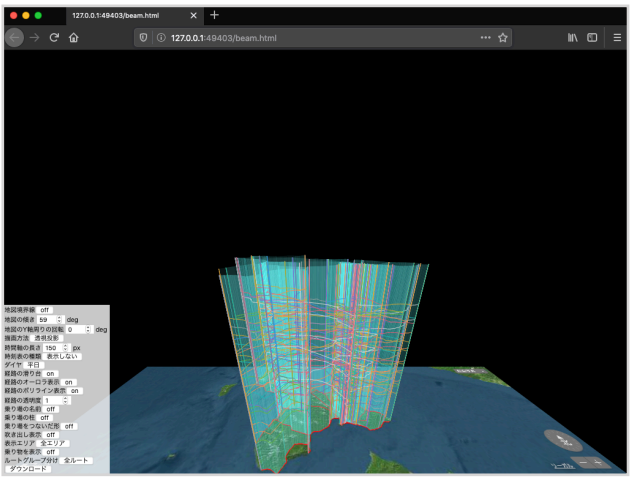
GSLetterNeo vol.139
2020年2月

Webブラウザにおける2Dコンテンツ上での透視投影変換を用いた3D表現（2）

松原 伸人 matubara@sra.co.jp

はじめに

テキストや画像や動画像に地図などWebブラウザ上のコンテンツの多くは二次元データです。今回はVol.137に続き、Webブラウザ上の透視投影変換を適用することで2Dコンテンツに情報を重ねて3D的に表示する方法を紹介します。既にある2Dコンテンツを3D的に表示できると、画像や地図の上に付加的な情報を立体的に重ねた表現ができます。オープンデータの１つとして、公共交通の運行情報が GTFS で公開されるようになってきています。次の画像は、開発中の、運行情報を地図上に表示するプロトタイプで BEAm（Bus Experience Amplified）と呼んでいます。



この画像は、プロトタイプを行うにあたり、共同研究を行っている公立はこだて未来大学の協力により、函館バスから提供いただいた運行情報を GTFS に変換して使用させていただきました。

北海道函館市で運行している函館バスの全運行経路を表示しています。地図上を這うようにして描かれている赤い線は、路線バスの各系統が走行する運行経路で、始発駅から停車する順に停留所を直線で結んで描いています。各運行経路の始発駅と終着駅の場所を垂直に線を描いて表しています。この垂直線は、上端が0時で下端が24時の時間軸も表しています。

GTFS での駅の位置と発着時刻

GTFSでは運行ルートを、複数の旅程を組み合わせた routes.txt で定義しています。旅程は、始発駅から終着駅まで停車駅の発着時刻と位置による移動行程として trips.txt に定義されています。各旅程の移動行程の各駅における発着時刻は、stop_times.txt に定義されています。駅の名称および位置は、

リファレンス | GTFS

リファレンス | Static Transit | Google Developers

stops.txt に定義されています。BEAm ではおもに、この routes.txt と trips.txt と stop_times.txt と stops.txt の4つのデータを用い、地図上に各系統の運行経路の駅の位置を直線でつないで表示し、各旅程の始発駅から終着駅までの各駅の発着時刻を直線でつないで表示します。

運行経路の表示

ある 1 つのルートの旅程の運行経路のデータは、stop_times.txt から旅程のID の trip_id を指定して得られる stop_time の配列として扱えます。GTFS の各テキストファイルは、カンマ区切りの CSV 形式で書かれています。BEAm では、各テキストの一行目に書かれているフィールド名をキーとして JSON 形式のオブジェクトとして扱っています。旅程ID を指定して運行経路データを得る JavaScript は次のように書けます。

```
let stop_times = gtfs.stop_timeByTrip_id[trip.trip_id].sort((s1, s2) => {
    return parseInt(s1.stop_sequence) - parseInt(s2.stop_sequence)
})
```

旅程ID が一致する stop_time の配列を得て、停車順序 stop_sequence フィールドに基づいて停車順に配列をソートしています。

stop_time の stop_id フィールドに書かれている停車駅の ID を用いて駅の位置が得られます。BEAm では各駅の stop_id をキーとする駅の辞書 stopDictionary をあらかじめ作成しています。

駅の位置は緯度と経度でそれぞれ stops.txt の stop_lat と stop_lon に定義されています。Vol.129 で紹介したように経緯度から地図上の表示位置 points が得られます。

```
let points = stop_times.map((stop_time) => {
    let aStop = gtfs.stopByStop_id[stop_time.stop_id]
    return mapPointToViewPoint(new mapkit.Coordinate(parseFloat(aStop.stop_lat), parseFloat(aStop.stop_lon)).toMapPoint(),
map.region.toMapRect(), rect)
})
```

得られた駅の座標の配列 points を直線でつないで canvas 上に描画します。

```
let ctx = canvas.getContext("2d")
ctx.beginPath()
points.forEach((p, i) => {
    if (i == 0) {
        ctx.moveTo(p.x, p.y)
    } else {
        ctx.lineTo(p.x, p.y)
    }
})
ctx.stroke()
```

運行経路の描画結果

今回のまとめ

今回は BEAm で用いている GTFS について説明し、運行経路の表示方法を紹介しました。説明したコードをまとめたプログラムは、次の URL を Webブラウザで開いて試せます。

https://www3.sra.co.jp/ktl/nobutomatsubara/test-draw_routes.html

このプログラムではGTFSデータがあるフォルダを指定すると、データを読み込んで地図上に全運行経路を表示します。運行経路は黒色の線で描いています。地図のズーム、移動操作時に座標を再計算して描画を行なっています。次回は、旅程の地図上での3D的な表示方法を紹介します。

GSLetterNeo vol.139
発行日 2020年2月20日
発行者 株式会社 ＳＲＡ 先端技術研究所
編集者 土屋 正人
バックナンバー https://www.sra.co.jp/gsletter/

お問い合わせ
gsneo@sra.co.jp
〒171-8513 東京都豊島区南池袋2-32-8
株式会社SRA