, pp. 265–318, 1999.

[Huang *et al.* 95] Xiaofen Huang, and Albert Mo Kim Cheng. Applying Imprecise Algorithms to Real-Time Image and Video Transmission. In *Proceedings of Real-Time Technology and Applications Symposium*, pp. 96–101, 1995.

[Huang *et al.* 97] Yun-Wu Huang, Ning Jing, and Elke A. Rundensteiner. Spatial joins using r-trees: Breadth-first traversal with global optimization. In *Proceedings of 23rd International Conference on Very Large Data Bases*, pp. 396–405, 1997.

[Koudas *et al.* 97] Nick Koudas, and Kenneth C. Sevcik. Size separation spatial join. In *Proceedings of ACM SIGMOD Conference*, pp. 324–335, 1997.

[Levy *et al.* 96] Alon Y. Levy, Anand Rajaraman, and Joann J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. In *Proceedings of 22nd International Conference on Very Large Databases*, pp. 251–262, 1996.

[Lin *et al.* 87] Kwei-Jay Lin, Swami Natarajan, Jane W.-S. Liu, and Tim Krauskopf. Concord: A System of Imprecise Computation. In *Proceedings of the 11th Annual International Computer Software and Applications Conference*, pp. 75–81. IEEE, 1987.

[Lo *et al.* 94] Ming-Ling Lo, and Chinya V. Ravishankar. Spatial joins using seeded trees. In *Proceedings of ACM SIGMOD Conference*, pp. 209–220, 1994.

[Lo *et al.* 96] Ming-Ling Lo, and Chinya V. Ravishankar. Spatial hash-joins. In *Proceedings of ACM SIGMOD Conference*, pp. 247–258, 1996.

[Longley *et al.* 01] Paul A. Longley, Michael F. Goodchild, David J. Maguire, and David W. Rhind. *Geographic Information Systems and Science*. John Wiley & Sons, Ltd, 2001.

[Madden *et al.* 02] Samuel Madden, and Michal J. Franklin. Fjording the Stream: An Architecture for Queries Over Streaming Sensor Data. In *Proceedings of 18 th International Conference on Data Engineering*, pp. 555–566, 2002.

[McCoy *et al.* 01] Jill McCoy, and Kevin Johnston. *Using ArcGIS Spatial Analyst*. ESRI, 2001.

[Moon *et al.* 00] Bongki Moon, Inés F.V. López, and Vijaykumar Immanuel. Scalable algorithms for large temporal aggregation. In *Proceedings of the 16th International Conference on Data Engineering*, pp. 145–154, 2000.

[OGC] Open GIS Consortium. http://www.opengis.org.

[Orenstein 89] Jack A. Orenstein. Redundancy in Spatial Databases. In *Proceedings of ACM SIGMOD Conference*, pp. 294–305, 1989.

[Papadopoulos *et al.* 97] Apostolos Papadopoulos, and Yannis Manolopoulos. Performance of Nearest Neighbor Queries in R-trees. In *Proceedings of 6th International Conference on Database Theory, ICDT '97, (LNCS-1186)*, pp. 394–408, 1997.

[Patel *et al.* 96] Jignesh M. Patel, and David J. DeWitt. Partition based spatial-merge join. In *Proceesings of ACM SIGMOD Conference*, pp. 259–270, 1996.

[Rigaux *et al.* 01] Philippe Rigaux, Michel Scholl, and Agnès Voisard. *Spatial Databases with Application to GIS*. Morgan Kaufmann, 2001.

[Roth *et al.* 97] Mary Tork Roth, and Peter M. Schwarz. Don't Scrap It, Wrap It! A Wrapper Architecture for Legacy Data Sources. In *Proceedings of International Conference on Very Large Databases*, pp. 266–275, 1997.

[Roussopoulos *et al.* 95] Nick Roussopoulos, Stephen Kelly, and Frédéic Vincent. Nearest Neighbor Queries. *SIGMOD Record*, Vol. 24, No. 2, pp. 71–79, 1995.

[Russel *et al.* 95] Stuart Russel, and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, 1995.

[Sellis *et al.* 87] Timos Sellis, Nick Roussopoulos, and Christos Faloutos. The R^+-tree: A dynamic index for multi-dimensional objects. In *Proceedings of 13th International Conference on Very Large Databases*, pp. 507–518, 1987.

[Shanmugasundaram *et al.* 00] Jayavel Shanmugasundaram, Kristin Tufte, David J. DeWitt, Jeffrey F. Naughton, and David Maier. Architecting a Network Query Engine for Producing Partial Results. In *Proceedings of the Third International Workshop on the Web and Databases*, pp. 17–22, 2000.

[Son *et al.* 96] Daeweon Son, and Ramez Elmasri. Efficient temporal join processing using time index. In *Proceedings of the 8th International Conference on Scientific and Statistical Database Management*, pp. 252–261, 1996.

[Soo *et al.* 94] Michael D. Soo, Richard T. Snodgrass, and Christian S. Jensen. Efficient evaluation of the valid-time natural join. In *Proceedings of 10th International Conference on Data Engineering*, 1994.

[TC211] ISO/TC211, Geographic Information/Geomatics. http://www.isotc211.org.

[Vrbsky 97] Susan V. Vrbsky. A data model for approximate query processing of real-time databases. *Data & Knowledge Engineering*, Vol. 21, pp. 79–102, 1997.

[Vrbsky *et al.* 93] Susan V. Vrbsky, and Jane W. S. Liu. APPROXIMATE – A Query Processor That Produces Monotonically Improving Approximate Answers. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 6, pp. 1056–1068, 1993.

[Wiederhold 92] Gio Wiederhold. Mediators in the Architecture of Future Information Systems. *IEEE Computer*, Vol. 25, No. 3, pp. 38–49, 1992.

[Zilberstein *et al.* 93] Shlomo Zilberstein, and Stuart J. Russell. Anytime Sensing, Planning and Action: A Practical Model for Robot Control. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pp. 1402–1407, 1993.

[Zilberstein *et al.* 96] Shlomo Zilberstein, and Stuart Russel. Optimal composition of real-time systems. *Artificial Intelligence*, Vol. 82, No. 1-2, pp. 181–213, 1996.

[Zilberstein *et al.* 99] S. Zilberstein, F. Charpillet, and P. Chassaing. Real-Time Problem-Solving with Contract Algorithms. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI-99)*, pp. 1008–1013, 1999.

[DPC] データベース振興センター. G-XML. http://gisclh.dpc.or.jp/gxml/.

[伊理 99] 伊理正夫. 計算幾何学と地理情報処理 第 2 版. 共立出版, 1999.

[横路 他 97] 横路誠司, 高橋克巳, 三浦信幸, 島健一. 位置指向の情報の収集，構造化および検索手法. 情報処理学会論文誌, Vol. 38, No. 3, pp. 606–615, 1997.

[IWATE] IWATE・UNU・NTT 環境ネットワーク共同プロジェクト. http://www.mcon.ne.jp/iwate/summit/index.html.

[環境省] 環境省 大気汚染物質広域監視システム. http://w-soramame.nies.go.jp/.

[間瀬 他 01] 間瀬茂, 武田純. 空間データモデリング – 空間統計学の応用, データサイエンスシリーズ, Vol. 7. 共立出版, 2001.

[岸野 他 00] 岸野文郎, 佐藤隆夫, 横矢直和, 相澤清晴, 有川正俊. 画像と空間の情報処理, 岩波講座 マルチメディア情報学, Vol. 5. 岩波書店, 2000.

[吉野 他 98] 吉野利明, 寺本良明, 川村旭, 益岡竜介, 吉田武俊. ファシリテータを利用した商用データベースの統合. 電子情報通信学会論文誌, Vol. J81-D-I, No. 5, pp. 460–467, 1998.

[Kyoto] デジタルシティ京都. http://www.digitalcity.gr.jp.

[高橋 他 98] 高橋克巳, 三浦信幸, 西部喜康, 島健一. 不均一で情報の構造情報集と関連付けによる統合 – 情報統合ディレクトリ – . 電子情報通信学会論文誌, Vol. J81-D-I, No. 5, pp. 443–450, 1998.

[NSDIPA] 国土空間データ基盤推進協議会. http://www.nsdipa.gr.jp.

[国土交通省 a] 国土数値情報ダウンロードサービス. http://nlftp.mlit.go.jp/ksj/.

[国土交通省 b] 国土交通省. 川の防災情報. http://www.river.go.jp/.

[GSI] 国土地理院. http://www.gsi.go.jp/.

[山崎 他 92] 山崎弘郎, 石川正俊. センサフュージョン –実世界の能動的理解と知的再構成. コロナ社, 1992.

[宗像 他 00] 宗像浩一, 吉川正俊, 植村俊亮. 鮮度と同期度に基づく周期データの選択方式. 情報処理学会論文誌：データベース, Vol. 41, No. SIG1(TOD5), pp. 140–153, 2000.

[小方 他 98] 小方登, 小長谷一之, 碓井照子, 酒井高正. GIS 原典 – 地理情報システムの原理と応用 [I] –. 古今書院, 1998.

[浅野 他 00] 浅野哲夫 (訳), M. ドバーグ, M. ファン. クリベルド, M. オーバマーズ, O. シュワルツコップ (共著). コンピュータ・ジオメトリ 計算幾何学：アルゴリズムと応用. 近代科学社, 2000.

[相良 他 00] 相良毅, 有川正俊, 坂内正夫. ジオリファレンス情報を用いた空間情報抽出システム. 情報処理学会論文誌：データベース, Vol. 41, No. SIG 6 (TOD 7), pp. 69–80, 2000.

[村井 02] 村井俊治. 空間情報工学 (改訂版). 日本測量学会, 2002.

[張 01] 張長平. 地理情報システムを用いた空間データ分析. 古今書院, 2001.

[CSIS] 東京大学空間情報科学研究センター. CSV アドレスマッチングサービス. http://spat.csis.u-tokyo.ac.jp/cgi-bin/geocode.cgi.

[日本気象協会] 日本気象協会. ワンクリック気象情報サイト. http://tenki.jp/.

[JMC] 日本地図センター. http://www.jmc.or.jp.

[JMC 98] 日本地図センター. 数値地図ユーザーズガイド (第 2 版補訂版), 1998.

[平松 他 00a] 平松薫, 石田亨. 地域情報サービスのための拡張 web 空間. 情報処理学会論文誌：データベース, Vol. 41, No. SIG 6 (TOD 7), pp. 81–90, 2000.

[平松 他 00b] 平松薫, 小林堅治, Ben Benjamin, 石田亨, 赤埴淳一. デジタルシティにおける情報検索のための地図インタフェース. 情報処理学会論文誌, Vol. 41, No. 12, pp. 3314–3322, 2000.

[北村 他 02] 北村泰彦, 野田知哉, 辰巳昭治. 動的情報メディエータのための知的情報手法. 電子情報通信学会論文誌 D-I, Vol. J84-D-I, No. 8, pp. 1256–1265, 2002.

[明野 他 97a] 明野和彦, 熊木洋太. 連載解説「地理情報システム」, 1. 地理情報システムと空間データインフラの概要. 情報処理, Vol. 38, No. 2, pp. 143–149, 1997.

[明野 他 97b] 明野和彦, 熊木洋太. 連載解説「地理情報システム」, 2. 空間データの標準化と整備の動向. 情報処理, Vol. 38, No. 3, pp. 232–238, 1997.

[野上 他 01] 野上道男, 岡部篤行, 貞広幸雄, 隈元崇, 西川治. 地理情報学入門. 東京大学出版会, 2001.

論文目録

【 主論文に関する公刊論文 】

1. 白石 陽, 安西 祐一郎, “空間検索のためのインクリメンタルなデータ提供手法”, 情報処理学会論文誌：データベース, Vol.42 No.SIG 15 (TOD 12) , pp.15–29, (2001 年 12 月).

2. 白石 陽, 安西 祐一郎, “インタラクティブなセンサデータ閲覧のためのオンライン空間集約手法”, 日本データベース学会 Letters, Vol.2, No.1, pp.51–54, (2003 年 5 月).

3. 白石 陽, 安西 祐一郎, “分散センサデータの閲覧のためのインクリメンタルなデータ提供方式”, 情報処理学会論文誌：データベース, Vol.44, No. SIG 12 (TOD 19), pp.123–138, (2003 年 9 月).

4. 白石 陽, 安西 祐一郎, “パーソナル空間情報システムのためのセンサデータマッピングフレームワーク”, 電子情報通信学会論文誌, Vol.J87-A, No.1, pp.96–107, (2004 年 1 月).

5. 白石 陽, 安西 祐一郎, “センサデータの視覚化のためのインクリメンタルな空間集約手法”, 情報処理学会論文誌：データベース, TOD 22, (採録決定)

【 国際会議発表 】

6. Yoh Shiraishi, Yuichiro Anzai, “Sensor Data Retrieval from Distributed Sensor Networks”, Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications, pp. 1501 – 1507, Las Vegas, Nevada, USA, (Jun. 2000).

7. Yoh Shiraishi, Yuichiro Anzai, “Incremental Spatial Data Providing for Internet GIS”, Proceedings of International Symposium on Asia GIS (CD-ROM), Tokyo, Japan, (Jun. 2001).

8. Yoh Shiraishi, Yuichiro Anzai, "A Retrieval Method for Real-time Spatial Data Browsing", Proceedings of 12th International Conference on Database and Expert Systems Applications, DEXA'01, (Lecture Notes in Computer Science 2113, Springer), pp.460 – 469, Munich, Germany, (Sep. 2001).

【 国内学会発表 】

9. 白石 陽, 安西 祐一郎, "センサネットワークアプリケーションのための センサデータエージェントの設計と実装", 第 7 回マルチ・エージェントと協調計算ワークショップ (MACC'98) オンラインプロシーディングス, 日本ソフトウェア科学会, (1998 年 12 月).

10. 白石 陽, 桜井 利彦, 鈴木 英之, 河野 通宗, 安西祐一郎, "Web ブラウザを利用したセンサノード群の 3 次元視覚化インタフェースの提案と実装", インタラクション' 99 論文集, 情報処理学会, pp. 163 – 164, (1999 年 3 月).

11. 白石 陽, 安西 祐一郎, "インクリメンタルなデータ転送機構を持つ空間データ提供システムの提案と実装", 電子情報通信学会データ工学研究会, 第 12 回データ工学ワークショップ (DEWS2001), CD -ROM 予稿集, (2001 年 3 月).

12. 白石 陽, 安西 祐一郎, "リアルタイム情報統合のためのセンサデータおよび空間データの検索機構の提案", 情報処理学会データベースシステム研究会と電子情報通信学会データ工学研究専門委員会合同ワークショップ（DBWS2001）, 情報処理学会研究報告 2001-DBS-125(I), pp. 201 – 208, (2001 年 7 月).

13. 白石 陽, 安西 祐一郎, "分散センサデータの視覚化のためのインクリメンタルな空間統合システム", 第 5 回空間 IT ワークショップ予稿集, 地理情報システム学会, pp.4–11, (2002 年 11 月).

14. 白石 陽, 安西 祐一郎, "位置情報に基づくセンサデータ統合のための逐次データ提供方式", 情報処理学会データベースシステム研究会, データベースと Web 情報システムに関するシンポジウム (DBWeb2002), pp.153–160, (2002 年 12 月).

15. 白石 陽, 安西 祐一郎, "インタラクティブなセンサデータ閲覧のためのオンライン空間集約手法", 第 14 回データ工学ワークショップ (DEWS2003), 電子情報通信学会, 第 1 回日本データベース学会年次大会, オンライン予稿集, (2003 年 3 月).

16. 白石 陽, 安西 祐一郎, “センサデータの視覚化のためのインクリメンタルな空間集約手法”, データベースと Web 情報システムに関するシンポジウム (DBWeb2003), pp.251–258, (2003 年 11 月).

【 同一著者による他の研究成果 】

17. 白石 陽, 納谷 太, 安西 祐一郎, “インクリメンタルな制約解消機構を用いたロボットタスクスケジューラの設計と実装”, 情報処理学会第 48 回全国大会論文集 (2), pp.103 – 104, (1994 年 3 月).

18. Yoh Shiraishi, Yuichiro Anzai, “Task Planning Based on Human-Robot Interaction for Autonomous Mobile Robots”, Proceedings of 5th IEEE International Workshop on Robot and Human Communication, pp. 445 – 450, Tsukuba, Japan, (Nov. 1996).

19. 中内 靖, 岩倉 裕子, 白石 陽, 安西 祐一郎, “実世界指向分散ヒューマンインタフェースシステムの提案”, 情報処理学会 ヒューマンインタフェース研究会研究報告 74-9, pp.49 – 54, (1997 年 9 月).

20. Yuko Iwakura, Yoh Shiraishi, Yasushi Nakauchi, Yuichiro Anzai, “Real-world Oriented Distributed Human-Robot Interface System”, Proceedings of 6th IEEE International Workshop on Robot and Human Communication, pp.188 - 193, (Sep. 1997).

21. Yuko Iwakura, Yoh Shiraishi, Yasushi Nakauchi, Yuichiro Anzai, “Multi-agent Interface Architecture for Real-world oriented Distributed Human Interface Systems”, Proceedings of IEEE International Conference on Systems, Man and Cybernetics, pp.4115 - 4120, (Oct. 1997).

22. 岩倉 裕子, 白石 陽, 中内 靖, 安西 祐一郎, “実世界指向分散ヒューマンインタフェースシステムのためのマルチエージェント・インタフェースアーキテクチャ”, 第 6 回マルチ・エージェントと協調計算ワークショップ (MACC’ 97), (1997 年 12 月).

23. 岩倉 裕子, 白石 陽, 中内 靖, 安西 祐一郎, “ユーザの行動の不確実性に対応した実世界指向分散ヒューマンインタフェースシステムの提案”, 情報処理学会論文誌, Vol.39, No.5, pp.1502 -1513, (1998 年 3 月).

24. Michimune Kohno, Mitsuhiko Ohta, Yoh Shiraishi, Yuichiro Anzai, “Distributed Sensor Network Management System for Dynamic Network

Reconfiguration", Proceedings of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications, pp.21 - 28, Las Vegas, Nevada, USA, (Jul. 1998).

25. 米山 華子, 白石 陽, 安西 祐一郎, "センサネットワーク利用のためのエージェントプラットフォームの実装", 第 7 回マルチ・エージェントと協調計算ワークショップ (MACC' 98) オンラインプロシーディングス, (1998 年 12 月).

26. 河野 通宗, 太田 充彦, 白石 陽, 安西 祐一郎, "開放分散環境のためのセンサノード管理システム", 情報処理学会論文誌, Vol.40, No.3, pp.1276–1284(1999 年 3 月).

27. 鈴木 英之, 白石 陽, 安西 祐一郎, "広域環境情報を利用する移動ロボットのための経路生成手法", 第 13 回人工知能学会全国大会, (1999 年 6 月).

28. 梅澤 猛, 白石 陽, 安西 祐一郎, "センサネットワーク利用のためのモバイルエージェントプラットフォームの設計と実装", 日本ソフトウェア科学会 第 16 回大会論文集, pp.417 - 420, (1999 年 9 月).