Model Predictive Control of Ride-sharing Autonomous Mobility-on-Demand Systems

Matthew Tsao[1], Dejan Milojevic[1,2], Claudio Ruch[2], Mauro Salazar[1,2], Emilio Frazzoli[2], and Marco Pavone[1]

***Abstract*— This paper presents a model predictive control (MPC) approach to optimize routes for Ride-sharing Autonomous Mobility-on-Demand (RAMoD) systems, whereby self-driving vehicles provide coordinated on-demand mobility, possibly allowing multiple customers to share a ride. Specifically, we first devise a time-expanded network flow model for RAMoD. Second, leveraging this model, we design a real-time MPC algorithm to optimize the routes of both empty and customer-carrying vehicles, with the goal of optimizing social welfare, namely, a weighted combination of customers' travel time and vehicles' mileage. Finally, we present a real-world case study for the city of San Francisco, CA, by using the microscopic traffic simulator MATSim. The simulation results show that a RAMoD system can significantly improve social welfare with respect to a single-occupancy Autonomous Mobility-on-Demand (AMoD) system, and that the predictive structure of the proposed MPC controller allows it to outperform existing reactive ride-sharing coordination algorithms for RAMoD.**

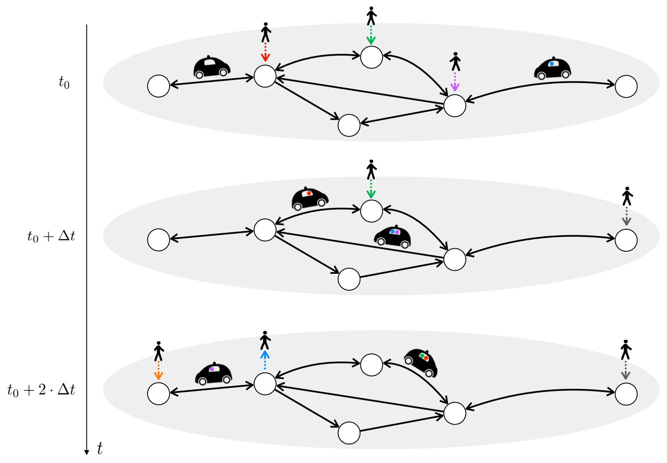

Fig. 1. Schematic representation of a double-occupancy RAMoD system. Customers entering the system (downward-facing arrows) are collected and dropped off (upward-facing arrows) in a first-in-first-out fashion.

I. Introduction

Traffic congestion is becoming a major challenge worldwide [1]. Space limitations and a slowly adapting infrastructure make congestion even more difficult to address in densely populated cities, where recent years have witnessed an increase in population and mobility demand [2]. Explicitly, from 2007 to 2013, the annual cost of congestion in the US increased has roughly doubled, from \$78 Billion to \$124 Billion [7], [8]!

Within this context, this paper investigates how optimized control of AMoD systems, which combine the two emerging transportation paradigms of ride-sharing service and AMoD, can address the congestion problem by improving traffic throughput. Specifically, an AMoD system consists of a fleet of unit-capacity, self-driving vehicles providing one-way on-demand mobility. AMoD systems differ from traditional, non-autonomous mobility-on-demand systems, as AMoD fleets are *actively* controlled in a centralized fashion, with the goal of optimizing the assignment of customers to vehicles and the routes of both customer-carrying and empty, rebalancing vehicles (that is vehicles that travel empty in order to align vehicle availability with anticipated future travel demand). Due to their operation flexibility, AMoD systems hold promise to diminish the societal cost of mobility [9], but might not directly lead to a reduction in congestion, and indeed could make congestion worse, for example, due to the presence of empty traveling vehicles or induce demand effects. This has prompted studies to route AMoD vehicles in a "congestion-aware" fashion [11], [12], and to investigate a synergistic integration with public transit [10]. This paper considers the complementary strategy of infusing ride-sharing service within the AMoD paradigm (see Fig. 1), with the objective of designing MPC algorithms for RAMoD systems that significantly reduce the number of vehicles on the road while providing high-quality mobility service.

Related literature: Our contribution pertains to two main research fields: i) AMoD systems and ii) ride-sharing. There are several approaches to analyse and control AMoD systems, such as simulation models [15]–[17], queuing-theoretical models [18], [19] and multi-commodity network flow models [9], [20]. Simulation models describe transportation systems with high precision, but at the cost of not being amenable to optimization. Queuing-theoretical models capture the stochasticity of the customer requests and can be used for control synthesis, but it is extremely hard to represent external constraints in this framework. Multi-commodity network flow models can be efficiently implemented in optimization frameworks, whilst still being very expressive and compatible with a variety of constraints. Consequently, they have been applied to a number of problems: from the control of AMoD systems in congested road networks [11], [12], [21], in coordination with the power network [22] and public transit [10], to the design of Model Predictive Control (MPC) algorithms [13].

The ride-sharing problem has been studied in static and dynamic environments, and with the objective of minimizing the mileage driven and the average travel time, or maximizing the number of customers served [23]. There are a number of contributions to the static ride-sharing problem, where all the requests are assumed to be known in advance. A study for the single-driver-single-rider setting is presented in [24]. The dynamic version of the ride-sharing problem captures new riders and drivers continuously entering and leaving the

[1]Autonomous Systems Lab, Stanford University, Stanford (CA), United States {mwtsao,pavone}@stanford.edu

[2]Institute for Dynamic Systems and Control ETH Zürich, Zurich (ZH), Switzerland {dejanmi,clruch,samauro}@ethz.ch

system. Ride-sharing features exist today in mobility services like Lyft and Uber, and have been studied in [14], [31]. These works, along with ride-sharing systems by Lyft and Uber are primarily *reactive* in the sense that they only consider serving the current demand. Considering the substantial performance gains presented by [13], [32] for single occupancy mobility networks, a natural question is whether ride-sharing systems can experience similar benefits by anticipating future customer demand. Against this background, the scope of the present paper is to devise MPC algorithms for RAMoD systems.

Statement of contributions: In this paper we present a MPC algorithm for RAMoD accounting for present and future travel demand. Specifically, the contribution of this paper is threefold: First, we develop a multi-commodity network flow model capturing the operations of the double-occupancy RAMoD system shown in Fig. 1. Second, we devise a MPC algorithm assigning multiple customers to vehicles, designing vehicle routes and rebalancing empty vehicles to anticipate future requests, with the goal to maximize social welfare, namely, a weighted combination of customers' travel time and vehicles' mileage. Third, we evaluate the performance of our algorithm against state-of-the-art unit capacity AMoD approaches as well as reactive ride-sharing algorithms. Our results show that a RAMoD system can significantly reduce overall costs with respect to a single-occupancy AMoD system, and that its predictive structure allows it to outperform existing reactive ride-sharing algorithms.

Organization: The remainder of this paper is structured as follows: Section II introduces the multi-commodity flow optimization model for RAMoD. The design and details of the RAMoD MPC algorithm is discussed in Section III. Section IV presents a real-world case-study for San Francisco, CA, where we test our approach and compare it with the state-of-the-art. We conclude the paper in Section V with a summary and a discussion on future research.

II. Flow Optimization Model for Double-occupancy RAMoD

In this section, we present a graphical model representation of road networks and pose the problem of coordinating a fleet of vehicles for mobility service as an optimization problem that leverages ride-sharing to service transportation requests in the road network. All service vehicles in this model are *double-occupancy*, i.e., they can carry up to two passengers at any given time, as shown in Fig. 1.

A. Modeling the Road Network

We model the transportation network as a spatio-temporal graph. The road network is partitioned into n stations, which are spatially disjoint regions where customers can request rides to and from. The nodes of the spatio-temporal graph are then $\mathscr{V} = [n] \times \mathbb{T}$ so that a node $(i,t) \in \mathscr{V}$ specifies a physical location i and a time t. We measure time in discrete intervals of Δt so that $\mathbb{T} = \Delta t \cdot \mathbb{N}$. Note that the size and number of stations dictate the resolution of the graph: Having small stations increases the granularity, but, as a trade-off, more stations are needed to cover the entire road network.

Each road in the network has a nominal driving speed which may depend on the level of exogenous traffic, and a fixed length. We model congestion as an exogenous influence on the nominal speeds of roads, but do not model the endogenous congestion effects induced by RAMoD vehicles. Therefore, there is no limit to the number of RAMoD vehicles that can be traveling at the nominal speed on a road at any given time. The travel time to traverse a road is given by its length divided by its nominal speed.

Paths are defined as an ordered sequence of roads, and the travel time of a path is simply the sum of the travel time of its roads. Then, for a given level of exogenous traffic, we denote the time needed to travel from station i to station j taking the fastest path as τ_{ij}. Directed edges in this graph correspond to paths in the road network. For any two nodes $(i,t_1),(j,t_2)$, a directed edge from (i,t_1) to (j,t_2) exists if $\tau_{ij} = t_2 - t_1$. Because we do not consider endogenous congestion effects from AMoD vehicles, we only need to consider shortest paths when routing vehicles. To allow cars to idle, we also include edges from (i,t) to $(i,t+\Delta t)$ for each $i \in [n]$, $t \in \mathbb{T}$. Defining $\mathscr{E}$ to be the set of all such edges, our weighted graph representation of the road network is $\mathscr{G} := (\mathscr{V}, \mathscr{E})$ where the weight along an edge is its corresponding travel time. The AMoD system is endowed with M self-driving cars whose position can be at stations if they are idling, or on edges if they are in transit. We denote the transportation demand for a set of times $\mathscr{T} \subset \mathbb{T}$ as $\Lambda_{\mathscr{T}}$, which is a $n \times n \times |\mathscr{T}|$ array, so that $\Lambda_{\mathscr{T}(t_0)}(i,j,t)$ is the number of customers that will request a trip from a location in station i to a location in station j at time $t \in \mathscr{T}$. We will use the shorthand notation $\lambda_{ijt} := \Lambda_{\mathscr{T}}(i,j,t)$ when the time set $\mathscr{T}$ is unambiguous.

B. Integer Network Flow Model for RAMoD Systems

We leverage network flow models and Integer Linear Programming (ILP) to devise an algorithm for controlling the double-occupancy RAMoD fleet shown in Fig. 1. In accordance with the model presented in Section II-A, this involves specifying actions for all vehicles in the fleet for a horizon of T time-steps, where each time-step is Δt minutes. At time t_0, the planning horizon $\mathscr{T}(t_0) \subset \mathbb{T}$ is then

$$\begin{aligned}\mathscr{T}(t_0) &:= \{t_1, t_2, \ldots t_T\} \\ &= \{t_0 + \Delta t, t_0 + 2 \cdot \Delta t, \ldots, t_0 + T \cdot \Delta t\},\end{aligned}$$

so that $t_k := t_0 + k \cdot \Delta t$ for $k \in [T]$.

In this setting, however, the control algorithm needs to provide instructions to empty vehicles as well as partially occupied vehicles. Since vehicles with different occupancy levels have varying levels of vacancy and commitments to customer destinations, a controller needs to treat different types of vehicles accordingly. Using the graph representation of the road network introduced in Section II-A, we represent the distinction between cars with different occupancy levels by introducing the following decision variables:

- $r \in \mathbb{N}^{n^2 T}$, where r_{ijt} represents the number of completely empty vehicles traveling from station $i \to j$ at time t;
- $\{x(m)\}_{m=1}^{n} \in \mathbb{N}^{n^2 T}$, where $x_{ijt}(m)$ represents the number of vehicles with exactly one passenger whose destination is station m, traveling from station $i \to j$ at time t;
- $p \in \mathbb{N}^{n^3 T}$ where p_{ijkt} is the number of cars at station i at time t with two passengers with destinations j and k, respectively.

Moreover, in our model, RAMoD vehicles interact with customers at stations in two phases:

- **Phase 1**: When a vehicle arrives at a station, it first delivers any customers onboard whose destination is in this region.

- **Phase 2**: After delivering customers, the vehicle has the option to pick up new customers before leaving the station.

Additionally, we assume that cars at maximum capacity are not controllable: They will drive directly from their current position to their first destination and cannot be assigned a different task until they drop off their first customer. Thus a car of type p_{ijkt} must drive from station i to station j.

We use $s_r \in \mathbb{N}^{nT}, s_x \in \mathbb{N}^{n^2T}$ to encode the current state of RAMoD vehicles in the system so that:

- $s_{r,it}$ specifies how many currently busy vehicles will become available as vacant cars in station i at time t;
- $s_{x,it}(m)$ specifies the number of currently busy cars that will become available as partially occupied cars in station i at time t whose on-board customers' destination is in station m.

These variables, however, are not enough to specify an actionable strategy. For example, suppose an algorithm computes $x_{1,2,t}(2)=1$, meaning that a car carrying one passenger should embark from station 1 to station 2 at time t. This does not tell us, however, whether the car should have picked up a customer, dropped off a customer, or done nothing before leaving station 1. To address these kinds of ambiguities, we introduce book-keeping variables $x^{(\mathrm{zo})}, x^{(\mathrm{so})}, p^{(\mathrm{zo})}, p^{(\mathrm{so})}$ where the superscripts $(\mathrm{zo}),(\mathrm{so})$ indicate that the car had *zero occupants* or a *single occupant* after finishing phase 1. Now if we have $x^{(\mathrm{zo})}_{1,2,t}(2)=1$, then we know that the car at station 1 should pick up a customer before leaving for station 2, since the car had zero occupants after phase 1. On the other hand, if we see $x^{(\mathrm{so})}_{1,2,t}(2)=1$, then the car should not pick up a customer before leaving the station, since it has one passenger after phase 1, and leaves with one passenger. Naturally, x and p are the sum of their zero and single occupant components which we enforce using the following constraints.

$$x_{ijt}(m) = x^{(\mathrm{zo})}_{ijt}(m) + x^{(\mathrm{so})}_{ijt}(m) \qquad \forall i,j,m \in [n], t \in \mathscr{T}(t_0) \tag{1}$$
$$p_{ijkt} = p^{(\mathrm{zo})}_{ijkt} + p^{(\mathrm{so})}_{ijkt} \qquad \forall i,j,k \in [n], t \in \mathscr{T}(t_0) \tag{2}$$

Additionally, constraints (3) and (4) enforce that phase 1 (dropping off current customers) must occur before phase 2 (picking up new customers and leaving the station)

$$x^{(\mathrm{so})}_{ijt}(i) = 0 \qquad \forall i,j \in [n], t \in \mathscr{T}(t_0) \tag{3}$$
$$p^{(\mathrm{so})}_{iijt} = 0 \qquad \forall i,j \in [n], t \in \mathscr{T}(t_0) \tag{4}$$

We now present the physical constraints that any actionable strategy for the planning horizon $\mathscr{T}(t_0)$ must satisfy:

$$s_{r,it} + \sum_{j=1}^{n}\left(r_{jit_{ji}} + x_{jit_{ji}}(i) + p_{jiit_{ji}}\right) = \sum_{j=1}^{n}\left(r_{ijt} + \sum_{m=1}^{n} x^{(\mathrm{zo})}_{ijt}(m) + \sum_{u=1}^{n} p^{(\mathrm{zo})}_{ijut}\right) \tag{5}$$
$$\text{where } t_{ji} = t - \tau_{ji},\ \forall i \in [n], t \in \mathscr{T}(t_0).$$

Constraint (5) specifies the options available to empty vehicles. Specifically, the left side of (5) counts the number of cars that will have zero occupants after arriving at station i at time t and finishing phase 1. These cars can either remain empty, pick up one passenger, or pick up two passengers before leaving the station, corresponding to the terms on the right hand side of (5).

$$s_{x,it}(m) + \sum_{j=1}^{n}\left(x_{ji(t-\tau_{ji})}(m) + p_{jim(t-\tau_{ji})}\right) = \sum_{j=1}^{n}\left(p^{(\mathrm{so})}_{imjt} + x^{(\mathrm{so})}_{ijt}(m)\right) \tag{6}$$
$$\forall i,m \in [n], i \neq m, t \in \mathscr{T}(t_0).$$

Constraint (6) specifies the options available to single occupant vehicles. Specifically, the left side of (6) counts the number of cars that will have one occupant whose destination is m after arriving at station i at time t and finishing phase 1. These cars can leave with or without picking up another passenger, corresponding to the first and second terms on the right side of (6) respectively. Interactions with customers are captured by the following two constraints:

$$a_{ijt} = \left(\sum_{u=1}^{n} x^{(\mathrm{zo})}_{iut}(j) + p^{(\mathrm{so})}_{iujt} + \mathbb{1}_{[j\neq u]}\cdot\left[p^{(\mathrm{zo})}_{iujt} + p^{(\mathrm{zo})}_{ijut}\right]\right) + 2p^{(\mathrm{zo})}_{ijjt} \tag{7}$$
$$\forall i,j \in [n], t \in \mathscr{T}(t_0)$$
$$d_{ijt} = \sum_{\tau=1}^{t} \lambda_{ij\tau} - a_{ij\tau} \tag{8}$$
$$\forall i,j \in [n], t \in \mathscr{T}(t_0)$$

The variable a in 7 counts the number of trips from station i to j that are serviced at time t. This is because the right side of (7) includes all the ways such customers can be picked up. The variable d in (8) counts the number of customers that are waiting for each trip type and time. This is because the right side of (8) is the total demand from i to j up to time t minus the number of those customers that have been served up until time t. Finally, because we cannot have fractional vehicles on the road, all of the decision variables must be integer.

$$r \in \mathbb{N}^{n^2T}, \{x^{(\mathrm{zo})}, x^{(\mathrm{so})}(m)\}_{m=1}^{n} \in \mathbb{N}^{n^2T}, p^{(\mathrm{zo})}, p^{(\mathrm{so})} \in \mathbb{N}^{n^3T}. \tag{9}$$

The goal of a RAMoD algorithm is to maximize social welfare. Specifically, we aim at minimizing a weighted combination of total travel time and operational costs represented by

$$J(r,x,p,d) = V_d \cdot \sum_{i,j=1}^{n}\sum_{t\in\mathscr{T}(t_0)} d_{ijt} \tag{10}$$
$$+ \sum_{i,j=1}^{n}\sum_{t\in\mathscr{T}(t_0)} \tau_{ij}\cdot\left(V_r\cdot r_{ijt} + \sum_{m=1}^{n} V_x\cdot x_{ijt}(m) + V_p\cdot p_{ijmt}\right),$$

where V_r, V_x, V_p, V_d are tunable coefficients representing the per unit time cost of rebalancing, driving customer carrying vehicles, and delaying customer pickup respectively. The first term measures service quality for customers and the last term represents operation cost of the system.

Our RAMoD algorithm for servicing transportation demands implements the strategy obtained by solving the following integer linear program

$$\underset{r,x^{(\mathrm{zo})},x^{(\mathrm{so})},p^{(\mathrm{zo})},p^{(\mathrm{so})}}{\text{minimize}} \quad J(r,x,p,d) \tag{11}$$
$$\text{subject to } (1)-(9).$$

C. Model Discussion

A few comments are in order. Partitioning the road network into stations is a general approach that encapsulates models with a wide range of focuses as special cases including, but not limited to, temporal resolution, spatial resolution, and different modes of transportation [10], [11], [13], [32]. We choose to study double-occupancy fleets because most vehicles have at least two passenger seats that can be utilized for ride-sharing. We avoid studying higher occupancy models due to computational complexity and the diminishing returns on performance of increased occupancy [31].

III. A Real-Time RAMoD MPC Algorithm

In this section we use the integer network flow framework discussed in Section II-B to derive a real-time MPC algorithm for RAMoD that leverages demand forecasts to improve service quality.

A. Real-time RAMoD MPC Algorithm

To extend the integer linear programming approach from Section II-B to an online MPC setting, we implement it in a receding horizon framework. Specifically, every Δt minutes, we collect the system state $\{s_r, s_x\}$, currently waiting customers $\Lambda_{\{t_0\}}$, and travel times $\{\tau_{ij}\}_{i,j\in[n]}$ computed from the road congestion levels at time t_0 as input to the ILP. The ILP uses a planning horizon of $\mathscr{T}(t_0)$ with T time-steps where t_0 is the current time. Since the true demand in our planning horizon $\Lambda_{\mathscr{T}(t_0)}$ is not known, we use a forecaster $\widehat{\Lambda}: \mathbb{T} \to \mathbb{N}^{n\times n\times T}$ instead so that $\widehat{\Lambda}(t_0)$ is an estimate of $\Lambda_{\mathscr{T}(t_0)}$. After executing the first step of the control strategy resulting from the ILP, we update the system state and re-solve the ILP in a receding horizon manner. In general, ILPs are NP-hard. Thus, there is no guarantee that this method will scale to large problem instances. However, as only the first step of the resulting strategy is implemented before the algorithm updates the system state and recomputes, only the decision variables at the first time-step t_1 need to be integer in order for the algorithm to be actionable. Thus, we can relax the integer constraint in (9) on all variables at times $t_2, t_3, ..., t_T$ to reduce the complexity of the problem while still having an actionable algorithm. With this heuristic in place, the number of integer constrained variables no longer depends on the planning horizon T, allowing for a larger planning horizon to be employed. This relaxation leads to the following mixed integer linear program (MILP):

$$\begin{aligned} &\min J(r,x,p,d) \qquad (12)\\ &\text{subject to } (1)-(8)\\ &\qquad r_{ijt_1} \in \mathbb{N}\ \forall i,j\in[n]\\ &\qquad \{x^{(\mathrm{zo})}_{ijt_1}(m)\}_m, \{x^{(\mathrm{so})}_{ijt_1}(m)\}_m, p^{(\mathrm{zo})}_{ijmt_1}, p^{(\mathrm{so})}_{ijmt_1} \in \mathbb{N}\\ &\qquad \forall i,j,m \in [n]\\ &\qquad \{x^{(\mathrm{zo})}(m)\}_m, \{x^{(\mathrm{so})}(m)\}_m, p^{(\mathrm{zo})}, p^{(\mathrm{zo})} \succeq 0\ \forall\, m \end{aligned}$$

In practice, the forecast $\widehat{\Lambda}(t_0)$ will not be a perfect estimate of $\Lambda_{\mathscr{T}(t_0)}$. An incorrect forecast can cause an algorithm to send a vehicle to a location to pick up a customer when in fact no customer will show up. Another important situation is if the forecaster underestimates demand and dispatches too few vehicles, leading to unserved customers. The first scenario causes fuel inefficiency and the second degredates service quality. For these reasons it is important for any predictive approach to have robustness to such inaccuracies. To accomplish this, we implement a matching algorithm that can be used to pickup the aforementioned unserved customers with vehicles that are told to idle by the solution of (12). Specifically, we choose a reserve fleet size $M_0 < M$ so that, if the solution to (12) assigns idle tasks to $M_0 + M_{\text{extra}}$ vehicles, then the matching algorithm is allowed to use M_{extra} cars to serve waiting customers. In this way, we have a reactive component to serve unexpected demand that will only act if it can gurantee that there will be M_0 vehicles available to the controller (12) for the next time-step. The purpose of the reserve fleet size M_0 is to prevent the matching algorithm from implementing a completely greedy approach, and ultimately striking a balance between prediction and reaction when the forecast is not perfect. Algorithm 1 describes the full receding-horizon algorithm for RAMoD.

Algorithm 1: Ride-Sharing Autonomous Mobility on Demand

```
1  RAMoD ($\mathscr{G}, \mathbb{T}, \widehat{\Lambda}, T, M_0$);
   Input : Graph rep. of road network $\mathscr{G} = (\mathscr{V}, \mathscr{E})$,
           operation horizon $\mathbb{T}$, number of timesteps in
           the planning horizon $T$, forecaster $\widehat{\Lambda}$ and
           reserve fleet size $M_0$.
   Output: Control actions
           $r, \{x^{(\mathrm{zo})}\}_m, \{x^{(\mathrm{so})}\}_m, p^{(\mathrm{zo})}, x^{(\mathrm{so})}$.
2  for $t_0 \in \mathbb{T}$ do
3      Collect the vehicle state of the system $s_r, s_x$ ;
4      Collect the current demand $\Lambda_{(t_0,t_0)}$ ;
5      Obtain forecast for the next $T$ timesteps $\widehat{\Lambda}(t_0)$;
6      Solve (12) to obtain
         $r, \{x^{(\mathrm{zo})}\}_m, \{x^{(\mathrm{so})}\}_m, p^{(\mathrm{zo})}, x^{(\mathrm{so})}$;
7      Implement the $r, \{x^{(\mathrm{zo})}\}_m, \{x^{(\mathrm{so})}\}_m, p^{(\mathrm{zo})}, x^{(\mathrm{so})}$
         instructions for the first timestep ;
8      if There are more than $M_0$ idling cars;
9       then
10         Pair idling cars with nearby customers until
             either there are only $M_0$ idling cars or no
             waiting customers.
11     end
12 end
```

B. Algorithm Discussion

A few comments are in order. First, by representing the vehicle actions as flows between locations, the number of decision variables in (12) does not depend on the number of vehicles or the number of customers, enabling the algorithm to operate effectively in highly populated areas. This is contrary to recent approaches presented in [14], [31] where the problem size increases with the number of cars and the number of customers. Second, the relaxation from (11) to (12) allows modern branch and bound solvers to find solutions in under one minute, allowing Algorithm 1 to be run in real-time. Third, Algorithm 1 is able to leverage forecasts of future demand into its strategy. Finally by updating the travel times using current congestion values when solving (12), we can model congestion effects in a dynamic and time

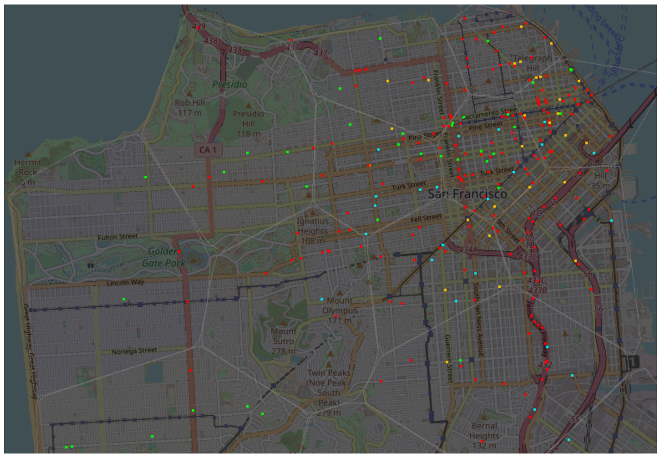

Fig. 2. Visualization of the San Francisco experiment with AMoDeus [35], whereby the green dots represent empty cars, blue dots are rebalancing cars, orange dots are cars on pick-up, and red dots are cars with customers. The stations are separated with gray lines.

varying manner, despite not directly considering endogenous contributions from the fleet.

IV. Numerical Experiments

In this section, we present numerical experiments to benchmark the performance of Algorithm 1 when servicing trips from transportation datasets in simulation. The code written for experiments in [14], [31] is protected by copyright, so we attempted to capture its characteristics in our own implementation of a reactive ride-sharing algorithm which we refer to as RAMoD-Reactive. The performance of Algorithm 1 is compared to the real-time MPC algorithm for AMoD in [32], RAMoD-Reactive, and to an existing, high-performing rebalancing heuristic [33].

A. Scenario

We focus on the transportation network of San Francisco, CA. The Fig. 2 shows a sequence of the simulation with the San Francisco map. The study uses a publicly available dataset of taxi traces recorded in the city of San Francisco [36]. The traces were recorded between May 17, at 03:00:04 and June 10, 2008, 02:25:34. The dataset contains a total of 464'045 trips for the entire period. We use 15 days from May 20 up to June 9, where the weekends are not included, to fit our model data, and May 19 for the evaluation. The model we fit is for the forecast of customer demand needed by the RAMoD and the AMoD MPC algorithms.

B. Simulation Environment

We use the AMoDeus [35] simulator to validate the algorithms. It is an open-source simulator to analyze and validate algorithms for mobility-on-demand (MoD) systems. Internally, it uses the agent-based transportation simulator MATSim [34], which includes well-tested high-fidelity simulation of road dynamics. The simulator is able to represent large MoD systems with unreduced fleet sizes and to compare them directly to existing benchmark algorithms, e.g., [33]. The simulator contains an inner loop to model road network dynamics as well as an outer loop to take into account varying or dynamic demand which may change as a function of the network dynamics. For this validation, we assumed a static demand profile, i.e., we assume that independent of cost and performance of our implemented RAMoD system, the stochastic user equilibrium [37] has been reached and is invariant.

C. Experimental Design

We simulate the scenario described in Section IV-A testing the following four controllers:

- *RAMoD-MPC*: The controller implementing Algorithm 1.
- *AMoD-MPC*: The controller described in [32] that also leverages short-term forecasts in a MPC manner, but does not have a ride-sharing feature.
- *RAMoD-Reactive*: The *RAMoD-MPC* algorithm modified to emulate a similar behavior to the methods presented in [14], [31]. We do this by making two modifications: First, we provide no forecast to make the algorithm reactive. Second, we eliminate vehicle paths with detours that cause significant inconveniences to customers, corresponding to the sharability graphs in [14], [31].
- *AMoD-Reactive*: The controller described in [33].

A fleet size of $M = 400$ vehicles is provided to all controllers. The *RAMoD-MPC*, *AMoD-MPC* and *RAMoD-Reactive* controllers use a time horizon of 150 minutes broken up into $T = 10$ time steps, each of length $\Delta t = 15$ minutes. The number of stations used for these controllers is 25. The *AMoD-Reactive* controller uses 10 stations instead of 25 because using 10 stations leads to better performance. The *RAMoD* algorithms uses a reserve fleet size of $M_0 = 100$. The network is split into stations using a k-means partitioning method on the request origin locations. For the MPC scheme with time horizon of $T = 10$, every 15 minutes the optimization problem is solved and the control inputs for the next 15 minutes in the simulation are applied. For every time in $\mathscr{T}(t_0)$, the demand forecast for each origin, destination pair is computed by taking the sample average of the corresponding values from different days in the dataset mentioned in section IV-A.

D. Results

The main results obtained with the four controllers are summarized in Table I. To evaluate the customer satisfaction we measure the waiting times and journey times. To measure the operational cost we use the total distance driven. The mean waiting time corresponds to the average waiting time of all customers during the whole day, whilst the mean journey time is equivalent to the average time of all journeys starting from the request submission time until the drop-off time at destination. The distance driven describes the sum of all distances traveled through the whole simulation day by all cars including pickup, drop-off, and rebalancing trips.

E. Discussion

A few comments are in order. As Fig. 3 shows, the ride-sharing algorithms drive 20% less distance than the single occupancy algorithms. Additionally, the algorithms using MPC achieve more than 40% lower mean waiting times than the reactive algorithms by preemptively rebalancing available vehicles to locations where future demand is expected to appear. From this we see that *RAMoD-MPC* is at least as good and somewhere better in terms of mean waiting time

TABLE I

PERFORMANCE OF THE CONTROLLERS

Controller	Mean Waiting Time	Mean Journey Time	Distance Driven
RAMoD-MPC	4 min 3 s	22 min 18 s	372'637 km
AMoD-MPC	4 min 15 s	17 min 30 s	453'450 km
RAMoD-Reactive	6 min 8 s	24 min 25 s	367'785 km
AMoD-Reactive	6 min 48 s	20 min	476'183 km

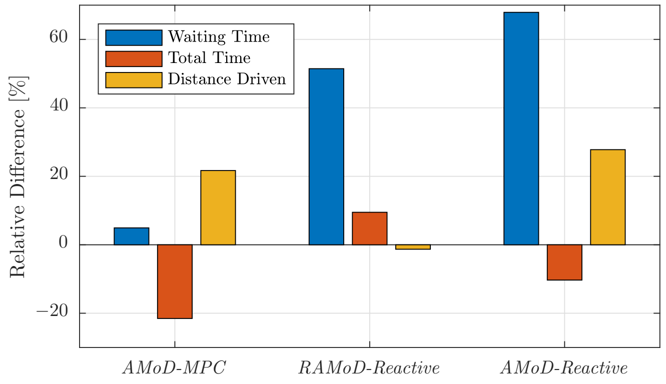

Fig. 3. Relative performance difference with respect to the proposed *RAMoD-MPC* algorithm.

and travel distance compared to the benchmark algorithms. For this case study, the time to solve (12) for Algorithm 1 was consistently under 15 seconds.

The number of waiting customers throughout the day is shown in Fig. 4. Interestingly, *RAMoD-MPC* has more waiting customers than *AMoD-MPC* between 7:00am and 9:00am. We believe this is because *RAMoD-MPC* waits until there are sufficiently many customers that can be grouped together using ride-sharing before picking up customers. By doing this it able to serve the demand with fewer cars, allowing it to be more prepared for later demand between 9:00am and 2:00pm. The driving distance for both fleets is also shown in Fig. 4, showing *AMoD-MPC* driving substantially more from 7:00am-2:00pm, aligning with our hypothesis.

These advantages, however, come at a price. As Fig. 3 shows, ride-sharing increases the average journey time of customers by more than 10%. This is caused by vehicles taking detours to service multiple customers. Despite the extra journey time, we emphasize that the societal and economic advantages of ride-sharing are substantial. Having the service fleet driving less reduces CO_2 emissions and fuel costs. In this particular case study, the *RAMoD-MPC* drives 80000 km less than the *AMoD-MPC* while having a similar mean wait time. Using the standard CO_2 emission rate for cars equipped with gasoline engines of 25 kg/100km [38, Chpt. 1], in this particular case ride-sharing would reduce CO_2 emissions by 7300 metric tons every year. In terms of fuel cost, considering a standard gasoline consumption of 8 L/100km [38] would lead to 2.3M L gasoline saved annually, resulting in savings exceeding 2M $ per year [39].

V. CONCLUSION

In this paper we presented a model predictive control (MPC) algorithm to coordinate a fleet of self-driving vehicles for servicing travel requests in a ride-sharing setting.

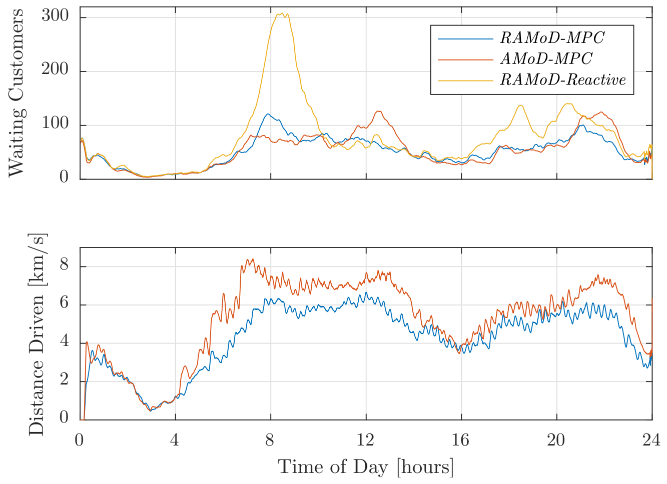

Fig. 4. Number of waiting customers for the controllers tested as a function of time (upper plot) and distance traveled by the fleet per unit time (lower plot).

To this end, we first derive an integer network flow model to represent the transportation network. We then designed a Ride-sharing Autonomous Mobility on Demand (RAMoD) algorithm based on receding horizon network flow optimization. We presented a case study for San Francisco, CA, and compared our algorithm to the state-of-the-art, using high-fidelity simulations in MATSim. Our experiments showed the proposed RAMoD-MPC algorithm outperforms the state-of-the-art unit-capacity mobility algorithms in terms of total driving distance and reactive ride-sharing algorithms in terms of mean wait time. In particular, by slightly increasing the total trip length for customers, the RAMoD-MPC algorithm is able to significantly reduce the distance traveled by mobility providers, and consequently can reduce the contribution of mobility services to urban congestion.

This work opens the field for several research directions. First, in addition to ride-sharing, recent work in adding features to Autonomous Mobility-on-Demand (AMoD) systems include interactions with public transit [10] and the power grid [22], congestion-aware routing algorithms [11], [12], [21] and improved real-time rebalancing leveraging demand forecasts [13], [32]. The models and algorithms presented in these works are not unified and there does not exist a general framework which considers all of these features together. Such a unified algorithm would be of great practical interest, since all of these factors play a crucial role in the operation and service quality of AMoD. Third, it is advisable to extend the study presented here and develop models with more than double-occupancy. Finally, because this MPC framework involves forecasts of future travel demand, improving the quality of the forecasts using time series and deep learning techniques could provide performance improvements.

ACKNOWLEDGMENTS

We would also like to thank Dr. Ilse New and Ramón Iglesias for their assistance with the proofreading, useful advice and comments. This research was supported by the National Science Foundation under CAREER Award CMMI-1454737 and the Toyota Research Institute (TRI). This article solely reflects the opinions and conclusions of its authors and not NSF, TRI, or any other entity.

References

[1] J. I. Levy, J. J. Buonocore, and K. Von Stackelberg, “Evaluation of the public health impacts of traffic congestion: a health risk assessment,” *Environmental Health*, vol. 9, no. 1, p. 65, 2010.

[2] (2018) The world factbook. Central Intelligence Agency. Central Intelligence Agency. Available at https://www.cia.gov/library/publications/the-world-factbook/fields/2212.html.

[3] R. Molla. (2018) Americans seem to like ride-sharing services like Uber and Lyft. But it’s hard to say exactly how many use them. Recode. Recode. Available at https://www.recode.net/2018/6/24/17493338/ride-sharing-services-uber-lyft-how-many-people-use.

[4] F. Siddiqui. (2018) Failing transit ridership poses an ‘emergency’ for cities, experts fear. The Washington Post. The Washington Post. available online.

[5] P. Berger. (2018) Mta blames uber for decline in new york city subway, bus ridership. The Wall Street Journal. Available online.

[6] W. Hu. (2017) Your uber car creates congestion. should you pay a fee to ride? The New York Times. Available online.

[7] D. Schrank, T. Lomax, and S. Turner, “The 2007 urban mobility report,” Texas Transportation Institute, Tech. Rep., 2007.

[8] B. Tuttle and T. Cowles, “Traffic jams cost americans $124 billion in 2013,” *Time - Money*, 2014.

[9] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, and M. Pavone, “Toward a systematic approach to the design and evaluation of Autonomous Mobility-on-Demand systems: A case study in Singapore,” in *Road Vehicle Automation*. Springer, 2014.

[10] M. Salazar, F. Rossi, M. Schiffer, C. H. Onder, and M. Pavone, “On the interaction between autonomous mobility-on-demand and the public transportation systems,” in *Proc. IEEE Int. Conf. on Intelligent Transportation Systems*, 2018, in Press. Extended Version, Available at https://arxiv.org/abs/1804.11278.

[11] F. Rossi, R. Zhang, Y. Hindy, and M. Pavone, “Routing autonomous vehicles in congested transportation networks: Structural properties and coordination algorithms,” *Autonomous Robots*, vol. 42, no. 7, pp. 1427–1442, 2018.

[12] M. Salazar, M. Tsao, I. Aguiar, M. Schiffer, and M. Pavone, “CARA: a congestion-aware routing algorithm for autonomous mobility-on-demand systems,” in *Annual Meeting of the Transportation Research Board*, 2019, submitted.

[13] M. Tsao, R. Iglesias, and M. Pavone, “Stochastic model predictive control for autonomous mobility on demand,” in *Proc. IEEE Int. Conf. on Intelligent Transportation Systems*, 2018, in Press. Extended Version, Available at https://arxiv.org/pdf/1804.11074.

[14] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, “Quantifying the benefits of vehicle pooling with shareability networks,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 3, pp. 13 290–13 294, 2014.

[15] S. Hörl, C. Ruch, F. Becker, E. Frazzoli, and K. W. Axhausen, “Fleet control algorithms for automated mobility: A simulation assessment for zurich,” in *Annual Meeting of the Transportation Research Board*, 2018.

[16] M. Maciejewski, J. Bischoff, S. Hörl, and K. Nagel, “Towards a testbed for dynamic vehicle routing algorithms,” in *Int. Conf. on Practical Applications of Agents and Multi-Agent Systems - Workshop on the application of agents to passenger transport (PAAMS-TAAPS)*, 2017.

[17] 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.

[18] R. Zhang and M. Pavone, “A queueing network approach to the analysis and control of Mobility-on-Demand systems,” in *American Control Conference*, 2015.

[19] ——, “Control of robotic Mobility-on-Demand systems: A queueing-theoretical perspective,” *Int. Journal of Robotics Research*, vol. 35, no. 1–3, pp. 186–203, 2016.

[20] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus, “Robotic load balancing for Mobility-on-Demand systems,” *Int. Journal of Robotics Research*, vol. 31, no. 7, pp. 839–854, 2012.

[21] R. Zhang, F. Rossi, and M. Pavone, “Routing autonomous vehicles in congested transportation networks: Structural properties and coordination algorithms,” in *Robotics: Science and Systems*, 2016.

[22] F. Rossi, R. Iglesias, M. Alizadeh, and M. Pavone, “On the interaction between Autonomous Mobility-on-Demand systems and the power network: Models and coordination algorithms,” *IEEE Transactions on Control of Network Systems*, 2018, submitted. Extended version available at https://arxiv.org/abs/1709.04906.

[23] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, “Optimization for dynamic ride-sharing: A review,” *European Journal of Operational Research*, vol. 223, pp. 295–303, 2012.

[24] A. Amey, “Proposed methodology for estimating rideshare viability within an organization: Application to the mit community,” in *Annual Meeting of the Transportation Research Board*, 2011.

[25] M. Savelsberg and M. Sol, “The general pickup and delivery problem,” *Transportation Science*, vol. 29, no. 1, pp. 17–29, 1995.

[26] R. Baldacci, V. Maniezzo, and A. Mingozzi, “An exact method for the car pooling problem based on lagrangian column generation,” *Operations Research*, vol. 52, no. 3, pp. 422–439, 2004.

[27] R. Calvo, F. de Luigi, P. Haastrup, and V. Maniezzo, “A distributed geographic information system for the daily car pooling problem,” *Computers & Operations Research*, vol. 31, no. 13, pp. 2263–2278, 2004.

[28] G. Berbeglia, J.-F. Cordeau, and G. Laporte, “Static pickup and delivery problems: a classification scheme and survey,” *Top*, vol. 15, no. 1, pp. 1–31, 2007.

[29] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, “Dynamic ride-sharing: a simulation study in metro atlanta,” *Transportation Research Part B: Methodological*, vol. 45, no. 9, pp. 1450–1464, 2011.

[30] A. Kleiner, B. Nebel, and V. Ziparo, “A mechanism for dynamic ride sharing based on parallel auctions,” in *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, 2011, pp. 266–272.

[31] 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.

[32] R. Iglesias, F. Rossi, K. Wang, D. Hallac, J. Leskovec, and M. Pavone, “Data-driven model predictive control of autonomous mobility-on-demand systems,” in *Proc. IEEE Conf. on Robotics and Automation*, 2018.

[33] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus, “Load balancing for Mobility-on-Demand systems,” in *Robotics: Science and Systems*, 2011.

[34] A. Horni, K. Nagel, and K. W. Axhausen, Eds., *The Multi-Agent Transport Simulation MATSim*. Ubiquity Press, 2016.

[35] S. Hörl, C. Ruch, and E. Frazzoli, “Amodeus, a simulation-based testbed for autonomous mobility-on-demand systems,” in *Proc. IEEE Int. Conf. on Intelligent Transportation Systems*, 2018.

[36] P. M., S.-D. N., and G. M. (2009) CRAWDAD dataset epfl/mobility (v. 2009-02-24). Downloaded from https://crawdad.org/epfl/mobility/20090224.

[37] H. Yang, “System optimum, stochastic user equilibrium, and optimal link tolls,” *Transportation Science*, vol. 33, no. 4, pp. 354–360, 1999.

[38] L. Guzzella and A. Sciarretta, *Vehicle Propulsion Systems*. Springer Berlin Heidelberg, 2007.

[39] Expatistan. (2018) The price of gas in San Francisco. Available at https://www.expatistan.com/price/gas/san-francisco.