Vol.2019-GN-106 No.34
Vol.2019-CDS-24 No.34
Vol.2019-DCC-21 No.34
2019/1/25

汎用的な地図フォーマットに対応可能な地図データキャッシュアルゴリズムの検討

影山勝彦[†1]　松本貴士[†1]　馬場哉也[†2]　織田勇[†2]

概要：より快適なカーナビゲーションのユーザインタフェースの実現のため，地図画面描画の高速化の研究を行っている．近年，NDS（Navigation Data Standard）に挙げられるような汎用性を意識した地図データフォーマットの規格化が進み，採用事例も増えつつある．既存の地図データ読み込みロジックに対して別の地図フォーマットを適用した場合に，地図データ読み込みのオーバヘッドが増大してアプリケーションの性能に影響するという課題がある．そのような課題に対応するために，筆者らは，汎用的な地図フォーマットへの対応を想定したキャッシュアルゴリズムを提案した．本報告では地図描画機能に着目して試作を行い，NDS 地図データを用いた評価を行った．評価結果により，地図描画のためのデータ処理時間を短縮できることを確認し，地図描画について本方式の有効性を検証した．

キーワード：カーナビ，地図フォーマット，キャッシュアルゴリズム，地図描画

Investigation of a map data caching algorithm supporting general map data formats for car navigation system

KATSUHIKO KAGEYAMA[†1]　TAKASHI MATSUMOTO[†1]
NAOYA BABA[†2]　ISAMU ODA[†2]

Abstract: In order to realize more comfortable user interface, we are investigating faster drawing map method. Recently general purpose map data formats are being standardized like NDS (Navigation Data Standard). Some commercial products are also in the worldwide market. If existing navigation system applies new map data format, it may affect application performance because of overhead to read and analyze information from the map data. In order to avoid the problem, we proposed a map data caching algorithm supporting general map data formats for car navigation system. We focused on the function of drawing map, implemented our idea, and measured performance using NDS map data. We confirmed that the algorithm could improve the performance of preparing data for drawing map.

Keywords: Car Navigation, Map Format, Caching Algorithm, Drawing Map

1. はじめに

カーナビゲーション（カーナビ）において，地図の表示，運転経路の検索，運転の誘導案内などの主要な機能は地図データを用いて実現している．国単位で広範囲をカバーする地図データは非常に多くの情報を含んでおり，それらの情報をいかに効率よく参照して利用するかによって，カーナビの性能に大きな違いが生じる．地図データには地図ベンダー独自のフォーマット以外にも KIWI フォーマット[1]や，Navigation Data Standard(NDS)[2]，Geographic Data Files(GDF)[3]等の標準フォーマットが存在している．

カーナビの開発においては，製品仕向け国や，顧客からの要望によっては，状況に応じて複数の地図データフォーマットへ対応する必要が発生する状況がありえる．開発コストを抑えつつそのような状況に対応するには，一つのカーナビシステムで複数の地図フォーマットへ対応できる仕組みが求められる．しかしフォーマットによっては，地図データの解析に必要となる処理時間に差がある場合や，ターゲットとするカーナビシステムが期待する地図データ構造と，地図のフォーマットがそのままでは適合しない場合がありえる．

汎用的な地図フォーマットがカーナビシステムの期待する既存の地図フォーマットと適合しない場合，地図データを利用するために，カーナビシステム向けの地図データの変換処理を行う必要が生じる．例えば地図を描画しカーナビ画面に表示する場合，描画のために必要となる道路形状座標列データや属性情報等を，地図データから所定のフォーマットに変換した後に，その変換済みデータを用いて実際の描画を行う必要がある．もしこのような変換処理の処理時間が長い場合，変換処理時間がオーバヘッドとなり描画全体の処理時間が長くなり，ユーザの快適な操作性を損なってしまう．

本研究は，カーナビの性能への影響を最小限にしつつ汎用の地図フォーマットに対応可能とするデータ構造及びデータキャッシュアルゴリズムを検討することを目的する．本報告では，製品の仕向け毎に異なる地図データを用いる

†1 株式会社日立製作所 研究開発グループ
Hitachi Ltd. Research & Development Group

†2 クラリオン株式会社
Clarion Co., Ltd

Vol.2019-GN-106 No.34
Vol.2019-CDS-24 No.34
Vol.2019-DCC-21 No.34
2019/1/25

カーナビを対象として，地図データのフォーマットによって地図データの解析時間が異なる条件であっても，カーナビが備える地図データを利用した地図表示機能や運転経路の探索機能などの基本機能への影響を軽減する地図データキャッシュの手法について提示し，特に地図描画に関して試作評価を行う．

2. 課題検討

地図フォーマットによらず，カーナビの操作性を快適に保つための要件と，本研究における課題を示す．

2.1 検討の前提条件

多様な地図データフォーマットに対応するために，地図データの格納フォーマットについて，以下のような点を前提条件として検討を行った．

地図の基本的な道路や構造物の形状などのデータは，データの管理を容易にするために，KIWI や NDS のように階層化された格子状のデータ形式で管理されているものとする．格子それぞれをタイルと呼称する．図 2-1 にデータ構造のイメージ図を示す．タイルは地表を所定の間隔の緯度経度で格子状に区分した領域であり，そのタイルに対応するように描画関連データや経路関連データが地図データに格納されている．またタイルは地図縮尺に対応する階層構造を有しており，ある縮尺を Level X とした場合，その上位の Level X-1 には Level X で表現される複数枚のタイルの範囲を一枚のタイルで表現する．Level の増減と縮尺の対応関係については地図フォーマットによって異なるが，本報告内では，便宜上 Level が低いほどより概略の情報を表現するものとする．

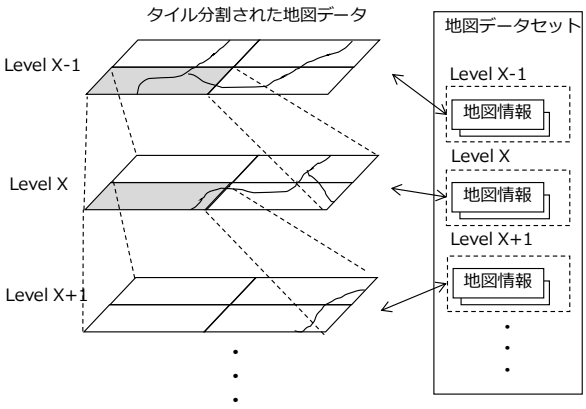

図 2-1 階層化されたタイル状データ構造
Figure 2-1 Hierarchical tiled data structure

地図データセットは，様々な用途に応じて使い分けが容易になるよう描画用の情報以外にも道路のリンク情報，名称情報，POI 情報など種別ごとに分けて格納されていることを前提とする．情報によってはタイル状に分割されず座標，ID などで検索可能なフォーマットである場合も想定する．

図 2-2 には NDS の地図データセットを用いて東京都新宿区近辺を描画した例を示す．図を 6 つに分割する区画線はタイルの境界線を表している．NDS の場合は，図 2-2 のような道路の形状を描画するために，描画関連のデータに経路関連のデータを加味する必要がある．また道路名称などの名称データも独立したデータとなっている．図 2-2 のような地図画面全体を描画するためには，これら複数の地図データを参照することが必要となる．

図 2-2 地図画面例（新宿近辺）
Figure 2-2 Example of drawing map(Shinjuku area)

2.2 快適な操作性を維持するための要件

地図データフォーマットの違いによる，画面更新の遅延や，ユーザ操作への応答の遅延，などが生じないようにするために，以下のような要件をあげた．

(1)地図画面を 1 枚描画するための実描画処理時間に対して，地図データの参照にかかる時間を十分小さくすること．
(2)経路の探索処理に対して，地図データの参照時間にかかる時間を十分小さくすること．
(3)走行中の経路誘導案内処理に対して，地図データの参照時間にかかる時間を十分小さくすること．

地図画面の描画は各種操作の結果表示等で多数行われるため，この地図データの参照時間も含めた総描画処理時間が長い場合，ユーザが操作のもたつきを感じることに繋がる．そのため(1)の要件を定義した．目的地までの経路探索処理および探索した経路に基づく走行中の誘導案内処理はカーナビとしての基本機能である．また走行中は処理の遅れは適切なタイミングでのドライバーへの情報提供を阻害する．これらの理由により(2)(3)の要件を定義した．

2.3 解決すべき課題

地図データフォーマットによって最適な解析方法やデータの蓄積方法は異なると考えられる．ある地図データフォーマットに特化した地図データの解析方式を，その他の地図データフォーマットに流用した場合，地図データ解析時間が長くなってしまう場合がある．

地図データの解析に時間がかかると，地図の描画や経路

Vol.2019-GN-106 No.34
Vol.2019-CDS-24 No.34
Vol.2019-DCC-21 No.34
2019/1/25

の探索処理などの地図データを利用した処理全般についての処理時間遅延につながり，要件を満たすことが困難になる．

地図データの解析時間が長くなる場合，アプリケーションが必要とする地図情報を，メモリ上でキャッシュすることが一般的に有効であると考えられる．地図情報をキャッシュする場合，下記のような課題がある．

(1)キャッシュデータ構造

地図描画や経路探索などのカーナビ機能それぞれによって地図データの利用方法は異なる．単に地図データをメモリ上に格納するだけでなく，カーナビ機能に適したキャッシュデータのデータ構造を考慮しなくてはいけない．

(2)キャッシュ生成処理の重複の回避

地図データのフォーマットによっては，地図の描画や経路の探索の機能を実現するために，複数の地図データの種別をまたがって参照する必要がある場合がある．

複数の機能で共通して参照が必要な地図データについては，キャッシュ生成時に機能毎に都度解析するのは無駄が多く，効率化が必要である．

(3)キャッシュ生成処理のタイミングの最適化

キャッシュ生成時間をユーザに意識させないために，直近に必要になる可能性の高い地図情報を進行中の処理を阻害することなく，適切に先読みするなどのキャッシュ生成タイミングの考慮が必要である．

(4)キャッシュの更新

カーナビが搭載するメモリは有限であるため，不要なキャッシュを削除し，直近必要になる可能性が高い情報をキャッシュする必要がある．

3. 関連研究

本研究に関連する過去の研究について示す．

3.1 キャッシュデータの管理方法

2.3 節(1)(2)で述べたようなキャッシュデータの管理方法に関する課題について，以下のような先行研究がある．

(1)複数のデータオブジェクトを統合してキャッシュを生成する際に，関連性の高いデータオブジェクトをまとめてキャッシュを生成する手法[4].

(2)スケールを考慮してタイルデータのIDをつけることで，タイルキャッシュの管理を効率化する手法[5].

本研究ではカーナビ機能が複数のデータを重複して参照する場合がある点に着目して管理方法を検討する．

3.2 キャッシュ生成処理のタイミングに関する手法

2.3 節(3)で述べたようなキャッシュデータの生成処理のタイミングに関する課題について，以下のような先行研究がある．

(1)WebGIS のタイルデータの先読み手法

タイル毎のアクセス可能性情報がある前提で，先読みするべきタイルデータを決める手法[6].

本研究では，タイルごとのアクセス可能性に関する事前情報は存在しないため，カーナビ機能の特性を考慮して地図データの先読みを行うことを検討する．

3.3 キャッシュ更新の手法

2.3 節(4)で述べたような課題について以下のようなキャッシュの置き換え手法が存在している．

(1)LFU 法

古典的な手法：参照頻度が少ないキャッシュを優先して破棄する

(2)LRU 法

古典的な手法：最後に参照されたキャッシュを優先して破棄する

(3)LFU-FF 法

タイルの隣接条件と参照回数を元に優先度を決定する手法[7].

既存の研究では条件を限定しない効率化の方法が検討されているが，本研究ではカーナビの機能により最適化した方法を検討する．

4. 提案手法

本章では，2.3 節で示した課題を解決するための手法を提案し，その内容について説明する．

4.1 概要

2.3 節(1)(2)の課題に対して，カーナビ機能間の参照を考慮した解析済み地図データのキャッシュデータ構造を提案する（4.2 節）．2.3 節(3)(4)の課題に対して，特に地図画面描画の高速化に着目して，地図画面遷移を考慮したキャッシュデータの先読み生成手法（4.3 節）および描画範囲を考慮したキャッシュの置き換え手法（4.4 節）を提案する．

キャッシュデータ構造の基本的な設計方針はカーナビ機能全体に適用可能であることを考慮している．2.3 節(3)(4)の課題に対しては，本研究ではまず影響範囲が大きい地図画面描画機能に着目して対応方法を検討した．これらの提案手法は組み合わせて適用することで，地図描画性能改善に効果を発揮することが期待される．

4.2 機能間参照を考慮したキャッシュデータ構造

提案手法の機能間参照を考慮したキャッシュデータ構造を図 4-1 に示す．主な特徴は，地図データセットから生成したキャッシュデータをデータ解析層と機能層の２層に分けてキャッシュする点である．データ解析層のキャッシュは地図データセット固有のデータ構造を継承したキャッシュであり，地図データフォーマットごとに固有である．対して機能層キャッシュはタイルごとに各種機能毎に最適化されたデータ構造を有しているものとする．これは地図データフォーマットの影響は受けないものとする．

このような構造とすることで，地図データフォーマットの仕様の違いをデータ解析層キャッシュで吸収し，複数の機能が地図データの種別を重複して参照する場合でも，都

Vol.2019-GN-106 No.34
Vol.2019-CDS-24 No.34
Vol.2019-DCC-21 No.34
2019/1/25

度地図データ解析を行なわずに済ませることが可能になる．

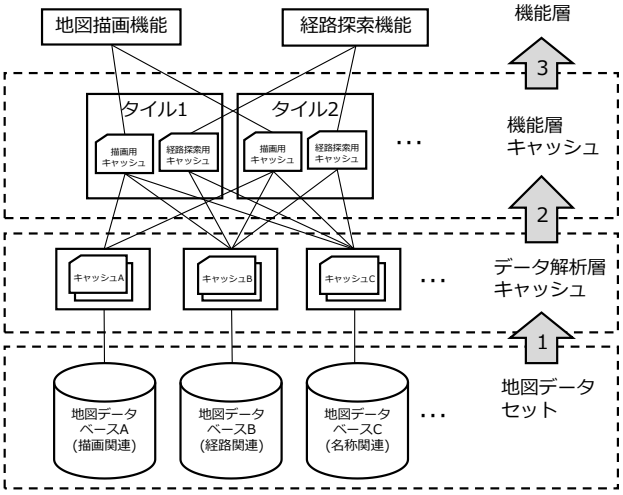

図 4-1 提案手法のキャッシュデータ構造

Figure 4-1 Proposed cache data structure

図 4-2 に，描画のために必要なタイルキャッシュデータを作成する手順を示している．キャッシュの生成は，地図データセット→データ解析層キャッシュ→機能層キャッシュの順に行われ，地図描画機能や経路探索機能が機能層キャッシュを用いて機能を実現する．データ解析層キャッシュでは地図データセットの種別ごとに解析された解析済みデータをキャッシュする．機能層キャッシュデータはタイルの ID をキーとして検索できるようにする．

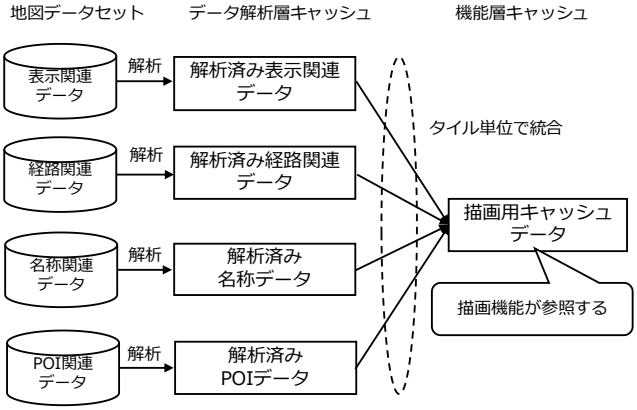

図 4-2 キャッシュデータの作成フロー

Figure 4-2 Flow diagram of making cache data

4.3 地図画面遷移を考慮したキャッシュの先読み生成

キャッシュ生成をバックグラウンドで行うために，以下のようにアルゴリズムを定めた．地図の描画を想定した場合，ユーザが行う可能性の高い操作を想定して，その操作に必要なタイルの範囲を先読みするものとする．キャッシュ生成範囲の指定は機能層キャッシュをベースに行う．

本検討では，ユーザの操作として，地図のスクロール操作，地図の縮尺拡大縮小の操作を想定し下記のような先読みの方針を定めた．

(1) 地図表示中心から 5×5 の範囲のタイルを先読みする

(2) 地図表示中の Level+1 の中心タイルを起点に 3×5 の範囲のタイルを先読みする

(3) 地図表示中の Level-1 の中心タイルを起点に 3×5 の範囲のタイルを先読みする．

表示中の Level では 5×5 の範囲のタイルを先読みする．これは，ユーザがスクロール操作などで参照する可能性が高いためである．

表示中 Level± 1 では 5×3 の範囲のタイルを先読みする．これは，表示中 Level に次いでユーザが縮尺変更の操作で参照する可能性が高いと想定したためである．経度方向に長いのは，タイルの分割が緯度経度基準であるため，日本の緯度では，1 タイルの実距離換算の形状が縦長になるためである．

図 4-3 には，先読み範囲定義の例を示す．基準表示レベルを表示中であるため，表示位置を中心に 5×5 の範囲が先読み対象になっている．また上下のレベルにも方針に従って先読み対象タイルが選ばれている．

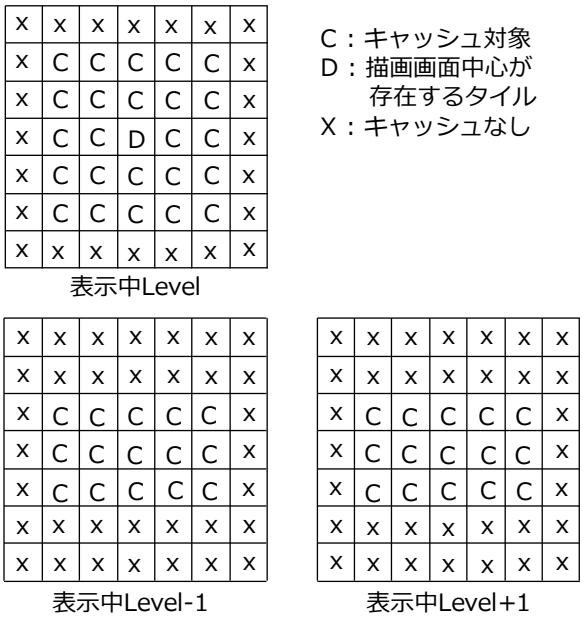

図 4-3 先読み範囲の定義

Figure 4-3 Definition of pre-read area

4.4 描画範囲に基づくキャッシュ置き換え手法

メモリは有限であるため，キャッシュの解放のタイミングも重要である．タイルキャッシュテーブル上のキャッシュの数にも上限を設定している．4.3 節の方針を元にした場合，5×5＋5×3＋5×3=55 が最低必要となる機能層キャッシュの数である．本検討では 55 の約 2 倍の 100 枚をキャッシュ作成数の上限とした．

キャッシュが上限に達した場合に新規にキャッシュをメモリ上に登録しようとする場合，既存のキャッシュを削除する必要がある．キャッシュの削除では機能層キャッシュ

Vol.2019-GN-106 No.34
Vol.2019-CDS-24 No.34
Vol.2019-DCC-21 No.34
2019/1/25

と機能層キャッシュに対応するデータ解析層のキャッシュを削除する．単に古いキャッシュから順番に削除してしまうと，地図表示範囲にあまり大きな変化が無かった場合，地図描画で使用中のキャッシュを削除してしまう可能性がある．

このような状態を回避するために下記のようなポリシーで削除を行うこととした．基本的な方針として，表示中心位置からできる限り遠いキャッシュから順番に消していく，ということを意図している．

(1) タイル毎に評価値を定め，評価値の大きいタイルのキャッシュから削除する．

(2) 表示中 Level における評価値は表示中心位置からタイルのマンハッタン距離とする．

(3) 上下の Level における評価値は表示中心位置からタイルのマンハッタン距離に N を加えたものとする．N は表示中 Level のタイル上で想定される最大評価値よりも大きい値とする

(4) 描画のために参照中のキャッシュは削除しない

(5) 読み込み中のキャッシュは削除しない（データの一貫性を保つため）

図 4-4 キャッシュの削除優先順位

Figure 4-4 Order of deleting cache data

図 4-4 に，キャッシュ削除の優先順位の例を示す．図中では，タイル内の数字が中心からの距離を示している．キャッシュのデータが上限に達したという前提で見た場合，最も中心から遠いタイルキャッシュは，表示中 Level-1 の N+6 のタイルキャッシュである．もしこのキャッシュが読み込み中または参照中ではない場合，このタイルキャッシュが最初に削除される．

本節で示した手法を適用することで，2.3 節に挙げた各問題点について下記のような効果が得られることが期待できる．

(1) データ読み込みのオーバヘッド削減

データセットからの読み込みが完了した後のデータをキャッシュしているため，キャッシュデータを参照する際には，データセットを読み込む必要がなくなる．

(2) 地図データの解析処理のスキップ

解析処理を施した結果の地図データをキャッシュすることで，処理時間の掛かる解析処理をスキップできる．

(3) 機能間の連携

キャッシュデータを作成する際に，1 枚のキャッシュデータにタイルに関連するデータベースから抽出した解析済みデータをセットでキャッシュすることで，カーナビ機能がキャッシュを参照する際に，データベース間の関係性を意識する必要がなくなる．

(4) 拡大縮小の際のタイルデータ読み込み

現在表示中のレベル±1 のレベルのタイルもキャッシュ対象に含めることで，拡大縮小の際に表示中レベルと隣接するレベルのタイルもキャッシュ済みである可能性が高くなり，キャッシュを利用して高速に描画することが可能になる．

5. 実装と評価

特にユーザが性能への影響を認知しやすいカーナビの地図描画機能に対して提案方式のアプリケーション用キャッシュを実装して評価を行った．図 5-1 には描画処理とキャッシュ生成処理の関連を示した．キャッシュ生成処理は先読みの方針に基づいてキャッシュ生成処理スレッドが独立して行う(1)．キャッシュ生成処理によって生成された機能層キャッシュおよびデータ解析層キャッシュはメモリ上に格納される(2)．描画スレッドは必要に応じて機能層キャッシュを参照して地図画面の描画を行う(3)(4)．もしキャッシュが生成されていなかった場合は描画スレッドがキャッシュ処理スレッドに通知を行い(5)キャッシュデータの生成を待ってから描画を行う．

地図データには，日本の地図データを格納した NDS を利用した．

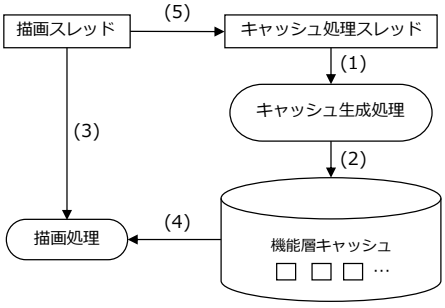

図 5-1 キャッシュ処理機能の実装概要

Figure 5-1 Implementation of caching function

Vol.2019-GN-106 No.34
Vol.2019-CDS-24 No.34
Vol.2019-DCC-21 No.34
2019/1/25

5.1 評価方法

評価には，図 2-2 で示した新宿近辺の描画に掛かる時間をカーナビの実機上で測定した．描画には 6 枚のタイルデータの参照が必要である．評価では提案手法を用いた場合と用いなかった場合それぞれについて測定を行った．測定はタイルごとの描画に必要な情報の読み込み解析時間と，実描画処理に掛かった時間を分けてスレッド時間を用いて測定した．実機上ではバックグラウンドでの各種 OS 処理等の影響もあるため 5 回測定しその平均を用いた．

5.2 測定結果

測定結果を表 5-1 に示す．

表 5-1 測定結果

Table 5-1 Result of measurement

	手法適用前	手法適用後
(A)タイル読込み時間	1366 msec	0.454 msec
(B)描画時間	52.6 msec	53.2 msec
総描画時間(A+B)	1422.6 msec	53.5 msec

キャッシュが作成済みの場合，0.454msec で 1 画面の描画に必要な 6 タイル分の地図データの参照が完了している．例えば映画のフレームレートが 24fps であること，すなわち一枚の画像の表示時間が約 41msec であることを考えると，描画を滑らかに行うために十分高速であると言える．

また画面のスクロールを行う場合，図 5-2 に示すように新しい描画範囲のタイルを参照する場合，２～４枚のタイルを参照する必要があるが，その場合でも 0.454msec 以内であると推測できるため，参照時間については十分高速ということが出来る．同様に地図画面の縮尺の拡大縮小についても同様の効果が見込まれる．参照時間に対して描画時間が約 53msec と比較的長いが，実験での実描画機能は試作品であるため，実描画処理の性能については本報告では考察しない．

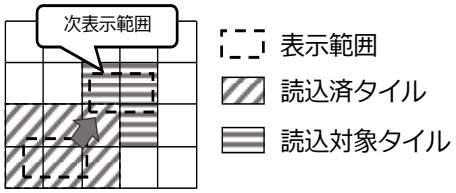

図 5-2 読み込み必要タイル範囲

Figure 5-2 Area to read tile data

6. おわりに

測定の結果，キャッシュが作成済みの場合，0.454msec で 6 タイル分の地図データの参照が完了しており，描画に求められる性能に対して十分高速であり，本手法の描画性能に関する効果を確認できた．

研究課題として以下のような点がある．今後これらの課題に取り組んでいく予定である．

- 地図描画機能以外の経路探索機能や，誘導案内機能も含めたカーナビシステム全体への手法の適用と評価を行い，２層のキャッシュ構造の効果を検証すること．
- NDS 以外の地図データフォーマットへ適用した場合についても評価を行い，結果を比較検証すること．
- 先読みの手法，キャッシュ置き換え手法について，現状は比較的シンプルな手法であり改善の余地があるため，よりカーナビシステムに最適化したアルゴリズムを考案し，その他の手法との比較検討を行うこと．

参考文献

[1] Kiwi フォーマット仕様書 Ver.1.22, http://kiwi-w.org/format/format_kihon.html
[2] Navigation Data Standard (NDS), NDS Association, http://www.nds-association.org/
[3] Geographic Data Files 5.0 ISO14825:2011
[4] 沈 虹,陳 漢雄, 山口 和紀, 北川 博之, 大保 信夫, 藤原 譲, "LRU-S：複合オブジェクト間の参照情報を用いるバッファ管理手法", 全国大会講演論文集 第 45 回(ソフトウェア), pp.123-124, (1992)
[5] Yansheng Zhang, Dancheng Li, Zhiliang Zhu, "A Server Side Caching System for Efficient Web Map Services", 2008 International Conference on Embedded Software. and Systems Symposia, pp.32-37 (2008)
[6] Yong-Kyoon Kang, Ki-Chang Kim, Yoo-Sung Kim, "Probability-Based Tile Pre-fetching and Cache Replacement Algorithms for Web Geographical Information Systems", Proceeding ADBIS '01 Proceedings of the 5th East European Conference on Advances in Databases and Information Systems, pp.127-140 (2001)
[7] 田頭 茂明, 吉岡 浩路, 藤田 聡, "周辺情報検索におけるプロキシシステムのためのキャッシュ置換アルゴリズム", 情報処理学会論文誌, Vol.45, No.10, pp.2384-2394 (2004)