SAMoD: Shared Autonomous Mobility-on-Demand using Decentralized Reinforcement Learning

Maxime Guériau and Ivana Dusparic
Enable - CONNECT Research Centre
School of Computer Science and Statistics, Trinity College Dublin
maxime.gueriau@scss.tcd.ie, ivana.dusparic@scss.tcd.ie

***Abstract*—Shared mobility-on-demand systems can improve the efficiency of urban mobility through reduced vehicle ownership and parking demand. However, some issues in their implementations remain open, most notably the issue of rebalancing non-occupied vehicles to meet geographically uneven demand, as is, for example, the case during the rush hour. This is somewhat alleviated by the prospect of autonomous mobility-on-demand systems, where autonomous vehicles can relocate themselves; however, the proposed relocation strategies are still centralized and assume all vehicles are a part of the same fleet. Furthermore, ride-sharing is not considered, which also has an impact on rebalancing, as already occupied vehicles can also potentially be available to serve new requests simultaneously. In this paper we propose a reinforcement learning-based decentralized approach to vehicle relocation as well as ride request assignment in shared mobility-on-demand systems. Each vehicle autonomously learns its behaviour, which includes both rebalancing and selecting which requests to serve, based on its local current and observed historical demand. We evaluate the approach using data on taxi use in New York City, first serving a single request by a vehicle at a time, and then introduce ride-sharing to evaluate its impact on the learnt rebalancing and assignment behaviour.**

I. Introduction

Shared mobility is one of the key pillars of mobility-on-demand (MoD) systems. It is estimated that in 2015 nearly 8 million people used car-sharing services globally, with that number predicted to increase to 36 million by 2025 [1].

However, numerous operational issues remain to be addressed, vehicle rebalancing being one of main open issues in one-way car sharing systems [2], [3]. Due to inherent differences in patterns of people movement at different times of the day, vehicles tend to accumulate in certain areas. For example, during the morning rush hour vehicle requests originate in residential areas and end in city center or commercial zones. Vehicles then need to be returned to the areas with higher demand in order to be able to serve further requests.

This problem will be partially addressed by the introduction of fully autonomous vehicles, in autonomous mobility-on-demand systems (AMoD). Simulations show that ability of vehicles to autonomously rebalance and predictively reposition to high-demand locations reduces both customer waiting time and overall required fleet size [4]. However, research remains to be done on strategies and algorithms for rebalancing; current approaches are centralized, consider the full network and are therefore computationally intensive, and assume all cars belong to a single fleet, i.e., are controlled by a central entity [3]. In addition, most current rebalancing approaches do not consider ride-sharing [3]; vehicles can either be occupied or unoccupied, and only unoccupied ones are considered as available. With the introduction of ride-sharing, partially occupied vehicles also need to be taken into account in the overall strategy, as they are able to potentially serve further requests along their travel route to the destination, i.e., effectively being rebalanced to the areas they are travelling through.

As rebalancing strategies can be informed by both current and historical demand, reinforcement learning (RL) [5] is increasingly considered as an approach to learn the optimal predictive rebalancing based on historical data. [6] propose a centralized RL process to learn re-allocation patterns for the whole fleet. In the decentralized approach described in [3], each vehicle uses Deep Q-learning to learn whether to take a rebalancing action during regularly scheduled rebalancing windows. A similar approach was developed in [7], where an individual autonomous taxi learns the most likely passengers pick up points. However, this work does not take into account behaviours of other autonomous vehicles, potentially leading to imbalance in the opposite direction (e.g., all vehicles learning to go to areas of higher demand, over-saturating it), as well as not modelling ride-sharing.

To address these open issues, in this paper we propose SAMoD, Shared Autonomous Mobility-on-Demand, a decentralized RL-based multi-agent approach to car sharing and dynamic ride sharing. Each vehicle is controlled by an intelligent agent implementing a RL process, which observes where the current demand is and learns where high demand areas were historically. Vehicles actively seek passengers, achieving both dynamic passenger assignment, effectively decentralizing the dispatch process, as well as learning to relocate to high demand areas, automating the rebalancing process.

In the rest of this paper we present our approach, starting by reviewing the related work (Section II), and then move onto the details of our proposed approach and the design of SAMoD agents (Section III), evaluation scenarios (Section IV), and results (Section V). We finish by providing the directions for

future work in Section VI.

II. Related work

The relocation of empty vehicles in shared MoD systems has been widely studied in the literature, and can be divided into operator-based approaches [8], [9] (where employees of the car-sharing service relocate the vehicles), user-based approaches [10] (where users are financially incentivized to return the vehicles to high-demand areas) and more recently those in shared autonomous vehicle (SAV) systems [4], [11], [12] (where driverless vehicles are autonomously relocated). Relocation strategies are generally based on ability of the system to anticipate the demand, and relocate the vehicles to the areas where high demand is expected. To denote the different areas of the system in order to map the demand to a geographical area, the network is generally divided into several zones [13], blocks [11] or hexagons [14]. Strategies range from those that use a short window of known future requests (e.g., 5 minutes in [11] and 30 seconds in [4]), based on historical demand (e.g., [8]) or using prediction techniques to predict future demand (e.g., [14]). Learning techniques are also being increasingly applied to learn demand patterns based on historical data. [6] rely on a single RL-process to capture system-wide demand patterns, while in [3] each agent uses its local data to relocate in 8 immediate directions (south, south-east, east, north-east, north, north-west, west, south-west). Rebalancing in this work is done only at designated fixed-time periods rather than dynamically.

An additional challenge for shared MoD systems that incorporate ride sharing is how to dynamically assign a vehicle to multiple users/trips. To simplify the problem of matching the origin and destination of multiple riders, the system often assumes designated fixed pick up and drop off points, and only passengers with same pick up and drop off points are grouped into shared rides (e.g., [15], [16]). More complex models are introduced in [17], [18], where additional passengers can be picked up en-route based on a cost model capturing the impact of diversion on the cost and travel time. Such ride-sharing approaches are almost exclusively centralized, and the assignment for the whole system is done by a central dispatcher rather, which, in the cases of explicit network modelling, limits the scalability of the approaches.

Most recently, rebalancing and ride sharing have been looked at simultaneously, as their impact of one on another has been recognized. [4] propose a centralized linear optimisation model to assign multiple ride requests to vehicles, where free vehicles are rebalanced to the areas of higher demand. [3], as discussed already, use a decentralized approach to rebalancing, and add the possibility of ride sharing if serving an additional request does not increase the trip time by more than 50%. However, request assignment, both the original and ride-sharing one, is still done by a centralized dispatcher rather than in a decentralized manner.

Our proposed approach fully integrates rebalancing, request assignment and ride sharing, in a fully decentralized manner. Both rebalancing and request pick up/ride-sharing criteria is learnt by each agent itself; rebalancing can be based on current demand or observed historical data, and ride sharing can be limited only to nearby (within the same zone) requests, or allow for limited (one extra zone) diversions.

III. On-Demand Autonomous Ride-Sharing enabled Mobility

In the next section we present full details of our proposed approach, including the design of RL agents which control SAMoD vehicles, as well as overall architecture and operation of the system arising from individual agents operating in the same environment.

A. SAMoD agent

Each vehicle in SAMoD has intelligence provided by an individual SAMoD agent; in the case of autonomous vehicles this agent can directly control the vehicle, and in the case of driver-operated vehicles, it can provide the action advice.

To implement the learning process, we use Q-learning [19], a widely used RL algorithm which does not require a pre-defined model of the environment. In Q-learning, Q-values $Q(S, A)$ capture agent's experiences gained by interacting with the environment. More precisely, $Q(S, A)$ represents a long-term value of taking action A (representing one of the actions an agent is capable of executing) in state S (representing the current description of the agent's environment). Q-values are updated at each time-step according to the formula:

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha(R_{t+1} + \gamma(\max_A Q(S_{t+1}, A_{t+1})) - Q(S_t, A_t))$$

where R_{t+1} is the reward received by transitioning from S_t to S_{t+1}, α is the learning rate, which affects how much new experiences change the Q-value, and γ is the discount factor, which influences the rate at which old experiences are discarded.

The agent initially assumes no knowledge about the environment (e.g., about the frequency of ride requests, their origins or destinations), and over time, through interaction with the system, it learns the locations where the requests originate. SAMoD agent's main goal (Q-learning policy) is to serve ride requests, i.e., to pick up passengers. RL reward of 100 points is given to an agent if the vehicle has passenger(s) and no reward is given otherwise. This goal can be achieved in two ways: serving current pending requests, including additional ride-sharing requests, and, if there are no requests in agent's vicinity, by rebalancing to the areas where there are requests or from where historically the most requests originate. Each agent, by receiving this reward signal, learns its own way to best maximise the cumulative reward, by learning the actions for each of its states that lead to maximizing it.

To discretize agent's environment, geographical areas are divided into zones, and requests are classified depending on the origin and destination zones. Each agent bases its current decision making considering only a limited number of zones - the zone it is currently in, and its direct neighbouring zones. The full RL agent state consists of (i) vehicle

state ($empty, hasPassengers, full$), (ii) presence of current active requests in agent's zone (yes, no) and (iii) presence of current active requests in neighbouring zones (yes, no). We make a distinction between $full$ and $hasPassengers$ rather than just mark the vehicle as occupied, to denote if vehicle is available for further ride-sharing requests. Three basic actions an agent can execute are defined as $actions = \{pickUp, rebalance, doNothing\}$. Pick-up action can refer to picking up the first passenger, or picking up a ride-sharing request. With respect to rebalancing, agents learn between a number of rebalancing strategies: (i) rebalance to a neighbouring zone with the most current pending requests, (ii) rebalance to a neighbouring zone with the biggest gap between vehicle supply and number of requests, (iii) rebalance to a neighbouring zone which historically has the most requests, or (iv) rebalance to a neighbouring zone which historically has the biggest gap between vehicle supply and number of requests. At the start of the learning process, agents does not have historical information; as learning episodes are executed, an agent records the observed requests in each zone it operates in and builds up the required information for the more sophisticated rebalancing strategies.

Algorithm 1 SAMoD Agent Learning Process

```
// execute on learning event trigger
state ← MAPLOCALENVTOSTATE(env)
action ← QLEARNING.PICKACTION(state)
// if picking up new passengers
if action == "pickUp" then
    PICKUPPASSENGERS(nearestRequestID)
end if
state ← GETVEHICLESTATE
if state! = "full" then
    LISTENFORRIDESHARINGREQUESTS
    PICKUPPASSENGERS(nearestRequestID)
end if
// serve requests and finish learning episode
DROPOFFPASSENGERS(RequestID)
UPDATEENVIRONMENTHISTORICALDATA(RequestID)
UPDATEQLEARNINGPROCESS(RequestID)
TRIGGERNEWEVENT(DropOff)
// if rebalancing
if action == "rebalance" then
    //determine the zone to rebalance to
    current ← GETPENDINGREQUESTS(zones)
    historical ← GETHISTORICALREQUESTS(zones)
    zone ←SELECTZONE(current, historical)
    REBALANCETO(zone)
    TRIGGERNEWEVENT(Rebalanced)
end if
```

SAMoD's learning process is summarized in Algorithm 1. New learning episode is triggered by either dropping off passengers or by finishing rebalancing to a different zone. When the new action decision is needed, an agent gets the state of the vehicle and the environment, determines whether there is a pending request in its current zone (in which case it selects $pickUp$ action, and keeps listening out of further requests eligible for ride sharing), or if rebalancing to a neighbouring zone is needed. After each episode completion, reward is given to update Q-learning process, and historical environment information is updated.

B. SAMoD system architecture

SAMoD agents are designed to be a part of fully flexible shared on-demand mobility system, which is free-floating (there are no fixed vehicle stations) and one-way (vehicles do not have to return to the original location from which they were engaged). They support dynamic ride sharing, where ride-sharing requests are considered in real-time as they come up; no knowledge of future requests is assumed, although historical data is gathered throughout an agent's lifetime to learn where requests typically originate. Requests can originate anywhere, defined by a set of GPS coordinates, i.e., are not limited to predetermined fixed stops.

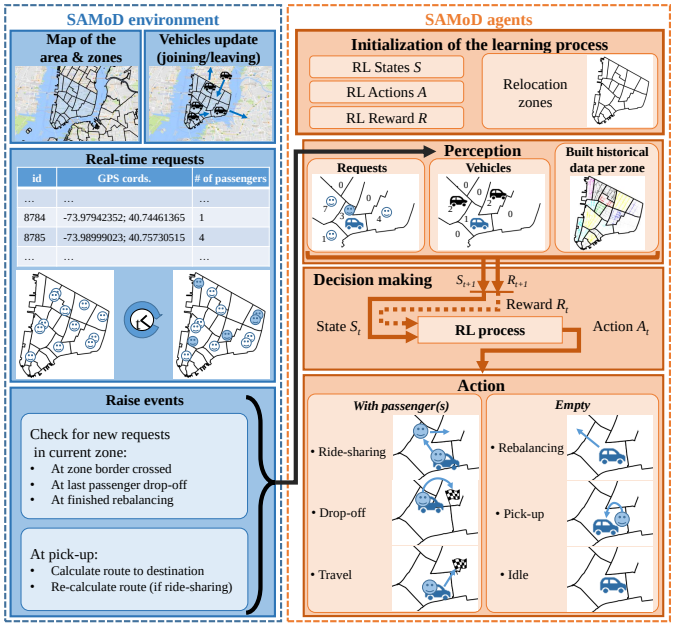

Fig. 1: SAMoD system

Figure 1 depicts the interactions between agents and the environment, from which the overall SAMoD system performance arises in a decentralized bottom-up manner. The system provides the map of the operating environment, which includes the road network. It also has the visibility of all the agents currently active, but agents can join and leave the system dynamically and seamlessly. Upon joining the system, each agent/vehicle initializes its learning process; first time it joins it starts from no knowledge and builds it up as it operates (including the possibility of online training or learning from a human operator). Each agents has access to the information about its own local environment only - number of the requests and other vehicles in its own zone and neighbouring zones.

Each agent in the zone has visibility of the same requests, and each agent selfishly picks up the request nearest to it; if the request is the nearest one to more than one agent, it is picked up by the first available agent. To prevent vehicles booking the requests before they are available to serve them, agents can execute a new pick up action only if one of the following events is raised by the system if the vehicle: 1) crosses a zone border, 2) drops off its last passenger or 3) reaches is relocation destination. Limiting the number of events ensures vehicles can pick-up only the requests in their own zone (or en-route for ride sharing), to prevent a vehicle for "booking" a far away request if much closer vehicle (in the same zone as the request) is available.

An advantage of this decentralized approach is that no adjustment is needed to the system or algorithm if a vehicle joins or leaves the system (unlike in centralized approaches where recalculation/reassignment might be needed), enabling easy vehicle breaks for maintenance, accounting for unpredictable traffic jams, etc. In addition, this approach seamlessly integrates driver-operated or privately owned autonomous vehicles, as those might be active in the system only for short and irregular periods of time rather than operate as a part of 24/7 fleet.

IV. Experimental Setup

A. General settings

The demand has been generated using the open NYC taxi dataset [20]. It describes recorded trips of yellow cabs in the Manhattan area. We extracted the trips from 50 consecutive Tuesdays between July 2015 and June 2016 to represent typical weekday demand patterns. We use a fixed fleet size of 200 shared autonomous vehicles, in order to observe cases when enabling ride sharing is needed to serve all requests (rush hours). Every vehicle has a capacity of 4 passengers. Each request includes: the time the user requested the trip, the number of passengers, origin coordinates, and destination.

To focus on rebalancing and ride-sharing strategies only, we have used a simplified traffic simulation without considering congestion. Autonomous vehicles drive themself to their current destination (rider pick-up or drop-off location or relocation zone). Travel times are computed in a similar way as in a grid network [21] and we assume a speed of 21 mph for peak hours (as in [17]).

B. Baselines

We compare our approach to baselines incorporating combinations (described in Table I) of the following strategies:

- *Centralized assignment*: a central dispatcher assigns the nearest vehicle to the request with highest waiting time
- *Decentralized assignment*: vehicles self-assign to the request with the highest waiting time in current zone or in a neighbouring zone
- *Rebalancing:* the vehicle drives towards the centre of the zone it was randomly assigned to at the start of simulation
- *Ride sharing:* a vehicle can pick-up more passengers until reaching maximum occupancy if the new request origin and destination are in the zones located on current route (or neighbouring zones)

TABLE I: Baseline scenarios

Summary	Assignment	Rebalancing	Ride sharing
C	Centralized	No	No
D	Decentralized	No	No
C_RB	Centralized	Yes	No
D_RB	Decentralized	Yes	No
C_RS	Centralized	No	Yes
D_RS	Decentralized	No	Yes
C_RB_RS	Centralized	Yes	Yes
D_RB_RS	Decentralized	Yes	Yes

C. SAMoD scenarios

Four configurations of SAMoD agent have been tested to assess the effect of two ride-sharing strategies and two rebalancing strategies (they are summarized in Table II).

TABLE II: SAMoD scenarios

Summary	Assignment	Rebalancing	Ride sharing
S_RB	Learnt	Learnt	No
S_RB_RS	Learnt	Learnt	Learnt current zone only
S_RB_RS+1	Learnt	Learnt	Learnt current zone+1
S_RB2_RS+1	Learnt	Learnt (limited)	Learnt current zone+1

Ride-share requests can be picked up only in the zone vehicle is already due to travel through, or diversions of maximum one extra zone are allowed. Rebalancing actions are learnt by an agent, and vehicles are allowed rebalance as many times in a row as needed to move to desired area, or rebalancing can be restricted to only one consecutive action. For all SAMoD scenarios, learning was performed on the first 40 days of the dataset, and results presented are using last 10 days, while vehicles act according to the learnt strategies.

D. Indicators

To evaluate SAMoD we use the set of most commonly used indicators in related work [4], [17], [11]:

- *Impact on the system*:
  - number and percentage of served requests;
  - number and percentage of timed-out requests (not assigned after waiting time exceeds 10 minutes).
- *Impact on riders*:
  - waiting time t_w: the time between the user request generation and the pick-up time;
  - detour time t_d: the difference between originally expected travel time and final travel time after serving ride-share request(s);
  - travel time TT: the time spent travelling in a vehicle.
- *Impact on vehicles*:
  - total Vehicle Miles Travelled (VMT) per vehicle;
  - empty VMT: VMT per vehicle without passengers;
  - engaged VMT: vehicle has one or more passengers;
  - shared VMT: vehicle serving more than one request;
  - occupancy: the number of passengers per trip.

V. Results and Analysis

In this section we present the results of SAMoD evaluation, first comparing it to the baselines, and then focusing in more detail on the impact of rebalancing parameters, ride-sharing parameters and request frequency/demand in the system.

A. Performance against baselines

As the results for different times of the day follow a similar pattern, we choose to present in detail the result set for morning peak traffic, 7-10am, with approximately 1,300 requests per hour (Table III). Results are split into rebalancing-only, and rebalancing and ride-sharing combined. We first compare SAMoD rebalancing-only approach to rebalancing baselines. With respect to global system performance, SAMoD significantly outperforms centralized baseline (C_RB), serving 91.91% of requests vs. 77.75% served by C_RB. However, decentralized baseline (D_RB) serves the most requests, 95.06%. From the perspective of a rider, waiting time is by far the best with SAMoD, 2.87 minutes vs. 11.07 and 4.57 with centralized and centralized baseline, respectively. From the vehicle perspective, decentralized baseline has the highest engaged VMT and lowest empty VMT, indicating the best performance from the vehicle perspective.

Looking at the combination of rebalancing and ride-sharing, centralized baseline (C_RB_RS) has the best performance from system and rider perspectives – it serves 98.75% requests, with average waiting time of 2.1 minutes. SAMoD performance is slightly lower: 97.32% of served requests, with 2.27-2.49 minutes average waiting time. SAMoD also achieves the best average occupancy: 3.19 passengers vs. 2.67 and 2.27 for centralized and decentralized baselines. The significance of this in real-world implementation depends on the pricing model, but in general increased occupancy increases the vehicle profit and decreases the rider cost.

Overall, SAMoD is only marginally worse than centralized baseline for system and rider perspective, with significant improvements from vehicle perspective – perhaps unsurprisingly, as the only goal vehicles are given is to maximize their own performance. Results show that vehicles locally optimizing their own performance results in near-optimal (i.e., near the performance of centralized baseline with full information and full control over all vehicles) performance from the system and rider waiting time perspective as well. Therefore, SAMoD is a suitable approach for vehicle rebalancing and ride-sharing in shared on-demand mobility systems, where decentralized approach is preferred (or the only feasible one).

B. Impact of rebalancing strategies

In this section we analyze the impact of different rebalancing strategies in SAMoD: a learnt balancing strategy, where each vehicle can rebalance as many times as it deems suitable (S_RB_RS+1), and the strategy where vehicle is allowed only one rebalancing action in a row (S_RB2_RS+1). We observe a significant impact on both empty and occupied VMT, and some impact on the number of requests served (respectively drops of 4% and 1.7%). Preventing consecutive rebalancing reduces empty VMT by on average 19 miles per vehicle per day. Effectively, these two strategies represent different trade-off points: in the first one, vehicle only maximizes the number of its pick-ups regardless of the empty VMT required to get to those requests, while in the second one vehicle sacrifices some of the farther away pick ups to lower its empty VMT. The preferred strategy in a real world deployment would depend on the cost per empty VMT and profit per mile with a passenger, and might also differ per vehicle type (e.g., electric vs. traditional fuel vehicles) or per time of the day (e.g., peak holiday/night-time pricing).

C. Impact of ride-sharing strategies

We now examine the impact of two SAMoD ride-sharing strategies: picking up only ride-sharing requests originating in the zones on the original route (S_RB_RS), or allowing up to one neighbouring zone diversions from the original route (S_RB_RS+1). The difference in the number of requests served between the two strategies is negligible, however there is an observable difference in the average occupancy (increase from 2.52 to 3.13 passengers), and average waiting and travel time for the riders (waiting time decreases from 2.46 minutes to 2.27 however the travel time increases by 3 minutes from 9.11 to 12.03 minutes). These results highlight the trade-off between reducing the cost both for the rider and the vehicle (by increasing the shared occupancy) and increasing the travel time by allowing longer diversions. It is possible that in real world deployments preferences with respect to such trade-off will be specified by passengers; some might prefer to optimize cost while other the travel time, and vehicles can dynamically switch between the strategies based on request parameters.

In conclusion, we demonstrated that SAMoD is a suitable approach to decentralized rebalancing and ride-sharing, with global performance arising from individual vehicle agents learning how to optimize their own individual performance with local information only. SAMoD achieves a performance close to centralized approach which has access to full global information and control over all vehicles in the system, even improving on some vehicle-oriented metrics such as vehicle occupancy and empty travel time. When analyzing the results we have identified a number of trade-offs that vehicles/system need to achieve when selecting the rebalancing and ride-sharing strategies, such as serving more requests at the cost of increasing empty travel time to reach those requests, and increasing vehicle ride-shared occupancy, at the cost of longer travel time in order to serve ride-sharing requests.

VI. Conclusions and Future Work

This paper presented a decentralized RL-based shared autonomous mobility-on-demand system (SAMoD), which includes dynamic car and ride sharing. Each shared autonomous vehicle is controlled by an intelligent agent able to learn to maximize the number of requests served through its own decentralized assignment process (incorporating dynamic ride sharing) and through a learnt rebalancing strategy. Global system optimization arises from individually learnt behaviours,

		No RB, No RS		Rebalancing		Ride-sharing		RB and RS		SAMoD			
		C	D	C_RB	D_RB	C_RS	D_RS	C_RB_RS	D_RB_RS	S_RB	S_RB_RS	S_RB_RS+1	S_RB2_RS+1
System	Satisfied requests	29667	35388	30191	36913	38327	38368	38346	38407	35691	37790	37679	36159
	% of total requests	76.4	91.13	77.75	95.06	98.7	98.81	98.75	98.91	91.91	97.32	97.03	93.12
	Not served requests	8675	3098	8150	1590	0	54	0	11	2903	693	726	2242
	% of total requests	22.34	7.98	20.99	4.09	0	0.14	0	0.03	7.48	1.78	1.87	5.77
Riders	Avg t_w (min)	11.63	5.48	11.07	4.57	2.41	2.56	2.1	2.6	2.87	2.46	2.27	2.49
	Avg TT (min)	5.8	5.69	5.79	5.72	10.31	9.21	10.19	8.73	5.69	9.11	12.03	12.12
	Avg t_d (min)	0	0	0	0	4.57	3.47	4.44	2.99	0	3.39	6.31	6.49
Vehicles	Avg VMT	863.8	735.79	884.71	861.4	690.28	716.49	760.06	845.02	882.85	865.94	869.94	644.32
	Avg empty VMT	428.48	228.29	442.24	330.04	117.02	147.9	181.56	268.52	371.95	352.6	335.81	147.37
	Avg engaged VMT	435.32	507.5	442.47	531.36	573.26	568.59	578.5	576.5	510.91	513.34	534.13	496.95
	Avg shared VMT	103	120.55	103.78	125.54	382.75	324.74	376.86	301.96	115.84	330.3	433.86	409.11
	Avg occupancy	1.47	1.48	1.47	1.48	2.67	2.39	2.63	2.27	1.45	2.52	3.13	3.19

TABLE III: Results of all the scenarios for 7-10am period for 10 days

resulting in a higher level of service at the system scale and improved riders' waiting times. Simulation results of different SAMoD rebalancing and ride-sharing strategies highlight several trade-offs vehicles need to make. Potential real-world deployments will require a more fine-tuned balance of these trade-offs, and would benefit from more sophisticated multi-objective learning techniques. We plan to extend SAMoD with W-Learning, a multi-objective RL strategy previously successfully applied to balance multiple trade-offs in an urban traffic control system [22]. SAMoD performance should also be evaluated in the presence of variable congestion and traffic control scenarios. We will do this using a simulation of REALT [23], an RL-based urban traffic control system, which will enable us to investigate dynamic car- and ride-sharing impact at a city scale on traffic congestion.

ACKNOWLEDGEMENTS

This research has been sponsored by the Irish Research Council through "Surpass: how shared autonomous cars will transform cities" New Horizons award.

REFERENCES

[1] Frost and Sullivan, "Future of carsharing market to 2025," 2016. [Online]. Available: http://www.frost.com

[2] C. Boldrini and R. Bruno, "Stackable vs autonomous cars for shared mobility systems: A preliminary performance evaluation," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2017, pp. 232–237.

[3] J. Wen, J. Zhao, and P. Jaillet, "Rebalancing shared mobility-on-demand systems: A reinforcement learning approach," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2017, pp. 220–225.

[4] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *Proceedings of the National Academy of Sciences*, vol. 114, no. 3, pp. 462–467, 2017.

[5] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[6] Oisin Dolphin, Good Travel Software, "AI has proven itself at the poker table but can it prove itself in car-sharing technology?" 2017. [Online]. Available: http://goodtravelsoftware.com/blog/

[7] M. Han, P. Senellart, S. Bressan, and H. Wu, "Routing an autonomous taxi with reinforcement learning," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2016, pp. 2421–2424.

[8] S. Weikl, K. Bogenberger, and N. Geroliminis, "Simulation framework for proactive relocation strategies in free-floating carsharing systems," in *Transportation Research Board 95th Annual Meeting*, no. 16-2725, 2016.

[9] S. Paschke, M. Balac, and F. Ciari, "Implementation of vehicle relocation for carsharing services in the multiagent transport simulation matsim," in *Transportation Research Board 96th Annual Meeting*, no. 17-06805, 2017.

[10] A. G. Bianchessi, S. Formentin, and S. M. Savaresi, "Active fleet balancing in vehicle sharing systems via feedback dynamic pricing," in *2013 IEEE 16th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2013, pp. 1619–1624.

[11] D. J. Fagnant and K. M. Kockelman, "The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios," *Transportation Research Part C: Emerging Technologies*, vol. 40, pp. 1–13, 2014.

[12] K. Winter, O. Cats, B. van Arem, and K. Martens, "Impact of relocation strategies for a fleet of shared automated vehicles on service efficiency, effectiveness and externalities," in *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2017, pp. 844–849.

[13] H. Dia and F. Javanshour, "Autonomous shared mobility-on-demand: Melbourne pilot simulation study," *Transportation Research Procedia*, vol. 22, pp. 285–296, 2017.

[14] K. Spieser, S. Samaranayake, W. Gruel, and E. Frazzoli, "Shared-vehicle mobility-on-demand systems: a fleet operators guide to rebalancing empty vehicles," in *Transportation Research Board 95th Annual Meeting*, no. 16-5987, 2016.

[15] J. Bischoff, M. Maciejewski, and K. Nagel, "City-wide shared taxis: A simulation study in berlin," in *Intelligent Transportation Systems (ITSC), 2017 IEEE 20th International Conference on*, 2017.

[16] M. W. Levin, K. M. Kockelman, S. D. Boyles, and T. Li, "A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application," *Computers, Environment and Urban Systems*, vol. 64, pp. 373 – 383, 2017.

[17] W. Zhang, S. Guhathakurta, J. Fang, and G. Zhang, "The performance and benefits of a shared autonomous vehicles based dynamic ridesharing system: An agent-based simulation approach," in *Transportation Research Board 94th Annual Meeting*, January 2015.

[18] D. J. Fagnant and K. M. Kockelman, "Dynamic ride-sharing and optimal fleet sizing for a system of shared autonomous vehicles," in *Transportation Research Board 94th Annual Meeting*, no. 15-1962, 2015.

[19] C. J. Watkins and P. Dayan, "Q-Learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[20] NYC Taxi and Limousine Commission, "Tlc trip record data," 2017. [Online]. Available: http://www.nyc.gov

[21] W. Shen, C. V. Lopes, and J. W. Crandall, "An online mechanism for ridesharing in autonomous mobility-on-demand systems," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*. AAAI Press, 2016, pp. 475–481.

[22] I. Dusparic and V. Cahill, "Autonomic multi-policy optimization in pervasive systems: Overview and evaluation," *ACM Transactions on Autonomous and Adaptive Systems*, April 2012.

[23] I. Dusparic, J. Monteil, and V. Cahill, "Towards autonomic urban traffic control with collaborative multi-policy reinforcement learning," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2065–2070.