This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Rinne, Mikko; Bagheri Majdabadi, Mehrdad; Tolvanen, Tuukka

Automatic Recognition of Public Transport Trips from Mobile Device Sensor Data and Transport Infrastructure Information

Published in:
Personal Analytics and Privacy

DOI:
[10.1007/978-3-319-71970-2_8](https://doi.org/10.1007/978-3-319-71970-2_8)

Published: 14/06/2017

Document Version
Peer reviewed version

Please cite the original version:
Rinne, M., Bagheri Majdabadi, M., & Tolvanen, T. (2017). Automatic Recognition of Public Transport Trips from Mobile Device Sensor Data and Transport Infrastructure Information. In *Personal Analytics and Privacy: An Individual and Collective Perspective - 1st International Workshop, PAP 2017, Held in Conjunction with ECML PKDD 2017, Revised Selected Papers* (pp. 76-97). (Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 10708 LNCS). SPRINGER. https://doi.org/10.1007/978-3-319-71970-2_8

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Automatic Recognition of Public Transport Trips from Mobile Device Sensor Data and Transport Infrastructure Information

Mikko Rinne, Mehrdad Bagheri, Tuukka Tolvanen, and Jaakko Hollmén

Department of Computer Science,
Aalto University School of Science, Espoo, Finland
`firstname.lastname@aalto.fi`

Abstract. Automatic detection of public transport (PT) usage has important applications for intelligent transport systems. It is crucial for understanding the commuting habits of passengers at large and over longer periods of time. It also enables compilation of door-to-door trip chains, which in turn can assist public transport providers in improved optimisation of their transport networks. In addition, predictions of future trips based on past activities can be used to assist passengers with targeted information. This article documents a dataset compiled from a day of active commuting by a small group of people using different means of PT in the Helsinki region. Mobility data was collected by two means: (a) manually written details of each PT trip during the day, and (b) measurements using sensors of travellers' mobile devices. The manual log is used to cross-check and verify the results derived from automatic measurements. The mobile client application used for our data collection provides a fully automated measurement service and implements a set of algorithms for decreasing battery consumption. The live locations of some of the public transport vehicles in the region were made available by the local transport provider and sampled with a 30-second interval. The stopping times of local trains at stations during the day were retrieved from the railway operator. The static timetable information of all the PT vehicles operating in the area is made available by the transport provider, and linked to our dataset. The challenge is to correctly detect as many manually logged trips as possible by using the automatically collected data. This paper includes an analysis of challenges due to missing or partially sampled information in the data, and initial results from automatic recognition using a set of algorithms comparing measured trips with both live vehicle locations and static timetables. Improvement of correct recognitions is left as an ongoing challenge.

1 Introduction

Automatic detection of the door-to-door trip chains of people has a multitude of applications. For infrastructure planners and public transport providers knowledge of the true origins, destinations and volumes of commuters gives a much better understanding of requirements for the road and transport networks than

disconnected counts of cars or people at points of observation of the current network by using loops, cameras, ticket systems or manual passenger counting campaigns.

To assist users of public transport with relevant information about opportunities and problems it is vital to be able to generate a prediction of the next destination, time of travel and the means of public transport the person is going to use. For many people the same trips are repeated with regular cycles, i.e. daily, on (certain) weekdays, weekly or monthly. Such travellers can be proactively given targeted information about disruptions in road traffic or public transport lines, which they frequently use, at their personal times of regular usage. Real-time recognition of the current means of public transport and a prediction of the likely destinations can also be used to assist connecting passengers.

Multiple smartphone-assisted ways for automatic detection of public transport usage can be envisaged:

- *Ticketing system*: If coupled with the payment of the trip in systems where both vehicle entries and exits are registered, precise and correct information about the public transport part of trips can be obtained. Requires integration with the fare payment system.
- *Radio beacon*: Some transport providers may provide vehicle-installed radio transmitters, e.g. *WiFi*¹ or *Bluetooth*², which can be detected and compared with a list of beacon identifiers to detect the proximity of a public transport vehicle.
- *Live positioning*: The live positions of public transport vehicles can be matched with the measured positions of the passenger, searching for a sequence of continuous matches with suitable accuracy.
- *Static timetable*: Processed information about a trip carried out by a person (start and end locations and times, route geometry) using a vehicle can be compared with the information in a static timetable.

In our test arrangement we did not have access to the ticketing system of the public transport provider. Additionally, the current policy in the Helsinki region does not require registration of vehicle exits, and one validation of a regional ticket allows up to 80 minutes of transfer without a new validation. Therefore ticket-based information would not have been accurate even if it were available. At the time of testing listed Wi-Fi beacons were only available on trams, which does not give an adequate coverage of the transport network. Live positioning of public transport vehicles was made available by the public transport provider for a part of their fleet. Static timetable information was available for all lines and departures.

The target was to create a dataset for testing and benchmarking algorithms for automatic recognition of public transportation trips. The dataset is composed of position and activity recognition samples of 8 researchers between 9 am and 4 pm EET+DST on August 26th 2016, manual bookkeeping of their trips and the

¹ <http://www.wi-fi.org/>

² <https://www.bluetooth.com/>

related transport infrastructure data. Data collection for the limited period was pre-agreed with every campaign participant to enable publication of the dataset without privacy concerns.

Seven participants executed as many public transportation trips as possible during the designated time, especially emphasising travel by subway, as it has been the most challenging transportation mode for automatic recognition. The eighth participant logged some private car trips to provide comparison data, which should not match with any public transportation.

Due to the exceptional amount of travel per person during one day this dataset cannot be used as a source for studying the regular travel habits of public transportation users. It also doesn't contain repeatedly visited locations such as homes or offices. The challenge is to correctly recognise as many trips listed in the manual log as possible by using the other forms of data available. The dataset consists of the following tables:

- *Device data*: samples from mobile device sensors.
- *Filtered device data*: selection of the perceived activity and exclusion of data points at times of staying still using our algorithms.
- *Device models*: phone models used by the participants.
- *Manual log*: manual trip bookkeeping entries of participants.
- *Live position samples*: public transport fleet positions.
- *Static timetables*: public transport timetables valid on the date of the experiment.
- *Train stop times*: measured train stop time information for the date of the experiment.

The complete dataset is available³ in *github*⁴.

2 Background

Transportation mode estimation and classification using mobile phone sensors has been discussed e.g. in [3, 5, 13]. Automatic recognition of bus trips using mobile phone sensors has earlier been addressed by the *Live+Gov*⁵ project. Their mobile client collected both hardware sensor (accelerometer, rotation vector, gyroscope, magnetic field) and processed activity detection data [6]. A project proprietary human activity recognition classifier was built and trained. Public transport recognition was attempted using public transport infrastructure data similar to our study (static timetables and live locations of public transport vehicles). The client software also supported user marking of activity, but users were shown the currently detected activity, and consequently they have generally registered their activities only in cases, where the detected activity was incorrect. Service line detection probability of 37% is reported [8].

³ Static timetable data is referenced from the resources of the transport provider.

⁴ <https://github.com/aalto-trafficssense/public-transport-dataset>

⁵ <http://liveandgov.eu/>

In one study bus passengers with mobile phones have been used as voluntary crowdsourced sensors for reporting the live locations of buses [1]. In the Live+Gov project traffic jams were detected from irregularities in public transport fleet location data [6]. The specific context of parking has also been considered, estimating from mobile phone sensor data, whether a parking area has space available [9, 11].

The present effort belongs to the *TrafficSense*⁶ project, a part of the *Aalto Energy Efficiency Research Programme*⁷. The project aims to save time and energy in traffic by understanding regular travel habits of individuals and proactively assisting them with information available in the internet. Sustainability and business model of the approach have been considered in [4]. The current prototype service uses the methods outlined in this paper to discover the public transport usage of travellers and automatically and proactively relay targeted information about disruptions in the greater Helsinki area based on the individually detected lines and times of usage.

The current TrafficSense mobile client is coded in native Android⁸. It is available on *Google Play*⁹, but currently restricted to Finland, because the public transport recognition and disruption information functions are only available locally. The main map view of the mobile application can show trips marked with the mode of transport, ranked regular destinations of the user, current traffic situation as well as the current location of the user. The mode of transport can be edited or confirmed by the application user. Public transport disruption bulletins matching the trip history and current location of the user are shown as notifications.

The types, formats and sources of open traffic information available were reviewed in [10]. For information on public transport the resources from *Helsinki Regional Transport*¹⁰ (HRT) were used. Their developer resources are currently provided and documented under the *digitransit*¹¹ umbrella. The resources follow the *SIRI* (Service Interface for Real-time Information [10]) specifications¹². Live locations of the fleet were sampled from their live data API¹³. The static timetable data is in *GTFS* (General Transit Feed Specification¹⁴) format as specified by Google Inc.

⁶ <http://trafficsense.aalto.fi>

⁷ <http://aef.aalto.fi/en/>

⁸ AndroidTM is a trademark of Google Inc.

⁹ <https://play.google.com/store/apps/details?id=fi.aalto.trafficsense.trafficsense>

¹⁰ <https://www.hsl.fi/en>

¹¹ <https://www.digitransit.fi/en/developers/>

¹² <http://user47094.vs.easily.co.uk/siri/>

¹³ At the time of the study, the data was available at <http://dev.hsl.fi/siriaccess/vm/json>. The address has been later replaced by <http://api.digitransit.fi/realtime/vehicle-positions/v1/siriaccess/vm/json>

¹⁴ <https://developers.google.com/transit/gtfs/>

Train stop time information is encoded as *JSON* (Javascript Object Notation¹⁵ [2]). It has been fetched from the *digitraffic*¹⁶ service¹⁷ operated by the *Finnish Transport Agency*¹⁸ (FTA). All other sampled data in the repository is made available in *CSV* (Comma-Separated Values [12]) format. The repository includes scripts for importing the CSV tables into a *PostgreSQL*¹⁹ database, which is used internally by the TrafficSense project.

3 Manual bookkeeping by test participants

All test participants manually documented the details of their trips during the day. The information provided for each trip leg is shown in Table 1. Because the timestamps were manually recorded, their values are approximate. In total 103 trips were recorded.

Table 1. Manually logged trip information (tz = timezone).

Label	Type	Description
device_id	integer	Device identifier, aligned with <code>device_id</code> in Section 4.
st_entrance	string	Description of entrance to station building, if applicable. A map of the letters to mark entrances at each subway station is provided in the repository.
st_entry_time	timestamp (no tz)	Station entry time, if applicable.
line_type	string	SUBWAY / BUS / TRAM / TRAIN / CAR
line_name	string	Identifier, e.g. 7A, 102T, U, V.
vehicle_dep_time	timestamp (no tz)	Vehicle departure time.
vehicle_dep_stop	string	Platform or other station where vehicle was boarded.
vehicle_arr_time	timestamp (no tz)	Vehicle stop time at the end of the trip.
vehicle_arr_stop	string	Platform or other station where the vehicle was exited.
st_exit_location	string	Exit to station building, if applicable. Subway exit letters have been marked in maps provided in the repository.
st_exit_time	timestamp (no tz)	Time of exiting station, if applicable.
comments	string	Freeform comments about the trip leg.

4 Mobile device measurements

Mobile device samples were collected using the TrafficSense mobile client. The client uses the fused location provider and activity recognition of *Google Play Services*²⁰. Both of them are *virtual sensors* [7], abstracting information from

¹⁵ <http://www.json.org/>

¹⁶ http://www.liikennevirasto.fi/web/en/open-data/digitraffic\#.W0t_a10G0Ho

¹⁷ http://rata.digitraffic.fi/api/v1/history?departure_date=2016-08-26

¹⁸ <http://www.liikennevirasto.fi/web/en>

¹⁹ <https://www.postgresql.org/>

²⁰ <https://developers.google.com/android/guides/overview>

available hardware sensors²¹ into current best estimates of the location of the device and the activity the person carrying the device is currently engaged in. The sampled parameters contained in the dataset are listed in Table 2. The coordinates are in *WGS84*²² format. The table in the dataset contains 6,030 entries. It is formatted as CSV and sorted by `time` and `device_id`.

Table 2. Parameters sampled from mobile devices (tz = timezone).

Label	Type	Description
<code>time</code>	timestamp (no tz)	From the clock of the mobile device.
<code>device_id</code>	integer	Stable identifier for the device.
<code>lat</code>	double	Latitude, in WGS84.
<code>lng</code>	double	Longitude, in WGS84.
<code>accuracy</code>	double	Radius, in meters, estimated by the <i>fused location provider</i> of the mobile device.
<code>activity_1</code>	enum	Activity with highest confidence, provided by <i>activity recognition</i> of Google Play Services. Values: <code>IN_VEHICLE</code> , <code>ON_BICYCLE</code> , <code>RUNNING</code> , <code>STILL</code> , <code>TILTING</code> , <code>UNKNOWN</code> , <code>WALKING</code> .
<code>activity_1.conf</code>	integer	Percentage of recognition certainty, 100 = best confidence.
<code>activity_2</code>	enum	Value of second-highest confidence activity.
<code>activity_2.conf</code>	integer	Percentage of recognition certainty.
<code>activity_3</code>	enum	Value of third-highest confidence activity.
<code>activity_3.conf</code>	integer	Percentage of recognition certainty.

4.1 Mobile client filtering algorithms

The client alternates between **ACTIVE** and **SLEEP** states. The current default value for the sleep timer is 40 seconds. If the detected position changes by a distance longer than the accuracy of the position fix during a period of the perceived activity indicating **STILL**, the timer is restarted. This rule aims to prevent erroneous transitions to **SLEEP** state while the device is moving, but activity detection perceives it as being still. Such situations typically occur during smooth trips on rails, i.e. trains, trams and subways.

In **ACTIVE** state position is requested from the *fused location provider*²³ provided by Google Play Services with *high accuracy* and a 10 second interval, which yields a sufficiently accurate description of the route up to motorway speeds (333 m sample interval at 120 km/h). In **SLEEP** state position requests are dropped to *no power* priority, which means that position fixes are passed to our client only if requested by another application. Activity reports are always requested with a 10 second interval, but as a form of device power saving, during **STILL** detections activity recognition interval has been observed to increase up to 180

²¹ the availability of a particular hardware sensor may vary between different device models

²² <http://gisgeography.com/wgs84-world-geodetic-system/>

²³ <https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderApi>

seconds. As a result, sometimes the client may need up to ≈ 200 s of movement to wake up from **SLEEP** state.

In our data format each accepted position record is coupled with the latest activity information. The timestamp of the entry is the timestamp of the position, not the activity detection. Therefore the same detected activity may repeat over multiple points. The received position fixes ('points') are filtered as follows (Figure 1):

- If 60 minutes have passed since the last accepted point, any point is accepted (to provide a “ping” effect and record that the client was running).
- Accuracy²⁴ must be better than 1000m.
- If `activity != last queued activity` and `activity` is ‘good’²⁵, the point is accepted.
- If `(activity == last queued activity) and (distance to last accepted point > accuracy)`, the point is accepted.

Unless stated otherwise above, the parameters have been chosen experimentally, searching for balance between accurate trip data, low mobile phone battery consumption and reasonable quantity of samples. Adapting the location sample rate to the velocity of the mobile device could further improve efficiency especially if satellite positioning can be switched off between samples. The biggest challenge was to keep the client from going to **SLEEP** during activity. The impact could be mitigated by a longer **SLEEP** timer at the cost of increased power consumption. A better approach would be to improve the accuracy of the activity recognition. Using the chosen parameters, the resulting dataset contains about 30% of the theoretical maximum²⁶ number of points.

The fused location provider used by the mobile client combines data from satellite, Wi-Fi and cellular positioning. Despite that, sometimes positioning errors occur. In a typical case the correct position alternates with a distant point, probably due to changing between different positioning methods within the fused location provider.

4.2 Filtered device data

The dataset also includes a table of filtered `device_data`. The filtering operation serves two main purposes:

- Remove periods during which the terminal was not substantially moving.
- Find the most likely activity for each remaining data point.

Filtered device data is included in the published data set, because some of our recognition algorithms use it. A new candidate solution is welcome to base itself on the more complete `device_data` instead, and implement other filtering approaches. The activity filtering algorithm has a clear impact on recognition

²⁴ Estimated by the *fused location provider*.

²⁵ not **UNKNOWN** or **TILTING**

²⁶ one point every 10 seconds from every terminal

Fig. 1. Filtering algorithm for incoming position fixes.

results, as can be seen from the data shown in Section 5.1. The following parameters are included in **device-data-filtered** (corresponding descriptions are the same as in Table 2):

1. `time` (timestamp no tz)
2. `device_id` (integer)
3. `lat` (double, latitude)
4. `lng` (double, longitude)
5. `activity` (enum value of the winning activity)

The table contains 5975 points. The CSV-version is sorted by `time` and `device_id`.

4.3 List of device models

A list of the models of the eight smartphones used by the test participants is provided as a separate table. It is included, because some differences were observed in e.g. activity recognition performance between different devices. The table contains the following columns:

1. `device_id` (integer, same as in Table 2)
2. `model` (string name of the model)

5 Public transport infrastructure information

Details of the information sourced from public transport infrastructure providers is described in this section.

5.1 Live positions of public transport vehicles

The live positions of the public transport fleet were obtained from HRT and sampled at 30 second intervals. The columns recorded into the dataset are shown in Table 3. The time period was restricted to the time of the trial. The maximum and minimum coordinates recorded by the participants were checked and the live vehicle position data was filtered (as a rectangle) to include only the area surrounding the locations sampled from the test participants. The table length is 229,451 entries.

Table 3. Live positions of public transport vehicles (tz = timezone). All data as provided by the transit live data API.

Label	Type	Description
time	timestamp (no tz)	The time the position was recorded.
lat	double	Vehicle location latitude, in WGS84.
lng	double	Vehicle location longitude, in WGS84.
line_type	enum	One of: SUBWAY / BUS / TRAM / TRAIN / FERRY.
line_name	string	Identifier for the public transport line, e.g. 7A, 102T.
vehicle_ref	string	Distinguish between different vehicles with the same line_name in traffic at the same time.

5.2 Static timetables

The static timetables from HRT are not included in the data repository, but the file is available through the following link: http://dev.hsl.fi/gtfs/hsl_20160825T125101Z.zip. It can be used e.g. with an *OpenTripPlanner*²⁷ (OTP) server to query for trips with matching start and end locations and start times. A more precise description for the specific task of querying timetable information can be found in the documentation for the digitransit²⁸ API. The data includes the static timetables for all the public transport vehicles (also local trains) used by the study participants.

5.3 Train stop times

The dataset also includes information about the recorded stop times of local trains at stations on the day of the study. The information²⁹ includes a JSON-format **junat** object, including the stopping times of trains at each station as

²⁷ <http://www.opentripplanner.org/>

²⁸ <https://www.digitransit.fi/en/developers/services-and-apis/1-routing-api/itinerary-planning/>

²⁹ <https://github.com/aalto-trafficsemp/public-transport-dataset/tree/master/trains-json>

described (in Finnish) at <http://rata.digitraffic.fi/api/v1/doc/index.html#Junavastaus>. Information on the referenced stations, including their locations, can be obtained as <http://rata.digitraffic.fi/api/v1/metadata/stations>. The description of the stations “Liikennepaikat” format is available (in Finnish) at <http://rata.digitraffic.fi/api/v1/doc/index.html#Liikennepaikkavastaus>.

6 Data coverage and limitations

No local trains and not all the buses are included in the live transit data. Travelled `line_name` specifiers (appearing in the manually logged data) **found** in `transit_live` are:

- Trams: 2, 3, 7A, 8, 9
- Buses: 16, 67, 72, 550, 560
- Subways (line_name in manual-log varies and cannot be used for comparison)

Travelled `line_name` specifiers **not found** in `transit-live` are:

- Trains: E, I, K, P, U
- Buses: Espoo18³⁰, 95, 102T, 103T, 105, 110, 132T, 154, 156

In terms of recorded trips, the 10 bus trips shown in Table 4 can therefore definitely not be found in `transit_live`. The 28 trips in Table 5 can be confirmed as being discoverable and identifiable using transit live data, as they have been found with our algorithm.

Table 4. Manually logged bus trips, which are **not** available in live transit data.

device_id	Departure time	line_name
1	09:07:00	154
5	09:08:00	110
5	09:16:00	18 (Espoo)
3	09:59:32	132T
2	10:38:51	103T
6	13:26:00	95
2	13:34:35	103T
3	14:21:08	105
3	15:26:39	102T
1	15:59:00	156

³⁰ Bus `line_name` specifiers <100 can be re-used by the adjoining cities of Helsinki, Espoo and Vantaa. Bus 18 from Helsinki is included in `transit_live`, but the test participant used bus 18 of Espoo.

Table 5. Manually logged trips correctly recognised from live data (28) using our algorithms (logged trips matching multiple segments have multiple rows).

dev_id	log_start	log_end	log_type	log_name	id	segm_start	segm_end	activity	recd_type	recd_name
1	09:41:00	09:44:00	TRAM	3	6	09:40:58	09:44:59	IN_VEHICLE	TRAM	3
1	10:06:00	10:19:00	TRAM	7A	12	10:07:24	10:13:23	IN_VEHICLE	TRAM	7A
1	10:06:00	10:19:00	TRAM	7A	13	10:13:34	10:14:56	ON_BICYCLE		
1	10:06:00	10:19:00	TRAM	7A	14	10:15:37	10:20:11	IN_VEHICLE	TRAM	7A
1	10:46:00	10:52:00	SUBWAY	V	19	10:43:41	10:52:45	IN_VEHICLE	SUBWAY	V
1	11:01:00	11:06:00	SUBWAY		21	11:01:12	11:09:28	IN_VEHICLE	SUBWAY	V
1	11:15:00	11:19:00	SUBWAY	M	22	11:16:07	11:20:23	IN_VEHICLE	SUBWAY	M
1	13:14:00	13:30:00	SUBWAY		29	13:13:31	13:30:27	IN_VEHICLE	SUBWAY	V
1	14:18:00	14:30:00	SUBWAY		34	14:17:32	14:22:30	IN_VEHICLE	SUBWAY	V
1	14:18:00	14:30:00	SUBWAY		35	14:22:41	14:26:37	ON_BICYCLE		
1	14:18:00	14:30:00	SUBWAY		36	14:27:18	14:31:15	IN_VEHICLE	SUBWAY	V
1	14:38:00	14:51:00	SUBWAY		37	14:31:25	14:39:23	WALKING		
1	14:38:00	14:51:00	SUBWAY		38	14:39:54	14:51:15	IN_VEHICLE	SUBWAY	V
1	14:57:00	15:05:00	SUBWAY		40	14:56:47	15:06:45	IN_VEHICLE	SUBWAY	V
1	15:36:00	15:55:00	SUBWAY		44	15:36:27	15:42:17	IN_VEHICLE	SUBWAY	M
1	15:36:00	15:55:00	SUBWAY		45	15:42:27	15:48:05	ON_BICYCLE		
1	15:36:00	15:55:00	SUBWAY		46	15:50:21	16:18:46	IN_VEHICLE		
2	13:14:34	13:30:03	SUBWAY	To west	54	13:14:01	13:31:34	IN_VEHICLE	SUBWAY	V
3	10:59:27	11:10:08	SUBWAY	To east	74	10:59:43	11:09:46	IN_VEHICLE	SUBWAY	V
3	11:27:06	11:34:11	SUBWAY	To west	76	11:28:05	11:34:36	IN_VEHICLE	SUBWAY	V
3	13:14:14	13:24:28	SUBWAY	To west	81	13:15:14	13:23:37	IN_VEHICLE	SUBWAY	V
4	10:17:00	10:28:00	TRAM	7A	105	10:14:27	10:38:15	IN_VEHICLE	TRAM	7A
4	10:47:00	10:53:00	SUBWAY	to east	107	10:45:59	10:53:39	IN_VEHICLE	SUBWAY	V
4	13:21:50	13:31:00	SUBWAY	to west	116	13:21:20	13:31:57	IN_VEHICLE	SUBWAY	V
4	13:44:00	13:47:00	SUBWAY	to west	118	13:45:12	13:47:30	IN_VEHICLE	SUBWAY	V
4	14:28:00	14:31:00	TRAM	9	122	14:26:04	14:31:10	IN_VEHICLE	TRAM	9
5	10:52:00	11:02:00	BUS	16	136	10:52:30	11:03:28	IN_VEHICLE	BUS	16
5	14:40:00	14:50:00	SUBWAY	R	149	14:40:46	14:50:51	IN_VEHICLE	SUBWAY	V
5	15:10:00	15:20:00	TRAM	9	151	15:08:38	15:24:01	IN_VEHICLE	TRAM	9
6	14:01:00	14:17:00	BUS	560	162	13:59:21	14:25:50	IN_VEHICLE	BUS	560
8	10:18:00	10:23:00	TRAM	9	174	10:17:00	10:23:55	IN_VEHICLE	TRAM	9
8	10:44:00	10:53:00	TRAM	9	178	10:44:22	10:54:44	IN_VEHICLE	TRAM	9
8	11:08:00	11:13:00	BUS	72	180	11:07:46	11:14:48	IN_VEHICLE	BUS	72
8	11:25:00	11:41:00	SUBWAY		181	11:26:23	11:42:24	IN_VEHICLE	SUBWAY	M
8	13:49:00	14:04:00	BUS	550	187	13:48:41	14:04:30	IN_VEHICLE	BUS	550

The 38 trips in Table 6 have detected IN_VEHICLE (sometimes also ON_BICYCLE) segments in `device_data_filtered` overlapping with entries in `manual_log`. Therefore these trips should be recognisable if the particular vehicle exists in `transit_live`, but as shown in the `recd_type` and `recd_line` columns, they have not been correctly recognised by the current algorithms. Logged trips matching multiple recorded segments (e.g. 09:39 trip with tram 9 matching segments 97, 98 and 99) are listed separately for each segment. For performance gains in recognising trips using live data, these are expected to be the best candidate trips.

Table 6. Manually logged trips having a corresponding IN_VEHICLE segment in sampled data (38), but not correctly recognised from `transit_live` by the current algorithms.

dev_id	log_start	log_end	log_type	log_name	id	segm_start	segm_end	activity	recd_type	recd_name
1	09:31:00	09:35:00	SUBWAY	V	4	09:31:44	09:36:37	IN_VEHICLE		
1	09:48:00	09:51:00	SUBWAY		8	09:49:19	09:54:48	IN_VEHICLE		
1	10:28:00	10:35:00	SUBWAY		17	10:28:00	10:37:33	IN_VEHICLE		
1	11:42:00	11:43:00	SUBWAY	M	24	11:40:36	11:46:04	IN_VEHICLE		
1	11:56:00	12:01:00	SUBWAY		26	11:55:14	12:02:53	IN_VEHICLE		
1	13:43:00	13:52:00	TRAM	9	31	13:41:40	13:46:33	ON_BICYCLE		
1	13:43:00	13:52:00	TRAM	9	32	13:46:54	14:05:12	IN_VEHICLE		
1	13:58:00	14:04:00	SUBWAY		32	13:46:54	14:05:12	IN_VEHICLE		
1	15:24:00	15:26:00	SUBWAY		42	15:24:38	15:28:24	IN_VEHICLE		
2	11:04:00	11:06:33	SUBWAY	To east	50	11:04:13	11:12:20	IN_VEHICLE		
2	11:26:18	11:50:00	SUBWAY	To east	51	11:22:41	11:58:31	IN_VEHICLE		
2	11:50:44	11:58:40	SUBWAY	To west	51	11:22:41	11:58:31	IN_VEHICLE		
3	10:21:21	10:23:01	SUBWAY	To east	70	10:17:28	10:25:54	IN_VEHICLE		
3	10:37:47	10:47:59	SUBWAY	To east	72	10:36:03	10:48:14	IN_VEHICLE		
3	11:56:11	11:58:43	SUBWAY	To west	78	11:57:56	12:01:18	IN_VEHICLE		
3	13:35:13	13:41:37	TRAM	7A	83	13:28:48	13:41:28	IN_VEHICLE		
3	13:45:37	13:47:12	SUBWAY	To west	85	13:45:42	13:48:24	IN_VEHICLE		
3	14:07:59	14:09:23	SUBWAY	To west	87	13:59:48	14:14:14	IN_VEHICLE		
3	14:11:30	14:13:34	SUBWAY	To west	87	13:59:48	14:14:14	IN_VEHICLE		
3	14:42:40	14:52:31	SUBWAY	To east	90	14:39:47	14:52:22	IN_VEHICLE		
3	15:02:03	15:13:57	SUBWAY	To west	92	15:01:12	15:16:46	IN_VEHICLE		
4	09:39:00	09:45:00	TRAM	9	97	09:38:15	09:41:02	IN_VEHICLE	TRAM	3
4	09:39:00	09:45:00	TRAM	9	98	09:41:12	09:42:25	ON_BICYCLE		
4	09:39:00	09:45:00	TRAM	9	99	09:43:08	09:45:15	IN_VEHICLE	TRAM	1
4	10:35:00	10:37:00	SUBWAY	to east	105	10:14:27	10:38:15	IN_VEHICLE	TRAM	7A
4	11:04:00	11:13:00	SUBWAY	to east	109	11:04:55	11:14:10	IN_VEHICLE		
4	11:26:00	11:28:00	SUBWAY	to west	111	11:23:59	11:31:36	IN_VEHICLE		
4	11:36:00	11:38:00	SUBWAY	to east	113	11:36:32	11:46:49	IN_VEHICLE		
4	11:45:00	11:47:00	SUBWAY	to east	113	11:36:32	11:46:49	IN_VEHICLE		
4	13:57:00	14:01:00	SUBWAY	to west	120	13:57:00	14:02:39	IN_VEHICLE		
4	15:15:00	15:20:00	SUBWAY	to east	125	15:15:29	15:20:13	IN_VEHICLE		
5	09:58:00	10:08:00	TRAM	7A	133	09:58:03	10:14:13	IN_VEHICLE		
5	10:12:00	10:13:00	SUBWAY	M	133	09:58:03	10:14:13	IN_VEHICLE		
5	11:11:00	11:14:00	SUBWAY	R	138	11:09:19	11:14:28	IN_VEHICLE		
5	11:18:00	11:20:00	BUS	67	140	11:18:23	11:20:53	IN_VEHICLE		
5	11:38:00	11:52:00	SUBWAY	M	142	11:38:53	11:50:59	IN_VEHICLE		
5	15:36:00	15:39:00	TRAM	9	153	15:37:43	15:40:10	IN_VEHICLE		
6	11:38:00	11:52:00	SUBWAY		156	11:45:26	11:50:26	IN_VEHICLE		
6	14:19:00	14:25:00	SUBWAY		162	13:59:21	14:25:50	IN_VEHICLE	BUS	560
6	15:50:00	15:59:00	TRAM	9	165	15:50:58	15:59:24	IN_VEHICLE		
8	11:52:00	11:58:00	SUBWAY		183	11:52:35	12:00:18	IN_VEHICLE		

Finally, the 13 trips in Table 7 have no overlapping `IN_VEHICLE` segment from the mobile device samples. It is also noted that:

- 3/13 trips (2 subway + 1 tram) have no overlapping mobile device samples at all
- 10/13 trips overlap with `WALKING` (in 6/10 cases shadowing the whole trip)
- 12/13 trips (`device_id` 5 and 6) were carried out with the same device model

The `device_data` sampled before and after a time segment with no data should still reveal that the mobile device has moved without recording positions, i.e. it has either erroneously been in `SLEEP` state or no position fixes have been received. Looking at the timestamps and locations recorded before and after the break period with geographical displacement, a set of candidate options could still be found from either the static timetable or live fleet position data. In downtown areas with many transport alternatives errors would be likely, but in less busy areas the set of alternatives will be smaller. Faulty recognitions changing car trips to public transport would still be very likely, so this type of recognition should only be attempted in scenarios where the penalty of false positive is not very high. This type of recognition has not been attempted in the current experiment.

Table 7. Manually logged trips having **no** corresponding `IN_VEHICLE` segment (13) in sampled data.

dev_id	log_start	log_end	log_type	log_name	id	segm_start	segm_end	activity
4	15:53:00	15:55:00	SUBWAY	to west	127	15:41:39	15:58:40	WALKING
5	11:30:00	11:31:00	SUBWAY	R	141	11:21:26	11:36:31	WALKING
5	13:22:00	13:28:00	SUBWAY	M	144	13:17:08	13:23:38	WALKING
5	13:34:00	13:43:00	SUBWAY	R	145	13:28:54	13:37:27	WALKING
5	13:54:00	14:02:00	SUBWAY	V				
5	14:20:00	14:30:00	SUBWAY	R	148	14:02:42	14:39:19	WALKING
5	15:36:00	15:39:00	TRAM	9	152	15:34:32	15:37:11	WALKING
5	15:51:00	15:52:00	TRAM	8				
5	15:56:00	15:57:00	SUBWAY	M				
6	13:47:00	13:49:00	SUBWAY		161	13:44:23	13:57:30	WALKING
6	14:29:00	14:30:00	SUBWAY		163	14:26:10	15:03:54	WALKING
6	15:02:00	15:20:00	SUBWAY		163	14:26:10	15:03:54	WALKING
6	15:41:00	15:43:00	TRAM	2	164	15:27:41	15:50:48	WALKING

7 Methods of automatic recognition

The collected live public transport vehicle locations and static timetables, in conjunction with the user location traces, were used to automatically recognise public transport trips taken by users. Trip legs consisting, after filtering, of a continuous sequence of `IN_VEHICLE` activity were matched against the vehicle position traces and timetable data. The train stop times were not used.

7.1 Live vehicle location

The vehicle location data from Table 3 is compared with the user locations collected by the personal mobile devices as described in Section 4. Potential issues in matching include:

- Missing vehicle or user data points
- Inaccurate location points
- Clock differences
- Distance between user and vehicle location sensor in longer vehicles
- Distance between location samples at higher vehicle velocities
- False matches to other public transport vehicles
- False positives where a car trip takes place near public transport vehicles
- Intermittent changes of the line name label on some vehicles in the live data

For performance reasons the number of user location samples used for matching a trip leg was limited to 40. To counteract the location accuracy and vehicle length issues, a distance limit of 100 metres was used for collecting vehicle matches. A greater limit may cause more false positive matches to appear.

With the sample rate of thirty seconds in the collected vehicle positions, a vehicle traveling at 80 km/h would produce samples every 667 metres, much in excess of the above mentioned distance limit. This could cause false negatives with the user location samples falling in between the vehicle points in such a way that they are not matched. To prevent this, the position sequence of each vehicle is processed into linestrings, and user point distances calculated against those line geometries.

The vehicle location points for composing the linestrings for comparison are collected in a ± 60 -second window around the timestamp of each user point sample. This allows for some clock difference, and sampling time difference.

For a vehicle to be accepted as a possible match, its linestring must be within the 100 metre distance limit for a minimum of 75% of the user location samples.

Each match within the permitted 100 metres’ distance accumulates a score of $100 - d$ when the distance is d metres. The vehicle with the highest score wins.

The line type and name are set according to the matched vehicle. In case of the vehicle having multiple line names in the matches, the most frequently occurring name is used. On some lines, a vehicle can intentionally change line name when passing a certain stop. Also, some vehicles in the data set change line name intermittently to false values.

In addition to the method described above (subsequently referred to as “New live”), a prior implementation (“Old live”) was evaluated. In the older implementation, four user trace sample points were used, and matched against vehicle location points in the surrounding time window. With the vehicle location sampling interval of 30 seconds, and the 100 metre distance criterion, this would be expected to cause otherwise optimal user trace point samples to potentially fall outside the matching radius once the vehicle speed exceeds $2 \times 100\text{m}/30\text{s} \approx 6.67\text{m/s} \approx 24\text{km/h}$.

For trams in Helsinki having an average speed of 14.7 km/h (2013–2014)³¹, with dense stops the new and old methods should produce similar results. More differences can be expected on the subway, and on bus routes with highway segments, where speeds are higher and stops more sparse.

7.2 Matching with static timetables

Comparison with static timetables is based on searching for public transport plans of past trips based on the sampled `IN_VEHICLE` segments. Firstly, a trip-plan query is sent to the OTP journey planner interface based on each candidate leg’s start-time, first point (origin) and last point (destination). Secondly, the resulting PT plans are compared to the leg’s start-time and end-time as well as the user location trace to identify whether the leg represents a PT ride. Potential issues in matching include:

- Missing or inaccurate user location points
- Inaccuracy in activity determination and filtered transition points
- Clock differences
- False positives when the route and timing of a car trip are similar to a segment of a public transport line

Table 8 explains the constants and Table 9 the variables and formulas used in the process. Figure 2 illustrates an example public transport trip and highlights the following parameters showing possible extra parts of the planned trip compared to the original sampled trip: $tWb, tWe, tPTb, tPTe$, where t indicates ”time”, W indicates ”walking” and PT indicates ”public transport”. The b denotes ”at the beginning of a trip or leg”, and e denotes ”at the end of a trip or leg”.

Table 8. Constants used in our query to the Open Trip Planner, validation and matching of the resulting trip plan with the recorded trip.

Parameter	Definition	Value
$dEmax$	Maximum acceptable distance inaccuracy as a result of displacement or delay in detecting the point of mode transition. This value denotes an offset in start point and/or end point of filtered <code>IN_VEHICLE</code> leg.	500 m
vW	Walking speed.	≤ 1.34 m/s
vPT	Speed of a PT vehicle.	≥ 3 m/s
$tEPT$	Deviation of the PT vehicle from schedule.	≤ 3 min

The data sampled from a mobile device can often have a fair amount of inaccuracy in location and activity detection. For this reason the filtered transition points starting and ending the vehicular trip leg can often appear between stops during the actual mass transit leg. Therefore almost all travel plans from the

³¹ https://www.hsl.fi/sites/default/files/uploads/13_2015_raitioliikenteen_linjastosuunnitelma_netti.pdf

Table 9. Variables and formulas used in our query to the Open Trip Planner, validation and matching of the resulting trip plan with recorded trip.

Par.	Definition	Formula	Value
tV	Duration of the sampled IN_VEHICLE trip leg.		
t	Duration of the planned trip.		From OTP
dEb	Distance offset (error) in detecting the starting point of a leg.		$\leq dE_{max}$
dEe	Distance offset (error) in detecting the end point of a leg.		$\leq dE_{max}$
tWb	Walking duration at the beginning of the planned trip to reach the boarding stop.	dEb/vW . For maximum acceptable inaccuracy: $tWb = 500/1.34 = 6.2min$	≤ 6.2 min
tWe	Walking duration at the end of the planned trip from the end stop to the destination.	dEe/vW	≤ 6.2 min
tPT	Duration of the PT leg included in the planned trip.		From OTP
$tPTb$	Extra transit ride at the beginning of the planned trip compared to the sampled IN_VEHICLE trip leg.	dEb/vPT	≤ 2.8 min
$tPTe$	Extra transit ride at the end of the planned trip compared to the sampled IN_VEHICLE trip leg.	dEe/vPT	≤ 2.8 min
ΔtPT	Extra transit ride of the planned trip compared to the sampled trip.	$ tPT - tV = tPTb + tPTe$	≤ 5.6 min
ΔtW	Extra walking in the planned trip compared to the sampled trip	$tWb + tWe$	≤ 12.4 min
Δt	Duration difference between the planned trip and the sampled IN_VEHICLE leg.	$\Delta tPT + \Delta tW = tWb + tPTb + tPTe + tWe$	≤ 18 min

Fig. 2. Matching a sampled trip with the static public transport timetable to detect whether or not the trip has been a PT trip as well as to identify the PT mode (e.g. Tram, Bus, Commuter Train).

Fig. 3. Matching the sampled trip route with the planned PT route. This figure illustrates an example of a mismatch between routes of planned PT transit and the sampled vehicular leg. PT vehicle of the planned trip takes a different route by partly passing different roads compared to the original sampled trip.

journey planner inevitably include walking sections in addition to the desired transit (vehicular) leg. The additional walking sections can appear at the beginning (tWb to walk from the detected start of vehicular leg to the public transport stop for boarding) and/or at the end of the trip leg (tWe to walk from the exit stop to the detected end of vehicular leg).

Selection of query parameters The origin and destination parameters of our OTP query are set equal to the leg's start and end geolocations respectively. The third query parameter, trip start-time, is set based on the leg's start-time as follows.

The maximum allowable inaccuracy in transition points is $dEmax = 500metres$. Therefore, to account for inaccuracy in time and place of origin, in our query to the journey planner the earliest permissible start-time of the trip is adjusted back by $tWb = 6.2minutes$ to allow 500 metres of walking at a speed of $vW = 1.34m/s$ to the boarding stop. Figuratively speaking, in case of offset in recorded IN_VEHICLE leg, we allow the traveller to catch the desired transit line according to actual timetable by walking to the nearest stop. If there is no delay and time and place of transition are completely accurate, the traveller would just wait 6.2 minutes at the boarding stop to take the transit line. Adjacent stops are assumed to be no more than one kilometre ($2 \times 500m$) apart.

Correspondingly, to relax both the origin and the destination, a maximum of $2 \times dE_{max} = 1,000$ metres of walking is requested for each resulting plan. The OTP query requests for three PT plans for each candidate vehicular leg.

Filtering and validation of query response Out of the three returned plans the best match, if good enough, is chosen. Plans that match the following criteria can be *discarded*:

- Plans having a total duration of more than $tEPT = 3minutes$ shorter than the user-recorded vehicle leg, thereby assuming that the transit vehicle must travel closely according to schedule to avoid false positive matches.
- Plans having a total duration of more than $\Delta dt = 18minutes$ longer than the user-recorded vehicle leg. This difference denoted by Δdt includes the walk times for dealing with location inaccuracy discussed above ($\Delta tW = 6.2min \times 2 = 12.4min$ for the whole trip), and the time assumed for a mass transit vehicle to travel between adjacent stops (maximum 1000m for the whole trip at minimum speed of $vPT = 3m/s$, equal to $\Delta tPT = 5.6minutes$).
- Plans where the duration of the included transit leg (tPT) differs by more than $\Delta tPT = 5.6minutes$ from the user-recorded vehicular leg (tV), based on assumed time of travel of the vehicle between adjacent sparse stops.
- Plans where the start of the included transit leg differs from that of the recorded vehicular leg by more than 5.8 minutes. This value considers a public transport vehicle traveling between adjacent stops at the beginning of the trip ($tPTb = 2.8min$, for 500m) plus an assumed $tEPT = 3min$ deviation of the transit line from its schedule.

The plans returned by the journey planner interface include location point sequences of the planned trips. As illustrated in Figure 3, the location point sequences are also matched against the recorded user trace to verify that the route of the public transport line matches the recorded user trace. For comparison the recorded user trace is sampled at no less than 100 metre intervals, and leading and trailing points may be ignored based on the assumed inaccuracy of origin and destination. A minimum of 70% of sample points must match a plan point within 100 metres, and no more than four adjacent sample points may fall outside 100 metres of the plan points for the plan to qualify. Out of the qualifying plans, the one with the closest start-time to that of the recorded leg wins.

Our current algorithm only validates plans containing a single vehicular leg. However, quick transfers between vehicles may not have been detected as separate vehicular legs by the activity detection and filtering of the mobile device. Allowing trip plans with transfers, and splitting the reference vehicular leg accordingly, could produce improved detection for such cases. An example of the fusion of a BUS leg and a SUBWAY leg to a single IN_VEHICLE leg can be found from device 6 at 13:59 to 14:25, where a short transfer has occurred between 14:17 and 14:19.

8 Recognition results

Compared with the 103 manually logged trips, 86 `IN_VEHICLE` segments were recognised by our filtering algorithm, 85 of them overlapping at least a part of a logged trip.

The 28 trips recognised with correct `line_name` from `transit_live` using the new live algorithm were detailed in Table 5. The statistics for all the recognition methods are collected to Table 10. The static approach yielded the highest number of matched trips, but with one false `line_type` and two false bus `line_names`. Live recognition performance suffered from the absence of many buses and all trains in the data, but especially subway trip detection was significantly improved with the new live algorithm compared with the old one. Looking at the combined results but without requiring the `line_name` to match, 60% of bus and tram trips, 44% of train trips and 43% of subway trips were recognised. If the correct `line_name` is required, bus recognition success drops to 47%.

Table 10. Trip matching statistics for all recognition methods compared to the manually logged trips (“line name” = matching line name, “line type” = matching line type).

	Static	Old live	New live	Combined	Logged
Bus	8	3	4	9	15
Bus (line name)	6	3	4	7	15
Tram	8	8	8	9	15
Tram (line name)	8	7	7	9	15
Train	4	0	0	4	9
Train (line name)	4	0	0	4	9
Subway	19	9	17	25	58
Public transport	40	20	29	48	97
Public transport (line type)	39	20	29	47	97

9 Discussion and conclusions

The referenced dataset describes the trips of seven study participants using public transportation in the Helsinki area for a day, and a reference participant using a private car. The dataset includes a manual log of the trips, automatically collected measurements from the mobile devices of the participants and the public transport infrastructure data, which was available from the public transport provider at the time of the test. The mobile device measurements include geographical position and activity estimates. The infrastructure data consists of live locations of public transport vehicles and static timetable data. The challenge is to correctly match as many public transport trips as possible using the various measurements. The results can be verified using the manual log.

The data suffers from multiple imperfections. The estimated activities from the mobile devices are not always correct, e.g. sometimes a passenger is perceived to be riding a bicycle during a tram trip. Due to the power saving features of the mobile device client application and problems in positioning in trains and underground scenarios, measurements can be intermittent or completely missing. Other problems in positioning sometimes result in locations hopping between the correct position and a single distant point. Live locations were not available from all public transport vehicles at the time of the trial.

Altogether 103 trips are logged in the dataset, 97 of them carried out using public transportation. Combining matches from the static timetable and live data 60% of bus and tram trips, and 43–44% of train and subway trips were recognised with the correct vehicle type. Recognition of correct line names was otherwise on the same level, but for buses the recognition result dropped to 47%. The joint combined public transport recognition reached a level of 48% for the correct line type.

The currently achieved results are approximate, but adequate for purposes, where exact recognition of every trip is not necessary. For e.g. public transport disruption information filtering the current recognition level would be sufficient, because the likelihood of a correct recognition increases fast for frequent trips on the same public transport line and the penalty for false positive recognitions is not very high. For purposes where the passenger can review past trips individually the current level of performance can be frustrating. For fare payment processing it would be unacceptable.

Improvements in activity recognition accuracy of the mobile device and better vehicle coverage of live transit data would both contribute to better recognition probability. With the current power saving algorithms the mobile device power consumption is reasonable but clearly noticeable, especially when the device is constantly in motion, causing positioning to be requested with high accuracy. Without radical improvements in battery and / or positioning technologies power consumption should be decreased rather than increased, which sets limitations to future improvements in the client sampling. One approach for testing would be to make more use of radio beacons in public transport vehicles. While in proximity of an identified radio beacon, the high accuracy positioning of the mobile device could be switched off and the system would rely on the positioning service of the public transport vehicle.

References

1. Corsar, D., Edwards, P., Baillie, C., Markovic, M., Papangelis, K., Nelson, J.: GetThere : A Rural Passenger Information System Utilising Linked Data & Citizen Sensing. In: Demo in International Semantic Web Conference 2013. pp. 1–4. Demo in ISWC 2013 (2013), http://www.iswc2013.semanticweb.org/sites/default/files/iswc_demo_22.pdf
2. ECMA-404: The JSON Data Interchange Format (2013), <http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf>, [Accessed March 7th 2017]

3. Feng, T., Timmermans, H.J.P.: Transportation mode recognition using GPS and accelerometer data. *Transportation Research Part C: Emerging Technologies* 37, 118–130 (2013), <http://dx.doi.org/10.1016/j.trc.2013.09.014>
4. Heiskala, M., Jokinen, J.P., Tinnilä, M.: Crowdsensing-based transportation services - An analysis from business model and sustainability viewpoints. *Research in Transportation Business and Management* 18, 38–48 (2016), <http://dx.doi.org/10.1016/j.rtbm.2016.03.006>
5. Hemminki, S., Nurmi, P., Tarkoma, S.: Accelerometer-based transportation mode detection on smartphones. *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems - SenSys '13* pp. 1–14 (2013), <http://dl.acm.org/citation.cfm?doid=2517351.2517367>
6. Hertmann, H., Schaefer, C., Kling, C., Janke, D., Niittylä, L., Chatzilari, E.: D1.2 - Mobile Sensor App with Mining Functionality. Tech. rep. (2014), <http://liveandgov.eu/deliverables/D1.2.pdf>
7. Kabadayi, S., Pridgen, A., Julien, C.: Virtual sensors: Abstracting data from physical sensors. In: *Proceedings - WoWMoM 2006: 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks*. pp. 587–592. IEEE Computer Society Washington (2006), <http://dl.acm.org/citation.cfm?id=1139466>
8. Minnigh, P., Niittylä, L., Agiretxa, M.d.A., Casasempere, J., Perl, J., Thimm, M., Thiele, F.: End results of trials and Live + Gov Methodology. Tech. rep. (2015), <http://liveandgov.eu/deliverables/D5.5.pdf>
9. Nandugudi, A., Ki, T., Nuessle, C., Challen, G.: PocketParker: pocket sourcing parking lot availability. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '14 Adjunct*. pp. 963–973. ACM Press, New York, New York, USA (sep 2014), <http://dl.acm.org/citation.cfm?id=2632048.2632098>
10. Rinne, M.: Towards interoperable traffic data sources. In: *10th ITS European Conference*. Helsinki, Finland (2014), https://research.aalto.fi/files/2574973/Towards_interoperable_traffic_data_sources_final.pdf
11. Rinne, M., Törmä, S.: Mobile crowdsensing of parking space using geofencing and activity recognition. In: *10th ITS European Conference*. Helsinki, Finland (2014), https://research.aalto.fi/files/2574946/Mobile_crowdsensing_of_parking_space_using_geofencing_and_activity_recognition_final.pdf
12. Shafranovich, Y.: RFC4180: Common Format and MIME Type for Comma-Separated Values (CSV) Files (2005), <https://tools.ietf.org/html/rfc4180>, [Accessed March 7th 2017]
13. Shin, D., Aliaga, D., Tunçer, B., Arisona, S.M., Kim, S., Zünd, D., Schmitt, G.: Urban sensing: Using smartphones for transportation mode classification. *Computers, Environment and Urban Systems* 53, 76–86 (2015), <http://dx.doi.org/10.1016/j.compenvurbsys.2014.07.011>