複合座標系の動的合成機能を有するFocus + Glue+Contextマップの実現とバス路線図への適用

竹内　健祐† 　金　 鎔煥† 　山本　大介† 　高橋　直久†

† 名古屋工業大学大学院工学研究科情報工学専攻　〒 466–8555 名古屋市昭和区御器所町
E-mail: †takeuchi@moss.elcom.nitech.ac.jp, ††{kim,yamamoto.daisuke,naohisa}@nitech.ac.jp

あらまし　我々は注目する地点を拡大表示する Focus 領域，その周辺の地域を表示する Context 領域，2 つの領域間の縮尺の違いによる歪みを吸収する Glue 領域の 3 つからなる Focus+Glue+Context マップを用いた Web マップシステム EMMA を提案してきた．しかし，場所によりスケールの異なる地図においては，複数のレイヤに分けて表示することができず，地図オブジェクトの描画・移動に関しては位置を計算して配置する必要があった．また，市バスに代表されるバス路線は，経路が複雑で本質的に分かり難い問題があった．上記の問題を解決するために本研究では Leaflet に代表される地図制御ライブラリを，可変スケール座標系の地図である Focus+Glue+Context マップに適用可能にする仕組みを提案する．これにより，均一スケール座標系の Leaflet と同様な操作・開発法で，Focus+Glue+Context マップを制御可能になる．また，路線データをクライアント側で読み込み，そのまま Focus + Glue+Context 型の Web マップで表示することのできる路線描画機能を実現する．また，提案システムの実験的評価により有効性を明らかにする．

キーワード　地理情報システム, Web マップシステム, Fisheye view, 路線描画, ナビゲーションシステム

1 はじめに

注目地点を拡大し周辺を縮小して視認性と一覧性を向上させる描画方式に関して古くから種々の研究 [1–3] がなされてきた．我々も，Focus+Glue+Context マップを提案し，伸縮自在のマップシステム EMMA（Elastic Mobile Map System）の研究を進めてきた [4–8]．Focus+Glue+Context マップは，図 1 に示すように，注目する特定の地域を拡大して表示する Focus 領域，周辺の地域を表示する Context 領域，スケールの異なる 2 つの領域間の縮尺率の違いによって生じる歪みを吸収する役割を果たす Glue 領域からなり，視認性と一覧性を両立させた Web マップを目指している．

また，OpenStreetMap [9] に代表されるオープンデータ形式の地図データや，Leaflet [10] や OpenLayers [11] に代表されるオープンソース型の地図制御用ライブラリ（以下，地図制御ライブラリと呼ぶ）の登場により，Web マップサービスは扱いやすいものとなった．地図制御ライブラリは，タイリング技術により早い応答速度で地図を描画することが可能であると同時に，マーカやポリゴンなどの地図オブジェクトを表示したり，多数のレイヤを重ね合わせて表示することができる．これらの特徴を生かし，文献 [12] のように，地図の情報量を増やすために他のサービスや情報をマッシュアップし，レイヤとして重ね合わせて表示することが可能になった．

一方で，既存の地図制御ライブラリは，場所により縮尺の変わらない均一スケール座標系での利用を前提としており，魚眼マップに代表されるような可変スケール座標系には適用困難であった．我々は，Focus+Glue+Context マップを Leaflet を用いて開発してきた [8] が，応答速度や，地図オブジェクトの描画に課題があり，地図制御ライブラリの特徴を生かしきれなかった．

また，市バスに代表されるバス路線は，都市部では路線数が 100 以上あるなど，経路が複雑で本質的に分かり難く，旅行者や外国人はもとより，地域住民であっても，事前の下調べ無しで容易に利用することができない．地域のバス路線の全体を俯瞰するためには，その地域の全域が表示できる広域の地図で描画できると望ましい．その一方で，バスを乗り換える際やバス停から目的地までの道を確認したい場合には，バス停付近の詳細な地図が表示できると望ましい．この相反する二つの要求を Focus + Glue+Context 型のマップは満たすことができるが，これまで実現されていなかった．

そこで本研究の目的は 2 つある．1 つ目は Leaflet に代表される地図制御ライブラリを，可変スケール座標系の地図である Focus+Glue+Context マップに適用可能にする仕組みを提案することである．これにより，均一スケール座標系の Leaflet と同様な操作・開発法で，Focus+Glue+Context マップを制御可能になる．2 つ目は前述の Focus + Glue+Context 型のマップにおいて路線図を描画する機能を提案することである．これらを実現する上での課題を以下に述べる．

課題 1　既存の地図制御ライブラリ [10, 11] は均一スケール座標系に対して，地図操作を可能にすることが前提のシステムである．そこで，複数の縮尺を持つ地図の差を吸収し，あたかも単一の縮尺の地図として動作可能にする仕組みが課題となる．

課題 2　既存の地図制御ライブラリは均一スケール座標系に対して，マーカやポリラインなどの地図オブジェクトを表示することが前提のシステムである．Focus+Glue+Context マップ

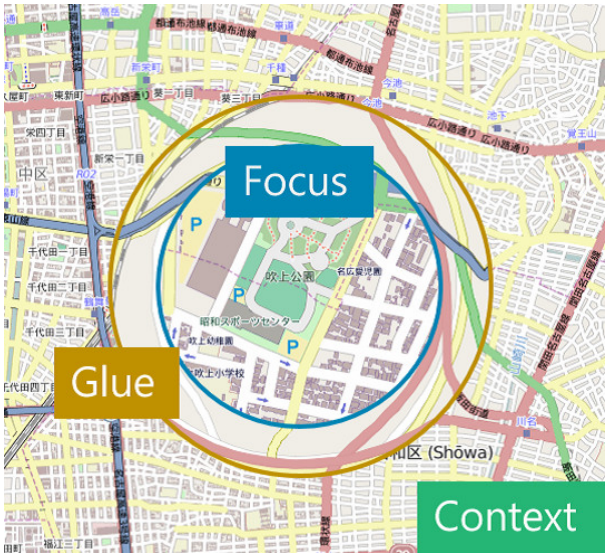

図 1　Focus+Glue+Context マップ

のような可変スケール座標系では，マーカやポリラインなどの地図オブジェクトを正しい位置に描画することが出来なかった．

課題 3　ユーザが路線図などのデータを新たに入力し，Focus+Glue+Context マップにおいて表示させたいとき，サーバ側でコードを編集・追加する場合では実現コストが高い．そのため，クライアント側での路線データを読み込み，Focus + Glue+Context 型の路線図の実現を容易にする仕組みが必要である．

これらの問題を解決するために以下の特徴を有する Focus + Glue + Context 型の Web マップシステム OpenEMMA を提案する．

EMMA レイヤ　表 1 のように，Focus の Leaflet（F-Leaflet）と Glue の地図画像（ラスタマップ）からなるレイヤ（EMMA レイヤ）を Context の Leaflet（C-Leaflet）に加えて，ネスト構造の Leaflet を構築する．各レイヤとサブレイヤの座標系を表 1 のように定めることにより，場所により縮尺を変えて描画する EMMA 方式に対応させる．これにより課題 1 を解決する．

複合座標系動的合成機能　F-Leaflet と Glue，C-Leaflet の間に入って各領域を制御する機能を実現し，縮尺の違いから生じる各領域間での表示の不整合が起きないように，表示位置や縮尺等を管理する．これにより課題 1 を解決する．また，縮尺が異なる各領域間で，地図オブジェクトを各領域の縮尺に応じて形状を変更したり表示位置を変更する．これにより課題 2 を解決する．

路線描画機能　路線データが入った GeoJson ファイルを読み込み，OpenEMMA 上に表示する．また，出発地と目的地，経由地などに Focus + Glue 領域を作成することで Focus + Glue+Context 型の路線図を実現する．これにより課題 3 を解決する．

2　関 連 研 究

一般に，人が頭の中に持つ地理的イメージ (認知地図 [13,14]) は，実際の地形と必ずしも一致せず，歪んでいることも多い．そこで，認知地図と実際の地形の対応関係の形成を助けるために，地図を歪めて表示させる研究は広く行われてきた [15,16]．

例えば，全体的な概要と詳細情報を一つの画面で表示することを目的とした Focus + Context 型マップに関する研究がある．Furnas ら [1] は，Focus からの距離に応じて周囲の情報の詳細度を変化させる魚眼レンズ状の情報表示手法について提案した．また，Sarkar ら [17,18] は，Focus+Context 型マップの概念を地理的な地図や図形に適用した．これらの論文では，単一の再配置関数を用いて，まるで魚眼レンズを当てたかのように地図全体を変形させる手法について述べている．また，Harrie ら [2] は，再配置関数を用いて，Focus 領域を歪み無く拡大して表示する代わりに，Context 全体に歪みを持たせて描画する手法について提案した．Craig ら [3] は，モバイル端末においての魚眼レンズマップの有用性を評価・提案している．Wang ら [20] は地下鉄マップに Focus+Context マップを適用している．これ [1–3,17,18,20] の可変スケール座標系マップの研究では，地図全体が歪んでいるため，Lealfet などの地図制御ライブラリを適用できず，画面全体を 1 枚の地図として描画している．

また，高橋 [4] は，再配置関数をベースにした地図描画に対して，歪みを吸収する Glue 領域を導入した Focus+Glue+Context 型マップを提案した．拡大で生じるマップの歪みを Glue に集中させることで，Focus と Context を歪みの無い均一なスケールで表示できる．山本ら [5] は，上記のシステムを Web マップサービスとして実現し，Web マップとして十分に許容される範囲の応答時間になることを実証しているが，オープンソース地図技術を用いていないクローズドなシステムであった．さらに，山本ら [8] では，提案手法と同様に，Leaflet と OpenStreetMap を用いて Focus+Glue+Context 型マップを実現した．しかしながら，1 か所（Context）のみしか Leaflet で描画しておらず，Focus はサーバーから一枚画像を取得して表示する方式であったため，Focus の応答速度や使い勝手に問題があった．

また，インタラクティブな地図を作るために，様々な地理的情報を，複数のレイヤに分けて表示することは一般的である．Zaslavsky [19] は，複数のレイヤを用いて XML ベースの空間データを統合した地図ソフトウェアを開発している．また，文献 [12] のように近年，地図の情報量を増やすために他のサービスや情報をマッシュアップし，レイヤとして重ね合わせて表示するものが増えている．Pikorec ら [21] は Web マップと関連付けて，多層ネットワークを描画することができる．しかし，これら [12,19,21] は，ベースとなる Web マップと同じ座標系を持つレイヤを重ね合わせることが前提のシステムであり，可変スケール座標マップなどの異なる座標系のレイヤを重ね合わせて表示することはできない．提案手法は，Focus + Glue+Context の異なる座標系のマルチレイヤ構造を持つと同時に，これらを正しく重ね合わせて表示できるシステムを提案し，可変スケールマップをベースにした位置情報サービスの実現を容易にする．

表 1　OpenEMMA のレイヤ

z-index	レイヤ	Leaflet	pane	サブレイヤ	Leaflet	座標系
200	Context	C-Leaflet	tile	-	-	Context
300	EMMA	C-Leaflet	focusglue	Glue	-	Glue
				Focus	F-Leaflet	Focus
400	道路・ポリゴン	C-Leaflet	overlay	-	-	Context
600	マーカ	C-Leaflet	marker	-	-	Context
700	ポップアップ	C-Leaflet	popup	-	-	Context

また，オープンデータとオープンソースを組み合わせた研究は多くある．たとえば，OpenStreetMap [9] に代表されるオープン地図データと，それを扱うオープンソースである Leaflet [10] や OpenLayers [11] を用いた研究 [23] [24] がある．さらに，航空交通 [22] や Linked Open Data [25] など，さまざまな形式のオープンデータが普及している．オープンデータとオープンソースを組み合わせたシステムを構築することの有用性は，新しい形式のオープンデータへの対応が容易であること，データとセットでシステムを提供することによりオープンデータの普及を促すことができることなどが挙げられる．

また，都市部のバス路線の路線図は，1 つの道路に対して 10 以上の路線が並走しているなど，とても複雑である．そのため，分かりやすいバス路線図の描画が求められるが，その実現は一般に難しい．この問題を解決する方式として，概略図を描画する方式 [20, 29] や，Web マップの道路上に直接描画する方式 [28] などがある．前者は，路線図が見やすい利点があるが，地図上のランドマークや道路との接続関係や位置関係が分かりにくい．後者は，地図上のランドマークや道路との接続関係は分かりやすいが，Web マップを拡大縮小するなどして全体の位置関係や詳細情報を確認する必要がある．

また，我々は [30] にて EMMA レイヤと複合座標系動的合成機能を有する OpenEMMA を提案した．本稿では汎用性の高い路線描画機能の導入により OpenEMMA を拡張し，名古屋市バス路線の描画に適用して，有効性を実験的に評価する．

3　提案システムの概要

提案システムは図 2 のようになる．

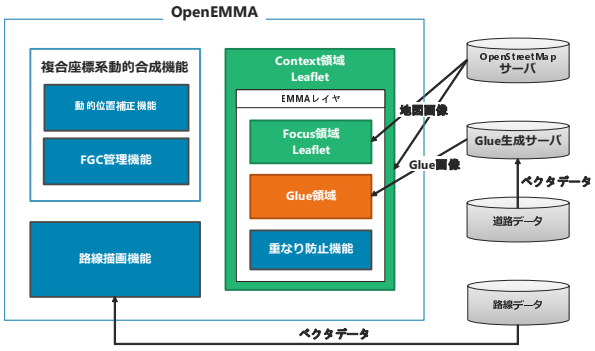

図 2　提案システムの構成図

OpenStreetMap サーバはタイリング用のラスタマップを生成し，Glue 生成サーバは Glue 領域用に道路を間引いて描画したラスタマップ [8] を生成する．F-Leaflet と C-Leaflet は OpenStreetMap サーバからラスタマップを取得して表示する．Glue は Glue 描画機能を用いて，Glue 生成サーバから道路網を総描したラスタマップ取得して表示する．EMMA レイヤはサイズ変更や移動，縮尺変更を行うことができる．複合座標系動的合成機能は常に Focus と Context の状態を監視しており，ユーザーの操作によって Focus や Context の状態が変化したときに動作する．

OpenEMMA システムは，Focus の生成，移動，サイズ変更のための操作インタフェースを提供する．ユーザーは，Web ブラウザを介して，これらの操作機能を利用し，所望の場所と形状の地図情報を表示した Focus+Glue+Context マップを得る．

また，路線描画機能は路線を描画する機能であり，GeoJson 形式のファイルを読み込むことで路線を表示し，出発地と目的地，経由地などに Focus + Glue 領域を作成することができる．

4　OpenEMMA

本章では，OpenEMMA システムの実現法について述べる．

4.1　EMMA レイヤ

EMMA レイヤは Focus と Glue の地図画像を合成して描画するレイヤである．

4.1.1　EMMA レイヤの作成

Focus は Context 上の指定された領域を拡大描画する．このため，Leaflet のイベントハンドラを用いて，以下の手順で，ユーザーがクリックにより指定した地点に Focus と Glue からなる EMMA レイヤを作成する．

手順 1　Leaflet のイベントハンドラを用いて，Context において地図上の点をクリックしたときの緯度経度を取得する．

手順 2　手順 1 で取得した緯度経度を Leaflet の latLngToLayerPoint メソッド [1] を用いて Web ブラウザ上での xy 座標標に変換する．

手順 3　EMMA レイヤの HTML 要素を生成し，その中に Focus と Glue の HTML 要素を円形に生成する．

手順 4　手順 2 で得た Web ブラウザ上の xy 座標が中心となるように Focus と Glue を移動させる．

手順 5　Focus の HTML 要素に Leaflet インスタンスを作成する．

手順 6　Glue サーバが提供するラスタマップを Glue に表示

1：緯度経度が与えられた場合，原点ピクセルを基準として対応するピクセル座標を返すメソッド．

する．

4.1.2 EMMA レイヤの Leaflet レイヤ化

Leaflet [10] は表 2 のように pane と呼ばれるレイヤから成る層構造になっており，マップのタイル画像を表示する tilePane や，マーカを表示する markerPane など，複数のレイヤを重ね合わせて描画している．このとき z-index の値が大きいレイヤが上になる．一般にこのレイヤは同じ座標系持ち，同一 Leaflet 内にあるレイヤは，地図の描画範囲が変わると連動して一緒に動く．

表 2　Leaflet の階層

Pane	Z-index	説明
tilePane	200	マップのタイル画像を表示する層
overlayPane	400	ポリゴンやパスなどを表示する層
shadowPane	500	影を表示する層
markerPane	600	マーカを表示する層
tooltipPane	650	ツールチップを表示する層
popupPane	700	ポップアップを表示する層

EMMA レイヤは場所によって縮尺が異なるため，各領域で座標系が異なる．また，Focus と Glue は一体となって動く必要がある．

そこで，異なる座標系のレイヤを連動させて動作させるためにネスト型の Leaflet を構築する．すなわち，C-Leaflet に EMMA レイヤを配置し，そのサブレイヤとして F-Leaflet と Glue を配置する（表 1）．これによって Focus と Glue は Context の動きに連動して一緒に動くようになる．また Focus と Glue の上にポリゴンやマーカなど，C-Leaflet の他のレイヤを重ねて表示できるようになる（図 3）．

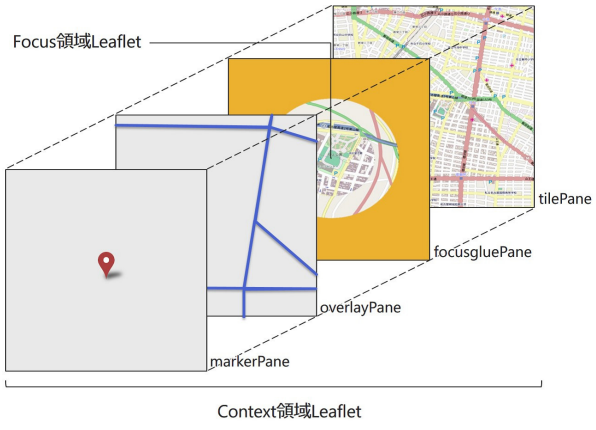

図 3　提案システムでの Leaflet 階層

4.1.3 EMMA レイヤのサイズ変更

ユーザーはドラッグやホイール操作で EMMA レイヤを操作する．本節ではサイズ変更操作について述べ，移動と縮尺変更の操作については 4.2.1 で述べる．

ユーザーは，Focus と Glue の特定の場所に対するドラッグ操作により，この領域のサイズ変更要求を指示する．以後 Focus と Glue を一体として扱う場合，この 2 つの領域を Focus+Glue 領域と呼ぶ．EMMA レイヤは，次の手順で，Focus ＋ Glue 領域の大きさを変化させ，再描画する．

手順 1　Glue 内にマウスポインタがあるときにドラッグをすると，サイズを変更する処理を開始する．ドラッグした際に C-Leaflet が動かないように，C-Leaflet においてオプションの dragging の値を false に設定する．

手順 2　ドラッグ中は常にマウスポインタの座標を取得し，Focus の中心座標からの距離を計算する．

手順 3　この距離を一定時間前の Focus+Glue 領域の半径と比較し，差分を各領域のサイズに適用する．

4.2 複合座標系動的合成機能

4.2.1 FGC 連携機能

Focus と Glue，Context の間に入って各領域を制御する機能である．

EMMA レイヤの移動

EMMA レイヤの移動は，Focus ＋ Glue 領域の座標移動とそれに伴う領域の再描画の 2 つからなる．EMMA レイヤは，ユーザーのドラッグ操作に応じて，以下のように Focus ＋ Glue 領域を移動させる．

ドラッグ開始時　Focus をドラッグした際に C-Leaflet が動かないように，C-Leaflet においてオプションの dragging の値を false に設定する．

ドラッグ中　Focus ＋ Glue 領域を移動させる．

ドラッグ終了時　C-Leaflet においてオプションの dragging の値を true に再設定する．

また，Focus+Glue 領域の座標が変化したときに，Web ブラウザ上での Focus ＋ Glue 領域の中心 xy 座標を Leaflet の layerPointToLatLng メソッド [2] を用いて緯度経度に変換する．得た緯度経度で Focus ＋ Glue 領域を再描画することにより，マウスポインタの位置の変化に，画面上の Focus ＋ Glue 領域を追従させる．

縮尺変化時の位置変更

Context や Focus の縮尺がユーザーの操作によって変化したとき，システムは Focus+Glue 領域を縮尺の変化に合わせて移動させる必要がある．Context と Focus の拡大縮小に対してのイベントリスナーを作成し，縮尺が変わるたびに縮尺変更後のブラウザ上での位置を計算して Focus+Glue 領域を再配置する．

4.2.2 動的位置補正機能

Focus+Glue+Context マップは場所により縮尺が異なり非均一な座標系であるため，提案システムは，各領域ごとに座標系を定め，地図オブジェクトの所在場所に応じて表示位置と形状を計算して描画する．また，Focus の生成，消滅，移動が発生した場合は画面上の地図オブジェクトの座標を再計算する必要がある．このとき，Context の座標系を基準として用いる．Context は均一の縮尺であるので，地図オブジェクトが Context にある場合は所在場所の緯度経度に応じた座標に描画すればよい．地図オブジェクトが Focus ＋ Glue 領域にある場

2：原点ピクセルを基準にしたピクセル座標が与えられた場合，対応する地理座標を返すメソッド．

合は，Context の座標系に変換してから描画する必要がある．Leaflet の地図オブジェクトは地理座標をもとに描画されているため，表示位置や形状を変化させるために，地図オブジェクトの持つ地理座標 (lat, lng) にある点 P を $P' = (lat', lng')$ に変換する．

まず，P が存在する領域を返す関数 $region(P)$ を実現する．

$$region(P) = Focus|Glue|Context|NULL$$

P が存在する領域は，Focus，Glue，Context のいずれかであるか，または画面上に存在しないかである．

次に地理座標 (lat, lng) を，Focus 座標系，Glue 座標系，Context 座標系の位置座標 $(x_F, y_F), (x_G, y_G), (x_C, y_C)$ に変換する次の 3 つの関数を実現する．

$$\begin{aligned} f_F(lat, lng) &= (x_F, y_F)(\text{P が Focus にある場合}) \\ &= NULL(\text{そうでない場合}) \\ f_G(lat, lng) &= (x_G, y_G)(\text{P が Glue にある場合}) \\ &= NULL(\text{そうでない場合}) \\ f_C(lat, lng) &= (x_C, y_C)(\text{P が Context にある場合}) \\ &= NULL(\text{そうでない場合}) \end{aligned}$$

また，Focus 座標系，Glue 座標系，Context 座標系の位置座標 $(x_F, y_F), (x_G, y_G), (x_C, y_C)$ を地理座標 (lat, lng) に変換する次の 3 つの関数を実現する．

$$\begin{aligned} f_F^{-1}(x_F, y_F) &= (lat, lng)(\text{P が Focus にある場合}) \\ &= NULL(\text{そうでない場合}) \\ f_G^{-1}(x_G, y_G) &= (lat, lng)(\text{P が Glue にある場合}) \\ &= NULL(\text{そうでない場合}) \\ f_C^{-1}(x_C, y_C) &= (lat, lng)(\text{P が Context にある場合}) \\ &= NULL(\text{そうでない場合}) \end{aligned}$$

ここで Focus，Glue，Context の位置座標原点を $(x_{FO}, y_{FO}), (x_{GO}, y_{GO}), (x_{CO}, y_{CO})$ とし，$(x_{CO}, y_{CO}) = (0, 0)$ である．

Focus 上の場合

$region(P) = Focus$ のとき，P の Focus での位置座標は $(x_F, y_F) = f_F(lat, lng)$ であるので，Context での位置座標は $(x_{FO} + x_F, y_{FO} + y_F)$ となる．これより変換後の地理座標 $P' = (lat', lng')$ は，

$$P' = (lat', lng') = f_c^{-1}(x_{FO} + x_F, y_{FO} + y_F)$$

で求められる．

Glue 上の場合

Glue は Focus-Glue 間と Glue-Context 間を滑らかに接続するために 3 次ベジエ曲線で表現される再配置関数を用いて地図の変形を行っている（図 4）．3 次ベジエ曲線で表現される再配置関数を $g(lat, lng) = (x_G, y_G)$ と定義すると，Glue での位置座標は $(x_G, y_G) = g(lat, lng)$ であるので，Context での位置座標は $(x_{GO} + x_G, y_{GO} + y_G)$ となる．これより変換後の地理

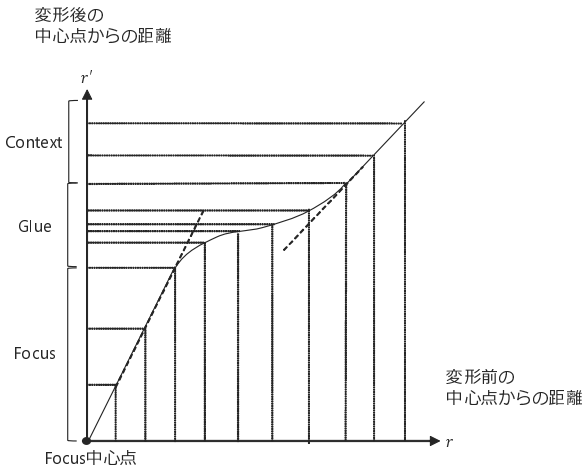

図 4　Focus+Context+Glue マップの再配置関数形状

座標 $P' = (lat', lng')$ は，

$$P' = (lat', lng') = f_c^{-1}(x_{GO} + x_G, y_{GO} + y_G)$$

で求められる．

4.2.3　重なり防止機能

複数の Focus ＋ Glue が重なると地理的な矛盾が生じるため，衝突する際にサイズ変更を行うことで重なりを回避する．EMMA レイヤ A と B に対して以下の手順で図 5 のように処理を行う．

手順 1　A が B に衝突したとき，B は元々の半径の値を保存する

手順 2　B は A に重ならないように半径を小さくする

手順 3　A が B から遠ざかる時，B はこの時 B の元々の半径を超えない範囲で A に重ならないように半径を大きくする

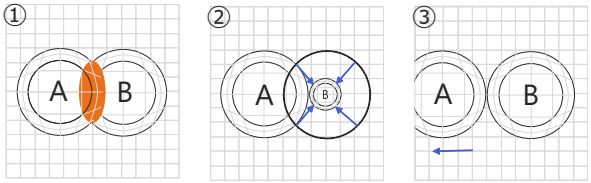

図 5　重なり防止機能

4.3　路線描画機能

路線描画機能は，様々な地理的データ構造をエンコードするための GeoJson 形式と呼ばれるファイルを読みこんで表示する．属性にはポイント（住所や座標），ライン（各種道路や境界線）などが含まれる．バス路線は一つの道路に対して 10 以上の路線が並走しているなど，とても複雑である．そのため，バス路線描画を実現するためには，分かり易いバス路線図の描画が必要である．今回利用する路線データは路線の区間を表す緯度経度の配列と，その区間を走るバス路線の配列を含むものとする．バス路線の配列と区間データから路線を表示するポリラインの太さと長さを求め，ポリラインにオフセットを持たせることのできる Leaflet Polyline Offset [27] を用いて路線が並走する区間を見やすく描画する．

点補填機能

動的位置補正機能は点に対して行うため，路線データなどにおける長い直線ではある点 g_n とその次の点 g_{n+1} の間に中間点がなく，うまく動的位置補正機能が働かないことが考えられる．そこで，以下の手順で g_n と g_{n+1} の間に点を補填する．

手順１　g_n と g_{n+1} の距離を求める

手順２　距離が一定値 d 以上の場合，距離/d 個の点を補填する

5　プロトタイプシステム

4 章で述べた全機能を有する提案システムのプロトタイプを開発した．開発には，Windows10，Visual Studio Community 2015，JavaScript，HTML，CSS を用いた．動作確認には主に Google Chrome を用いた．

5.1　操作コマンド

プロトタイプを起動すると，Context のラスタマップ，および，ツール選択（画面右下）のプルダウンメニューが表示される．メニューからは，Focus 生成，削除の操作コマンドを選択する．Focus 生成を選択すると，Focus+Glue 領域をクリックした場所に作り，描画する．

5.2　Focus+Glue 領域の移動とサイズ変更

Focus+Glue 領域を移動するにはユーザーは Focus または Glue 上にマウスを持っていき，ドラッグする．Focus+Glue 領域のサイズを変更するにはまずユーザーは Focus または Glue の境界線上にマウスを持っていき，ドラッグする．

5.3　路線図表示

システムは GeoJson ファイルを読み込むことで路線図を描画できる．描画された路線図は複合座標系動的合成機能により各領域に合わせた位置に変形しながら描画される（図 6）．

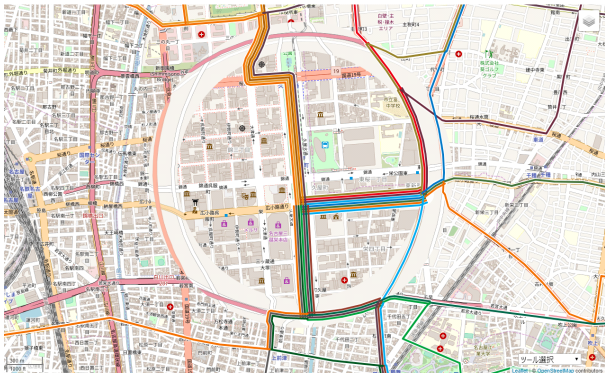

図 6　各領域に応じた路線図の表示

6　評 価 実 験

Windows10，DELL Precision 3420，Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz 3.19GHz，RAM 8.00GB 環境のもと，Google chrome でプロトタイプを動作させて，提案システムの利便性を実験的に評価した．

6.1　実験 1 Emma レイヤーの Focus+Glue ラスタマップ描画機能の有効性評価

Focus の描画に Leaflet を用いる場合と，用いらない場合での描画速度の違いを測定する．

比較対象

（1） 提案手法．Web ブラウザが Leaflet のタイリング機能により Focus と Glue のラスタマップを合成する．提案システムのプロトタイプを使用．

（2） 従来手法．サーバが Focus と Glue のラスタマップを合成する．今回は従来手法の実現のために，中心座標縮尺サイズをパラメータとして地図画像を返すサーバを作成．

比較項目

- Focus の平均描画時間（ms）

測定対象

400 × 400，500 × 500，600 × 600 の 3 つの大きさ（px）の Focus に対して，各 30 回測定する．

6.2　実験 2 動的位置補正機能および路線図描画機能の有効性評価

Focus ＋ Glue+Context 型 Web マップにおける路線図の描画において，クライアント側でベクタデータを変形させ描画する方式（提案手法）と，サーバ側でラスタ画像を生成して表示する方式 [8]（従来手法）の平均描画時間を計測・比較することで提案手法の有効性をはかる．

比較対象

（1） 提案手法．路線図上で Focus+Glue 領域をドラッグで移動させて動的位置補正機能を 1000 回繰り返し作動させ，その平均処理時間 [ms] と平均フレーム数 [fps] を計測する．

（2） 従来手法．Glue 画像を生成するサーバにあらかじめ路線図データを格納しておき，クライアント側のリクエストに応じて Glue 画像上に路線図を表示した画像を生成する機能を持たせる．クライアント側が Glue 画像を Glue サーバにリクエストして，Glue サーバから Glue 画像が返ってくるまでの時間を 1000 回計測し，その平均時間 [ms] をとる．また平均時間を 1000ms で割ることで平均フレーム数 [fps] を算出する

使用するデータ

- 筆者が作成した名古屋の栄から出るバス路線を表すポリライン（総頂点数 6200 個）を使って実験を行う（図 6）．

6.3　実験 3 出発地から目的地までのバスルート確認作業タスクによる有効性評価

現在地から目的地までのバスを使ったルートを 3 種類用意する．提案手法と，従来手法とで 3 種類のルートを確認してもらい，下記の穴埋め問題に対して回答してもらう．学生にはできるだけ早く答えてもらい，回答までにかかった時間と地図の操作回数を計測する．これによりルートを理解するまでの時間や操作の手間などをはかる．

比較対象

（1） 提案手法．目的地付近に Focus ＋ Glue 領域を設置した Focus ＋ Glue+Context 型の Web マップシステムで 3 種類

のルートを確認してもらう．

（2） 従来手法．Focus + Glue 領域を使用しない均一スケールの Web マップシステムで 3 種類のルートを確認してもらう．

使用するデータ

- 現在地から目的地までのバスを使ったルート 3 種類．例を図 7 に示す．

図 7　実験 3 の例 (左：提案手法，右：従来手法)

穴埋め問題

_____バス停から_____に乗り，_____で降りて，____の方角に進み，_____ところに目的地があります．

6.4　結果と考察

実験 1　結果を 8 に示す． 従来手法は表示する場所が決まってからサーバで画像を生成するため，表示までに時間がかかってしまう．それに対し提案手法では，Leaflet のキャッシュ機能により周辺の地図画像をあらかじめ読み込んでおけるため，表示するまでの時間はサイズによらず平均で 1ms を下回った．ユーザーが使う上でレスポンスは非常に大事な項目であり，表示までのラグがほとんどない提案手法は優れているといえる．なお，既存の魚眼マップに関する研究 [1-7][13-16][20] を Web マップサービスとして実現するためには，画面全体をサーバ側で動的に描画する必要がある．よって，Focus+Glue 領域のみサーバ側で動的に描画する必要がある従来手法よりも，さらに遅い応答速度となることが想定できる．

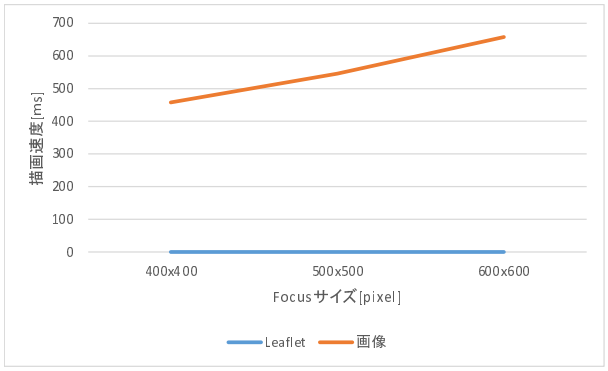

図 8　実験 1 の結果

実験 2　結果を図 9,10 に示す．処理時間に関して，従来手法はクライアント側のリクエストを受け，サーバで道路データを処理して画像を生成しクライアント側に送るため，生成する画像のサイズが大きいほど表示までに時間がかかってしまい，例えば直径が 400px の場合では 142.5ms という結果になった．それに対し提案手法では，処理時間が Glue のサイズによらず，路線データの頂点数によって決まるため，表示するまでの時間はサイズによらず 50.1ms という結果になった．サイズによるが提案手法は従来手法の 21〜65%の時間で処理できることがわかる．またフレームレートに関しても，提案手法が従来手法よりも優れている結果となった．サイズによるが提案手法は従来手法よりもフレームレートが 23〜200%高かった．ユーザーが使う上でレスポンスは非常に大事な項目であり，表示までのラグがほとんどない提案手法は優れているといえる．

しかし，路線データ（マーカやポリラインなど）が増えるほど，描画する図形が多くなるためフレームレートが下がることが予想される．これについて路線データが多い場合には従来手法を，そうでない場合には提案手法を使うなどうまく組み合わせることで解決できると考える．

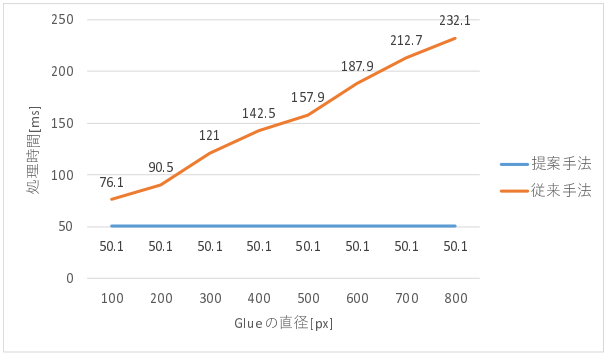

図 9　実験 2 の結果（処理時間）

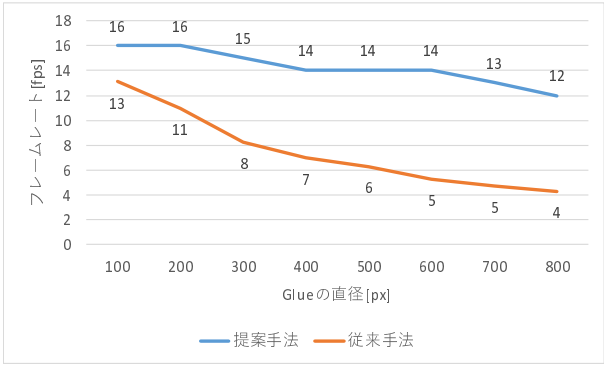

図 10　実験 2 の結果（fps）

実験 3　図 11 を見ると，提案手法が従来手法よりも時間と操作回数ともに優れている結果となった．今回の実験では提案手法は従来手法と比べて時間は平均 83%で，回数は平均 23%という結果になった．目的地最寄りのバス停から目的地までの道のりを，少ない操作回数で素早く理解することができることは道案内において非常に重要な項目であるといえる．また，近年では実際のバス停の時刻表が液晶で表示されているところもあり，これに提案システムを導入すればバス停の利便性が高まると考えられる．

	提案手法		従来手法	
	時間[s]	回数	時間[s]	回数
ルート1	86.80	0.20	105.20	6.40
ルート2	81.80	1.60	103.00	4.20
ルート3	82.20	1.80	91.40	4.80
平均	83.60	1.20	99.87	5.13

図 11　実験 3 の結果

7 おわりに

本論文では，構造化した 2 つの Leaflet を有し，Focus・Glue・Context の異なる座標系のマルチレイヤ構造を持つ Web マップシステム OpenEMMA を提案した．さらに，各領域間を連携させて整合性を保つ機能，および，地図オブジェクトの描画・移動操作を監視して動的に位置補正する機能を提案し，これにより，可変スケールマップをベースにした位置情報サービスの実現を容易にした．また，路線描画機能の導入により OpenEMMA を拡張し，名古屋市バス路線の描画に適用して，有効性を実験的に評価した．実験を通して，Focus ＋ Glue+Context 型のバス路線図の有効性を示すことができたが，処理するデータ量が多くなる場合に対策が求められる結果となった．今後は Leaflet のプラグインとして動作可能な BSD ライセンスでの公開を目指している．

8 謝　　辞

本研究は JSPS 科研費 26330136，25700009，および，総務省 SCOPE の助成を受けたものです．この場を借りて，感謝の意を表します．

文　　献

[1] George W. Furnas, ”Generalized Fisheye Views”, Proceedings of the CHI’86, pp.16-23, 1986.
[2] L. Harrie, L. T. Sarjakoski, L. Lehto, ”A Mapping Function for Variable-Scale Maps in Small-Display Cartography”, JGE, Vol.4, No.2, pp.111-123, 2002.
[3] Craig, H. Chen, and F. Houssen. ”A Task Based Evaluation of Fisheye Maps for Mobile Navigation.” 1st Conference on Emerging Topics in Interactive Systems, Suzhou, China. 2016.
[4] N. Takahashi, ”An Elastic Map System with Cognitive Map-Based Operations”, In- ternational Perspectives on Maps and Internet Vol.1, M. P. Peterson and J. Liu (Eds), Lecture Notes in Geoinformation and Cartography, Springer-Verlag, Feb. 2008.
[5] Daisuke Yamamoto, Shotaro Ozeki, Naohisa Takahashi, Focus+Glue+Context: An Improved Fisheye Approach for Web Map Services, Proceedings of the ACM SIGSPATIAL GIS 2009, Seattle, Washington, pp.101-110, 2009.
[6] 加藤 史也，山本 大介，高橋 直久，任意形状 Focus 生成機能を有する Focus+Glue+Context マップシステムの実現，電子情報通信学会技術研究報告．DE，データ工学，Vol.112，No.346，pp.119-124，2012.
[7] H. Mizutani，D. Yamamoto，and N. Takahashi，”A Fusion of Multiple Focuses on a Focus+Glue+Context Map”，Proceedings of the International Conference on Intelligent Interactive Multimedia Systems and Services (IIMSS 2012)，pp.11-21，2012.
[8] Daisuke Yamamoto，Masaki Murase and Naohisa Takahashi，Fisheye Map Using Stroke-based Generalization for Web Map Services，IEICE Transactions on Information and System，Vol.E101-D，No.1，pp.171-180，2018.
[9] OpenStreetMap Japan，https://openstreetmap.jp/
[10] Leaflet - a JavaScript library for interactive maps，http://leafletjs.com/
[11] OpenLayers，https://openlayers.org/
[12] Rich Gibson and Schuyler Erle，”Chapter 3. Mashing Up Google Maps”，Google Maps Hacks，O’Reilly，2006.
[13] K. Lynch. The image of the city. MIT Press, 1960.
[14] P. Gould and R. White. Mental Maps. Penguin Books Ltd, Harmondsworth, Middlesex, England, 1974.
[15] B. Jenny and L. Hurni, ”Studying cartographic heritage: Analysis and visualization of geometric distortions,” Computers & Graphics, vol. 35, pp. 402-411, 2011.
[16] J. Brainerd and A. Pang, ”Interactive map projections and distortion,” Computers & geosciences, vol. 27, pp. 299-314, 2001.
[17] Manojit Sarkar, Marc H. Brown, ”Grapahical Fisheye views of graphs”, Proceedings of the CHI ’92, pp.83-91, 1992.
[18] Manojit Sarkar, Scott S. Snibbe, Oren J. Tversky, Steven P. Reiss, ”Stretching the Rubber Sheet: A Metaphor for Viewing Large Layouts on Small Screens”, Proceedings of the UIST’93, pp.81-91, 1993.
[19] Ilya Zaslavsky. ”Online cartography with XML.” Maps and the Internet. pp.171-196, 2003.
[20] Y. S. Wang and M. T. Chi, ”Focus+Context Metro Maps,” in IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 12, pp. 2528-2535, Dec. 2011.
[21] Pikorec M., Sluban B., muc T., MultiNets: Web-Based Multilayer Network Visualization, Proceddings of the ECML PKDD 2015: Machine Learning and Knowledge Discovery in Databases, pp 298-302, 2015.
[22] 岡恵, 福田豊. ”招待講演 航空交通のオープンデータとその活用”, 電子情報通信学会技術研究報告, Vol.117, No.301, pp.95-100, 2017.
[23] 早川浩平, 早川知道, 伊藤孝行. ”OpenStreetMap を活用したイベント情報共有支援システムの試作”, 情報処理学会研究報告, 知能システム (ICS), Vol.2013-ICS-172, No.4, pp.1-6, 2013.
[24] 河村郁江, 伊藤 孝行, 郷土食による地域理解支援システム「もちマップ」の試作, 人工知能学会研究会資料, SIG-SWO-041-08, pp.1489-1490, 2018.
[25] 大向一輝. ”オープンデータ活用: 1. オープンデータと Linked Open Data.” 情報処理 54.12, pp.1204-1210, 2013.
[26] 竹内健祐, 山本大介, 高橋直久. ”Leaflet と OpenStreetMap を用いた Focus+Glue+Context マップインタフェースの開発と評価”, 情報処理学会論文誌, Vol.59, No.12, pp.2221-2231, 2018.
[27] Leaflet Polyline Offset, https://github.com/bbecquet/Leaflet.PolylineOffset
[28] 吉村 元秀, 松田 佳奈実, 地図を用いたバス路線および時刻表検索システムの作成, 長崎県立大学研究紀要 Vol.15,pp.225-229, 2014.
[29] J. Stott, P. Rodgers, J. C. Martinez-Ovando and S. G. Walker, Automatic Metro Map Layout Using Multicriteria Optimization, IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 1, pp. 101-114, Jan. 2011.
[30] 竹内健祐，山本大介，高橋直久：Leaflet と OpenStreetMap を用いた Focus+Glue+Context マップインタフェースの開発と評価，情報処理学会論文誌，Vol.59，No.12，pp.22212231 (2018).