Multi-agent learning for energy-aware placement of autonomous vehicles

Ömer Ibrahim Erduran
Goethe University Frankfurt
Business Information Systems
Frankfurt am Main, Germany
erduran@cs.uni-frankfurt.de

Mirjam Minor
Goethe University Frankfurt
Business Information Systems
Frankfurt am Main, Germany
minor@cs.uni-frankfurt.de

Lars Hedrich
Goethe University Frankfurt
Electronic Design Methodology
Frankfurt am Main, Germany
hedrich@em.cs.uni-frankfurt.de

Ahmad Tarraf
Goethe University Frankfurt
Electronic Design Methodology
Frankfurt am Main, Germany
tarraf@em.cs.uni-frankfurt.de

Frederik Rühl
Continental Teves AG & Co. oHG
Future Mobility Concepts
Frankfurt am Main, Germany
Frederik.Ruehl@continental-corporation.com

Hans Schroth
Continental Teves AG & Co. oHG
Future Mobility Concepts
Frankfurt am Main, Germany
hans.schroth@continental-corporation.com

***Abstract*—Mobility gets an increasing amount of meaning and significance in the modern society. In this paper, we introduce a multi-agent learning application for a multi-agent system in e-mobility. In particular, we propose a geospatial model for free-floating and autonomously driving e-trikes and demonstrate a calculation method of positioning e-trikes on a given area by using different methods of cluster analysis. The solution of the cluster analysis contains cluster centers which represent a positioning for the e-trikes. The solution is then evaluateded by a simulation model with more sophisticated parameters. This research field opens different opportunities of application scenarios, which are discussed in the conclusion.**

***Index Terms*—Machine learning, multi-agent systems, multi-agent learning, Unmanned autonomous vehicle, Clustering methods**

I. Introduction

Multi-agent Learning (MAL) [1] is a subfield of machine learning that aims to improve the behavior of agents in a multi-agent system (MAS). MAL "integrates machine-learning techniques in MAS and studies the design of algorithms to create (...) adaptive agents." [1]. It focuses on the agents' capabilities, the decision process, or the model of the environment addressing the behavior either of a single agent or of multiple agents which are interacting with each other in the MAS.

A prominent vision for future mobility is seamless traveling, often refered to as Mobility as a service (MaaS). MaaS "stands for buying mobility services as packages based on consumers' needs instead of buying the means of transport" [2]. E-mobility is a promising application field for MAL since it offers novel opportunities for MaaS applications. One goal of MaaS is to provide the support for customers during the whole process of mobility usage. E-mobility offers convenient solutions for such MaaS applications. Recent literature on applying multi-agent systems for energy efficiency issues in e-mobility have a wide variety of applications such as optimizing the number of spare drones in a fleet of drones to enable drone replacement before their batteries drain [3], or agent-based simulation of autonomous taxis [4]. E-trike sharing with a fleet of autonomous trikes is a novel application scenario that raises many research questions related to energy issues. The optimal usage of battery power is a challenge for practical applications, as this requires an optimal assignment of vehicles to trips. In the literature, this is solved by optimization approaches, such as the vehicle routing problem (VRP) [3], [5], [6]. The VRP aims to the generation of an optimal set of tours for a fleet of vehicles [7]. However, traditional optimization approaches suffer from a lack of flexibility and scalability. For example, they are not fitted for scenarios where the customers miss or cancel an appointment. Further, massive scalability requirements arise today from the number of customers, trips or – due to the free-floating nature of autonomous vehicles – potential trip origins. A research gap for adaptive and scalable solutions for operating fleets of autonomous vehicles has been identified. MAS with their self-organizing capabilities provide a promising alternative to these traditional optimization approaches. Each trike is represented by a software agent that negotiates autonomously with other agents about trip assignments and decides where to wait or travel to.

In this paper, a smart power saving solution for operating fleets of autonomous e-trikes in a MAS is presented. A novel MAL approach is introduced that uses cluster analysis to determine the agents' waiting positions and, thus, save energy for the journey to the customer. The main technical contribution of the paper is the comparison between two different clustering methods for improving the agents' decisions on their optimal positions. A geospatial model is introduced and integrated with an energy model. Training and test data are generated from a real data set of bike rental data. The resulting trike positions are evaluated by means of experiments using a simulation. Therefore we set up the hypothesis:

A scheme for positioning e-trikes can be achieved by a cluster analysis of rental data. The solutions are better with respect to the energy savings than a naive solution where the e-trikes are positioned at one place and are driving from this place to the customers.

The paper is organized as follows. In Section II, related work is discussed. In Section III, the use case is explained in detail. Section IV provides a formal representation of the geospatial model and a narrow energy model that is integrated with the spatial model. Section V presents an extended version of the distance function from Section IV that leads towards a more sophisticated energy model. This is realized in a control loop that uses a non-linear single track model of a vehicle. Section VI describes the multi-agent learning approach for the waiting positions. In Section VII, a formative evaluation of the learning approach based on lab experiments is elaborated. Future work is discussed in Section VIII. In Section IX, the paper concludes with a summary.

II. Related work

In the following, we discuss related work from two fields that are relevant for the work. First, some approaches for solving the VRP are presented. Second, work on data generation is investigated.

In its original intention, the VRP is solved with the goal to minimize the total travel distance of a fleet, for instance in an application of unmanned aerial vehicles (UAV's) for a security escort service [8]. The method in [8] is based on a linear programming approach „mixed integer linear programming"where the focus lies on the scheduling of the UAVs.

A slightly different goal of solving the VRP is to minimize the number of vehicles for surveillance of different objects by a fleet [3], [5], [6], [9]. In [3] the VRP is solved by using the Bin Packing problem. The approach aims to reduce the number of spare UAV's that are in use for the surveillance task. The problem handled in [6], is about calculating the number of drones to provide a persistent surveillance by using petri nets. The problem is therefore modified in such a way that it is comparable to the Traveling Salesman Problem with fuel constraints as well as different objective functions. Also in this work, a heuristic method and an enumeration method are considered. The optimal solution is referred to the minimum number of resources. In [5], the VRP is solved in conjunction with negotiation. An insertion heuristic approach is presented for dynamic environments by also using negotiation mechanism for energy resources. The method called „IDIH-Reserve"constructs routes for all vehicles in parallel. The assignment of batteries to vehicles is performed according to priorities via a negotiation mechanism. [9] combines the VRP with an auction algorithm. The surveillance task addresses a patrolling problem, i.e. minimizing the time lag between two visits to the same location during surveillance [10]. Our work is related to the above VRP approaches since it intends to reduce the total travel distance of a fleet by combining a learning approach for waiting positions with multi-agent negotiation. In contrast to the work reported from the literature, it abandons optimization methods for reasons of scalability.

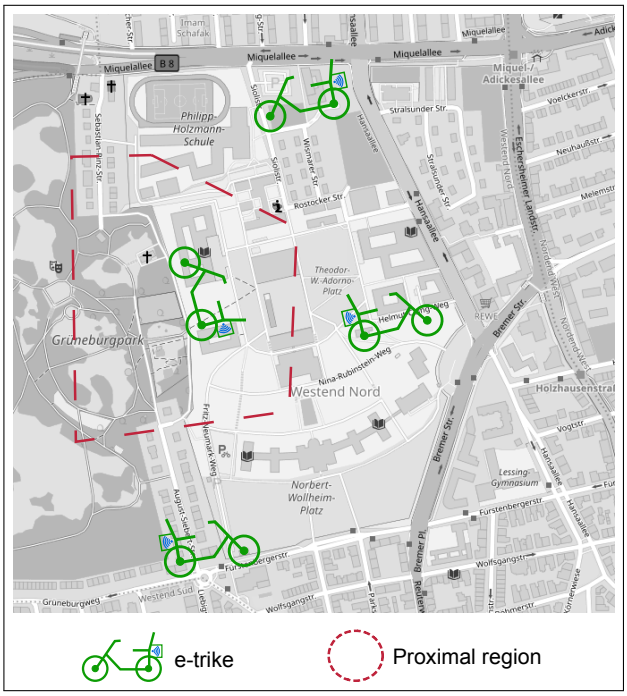

Fig. 1. Sample fleet of e-trikes.

The data preparation and the generation of random data points for the cluster analysis is achieved by using a geo information system. Related work to this problem is in [11] where a stationary bike sharing system is simulated by using monte carlo simulation generating different parameters with different statistical distributions. An other approach is mentioned in [12] where rental data is generated by using gaussian mixed model and classification trees. Hence the application scenario in our problem is based on a free float model, our focus in the data generation process lies on data points which are dependent to their GPS coordinates.

III. Application scenario

In our application scenario, a fleet of free-floating e-trikes provides shuttle services for customers. Fig. 1 depicts the Campus Westend of Goethe University Frankfurt to illustrate a sample area for this application. If a customer at an arbitrary location calls an e-trike, it arrives autonomously within reasonable time and with reasonable energy consumption. The e-trikes organize themselves in a MAS. An e-trike is represented by a software agent that decides in negotiation with other agents which trike takes over which customer request. Agents decide themselves on waiting positions that are promising to save energy for the journey to the customer. They are able to learn waiting positions from rental data records.

We apply partitioning machine learning methods in this MAL scenario to calculate the positions in a geospatial model. E-trikes that are ready to use, aim to place themselves in and around an area where it is likely that they are called up by the customers via a smartphone app. Idle e-trikes turn back to their waiting positions.

Customer requests are satisfied by the entire fleet in cooperation. Initially, a customer request is assigned to the agent that is closest to the customer's place of departure (cmp. the sample proximity region of a trike in Fig. 1 for which the trike currently takes responsibility). The agent decides with a utility function whether to approach to the customer itself or to delegate the request to another agent in a bidding process. Potential negotiation partners are identified following an „arm length"approach by means of the geospatial model (i.e. only agents are contacted for bidding that are in reasonable distance). The utility function considers the expected length and energy consumption of the journey to the customer as well as further criteria, such as the battery level or the degree of utilization.

The implementation of the MAS is ongoing work. The MAL approach for the waiting positions is a first step to pursue the application of the MAS in practice. The geospatial model is fundamental for the MAL approach but also for the agents' further functionality sketched above.

In an extended version of the MAS, the agents will be also responsible for charging up their batteries. Thus, the trips for a recharging process will be planned and executed autonomously. Moreover, the trikes will be capable to exchange battery power among each other.

IV. Geospatial model

A geospatial model is an important, basic component of our MAL approach. It comprises a representation of geolocations and routes in a map that is used to estimate the energy consumption of a trip. The agents implement such an estimation as part of a utility function for decisions on waiting positions or customer requests.

A *geolocation* is represented by an $X - Y$-coordinate pair containing the geographical latitude and longitude [13, p. 66] following the world geodetic system (WGS84) [13, p. 72]. Both coordinates are recorded in decimal degrees with a granularity of 0.0001 degrees. A sample place near the Goethe University's dining hall on Campus Westend has the geolocation $l = (latitude : 50.125673, longitude : 8.665367)$. A geolocation can be projected into the plane grid system universe transverse Mercator (UTM) to straighten the curvation of the earth. The distortion of UTM coordinates is approximately 1 meter variation in every 2,500 meters distance [13, p. 69]. A *customer request* comprises a geolocation for the origin of a trip, enriched by non-geographic data such as destination time. Inspired by usual Geographic Information Systems (GIS), a map in our geospatial model has several layers. The *route layer* uses vector data (series of coordinate pairs) to represent points, lines, and areas where the e-trikes are allowed to drive. The *position layer* reduces the route layer to the points, lines, and areas where the e-trikes are allowed to park, i.e. potential waiting positions.

The estimation of the energy consumption of a trip requires a distance function between geolocations. We denote by $dist(a, b)$ the distance between two geolocations a and b. So far, three different distance functions have been implemented in our geospatial model. The euclidean distance and a shortest-path approach are straight-forward functions explained below. A third, more sophisticated function will be introduced in Section V. The model can be extended by further, yet more sophisticated distance fuctions (compare the outlook in Section IX).

In the K-Medoids clustering (KMEDOIDS) and the CLARA clustering (CLARA) [14] we use a partitioning approach with the euclidean distance measure to spot differences in the cluster solutions.[1] The euclidean distance is defined as follows:

$$dist_{euclid}(a, b) := [\sum_{j=1}^{J} |a_j - b_j|^2]^{1/2} \quad (1)$$

For two-dimensional geolocations, J takes the value 2.

One more distance measurement in the KMEDOIDS clustering is the application of the Origin-Destination-Matrix (ODM) which contains the distances between all randomly generated data points along the position map in meters. The values of $dist_{ODM}$ are calculated by a shortest-path approach for the route layer of the map. Thus, the ODM is a symmetric distance matrix.

Based on the geospatial model, a narrow energy model is defined. Following an energy model of Hartuv et al. [3], travel times, battery power consumption, as well as battery level can be estimated in a straight-forward manner. v denotes the velocity of a trike when traveling autonomously. We assume v to remain constant, for instance 6 kilometers per hour. The travel time t_{ij} between two geolocations l_i and l_j is estimated by

$$t_{ij} := \frac{dist(l_i, l_j)}{v}. \quad (2)$$

c is the rate of discharge per time unit when travelling autonomously. The battery power consumption BPC for a journey from l_i to l_j is computed as the amount of charge units required by the trike to travel from l_i to l_j.

$$bpc(l_i, l_j) := c * t_{ij} \quad (3)$$

$E(t)$ denotes the battery level of a trike at time t. At the time of having finished loading t_r, the trike has full battery

[1] More precisely, we approximate the euclidean distance for two geolocations a and b that have been projected to two UTM coordinate pairs with a very high accurary (cmp. the above introduction of UTM). For reasons of simplicity, we stick to the usual euclidean notation instead of $dist_{euclid'}(a, b) := [\sum_{j=1}^{J} |a'_j - b'_j|^2]^{1/2}$ where a' and b' are WGS-to-UTM projected coordinates.

charge, denoted by $E(t_r) = L$ where L is the maximal battery capacity.

If we assume for reasons of simplification that E decreases linearly with time when traveling autonomously and does not decrease further while the trike is waiting or being rode by a customer, the battery level after a trip from l_i to l_j starting at time point t_s and ending at time point t_e is $E(t_e) = E(t_s) - bpc(l_i, l_j)$.

V. Simulation Based Distance Function

Battery driven autonomous vehicles consume varied amount of energy while driving on a road with different slopes. This part is significant, especially for driving with low velocities, as the other forces acting on the system strongly depend on this value. Moreover during motion, a vehicle facing a road with a downward slope will consume less energy, thus discharging the battery slower. While the opposite is true for upward-sloping roads. These effects are considered in the development of innovative battery charging management systems to increase the energy efficiency [15].

To estimate the distance covered by the vehicle driving on a given road ($dist_{simu}$), the vehicle is modeled in Simulink and simulated in a control loop. The model represents a nonlinear single track model, as stated in [16]. Moreover, this nonlinear single track model has been extended to accommodate for the road slope. The final system has 5 input values: steering wheel angle, acceleration pedal position, brake pedal position, driving gear, slope of the road. The kinematic system equation for the vehicle are described as following:

$$m\ddot{x} = F_{f,x} + F_{b,x} - F_{w,x} - mg\sin(\delta_z) \quad (4)$$

$$m\ddot{y} = F_{f,y} + F_{b,y} - F_{w,y} \quad (5)$$

$$m\ddot{z} = F_{f,z} + F_{b,z} - F_{w,z} - mg\cos(\delta_z) \quad (6)$$

$$\theta_{zz}\dot{\psi} = {}^{f}F_{f,y}l_f - {}^{b}F_{b,y}l_b \quad (7)$$

Where m, l and θ_{zz} stand for the total mass, length and moment of inertia around the z-axis of the vehicle. The sub-indices f and b indicate forces acting on the front or back wheel. While a left index indicates that the force is considered in the coordinate system of the corresponding wheel. Note that the absence of a letter indicates the correspondence to the global coordinate system. The slope is denoted by δ_z and the yaw rate by $\dot{\psi}$. In order to solve these equations, the forces acting on the wheels in both coordinate systems (global and at each wheel) are needed. By using the Dynamic equations of both wheels, as stated in [16] with the addition of the road slope, these forces can be found. This means that when doing a transformation from the original coordinate system to the front wheels, the yaw, steering, and slope angles have to be considered, while only the yaw and slope angles have to be considered for the back wheel.

The implemented vehicle model in Simulink is represented in Fig. 2. As illustrated, the upper part of the image deals with bicycle kinetics, while the lower part handles the wheel dynamics. This model is inserted in the control loop depicted in Figure 3. Both the speed of the vehicle as well as the yaw

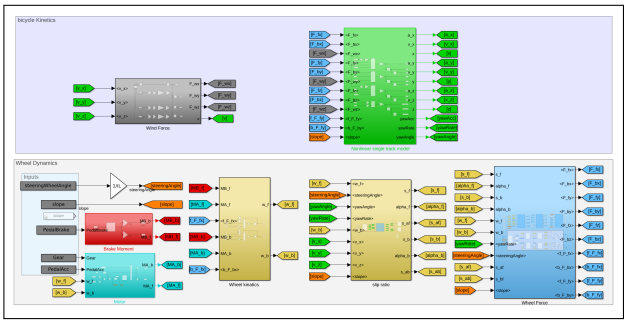

Fig. 2. Nonlinear single track model implementation in Simulink.

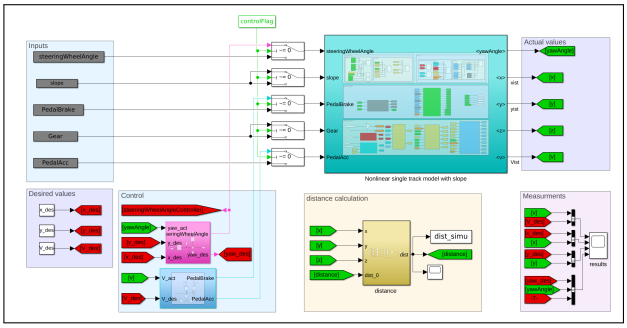

Fig. 3. Control loop of the model shown in Fig. 2.

angle are controlled using PID controllers. In order to achieve the desired speed V_{des}, the position of the braking as well as the accelerating pedals are used as the control inputs. On the other hand, in order to follow a given path (x-y), the desired yaw angle is calculated and achieved by controling the steering wheel angle. The slope of the road as well as the current gear are given time vectors.

For a given road (x_{des}, y_{des}, slope) and a desired speed (V_{des}), the pre-computed distance covered by the bicycle can be calculated as following:

$$dist_{simu} = dist_{simu,0} + dist_{step}$$

$$dist_{step} = \sqrt{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2}$$

Where the sub-index "0" indicates the result of a previous computation step. At every computation step, the previous calculated distance ($dist_{simu,0}$) is added to the computed distance from the current time step ($dist_{step}$).

Compared to the value calculated in (1) , $dist_{simu}$ represents a more accurate value as the bicycle dynamics along with the road properties are considered in this calculation.

VI. Multi-Agent Learning - Energy Efficient Placement of E-Trikes

The placement of a fleet of e-trikes is objective of our learning approach. The aim of learning is to find promising waiting positions for the trikes for customer requests to be expected. We assume that positions which are optimal for historical rental data are suitable also for recent customer requests.

As an unsupervised method for separating data points into different groups different cluster analysis methods [14] are used by a partitioning approach. In partitioning clustering, k clusters are defined before the data points are assigned to the clusters. Furthermore, the clustering methods used in this work are hard clustering methods, which means that every data point is assigned to one and only cluster. Clustering methods group the historical data into clusters where the distance of the trike position to the start positions of the customer requests is minimal.

CR denotes a set of customer requests. Based on the presented application scenario we determine the amount of e-trikes needed for a set CR as well as their waiting positions using clustering methods. The set of geolocations O for the origins of all trips in CR serve as the data points to be clustered, where $k \leq |O|$. Here, $O_i \subseteq O$ denotes the data points of the i-th of k clusters, $1 \leq i \leq k$. Each $l_o \in O_i$ is a point of the position layer of a given map. $center(i)$ denotes the center of cluster i representing the waiting position of a trike. The objective function for the clustering methods is:

$$f(x) := \sum_{i=1}^{k} \sum_{l_o \in O_i} dist(center(i), l_o) \tag{8}$$

Two clustering methods namely KMEDOIDS and CLARA [14] are used. A weakness of the KMEODIDS is the runtime complexity $O(k(n-k)^2)$ during the comparison process of all data points to the k clusters. The main advantage of KMEDOIDS is the robustness to outliers.To overcome the high complexity of KMEDOIDS, it is beneficial to use a sampling-based clustering method which is known as the CLARA algorithm.

The cluster solutions are visualized with the support of Voronoi diagrams showing the cluster partitions in the hard clustering. In this case the Voronoi diagram shows the separation of the data points as customer trips.

VII. Evaluation

The hypothesis (cmp. Section I) is investigated by experiments comparing two clustering methods with two different distance functions. The experimental data is prepared based on historical rental data as described below. Further, we explain the experimental setup in detail and discuss the experimental results.

A. Preparation of experimental data

The goal of data preparation is to achieve training and test data sets with customer requests $CR_{Training}$ and CR_{Test} in a free-float scenario. We generate free-float data based on real-world data from a bike sharing company [17] providing station-based rental data. The raw data contains over 16 Mio. of recordings of historical trip data from 2014 to 2017. A trip comprises values for the start station, start date and time, end station, end date and time, the identification number of the bike, the customer ID and further attributes such as input channel of booking. To build $CR_{Training}$ and CR_{Test} sets, the raw data is reduced in number of records and attributes. Free-floating geolocations are generated to substitute the station-based origin of each trip record.

A filter extracts the records from the given dataset, which have one of seven stations in and around the Campus Westend of Goethe University Frankfurt as a start station. For our experiments, the start date and time are relevant as well as the geolocation of the start station. The data reduction results in a corpus of 183,120 trips with three attributes "Start_date_and_time", "Start_longitude" and "Start_latitude".

The coordinates have to be transformed into geolocations at various places in the proximity region of the rental station. The proximity regions around the seven stations are identified manually for the considered area. The transformation process of the geocoordinates is implemented using the geo information system QGIS [18], [19]. The route layer of the map for the considered area is drawn with QGIS which extracts the GPS coordinates of the rental stations from OpenStreetMap [19] and partitions the map into the seven proximity regions around the stations. In the next step an algorithm integrated in QGIS called "Random points along a line" computes a set of randomly generated GPS coordinates as geolocations, which lay exactly on the position layer of the map. For each rental record, a generated geolocation in the proximity region of its rental station is assigned arbitrarily. This procedere follows the intuition that the customers who rented a bike from a station would presumably call an autonomous trike in its proximity region. The resulting dataset thus contains records, which indicate various starting positions of the rides. Their geolocations serve as two-dimensional data points for the clustering analysis.

The different distance functions require also some data preparation. To generate the ODM for $dist_{ODM}$ on the route map, the generated data points as well as the application QGIS is used. The ODM is generated in QGIS with the package "QNEAT3" where the algorithm "OD Matrix from Points as Table (n:n)" is used, generating the distance between 1000 randomly generated GPS-coordinates. Thus an ODM has 1.000.000 entries containing the distances between the 1000 data points.

B. Experimental setup

As we stated in our hypothesis our approach to position the e-trikes is processed by cluster analysis. In particular, we use the appropriate KMEDOIDS as well as CLARA cluster analysis. The experimental setup is therefore build up considering these two clustering methods with different configurations and we assume that the results deliver positioning concepts which are more energy efficient than a naive solution with one stationary place for all e-trikes. The evaluation consideres clustering indices as well as a comparison of the average journey to the customers.

The application of cluster analysis is divided into several runs with different parameterizations of "Algorithm", "Number of data points", k and "Distance function". Table I

describes the parameter settings for entirely 15 runs. For reasons of computational complexity, 1000 data points are used as $CR_{Training}$. CLARA runs on the same set $CR_{Training}$ as the two configurations of KMEDOIDS. The value for the number of clusters is iterated over $k = 3, 4, ..., 7$.

The cluster solutions are evaluated by the two well-known clustering indices Davies-Bouldin-Index (DBI), Silhouette Index (SIL), and by a novel, task-specific measure that we call Average-Journey-to-the-customer (AJ).

The Davies-Bouldin-Index (DBI) is defined as follows [20] (as cited in [21]):

$$DB(k) = \frac{1}{k}\sum_{i=1}^{k} \underset{j \neq i}{max} \frac{\Delta(C_i) + \Delta(C_j)}{\delta(Ci, Cj)} \tag{9}$$

where ΔC_i denotes the intra-cluster distance, calculated as the average distance of all the cluster objects C_i to the cluster medoid, whereas $\delta(Ci, Cj)$ denotes the distance between the clusters C_i and C_j (distance between the cluster medoids). The optimal value for k is where $DB(k)$ is the least.

The Silhouette Index is measured as follows [22] (partially as cited in [21]): The function $s(i)$ measures how well an object i (a data point) has been classified, which is defined as follows:

$$s(i) = \frac{b(i) – a(i)}{max(a(i), b(i))}$$

where $a(i)$ is the average distance from data point i to all other data points within the cluster and $b(i)$ is the average distance from data point i to the next cluster.

The silhouette for every cluster C_j is defined as follows:

$$Sil(C_j) = \frac{1}{|C_j|} * \sum_{i \in C_j} s(i)$$

The silhouette of the whole partition is the sum of the average cluster silhouettes:

$$Sil(k) = \frac{1}{k}\sum_{r=1}^{k} Sil(C_r) \tag{10}$$

The optimal value for k is maximizing $Sil(k)$.

TABLE I
PARAMETER SETTINGS OF THE EXPERIMENTAL CLUSTER ANALYSIS

Configuration	Points	Algorithm	k	Distance measure
1	1000	KMEDOIDS	3-7	euclidean distance
2	1000	KMEDOIDS	3-7	ODM
3	1000	CLARA (Sample: 50)	3-7	euclidean distance

In addition to the cluster measures, we introduce a task-specific measure as part of a simulation-based evaluation. We use a test set CR_{Test} of 30 customer requests that are not included in $CR_{Training}$. The cluster results are used as classification model M to assign the data points in CR_{Test} to the calculated clusters. $l_{medoid}(l)$ denotes the medoid of the cluster containing data point l. For each cluster solution M resulting from a particular run, we measure the distance from the test data points to their l_{medoid}, i.e. the average journey to the customer, in meters:

$$AJ(M, O_{Test}) = \frac{\sum_{l_O \in O_{Test}} dist_{simu}(l_{medoid}, l_o)}{|O_{Test}|} \tag{11}$$

The clustering experiments are executed in the ELKI framework which is a helpful tool providing different clustering algorithms and a visualization for the clustering results [23]. The cluster algorithms are processed in ELKI 0.7.5 on a virtual machine with an Intel Xeon CPU E5-2630 @ 2.30 GHz with 4 cores, 16GB Memory, and Win 7 64-Bit.

C. Experimental results

Fig. 4 to 6 illustrate the learned clusters for $k = 6$ in the three different configurations as an example. The tailoring of clusters differs slightly between the configurations. However, the cluster measures achieve quite similar results. Table II depicts the experimental results for the cluster measures.

TABLE II
MEASURED RESULTS FOR KMEDOIDS AND CLARA

Config.	Measure	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$
1	SIL	0.4476	0.3720	0.3504	0.3744	0.3466
2	SIL	0.4274	0.3930	0.3549	0.3505	0.3438
3	SIL	0.4473	0.3718	0.3556	0.3095	0.3269
1	DBI	0.8280	0.9515	0.8704	0.8467	0.8476
3	DBI	0.8260	0.9939	1.2430	1.1877	1.0573
1	AJ	457.73	438.33	494.87	420.47	378.38
2	AJ	518.49	438.90	442.03	425.87	376.69
3	AJ	567.82	468.12	394.37	416.71	490.63

A higher SIL value means a better separation of the data points into clusters. A lower DBI value indicates a better compactness of the clusters. As expected, CLARA (configuration 3) results in slightly worse SIL values in comparison to KMEDOIDS for the most cases due to the reduced number of data points in the samples. Surprisingly, ODM (configuration 2) does not achieve significant improvements for all the three measures in comparison to the euclidean distance (configurations 1 and 3). It seems that the euclidean distance provides a suitable approximation already. A significant improvement of chosing a higher k value is not observable in Table II. However, the experiments provide very good results in comparison to the naive solution. For an arbitrarily chosen geolocation approximately in the center of the considered area, the average AJ is 1201.05 meters. Any of the cluster models investigated saved more than the half the route distances for the journey to the customer. This result is very promising concerning the power saving capabilities of our approach. The differences in the theoretical computational complexity between KMEDOIDS and CLARA are reflected by the

measured runtimes. For $k = 7$, KMEDOIDS with euclidean distance has a measured duration of 1,346 ms. The application of ODM in KMEDOIDS nearly doubles the runtime to 2,393 ms. CLARA significantly outperforms KMEDOIDS with 79 ms.

The main advantage of CLARA is the sampling approach which provides a faster processing of the cluster analysis. In the intended application scenario, the e-trikes will use the cluster analysis process as an online learning method. Therefore a fast and efficient cluster method is needed which is provided by the CLARA algorithm. The amount of data to be clustered can be high and changes frequently over the time. Therefore an approximative approach is suitable by using CLARA with a sampling method. In general, the experimental results clearly confirm our hypothesis. CLARA with the euclidean distance seems to achieve satisfying results and is preferable for reasons of computational complexity.

VIII. Future work

The results raise many novel research questions to be investigated in our future work.

The clustering approach can be improved towards different directions. First, time intervals can be considered. For instance, the historical data can be partitioned into weekends and working days, or rush hours, medium traffic hours and night hours. Second, the traditional clustering can be extended by online learning for current operating data. Third, the simulation based distance function can be extended from a 2D model to a 3D model considering the slope of the area. This requires to extend the geolocations by a third parameter and the map by an additional layer for topographical information. Further, the simulation model is to be extended by the slope. Since the resulting distance function will not be a symmetric measure any more, the clustering methods will have to be replaced by clustering methods for directed graphs.

In the distant future, more profound improvements of the learning method can be considered. The model proposed in Sections IV and V results in an estimation for the distance covered by the bicycle. Dynamic velocity values can be integrated to further extend the distance function towards a time-aware distance function. Our narrow energy model is directly derived from the geographical distance in a linear way. In the opposite direction, an extended energy-aware distance function can be developed that is derived from a more sophisticated, non-linear battery model. Future work will include developing a battery model which will be linked to the existing model, leading to a very accurate estimation and preventing running out of energy at steep slopes. Additional impact factors, such as the ageing of the battery, can be considered for this.

The use of the learned model can be improved as well. For instance, idle trikes can select the closest free waiting position instead of traveling back to their own, probably more distant waiting position.

The application should be tested with real data observed from operating a fleet of free-floating e-trikes instead of using historic data from a station-based fleet. A use case scenario for the elderly society highlights the potential benefit of our approach for future mobility solutions, such as a shuttle service to medical appointments, shopping, or leisure activities. Especially, one strength of e-trikes is the ease of use for elder persons.

IX. Conclusion

In this paper we presented a multi-agent learning approach for an energy-aware positioning of autonomous agents that represent e-trikes. The approach includes a formal, geospatial model of a map with a generic distance function, three sample implementations of this distance function, and different clustering methods to learn waiting positions for the trikes. An experimental dataset from a bike sharing provider containing rental data is transformed by their GPS coordinates to simulate a free-float model. As a support a GIS is used to generate random coordinates as well as a drawn map to determine the considered area, where the e-trikes can be placed and where they are allowed to drive. A formative evaluation of the cluster analysis has been conducted and led to promising results for saving energy consumption in operating the fleet of e-trikes.

The contribution of this paper is part of ongoing work on e-mobility. Our MAL approach provides an initial building block towards novel research perspectives for MaaS. In future, we expect this field to be of significant impact for both, economy and society.

References

[1] K. Tuyls and G. Weiss, "Multiagent learning: Basics, challenges, and prospects," *Ai Magazine*, vol. 33, no. 3, pp. 41–41, 2012.

[2] M. Kamargianni, W. Li, M. Matyas, and A. Schäfer, "A Critical Review of New Mobility Services for Urban Transport," *Transportation Research Procedia*, vol. 14, pp. 3294–3303, Jan. 2016.

[3] E. Hartuv, N. Agmon, and S. Kraus, "Scheduling Spare Drones for Persistent Task Performance under Energy Constraints," in *Proceedings AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pp. 532–540, International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018.

[4] F. Dandl, B. Bracher, and K. Bogenberger, "Microsimulation of an autonomous taxi-system in Munich," in *MT-ITS 2017*, pp. 833–838, IEEE, 2017.

[5] V. Mersheeva and G. Friedrich, "Multi-UAV Monitoring with Priorities and Limited Energy Resources," in *Twenty-Fifth International Conference on Automated Planning and Scheduling*, Apr. 2015.

[6] H. Park and J. R. Morrison, "On the resources required to provide persistent robotic service: Multiple immobile customers and a single service station," in *Proceedings APIEMS'14*, 2014.

[7] J. Ehmke, *Integration of Information and Optimization Models for Routing in City Logistics*. International Series in Operations Research & Management Science, New York: Springer-Verlag, 2012.

[8] B. D. Song, J. Kim, J. Kim, H. Park, J. R. Morrison, and D. H. Shim, "Persistent UAV Service: An Improved Scheduling Formulation and Prototypes of System Components," *Journal of Intelligent & Robotic Systems*, vol. 74, pp. 221–232, Apr. 2014.

[9] N. Nigam and I. Kroo, "Persistent surveillance using multiple unmanned air vehicles," in *2008 IEEE Aerospace Conference*, pp. 1–14, IEEE, 2008.

[10] A. Almeida, G. Ramalho, H. Santana, P. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyre, "Recent advances on multi-agent patrolling," in *Brazilian Symposium on Artificial Intelligence*, pp. 474–483, Springer, 2004.

[11] S. Ji, C. R. Cherry, L. D. Han, and D. A. Jordan, "Electric bike sharing: simulation of user demand and system availability," *Journal of Cleaner Production*, vol. 85, pp. 250–257, Dec. 2014.

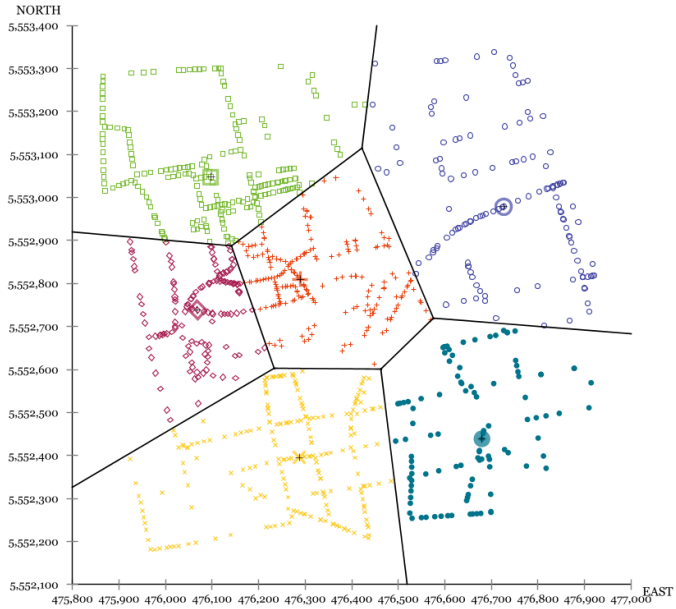

Fig. 4. Clustering result with KMEDOIDS for $k = 6$ and euclidean distance.

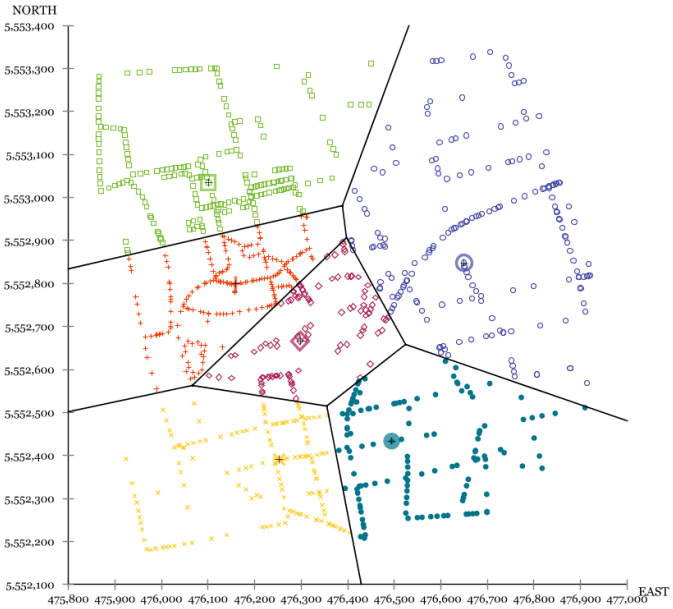

Fig. 5. Clustering result with CLARA for $k = 6$ and euclidean distance.

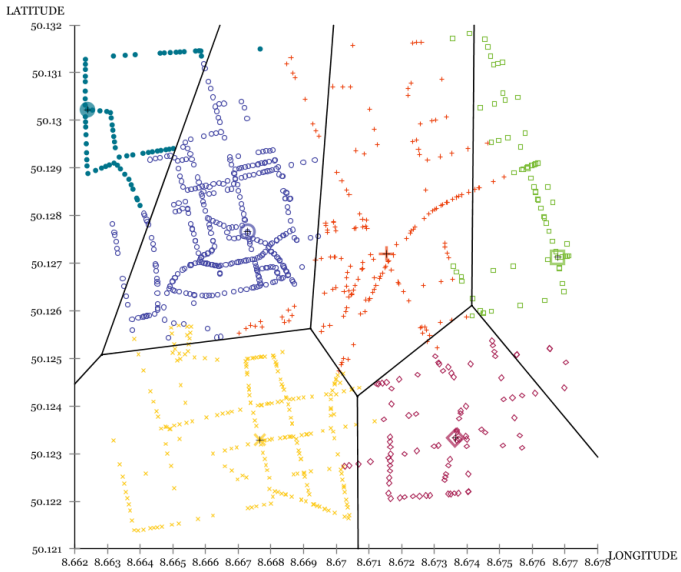

Fig. 6. Clustering result with KMEDOIDS for $k = 6$ and ODM.

[12] A. B. Brendel, C. Rockenkamm, and L. M. Kolbe, "Generating Rental Data for Car Sharing Relocation Simulations on the Example of Station-Based One-Way Car Sharing," in *HICSS*, 2017.

[13] M. N. DeMers, *Fundamentals of geographic information systems*. John Wiley & Sons, 2008.

[14] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*, vol. 344. John Wiley & Sons, 2009.

[15] C. G. H. Schroth, "Verfahren zum Laden eines Fahrzeugs mit elektrischem Antrieb, Ladesteuerung und Ladestation." German patent DE 102017219204 A1 (published).

[16] D. Schramm, M. Hiller, and R. Bardini, *Modellbildung und Simulation der Dynamik von Kraftfahrzeugen*. Berlin: Springer Vieweg, 3., aktualisierte und ergänzte auflage ed., 2018. OCLC: 992986573.

[17] Deutsche Bahn AG, "Datensätze - Open-Data-Portal – Deutsche Bahn Datenportal." https://data.deutschebahn.com/dataset, last access 09-05-2019.

[18] QGIS.ORG association, "QGIS - A Free and Open Source Geographic Information System." http://qgis.com, last access 09-05-2019.

[19] A. Bruy and D. Svidzinska, *QGIS By Example*. Packt Publishing, June 2015.

[20] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.

[21] A. Albalate and W. Minker, *Semi-supervised and unsupervised machine learning*. London, UK: ISTE/Wiley, 2011.

[22] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[23] E. Schubert, A. Koos, T. Emrich, A. Züfle, K. A. Schmid, and A. Zimek, "A Framework for Clustering Uncertain Data," *Proc. VLDB Endow.*, vol. 8, pp. 1976–1979, Aug. 2015.