列車の到着時間案内アプリの実用性向上

井上 晴稀[1]　梶 克彦[2]

概要：日本の鉄道は複雑に発達しており，定時制のある運行を行っている．公共交通機関を利用する乗客が円滑に移動するためには，乗客に対して列車に関する情報を提供する必要がある．以前，我々はスマートフォンを使い，GPS を利用した位置情報から乗客の乗っている列車を自動で推定し，到着時間を乗客に提示するアプリケーションを提案した．しかし従来の推定処理は，乗客の移動履歴が長くなってしまうと，列車の推定にかかる処理時間が増大してしまうという問題点を抱えている．そこで，列車の推定にかかる時間を短縮させるため，長大な乗客の移動履歴を構成する位置情報の数を間引いて推定処理を行うように変更する．また，乗客の乗っている列車の候補順位を指標とする精度を維持するために，路線推定処理と列車推定処理においてハウスドルフ距離を用いて類似度を出す処理に変更し実装する．他にも，従来システムでは候補順位が表れていない部分が発生する問題があったため，方向推定処理内の処理も変更する．我々は従来システムと比較して，推定にかかる時間や精度がどのように変化したか評価実験を行った．推定処理時間を比較する評価実験では，今回実装したシステムの推定処理において，300 秒の移動履歴が渡された場合の列車の推定にかかる時間を確認した．その結果，従来システムにおける推定処理の平均時間が 47.25 秒であったのに対し，今回実装したシステムでは 3.17 秒であると確認し，従来システムより推定処理時間が短縮した．また今回実装したシステムにおいて，移動履歴の長さに伴う乗客の乗っている列車の候補順位の変動を比較する評価実験を行った．今回実装したシステムでは，候補順位が表れていない部分は解消し，実験データ 20 件中 17 件において候補順位が 5 分間 1 位を保ち続けているのを確認した．しかし，20 件中 3 件の実験データにおいて従来システムより候補順位が下がっているのを確認した．

Improving the Practicality of Applications that Display Train Arrival Time to Passengers

HARUKI INOUE[1]　KATSUHIKO KAJI[2]

1. はじめに

日本の鉄道は，海外と比較して複雑に発達しており，定時性のある運行をしている．日本においては，東京を中心とする首都圏や大阪を中心とする関西圏を中心に，複数の鉄道会社が発達している．日本では大都市圏以外にも，地方都市同士を結ぶ鉄道路線網も発達している．このように大きく鉄道路線網が発達している環境下で，日本は海外に比べて遅れが少ない定時制のある運行を行っている．日本の鉄道は，15 秒間隔で運行スケジュールが決められており，それに従い運転を行っている．また，一部の路線では列車の運行を自動化し，さらに運行の定時制を高めている路線も存在する．

このように複雑に発達した鉄道網において，乗客が円滑に移動できるようにするには，列車に関する様々な情報を提供する必要がある．日本の鉄道では，列車内において次の停車駅や停車駅までの簡単な所要時間，停車駅における乗換案内などを提供している．このように列車の車内において情報を提供する場合，都市圏や列車によって乗客に提供される情報の量に差が出てしまう．他にも，列車に乗っている乗客がスマートフォンアプリケーションから自分の乗る列車の詳細な停車駅の情報を確認する方法もある．

これまで我々は，乗客のスマートフォンに搭載している GPS を用いて取得した移動履歴を用いて，乗客に到着時間を提示するアプリケーションを提案している [1]．我々が提案したアプリケーションの概要を図 1 に示す．乗換案

[1] 愛知工業大学大学院 経営情報科学研究科
[2] 愛知工業大学 情報科学部情報科学科

図 1　アプリケーションの概要

内アプリケーションを利用して自分の列車に関する情報を検索する場合，乗客が出発する駅の情報や到着する駅の情報を自ら入力する必要があり，手間がかかってしまう．そこで，我々は乗客が列車に乗車してスマートフォンアプリケーションを開くと，アプリケーションが自動的に乗客の乗っている列車を推定して到着時間を乗客に提示するアプリケーションを提案した．

以前に我々が提案したシステムは，乗客の移動履歴が長くなってしまうほど，列車を推定する処理時間が長大になってしまう問題がある [1]．従来システムは，スマートフォンの位置情報から乗客の移動履歴を生成し，移動履歴を別のサーバに送信し列車を推定する処理を行う．しかし，移動履歴が長くなってしまうと，それに伴い処理する量が増えていき，列車を推定するまでにかかる時間が長くなってしまう問題を抱えている．

この研究では，我々が提案したアプリケーションの実用性を高めるため，候補順位の精度を維持しながら乗客の列車を推定する処理を変更する．我々は，移動履歴が長くなってしまうと処理時間が増大してしまうという点に着目し，列車推定処理の変更を行い実装する．また，処理時間を短くした場合でも，乗客の乗っている列車の候補順位を指標とした精度を維持するために列車推定処理を変更する．

2. 関連研究

スマートフォンに内蔵されているセンサを使い，列車の位置を推定する研究は多く提案されている．スマートフォン端末の気圧センサを使って地下鉄の走行位置を推定する手法 [2] が提案されている．この手法は，予め事前に観測された気圧の変化や標高の変化等を記録しておき，記録した結果とスマートフォンで観測したデータと比較して走行位置推定を行っている．他にも磁気センサを使った地下鉄の発車および停車検知を行っている研究 [3] や，磁気センサを使って記録したデータを用いて地下鉄の列車の停車を検知し，停車した時間を地下鉄の運行スケジュールに反映するシステムの構築を行っている研究 [4]，スマートフォンに内蔵されている GPS を用いた到着時間予測システムの構築に関する研究 [5] が行われている．本研究で開発するアプリケーションにおいては，乗客の現在地情報は比較的容易に取得できる GPS を用いて取得するが，地下鉄区間においては GPS による現在地取得が難しい．そのため，スマートフォンのあらゆるセンサを用いて現在地情報を取得する手法も考慮しなければならない．

公共交通機関の一つであるバスに対して所要時間を予測する研究や手法が様々な手法で提案されている．綿貫ら [6] は，実際の運行データを基に低ランク双線形ガンマ回帰モデルを使用して翌日以降の路線バスの所要時間推定を行っている．この他にも，人工ニューラルネットワークモデルを用いた手法 [7] や，k-近傍回帰アルゴリズムを用いた手法 [8]，カルマンフィルタを用いた手法 [9] で路線バスの所要時間推定を行っている研究が存在する．これらの研究は過去の運行データを機械学習させ所要時間を推定しているが，本研究ではこのような機械学習を用いず乗客の乗っている列車を推定し，乗客に到着時間情報を提示する．

鉄道においても，到着時間を予測する研究や手法が提案されている．Pongnumkul ら [10] は，タイの国営鉄道で提供されている列車遅延表示サービスを改善するため，過去の旅行時間を用いた列車の到着時間予測を改善する手法を様々なアルゴリズムで構築している．過去の到着時間履歴を基に移動平均手法を用いて到着時間を予測するモデルと，同じく過去の到着時間履歴を基に k-近傍回帰手法を用いて到着時間を予測するモデルを実装し，列車の到着時間を予測する際に発生した誤差が，従来のアルゴリズムに比べて縮小しているのを確認している．また，Liu ら [11] は鉄道輸送における逆伝搬ニューラルネットワーク，ウェーブレットニューラルネットワーク，遺伝的アルゴリズムを用いた到着時間予測モデルについて提案している．これらの研究も，過去の運行データから機械学習させ到着時間を予測する手法を用いている．本研究では，過去の運行データではなく時刻表データから乗客の乗車している列車を推定する．

本研究で提案しているアプリケーションに類似しているシステムとして，駅.locky[12] が挙げられる．このサービスはスマートフォンアプリケーションを起動しただけで，利用者の最寄り駅の発車時刻までの残り時間が提示されるサービスである．アプリケーションを起動しただけで情報が提示される点や，路線を気にすることなく最寄り駅の発車時刻までの情報が提示される点が，本研究で開発するアプリケーションと類似している．この関連研究は，列車に乗る前にスマートフォンアプリケーションを起動する利用想定であるが，本研究は，列車に乗っている最中にスマートフォンアプリケーションを起動する利用想定をしている．

このアプリケーションは，公共交通機関の種類や会社を考慮せずサービスを提供する考え方である MaaS(Mobility

as a Service) という考え方を取り入れているが，MaaS に関する研究も行われている．日本においては，タクシーやカーシェアリングサービスの実情や公共交通機関の仕組みを考慮した MaaS モデルを構築し，構築したモデルを基にしたシステムアーキテクチャの検討を行っている研究 [13] が行われている．また，全ての移動手段に対して利用者が求める要件をバランスよく満たすルートを提案するアルゴリズムを構築している研究 [14] が存在する．この研究は，移動に関して利用者が求めるコストと待ち時間と快適さの 3 つの要求をバランスよく満たしてくれるルートを提示するアルゴリズムを構築し，検証実験を行っている．

3. 乗客に到着時間を提示するまでの流れ

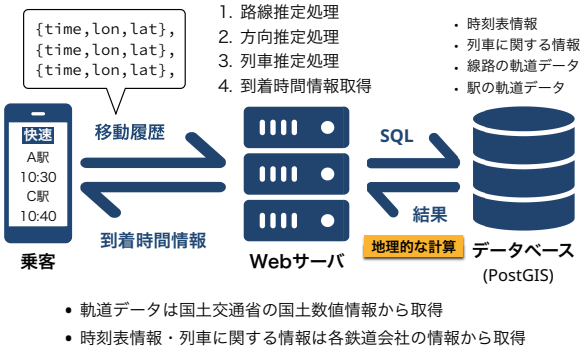

図 2　列車を推定する処理の流れ

到着時間を案内するアプリケーションは，乗客のスマートフォンの GPS から取得した移動履歴を用いて乗客の乗っている列車を推定する．乗客の移動履歴を生成して乗客の乗っている列車を推定するまでの処理の流れを図 2 に示す．乗客がアプリケーションを開くと，スマートフォンの GPS を用いて 1 秒毎に位置情報を取得する．アプリケーションは 1 秒毎に取得した位置情報と取得時刻の情報から，時系列的な位置情報を表現した移動履歴を GeoJSON 形式で生成する．アプリケーションで生成した移動履歴は，列車推定を処理する Web サーバに送信する．

乗客の移動履歴は Web サーバに送信された後，Web サーバは路線推定処理や方向推定処理をして，乗客の乗っている列車を推定する．Web サーバはアプリケーションから送られてきた乗客の移動履歴を用いて，乗客の乗っている列車を推定する．乗客の乗っている列車を推定するために，乗客の乗っている路線を推定する路線推定処理，乗客の乗っている列車の進行方向を推定する方向推定処理を行った後，それらの推定結果を基に乗客の乗っている列車を推定する．

Web サーバは，路線の地理的な軌道データや時刻表データなどを格納しているデータベースと連携しながら列車推定処理を行う．データベースには，列車やバス等の公共交通機関情報に関する情報を格納している．公共交通機関に関する情報は，Google 社が提唱しているオープンフォーマット形式である GTFS 形式に則り格納している．これとは別に，データベース上で地理的な情報を扱える PostGIS を用いて，路線の地理的な軌道情報も格納している．時刻表や列車の情報といった公共交通機関に関する情報は各鉄道会社の情報から，地理的な路線の軌道情報は国土交通省の国土数値情報から，それぞれ参照し格納している．

路線推定処理は，乗客の移動履歴と路線の線形データとの類似度を算出して推定する．路線推定処理は，路線の軌道上に対して許容幅を設定し，乗客の位置情報が許容幅内にいくつ存在するか割合を算出する．その後，算出した割合を尤度として設定し，最も尤度が高い路線を乗客が乗っている路線として推定する．また，並走している路線を考慮するため，最も高かった尤度とそれぞれの路線の尤度との差を算出し，しきい値以内の場合は乗客の乗っている路線の候補として挙げる処理を行う．

方向推定処理は，路線推定処理の結果と乗客の移動履歴を用いて，路線の軌道に対する進行度の大きさを比較して進行方向を推定する．Web サーバは，乗客の位置情報が路線の軌道に対してどのくらい進行しているのかを示す進行度合を，乗客の移動履歴に対して算出する．進行度合の算出は，路線推定処理において候補として挙げた路線に対して行う．時系列で示された進行度合を前半と後半に 2 分割し，それぞれ進行度合の平均を算出する．その後，それぞれの進行度合の平均の大きさを比較し，乗客の乗っている列車の進行方向を推定する．

路線推定処理と列車方向指定処理から算出した結果を基に，乗客の乗っている列車を推定する処理を行う．Web サーバは，データベースに格納されている時刻表データと進行方向の算出結果を基に，走行している列車の絞り込み処理を行い，列車の走行位置を推定する．列車の推定走行位置は，乗客の移動履歴と同じ時系列の長さで算出する．その後，Web サーバは乗客の移動履歴と列車の推定走行位置がどれくらい離れているかを示す距離を算出する．Web サーバは，それぞれの列車で算出した時系列な距離に対して RMS(Root Mean Square) を算出し，類似度を算出する．算出した類似度を列車の尤度として定め，Web サーバは最も尤度が高かった列車を乗客の乗っている列車として推定する．

列車推定処理内で算出する列車の推定走行位置は，列車の加速と減速を考慮した列車の走行モデルを用いて算出する．我々が提案する列車の走行モデルを図 3 に示す．以前の研究において，列車の動きに着目した列車の走行モデルを提案した．具体的には，発車する駅からしばらく加速し，加速が終わると一定の速度を保ち走行し，到着する駅に近づくと減速し到着する駅に停車するという列車の動きを考慮した走行モデルである．それに加え，この走行モデルは

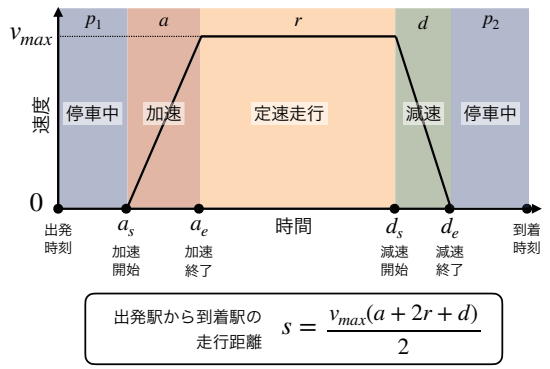

図 3　列車の走行モデル

発車する時刻から発車するまでにかかる停車時間と，到着してから到着時間にかかる停車時間も考慮している．この我々が提案した走行モデルを使い，Web サーバはそれぞれの列車に対して推定走行位置を算出する．

4. 実用性を高めるための変更

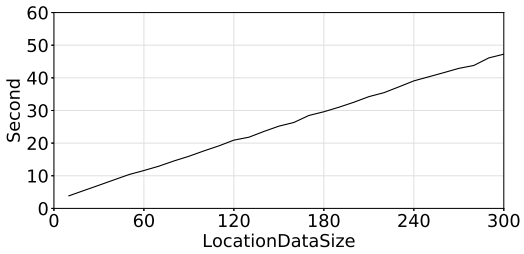

図 4　従来システムの処理時間の平均の推移

従来システムは，乗客の移動履歴が長くなってしまうと，乗客の乗っている列車を推定する処理にかかる時間が長くなってしまうという問題点を抱えている．従来システムにおける処理時間の時系列グラフを図 4 に示す．この処理時間の計測は，Docker を用いてコンピュータ内に推定処理システムとデータベースの仮想コンテナを立ち上げて計測した．以下に Docker の設定環境を記す．処理時間のグラフから，移動履歴の長さが 60 秒のとき 11.61 秒，300 秒のとき 47.25 秒の処理時間がかかると確認した．この処理時間の長さは，スマートフォンアプリケーションを利用する上では実用性に欠ける処理時間である．

- CPU:2 コア (Intel Core i5 3.0GHz)
- メモリ:4GB (2667 MHz DDR4)
- スワップ容量:1GB

そこで，候補順位を指標とした精度を保ちつつ推定にかかる処理時間を短くし，アプリケーションの実用性を向上させるため推定処理を変更する．従来システムから処理時間を短くするため，乗客から送られた移動履歴が長大になっても処理時間が短くなるようにアルゴリズムを変更し実装する．また，列車を推定する時間が短くなった場合，乗客の乗っている列車の候補順位が下がってしまう恐れがある．そこで，候補順位を指標とした精度を保つため，類似度の算出するアルゴリズムを変更し実装する．

4.1　乗客の移動履歴を間引いた列車推定処理

移動履歴が長くなるにつれて列車推定処理時間が長くなっているため，移動履歴を短くした上で列車を推定する処理をする必要がある．従来システムは，乗客の移動履歴を構成する位置情報を個々に取り出し，列車の推定位置を算出している．例えば，乗客の持っているスマートフォンから 1 秒毎に位置情報を取得し，180 秒分の移動履歴を生成して列車推定処理を行った場合を考える．このとき，列車推定処理において 180 個の地理的な点データを全て使い，列車の走行位置を推定したり類似度を出したりする処理を行っている．そのため，移動履歴が長くなってしまうほど，列車推定処理にかかる時間が増大していく．

移動履歴の長さに伴う処理時間の増大を防ぐため，乗客の移動履歴を構成している複数の地理的な点データを指定した個数に間引いた後，列車推定処理を行う．従来システムでは，移動履歴を構成する地理的なポイントを全て使って列車推定処理を行っている．我々は移動履歴を構成する地理的ポイントを，我々が設定した個数分に間引いた後，列車推定処理を行う処理を実装する．例えば，180 秒分の移動履歴が Web サーバに送られた場合を考える．最初に Web サーバは，送られてきた移動履歴を間引く処理を行う．間引いた後の個数を 10 個と設定した場合，180 個の地理的ポイントで構成された移動履歴は，均等な間隔で 10 個に間引いた時系列的な地理的ポイントに再構成する．その後，Web サーバは間引いた後の移動履歴を用いて列車推定処理を行う．

Web サーバが乗客の移動履歴を設定した個数分に間引くため，乗客の乗っている列車を推定する処理の計算量は移動履歴の長さに依存しなくなる．アプリケーション側から移動履歴がどのような長さで Web サーバに送信されても，移動履歴は我々が設定した個数分に間引いた後に列車推定処理を行う．例えば間引いた後の個数を 10 個と設定した場合，Web サーバに 180 秒分の移動履歴が送られてきたときや，300 秒分の移動履歴が送られてきたときにおいても，移動履歴を構成する地理ポイントを 10 個に間引いた後に列車推定処理を行う．そのため，乗客の移動履歴の長さが非常に長大になっても，列車推定処理内で行う計算量は一定になり，列車推定処理にかかる時間が移動履歴に依存しなくなる．

4.2　推定にかかる時間の短縮や従来システムの問題点を解消するための他の変更点

乗客の移動履歴は，GeoJSON 形式で生成しているため，JSON と親和性が高いフレームワークに変更して列車の推

定処理を行う．従来システムでは，Ruby on Rails を利用して列車推定処理を行い，推定した結果をアプリケーション側に返している．しかし，移動履歴は GeoJSON 形式で扱っているため，受け取った移動履歴は Ruby on Rails で扱える変数に変換しなければならない．そのため，GeoJSON 形式と親和性が高い Node.js の Express を用いて列車推定処理の実装を行う．

また，列車推定処理においてデータベースによる問い合わせを極力少なくするように列車を推定するアルゴリズムを改善する．従来システムは，列車推定処理において大量のデータベースの問い合わせをしている．具体的には，乗客の移動履歴を路線における進行度合に変換する処理や，推定した列車の推定走行位置を路線の進行度合に変換する処理においてデータベースの問い合わせが多く行われており，処理時間の増大に影響を与えている．このような問題点を解消するため，列車推定処理においてデータベースの問い合わせを伴う処理のアルゴリズムを改善し実装する．

他にも，駅に停車しているときに乗客がアプリケーションを起動すると，異なる列車が表示される問題点があったため，方向推定処理に改善を施す．従来システムでは，列車が停まっているときに乗客がアプリケーションを起動すると，方向が正しく推定されず異なる列車が乗客に対して提示されてしまう問題点があった．これは，駅に停車しているとき，位置情報が進行方向とは逆の方向に少しずつ動いてしまい，列車の動いている方向が本来の列車とは逆方向に推定してしまうのが原因である．このような問題点を解消するため，乗客の移動履歴の進行距離を算出し，設定したしきい値以下の場合は方向推定をせず，対象路線の全ての列車を対象に列車推定処理を行う処理に変更する．

4.3 類似度の算出方法の変更

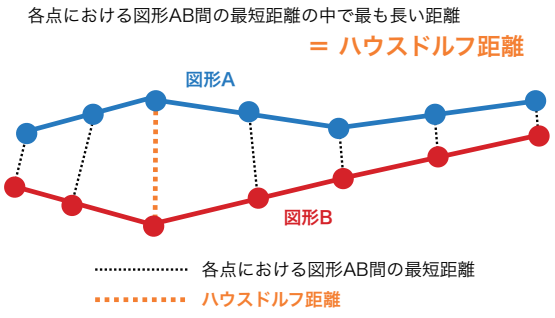

図 5　ハウスドルフ距離

乗客の移動履歴を間引いた後に列車推定処理を行うと，候補順位を指標とした精度が下がる可能性がある．Web サーバは，アプリケーションから送られた移動履歴を構成する地理的ポイントを，我々が指定した個数分に間引いた後に列車推定処理を行う．Web サーバが移動履歴を間引いた後に処理を行うと，列車推定処理に使用する地理的ポイントの数が減るため，乗客の乗っている列車の候補順位を指標とした精度が下がる恐れがある．

他にも，従来システムでは，移動履歴の長さにつれて候補順位が向上していかない問題点も抱えている．移動履歴が長くなると列車推定処理に使用する地理的なポイント数が増えるため尤度が改善され，候補順位を指標とした精度が上がっていくと予測していた．しかし，実際に移動履歴の長さを増やしていき候補順位の変動を確認した結果，移動履歴の長さにつれて候補順位が向上しない実験データがあると確認した．

これらの諸問題を解決するために，路線を推定する処理や列車を推定する処理においてハウスドルフ距離を用いた類似度の計算手法に変更する．ハウスドルフ距離 [15] は，2 つの図形間における距離を計測する手法である．ハウスドルフ距離の概要を図 5 に示す．例えば，図形 A と図形 B があると仮定した場合，図形 A と図形 B を構成する各ポイント間の最小距離をそれぞれ求め，その中から最も距離が大きい値をハウスドルフ距離として算出する．ハウスドルフ距離は，2 つの地理的なジオメトリ間においての類似度の指標となる．このハウスドルフ距離は，PostGIS で地理的なジオメトリにおけるハウスドルフ距離を算出できる．図形 A と図形 B のハウスドルフ距離を $h(A,B)$，図形 A と図形 B を構成する各ポイントを $A=(a_1,a_2,...,a_n), B=(b_1,b_2,...,b_m)$，図形 A と図形 B を構成する各ポイント間の距離を $d(a,b)$ としたとき，ハウスドルフ距離は以下のように表す．

$$h(A,B)=max_{a\in A}\{min_{b\in B}\{d(a,b)\}\}$$

路線推定処理では乗客の移動履歴と路線の線形データ，列車推定処理では乗客の移動履歴と時刻表から算出した列車の推定走行履歴とのハウスドルフ距離を算出する．従来システムでは，路線推定処理において乗客の移動履歴を構成する地理的なポイントが，路線の軌道データに設定した許容幅内にどれだけ入っているかの確率を計算し，その値が大きい値を乗客の乗っている路線として推定している．今回実装するシステムでは，乗客の移動履歴の線形データと路線の軌道を示す線形データとのハウスドルフ距離を算出し，その距離の値が一番小さい路線を乗客の乗っている路線と推定する処理を実装する．また，従来システムの列車推定処理は，列車の推定走行位置と乗客の移動履歴に対して時系列的に距離を算出し，それぞれの列車の時系列的な距離に対して RMS を出し，その値が一番小さい値を乗客の乗っている列車と推定している．今回実装するシステムでは，列車の推定走行位置の線形データと，乗客の移動履歴とのハウスドルフ距離を算出し，その距離が一番小さい値列車を乗客の乗っている列車として推定する．

5. 評価実験

従来システムとの処理時間や候補順位を指標とした精度を比較するために，それぞれについて評価実験を行う．処理時間を比較する評価実験は，従来システムにおいての列車推定にかかる時間と，今回実装したシステムにおいての列車推定にかかる時間を比較し，処理時間がどの程度短縮したのか確認する．候補順位を指標とした精度を比較する評価実験は，それぞれにシステムにおいて普通列車や優等列車を含めた複数の列車において候補順位の変動を確認する．また，それぞれのシステムにおいて普通列車が優等列車に追い越される場合における候補順位の変動も確認する．この評価実験では，実際に複数の路線に乗車して収集した長大な移動履歴のログデータの中から，無作為に一部分を抽出する．その後，列車推定処理をシミュレーションして推定の処理時間や候補順位を算出する．この評価実験内の推定処理では，移動履歴の間引いた後の個数は 10 個，方向推定処理において停車中判定に必要な移動履歴の走行距離のしきい値は 56m に設定して評価実験を行う．また，この評価実験では，Docker を用いてコンピュータ内に列車推定システムとデータベースの仮想コンテナを立ち上げ評価実験を行う．評価実験時における Docker の設定環境は 4 章で記されている設定環境と同じである．

5.1 処理時間を比較する評価実験

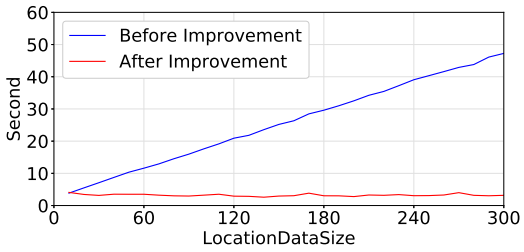

図 6　処理時間の平均の比較

従来システムと今回実装したシステムで処理時間の比較を行うための評価実験を行った．従来システムと今回実装したシステムにおいて，長大な移動履歴から一部分を抽出した移動履歴を 10 秒毎に伸ばしていき，列車の推定にかかる処理時間を計測する．それぞれのシステムにおいて複数の抽出した移動履歴を用いて推定処理を行い，最終的に 10 秒毎の処理時間の平均を算出する．

この評価実験においては，以下に記している移動履歴を用いて，従来システムと今回実装したシステムの推定処理時間を計測する．この評価実験では下記の路線で収集した長大な移動履歴の中から，部分的に 5 分間の移動履歴を抽出して推定処理を行う．今回は，各路線に対して抽出した移動履歴を上り線 5 個，下り線 5 個，合わせて 10 個用意する．よって，20 個の移動履歴を用いて推定処理の処理時間を計測する．

- 近鉄大阪線　名張駅から大阪上本町駅　上下線
- 名鉄名古屋本線　豊橋駅から名鉄名古屋駅　上下線

評価実験の結果から，今回実装したシステムは従来システムより処理時間が短縮したと確認した．従来システムと今回実装したシステムにおける処理時間の平均の推移を図 6 に示す．従来システムでは，移動履歴が長くなるにつれて処理時間の平均が大きくなっているが，今回実装したシステムでは，移動履歴が長大になっても処理時間の平均が大きくならず推移している．具体的には，移動履歴の長さが 60 秒の場合においての処理時間の平均は，従来システムが 11.61 秒であるのに対し，今回実装したシステムが 3.50 秒であった．また，移動履歴の長さが 300 秒の場合においての処理時間の平均は，従来システムが 47.25 秒であるのに対し，今回実装したシステムが 3.17 秒であった．この結果から，我々は今回実装したシステムでは移動履歴が長大になっても処理時間は短縮され，推定処理の実用性が向上したと考える．

5.2 候補順位の推移を比較する評価実験

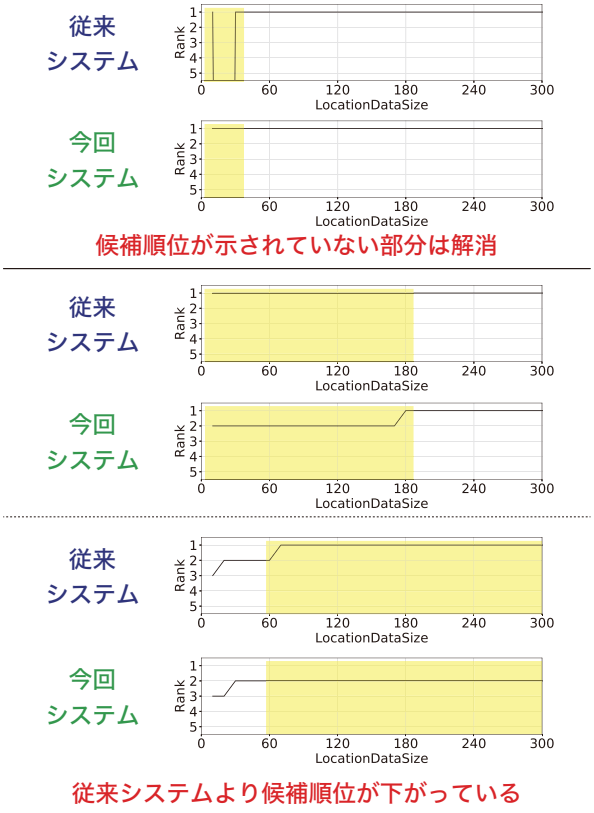

図 7　従来システムと今回実装したシステムの候補順位の比較 (一部)

従来システムと今回実装したシステムで候補順位を指標とした精度を比較するために評価実験を行った．この評価

実験は，5.1 節の評価実験と同様に，従来システムと今回実装したシステムにおいて，長大な移動履歴から一部分を抽出した移動履歴を 10 秒ずつ伸ばしていき，乗客の乗っている列車の候補順位を比較する．それぞれのシステムにおいて複数の抽出した移動履歴を用いて推定処理を実行させ，移動履歴毎に候補順位の時系列的な推移を確認する．

この評価実験においては，5.1 節と同様の移動履歴を用いて，従来システムと今回実装したシステムの候補順位を比較する．この評価実験においても 5.1 節で記されている路線で収集した長大な移動履歴の中から，部分的に 5 分間の移動履歴を抽出して推定処理を行う．今回は，各路線に対して抽出した移動履歴を上り線 5 個，下り線 5 個，合わせて 10 個用意する．よって，20 個の移動履歴を用いて候補順位の推移を確認する．

評価実験の結果から，今回実装したシステムは候補順位が出ていない部分はなくなったものの，候補順位が以前より低下している移動履歴もあると確認した．従来システムと今回実装したシステムの候補順位の比較の一部を図 7 に示す．図中で黄色くハイライトしている部分が，従来システムと比較して大きな変化があった部分である．今回実装したシステムの方向推定処理において，駅に停車中と判定した場合，方向推定をせずに列車推定処理を行うように変更した．その結果，今回実装したシステムにおいて候補順位が表れていない部分はなくなった．今回実装したシステムにおいては，乗車している列車の候補順位が 5 分間 1 位を保っている実験データが 20 件中 17 件であると確認した．しかし，従来システムより候補順位が下がっている実験データが 20 件中 3 件あると確認した．このことから，我々は従来システムより候補順位が大幅に下がっていないと考えるものの，候補順位が従来システムより下がっている実験データに対しては原因究明と改善が必要であると考えた．

5.3 普通列車が優等列車に追い越される環境における候補順位の推移を比較する評価実験

従来システムと今回実装したシステムで追い抜きや追い越しといった難しい環境下における候補順位の推移を比較するために評価実験を行った．この評価実験は，従来システムと今回実装したシステムにおいて，普通列車が駅で優等列車に追い越される場合を想定し，候補順位がどのように推移するか確認する．この実験は，今回実装したシステムにおいて，通過駅の存在する優等列車が走行している環境下でも乗客の乗っている列車の推定ができるか確認する目的で行う．

この評価実験では，次の設定条件を従来システムと今回実装したシステムに適用して候補順位の推移を確認する．この評価実験では，以下に記されている路線で収集した長大な移動履歴の中から，普通列車が優等列車の追い越しのために停車している時刻から 5 分間の移動履歴を抽出して推定処理を行う．また，抽出した 5 分間の移動履歴を 10 秒ずつ伸ばして推定処理を行い，候補順位の変動を確認する．今回は，以下に記されている 3 路線の移動履歴を用いて候補順位の推移を確認する．

- 近鉄京都線　竹田駅から新田辺駅
- 近鉄大阪線　布施駅から高安駅
- 近鉄大阪線　高安駅から布施駅

評価実験の結果から，今回実装したシステムは追い越される環境下においても乗客の乗っている列車を推定できていると確認した．従来システムと今回実装したシステムの候補順位の比較を図 8 に示す．図中で黄色くハイライトしている部分が，従来システムと比較して大きな変化があった部分である．5.2 節の評価実験と同様に方向指定処理を変更したため，従来システムで候補順位が表れていなかった部分は，今回実装したシステムで候補順位が表れるようになった．そのため，今回実装したシステムにおいて，評価実験に用いた 3 件の実験データともに候補順位が 5 分間 1 位を保ち続けているのを確認した．このことから，我々は今回実装したシステムでも，普通列車が優等列車に追い越されるという難しい環境下で乗客の列車が推定できていると考えた．

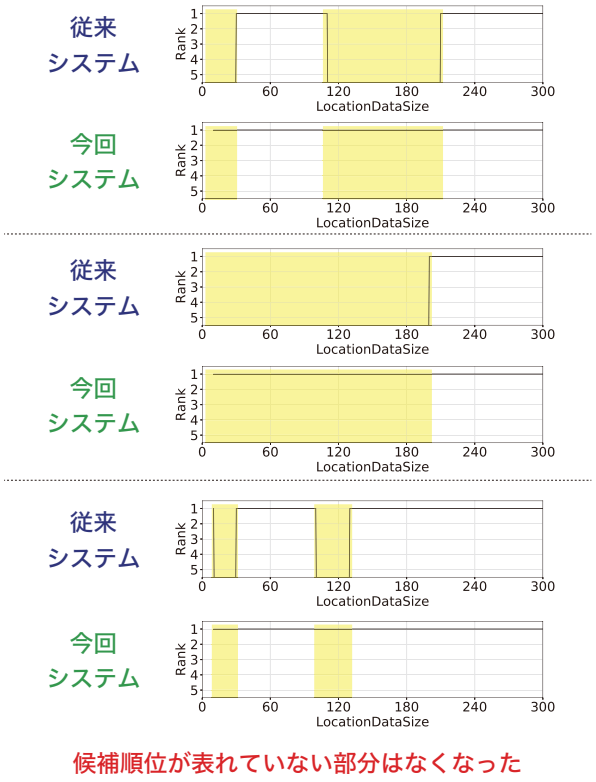

図 8　追い越される環境における候補順位の比較

6. まとめ

我々は，スマートフォンを用いて乗客の乗っている列車

を推定するアプリケーションにおける列車を推定する時間を短縮させるために，推定処理の変更を提案および実装をした．従来ステムにおいて，移動履歴が長大になるにつれて処理時間が大きくなってしまう問題を抱えていた．そのため，列車を推定する際に，乗客の移動履歴を構成する時系列的な位置情報を間引き，乗客の列車を推定する処理に変更した．また，路線推定処理や列車推定処理における類似度の算出処理を，ハウスドルフ距離を用いて類似度を算出処理に変更した．他にも，駅に停車しているときに推定処理を始めた場合，方向推定処理を行わずに列車推定処理を行うように変更した．

我々は従来のシステムより推定処理の時間が短縮されたか，また従来システムとの候補順位を比較するため評価実験を行った．推定処理の時間においては，従来のシステムでは移動履歴の長さが 300 秒のとき，平均で 47.25 秒の時間がかかっていたが，今回実装したシステムでは平均で 3.17 秒に短縮したのを確認した．この結果らは，我々は処理時間の面において推定処理の実用性は向上したと考える．次に，従来システムと今回実装したシステムにおいて候補順位の比較を行った．従来システムと比較して，候補順位が表れていない部分は無くなった．そのため，今回実装したシステムにおいて，実験データ 20 件中 17 件は，候補順位が 5 分間 1 位を保ち続けているのを確認した．しかし，従来システムより候補順位が下がっているデータが 20 件中 3 件あるのを確認した．他にも，普通列車が優等列車に追い越される環境下で候補順位がどのように推移するか確かめるための評価実験を行った．その結果，3 件の実験データ中 3 件とも候補順位が 5 分間 1 位を保ち続けたのを確認した．

今後の課題として，従来システムに比べて候補順位が下がっている事例に対して原因究明と改善をする課題や，このシステムの応用例に関する課題が挙げられる．列車推定処理において，ハウスドルフ距離を用いて類似度を算出する処理に変更したが，候補順位が従来システムよりも下がっている移動履歴に対して，ハウスドルフ距離がどのように推移をしているのか確認する必要があると考える．候補順位が従来システムより下がっている原因が判明すれば，類似度を出す処理に対して改善を加える必要もあると考える．また，我々はこのシステムを応用して，今後はスマートフォンを使い乗客の乗っている列車が，どの程度遅れているか推定するシステムを検討している．

参考文献

[1] 井上晴稀， 梶克彦：乗車中の列車の各駅到着時間を自動表示するスマートフォンアプリケーション，マルチメディア, 分散協調とモバイルシンポジウム 2019 論文集，Vol. 2019, pp. 506–514 (2019).

[2] Hyuga, S., Ito, M., Iwai, M. and Sezaki, K.: An online localization method for a subway train utilizing the barometer on a smartphone, *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, p. 50 (2016).

[3] Shin, H., Lee, G. and Han, D.: Subway Stop/Departure Detection using a Magnetic Sensor of the Smartphone, *Proceedings of the 2nd International Conference on Vision, Image and Signal Processing*, ACM, p. 37 (2018).

[4] Lee, G. and Han, D.: Subway train stop detection using magnetometer sensing data, *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 766–769 (2014).

[5] Jayawardena, A. N., Sachith, K. and Wijekoon, J. L.: Trainko: Poster Abstract: Train Arrival Time Prediction System for Sri Lanka, *Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '18, Piscataway, NJ, USA, IEEE Press, pp. 110–111 (2018).

[6] 綿貫圭太，下坂正倫：バス運行状況ウェブサービス情報を用いた乗換案内アプリのための路線バス所要時間推定，研究報告ユビキタスコンピューティングシステム (UBI)，Vol. 2018, No. 5, pp. 1–8 (2018).

[7] Jeong, R. and Rilett, R.: Bus arrival time prediction using artificial neural network model, *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749)*, pp. 988–993 (2004).

[8] Chang, H., Park, D., Lee, S., Lee, H. and Baek, S.: Dynamic multi-interval bus travel time prediction using bus transit data, *Transportmetrica*, Vol. 6, No. 1, pp. 19–38 (2010).

[9] Bai, C., Peng, Z.-R., Lu, Q.-C. and Sun, J.: Dynamic Bus Travel Time Prediction Models on Road with Multiple Bus Routes, *Computational intelligence and neuroscience*, Vol. 2015, p. 432389 (2015).

[10] Pongnumkul, S., Pechprasarn, T., Kunaseth, N. and Chaipah, K.: Improving arrival time prediction of Thailand's passenger trains using historical travel times, *2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 307–312 (2014).

[11] Liu, Y., Tang, T. and Xun, J.: Prediction algorithms for train arrival time in urban rail transit, *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6 (2017).

[12] 矢野幹樹，岩崎陽平，河口信夫：駅.Locky : 無線 LAN 位置推定を用いた時刻表アプリの開発，全国大会講演論文集，Vol. 72, pp. 289–290 (2010).

[13] 日高洋祐：日本版 MaaS (Mobility as a Service) モデルのシステムアーキテクチャの検討，研究報告高度交通システムとスマートコミュニティ (ITS)， Vol. 2018, No. 10, pp. 1–6 (2018).

[14] Kamau, J., Ahmed, A., Rebeiro-H, A., Kitaoka, H., Okajima, H. and Ripon, Z. H.: Demand responsive mobility as a service, *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, pp. 1741–1746 (2016).

[15] Tang, M., Lee, M. and Kim, Y. J.: Interactive Hausdorff Distance Computation for General Polygonal Models, *ACM Trans. Graph.*, Vol. 28, No. 3 (2009).