A Winning Strategy to Park

EU-TP1241

A Winning Strategy to Park

Jean-Sébastien Gonsette[1*], Noémie Meunier[1]

1. AISIN AW - AWTC Europe S.A., Belgium

Tel: +3223892478, e-mail: jean-sebastien.gonsette@awtce.be

19 Avenue de l'Industrie, 1420 Braine l'Alleud, Belgium

Abstract

Parking search is a challenging task for drivers in all big cities. In addition to making the driver loose her time, inefficient search for free parking spots may cause stress, traffic congestion and pollution [1]. In this paper, we propose an algorithm to guide the driver through streets that are at the same time close to her destination, with a good chance of parking, and that are fast to drive through. We also define an algorithm using machine learning techniques in order to measure the availability of parking spots. This algorithm uses collected vehicle traces to produce probabilistic estimates of free parking spots over time. The different algorithms have been evaluated trough simulations and show promising results, which comfort us to continue with field evaluations. All our algorithms can be implemented on distributed architectures.

KEYWORDS:

Available Parking Spaces, Informed Parking Route, Reduce Traffic Congestion

Introduction

AISIN AW, through its division Vehicle Information Technology, is a global leader in Infotainment / ADAS & Connectivity. Since the early age of Navigation, AISIN AW has a successful track of records of innovative technologies, with many world's first. Today, one of our directions is to continue to help the driver by a continuous improvement of its daily routing, and globally to contribute to reduce traffic and CO_2 emissions. This work is definitely an area in line with this direction.

The number of vehicles has significantly increased in urban areas during the last decade. A direct consequence of this expansion is the challenging task for drivers to park in all big cities. The impacts of inefficient search of free parking spots have been studied in [1] for 20 international cities. In addition to drivers who lose their time, the authors highlight the increase of traffic and pollution, along with the stress that this situation can lead to. They reported that about 30 percent of the traffic in big cities is caused by drivers that look for available parking spots. Furthermore, looking for free parking spots is not only a human problem, but is also becoming a concern for tomorrow's autonomous cars.

A part of the lack of efficiency of parking search is due to the lack of information about the availability of parking spots near the driver's destination. Some solutions currently exist to provide real time information about free parking spots, but they are far from being largely available. For instance, some of them either rely on sensors implanted in the vicinity of the parking area [2,3] (e.g. under the pavement or near the parking spot), or depend on the driver car to act as probes for scanning the spots with their sensors [4] (e.g. side ultrasonic sensors). Unfortunately, the two kinds of solutions are still costly and may take time to be generally available. In this work, we take an approach that does not require any sensor in the streets or in the cars, but relies on the availability of driver floating car data. This assumption is not unreasonable given the number of connected automotive applications already available, and the number of connected cars in full growth.

However, the knowledge of the free parking spot probabilities in the streets is only one part of the problem. Indeed, it is still necessary to elaborate a good strategy to drive around the user's destination. This strategy must minimise the expectation of the time needed to park nearby, then to walk right up there. This strategy cannot simply rely on classical routing algorithms for the main reason that, because of the probabilities, there is no deterministic route to compute. For example, the optimal strategy may contain cycles around some streets with high turnover.

Background Many approaches have been deployed for the detection of parking spaces these last years. As a first example, a number of modern private parking garages have sensors that detect in and out coming vehicles. Sensor infrastructures have also been deployed in some cities to provide real-time information about free parking spots. For instance, the SFpark system [3] put sensors in around 7.000 metered parking spots in the city of San Francisco to help adjust the price according to the demand. This product also provides real-time occupancy and pricing information to the users through Internet. The ParkHere system [2] provides real-time information and prediction about free parking spots with the help of sensors installed under the pavement of some cities. A stereovision based approach has also been studied to detect truck parking occupancy [4].

Another approach consists in scanning the free parking spots with the help of sensors embedded into vehicles acting as probes. The solution proposed by the ParkNet system [5] uses vehicles equipped with a GPS receiver and ultrasonic sensors to detect vacant parking spaces. Parking activity can also be monitored thanks to dedicated smartphone applications, as explained in [6], so that this information can be communicated to the drivers. Those applications rely on the phone GPS, accelerometer and Bluetooth sensors to sense the parking events.

Some solutions like ZenPark [7] try to bypass the parking search problem by providing means to book and pay online for precise parking spaces. Other researchers [8] propose algorithms using game theory to choose parking spots ideally with the knowledge of free parking spaces at any times. More recently, an algorithm has been proposed to compute an informed parking route [9]. The algorithm works by maximising a utility function defined thanks to parking information related to the driver. This information includes the current car location, a destination, and parking preferences of the driver like price elasticity, a willingness to walk and so on.

Contributions Many solutions concerning the street parking problem are difficult to scale up because of their cost and the technical means they require. They are often limited to well-defined cities centre and struggle to expand globally. In this paper, we build on the assumption that OEM will increasingly be able to collect anonymized vehicle traces thanks to their connected car fleets. Among other use cases, these traces are well suited to estimate the possibilities to park inside areas dense in traffic.

We first propose a method using collected vehicle traces to estimate the probability to find a free parking spot in each street of a map. Our method uses machine learning techniques to recover the probability distributions over time. In addition, this work also defines a winning strategy built straight upon those probabilities. This strategy guides the driver around the streets near her destination, by favouring those that are fast to ride through and that have good chance of parking. The final routing algorithm is a seamless combination of this winning strategy with a classical routing used to compute an efficient route up to the winning strategy area.

Parking Availability

This section presents the methodology used to compute the distribution of the probability to park over time for the different streets of a map area. We first discuss the method used in this work before showing some results obtained in simulation.

Methodology

At its fundamental level, parking availability in a street evolves according to the arrivals and departures of cars happening in this street. The problem would be simple if we could just count all these departures and arrivals. This is unfortunately not the case and we can only observe a small fraction of all these events to derive a probability model. As a result, this model would ideally deal with the various macroscopic factors that can influence the parking availability probability: time of the day, weekdays, the weather, special events, and so on.

Our model is based on machine learning techniques and is trained on features extracted from a collection of traces in the area of interest. We first explain the general methodology and the features used to train the model, then we present the results obtained in simulation.

Model

Given a street, the problem of determining if 'Yes' or 'No' it holds at least one available parking spot belongs clearly to the class of classification problems. However, recall that it is unrealistic to answer directly this question by just observing a fraction of parking events in this street. Our lack of knowledge makes it more realistic to model a *probability* to find a parking spot, given the street and other features we can easily identify. Our approach is based on the *logistic regression*, with several differences due to the very nature of the problem.

The model processes a collection of vehicle traces that are filtered out in such a way to be relatively confident that the end of a trace corresponds to a parking event. Such event can only be observed if two situations happen at the same time. Of course, there first must be a free parking spot in the street (called event A). Secondly, the driver is willing to park in this street (called event W). These two events are clearly independent, so that the probability to observe a parking event P can be written as:

$$\mathbb{P}(P) = \mathbb{P}(A).\mathbb{P}(W).$$

Our goal is to estimate the probability $\mathbb{P}(A)$ that a parking spot is available on a street. However, we also need to estimate the willingness to park $\mathbb{P}(W)$ since only P is observable. The willingness to park is important in the sense that, knowing that a driver wants to park but does not, infers that no parking spot is available.

An accurate model of those probabilities is not tractable because it would require the knowledge of all the parking events in every street and a complete comprehension of the driver's behaviour. On the contrary, our model takes a macroscopic approach by using a customizable set of features based on the following observations:

- The number of parking spots on a street in a precise period of time depends on the number of parking spots available in the previous period, and on the number of departures and arrivals;
- In a time period, if the street is saturated, a higher activity on the street will increase the chances to find a free parking spot. On the contrary, when the street is far from saturation a higher activity will decrease the probability to have an available parking spot on the street;
- The occupancy rate at the start of the day has a direct influence on the moment when the street saturates;
- If a car travels on a street but does not park on it, it may not necessarily indicate the absence of free parking spots in the street. Indeed it depends on the distance to its final destination;
- Drivers may drive differently when they look for a parking spot, for instance riding slower or irregularly;
- The availability of parking spots: a driver is willing to park further away from its destination when few parking spots are available;
- Weather and driver habits could be other features one could take into account.

Our model takes inspiration of the logistic regression and can infer jointly the parking availability and the willingness to park by processing a set of traces in a given area. The features described above are extracted from these traces and maximum likelihood fitting process is used to recover the different probabilities for the different streets at the different time periods.

Results

Our model has been tested on a synthetic dataset in order to determine up to which it is possible to recover the link probability to park P on the sole observation of partial parking events. The simulation setup involves the creation of a random set of streets associated with different probabilities to park $\mathbb{P}(A)$. Then, we generate a random set of parking attempts. Each one is associated with the probability of the link $\mathbb{P}(A)$ and the willingness probability $\mathbb{P}(W)$ depending on a random goal. Finally, an observable parking event is generated based on the joint probabilities $\mathbb{P}(A)$, $\mathbb{P}(W)$ of each attempt.

Figure 1 illustrates an experiment where the same probabilities are estimated with an increasing number of observations. As we can expect, the larger the dataset, the closer the estimated probabilities $\mathbb{P}(A)$ are to the generated values. It shows that our method successfully converges towards the real distribution. It is also interesting to see the impact of the number of collected parking events on the accuracy of the recovered distribution. Collecting ten events by link is clearly not enough to recover the initial distribution correctly. However, collecting one hundred events by link seems to be a fair compromise to approximate the ground-truth distribution.

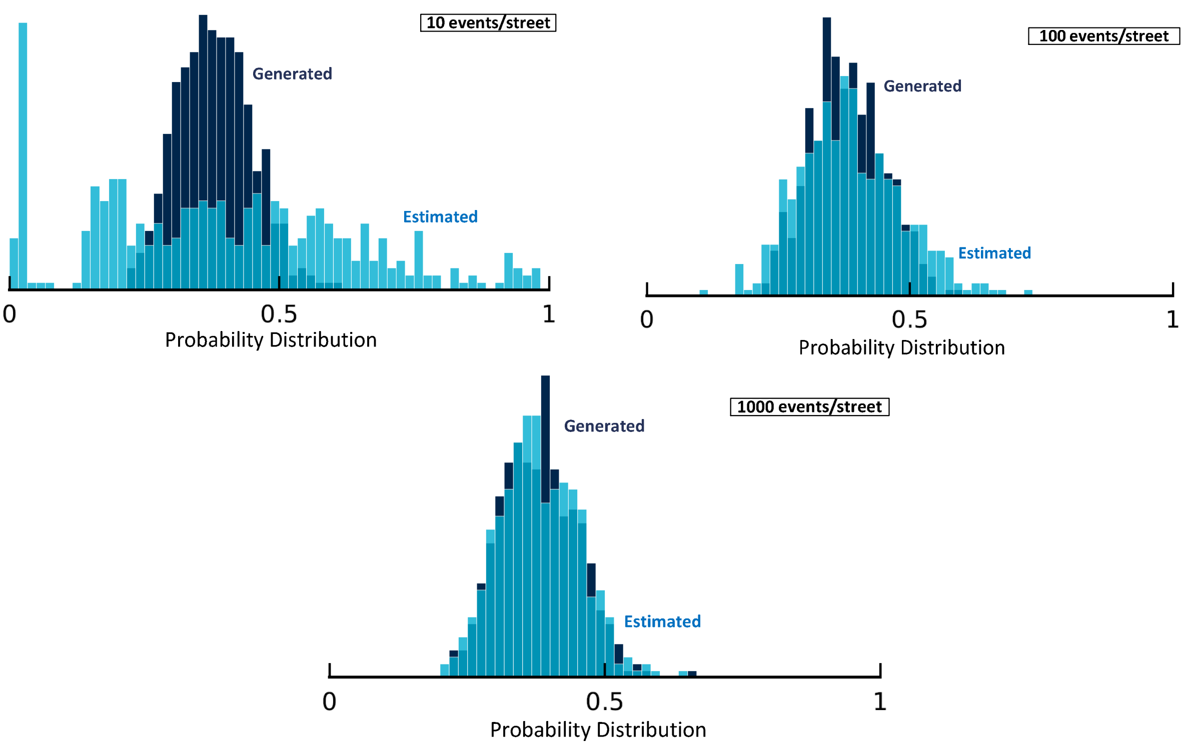

Figure 1: Results of a simulation consisting in estimating the ground truth probability to park $\mathbb{P}(\boldsymbol{A})$ for each street of an area, on the basis of a sub-sample of parking events detection in each street. **Top-left**: 10 events are collected by street and the result clearly diverges from the ground truth distribution. **Top-right**: with 100 events by street the estimated distribution fits well the ground truth distribution. **Below**: 1000 events are collected by street for a nearly perfect match.

A Winning Strategy to Park

In the previous section, we presented a method to estimate the probability to find a free parking spot on each street of a map for a given time period. In this section, we use this information to find a parking informed route. The computed route aims at reaching the driver's destination as fast as possible, starting from the source, with the constraint that the path must lead to a parking opportunity along the way with high probability. We first discuss the method used in this work before showing some results obtained in simulation.

Methodology

The total trip duration of a driver up to her destination is clearly given by the addition of the time to reach a parking spot along her route with the time needed to walk from this parking spot to her final destination. It is precisely this time that we want to minimise in this work.

However, this work is based on the assumption that we do not know in advance the actual availability of the parking spots along the streets, but only the probabilities to find a free one (see previous section). As a result, the route we want to compute is the one minimising the expected time to reach the destination from the driver's location, with the constraint that the car is parked along the way.

A naïve brute force approach consisting in computing the expected time of all the possible routes for different lengths joining the driver's location to any point around the destination is computationally infeasible. Indeed, the problem scales exponentially with the length of the routes to be examined. An additional problem comes from the fact that asking the driver to park as soon as she finds an empty spot might not be interesting if she is at the beginning of a long route. Indeed, the choice to park or not really depends on what it could expect by driving a little bit longer towards her destination. Everyone has already experience this dilemma: "should I take this empty spot now and walk more, or try my luck a bit further?". Our goal in the next section is thus twofold. In addition to the efficient computation of the driver's route, we also want to inform the driver if she should wait a bit longer before to park in an empty spot.

Isochrones computation

Let us imagine that, for each street of the driver's route, we know the *expected time* needed to reach the driver's destination. As explained above, this time includes both the time to drive to a parking spot and the time needed to walk from there to the final destination. In addition, this time is an expectation, which means that it is correct in *average*. This expected time is thus a prediction of the future for all the next streets of the route. Then, answering the question "should I park or not?" is simple. A driver should try to park only if the expected time of the current street is greater than the time needed to walk to the destination from this very street. The reason is that continuing to seek a spot further is likely to take more time than just to stop and walk.

Intuitively, the expected time E_s for each street s of a route is computed thanks to the two following observations. With probability P_s the car parks on street s and the driver reaches the destination by walking with time W_s. Else, with probability $(1 - P_s)$, the car does not park on street s and the driver will continue driving on this street s with time R_s. In addition, the expected time of the next street in the route must be credited to R_s. With the previous notations, this intuition can be recursively formulated as follows. Given a route defined by a finite sequence of streets $r = s_0 s_1 \dots s_N$, then for each street s_n with $0 \leq n < N$, the expected time of s_n is given by:

$$E_{s_n} = P_{s_n}.W_{s_n} + \left(1 - P_{s_n}\right).\left(R_{s_n} + E_{s_{n+1}}\right).$$

We defined an algorithm that can solve this recurrence for all the streets of a map area. For a time period and a destination being fixed, the algorithm initializes the expected time of all the streets to an upper bound. In addition, it computes the time needed to reach the destination by walking from every street in the area. This computation can be done with a standard Dijkstra algorithm but it is interesting to do it on a pedestrian data base to take into account paths that are forbidden for cars. Finally, if traffic information is available, it is used to initialize the time needed to travel each street by car.

Then, the recurrence is solved with an iterative converging process. For each iteration, the expected time of each street is potentially decreased by examining all its directly reachable neighbouring streets. The recurrence formula above is used to compute the new expected time to destination, knowing what is expected according to the neighbours. The algorithm stops as soon as it stabilises around a fixed point.

The output of this algorithm is thus the initial map of the streets, where each street is augmented with an expected time to reach the destination. It can be seen as a 3D map where peaks represent areas to avoid, and where valleys are area to climb down to maximise the chances to find a free spot. We call this map an *isochrone map* because all the links that are at the same level have the same expected time to reach the destination. In practice the algorithm converges very fast, in less than 20 iterations. Given that the area of interest around the destination is always limited, the computational cost of this method is virtually non-existent.

Besides giving an expected time to the driver's destination for each street, the isochrone map directly provides a *Winning Strategy routing* method, computing an informed route from each street as follows. Given a starting street, the next street to go next is just the reachable one that has the lowest expected time to destination. In addition, indicating if the driver should try to park or not in the street is determined by comparing the expected time of the street with the walk time of the street.

Integration inside a complete strategy

Ideally, the global strategy to go from one location to a given destination must include both the strategy to maximise the parking chances around the destination, but also a traditional approach to compute a potentially long route up to this area. For instance, the problem consists in determining the border where the traditional routing stops and where the Winning Strategy routing described in the previous section begins. In addition, if we use a traditional routing to compute the first part of the journey, what is the destination to set for? The problem is thus to combine the two approaches seamlessly and in an efficient way.

To combine the methods and to answer the question above, the proposed method is as follows. First, a suitable area is defined around the driver's destination. The size of this area is not too much important. It can be easily set with an upper bound related to the maximum distance the driver can accept to walk. Then, the isochrone map is computed for this area according to the method described in the previous section. In addition to the expected times, the algorithm also provides a flag telling what streets are interesting to park or not.

Knowing those flags, we remove from the map all the links corresponding to streets where it is interesting to park. This operation has for effect to create a hole around the destination. During this operation, we identify all the links that see one of their neighbouring links disappear. All these links correspond to potential *entry points* in the area where the Winning Strategy routing will start. As a result, they are potential destinations for the traditional long distance routing algorithm. Each potential entry point is then directly re-connected to the destination with a virtual link whose travel time is the expected travel time of the source street. Therefore, all these virtual links have for effect to provide a graph suitable for the classical routing algorithm and that also contains the driver's destination. Finally, it is just a matter of running the classical routing algorithm on this graph to naturally choose the best entry point in the winning strategy area, according to the driver's starting location. This entry point is the one that minimises the sum of the deterministic time to drive up to the parking strategy area with the average time to park expected from there.

Results

A plot of an isochrone map is displayed in Figure 2 for the Munich city centre. The map is computed for the destination symbolised by the green circle and with synthetic probabilities to park along the streets.

A Winning Strategy to Park

Streets are coloured according to the expected time it would need to reach the destination from there, with the car parked. We see that the more we move away from the destination, the more the expected time is high. However, the individual street probabilities distort this tendency by pulling the expected time upwards or downwards.

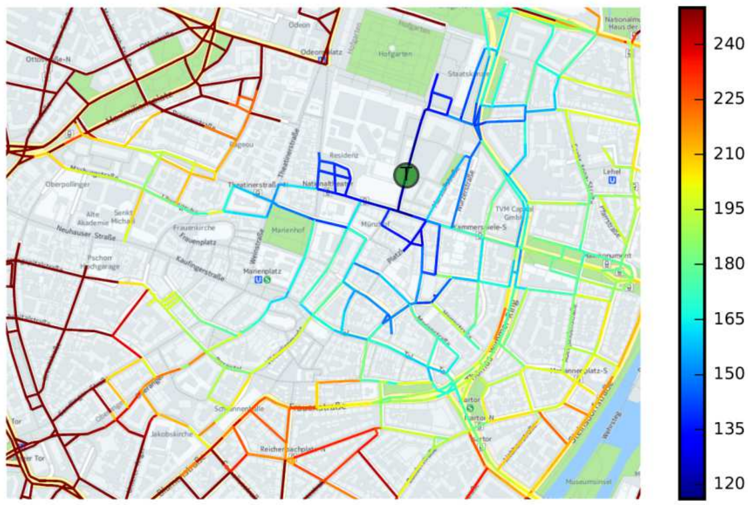

Figure 2: Example of isochrone map drawn on the Munich city centre and based on synthetic probabilities. The target destination is symbolized by the green circle. The color scale indicates the expected time to reach the destination with the car parked, in seconds.

Our evaluation of the Winning Strategy routing is based on the comparison of the performances with a simple agent that drives straight to the destination, then tries to park as soon as possible while driving randomly around. Pairs of departure and destination points are randomly generated and the same simulation is done several times on each pair, in order to take the average of the measures. Each measure corresponds to the total time needed to park the car and then walk up to the destination. As a result, we obtain for each pair an average time for the simple agent to park and for the Winning Strategy routing to park. Figure 3 shows the results we get from this simulation. Each dot corresponds to a departure/destination pair. The abscissa coordinate of a dot indicates the time needed for the simple agent to park, while the ordinate coordinate corresponds to the time for the Winning Strategy routing to park. The blue line corresponds to the equation y=x and simply delimits the area where the Winning Strategy routing is better (below) from the area where the simple agent wins (above).

Conclusion

Parking in highly urbanized areas is a daily challenge for many people and it has a direct impact on other factors like traffic, pollution, stress, etc. Unfortunately, urban areas are increasingly populated and this problem is very likely to accentuate if solutions are not proposed to mitigate it. Besides other mobility solutions aimed at reducing the traffic inside city centres, other approaches aimed at optimizing drivers parking experience have also their role to play. As with many things, the first step towards improving a process requires to measure what really happens in the inside. Our approach is built on the hypothesis that collection of driver's traces are available in order to predict historical probabilities to park on street at different days of the week and different hours of the day. Prediction of probabilities are based on features that could be extended to sill improve the results. Then, we continue further by using these probabilities to compute the so called isochrone map. This map provides an efficient solution to compute a route going around the destination while trying to minimise the total time to find a parking spot and walk up to the destination. Finally, the Winning strategy is seamlessly integrated with our traditional routing algorithm in

A Winning Strategy to Park

order to provide a full solution to propose to the driver. Because of the promising results we obtain in simulation, we will now continue our evaluation with field validations.

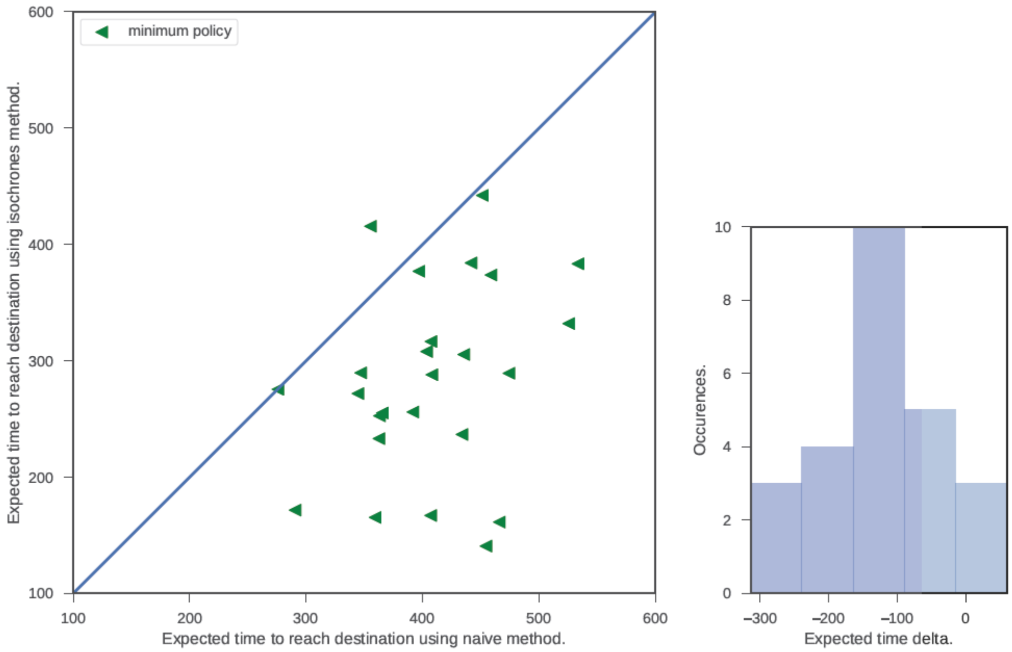

Figure 3: Evaluation of the isochrone based Winning Strategy routing by comparing the time needed to park with a simple agent that provides a base line for the comparison.

References

1. IBM (2011). IBM Global Parking Survey: Drivers Share Worldwide Parking Woes, *https://www-03.ibm.com/press/us/en/pressrelease/35515.wss*, Accessed February, 2017.
2. ParkHere. *http://park-here.eu/en/home/,* Accessed February, 2017.
3. SFPark. *http://sfpark.org/,* Accessed February, 2017.
4. Deruytter, M., K. Maddelein, W. Favoreel, D. Tsishkou, S. Bougnoux, R. Bendahan (2012). Stereovision truck parking occupancy detection, In *19th ITS World Congress*.
5. Mathur, S., T. Jin, N. Kasturirangan, J. Chandrasekaran, W. Xue, M. Gruteser, W. Trappe (2010). Parknet: drive-by sensing of road-side parking statistics, In *Proceedings of the 8th international conference on Mobile systems, applications, and services,* pp.123-136.
6. Stenneth, L., O. Wolfson, B. Xu, P.S. Yu (2012). PhonePark: Street parking using mobile phones, In *13th International Conference on Mobile Data Management (MDM),* pp. 278-279.
7. Zenpark. *https://zenpark.com/*, Accessed February, 2017.
8. Ayala, D., O. Wolfson, B. Xu, B. Dasgupta, J. Lin (2011). Parking slot assignment games, In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 299-308.
9. Djuric, N., M. Grbovic, S. Vucetic (2016). ParkAssistant: An Algorithm for Guiding a Car to a Parking Spot, In *Transportation Research Board 95th Annual Meeting*.