A dataset for determining user preferences of users on personal vehicles

САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

A.A. Borodinov, V.V. Myasnikov

Data collection

We collected data in Samara in a large city with a population of about a million for 6 months from June to December 2019. Nine people of different sex, age, marital status and income, who are employees of Samara University, recorded the tracks of their trips. Users recorded work trips (a trip from home to work and from work to home) in an amount of at least 25 tracks and personal trips (all other trips) in an amount of at least 25 tracks. We define the route from the departure point to the destination point as a trip. In total, users recorded 489 tracks. 338 tracks were recorded on weekdays and 151 tracks were recorded on weekends. The generalized characteristics of the obtained data for all recorded tracks are presented in table.

Data Characteristic	Trip distance	Trip time
Total value	4523 km	183 h 54 min 20 s
Mean	9249 m	22 min 33 s
Median value	5783 m	16 min 35 s
Maximum value	74405 m	2 ч 18 min 50 s
Minimum value	1264 m	2 min 13 s

Figure 1. Collected tracks on a small-scale map

Algorithm for building a track using GPS points

Let $\{\overline{x}_i, t_i\}_{i=\overline{0,I-1}}$ - the data recorded during the trip, where $\overline{x}_i = (x_i, y_i, z_i)$ are the GPS coordinates of the trip, t_i is the recording time of the *i-th* route coordinate.

We describe the road network as a directed graph $G = (V, W)$, where V - is the set of vertices and W - is the set of edges.

Algorithm

- Step 1. For each point $\overline{x}_i, t_i$ we find the nearest edge w and match the point onto the edge.
- Step 2. Consistently look at all the points by K pieces. If all points are less than $\rho_{\min}$, then write them to the result. In the case when the sequence is violated in time and position, we use the algorithm for linking points to a specific path.

After performing step 2, we get the matched sections of the path with gaps as shown in Figure 1. The blue color represents the attached points to the corresponding edges of the graph of the road network.

Next, we consider fragment from i_0 to i_1, i.e. points $\{\overline{x}_{i_0}, \overline{x}_{i_0+1}, \ldots, \overline{x}_{i_1}\}$.

- Step 3. We determine the time interval for each point from $i_0 \to i_1$ to the extreme and determine the appearance physical possibility of this point. If $\frac{\rho(\overline{x}_i, \overline{x}_{i_0})}{t_i - t_{i_0}} > \upsilon_{\max}$ or $\frac{\rho(\overline{x}_{i_1}, \overline{x}_i)}{t_{i_1} - t_i} > \upsilon_{\max}$, then the point is not taken into account and is further considered an outlier.
- Step 4. We define a subgraph from point i_0 to `i_1. We define the shortest path for this $i_0 \to i_1$. After that, we find a point $\overline{x}$ in the center of the shortest path and build a circle with a radius $R = (1+\delta)\cdot\max(\rho(\overline{x}_i, \overline{x}_{i_0}), \rho(\overline{x}_{i_1}, \overline{x}_i))$. In the subgraph we include all the vertices that fall into this circle and the corresponding edges.
- Step 5. We find all the paths without loops in the resulting subgraph between i_0 and i_1. Denote this set P_{i_0,i_1}, where $\forall\, p \in P_{i_0,i_1} : p = (w_{i_0,i*}; w_{i*,\ldots}; \ldots; w_{\ldots,i_1})$.

For each path $p \in P_{i_0,i_1}$ we apply the developed algorithm for matching points to a specific path based on dynamic programming.

Algorithm for matching points to a specific path

for $i = I-1,0$
  for $n = \overline{N-1,0}$
    if $(i == I-1)$
      if $(n == N-1)$
        $\widetilde{\varphi}_i(N-1) = \varphi_i(N-1)$
        $list = new\ List$
        $list.add(N-1)$
        $\pi_i(N-1) = list$
      else
        if $\varphi_i(n) > \widetilde{\varphi}_i(n-1)$
          $list = new\ List$
          $list.add(n)$
          $\pi_i(n) = list$
          $\widetilde{\varphi}_i(n) = \varphi_i(n)$
        else
          $\widetilde{\varphi}_i(n) = \widetilde{\varphi}_i(n-1)$
          $\pi_i(n) = \pi_i(n-1)$
    else // $(i == I-1)$
      if $(n == N-1)$
        $\widetilde{\varphi}_i(N-1) = \varphi_i(N-1) + \widetilde{\varphi}_{i+1}(N-1)$
        $\pi_i(N-1) = \pi_{i-1}(N-1)$
        $\pi_i(N-1).add(N-1)$
      else
        if $\varphi_i(n) + \widetilde{\varphi}_{i+1}(n) > \widetilde{\varphi}_i(n-1)$
          $list = new\ List$
          $\pi_i(n) = list$
          $list = copy(\pi_{i+1}(n))$
          $list.add(n)$
          $\widetilde{\varphi}_i(n) = \varphi_i(n) + \widetilde{\varphi}_{i+1}(n)$
        else
          $\widetilde{\varphi}_i(n) = \widetilde{\varphi}_i(n-1)$
          $\pi_i(n) = \pi_i(n-1)$

Where:

$$\widetilde{\varphi}_j(n) = \max_{n(i):i\le j} \sum_{i=0}^{j} \varphi_i(n(i)),$$

$$\max_{n(i)} \sum_{i=0}^{I-1} \varphi_i(n(i)) = \max_{\underline{n(i_l)=n(i_{l-1}),N}} \begin{bmatrix} \varphi_i(n(i_l)) + \\ \max\limits_{n(i)\le n(i_l)} \sum\limits_{i=0}^{i_l-1} \varphi_i(n(i)) + \\ \max\limits_{n(i)\ge n(i_l)} \sum\limits_{i=i_l-1}^{I-1} \varphi_i(n(i)) \end{bmatrix},$$

$$\varphi_i(n) = \exp(-\alpha \left\| \overline{x}_i - p(n) \right\|^2).$$

The result of the algorithm for matching the track to the road network is shown in Figure 2. The purple line shows the GPS coordinates of the track; the green line shows the matching to the road network.

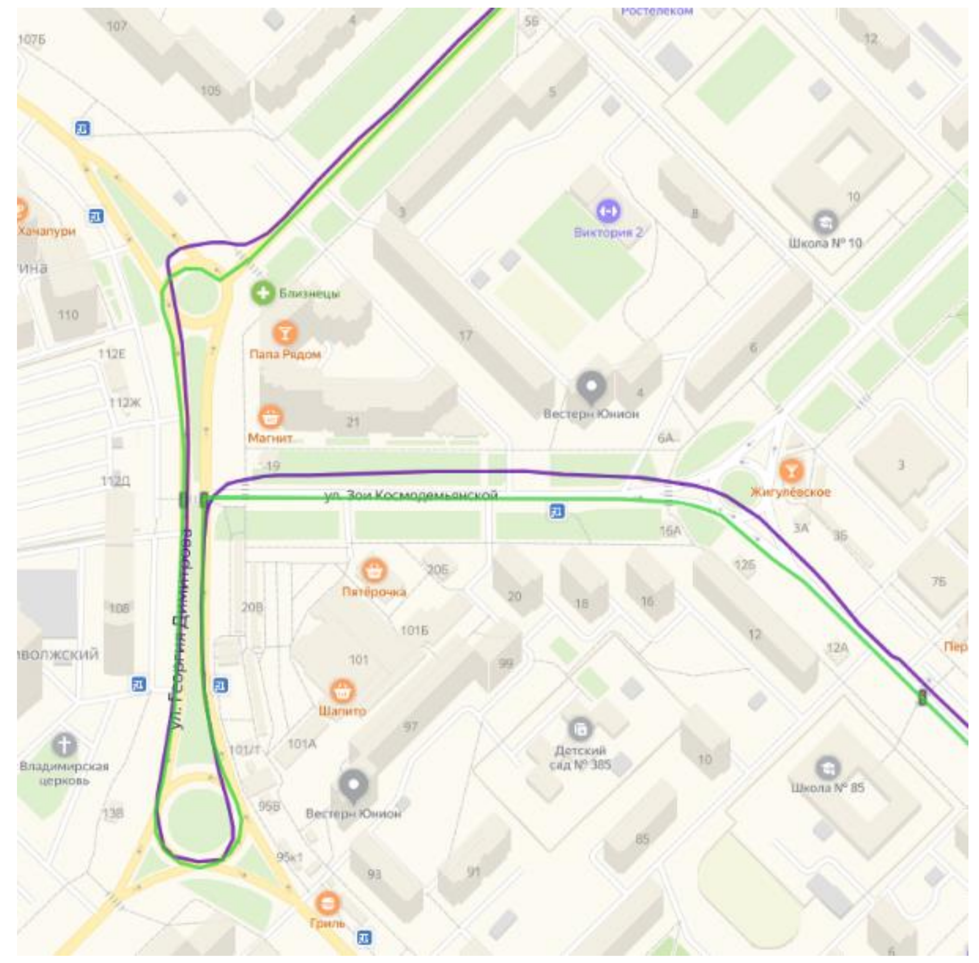

Figure 2. Track matched to the road network (green) and track with raw GPS coordinates (purple)

Acknowledgments

The work was funded by the Ministry of Science and Higher Education of the Russian Federation (unique project identifier RFMEFI57518X0177).

 ALEKSANDR BORODINOV– aaborodinov@yandex.ru VLADISLAV MYASNIKOV – vmyas@geosamara.ru