Transportation Engineering 4 (2021) 100062

Contents lists available at ScienceDirect

Transportation Engineering

journal homepage: www.elsevier.com/locate/treng

A simulation study of platooning AV fleet service in shared urban environments with uncertainties

Ran Dong[a,*], Roger Woodman[a], Paul A. Jennings[a], Simon Brewerton[b], Stewart A. Birrell[a], Matthew D. Higgins[a]

[a] WMG, University of Warwick, Coventry, United Kingdom
[b] RDM Ltd., Coventry, United Kingdom

A R T I C L E I N F O

Keywords:
Electric Autonomous Vehicle
Platooning
Urban transport
Uncertainty
Monte Carlo Simulation

A B S T R A C T

This paper conducts a simulation study of low-speed Autonomous Vehicles (AVs), referred to as "pods", platooning in shared urban environments. The proposed on-demand transport service can help solve the "last mile" challenge and improve mobility for non-drivers, elderly, and disabled people. To help the industry understands the dynamic system for deployment, this paper provides a practical prospect for pod platooning without prior planning. We designed a simulation system for the new transport. Non-homogenous Poisson processes were adopted to simulate arrivals of user requests. A three-density-zone map was designed from real-world city layouts. Practical time and location patterns of user demand were used. The supervision cost reduction as a benefit of platooning was estimated. System performance and user experience were evaluated. We identified how deployment can influence platooning. We found that platooning can reduce supervision cost by approximately 20% and 14%, during peak times on weekdays and midday on weekends respectively.

1. Introduction

1.1. Background and motivation

A challenge for public transport and urban planners is a phenomenon known as the "last-mile", which describes the difficulty in getting people from a transportation hub, such as railway and bus stations, to their destinations [1,2]. A low-speed autonomous transport system consisting of electric fully autonomous vehicles (EAVs), or colloquially as "pods" in the literature, would help solve the last-mile challenge, satisfy the increasing personal travel demand and ensure accessible and convenient public transport for sectors of our society, including elderly, people with mobility issues and non-drivers [3].

As an environmentally friendly transport, electric pods, like the type used in this study, have the potential to improve people's mobility at a more affordable price than taxis. A unique advantage of the pod service we propose, comes from its ability to operate in shared spaces. In this paper, the "shared space" includes spaces travelled by both pedestrians and vehicles, and some suitable car-free pavements for pedestrians. This means pods will be able to pick up and drop users where taxis cannot reach. Since pod will typically not travel on road, it would avoid the fast traffic environment. On one hand, the pod would increase the transport capacity without increasing traffic on road; on the other hand, pods will be able to take shorter routes than other vehicles. For example, pods can run through a square.

At present, UK Government legislation requires AVs to be supervised by a human at all times, either in the vehicle or remotely. This need of a human supervisor for each vehicle makes the value proposition of small vehicles too costly for both passengers and the company. To cope with this practical difficulty and to make the service commercially viable, the SWARM project proposes the use of platooning pods [4]. With a platoon of pods it opens up the possibility of having one supervisor supervising multiple pods (up to 5 in our system), therefore reducing the number of supervisors required to manage a fleet, which should ultimately lower operating costs. In other words, several pods running in a platoon share a supervisor. The idea of sharing in transport makes a difference. For example, ride-sharing services could help with traffic congestion, save energy consumption while satisfying people's travel needs. To name a few studies on it, Fagnant and Kockelman [5] studied shared autonomous vehicles, Caulfield [6] conducted a case study on ride-sharing for Dublin. Supervision of operations of an AV fleet has been studied [7].

Research into vehicle platooning originated from studies of traffic dynamics and behaviours more than 50 years ago [8,9]. Since the transportation sector is responsible for the largest share of increased oil consumption (see e.g. [10–12]), truck platooning has attracted increased

* Corresponding author.
E-mail address: raneileendonald@163.com (R. Dong).

https://doi.org/10.1016/j.treng.2021.100062
Received 3 December 2020; Received in revised form 14 February 2021; Accepted 26 February 2021

attention in recent years, due to its advantage of energy saving. This is because vehicles travelling close behind each other can reduce aerodynamic drag, reducing fuel consumption. Liang et al. [11] studied how several scattered heavy-duty vehicles (HDVs) can cooperate to form platoons on highways in a fuel-efficient manner and proposed an algorithm that coordinates neighbouring vehicles pairwise. Many interesting problems, such as systems of distributed controllers for HDV platooning [13], mathematical framework for trucks platooning in road networks [14] and the trade-off between energy savings and delays caused by platooning [15] have been studied.

Many aspects of vehicle platoon have been studied, such as planning strategy, communications technology and control problems. To name a few, Feng et al. [16] proposed a composite platoon trajectory planning strategy to maximize the intersection throughput; Xu et al. [17] proposed a distributed platoon formation framework considering vehicle dynamics; Gong and Du [18] developed a cooperative platoon control for a mixed traffic flow including human drive vehicles and connected and autonomous vehicles; control strategies for platooning based on model predictive control have also been studied (see e.g. [19,20]). In addition, communication technologies have also been studied for platoon, such as V2X (vehicle-to-everything), 5G V2X and V2V (vehicle-to-vehicle) (see e.g. [21–23]).

Moreover, simulation studies have also been conducted for platoon. To name a few, Teixeira et al. [24] presented a simulation framework that unifies vehicular, communications and multi-agent system simulators in order to test the coordination mechanisms more effectively and realistically; Elbert et al. [25] developed an agent-based simulation model for analysing the trade-off between savings and waiting times due to platooning. Bhargava et al. [26] developed a traffic simulation model to analyse the impact of tunnel closures necessary for monitoring the flow of dangerous goods vehicles and abnormal load vehicles.

Travelling together as a platoon can reduce cost and risk. An efficient fleet management is therefore important. For example, Wesolkowski and Wojtaszek [27] used fleet configuration to improve the scheduling and Billhardt et al. [28] proposed to employ an event-based architecture for dynamic fleet management and applied it to the coordination of an ambulance fleet in a medical emergency scenario to reduce response time.

A similar concept of platooning is swarming. Proposed in 1989 [29], swarm intelligence has been developed and transferred from robotics to transportation studies [30,31]. For example, particle swarm optimization has promoted the study of scheduling and routing problems (see e.g. [32–35]).

1.2. *Paper outline*

Most studies about platooning focus on trucks travelling on roads. Although pod platooning will not reduce the energy consumption as significantly as truck platooning, it can reduce the number of human supervisors needed, hence reducing the operation cost. A detailed simulation study on it is of interest to the industry.

This paper will investigate platooning in shared urban environments with the emphasis on supervision cost reduction. To the best of the authors' knowledge, this is the first simulation study on supervision cost reduction by EAV platooning. There are many practical questions to be answered before the industry can provide the service to the public through feasible deployment. For example, how many pods we need such that platooning can happen a lot to obtain visible cost reduction, how large the operating area should be, how the system would perform under different user demand patterns with practical uncertainties, etc. The answers to these questions can be found from our simulation.

This paper provides a practical prospect for pod platooning without prior planning in the real world. The reasons for omitting planning are, firstly it is simple and does not require central control or orchestration, and secondly, it can provide a baseline for other platooning strategies in future study. We also incorporated practical uncertainties into the simulation by Monte Carlo method, using suitable stochastic models based on real data. This simulation study has two main parts. The first part is to investigate how deployment can influence pod platooning. Study of this specific problem used the standard (uniform) grid maps. The second part is to predict the system performance and user experience in a day of service. This service is planned to run from 06:00 to 24:00. The simulation of one day trial used a three-density-zone map including four hotspots, with higher path density and lower speed limit in the centre. We considered higher user demand in the centre and at hotspots. Time patterns of user demand for weekdays and weekends were set based on real data and a previous study [36].

Practical uncertainties were integrated into the simulations. We adopted Poisson processes, a type of stochastic processes which can reasonably and effectively model customer arrivals, to simulate arrivals of user requests. We considered that some users will not show up in time at the pickup node, resulting in cancellation of the request. We set reasonable dynamic probabilities to decide whether pods can "see" each other to form a platoon when they are getting close. Random disturbances were also considered into pods' speeds and time for users to get on/off board.

Here are the key findings from the simulation. We identified several influencing variables on deployment on platooning opportunity and helped understand the influence by statistical modelling. We provided a list of suitable fleet sizes for user demand from low to high levels. Users' waiting time during peak and off-peak hours on weekdays and weekends is evaluated. We found platooning during the rush hours on weekdays and platooning around midday on weekends can reduce the supervision cost by roughly 20% and 14% respectively. Therefore it is worth trying to platoon during these time periods.

2. Methodology and experimental setup

The "last-mile" problem can be largely relieved in our transport service by setting suitable "pod stops". In our experiment, a node is referred to as a "pod stop", where users get on and off to start and complete journeys. By setting several pod stops at a residential area, pods can bring convenience to residents, especially the elderly, people with mobility issues and non-drivers. Notice that our nodes could be set on suitable pavement at front of a store, and in a square where people sit for leisure. For example, carrying residents between their houses and stores or square in the town centre. Now let us investigate the transport system which provides the convenient service through a simulation study.

To investigate the dynamic system of cooperative EAVs, we created a multi-agent system simulation in MATLAB. Using this simulation as an experimental platform, we developed an AV platooning algorithm for simulation purpose.

The results from our simulation study will eventually be implemented on a fleet of SWARM pods being developed by RDM. These pods, an example of which is provided in Fig. 1, are fully electric and can seat up to 4 passengers. The working parameters of the pod form the basis of the simulation developed for this study.

2.1. *System framework*

The system framework defines a set of rules for which our simulation are built on. The framework of our system operation has the following assumptions in Table 1:

This on-demand transport service allows users to request to travel between any two nodes on our map immediately. After a user request is received, the closest available pod will be dispatched to pick up the users at the departure node (we also refer it to "pickup node"). The pod will wait the users for up to 3 min at the departure node. Users will use facial recognition for identity verification, as it is secure, fast and easy for users. After users get on board, a two-minute safety announcement will be broadcast. Then pod will depart for the destination node according to the shortest path.

Fig. 1. RDM's SWARM pod at the University of Warwick campus.

Table 1
System assumptions.

Assumption 1:	All paths are two-way traffic.
Assumption 2:	Pods can bypass other pods and pedestrians.
Assumption 3:	Pods are not running exactly in the middle of the path and two pods may "see" each other even though there are other pods running between them.
Assumption 4:	Pods can wait for users at the departure node for up to 3 min, if users have not shown up in time, the user request will be cancelled.
Assumption 5:	Supervisors only supervise moving pods. Pods which already have terminated at nodes are not supervised. We always have enough supervisors.

Our map design is based on undirected graph, where each edge connecting two nodes is referred to as a "path". A node is referred to as a "pod stop", where users get on and off to start and complete journeys. In our simulation, each path can have its local speed limit, all lower than the legal speed limit 24 km/h. When pods need to move, we always chose the shortest path as the pod's route.

To make the simulation accurate and to integrate as much practical details as possible, the time unit for simulation was chosen to be one second. In other words, the change of the dynamic system at every second was simulated.

2.2. Pod platooning

There are three modes for running pods. Each running pod is in one of the three modes: (1) running alone, (2) joining by changing speeds to form a platoon, and (3) running in a platoon. Pods in a platoon have the same speed and are considered to have the same location for simplicity.

When two pods are joining, namely in mode (2), they are referred to as a "prospective platoon", the front pod slows down and rear pod speeds up. When the distance between them are within 3 m, we consider they have joined successfully and become a platoon. In other words, their running modes both change from (2) to (3). Sometimes it is "too late" to join, pods in mode (2) may not always join together successfully. For example, when the front pod has passed their common paths; or the distance between them is increasing to be far (an edge length) due to their original speeds (front pod very fast and rear pod very slow). If two pods fail to join, then both pods run alone, namely change their modes back from (2) to (1). If two pods in a platoon have passed their common paths, which means they will enter different paths, or one pod has arrived at the destination, then the platoon splits, the remaining running pod has its mode changed from (3) to (1).

Table 2
Conditions to join.

Condition 1:	Pods are within 50 m of each other.
Condition 2:	Pods will not terminate at their next nodes.
Condition 3:	Pods are currently travelling, or will have the chance to travel, on the same path, in the same direction.
Condition 4:	New platoon will have no more than 5 pods.
Condition 5:	Pods have line of sight with each other.

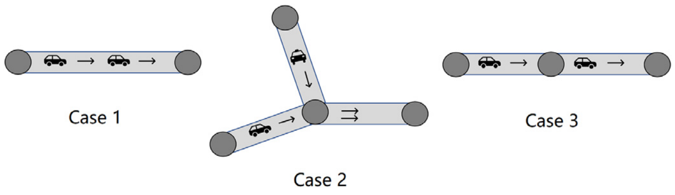


Fig. 2. Three pod-path cases where platoon can form. The dark grey circles are nodes, arrows point to the running directions of the pods next to it.

Table 3
Pod-path cases.

Case 1:	Pods are travelling on the same path in the same direction.
Case 2:	Pods running on different paths will enter the same path with the same direction.
Case 3:	A pod's current path is the other pod's next path and they will run in the same direction.

A platoon cannot exceed five pods for safety purpose. The communication for joining is between two pods. For a platoon this is done by its leading pod. There are three types of joining and it can happen between: (1) two single pods which are running alone, (2) a single pod and a platoon of fewer than five pods, and (3) two platoons if the they have no more than five pods in total.

Joining can happen at any time and in any location, as long as the following conditions (in Table 2) are met:

Specifically, Condition 3 includes three pod-path cases, shown in Fig. 2, where the dark grey circles represent nodes and arrows point to the running directions of the pods next to it. The three pod-path cases where platoon can form are specified as follows (see Table 3):

In Case 1, both pods have the same running direction (here from left to right), it is easy to see that the right pod is at front. So the right (front) pod slows down and the left (rear) pod accelerates for joining. In Case 2, two pods will pass the same node and both enter the right path. For Case 2, it is not clear to tell which is the front pod. In our simulation, we let two pods run as normal, until one pod has reached their common next node. Then this pod is set to be the front pod and starts decelerating, the other pod starts accelerating as the rear pod. If two pods happen to pass the common node at the same second, then they become a platoon directly. In Case 3, the left pod will pass the middle node and enter the same (right) path as the right pod, then it becomes Case 1. Obviously the right pod is at front.

In Condition 5, we considered that pods may not always be able to "see" each other for cooperatively joining. For example, sometimes there can be pedestrians or other pods travelling between the two pods and "blocking view". Then the two pods fail to detect each other at that second. As long as other conditions still hold, pods still have the chance to detect each other at next second. This is because pedestrians may already walk away and our Assumption 3 in Section 2.1 guarantees that pods can only be temporary obstacles for line of sight. Simulation of this uncertainty will be explained in Section 2.3.

Low acceleration rate was considered in platooning. The acceleration rate for both front and rear pods is 0.8 m/s^2 or lower when they are not too close. The front pod reduces speed until reaching the very low speed 0.5 m/s. The rear pod may run as fast as the speed limit when it is over 8 m away from the front pod. When they are getting closer (within 8 m), the rear pod's speed depends on both their distance and the front pod's speed. A specific speed control was used for this situation to avoid overtaking. This will be specified in Algorithm 3 in Appendix.

When two pods are getting close to be within 50 m, they would communicate by the "hand-shake" protocols, to exchange informations such as their locations and next paths, to determine platooning or not according to the conditions in Table 2. If pods find conditions to join are all satisfied, they would identify which is at front and which is at rear, then pods would change speeds accordingly. So far, we have set conditions for when pods could join and how they could make it cooperatively. Every pod as an agent is equipped with the same platooning rules, and no central control is required. For other travelling apart from platooning, each pod runs on its own independently as an agent. Hence we have created a multi-agent system simulation for pods' travel.

Passing node can bring important changes to the prospective platoon. Changes depend on many factors, such as the pod-path case, it is the front or rear pod that is passing a node, etc. When the pod-path case is Case 1: if the front pod only is passing a node, then change the pod-path case to be Case 3; nothing needs to be changed if both the front and rear pods are passing a node. (Notice that it is impossible that the rear pod only is passing a node in Case 1.) For Case 2: if only one pod is passing the node, then it becomes Case 3, otherwise, the platoon is formed successfully directly as mentioned above. For Case 3: if only the front pod is passing a node, this means the distance between the front and rear pods are too far away (over an edge length), then give up platooning (this platooning is failed); if only the rear pod is passing a node, then it becomes Case 1; if both front and rear pods are passing a node, then no setting needs to be changed, in other words pods are still joining in Case 3.

When an (already formed) platoon is passing a node, we took these actions. Check whether any pod is terminating at this node. If so, drop the stopping pod. If there is more than one pod remaining running in the platoon, check whether all the remaining pods are entering the same next path. If so, the platoon keeps going; otherwise, this platoon splits. Group the remaining pods according to their next paths. Let pods going to the same next path form a new platoon to run away. Any pod whose next path is different from all others' run away alone.

2.3. Uncertainty simulation

This section explains how we incorporated practical uncertainty into simulation through Monte Carlo simulation. For user requests, whether we would receive a request at this second and the requested journey, which is consisting of both a departure node and a destination node, are both decided by uniform random variables. Combining practical experience and system assumptions, we used a uniform random variable to set that, 7% users cannot show up at their departure nodes within 3 min after their pods' arrival, and those arriving in time would show up at a time between 10 s to 3 min equally likely after the pods' arrival. Then an identity recognition which takes 10 s would open the door.

Time for users to getting on board can be varied between seconds to minutes. In practice, those taking lots of luggage, wheelchair, stroller, children, older people or disabled people would need more time and it is these people who are very likely to occupy a large proportion of the potential users, as people travelling with few belongings may either take a taxi to save time or simply walk to their destinations. Considering this, we set the average time duration to be 20 s. It is natural and reasonable to use exponential distribution to simulate the time duration for loading or unloading passengers. Therefore the relevant time duration in our simulation follows an exponential distribution with mean 20 s plus a constant 5 s for door's opening and closing.

As mentioned in Section 2.2, our simulation considered the uncertainty in Condition 5, which requires pods can "see" each other. We used uniform random variables based on the distance between the two pods and their pod-path case to decide whether Condition 5 holds. In our setting, the closer they are, the more likely that they find each other, because the less likely that they can be blocked by other obstacles. We also considered in practice, it would be easier for pods to find each other when they are running on the same path. In other words, we set that pod-path Case 1 has relatively more chance than other two cases (while the distance is same).

Since the practical shared environment is rarely smooth like a rail, we considered small fluctuations in pod's speed. So the pod's speed was set to be normally 80% of the speed limit plus a small noise, without exceeding the speed limit. As we know, the noise is a normal random variable with mean 0. The standard deviation was set to be 0.06, this is because such setting can guarantee the random change of pod's speed from last second is usually not greater than 0.36 m/s.

After arriving to the destination, those who get on slowly usually also get off slowly, and vice versa. So for the same user request, we simulated the time to unload users through a normal distribution with mean equivalent to their loading time.

2.4. Map setting and user demand location pattern

This simulation study has two main parts. The first part is to investigate a specific problem: how deployment would affect pod platooning and the benefit of it. To eliminate the influence of randomness of user requests' arrivals, we set that fixed numbers of requests can be received in every second. To eliminate the influence of various map shapes on this problem, we used the uniform square grid maps, which is shown in Fig. 3a. On such a map, each edge is of the same length and nodes are equally distributed. We also set the location pattern of user demand is uniformly distributed. In other words, a journey can be any two different nodes on the map and each node can be requested equally likely.

The second part of this simulation study is to provide a practical estimation of the system performance. The map in Fig. 3b was designed to be more representative of real-world city layouts, with higher path density in the centre and sparser longer paths in the outer area. We set three zones and four hotspots. Zone 1 is the centre of the map and comprised of 25 nodes; zone 2 surrounding zone 1 has 68 nodes; zone 3 is the outer most zone containing 56 nodes. In addition to the 4 hotspots, the map has 153 nodes in total. The four hotspots represent key focal points for people in a city, for example, business centre, train and bus stations. The whole map is of 3.2 km×3.2 km. The centre area is of 400 m×400 m with each edge length 100 m. The inner and outer square areas have path lengths 200 m and 400 m respectively. The speed limits for the centre, inner and outer areas were set to be 9 km/h, 15 km/h and 22 km/h respectively.

On the uniform grids, user request would be rejected if there is no pod available. For the three density zone map, we added a waiting list

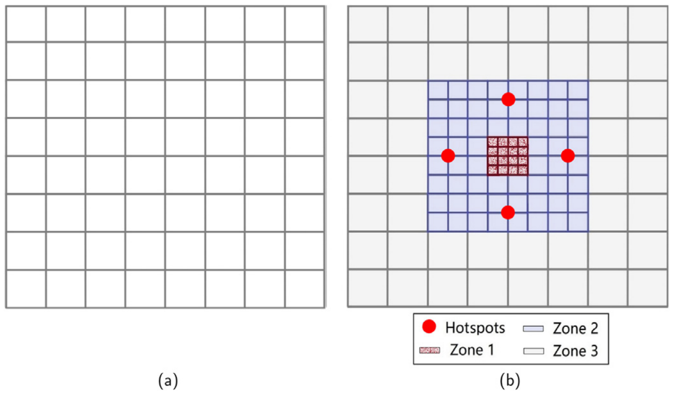


Fig. 3. Map configurations used for simulation; (a) Uniform grid; (b) Three density zone map.

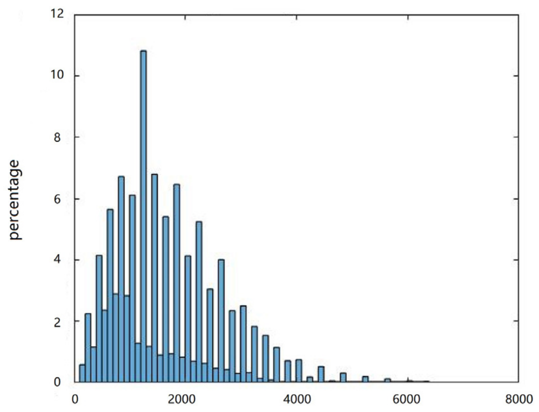


Fig. 4. Distribution of distances of requested journeys.

for user requests to the simulation algorithm. When a pod becomes available, waiting requests will be served on a "first come first served" basis.

We designed the location pattern of user demand such that, hotspots and centre get higher travel frequencies than other places. Specifically, 1) the probability that each hotspot get requested is 0.06; 2) zone 1 would be travelled from/to with probability 0.2 and each node would be requested equally likely, which means the probability of getting requested for each node in zone 1 is 0.008; 3) other 124 nodes would be requested with probability 0.56 in total and each node would be requested equally likely, which means each remaining node has the request probability 0.0045.

The histogram in Fig. 4 shows the distribution of journey distance from 50,000 simulated requests. Around 11% requested journeys have distance between 1.2 km and 1.3 km. Journeys over 4 km are negligibly few. Under this location pattern, most journeys are within 2 km and the average journey distance is 1579 m.

2.5. User demand time pattern

To ensure the second part of the simulation study is as close to practice as possible, we not only designed a practical map (Fig. 3b) and a location pattern of user demand, but also simulated user requests' arrivals according to Poisson processes and practical time patterns.

For the simulation of requests' arrivals, we generated a sequence of exponential random numbers, to represent the time interval between two consecutive user requests. Since our simulation time unit is one second, we discretised the time point. For example, if a user request arrives at time 2.3 s, then we treat it as the 3rd second.

Our user demand to travel should have a similar time pattern as the road and rail traffic. This user demand was generated based on a previous study [36], which used traffic count data of all motorised vehicles in Coventry city centre in a 10 year period. The data sources are OpenStreetMap (OSM) and the Department of Transport [37]. In addition, we also incorporated visit counts from Google map into our time pattern setting. In this way, we set the time pattern of user demand for weekday (Mon–Fri) and weekend (Sat–Sun). Fig. 5 shows the Poisson rate, which means the average number of requests we can receive in one second, for each hour from 06:00 to 24:00.

Fig. 5 a presents time pattern of user demand for a weekday. We set rush hours in the morning and afternoon. In 06:00–07:00 and 08:00–09:00, we can expect to receive one request in 2 s. In 07:00–08:00, user demand would reach the highest level of one request per second. After 09:00 the demand decreases. At noon, the average time intervals between two consecutive user requests are 8 s. In 16:00–17:00 and 17:00–18:00, we can expect a request in 1.5 s and 2 s respectively. Later user demand would decrease.

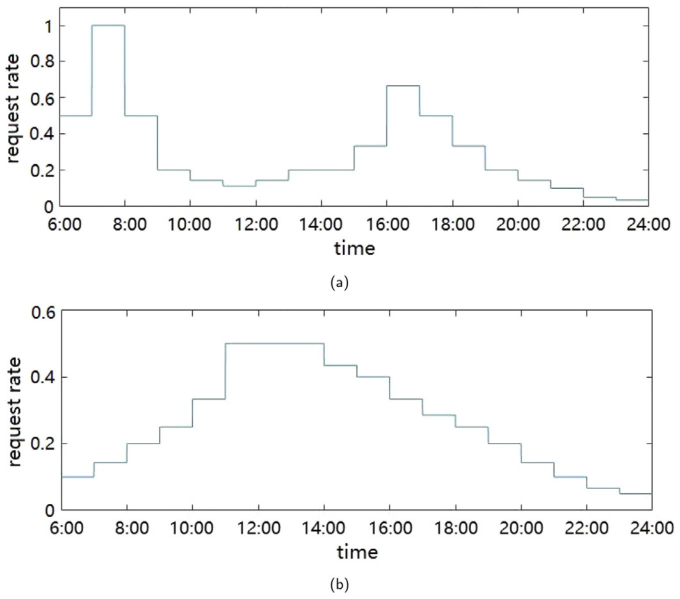


Fig. 5. Time pattern of user demand from 06:00 to 24:00 on weekday and weekend. (a) Poisson rate of requests per second for weekday; (b) Poisson rate of requests per second for weekend.

Fig. 5 b shows the time pattern of user demand for weekend. Different to weekday, there is no rush hours in the morning or afternoon on weekend. Instead, user demand becomes highest around noon. In 11:00–14:00, we can expect to receive a request in average 2 s, which becomes 2.5 s in 14:00–16:00. Then user demand would drop. At late night, we would have slightly higher demand on weekend than on weekday.

2.6. Simulation algorithm

According to the Poisson rate set in Fig. 5, we simulated the arrivals of user requests as a non-homogeneous Poisson process. Algorithm 1 shows how we simulated user requests in terms of when

```
foreach hour = 06:00 to 24:00 do
    repeat
        simulate the time interval between two consecutive
        requests by generating an exponential random number
        with rate for this hour ;
    until sum (time intervals) > an hour;
end
convert the sequence of time interval generated above as how
many requests in every second;
foreach request do
    choose the departure node and destination node randomly
    according to the location pattern;
end
```

Algorithm 1: Generate user requests.

we receive the request and the requested journey.

At each second of the simulation, the dynamic system is updated according to Table 4. Steps 3–6 are specified in Algorithm 2–5 in Appendix.

Given user requests, the system would start from finding pods to serve the requests, which is the Step 1 in Table 4. Then what will happen before departure of the requested journey is simulated in Step 2. Pod's speed depends on its running mode, so the first thing once a pod can move is to check platooning opportunity, which is Step 3. If a pod can join other pods, some status needs to be determined immediately, for example, the pod-path case, it is front or rear pod. The speed setting for

Table 4
Algorithm steps.

Step 1:	**Pod dispatch** For user request received at this second, find the available pod and dispatch the closest one to the departure node, otherwise reject this request (for uniform grid map) or add this request to the end of the waiting list (for the zoned map).
Step 2:	**Before departure** If users fail to show up in time at the departure node, cancel the request and set the pod free. Otherwise, users would get on board, listen to the safety announcement then depart.
Step 3:	**Check platooning opportunity** For all running pods, check whether they can platoon according to the platooning conditions (in Section 2.2).
Step 4:	**Set platooning** For joining pods in prospective platoons (running mode (2)), determine which pods are at front or rear, then set their speeds and check whether the platoon is formed successfully at this second according to the distance between the front and rear pods.
Step 5:	**Pods running forward** For all running pods, set speeds and locations at next second. Set speed for platoons (running mode (3)) and pods running alone (running mode (1)) at this second. Then set location for all running pods at this second.
Step 6:	**Check path update** Check whether any pod is passing a node and determine what to do. Pod within 4 m to its next node is considered to be passing that node. Then pod will enter the next path or terminate if that node is the destination.
a:	If a joining pod (in a prospective platoon) is passing a node and is not terminating, update the pod-path case for this prospective platoon; if it is terminating, drop the pod from the prospective platoon, check the remaining pods in this prospective platoon and update their running modes.
b:	If a platoon is passing a node, check whether any pod of it is terminating. If so, check whether the platoon has more than one pod remaining. If so, check and group remaining pods according to their next paths to form new (small) platoons directly from the previous bigger platoon. Update running mode for pods entering their next paths alone.
Step 7:	**After arrival** After arrival at the destination, wait for users to get off and then set the pod free.

a joining pod is complex, as it depends on the distance to other pods. These are determined in Step 4. Finally we can let all running pods move forward by straightforward settings and this is Step 5. When a pod is passing a node, many corresponding attributes or status, such as it is terminating or not, the platooning is successful or failed, change of pod-path case, whether other pods in the same prospective platoon can still try to join as before, may all change consequently. The complicated changes caused by this are determined in Step 6. If the pod is arriving at the destination node, Step 7 tells roughly the pod's job before it becomes available to other users, which goes back to step 1.

In Step 6, the distance tolerance for a pod to reach its next node is 4 m. This is because the simulation time unit is 1 s and a 4-m tolerance can avoid that when pod runs fast, both distances before and after its passing are too far to be detected. The maximum legal speed limit 24 km/h is equivalent to 6.667 m/s. So a circle with radius of 4 m can always cover the pod's location at the second right before or after its passing, even at the maximum speed.

In addition, the simulation tells pod's location by both its 2D location coordinates and tracking the nodes they've passed, as the route in this study is a sequence of adjacent nodes. Since rerouting is not considered, tracking nodes can guarantee that temporary changes of pod's direction such as bypassing or turning around will not confuse the system. This is also helpful when some paths are very close or an intersection node is connecting several paths from close directions.

2.7. Limitations

As this study is being conducted as part of a project ending soon, there are some limitations worth discussing. For example, in order to simplify the user generation process and avoid scheduling problem, we did not consider users' pre-booking, pre-allocation of pods (to hotspots), allocation of supervisors and pod charging. To avoid the complexity on optimisation, we did not consider request swap, which means that after a user request was assigned to a pod, another pod closer to the pickup node becomes available, then the closer pod will take that request instead.

We realise that in practice, supervision work may be more than necessary. For example, when a platoon splits to several pods, each pod or new smaller platoon will need a supervisor, for simplicity we did not considered that these supervisors should actually get ready before a platoon splits in the simulation. However, these are negligible compared to the total supervision cost.

Due to time limits, the low acceleration rate was considered only for platooning, and more other uncertainties such as pedestrian interference was ignored but corresponding work may be set later.

3. Results and discussion

We used Monte Carlo method to simulate what would happen in the transport service system per second, given the map setting and user demand patterns, adopting the designed pods' platooning rules, considering practical uncertainties. Simulation of the dynamic system was mainly from the pods' travelling point of view, but also included the influence of users' behaviours. The transport service is ready for operation in our simulation, let us start experiment on it.

Starting by a pre-trial study, we will firstly identify how deployment could affect platooning and the consequent benefit, through computer simulation and statistical regression analysis, in Section 3.1; then we will find a suitable fleet size before conducting trials for one day of service, according to different patterns of user demand for weekday and weekend, in Section 2.5.

3.1. Study using uniform grid map

In this section, we investigate the influencing variables about deployment on platooning, using the uniform square grid maps. We focus

Table 5
12 simulation cases for different pod densities on three maps.

Map (km)	Map area (km^2)	Requests per second	Number of pods
1×1	1	1	50, 100, 150, 200
3×3	9	9	450, 900, 1350, 1800
5×5	25	25	1250, 2500, 3750, 5000

on supervision cost reduced by platooning, which is of great interest to the industry. To quantify this benefit, we calculated the overall reduced supervision time in percentage, which is defined as

$$\text{overall reduced supervision time in percentage} = \frac{\text{total reduced supervision time}}{\text{total supervision time without platooning}}.$$

Total reduced supervision time is the sum of reduced supervision time in every second over the simulation time. In every second, the fleet has:

$$\text{reduced supervision time} = \text{supervision time without platooning} - \text{supervision time with platooning}.$$

Since we assume that supervisors would not work on a pod stopping at a node, the supervision time without platooning is equivalent to the fleet's travelling time. In one second, this is equivalent to the number of running pods.

Without prior knowledge on map size and fleet size for platooning, we conducted a pretrial study by simulating different fleet sizes running on three sizes of maps for different levels of user demand. We found that when user demand is fixed at one request per second, platooning can reduce almost 18% overall supervision time for a fleet of 300 pods running on a small map of 1 km×1 km. However, this benefit is reduced to only 1.2% when we have 1600 pods running on a large map of 15 km×15 km. Since the benefit of platooning on such a large map is negligible, we decided not to consider such a large area in the formal study. The dramatic difference inspired us that, it is the pod density with respect to map area that makes a difference instead of the number of pods. Therefore, we conducted the following study.

We simulated different pod densities on different sizes of maps but using the same request densities. The maps we used for this problem are uniform grid of edge length 100 m. We used three maps of 1 km×1 km, 3 km×3 km and 5 km×5 km. A larger area should cover more people and thus should have higher user demand. We set the user demand to be one request in every 1 km^2 in every second. This is a very high user demand over the fleets' capacities in the trials. So pods are almost always busy in service and most are travelling. For each map, pod densities are chosen to be 50, 100, 150 and 200 pods in every 1 km^2. In total, we simulated 12 cases shown in Table 5.

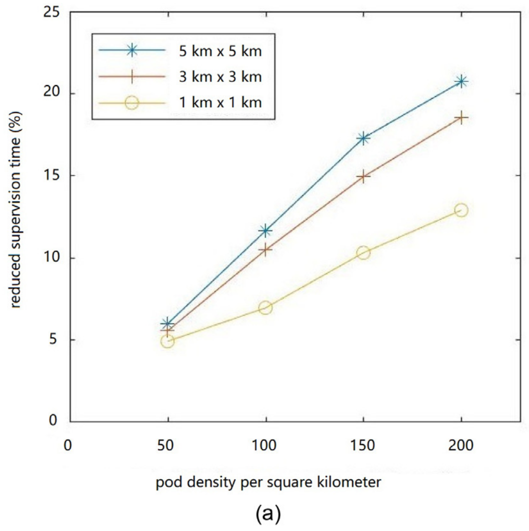
To obtain steady results, we simulated 30 h of the system. Results are shown in Fig. 6a. By comparing the three lines, we found that larger maps can yield more supervision cost saving, under the same request density and pod density. By looking at each line, we found that a higher pod density can reduce more supervision time.

Based on the simulation result, we obtained the following regression model using RStudio after lots of trials:

$$\text{Overall reduced supervision time in percentage} = 2.8 * log(1.6 + Dens\_edge) * log(Num\_jrne) - 1.5 * log(Num\_jrne) + 2.4.$$

The variable Num_jrne means number of journey choices, namely how many ways to choose two different nodes from the map, as departure and destination nodes of a journey. For example, the map of 1 km×1 km has 121 nodes, then there are 14,520 $(121 * 120)$ journey choices. The variable $Dens\_edge$ means pod density with respect to edge, namely how many pods we have for each edge, i.e., for every 100 m. Table 6 shows the numbers of nodes, edges, journey choices and pod density with respect to edges.

The regression model indicates that, supervision time reduction is negatively related to the number of journey choices, positively related to the product of the pod density with respect to edge and the number of journey choices. Moreover, the model tells us two points. Firstly, the more pods we have over every 100 m on the map, the more supervision time can be reduced by platooning. Secondly, a larger map has two opposite influences: on one hand, on a larger map, pods can deliver longer journeys, so during the journey they have more chance to meet other pods sharing some common paths, which can lead to



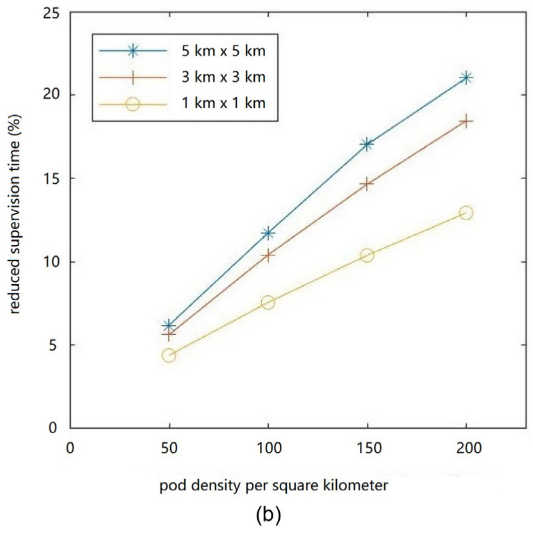


Fig. 6. Overall reduced supervision time in percentage for different pod densities on three maps; (a) Simulation result; (b) Estimation result of modelling.

Table 6
Relevant attributes of the three maps.

Map (km)	Nodes	Edges	Journey choices	Pod density w.r.t edge
1×1	121	220	14,520	0.227, 0.455, 0.682, 0.909
3×3	961	1860	922,560	0.242, 0.484, 0.726, 0.968
5×5	2601	5100	6,762,600	0.245, 0.490, 0.735, 0.980

more platooning; on the other hand, on a larger map, users would have more different journey choices, so relatively speaking, pods are more likely to run on different paths, which can reduce platooning opportunity.

The last column of Table 6 indicates that, while pod density with respect to map area is equal, a larger map can have slightly higher pod density with respect to the edge. This explains why in Fig. 6a, a larger map can yield more supervision time saving.

Estimates of the overall reduced supervision time in percentage given by the model are shown in Fig. 6b. The adjusted R-squared of the model is 0.9969, very close to 1. This means the model explains 99.69% variations of the supervision cost saving. The model has a very good fit and gives accurate estimates. We also checked that the model residuals are acceptable, as residuals are normally distributed with mean 0 and no obvious irregular pattern was found.

A suggestion from the model is, to reduce supervision cost through platooning, we need to increase the travelling pod density with respect to edge, considering that the high user demand keeps most pods travelling. This can be done by increasing the number of travelling pods and reducing the total path length. The influence of pod density with respect to edge was also confirmed from our simulation trials using maps of different path densities but same map size and fleet size. Details on it is not given here due to paper length.

3.2. *Study using multiple density zone map*

As for the second part of this simulation study, we used the three density zone map in Fig. 3b and location pattern stated in Section 2.4 to predict system performance and user experience. Since the user demand from 06:00 to 24:00 can vary from one request per second on average, to two requests per minute on average, we found the suitable fleet sizes for different levels of user demand. Results are shown in Table 7. The first column shows the average time interval between two consecutive requests in time unit second. (Due to time limits, for this table, pods' regular speeds were set to be 90% of the speed limits.)

These fleet sizes can produce, not only a high fleet efficiency but also acceptable waiting time for users. In this paper, we say a fleet has a high efficiency if most pods are being used and only a few pods are idle. Acceptable waiting time used for Table 7 means that, users who request within the first hour of service can be served immediately, and requests after two hours can still be served within 15 min. The map of 3.2 km×3.2 km may be used in medium cities such as Coventry and Milton Keynes. This means when shared environments become wide enough in the future, the list can provide a reference for fleet size.

To balance the system efficiency and user experience, we tried simulating 460 pods to serve one weekday, but this fleet size is too small and lead to four-hour waiting time for users. Therefore we chose the fleet size to be 500 pods and conducted trials for one day of service in the following two sections.

Table 7
Suitable fleet sizes for user demand from low to high levels.

Average time interval between requests (s)	Average number of requests in 1 min	Suitable fleet size
1	60	750
2	30	400
3	20	270
4	15	200
5	12	160
6	10	140
8	7.5	100
10	6	80
20	3	40
30	2	27

3.2.1. *Weekday trial*

This section shows our simulation result of 500 pods in a weekday of service from 06:00 to 24:00. The normal speed setting was used, namely the pods' regular speeds are 80% of the speed limits.

Fig. 7 a shows the dynamic fleet efficiency in a weekday from 06:00 to 24:00. Fleet efficiency has similar pattern as user demand shown in Fig. 5a. During the rush hours in the morning and afternoon, fleet efficiency keeps at 100%. This means all pods are busy in service in rush hours. During the off-peak hours between 11:00 and 15:00, only 20% to 30% pods are in service and most pods are idle. At 20:00, only 20% pods are in service. Then fleet efficiency tends to drop.

Fig. 7 b shows the total journey length the fleet travelled in each hour. During the rush hours in the morning and afternoon, the fleet of 500 pods can travel 5000 km in an hour. Fleet journey length has similar pattern as fleet efficiency and user demand.

Fig. 8 shows the dynamic proportions of pods running in different modes in a weekday. The blue, green and red lines represent pods running in mode (1), (2) and (3) respectively, namely running alone, joining and running in a platoon respectively, as defined in Section 2.2. The figure shows that there are always more pods running alone than joining or running in platoons. During rush hours in the morning and afternoon, over 20% pods are running in a platoon, roughly 30% pods are cooperatively joining and almost 50% pods are running alone. At noon, user demand drops and most pods are idle, so only less than 5% pods are in platoons. After 22:00, pods joining or in platoons are negligibly few.

Fig. 9 shows the dynamic benefit of platooning with 500 pods in a weekday. Fig. 9a shows the percentage of reduced supervisors,

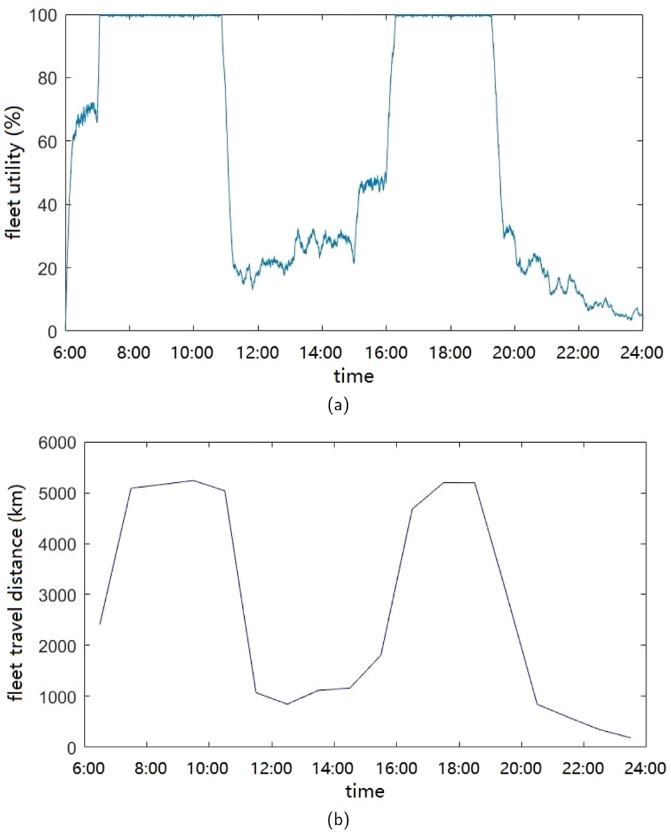


Fig. 7. Fleet efficiency and hourly travel distance in a weekday from 06:00 to 24:00; (a) Fleet efficiency; (b) Number of kilometres the 500 pods travelled in each hour.

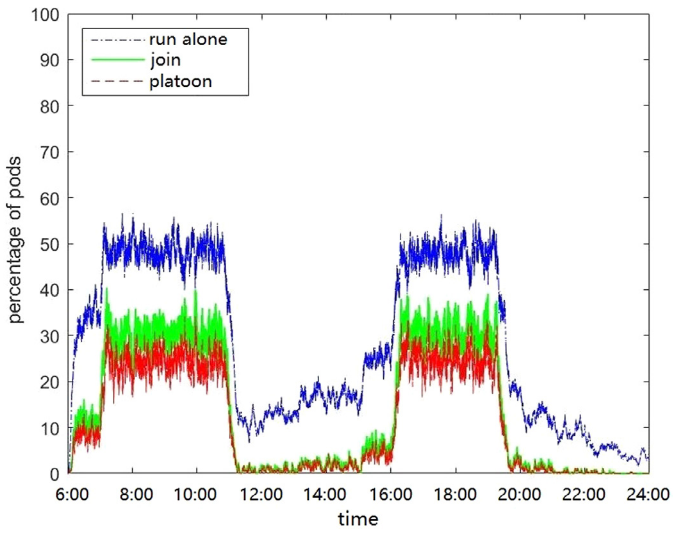


Fig. 8. Percentage of pods running in three modes in a weekday. Blue dash–dotted line: pods are running alone – mode (1); Green thick solid line: pods are joining – mode (2); Red dashed line: pods are running in a platoon – mode (3).

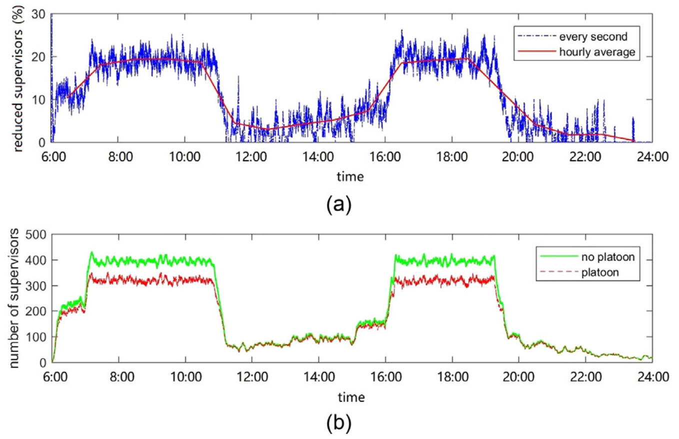


Fig. 9. Reduced supervisors needed in a weekday; (a) Percentage reduction in supervisors needed; Blue dash–dotted line: the reduced supervisors needed at every second; Red thick solid line: hourly average of the reduced supervisors needed; (b) Number of supervisors needed with and without platooning; Green thick solid line: supervisors needed without platooning; Red dashed line: supervisors needed with platooning.

with blue dash–dotted line and red thick solid lines respectively representing the reduced percentage in every second and the average in each hour. In Fig. 9b, the red dashed line and green thick solid line represent how many supervisors needed with and without platooning respectively. It can be seen that supervision reduction has similar time pattern as the fleet journey length, fleet efficiency and user demand.

During rush hours in the morning and afternoon, we need approximately 320–330 supervisors, approximately 20% supervisors can be saved by platooning, and we would need 400 supervisors without platooning. Around noon, the supervision cost saving by platooning is negligible and the hourly average is only 5%. At 20:00, only 100 pods are enough to serve the relatively low user demand. After that, supervisors needed decreases as user demand decreases and fewer pods are travelling.

Fig. 10 shows how many minutes the users need to wait on average since their requests are received, until available pods are dispatched, until pods have arrived at the departure nodes, until pods carrying users depart. When more than one request were received at the same time, we calculated the average of waiting time for these requests.

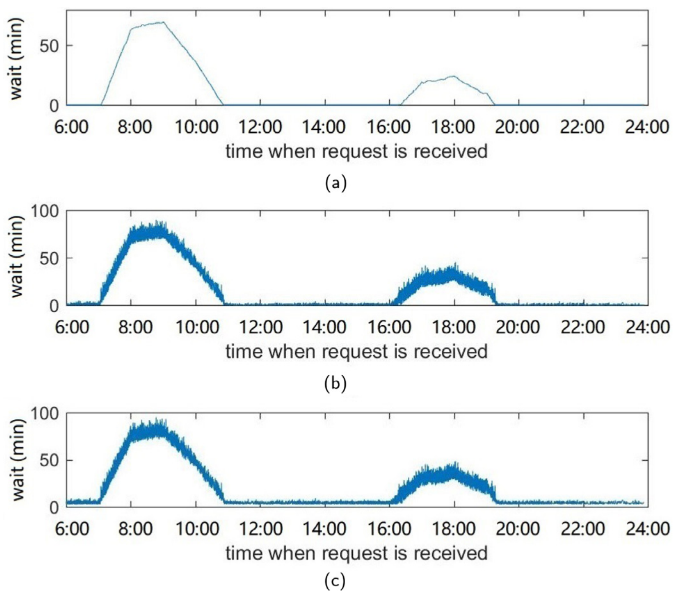


Fig. 10. Users' waiting time for pods' dispatch, arrivals to the departure nodes and departure, in a weekday; (a) Waiting time for pods' dispatch; (b) Waiting time for pods' arrivals to the departure nodes; (c) Waiting time for departure.

Waiting time for pods' arrivals for picking up and departure both change rapidly in every second and look like very thick lines. This is because these heavily depend on the distance between departure node of the new request and destination of the finishing request. Departure time also heavily depends on how long users can show up and get on board. During the rush hours, all pods are fully used and busy in service. So the time when any pod becomes available heavily depends on the fleet service capacity, instead of on probability or uncertainty. This is why the waiting time for pods' dispatch is a normal line with only tiny fluctuations. Waiting time for pods' arrivals for picking up and departure both change rapidly in every second and look like very thick lines. This is because these heavily depend on the distance between departure node of the new request and destination of the finishing request. Departure time also heavily depends on how long users can show up and get on board. During the rush hours, all pods are fully used and busy in service. So the time when any pod becomes available heavily depends on the fleet service capacity, instead of on probability or uncertainty. This is why the waiting time for pods' dispatch is a regular line with only tiny fluctuations. Notice that waiting time for departure does not start from 0. This is because, after pod's arrival to the departure node, it needs to wait for the users to show up and get on board, then broadcasts safety announcement, before it can depart.

The three sub-plots all indicate that, users have to wait for very long time during rush hours in the morning and afternoon. Waiting time for dispatch can be longer than an hour in the morning but less than 30 min in the afternoon, because user demand is higher in the morning. In contrast, user requests during off-peak times can be served immediately. If users request a pod at 07:30, then they would need to wait for 30 min before any pod can be dispatched to them. If a request is made at 17:00, then the user would need to wait for 20 min before a pod becomes available, and 40 min before departure.

In summary, platooning during rush hours on weekdays can indeed reduce substantial supervision cost – approximately 20%. Supervision cost saving, platooning opportunity and fleet efficiency all have similar time pattern as that of user demand. We notice that, the majority of pods are idle at off-peak midday, while users have to wait for a long time during rush hours, so the fleet operator may want to adjust user demand or deployment to balance the efficiency and user experience. Besides, the service after 20:00 may be cancelled due to low demand.

3.2.2. *Weekend trial*

This section shows our simulation result of 500 pods on weekend from 06:00 to 24:00, with the same speed setting used in Section 3.2.1.

Fig. 11 shows the dynamic fleet efficiency on weekend from 06:00 to 24:00. Fleet efficiency on weekend has similar pattern as user demand shown in Fig. 5b. On weekend, there is no rush hour in the morning or afternoon, instead we expect to receive more user requests around noon, when fleet efficiency can reach the highest level, 80% approximately. This means 20% pods are still not used in busiest periods on weekend, and at most 400 pods are in service. Before 10:00 and after 18:00, less than 40% (200 pods) are used. Recall that the fleet can travel at most 5000 km in an hour on weekdays, while this is reduced to 3300 km on weekends due to lower user demand.

Fig. 12 shows the dynamic proportions of pods running in different modes on weekend. Similarly to weekday, there are always more pods running alone than other two modes on weekend. During the busiest hours around midday, pods' have the most running and platooning. Approximately 12% pods are running in platoons, 17% pods are joining and 40% pods are running alone. Platooning in other time periods is negligible. This is in contrast to the weekday, when 25% pods are running in platoons during rush hours, which is reduced to 12% for weekend around noon. On weekend, we have fewer journeys and hence less platooning.

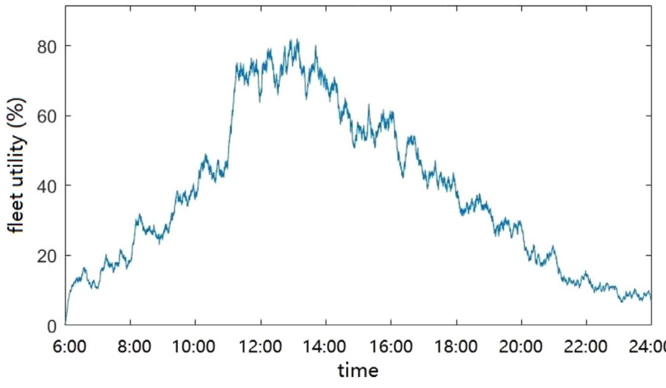


Fig. 11. Fleet efficiency on weekend from 06:00 to 24:00.

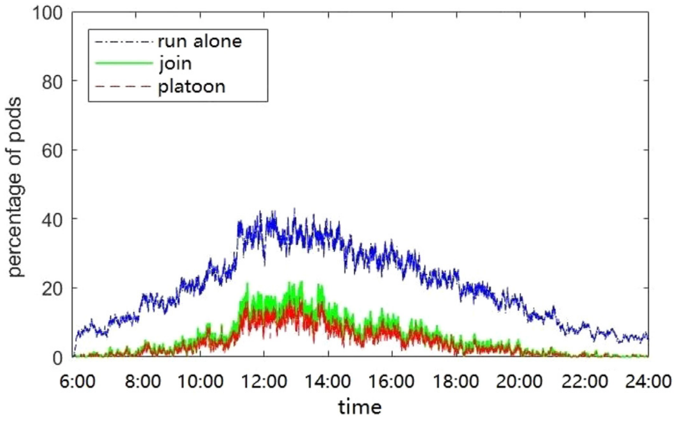


Fig. 12. Percentage of pods running in three modes on weekend. Blue dash–dotted line: pods are running alone – mode (1); Green thick solid line: pods are joining – mode (2); Red dashed line: pods are running in a platoon – mode (3).

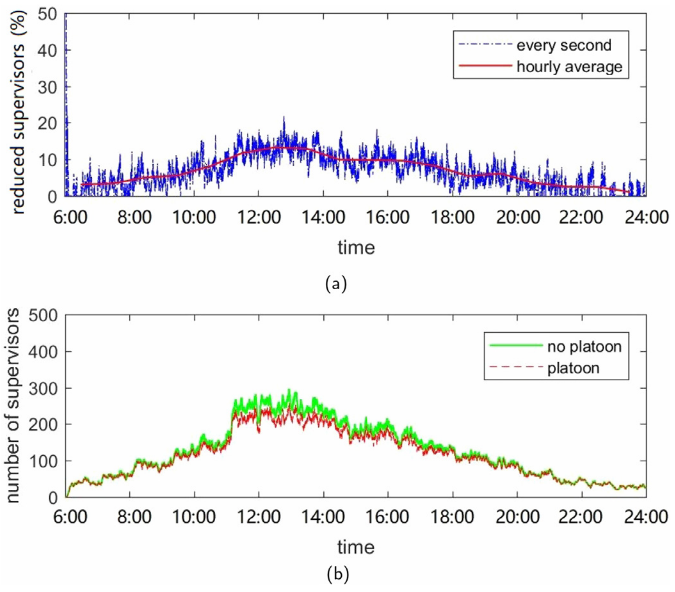


Fig. 13. Reduced supervisors needed on weekend; (a) Percentage reduction in supervisors needed; Blue dash–dotted line: the reduced supervisors needed at every second; Red thick solid line: hourly average of the reduced supervisors needed; (b) Number of supervisors needed with and without platooning; Green thick solid line: supervisors needed without platooning; Red dashed line: supervisors needed with platooning.

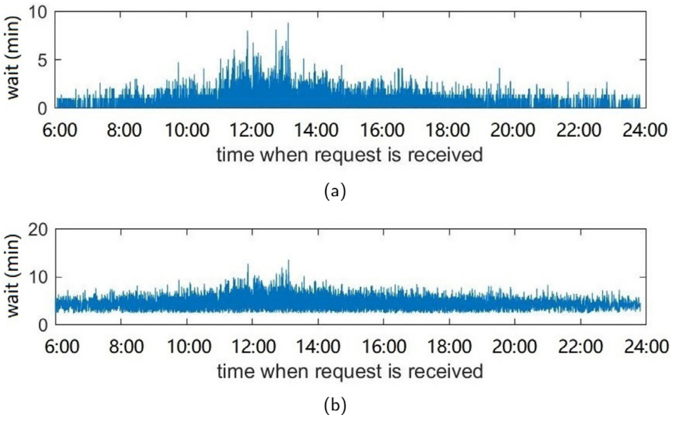


Fig. 14. Users' waiting time for pods' arrivals to the departure nodes and departure, on weekend; (a) Waiting time for pods' arrivals to the departure nodes; (b) Waiting time for departure.

Fig. 13 shows that the dynamic benefit of platooning with 500 pods on weekend. Demand for supervisors is high when pods travel the most, namely when user demand is high, i.e., around noon for weekend. Approximately 270 supervisors would be needed if pods do not share supervisors. Platooning can reduce this number to 220. This is in contrast to the weekday, when we can save up to 25% of the supervisors with hourly average of 20%. On weekend, these are reduced to 20% and 14% respectively, due to lower demand and less travel.

Fig. 14 shows how many minutes the users need to wait on weekend. The simulation result shows that all user requests can be served immediately. In other words, users do not need to wait for pods' dispatch. This is because we have more than enough pods.

Recall that fleet efficiency in Fig. 11 shows 500 pods are not fully utilized. Fig. 14a shows usually it would take 2–5 min for a pod to arrive at the departure node after the request is received. This waiting time is still less than 10 min at the busiest midday. Fig. 14b shows that usually users can depart for destinations within 10 min. This is much better than the rush hours on weekdays.

In summary, compared to weekday, we would need fewer pods, have less travel, less platooning and less supervision saving on weekend; although 14% supervisors can be saved by platooning around noon; the benefit is that users can be served immediately on weekend and avoid long waiting time as in rush hours on weekdays.

Fig. 13 b shows fewer than 300 pods are running at the same time, so we simulated 300 pods to serve the weekend and but found it is not enough. This is because some pods in service are waiting at nodes for users. Then according to the fleet efficiency, which indicates at most 80% pods are in service, we estimate that 400 pods should be enough.

In addition, we also tried different speed settings. Recall that in Sections 3.2.1 and 3.2.2, pods' regular speeds are 80% of the speed limits – 9 km/h, 15 km/h and 22 km/h for zone 1, zone 2 and zone 3 respectively. We changed the speed setting to be 90% of the speed limits – 12 km/h, 18 km/h and 24 km/h, for weekday trial. Our simulation results show obvious differences for afternoon rush hours: fleet efficiency drops from 100% to 85%; percentage reduction in supervisors needed drops from 20% to 12%; user requests can be served immediately. This is because the faster travelling increases fleet's service capacity, to be greater than user demand. Under previous speed setting, the service capacity is lower than the user demand in the afternoon. With or without platooning, both numbers of supervisors needed drop slightly. The number of supervisors needed with platooning drops from 320–330 to 300, although slightly less pods are running in a platoon. In the morning, fleet's hourly travel distance increases from 5000 km to 6000 km; users' waiting time to be served drops from 70 min to 35 min. We also changed the speed setting to be 60% of the speed limits – 9 km/h, 15 km/h and 22 km/h, for weekday trial. Our simulation results show more platooning and supervision savings, but at the expense of requiring more supervisors, and users' waiting time is unacceptably long.

In summary, we found that pods' faster travel can lead to shorter waiting time for users, less picking up, less travel, requires fewer supervisors although less platooning is yielded. Less travel is because faster delivery means shorter journey times and therefore more available pods. When many pods are available, the closest pod would be chosen and dispatched to take the request.

4. Conclusions

This paper presents a simulation study of a low-speed electric autonomous transport system, consisting of pods, running on shared urban environments, with ability to cooperate to form platoons. This is the first simulation study on supervision cost reduction by EAV platooning. The on-demand transport strategy adopted for this study, is designed to help address the "last mile" challenge, to ensure accessible convenient public transport.

The main contribution of this paper is that it provides a framework for low-speed AV platooning in shared urban environments and a prospect for opportunistic pod platooning without prior planning in the real world.

We found from the simulation that platooning can reduce substantial supervision cost. Specifically, approximately 20% supervision cost during peak times on weekdays and 14% supervision cost at midday on weekends can be saved. For a weekday on the multiple density zone map, we found that 400 supervisors are sufficient for a fleet of 500 pods without platooning, as not all pods in service are running and pods waiting for users do not need a supervisor. This can be reduced further to only 330 supervisors when platooning is employed.

An most important benefit of platooning is supervision cost reduction, its influencing factors of it is identified by computer simulation and statistical modelling. Corresponding suggestions were provided: increasing travelling pod density with respect to total path length, which can be done by increasing user demand, and reducing journey choices. Various practical uncertainties such as noise and users' absence were considered. Arrivals of requests were simulated by reasonable Poisson processes. Practical time and location patterns of user demand were set. System performance and user experience were evaluated from several aspects. Users usually can be served immediately except rush hours on weekdays, when they need to wait for long. We also provided a reference of suitable fleet sizes for different levels of user demand, according to system efficiency and user experience.

In addition, our simulation result suggests that the service supplier may want to change some business strategies before deployment, for example, to balance the fleet efficiency and users' waiting time caused by current user demand pattern, and to cancel the service after 20:00 due to low demand.

We incorporated practical uncertainties into the simulation by Monte Carlo method, using suitable stochastic models based on real data. The algorithms built for this study can be used as a platform for further investigation of practical uncertainties in transport service systems.

As the transport system studied in this paper is new and still under development, some problems are still to be addressed, which would require further research to be conducted. For example, users' pre-booking, pods' pre-allocation, setting main roads and optimising pods' routes without disturbing user experience. Bringing this service to the real world requires contributions from various fields.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work has been carried out as part of the Self-organising Wide area Autonomous vehicle Real-time Marshalling (SWARM) project. The project was funded by Innovate UK, the UK's innovation agency, under the grant ref. 103287, and is a collaboration between RDM Ltd., University of Warwick, and Milton Keynes Council. All authors would like to thank the Innovate UK for funding and thank the editors and reviewers for their very helpful comments and suggestions.

Appendix A

```
for every two pairs of pods running alone do
    if two pods can join then
        create a prospective platoon;
    end
end
for every single pod and platoon do
    if pod and platoon can join then
        add the pod into the platoon to form a bigger prospective
        platoon;
    end
end
for every two pairs of platoons do
    if two platoons can join then
        add one platoon to the other to form a bigger prospective
        platoon;
    end
end
```

Algorithm 2: Simulation algorithm for Step 3.

```
foreach prospective platoon do
    if pod-path Case 1 then
        determine which pods are at front and rear;
        set speeds and update platooning status according to their distance;
        front pods speed($t$) = max(0, max(0.5, front pods speed($t$−1)-0.8) + noise);
        calculate distance between front and rear pods;
        if distance > 8 m then
            rear pods speed($t$) = min(min(rear pods speed($t$−1) + 0.8, speedlimit) + noise, speedlimit);
        else if distance > 5 m then
            rear pods speed($t$) = min(5 m/s + front pods speed($t$), speedlimit);
        else if distance > 3 m then
            rear pods speed($t$) = min(distance + front pods speed($t$), speedlimit);
        else
            this prospective platoon becomes a platoon;
        end
    else if pod-path Case 2 then
        for every pod in the prospective platoon do
            speed($t$) = min(80%*speedlimit + noise, speedlimit);
        end
    else if pod-path Case 3 then
        determine which pods are at front and rear;
        set speeds and update platooning status according to their distance;
end
```

Algorithm 3: Simulation algorithm for Step 4.

```
foreach prospective platoon do
    foreach pod in the prospective platoon do
        location($t$) = location($t$-1) + speed($t$);
    end
end
foreach pod running alone do
    speed($t$) = min (80% * speedlimit + noise, speedlimit);
    location($t$) = location($t$-1) + speed($t$);
end
foreach platoon do
    calculate speed and location for the leading pod:
    speed($t$) = min (80% * speedlimit + noise, speedlimit);
    location($t$) = location($t$-1) + speed($t$);
    for every pod in the platoon do
        speed($t$) = leading pod speed($t$);
        location($t$) = leading pod location($t$);
    end
end
```

Algorithm 4: Simulation algorithm for Step 5.

```
foreach running pod do
    if distance { location(t), next node } < 4 m then
        if next node is the destination then
            this pod will terminate at the destination;
            if this pod is in a platoon or prospective platoon then
                drop this pod from the platoon or prospective platoon;
            end
        else
            this pod will enter the next path;
        end
    end
end
foreach platoon do
    if the platoon is passing a node then
        group remaining running pods according to their next paths;
        foreach group do
            if this group has only one pod then
                this pod leaves alone;
            else
                these pods become a platoon;
            end
        end
    end
end
foreach prospective platoon do
    if both parts are passing a node then
        check platooning opportunity according to pods' current and next paths;
        let corresponding pods become a platoon or a prospective platoon;
        update pod-path Case 2 to be Case 1;
    else if only front part is passing a node then
        if pod-path Case 1 then
            check whether this platooning fails due to front part is terminating;
            check whether any pods can cooperate to join as pod-path Case 3;
            check whether any pods from the same (front or rear) part can become a platoon;
        update pod-path Case 2 to be Case 3;
        else if pod-path Case 3 then
            check whether this platooning fails according to Section 2.2;
            check whether any pods from the same (front or rear) part can become a platoon;
    else if only rear part is passing a node then
        update pod-path Case 2 to be Case 3;
        update pod-path Case 3 to be Case 1;
end
```

Algorithm 5: Simulation algorithm for Step 6.

References

[1] M.D. Yap, G. Correia, B.V. Arem, Preferences of travellers for using automated vehicles as last mile public transport of multimodal train trips, Transp. Res. Part A: Policy Pract. 94 (2016) 1–16, doi:10.1016/j.tra.2016.09.003.

[2] N. Tilahun, P.V. Thakuriah, M. Li, Y. Keita, Transit use and the work commute: analyzing the role of last mile issues, J. Transp. Geogr. 54 (2016) 359–368, doi:10.1016/j.jtrangeo.2016.06.021.

[3] R. Woodman, K. Lu, M.D. Higgins, S. Brewerton, P.A. Jennings, S.A. Birrell, Gap acceptance study of pedestrians crossing between platooning autonomous vehicles in a virtual environment, Transp. Res. Part F: Traffic Psychol. Behav. 67 (2019) 1–14, doi:10.1016/j.trf.2019.09.017.

[4] R. Woodman, K. Lu, M.D. Higgins, S. Brewerton, P.A. Jennings, S.A. Birrell, A human factors approach to defining requirements for low-speed autonomous vehicles to enable intelligent platooning, in: Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), 2019, pp. 2371–2376, doi:10.1109/IVS.2019.8814128. Paris, France

[5] D.J. Fagnant, K.M. Kockelman, Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in Austin, Texas, Transportation 45 (2018) 143–158, doi:10.1007/s11116-016-9729-z.

[6] B. Caulfield, Estimating the environmental benefits of ride-sharing: a case study of dublin, Transp. Res. Part D: Transp. Environ. 14 (2009) 527–531, doi:10.1016/j.trd.2009.07.008.

[7] J. Conesa-Muñoz, M. Gonzalez-de Soto, P. Gonzalez-de Santos, A. Ribeiro, Distributed multi-level supervision to effectively monitor the operations of a fleet of autonomous vehicles in agricultural tasks, Sensors 15 (2015) 5402–5428, doi:10.3390/s150305402.

[8] L.A. Pipes, An operational analysis of traffic dynamics, J. Appl. Phys. 24 (1953) 274–281, doi:10.1063/1.1721265.
[9] R. Rothery, R. Silver, R. Herman, C. Torner, Analysis of experiments on single-lane bus flow, Oper. Res. 12 (1964) 913–933, doi:10.1287/opre.12.6.913.
[10] K.M. Tan, V.K. Ramachandaramurthy, J.Y. Yong, Integration of electric vehicles in smart grid: a review on vehicle to grid technologies and optimization techniques, Renew. Sustain. Energy Rev. 53 (2016) 720–732, doi:10.1016/j.rser.2015.09.012.
[11] K.Y. Liang, J. Mårtensson, K.H. Johansson, Heavy-duty vehicle platoon formation for fuel efficiency, IEEE Trans. Intell. Transp. Syst. 17 (2015) 1051–1061, doi:10.1109/TITS.2015.2492243.
[12] F. Leach, G. Kalghatgi, R. Stone, P. Miles, The scope for improving the efficiency and environmental impact of internal combustion engines, Transp. Eng. 1 (2020), doi:10.1016/j.treng.2020.100005.
[13] J. Larson, C. Kammer, K.Y. Liang, K.H. Johansson, Coordinated route optimization for heavy-duty vehicle platoons, in: Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), 2013, pp. 1196–1202. 10.1109/ITSC.2013.6728395
[14] E. Larsson, G. Sennton, J. Larson, The vehicle platooning problem: computational complexity and heuristics, Transp. Res. Part C: Emerging Technol. 60 (2015) 258–277, doi:10.1016/j.trc.2015.08.019.
[15] A. Adler, D. Miculescu, S. Karaman, Optimal policies for platooning and ride sharing in autonomy-enabled transportation, in: Proceedings of the Workshop on Algorithmic Foundations of Robotics (WAFR), 2016. http://peris.mit.edu/AdlerMiculescuKaramanWAFR16.pdf
[16] Y. Feng, D. He, Y. Guan, Composite platoon trajectory planning strategy for intersection throughput maximization, IEEE Trans. Veh. Technol. 68 (2019) 6305–6319, doi:10.1109/TVT.2019.2914163.
[17] L. Xu, W. Zhuang, G. Yin, C. Bian, Distributed formation control of homogeneous vehicle platoon considering vehicle dynamics, Int. J. Automot. Technol. 20 (2019) 1103–1112, doi:10.1007/s12239-019-0103-y.
[18] S. Gong, L. Du, Cooperative platoon control for a mixed traffic flow including human drive vehicles and connected and autonomous vehicles, Transp. Res. Part B: Methodol. 116 (2018) 25–61, doi:10.1016/j.trb.2018.07.005.
[19] M. Yan, W. Ma, L. Zuo, P. Yang, Dual-mode distributed model predictive control for platooning of connected vehicles with nonlinear dynamics, Int. J. Control Autom. Syst. 17 (2019) 3091–3101, doi:10.1007/s12555-018-0828-9.
[20] J. Wang, S. Gong, S. Peeta, L. Lu, A real-time deployable model predictive control-based cooperative platooning approach for connected and autonomous vehicles, Transp. Res. Part B: Methodol. 128 (2019) 271–301, doi:10.1016/j.trb.2019.08.002.
[21] D. Jia, D. Ngoduy, H.L. Vu, A multiclass microscopic model for heterogeneous platoon with vehicle-to-vehicle communication, Transp. B: Transp. Dyn. 7 (2019) 311–335, doi:10.1080/21680566.2018.1434021.
[22] P. Wang, B. Di, H. Zhang, K. Bian, L. Song, Platoon cooperation in cellular V2X networks for 5G and beyond, IEEE Trans. Wirel. Commun. 18 (2019) 3919–3932, doi:10.1109/TWC.2019.2919602.
[23] J.V. Saiáns-Vázquez, E.F. Ordóñez Morales, M. López-Nores, Intersection intelligence: supporting urban platooning with virtual traffic lights over virtualized intersection-based routing, Sensors 18 (2018) 4054, doi:10.3390/s18114054.
[24] M. Teixeira, P.M. d'Orey, Z. Kokkinogenis, Simulating collective decision-making for autonomous vehicles coordination enabled by vehicular networks: a computational social choice perspective, Simul. Model. Pract. Theory 98 (2020) 101983, doi:10.1016/j.simpat.2019.101983.
[25] R. Elbert, J.K. Knigge, A. Friedrich, Analysis of decentral platoon planning possibilities in road freight transport using an agent-based simulation model, J. Simul. 14 (2019) 64–75, doi:10.1080/17477778.2019.1675480.
[26] K. Bhargava, K.W. Choy, P.A. Jennings, S.A. Birrell, M.D. Higgins, Traffic simulation of connected and autonomous freight vehicles (CAV-f) using a data-driven traffic model of a real-world road tunnel, Transp. Eng. 2 (2020) 100011, doi:10.1016/j.treng.2020.100011.
[27] S. Wesolkowski, D. Wojtaszek, SaFESST: stochastic fleet estimation under steady state tasking via evolutionary fleet scheduling, in: Proceedings of the 2012 IEEE Congress on Evolutionary Computation, 2012, pp. 1–8, doi:10.1109/CEC.2012.6256537.
[28] H. Billhardt, A. Fernández, L. Lemus, Dynamic coordination in fleet management systems: toward smart cyber fleets, IEEE Intell. Syst. 29 (2014) 70–76, doi:10.1109/MIS.2014.41.
[29] G. Beni, J. Wang, Swarm intelligence, in: Proceedings of the 7th Annual Meeting of the Robotics Society of Japan, RSJ Press, 1989, pp. 425–428, doi:10.1007/978-3-642-27737-5_530-4.
[30] M.G. Hinchey, R. Sterritt, C. Rouff, Swarms and swarm intelligence, Computer 40 (2007) 111–113. https://pure.ulster.ac.uk/ws/files/11284565/04160239-2007-Swarms_and_Swarm_Intelligence.pdf
[31] Y. Zhang, F. Tian, B. Song, X. Du, Social vehicle swarms: a novel perspective on socially aware vehicular communication architecture, IEEE Wirel. Commun. 23 (2016) 82–89, doi:10.1109/MWC.2016.7553030.
[32] X. Zhang, X. Zheng, H. Ye, Vehicle scheduling base on the improved particle swarm, in: Proceedings of the 2016 World Automation Congress (WAC), IEEE, 2016, doi:10.1109/WAC.2016.7583057.
[33] I. Hwang, Y.J. Jang, Y.D. Ko, M.S. Lee, System optimization for dynamic wireless charging electric vehicles operating in a multiple-route environment, IEEE Trans. Intell. Transp. Syst. 19 (2017) 1709–1726, doi:10.1109/TITS.2017.2731787.
[34] M.S. Innocente, P. Grasso, Self-organising swarms of firefighting drones: harnessing the power of collective intelligence in decentralised multi-robot systems, J. Comput. Sci. 34 (2019) 80–101, doi:10.1016/j.jocs.2019.04.009.
[35] Y.N. Guo, J. Cheng, S. Luo, D. Gong, Y. Xue, Robust dynamic multi-objective vehicle routing optimization method, IEEE/ACM Trans. Comput. Biol. Bioinform. 15 (2017) 1891–1903, doi:10.1109/TCBB.2017.2685320.
[36] R. Woodman, W. Hill, S. Birrell, M.D. Higgins, An evolutionary approach to the optimisation of autonomous pod distribution for application in an urban transportation service, in: Proceedings of the 23rd International Conference on Mechatronics Technology, 2019, doi:10.1109/ICMECT.2019.8932138. Salerno, Italy
[37] Department of Transport, Road Traffic Statistics – Local Authority Coventry, 2018, [Online], Available: https://roadtraffic.dft.gov.uk/local-authorities/152. [Accessed March 18, 2018].