時間制約を考慮した観光ルート推薦システムの実装

2014SC078 高橋史也　2014SC011 平片友詞
指導教員：河野浩之

1　はじめに

現在 web の進化, スマートフォンの普及と共に地図サービスも進化を続けている．カーナビゲーションでは推薦ルートとして有料道路優先, 一般道路優先, 距離優先などの優先したい項目からユーザは好きなルートを選択する．その際有料道路を走る場合は料金が表示される [1]．また同時に距離, 到着時間なども表示される．スマートフォンの地図アプリでは徒歩, 車など移動手段に適したルート案内サービスなどがある [2]．それらは観光など初めて訪れる地では今や欠かせないものとなっている．しかし, それらは一つの目的地のルート推薦しか行うことができない．

本研究では, 施設の営業時間や滞在時間などの時間制約を考慮した観光スポットの推薦, 訪れる観光スポットを経由した出発地から目的地までのルートの出力を目的とする．Twitter などの SNS を用いた観光ルートの推薦は過去に行われている．しかし SNS のデータ量は多いが, 情報量が少ないため営業日や営業時間が不正確であり観光スポットも偏ってしまうと考えられる．そこで我々は地図データに Google Maps を使用する．Google Maps APIs の中の 3 つの API を用いて緯度経度やルートの出力などのデータを抽出する．また観光データも Google Maps に掲載されている観光情報を使用する．データの抽出には Google Places API を用いて観光スポットやスポットの営業時間などの抽出を行う．スポットの抽出には, スポット探索の中心となるスポットと目的地を結ぶ直線を引き, スポットを通る垂線を引く．次に, その垂線を境界線として目的地に近いスポットだけを立ち寄り地の候補として抽出する．

以下, 2 章では観光ルート推薦システムに関する先行研究について紹介する．3 章では 2 章で述べた先行研究の課題をもとに我々が提案する時間制約を考慮した観光ルート推薦システムを提案し, 4 章では提案システムのアルゴリズムについて述べる．5 章では提案システムの実装結果を示す．6 章をむすびとする．

2　観光ルートに関する先行研究

この章では本研究に対する先行研究について紹介する．2.1 節では, Twitter を用いた観光ルート推薦システムの手法について, 2.2 節では, Twitter と Google Places API を用いた観光ルート推薦システムの手法について紹介する．

2.1　位置情報付きツイートを利用した観光ルート推薦 [3]

中嶋らは位置情報付きツイートと Foursquare や Instagram のサービス, および旅行者のツイートに頻繁に現れる特徴語を用いて旅行情報を含むツイートを収集する手法を提案した．さらに, このツイートを収集したユーザーのタイムラインから観光スポット, 観光ルートおよび観光の関連情報を抽出する手法を提案している．しかし中嶋らの手法では, Twitter ユーザーのタイムラインからのみ観光スポットを収集しているため, 得られる観光スポットが少ないという問題がある．

2.2　マイクロブログを利用した観光ルート推薦における移動効率の改善 [4]

中川らは前述した中嶋らの研究を先行研究とし, 位置情報の添付されていないツイートも収集対象とした観光ルート推薦システムの手法を提案している．投稿文からユーザが実際に観光スポットに訪れていると推測されているもののみを収集し, Google Places API と組み合わせて観光スポットの抽出をしている．また, 投稿された時間から観光スポットの開園時間を予測し, ただ観光スポットに訪れるだけではなく, 移動効率をユーザの満足度を指数として考慮したルート推薦システムを構築している．

3　時間制約を考慮した観光ルート推薦システムの提案

この章では, Google Maps を用いた時間制約を考慮した観光ルート算出の提案について示す．3.1 節では, Google Maps API について, 3.2 節では, ルート推薦システムの手法について説明する．

3.1　Google Maps API について

Google Maps API とは, Google Maps を無料で自分の開発する Web サイトやアプリに掲載することができる仕組みである．Google Maps API は 18 種類用意されており, 用途に合わせて使用することが出来る．表 1 に我々が使用する Google Maps API を紹介する．始めに Google Maps Geocoding API ですべての施設の住所を地理的座標 (緯度, 経度) に変換する．次に目的地周辺の施設を抽出するために Google Places API のプレイス検索を用いる．

表 1　Google Maps APIs

Google Maps Places API	商業施設や観光スポットが取得可能
Google Maps Geocording API	住所を地理的座標にすることが可能
Google Maps Distance Matrix API	目的地の移動時間と距離を取得
Google Maps Directions API	複数ポイントのルート探索を取得

プレイス検索を用いることで目的地周辺の施設の住所や電話番号, ユーザの口コミなどを取得することができる．目的地から Google Places API で取得した施設までの距離を Google Maps Distance Matrix API で取得する．最後に Google Maps Direction API を用いてルート探索に利用する．

3.2 ルート推薦手法

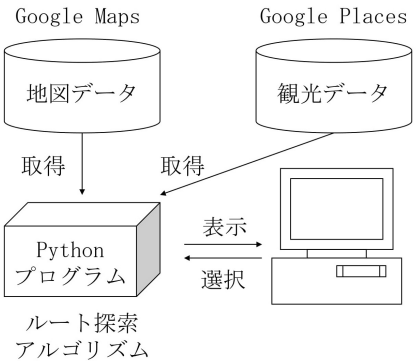

図 1 提案システムのアーキテクチャ

図 1 にルート推薦システムのアーキテクチャを示す．Google Maps API(地図データ) と Google Maps に掲載されている観光情報 (観光データ) から取得したデータを Python のプログラムを用いて処理する．ユーザは PC の画面に表示された優先したい項目を選択する．その後出力された自分の好みにあった観光スポットをその都度, 選択することによって希望の観光ルートが決定される．

4 観光ルート推薦のアルゴリズム

この章では我々が作ったシステムの全体的な流れについて示す．4.1 節では観光ルート推薦のアルゴリズムについて．4.2〜4.4 節では Google Maps API を用いたモジュールについて．4.5〜4.6 節ではスポットの選別について．4.7 節ではスポットの詳細取得について．4.8 節では立ち寄り地, ルート決定について説明する．

4.1 観光ルート推薦のアルゴリズム

第 4 章では図 2 のアルゴリムについて順に説明する．観光ルートを推薦するためにユーザは出発地点, 出発の時刻, 到着地点, 到着時刻, また, なにを優先したルートを生成したいかを入力する．図 2 の 1 について 4.2 節, 2 について 4.3 節, 3 について 4.4 節, 4 について 4.5 節, 5 について 4.6 節, 6 について 4.7 節で説明する．

4.2 Google Maps Geocoding API を用いた座標の特定

Google Maps Geocoding API の周辺検索リクエスト用の URL を作成し, JSON 形式で受け取った応答を Python で処理できるよう辞書型として扱う．パラメータとして座標を調べたいスポットの名前, または住所を指定する．応

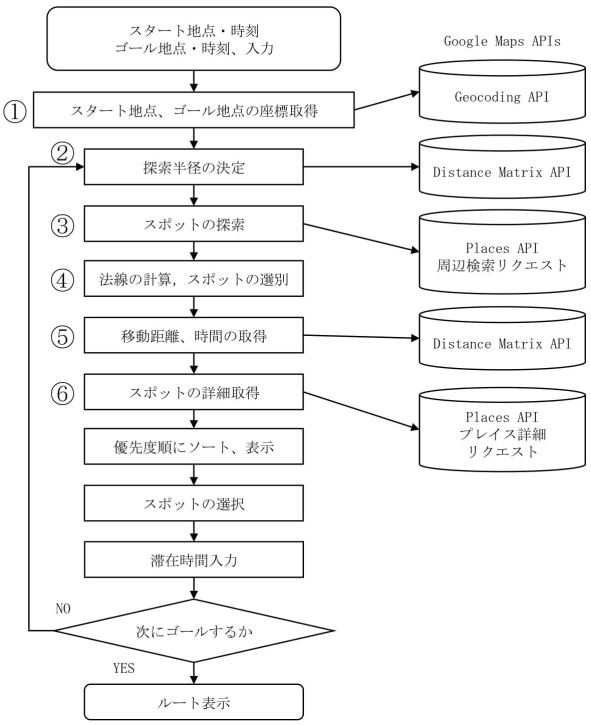

図 2 提案システムのアルゴリズム

答として緯度と経度を受け取るため, それを配列に格納して処理する．

4.3 Google Places API でのスポット探索半径の決定

本研究ではユーザが観光で訪れるスポットの候補を Google Places API を用いて探索する．その時にパラメータとして探索する半径の距離 [m] を設定しなければならない．それをユーザが選んだスポットと到着との距離を考慮して可変とする．具体的には, 立ち寄り地と到着地点との距離の 3 分の 1 を探索半径に設定する．2 点間の距離を求めるのには Google Maps Distance Matrix API を用いる．図 3 の説明をする．ユーザが設定した出発地 (S) と目

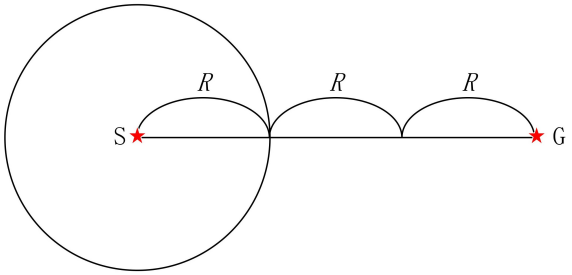

図 3 探索半径決定のイメージ

的地 (G) の座標が分かったあと, それを結ぶ線分 SG を引く．SG の 3 分の 1 の値を計算し Google Places API で周辺スポットを探索する半径 R に設定する．また, Google Places API の制約があるため, R の上限を 50000[m] とする．さらに, 到着に近づくにつれ R が小さくなるため, 下

限を 2000[m] とした.

4.4 Google Places API を用いた周辺スポットの探索

Google Places API の周辺検索リクエスト用の URL を作成し, JSON 形式で受け取った応答を Python で処理できるよう辞書型として扱う. パラメータとして, 探索の中心となる地点の座標と探索の半径 [m], スポットのタイプ (遊園地, 水族館, 動物園, 美術館, 博物館, 店, カフェ, レストラン) を指定する. 座標は後述する Google Maps Geocoding API を使って求めた座標を用いる. 半径は, ユーザが選択した立ち寄り地に応じて変動する. API からの応答には 20 件のスポットのデータが含まれている. それをパースしてハッシュとして扱う.

4.5 座標を考慮したスポットの選別

ユーザが立ち寄り地を選ぶ際, 選んだスポットによって到着から離れることは避けたい. そのため, Google Places API で探索したスポットからユーザに立ち寄り地の候補としてあげるスポットを選別する必要がある. 方法としては座標を考慮して必ず到着へと近づくようなスポットを計算する. 探索の中心となるスポット (S とする) と到着を結ぶ直線を引き, S を通るその垂線を引く. 垂線より到着から離れたスポットは候補として不適切とし, 除外する. 図 4

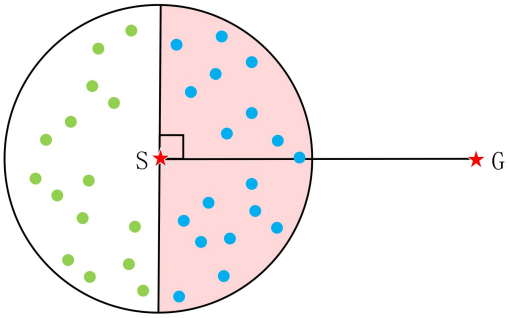

図 4 スポットを選別するイメージ

の説明をする. 立ち寄り地である S と目的地である G を結ぶ線分 SG を引く. そして, S を通る SG の垂線を引く. その垂線を境に探索されたスポットの内, G に近いスポットだけを立ち寄り地の候補として残す. それ以外のスポットは立ち寄り地の候補から除外する.

4.6 時間内に立ち寄れるスポットの選別

立ち寄り地を決定する際, ユーザが最初に設定した時間制約を考慮する必要がある. 出発地点からスポットまでの移動時間とスポットから到着までの移動時間の和が設定された観光時間を超える場合はそのスポットを除外しなければいけない. それを計算するために Google Maps Distance Matrix API を使用した. この API は, 複数の地点間の移動時間と距離を一回のリクエストで計算できる. また, 出発する時間や交通手段を設定できる. 今回パラメータとしては車での移動を設定した. Google Maps

```
def fx(sspot_array, gspot_array, wspot_array):
    sx = sspot_array[1] //座標を変数としておく
    sy = sspot_array[0]
    gx = gspot_array[1]
    gy = gspot_array[0]
    wx = wspot_array[1]
    wy = wspot_array[0]
    a = (gy - sy)/(gx - sx) //直線の傾きを求める
    a1 = a
    a2 = -1/a
    b1 = -a1 * sx + sy  //直線の方程式を求める
    b2 = -a2 * sx + sy
    if (gy>a2 * gx+b2) == True: //座標の判定
        if (wy > a2 * wx + b2) == True:
            choise = 1
        else:
            choise = 0
    elif (gy < a2 * gx + b2) == True:
        if (wy < a2 * wx + b2) == True:
            choise = 1
        else:
            choise = 0
    return choise
```

図 5 スポットを選別するプログラム

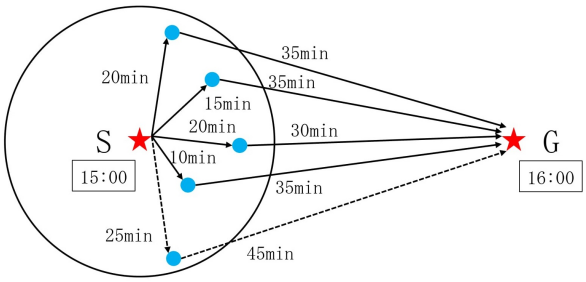

図 6 時間制約でスポットを除外するイメージ

Distance Matrix API で S からスポットまでの移動時間と, スポットから G までの移動時間を調べて, その和を求める. するとスポットに立ち寄るには上から順に合計で, 55 分, 45 分, 50 分, 45 分, 70 分の移動時間がかかる. しかし, 仮に S を 15:00 に出発して G に 16:00 までに到着しなければならない設定の場合, 一番下のスポットは間に合わない. そうしたスポットを立ち寄り地の候補から除外する.

4.7 スポットの詳細取得

座標, 時間制約を考慮して選別されたスポットの詳細情報を取得する. Google Places API のプレイス検索周辺検索リクエストではスポットの名前や住所, プレイス ID 等しか取得できない. ユーザにスポットを推薦する際, より詳しいスポットの情報が必要となる. そのため, Google Places API のプレイス詳細リクエストを使用する. この API は周辺検索リクエストで取得したプレイス ID をパラメータとしてスポットの詳細を取得することができる. 住所や電話番号, 営業時間, 写真, 価格帯, レビュー等が取得できる.

4.8　立ち寄り地, 推薦観光ルートの決定

ユーザはプログラムを実行し, 出発地点, 出発時刻, 到着地点, 到着時刻を入力する. また, なにを優先したいか (距離, レビュー, 価格帯) を選択する. プログラムで, これまでの流れで取得してきた情報を整理して表示し, ユーザはどのスポットに立ち寄るか選択する. また, その滞在時間を入力する. 立ち寄り地が全て決定したらブラウザが開き, Google Maps でルートが表示される.

5　提案システムの実装

この章では作成したシステムを使って実際にルートの生成を行う. 5.1 節では実際にルートの生成を行う. 5.2 節では本システムの実装結果の考察を述べる.

5.1　ルートの生成

実験環境は以下, 表 2 の通りである. プログラムを実行

表 2　実験用 PC

CPU	Core(TM)i5
OS	Windows8.1 Pro
メモリ	4.00GB
使用言語	Python3.6.1

して出発地点と到着地点, また, それぞれの時刻を入力する. 次に [距離, レビュー, 価格] から何を優先させたルートにしたいかを選択する. 例では距離を優先したルートになるようにスポットを選択する. 優先項目を選択するとスポットが探索される. 図 7 は [スポットナンバー：スポット名　ジャンル　移動所要時間　距離　レビュー　価格帯　最大可能滞在時間] で表示されている. スポットナンバーを入力して選択する. 999 を入力すると到着地点になる. スポットを選択したら滞在時間を入力する. 例では 93 番のモーニング喫茶リヨンに 100 分滞在することになる. (***部分がユーザの入力.) この流れを繰り返し, スポットナンバー 999 を選択するとブラウザが起動してルートが表示される.　図 8 では名古屋駅を出発し 3 つのスポットを

```
R=6578m
0:名古屋市博物館 museum 24分 10.7 km ★3.7
￥Unknown Max:309分
1:猫カフェねこまんま amusement_park 24分 10.3 km
★3.6 ￥普通 Max:312分
・・・(略)・・・
92:木曽路 名駅IMAIビル店 restaurant 8分 1.2 km
★3.6 ￥普通 Max:327分
93:モーニング喫茶リヨン cafe 6分 0.9 km ★3.4
￥安い Max:329分
999:GOAL
spot No.93
        **
staytime(min) >>100
               ***
```

図 7　プログラム実行例

図 8　名古屋駅から藤ヶ丘駅までのルート例

経由して藤が丘駅に到着している, また, 総移動時間は 69 分である.

5.2　実験結果の考察

本システムの実装では名古屋で観光ルートの出力を行った. 正しく観光ルートがブラウザに出力された. ただ, アメリカ等の海外の国での観光ルートを出力する際に注意が必要である. 最初の出発地点と到着地点を設定する際に英語で入力しないと日本国内のスポットが検索されて設定されてしまう場合がある. (例：「ワシントン」と入力すると日本の「ワシントンホテル」が探索されてしまうため, 「Washington」と入力する必要がある.)

6　むすび

本研究では Google Maps API を活用して時間制約を考慮した観光ルート推薦システムの実装ができた. 本システムでは 4.5 節の座標と垂線の計算により目的地まで一方向に進むことができ, ユーザの移動の負担を軽減させ移動効率をあげることができた. また, 4.6 節の時間内に立ち寄れるスポットの選別により到着時間に合わせたスポットの出力, ルートを生成することができた. スポットと時間はユーザが選択する形をとっているため, 自由度が高い観光ルートを生成することができた.

参考文献

[1] carrozzeria
http://pioneer.jp/carrozzeria/cybernavi/08cybernavi/function/root/benri.html (参照 2018-1-18)

[2] Google Map ナビ機能の使い方
http://appllio.com/how-to-use-google-map-car-navigation (参照 2018-1-18)

[3] 中嶋勇人, 新妻弘崇, 太田学, “位置情報付きツイートを利用した観光ルート推薦,” 情報処理学会. データベース・システム研究会報告, Vol.2013-DBS-158, No.28, pp.1-6, 2013.

[4] 中川智也, 新妻弘崇, 太田学, “マイクロブログを利用した観光ルート推薦における移動効率の改善,” 第 8 回データ工学と情報マネジメントに関するフォーラム (DEIM2016), H1-3, pp.1-8, 2016.