Recognition and Recommendation of Parking Places

Andrew Koster, Allysson Oliveira, Orlando Volpato, Viviane Delvequio and Fernando Koch

SAMSUNG Research Institute
{andrew.k, allysson.o, orlando.f, v.franco, fernando.koch}@samsung.com

Abstract. Current solutions to recommend available parking spaces rely on options like: intentional user feedback; installing data collectors in volunteering fleet vehicles, or; installing static sensors to monitor available parking spaces. In this paper we propose a solution based application that runs on commodity smartphones and makes use of the advanced sensor capabilities in these devices, along with methods of statistical analysis of the collected sensor data to provide useful recommendations. We exploit a combination of k-medoid clustering and Conditional Random Fields to reliably detect a user parking with a limited sensor capability. Next, we outline a method based on Markov Chains to calculate the probability of finding a parking space near a given location. We also enhance the solution with more sensor capability to discover desirable properties in parking spaces.

1 Introduction

Finding parking spaces in big cities is aggravating. Reports indicate that from 30% to 40% of light vehicle traffic is made up of drivers actually searching for a free parking space [3]. This represents a lot of wasted time and fuel, and increased emissions of pollutant and green-house gases.

Current solutions to indicate available parking spaces rely on orthodox approaches like: a) intentional user feedback, i.e. tapping a button to indicate an available parking space; b) installing sensors, and its corresponding processing equipment in volunteering fleet vehicles, like taxi drivers, to detect available parking spaces; or c) installing sensors and/or cameras on the parking lots with an associated system to process those signals.

We introduce an alternative solution consisting of a mobile app to automatically detect that the user is parking, and a server-based system to recommend available parking spaces. This solution is grounded on the concept of serendipitous recommendation: that is, providing useful and pleasant information based on automatically collected environmental data and some form of understanding of users' demands. Exemplary solutions are SAMSUNG S-Health[1] and Google Now[2].

[1] http://content.samsung.com/us/contents/aboutn/sHealthIntro.do
[2] http;//www.google.com/landing/now/

In our case, we developed new models of statistical analysis of smartphone's sensor data to detect salient movement signature such as "leaving a parking space" or "entering a park space". The mobile app performs statistical analysis upon sensor data and, once these signatures are detected, notifies the Parking Space Recommender server. The new information is stored in a repository, which in turn is used to calculate the probability of available spots, when this is requested. Simultaneously the recognized movement signature is evaluated in the app, to improve the recognition algorithm, and adapt it to better function in the user's context. By collecting historic parking information, contextualized by geographic region, calendar day, time of day, weather condition, ephemerides, real time traffic information and special events; a model can be built to predict free parking spaces generation and consumption rates.

Finally, we describe how our app presents the information about a parking space: along with the information about availability of spaces, we collect information about the desirability of these spaces. For instance, how far is it from the user's destination, does it have shade, or a street lamp, and any additional problems in the area. In Figure 1 we give an overview of how the system works. In the following, we describe the various computational methods used to achieve this.

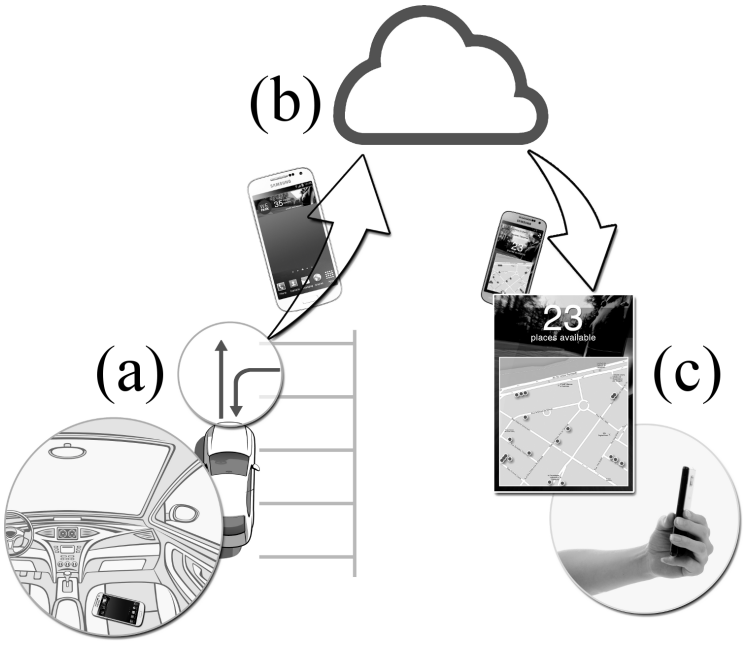


Fig. 1. Functional overview: (a) movements are automatically recognised and (b) notifications sent to a cloud-based service, which groups and classifies information from many devices; (c) the service is able to identify nearby available parking spaces to reply to users' geolocated requests.

2 Related Work

There are many systems envisioned to inform and manage available parking spaces in a context of smart city. Generally these require the installation of sensors as part of a city's infrastructure. Some of them are based on installing sensors on every parking space to detect directly whether there is a parked vehicle or not [10,13], which places a large demand on standardisation of sensors, data collection and processing equipment. Moreover, it demands a large monetary investment on procurement, installation and maintenance.

Other solutions use a fleet of voluntary vehicles to detect parking occupancy. Suhr et al. [16] proposes a free parking space detection system by using a fish-eye camera installed on the rear end of a vehicle to detect available spots when a user drives by them. Mathur et al. [9] describe a system for detection of available parking spaces by having a fleet of vehicles equipped with ultrasonic sensors in the vehicle's passenger door, which detects available spaces on the street. These solutions demand the use of dedicated equipment to be installed on vehicles that should be continuously passing on most of the streets. They are intended mostly for detecting street parking, as the vehicles are not meant to go to inside parking lots. However, these solution imply capital expenditure and continuous operation costs to fleets of vehicles such as taxi or buses, who are not the primary consumers of the product.

Rather than explicitly instrumenting the city, some solutions envision that the citizens report free parking spaces. Some require intentional action by the user, for instance, by tapping a button on their mobile phone's application indicating that they parked, or just left, a parking space [4,6]. Google's "Open Spot" application [4] continuously shows vacant parking spaces in a 1 mile radius of the user's location. The availability of parking spaces should be indicated as the user enters or leaves a space. This application did not become popular because it required the intentional action of the user, despite a system of points intended to incentivise the use of the system [12]. Instead, the recognition of a parking space should be automatic, as suggested in "PhonePark" [15], which uses, in addition to other things, GPS location and the connection/disconnection of the Smartphone with the Bluetooth based hands free system of the vehicle. The main disadvantage of this approach is that not all vehicles have a hands free system, especially in Brazil, where most of the fleet is composed of low technology content vehicles. For recognizing parking spaces we use a system similar to the Phonepark System, but make less assumptions on the available technology. In order to recognize a parking space, we require an accelerometer, a gyroscope and some method of sensing location (whether it be through triangulation in the cellphone grid, or GPS). However, our system builds upon these reports by using statistical machine learning techniques to better report the available parking spaces.

3 Detecting Parking Spaces

Whilst various methods are able to distinguish between different activities [7,14], they are not sufficiently precise to recognise parking movements. For instance, in commercially available software, the SAMSUNG Motion package in the Mobile SDK[3] recognises an elevator as a vehicle, which is clearly a problem: if we were to use the mode change from walking to vehicular transport and vice versa to recognise parking spaces, this would result in a large number of false positives. A second problem with the state-of-the-art is that they rely on labelled data. While this is fairly easy to obtain for rough categories like transportation mode, more detailed activities are harder to label manually, and of the data we have obtained, only a small part is unreliably labelled, whereas the rest is entirely unlabelled.

In order to deal with these shortcomings, we propose to combine the recognition of rough categories, such as changes in transportation mode, with what we call "parking signatures". In Figure 2 we have graphed a typical signature we captured for a park-out manoeuvre, reconstructed from accelerometer and gyroscope data. The motion is normalised to start at (0, 0) in the north direction. It is easy to see that the car reversed out, turned, and drove away. While not all moves are as easy to recognise visually, all park-out manoeuvres have, as a bare minimum, in common that they start from standstill, and accelerate in the XY-plane. Nevertheless, this description also includes other movements, such as starting from a traffic light, and if we do not take initial velocity into account (which must be computed, rather than sensed directly) these signatures may be confused with a host of other signatures, such as going around a corner.

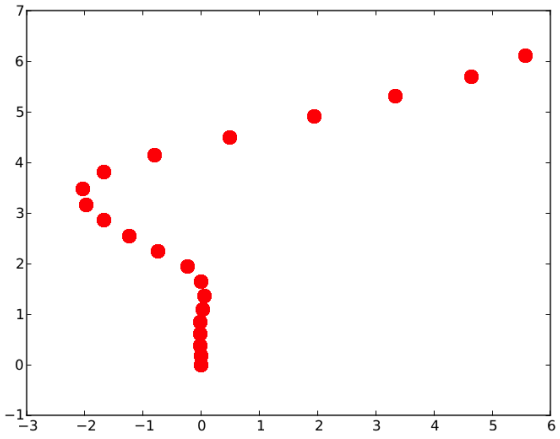


Fig. 2. Reconstruction of the car's position in the XY-plane over time.

[3] http://developer.samsung.com/samsung-mobile-sdk

Rather than attempting to describe all potential movements manually in a rule-based system, we use machine learning. In specific, we use k-medoids [5], an unsupervised learning algorithm, to cluster a set of mostly unlabelled sensor readings, collected by volunteer motorists. As a distance measure we use Dynamic Time Warping (DTW) [11], a well-known distance measure for time series data that, despite its age, is still among the most accurate measures [1].

While there are other learning algorithms that work with unlabelled data, the most common are clustering algorithms. These algorithms aim to group together similar samples, while maintaining the groupings themselves as distinct as possible. Out of the possible clustering algorithms we choose to use k-medoids, because almost all other algorithms rely on a method for calculating a central point for each cluster. This is a notoriously hard problem for time series data, and an initial experiment using k-means and the arithmetic mean of the points in a cluster led to unsatisfactory results: often the algorithm would not even converge. k-medoids avoids entirely the calculation of a central point by choosing the most central of the data points as the representative of the cluster.

Based on preliminary research, we found that the best results are obtained with $k = 50$, with a Davies-Bouldin index of 0.066 (in comparison, with just 5 clusters, the best Davies-Bouldin index obtained was 0.209). The labelled samples we have serve to label each cluster as containing park-in manoeuvres, park-out manoeuvres or other manoeuvres. The result is a large collection of *labelled* movement signatures. These, in turn are used as input to train a Conditional Random Field (CRF) [8] model to recognise new series of sensor readings in real-time. By using cross-validation, we can test the CRF part, however this does not truly test whether the label assignment by the clustering algorithm was performed correctly. As such, sufficiently testing the classifier requires the composition of a set of labelled data, and will be performed when the parking app goes into beta testing: it will include a button to give feedback about whether a manoeuvre was recognised correctly by the algorithm, thus providing labelled samples.

4 Reporting Probable Spaces

Detected parking spaces are reported by the app to the server, which collects both current, and historic data about parking spaces. When an app requests information about available spaces, it reports the central location, around which a space should be found. This can be the user's current location, or anticipating the user travelling, his or her destination. The server performs two computations. Based on historic data, and using a Bayesian approach, it computes the distance around the point for which the probability of finding a parking spot at the given time is sufficiently high. Secondly, for the actual known spaces that are within the calculated distance, the probability of them in fact being free, given their last reported status, at the given time is calculated using a Hidden Markov Model.

4.1 Calculating parking distance

The historic data that is collected on the server can be interpreted as a time series for each individual parking space. In an ideal scenario, the sensor readings would be perfect, and would give a record of when any parking space was occupied or not, and we could use this to calculate the probability of the space being available, based on its availability at similar times in the past. However, in reality, the data will be very sparse, and we will be missing data for many parking spaces, and for those we do have data it may be updated infrequently: not everybody (or even most people) use the app, and even those that do may switch it off. It is thus important to assume (very) incomplete information regarding the availability of parking spaces in an area.

We solve this, by calculating a distance threshold, for which the expectation of finding a parking space is above a configurable probability threshold. Algorithm 1 computes this. For each distance, the algorithm collects all the historic sensor readings within range, and calculates for each location the probability that it is occupied using the beta probability distribution, and a hill-climbing method is used to find the optimum distance. For any location (or parking space), we generate the entire time series, by simply assuming that each reading is the status for the entire time period until the next reading. We discretise time and use the maximum likelihood estimation to calculate the parameters of a beta probability distribution for whether or not a position is occupied. Due to the law of large numbers, with sufficient readings, the errors from having sparse data will average out; hence the need to average over an area, rather than considering each parking space individually.

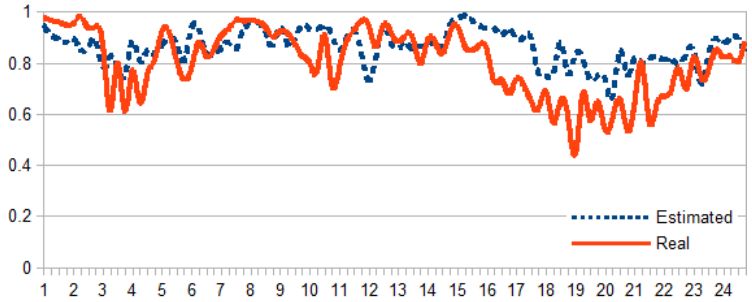


Fig. 3. Probabilities of a parking space being free.

We tested the efficacy of the estimator of availability using the data set from the NYC Department of Traffic [4], averaged over 15 minute time windows during a day. The results are plotted in Fig. 3 and have a Pearson correlation coefficient of 0.57.

[4] http://www.nyc.gov/html/dot/html/motorist/realtimeparking.shtml

Algorithm 1: ParkingDistance

Input: Historic sensor readings of parking spaces, central location
Input: Probability threshold that is required
Result: Minimum distance from central location within which there is an available parking space with probability above the threshold

```
tested ⟵ ∅
distance ⟵ initialdistance
repeat
    tested ⟵ tested ∪ {distance}
    Readings ⟵ getDataWithinRange(distance)
    probability ⟵ 1.0
    for location ∈ getLocations(Readings) do
        probOccupied ⟵
                BetaDistrib(Readings.at(location))
        probability ⟵ probability * probOccupied
    if probability > threshold then
        distance ⟵ getNearer(distance)
    else
        distance ⟵ getFarther(distance)
until (1 − probability) > threshold ∧ distance ∈ tested
```

4.2 Calculating parking space probability

After finding a region for which the probability is sufficiently high, we can further help the user by computing the actual probability of the known parking spaces being available, given their last known status. For this, we use a Hidden Markov Model (HMM) [2] to model the conditional probability that a space is occupied, given its last known status, and the time of day. To learn the HMM, we consider all spaces, and thus all readings within the area as sufficiently similar, in order to give sufficient data for the parameter estimation. For the time-of-day we consider 15-minute intervals, and for the time since the last report, we consider anything longer than 30 minutes previous as "long ago". This gives us a space of 60×96 possible values for the observable signals, and 2 possible labels (occupied, or free) for the target. An example of a data point is:

(reported occupied 23 minutes ago, 13:00-13:15) $\rightarrow$ **occupied**.

The parking spaces with a sufficient probability of being free are sent to the app for reporting to the user.

5 Recommending Desirable Spaces

Finally, the mobile app provides parking recommendations using the data above. Available parking spaces are colour coded according to the estimated probability of them being free, and upon selecting any one of them, additional information is displayed: the estimated traffic density, and if a destination is provided, the time to walk there from the parking space is given. Note that this destination is

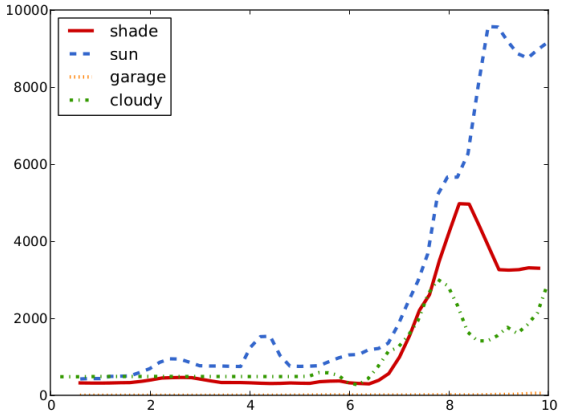


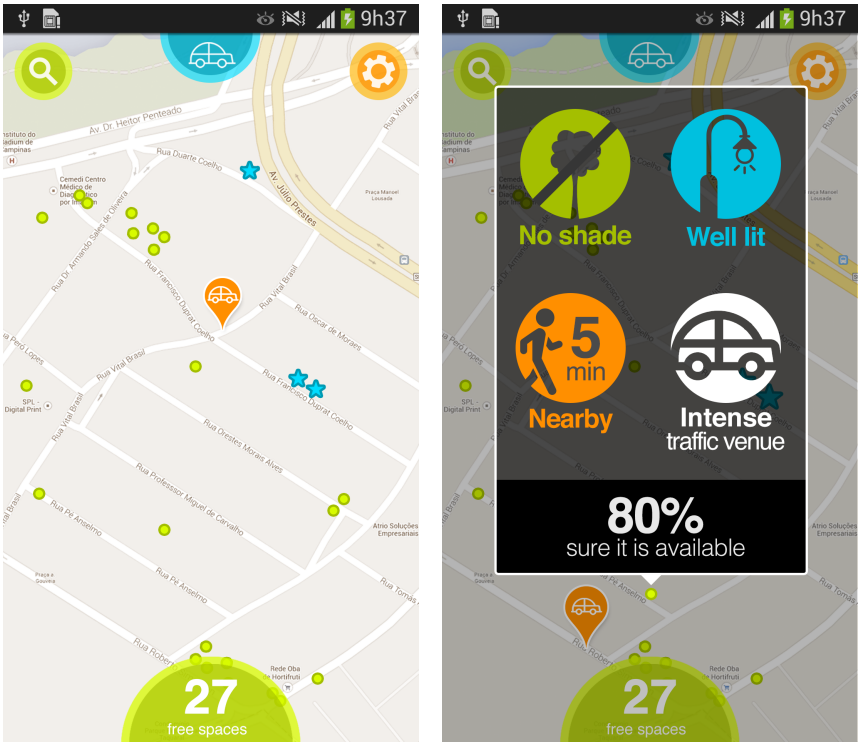
Fig. 4. Light, in lumens, upon exiting the car when parked in varying situations.

not necessarily input by the user, but can be inferred from historical data about the user's behaviour, or extracted from his calendar.

Moreover, the system learns whether a parking space has shade during the day and light at night. As with the rest of the app, this data is also collected without needing any action from the user beyond normal use of his or her smartphone. When a parking signature is detected, as described in Section 2, the smartphone's photometer is activated. A rule-based system is used to detect whether the spot is shady or sunny during the day, and illuminated or dark during night-time. In Fig. 4 we graphed the luminosity readings for exiting the car in a sunny spot, and exiting the car in a shady spot. While the rules require more sophistication, depending on the time-of-day, weather, season, and location to estimate how big a jump in luminosity triggers a positive reading for either sunny or shady, the graph shows how to go about it. The rule-based system can additionally fuse data from other sensors; for instance, by using the proximity sensor to distinguish between dark of night and it being dark because the mobile phone is in the user's pocket.

This information is sent to the server, and maximum likelihood estimation is used to obtain the most likely situation. When searching for a spot, this situation is displayed. Screenshots of the mobile app are in Fig. 5.

We apply special interface marks to represent different park space categories, e.g. green dots for regular spaces and blue stars for recommended spaces. Currently we just choose the most likely to be free, but an extension of the app is to learn from user behaviour, and recommend spaces according to his or her preferences: for instance, if the user prefers to park slightly further away, but in a shady spot, this can be taken into account.



(a) Map display of the available parking spots, and the number of parking spots within range.

(b) Display with details for the selected parking spot.

Fig. 5. Screenshots of the parking recommender app.

6 Conclusion and Future Work

In this paper we present a serendipitous solution to finding parking spaces: our vision is that this app works seamlessly with the user to collect and provide information when required without intruding or interrupting his or her day-to-day rhythm. The information is collected automatically when a park-in or park-out event is recognized using the activity recognition method we presented in Section 3, it is processed intelligently to calculate, as accurately as possible, the availability of parking spaces, and is reported to users who are detected to be searching, or will soon be needing, a parking spot.

The system is currently entering the last stages of development, and will soon enter alpha testing, when we will extensively test the algorithms developed in internal experiments. In particular this will serve to fine-tune our recognition algorithm and start populating our repository of historical parking data. While our current focus is on rolling out this service as fast as possible, in order to collect more data and better train our algorithms, we realise that the usability of the app should be at the forefront. Therefore there are already a number of improvements planned for its use in a car.

Because the app is meant to be used in a car, it is necessary to minimise the need to fiddle with the phone. The app will respond to voice commands, and in addition to the map interface of Fig. 5a, it will output suggestions and directions using spoken language. Additionally, we intend to use a similar system to the one outlined in Section 3 to predict that a user is looking for a parking space. Search patterns are fairly typical and we expect that a similar approach will allow us to recognise these. Furthermore, if the user's routines are known, or he has saved appointment locations in his agenda, we can use this information to predict when and where a user will search for a space. Any one of these approaches will trigger the app to ask the user if he is indeed in need of a free space (in spoken language), and upon confirmation, display the available spaces and guide the user to the best one. In this scenario, the user does not have to touch his phone, or even look at it, to get the desired functionality.

Moreover, the method we use to calculate the expected distance for finding a parking space in Section 4.1 can be used in anticipation to help users to plan for future decisions. The service can estimate how many free spaces will be available for an area, given a time period and date. When users are planning their day, they can obtain information about the availability of parking. For instance, they might have their lunch break slightly earlier to avoid parking hassles near a restaurant, or take such information into account when planning meetings. Additionally, city planners could use this information to easily see where parking problems are greatest and plan new projects accordingly.

As next steps, we are looking at reinforcement learning techniques [17] to adapt the general model described in Section 3 to detect a user's parking signature. Every person has different ways of doing things, and this is no different for driving. We thus expect that by adapting the parameters of the learned graphical model to the user's behaviour will lead to a more accurate classifier.

References

1. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: experimental comparison of representations and distance measures. Proceedings of the VLDB Endowment **1**(2) (2008) 1542–1552
2. Elliott, R.J., Aggoun, L., Moore, J.B.: Hidden Markov Models. Springer, Heidelberg, DE (1995)
3. Gallivan, S.D.: IBM global parking survey: Drivers share worldwide parking woes. Technical report, IBM (2011)
4. Hildenbrand, J.: Google releases open spot for android — find and share parking (July 10 2010) http://www.androidcentral.com/googl-releases-open-spot-android-find-and-share-parking retrieved June 24, 2013.
5. Kaufman, L., Rousseeuw, P.: Clustering by means of medoids. In: Statistical Data Analysis Based on the L1 Norm and Related Methods. Springer, Amsterdam, NL (1987) 405–416
6. Koster, A., Koch, F., Bazzan, A.L.C.: Incentivising crowdsourced parking solutions. In Nin, J., Villatoro, D., eds.: Citizen in Sensor Networks (CITISEN'14). Volume 8313 of LNAI. Springer (2014) 36–43
7. Kwapisz, J.R., Weiss, G.M., Moore, S.A.: Activity recognition using cell phone accelerometers. ACM SIGKDD Explorations Newsletter **12**(2) (2010) 74–82
8. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML, San Francisco, CA, USA, Morgan Kaufmann (2001) 282–289
9. Mathur, S., Tong, J., Kasturirangan, N., Chandrasekaran, J., Xue, W., Gruteser, M., Trappe, W.: ParkNet: drive-by sensing of road-side parking statistics. In: Proc. of MobiSys '10, ACM (2010) 123–136
10. Park, W.J., Kim, B.S., Kim, D.S., Lee, K.H.: Parking space detection using ultrasonic sensor in parking assistance system. In: proc. of the IEEE Intelligent VEhicles Symposium, IEEE (2008) 1039–1044
11. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech and Signal Processing **26**(1) (1978) 43–49
12. Sherwin, I.: Google Labs' Open Spot: A useful application that no one uses (July 10 2011) http://www.androidauthority.com/google-labs-open-spot-a-useful-application-that-no-one-uses-15186/ retrieved May 15, 2014.
13. Srikanth, S., Pramod, P.J., Dileep, K.P., Tapas, S., Patil, M.U., Sarat, C.B.N.: Design and implementation of a prototype smart PARKing (SPARK) system using wireless sensor networks. In: Proceedings of the Advanced Information Networking and Applications Workshops (WAINA '09). (2009) 401–406
14. Stenneth, L., Wolfson, O., Xu, B., Yu, P.S.: Transportation mode detection using mobile phones and GIS information. In: Proc. of the 19th ACM SIGSPATIAL International Converence on Advances in Geographic Information Systems, ACM (2011) 54–63
15. Stenneth, L., Wolfson, O., Xu, B., Yu, P.S.: PhonePark: Street parking using mobile phones. In: Proceedings of the 13th IEEE International Conference on Mobile Data Management (MDM '12), IEEE (2012) 278–279
16. Suhr, J.K., Jung, H.G., Bae, K., Kim, J.: Automatic free parking space detection by using motion stereo-based 3D reconstruction. Machine Vision and Applicatons **21**(2) (2010) 163–176
17. Sutton, R.S., Barto, A.: Reinforcement Learning: An Introduction. MIT Press (1998)