Model Predictive Control of Autonomous Mobility-on-Demand Systems

Rick Zhang, Federico Rossi, and Marco Pavone

Abstract— **In this paper we present a model predictive control (MPC) approach to optimize vehicle scheduling and routing in an autonomous mobility-on-demand (AMoD) system. In AMoD systems, robotic, self-driving vehicles transport customers within an urban environment and are coordinated to optimize service throughout the entire network. Specifically, we first propose a novel discrete-time model of an AMoD system and we show that this formulation allows the easy integration of a number of real-world constraints, e.g., electric vehicle charging constraints. Second, leveraging our model, we design a model predictive control algorithm for the optimal coordination of an AMoD system and prove its stability in the sense of Lyapunov. At each optimization step, the vehicle scheduling and routing problem is solved as a mixed integer linear program (MILP) where the decision variables are binary variables representing whether a vehicle will 1) wait at a station, 2) service a customer, or 3) rebalance to another station. Finally, by using real-world data, we show that the MPC algorithm can be run in real-time for moderately-sized systems and outperforms previous control strategies for AMoD systems.**

I. Introduction

Current mobility trends, anchored by the continued growth of privately owned automobiles, are creating high levels of air pollution and traffic congestion, especially in densely populated cities with limited space for road infrastructure and parking. Urban transportation in the US account for over half of the total oil consumption [1] while producing 20% of total carbon dioxide emissions [2]. With world urban population projected to increase by 2.5 billion by 2050 [3], current urban transportation trends are widely viewed as unsustainable for the future.

The eventual solution to this problem will likely involve the convergence of several key emerging technologies. First, one-way carsharing has emerged as a promising solution to increase vehicle utilization and promote sustainable urban land use, and the rise in mobile technology has enabled on-demand taxi services like Uber [4]. Second, electric vehicle technology has the potential to drastically reduce emissions and dependence on oil, and promote the generation of renewable energy. Finally, the advancement in autonomous driving technology promises to further increase convenience (through on-demand service), safety, and mobility for people unable or unwilling to drive. These emerging pieces lead to a transformational technology known as autonomous mobility-on-demand (AMoD) [5], [6], [7], whereby shared driverless electric cars provide personal on-demand transportation for customers. Its many potential benefits have led a number of companies to aggressively pursue AMoD technology [8], [9]. However, the optimal coordination of these robotic electric vehicles in a transportation network remains a challenge.

Statement of contributions: The objective of this paper is to design a model predictive control (MPC) approach to optimize vehicle scheduling and routing in an AMoD system. Model predictive control (also known as receding horizon control) is a control technique whereby an open-loop optimization problem is solved at each time step to yield a sequence of control actions up to a fixed horizon, and the first control action is executed. Due to its iterative nature, MPC can achieve closed-loop performance, is robust to model errors, and is well suited for complex, constrained systems. Previous work on mobility-on-demand (MoD) and AMoD systems has focused on design and fleet sizing, where the steady state of the system is characterized using a fluidic model [5], [10], a queueing network model [7], [11], [12] or a Markov model [13]. Real-time control strategies devised in these works are heuristics based on the steady-state model and are exclusively concerned with the "rebalancing" problem (where the robotic vehicles redistribute themselves to align with asymmetric customer demand). In particular, these models do not allow easy integration of real-world constraints such as electric vehicle charging and parking capacities, which limits their practical application. Control algorithms for AMoD are also related to dispatch algorithms for taxis [14], [15], but these algorithms typically cannot enforce a vehicle distribution since taxi drivers make the final decisions.

The key feature of our approach is that it is amenable to real-time optimization and receding horizon control while having the flexibility to account for many real-world phenomena such as battery charging constraints, customer priorities, and parking space limitations. Specifically, the contribution of this paper is threefold. First, we propose a novel discrete-time model of an AMoD system and we show that this formulation allows the easy integration of a number of real-world constraints, with a special focus on electric vehicle charging constraints. Second, leveraging our model, we design a model predictive control algorithm for the optimal coordination of an AMoD system and prove its stability in the sense of Lyapunov. Finally, by using real-world data, we show that the MPC algorithm can be run in real-time for moderately-sized systems and compare its performance to four other AMoD control algorithms and taxi dispatch algorithms in the literature. We show that the MPC algorithm not only outperforms other algorithms in terms of customer wait times, but can also be used as an optimal performance benchmark to evaluate other dispatch algorithms.

Our work draws inspiration from time-space network models such as [16] for optimizing bus routes and [17], [18] for vehicle redistribution policies in carsharing systems.

This research was supported by National Science Foundation under CAREER Award CMMI-1454737 and by the Dr. Cleve B. Moler Stanford Graduate Fellowship.

Rick Zhang, Federico Rossi, and Marco Pavone are with the Department of Aeronautics & Astronautics, Stanford University, Stanford, CA 94305 {rickz, frossi2, pavone}@stanford.edu

Receding horizon (or model predictive) control techniques have been used in the context of transportation systems [19], [20]. The key difference between our approach and these works, besides the modeling differences, is that we provide a rigorous proof of stability within an MPC framework. In this perspective, our approach is related to the one used in [21] for capacity maximization in battery networks and [22], [23] for cooperative multi-agent systems.

Organization: The remainder of this paper is organized as follows: In Section II we present our AMoD model and discuss the inclusion of operational constraints, in particular battery charging. In Section III we formulate the problem of regulating an AMoD system. In Section IV we present two MPC algorithms to control the AMoD system and prove their stability. Simulation studies are presented in Section V to assess the performance of the MPC algorithms and characterize the effect of charging constraints on system throughput. In Section VI we discuss the inclusion of additional operational constraints, e.g., customers' priorities. Lastly, in Section VII we draw our conclusions and present directions for future research.

II. Model

In this section, we first introduce a linear discrete-time model of an AMoD system with similar assumptions as those proposed in the fluidic model in [5]. We then expand the model by introducing charging constraints associated with using electric vehicles. Such constraints are both of practical and theoretical interests. In particular, as we will see, with the addition of charging constraints (which are piecewise-linear), the system is no longer strictly a linear system. However, optimization in the form of a mixed-integer linear program (MILP) can still be performed with these additional constraints. The inclusion of additional operational constraints is discussed in Section VI.

A. Linear AMoD model

We consider a discrete-time system with N stations and m single-occupancy vehicles. Let $\mathcal{N}$ represent the set of stations, $|\mathcal{N}| = N$, and let $\mathcal{V}$ represent the set of vehicles, where $|\mathcal{V}| = m$. At each time step, customers arrive at each station and wait for vehicles to transport them to their desired destinations. In our model, customers may not be serviced on a first-come-first-serve basis, as the system determines the best ordering for serving the customers. While this idea may at first seem to be at odds with the notion of "fairness," it is a standard strategy for ride-sharing services like SuperShuttle [24], where it is important to cluster customers traveling in the same direction. This strategy is also more natural in an equivalent system where "stations" are geographical regions (rather than physical infrastructure) and customers request transportation via a mobile app. Furthermore, we will show in Section VI that customer priority can be easily integrated into the model.

Before defining the system states, we first define the "control" variables of the AMoD system. A control decision is made at each time step for each vehicle parked at a station. The two possible actions each vehicle can take are 1) transport a customer from one station to another, and 2) rebalance the system by driving itself from one station to another (this is a key advantage of robotic vehicles). We can encode these actions using binary variables. Let $v_{ij}^k(t) = 1$ if vehicle k is transporting a customer from station i to station j beginning at time t, and arriving at station j at time $t+t_{ij}$. The travel time t_{ij} is assumed to be deterministic and known. Similarly, let $w_{ij}^k(t) = 1$ if vehicle k is rebalancing from station i to station j beginning at time t and arriving at time $t + t_{ij}$.

Denote by $d_{ij}(t)$ the number of customers waiting at station i at the start of time period t whose destination is station j. Denote by $c_{ij}(t)$ the number of customers that arrive at station i at time t heading to station j. The dynamics of $d_{ij}(t)$ are propagated as follows:

$$d_{ij}(t+1) = d_{ij}(t) + c_{ij}(t) - \sum_{k\in\mathcal{V}} v_{ij}^k(t), \tag{1}$$

where the last term represents the number of passenger-carrying vehicles leaving station i at time t. Note that $d_{ij} \geq 0$ for all $i, j \in \mathcal{N}$ and for all time. This means that if $d_{ij}(t) = 0$ and $c_{ij}(t) = 0$, then $v_{ij}^k(t) = 0$ for all $k \in \mathcal{V}$. The number of waiting customers plays a significant role in characterizing the performance of the system, hence $d_{ij}(t)$ is modeled as a state variable.

When a vehicle is on the road, it is necessary to keep track of how long it will be traveling before it reaches its destination. We represent this by the binary variables ${}^{T_i}p_i^k(t) \in \{0, 1\}$, where $k \in \mathcal{V}$ and $T_i \in \{0, \max_j\{t_{ji}\} - 1\}$ is the number of time steps remaining until the vehicle reaches station i, $i \in \mathcal{N}$. Suppose vehicle k leaves station j destined for station i at time t; then ${}^{t_{ji}-1}p_i^k(t+1)$ is set to one at the next time step to indicate that the vehicle is $t_{ji} - 1$ time steps from station i. In the subsequent time step, ${}^{t_{ji}-2}p_i^k(t+2)$ is set to one to indicate the progress of the vehicle along its path. Eventually ${}^{0}p_i^k(t+t_{ji})$ is set to one to signal that the vehicle has arrived at station i. The propagation of ${}^{T_i}p_i^k(t)$ is formally defined as follows:

$$\begin{aligned} &{}^{T_i}p_i^k(t+1) \\ &= \begin{cases} {}^{T_i+1}p_i^k(t) + \sum\limits_{j:t_{ji}-1=T_i} (v_{ji}^k(t) + w_{ji}^k(t)) & \text{if } T_i < T_{\max,i} \\ \sum\limits_{j:t_{ji}-1=T_{\max,i}} (v_{ji}^k(t) + w_{ji}^k(t)) & \text{if } T_i = T_{\max,i}, \end{cases} \end{aligned} \tag{2}$$

where $T_{\max,i} = \max_j t_{ji} - 1$. Note that since each vehicle can only be in one place at one time, the ${}^{T_i}p_i^k$'s are subject to the constraints

$$\sum_{i\in\mathcal{N}} \sum_{T_i} {}^{T_i}p_i^k(t) \leq 1. \tag{3}$$

Constraint 3 will not be enforced explicitly, and instead will result from constraints on the control variables. The dimension of ${}^{T_i}p_i^k$ depends on the travel time from each station to every other station. For each vehicle and each station i there are $T_{\max,i}$ such variables, so the total dimension of ${}^{T_i}p_i^k$ is $\mathcal{D} = |\mathcal{V}| \sum_{i\in\mathcal{N}} T_{\max,i}$.

Finally, let $u_i^k(t)$ be the state variable associated with waiting at a station, that is $u_i^k(t) = 1$ if vehicle k waited at station i from time $t-1$ to time t. The dynamics of $u_i^k(t)$ are modeled as follows

$$u_i^k(t+1) = u_i^k(t) + {}^0\, p_i^k(t) - \sum_{j\in\mathcal{N}} (v_{ij}^k(t) + w_{ij}^k(t)). \tag{4}$$

Equation (4) ensures that 1) a vehicle can only perform an action (via v_{ij}^k or w_{ij}^k) if it is at a station, and 2) if a vehicle does not perform an action, it waits at a station. In other words, each vehicle must complete a task before starting another. However, a vehicle cannot perform an action if it is already on the road, thus a constraint is needed between $u_i^k(t)$ and ${}^{T_i}p_i^k(t)$ which ensures that a vehicle is either waiting at a station or traveling. This is formalized as

$$\sum_{i\in\mathcal{N}} u_i^k(t) + \sum_{i\in\mathcal{N},T_i} {}^{T_i}p_i^k(t) = 1. \tag{5}$$

Finally, the following constraint between u_i^k, v_{ij}^k, and w_{ij}^k ensures that a vehicle only performs one task at a time

$$\sum_{i\in\mathcal{N}} \Big(u_i^k(t+1) + \sum_{j\in\mathcal{N}} v_{ij}^k(t) + \sum_{j\in\mathcal{N}} w_{ij}^k(t)\Big) \leq 1, \tag{6}$$

where the sum is zero when vehicle k is traveling (i.e., $\sum_{i\in\mathcal{N},T_i\neq 0} {}^{T_i}p_i^k(t) = 1$).

The variables $d_{ij}(t)$, ${}^{T_i}p_i^k(t)$, and $u_i^k(t)$ make up the state of the system, as they completely define the customer demand and the state of all vehicles. Let $x(t)$ be the state vector, that is, the column vector created by reshaping and concatenating $d_{ij}(t)$, ${}^{T_i}p_i^k(t)$, and $u_i^k(t)$. We define the set of feasible states by $\mathcal{X}$ where

$$\mathcal{X} := \left\{ x = [d_{ij}\ {}^{T_i}p_i^k\ u_i^k]^\intercal \;\middle|\; \begin{array}{l} d_{ij} \in (\mathbb{N}\cup\{0\})^{N^2},\ d_{ii}=0 \\ {}^{T_i}p_i^k \in \{0,1\}^{\mathcal{D}},\ {}^{T_i}p_i^k \text{ satisfies (3)} \\ u_i^k \in \{0,1\}^{N|\mathcal{V}|},\ u_i^k \text{ satisfies (5)} \end{array} \right\} \tag{7}$$

The variables $v_{ij}^k(t)$ and $w_{ij}^k(t)$ make up the control input of the system. Let $u(t)$ be the control vector, that is, the column vector created by concatenating $v_{ij}^k(t)$ and $w_{ij}^k(t)$. Any feasible control v_{ij}^k which sends vehicles to transport customers cannot transport more customers than there are waiting, thus

$$\sum_{k\in\mathcal{V}} v_{ij}^k(t) \leq d_{ij}(t) + c_{ij}(t). \tag{8}$$

We collect our system constraints to form the set of feasible controls, $\mathcal{U}(t)$, where

$$\mathcal{U}(t) := \left\{ u = [v_{ij}^k\ w_{ij}^k]^\intercal \;\middle|\; \begin{array}{l} v_{ij}^k \in \{0,1\}^{|\mathcal{V}|N^2}, v_{ii}^k = 0 \\ w_{ij}^k \in \{0,1\}^{|\mathcal{V}|N^2}, w_{ii}^k = 0 \\ u \text{ satisfies (6) and (8)} \end{array} \right\}. \tag{9}$$

Note that since (6) and (8) are time dependent, $\mathcal{U}(t)$ is time dependent. With this formulation, we have modeled an AMoD system (without battery charging or other operational constraints) as a linear system in the form of

$$x(t+1) = Ax(t) + Bu(t) + c(t), \tag{10}$$

where $x(t)\in\mathcal{X}$, $u(t)\in\mathcal{U}(t)$, $c(t) = [c_{ij}(t)\ \ \mathbf{0}\ \ \mathbf{0}]^\intercal$, and A and B are the coefficient matrices associated with (1), (2), and (4). The vector $c(t)$ represents new customers that arrive every time step and constitutes an exogenous disturbance for the system.

A few comments are in order. First, one may wonder why information about whether a vehicle is rebalancing or ferrying a passenger is not encoded in the state vector. This is because we have assumed that as soon as a customer boards a vehicle, he/she has been serviced and the vehicle is identical to one that is rebalancing (traveling without a customer). The only thing that matters is the time at which the vehicle arrives at its destination. Second, we have assumed that each station has sufficient parking space for as many vehicles as needed. This may be indeed true if the stations are geographical regions and vehicles are loitering within the region while waiting for customers. However, limited parking spaces are a real concern especially if parking spaces serve as charging stations for electric vehicles. In Section VI we discuss how this AMoD framework can easily be extended to include limited parking capacity. In the next section we outline additional considerations associated with electric vehicles, in particular charging constraints (additional operational constraints are discussed in Section VI).

B. Charging constraints

For AMoD systems using electric vehicles, range is a major concern. To take into account the limited range of each vehicle, we define as an additional state variable the state of charge of each vehicle, $q^k(t)\in[0,1]$. A value $q^k(t)=1$ means that the batteries are fully charged while $q^k(t)=0$ means that the batteries are depleted. Vehicles' batteries discharge while driving, and can be charged at the stations while waiting for customers. The capacity of the batteries is limited, so once the batteries are full (i.e., $q^k(t)=1$), charging stops. Each vehicle's charge evolves according to

$$\begin{aligned} q^k(t+1) = \min\{q^k(t) + \alpha_c \sum_{i\in\mathcal{N}} u_i^k(t+1), 1\} \\ - \alpha_d \sum_{i\in\mathcal{N},T_i} {}^{T_i}p_i^k(t+1), \end{aligned} \tag{11}$$

where $\alpha_c > 0$ is the rate of charge at a charging station and $\alpha_d > 0$ is the rate of discharge while driving.

The charge of the vehicle restricts its range, and in some scenarios, a vehicle may have to wait at its charging station to charge rather than to transport a waiting customer. The charging constraints ensure that each vehicle has enough charge to complete its trip:

$$q^k(t) \geq v_{ij}^k(t)\,\alpha_d\,t_{ij}, \tag{12}$$

$$q^k(t) \geq w_{ij}^k(t)\,\alpha_d\,t_{ij}. \tag{13}$$

Constraint (12) ensures enough charge for a customer trip and (13) ensures enough charge for a rebalancing trip.

C. Objectives

The primary objective is to service all of the waiting customers as quickly as possible. A secondary goal is to ensure that rebalancing is done in an efficient manner and that vehicles do not rebalance when not necessary to avoid adding congestion on the road. Hence, for each time step t we have the cost functions

$$J_x(x(t)) = \sum_{i,j\in\mathcal{N}} d_{ij}(t), \text{ primary objective,} \tag{14}$$

$$J_u(u(t)) = \sum_{k\in\mathcal{V}} \sum_{i,j\in\mathcal{N}} t_{ij} w_{ij}^k(t), \text{ secondary objective.} \tag{15}$$

When charging constraints are considered, we may include the vehicles' final state of charge as an objective to maximize. This allows us to trade off short-term quality-of-service

and long-term battery capacity. To this end, we define the cost function

$$J_c(x(t_{\text{hor}})) = \sum_{k\in\mathcal{V}} q^k(t_{\text{hor}}). \tag{16}$$

III. Problem Formulation

The objective of this paper is to design model predictive control algorithms that are 1) provably stable in the sense of Lyapunov and 2) robust against the exogenous disturbance $c(t)$ (customer arrivals). We can now rigorously formulate the first problem, namely the AMoD regulation problem:

> **AMoD Regulation Problem (ARP)**: Assume $c_{ij}(t) = 0$ for all time $t > 0$. For each time t, select feasible control inputs $u(t) \in \mathcal{U}(t)$ such that as $t \to \infty$, $J_x(x(t)) \to 0$.

In the definition of the ARP it is assumed that the exogenous disturbance $c(t)$ is identically equal to zero for $t > 0$, in other words no new customers arrive after time zero. Hence, the ARP captures the *minimal* requirement that an initial set of customers is eventually transported to the respective destinations, under the assumption of zero future customer arrivals—hence the name "regulation problem."

While the ARP can also be solved by straightforward algorithms such as nearest neighbor dispatch (presented in Section V), it is nevertheless critical to show that our MPC algorithm does not only offer good real-world performance but also guarantees analytical stability. The second objective, robustness against the exogenous disturbance $c(t)$, is analyzed in simulation in Section V.

In the ARP, we note that $J_x(x(t)) \to 0$ will cause $J_u(u(t)) \to 0$ which implies that $u(t) \to \mathbf{0}$ by the definitions of $J_x(x(t))$ and $J_u(u(t))$.

Before presenting our MPC algorithms, we show some important structural properties of our problem setup. We first show that given $x(t) \in \mathcal{X}$, and $u(t) \in \mathcal{U}(t)$, the state of the undisturbed system at the next time step, $x(t+1) = Ax(t) + Bu(t)$, automatically satisfies (3) and (5), and thus $x(t+1) \in \mathcal{X}$. This property ensures the *persistent feasibility* of the MPC algorithms presented in Section IV. The proof of the following proposition can be found in [25].

Proposition 3.1 (Feasible Sets): Given $x \in \mathcal{X}$, $u \in \mathcal{U}(t)$, and x^+ given by $x^+ = Ax + Bu$, then $x^+ \in \mathcal{X}$.

With this result we can be sure that the reachable space of the system is feasible. We can now define the N-step and ∞-step reachable sets for the undisturbed system (10). These definitions will be useful when proving Lyapunov stability of our MPC algorithms.

Definition 3.2 (N-step Reachable Set): Given an initial condition $x(0) \in \mathcal{X}$, the N-step reachable set is defined recursively as

$$\mathcal{R}_{i+1} := \left\{x^+ \in \mathcal{X} \mid \exists x \in \mathcal{R}_i, u \in \mathcal{U}(i) \text{ s.t. } x^+ = Ax + Bu\right\}, \tag{17}$$

for $i = 0...N-1$ and $\mathcal{R}_0 = x(0)$.

Definition 3.3 (∞-step Reachable Set): Given $x(0) \in \mathcal{X}$, the ∞-step reachable set of system (10) subject to (9) is

$$\mathcal{R}_\infty := \limsup_{N\to\infty} \mathcal{R}_N, \tag{18}$$

where the above limit is in a set-theoretical sense (i.e., $\limsup_{N\to\infty} \mathcal{R}_N = \cap_{N\geq 1} \cup_{m\geq N} \mathcal{R}_m$).

IV. Model Predictive Control of AMoD

In this section we present two MPC algorithms to optimize vehicle scheduling and routing in an AMoD system. Specifically, the first MPC algorithm addresses the case without charging constraints (Section IV-A), while the second MPC algorithm allows the inclusion of charging constraints (Section IV-B). We prove that both algorithms solve the ARP (our technical approach is to prove stability in the sense of Lyapunov). Note that in general Lyapunov stability is a stronger result than simply proving $J_x(x(t)) \to 0$. However, due to the boundedness of the number of customers ($d_{ij}(t)$), Lyapunov stability coincides with solving the ARP in this case. We remark that proving stability in the sense of Lyapunov does not only guarantee that the number of passengers will decrease to zero (hence, wait times will not grow unbounded) but also implies that, if initial conditions are small (i.e. few passengers are requesting service), wait times will also be small. We numerically characterize the performance of the MPC algorithms in Section V (in particular, we study their ability to deal with a continuous stream of arriving customers).

A. MPC without charging constraints

In this section we present the MPC algorithm for solving the AMoD regulation problem without charging constraints. In an MPC algorithm, an optimization problem is solved at each time instant giving a sequence of control actions up to a time horizon t_{hor}. The first step of the control sequence is implemented and the system is re-optimized at the next time instant. Let $u(t+k)_{|t}$ be the control action at time $t+k$, solved at time t, where $k \in \{0, t_{\text{hor}} - 1\}$.

Algorithm 1 (MPC without charging constraints): Given $x(t) \in \mathcal{X}$, at each time instant $t \in \mathbb{N}$ the controls $u(t)_{|t}, u(t+1)_{|t}, \ldots, u(t+t_{\text{hor}}-1)_{|t}$ are obtained by solving the optimization problem

$$\begin{aligned}
\underset{u(t),\ldots,u(t+t_{\text{hor}}-1)}{\text{minimize}} \quad & \sum_{\tau=t}^{t+t_{\text{hor}}-1} J_x(x(\tau+1)) + \rho_1 J_u(u(\tau)) \\
\text{subject to} \quad & x(\tau+1) = Ax(\tau) + Bu(\tau) \\
& x(\tau+1) \in \mathcal{X} \\
& u(\tau) \in \mathcal{U}(\tau) \\
& \tau = t, \ldots, t + t_{\text{hor}} - 1.
\end{aligned}$$

where $\rho_1 > 0$ and $J_x(x(\tau))$ and $J_u(u(\tau))$ are given by (14) and (15), respectively. Implement $u(t)_{|t}$ and repeat the optimization at the next time instant.

Remark 4.1: The purpose of $\rho_1 J_u(u(\tau))$ in the objective is to avoid unnecessary vehicle rebalancing. However, since not enough rebalancing may result in customers not receiving service, this term is secondary to the primary objective of servicing customers so ρ_1 should be set to a small value.

We are now ready to present the main result of this section, which shows that Algorithm 1 solves the ARP problem.

Theorem 4.2 (Asymptotic stability of Algorithm 1): Suppose $t_{\text{hor}} \geq 2\max_{i,j\in\mathcal{N}} t_{ij}$. Then Algorithm 1 solves the AMoD regulation problem.

The proof of Theorem 4.2 can be found in the Appendix of [25]. The key idea of the proof is to show that at least one customer is serviced every t_{hor} time steps. We can do this by defining a new linear system equivalent to (10) where

t_{hor} time steps in (10) correspond to one time step in the new system. We can then use an extension of Lyapunov stability for set-valued functions ([23, Theorem 4]) to prove the asymptotic stability of the system.

B. MPC with charging constraints

In this section we extend the results in the previous section to account for range limitations and charging constraints associated with electric vehicles. To do this, we first augment the state vector $x(t)$ with the charge of each vehicle, $q^k(t)$. The new state vector becomes $x' = [d_{ij}\ {}^{T_i}p_i^k\ u_i^k\ q^k]^\intercal$. The state $q^k(t)$ is propagated at each time step according to (11). However, (11) is piecewise linear (due to the $\min$ operator) so the extended system cannot be written in the form of (10). As we will see, this will not be an issue for the MPC algorithm. Finally, we add the range constraints (12) and (13) to the definition of $\mathcal{U}(t)$. To summarize, the augmented feasible states and controls are

$$\mathcal{X}' = \left\{ x' = [x\ q^k]^\intercal \;\middle|\; \begin{array}{l} x \in \mathcal{X} \\ q^k \in \mathbb{R}^{\mathcal{V}},\ 0 \le q^k \le 1 \end{array} \right\}, \tag{19}$$

$$\mathcal{U}'(t) = \left\{ u' = [v_{ij}^k\ w_{ij}^k]^\intercal \;\middle|\; \begin{array}{l} v_{ij}^k \in \{0,1\}^{|\mathcal{V}|N^2}, \quad v_{ii}^k = 0 \\ w_{ij}^k \in \{0,1\}^{|\mathcal{V}|N^2}, \quad w_{ii}^k = 0 \\ u' \text{ satisfies 6, 8, 12, and 13} \end{array} \right\}. \tag{20}$$

We turn our attention back to $q^k(t)$ and notice that from (11), $q^k(t+1)$ satisfies the following two linear inequalities

$$q^k(t+1) \le q^k(t) + \alpha_c \sum_{i \in \mathcal{N}} u_i^k(t+1) - \alpha_d \sum_{i \in \mathcal{N}, T_i} {}^{T_i}p_i^k(t+1) \tag{21}$$

$$q^k(t+1) \le 1 - \alpha_d \sum_{i \in \mathcal{N}, T_i} {}^{T_i}p_i^k(t+1). \tag{22}$$

Equation (11) can be satisfied in our MPC algorithm by satisfying (21) and (22) and maximizing q^k.

Algorithm 2 (MPC with charging constraints): At each time instant $t \in \mathbb{N}$ the controls $u'(t)_{|t}, u'(t+1)_{|t}, \ldots u'(t+t_{\text{hor}}-1)_{|t}$ are obtained by solving the optimization problem

$$\begin{aligned} \underset{u'(t),\ldots,u'(t+t_{\text{hor}}-1)}{\text{minimize}} \quad & \sum_{\tau=t}^{t+t_{\text{hor}}-1} \Big(J_x(x(\tau+1)) + \rho_1 J_u(u'(\tau)) \\ & - \rho_2 \sum_{k \in \mathcal{V}} q^k(\tau+1) \Big) - \rho_c J_c(x(t_{\text{hor}})) \\ \text{subject to} \quad & x(\tau+1) = Ax(\tau) + Bu(\tau) \\ & q^k(\tau+1) \le q^k(\tau) + \alpha_c \sum_{i \in \mathcal{N}} u_i^k(\tau+1) - \\ & \alpha_d \sum_{i \in \mathcal{N}, T_i} {}^{T_i}p_i^k(\tau+1) \\ & q^k(\tau+1) \le 1 - \alpha_d \sum_{i \in \mathcal{N}, T_i} {}^{T_i}p_i^k(\tau+1) \\ & x'(\tau+1) \in \mathcal{X}' \\ & u'(\tau) \in \mathcal{U}'(\tau) \\ & \tau = t, \ldots, t + t_{\text{hor}} - 1 \end{aligned}$$

where $\rho_1 > 0$, $\rho_2 > 0$, and $\rho_c > 0$. Implement $u'(t)_{|t}$ and repeat the optimization at the next time instant.

The next theorem shows that Algorithm 2 solves the ARP with charging constraints.

Theorem 4.3 (Asymptotic stability of Algorithm 2): Suppose $t_{\text{hor}} \ge 2(1 + \frac{\alpha_d}{\alpha_c}) \max_{i,j \in \mathcal{N}} t_{ij}$. Then Algorithm 2 solves the AMoD regulation problem with charging constraints.

The proof for this theorem follows the same procedure as the proof of Theorem 4.2 and can also be found in [25]. As in Theorem 4.2, we define a time-scaled version of (10). The key difference is to note that a vehicle may be completely depleted of charge after a trip and requires $(\alpha_d/\alpha_c)t_{ij}$ time steps to charge before departing on its next trip. We can further observe that fast charging reduces the time horizon needed to maintain stability.

Remark 4.4: The time horizon bounds given in Theorem 4.2 and 4.3 assume the worst case scenario, which would very rarely occur in practice. Thus, in most practical cases, a shorter time horizon (which reduces computational cost) should also reduce d_{ij} to zero. In Section V-A we will show this is indeed the case.

In the next section we show through simulation that the MPC algorithms solve the AMoD regulation problem and we benchmark their performance against other algorithms in the literature.

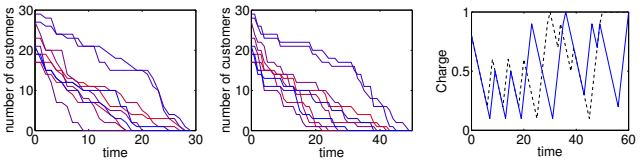
V. Simulation Results

In this section, we present three sets of simulation results that demonstrate the correctness and performance of our MPC approach. First, we show through simulation that the AMoD regulation problem can indeed be solved using Algorithm 1 and Algorithm 2. Second, we show using real taxi data that Algorithm 1 yields real-time performance on small to medium-sized systems and outperforms several state-of-the-art algorithms from the literature in terms of customer wait times. Finally, we study how the charge/discharge rate of batteries affect the performance of an AMoD system with electric vehicles. For all simulations, Algorithm 1 and 2 were implemented using the IBM CPLEX solver for mixed-integer linear programs (MILP) [26].

A. AMoD regulation

To validate Algorithms 1 and 2, an initial d_{ij} was randomly generated with up to 30 customers at each station while $c_{ij}(t)$ was set to zero for all t. The simulation was performed with 30 vehicles and 10 stations, with 3 vehicles at each station to begin with. The maximum travel time between two stations was 7 time steps. For the system without charging constraints, t_{hor} was set to 10 steps while for the system with charging constraints, t_{hor} was set to 20. Figure 1(a) shows the number of customers waiting at each of the 10 stations as a function of time. Figure 1(b) shows the number of customers vs. time for a system with charging constraints. In this case, the initial charge of all vehicles was set to 0.8 and vehicles could charge twice as fast as they could discharge ($\alpha_c = 0.2$ and $\alpha_d = 0.1$). Figure 1(c) shows the charge levels of two of the vehicles. It appears that when there are multiple outstanding customers, a general strategy for each vehicle is to service customers until its batteries are almost depleted, then charge just enough to service the next customer. The need to recharge after trips results in longer

wait times and in this case, a longer total time to service all the customers (50 minutes for the case with charging constraints and 30 minutes without).



(a) (b) (c)

Fig. 1. 1(a): Number of customers waiting at each station as a function of time for 10 stations, 30 vehicles without charging constraints. 1(b): Number of customers waiting at each station as a function of time with charging constraints. 1(c): State of charge for two vehicles as a function of time.

B. Performance of MPC

To evaluate the performance of our MPC algorithm, we conducted an extensive simulation study comparing Algorithm 1 to several other AMoD and taxi dispatch algorithms found in the literature using real New York taxi data[1]. We show that Algorithm 1 not only outperforms other algorithms in terms of customer wait times, but can be used as an "optimal" baseline to quantitatively evaluate the performance of other taxi dispatch or AMoD algorithms. The simulations are performed with 40 vehicles for 24 hours with a time step of 6 seconds. Taxi trips within New York City's Financial District area (Lower Manhattan, south of Canal St.) are extracted for nine Mondays in March and April of 2012, resulting in 2300-3500 trips per day. The simulated vehicles move along the Manhattan distance between pickup and drop-off locations, with speeds estimated from the data to account for congestion. The Financial District is divided into 15 regions: the center of each region (a "station") is computed using *k*-means clustering on historical customer origin/destination data. In the simulation, customers are *not* required to go to a station to receive service: once a vehicle is assigned to a customer, it drives to the customer's location and drops her/him off at the requested destination, then drives to the nearest station. The role of stations is to 1) model the availability of parking and charging facilities and 2) provide a discretized model for the vehicle rebalancing problem.

For this simulation study, we implemented six dispatch algorithms, including two versions of Algorithm 1:

1) *Nearest-neighbor dispatch (NN):* Each customer is assigned the nearest free vehicle. If no vehicles are available, the customer request is added to a first-in, first-out queue. Free vehicles move according to a random walk until assigned to a customer.
2) *Collaborative dispatch (CD) [14]:* customer requests are aggregated in a queue (a single queue is used for the entire Financial District) and, when the queue size reaches a threshold x, the same number of free vehicles are dispatched to the customers. Vehicles are matched to customers to minimize the total distance they need to drive empty. The algorithm is modified from [14] in two ways: 1) a time-varying queue size $x(t)$ is employed to account for highly time-varying demand, and 2) the optimization is solved in a centralized fashion.
3) *Markov redistribution (MR) [13]:* Customer demand information is used to rebalance empty vehicles among stations in order to drive the vehicles' distribution towards the distribution of passenger arrivals. The problem is cast as a linear program (LP) and yields a randomized rebalancing strategy for each empty vehicle. The algorithm implemented in our simulation is modified from [13] in two ways: 1) the problem is solved exactly as an LP and 2) in order to accommodate time-varying demand, the algorithm rebalances vehicles based on the sum of (i) estimated future customer demand and (ii) current number of passengers waiting. Since the MR algorithm is randomized, its performance can vary widely between trials: for each day, the results presented are the median of ten executions.
4) *Real-time rebalancing (RR) [5]:* This algorithm rebalances vehicles based on current waiting customers. At each rebalancing epoch (every 2 minutes) the algorithm computes the number of vehicles at or enroute to each station and solves a linear program to evenly distribute excess free vehicles throughout the system. It uses the same fifteen stations as the MR algorithm.
5) *Algorithm 1, MPC with sampled customer arrivals (MPCS):* The algorithm uses the same 15 stations employed by MR and solves the problem with a time horizon of 15 minutes. Customer arrival rates, computed using historical data, are sampled as a Poisson process and fed into Algorithm 1 as predicted future arrivals (c_{ij}). This sampling is done every 2 minutes to prevent unnecessary rebalancing. Finally, a small term is added to the cost function to promote a uniform distribution of vehicles among the stations at the end of the optimization horizon.
6) *Algorithm 1, MPC with full arrival information (MPCF):* The actual customer arrivals over the time horizon (15 minutes) are fed into Algorithm 1 as c_{ij}. The algorithm optimizes vehicle assignments with perfect knowledge of the next 15 minutes, and can therefore serve as a baseline for optimal performance, as long as wait times are small compared to the optimization horizon.

Table I shows the peak customer wait times for each algorithm over the nine days that were simulated (every Monday from March 5 to April 30, 2012). The algorithm that yielded the best performance for each day (shortest peak wait time) is shown in bold (MPCF was excluded since it is non-causal). We note that in all the days where the peak wait time for MPCF is less than the optimization horizon of 15 minutes (day 3, 5, 6, and 9), MPCF achieves the best performance and can effectively serve as an optimal baseline. When the customer demand is high and the wait time is greater than the optimization horizon, the MPC algorithms (MPCS and MPCF) cannot optimize far enough into the future and thus performance suffers. Nevertheless, MPCS achieve the best performance in 7 of the 9 days simulated. It's also worth noting that over the 4 days with short wait times, MPCS achieved peak wait times that were on average 34% shorter than the next best algorithm, RR. Figure 2 shows the simulation results for day 5 (April 2). In addition to achieving a lower peak wait time, the MPC algorithms are also able to

[1]Courtesy of the New York City Taxi & Limousine Commission

recover quickly to service the remaining customers after peak demand. This is illustrated in Table II, which lists the fraction of time spent for each algorithm where the average wait time was at least 50% of the respective peak. In this respect, MPCS again consistently outperforms the other algorithms.

A few observations about the other algorithms are in order. First, performance of the CD algorithm is generally slightly better than the NN algorithm, and is consistent with the results obtained by the authors in [14]. However, the problem of selecting a threshold queue size (i.e. the algorithm's tuning parameter) to maximize performance remains open. As a matter of fact, in three instances the NN algorithm outperforms CD in our simulations. Second, as neither NN nor CD rebalance empty vehicles so as to anticipate future demand, their performance highlights the critical importance of preemptive routing to achieve good quality-of-service. Third, since the MR algorithm is a randomized algorithm, its performance can be (and occasionally is) very suboptimal and, at times, significantly worse than NN. Median performance (over 10 runs), however, is generally better than both NN and CD. Finally, MR is conceived for steady-state systems. Thus, while the algorithm can be extended to time-varying systems, it is unclear whether the performance presented in [13] can be replicated in scenarios with highly variable customer demand.

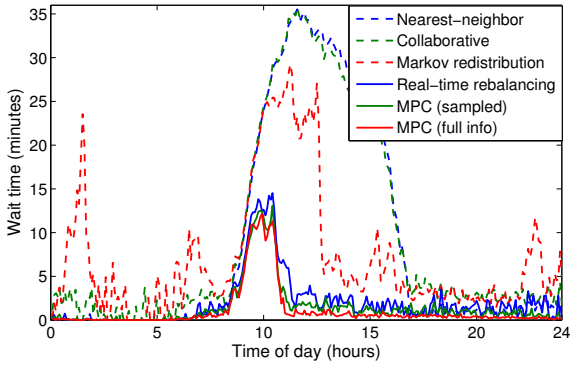


Fig. 2. Average customer wait times throughout the day (April 2, 2012) for all dispatch algorithms.

TABLE I

PEAK WAIT TIME IN MINUTES FOR EACH ALGORITHM

Day	1	2	3	4	5	6	7	8	9
NN	36	34	27	56	36	9	38	41	24
CD	35	34	24	56	36	10	39	41	25
MR	37	53	11	31	29	13	32	35	16
RR	**16**	16	7	20	15	7	27	**21**	10
MPCS	17	**15**	**4**	**18**	**13**	**3**	**24**	24	**7**
MPCF*	18	19	3	19	12	2	29	24	6

TABLE II

FRACTION OF TIME WHERE THE AVERAGE WAIT TIME WAS AT LEAST 50% OF PEAK

Day	1	2	3	4	5	6	7	8	9
NN	0.15	0.16	0.18	0.34	0.27	0.07	0.17	0.19	0.12
CD	0.15	0.16	0.16	0.35	0.26	**0.07**	0.17	0.19	0.11
MR	0.09	0.06	0.12	0.09	0.13	0.08	0.10	0.08	0.09
RR	0.08	0.07	0.11	0.09	0.07	0.07	**0.07**	0.06	0.05
MPCS	**0.06**	**0.05**	**0.09**	**0.08**	**0.06**	0.17[2]	0.08	**0.06**	**0.04**
MPCF*	0.07	0.04	0.06	0.08	0.06	0.08	0.07	0.06	0.03

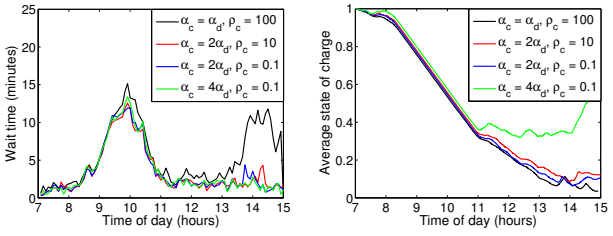
[2]Note that the peak wait time for MPCS is only 3 minutes

The median runtime per iteration of Algorithm 1 on a 2.8 GHz Intel Core i7 PC with 16GB of RAM was 5.5 seconds.

C. Effect of charge rate

In this section we explore the performance limitations of AMoD systems with electric vehicles. Specifically, we would like to answer the question: how does charging rate affect the ability of an AMoD system to service customer demand? To this end, simulations were performed using 40 vehicles with taxi data from 7 am to 3 pm (period of high demand) on April 2, 2012. The simulations were performed for the MPCS algorithm (see Section V-B) with charging constraints (Algorithm 2). The discharge rate, α_d, was chosen to be 0.0037, which corresponds to an electric vehicle such as a Nissan Leaf or BMW i3 driving at 20 km/h while using 70% of its battery capacity (to avoid over-discharging, which could damage the batteries). Three charging rates were used: $\alpha_c = \alpha_d$, $\alpha_c = 2\alpha_d$, and $\alpha_c = 4\alpha_d$, which correspond to a charging time of 4 hours, 2 hours, and 1 hour, respectively. The charging times represent realistic current electric vehicle charging capabilities. To avoid prematurely depleting the batteries, a higher final charging cost ρ_c was assigned to the cases with slower charge rates. Figure 3(a) shows the customer wait times for the four simulations performed, and 3(b) shows the average state-of-charge of vehicles over time.

We first note that as long as there is excess battery capacity in the system, the customer wait times for an electric AMoD system are comparable to the AMoD system without charging constraints (see Figure 2). In this case, a charge rate of $4\alpha_d$ is stabilizing and is able to support future demand. A charge rate of $2\alpha_d$ is able to service demand without increasing wait time, but the batteries are almost fully depleted by the end of the simulation. A charge rate of α_d is too slow to support customer demand for the entire simulation duration. The large final charging cost $\rho_c = 100$ trades off quality of service with battery capacity, resulting in a slightly higher peak wait time. Even so, batteries become depleted near the end of the simulation period and the algorithm begins to prefer battery charging over servicing customers, resulting in longer wait times.



(a) (b)

Fig. 3. 3(a): Average customer wait time for April 2, with charging constraints. 3(b): Average vehicle charge as a function of time for different charging rates and final charging costs.

VI. ADDITIONAL MODEL EXTENSIONS

While we have mainly focused on extending the AMoD model for charging constraints, many other real-world constraints can be directly incorporated into our modeling framework with little modification. We briefly touch on some of the possible extensions to highlight the flexibility and potential of our approach.

1) **Limited number of charging stations**. Let $h_i, i \in \mathcal{N}$, represent the number of charging stations at each station. We can assume that $\sum_{i\in\mathcal{N}} h_i \geq |\mathcal{V}|$, that is, there is at least one charger for each vehicle in the system. Vehicles can only wait at a charging station, though they can still pick up and drop off passengers at stations with no free charging stations. This adds the constraint $\sum_{k\in\mathcal{V}} u_i^k(t) \leq h_i$ for all $i \in \mathcal{N}$. A limited number of charging stations will also promote rebalancing, as it forces vehicles to travel to stations with free charging stations (or likely a deficit of vehicles).
2) **Customer priorities**. Taking into account priority waiting involves simply adding a weighting matrix to the objective function. Rather than minimizing $\sum_{\tau=t}^{t+t_{\text{hor}}-1} J_x(x(\tau+1))$ we minimize $\sum_{\tau=t}^{t+t_{\text{hor}}-1} Q(\tau+1)J_x(x(\tau+1))$. In this way, we can give a higher priority to customers who have been waiting for a longer period of time, or assign the weights based on price incentives. This approach can also be used for customer arrivals with known time windows.
3) **Interaction with the smart grid**. Electric vehicles may act as energy storage devices to enable intermittent renewable energy such as solar and wind [6]. Vehicles can "sell" their energy to the grid during peak hours and charge themselves during off-peak hours. If the charging/discharging schedule is known, the charging rates α_c can be adjusted accordingly to facilitate the energy transfer, at the same time maintaining quality of service in the AMoD system.

VII. Conclusions and Future Work

In this paper we presented a model predictive control approach to optimize vehicle scheduling and routing in an AMoD system. Our approach allows the easy integration of a number of real-world constraints, in particular we focused on charging constraints. We presented two MPC algorithms and rigorously showed that they are able to regulate an AMoD system (i.e., drive an initial customer demand to zero assuming no additional customers arrive over time). Algorithm performance for the case of dynamic arrivals was evaluated using real-world data through simulations. Overall, numerical results show that the proposed MPC algorithms outperform previous control strategies for AMoD systems.

This paper leaves numerous important extensions open for further research. First, we plan to address the algorithmic aspect of scaling up the MILP formulation of the optimization problem to large-scale (i.e., city-wide) systems. This will likely require the use of parallel architectures and ad hoc approximations. Second, we plan to study the inclusion of additional operational constraints (e.g., time windows), address the congestion aspect (currently roads are assumed to have infinite capacity), and study the impact on computation time. Third, it is of interest to couple an AMoD system with alternative mass transit options and develop an MPC approach for the optimal coordination algorithms of such an intermodal system. Fourth, we plan to consider additional, larger-scale case studies to derive economic guidelines about the development of AMoD systems. Finally, we plan to demonstrate the algorithms on real driverless vehicles providing AMoD service in a gated community.

References

[1] U.S. Energy Information Administration, "International energy outlook 2013," Tech. Rep., 2013.
[2] United Nations Environment Programme, "The emissions gap report 2013 - UNEP," United Nations, Tech. Rep., 2013.
[3] UN, "World Urbanization Prospects: The 2014 Revision," United Nations, Tech. Rep., 2014.
[4] Uber, "Uber: Your Ride, On Demand," 2015.
[5] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus, "Robotic load balancing for mobility-on-demand systems," *International Journal of Robotics Research*, vol. 31, no. 7, pp. 839–854, Jun. 2012.
[6] W. J. Mitchell, C. E. Borroni-Bird, and L. D. Burns, *Reinventing the Automobile: Personal Urban Mobility for the 21st Century*. Cambridge, MA: The MIT Press, 2010.
[7] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: A queueing-theoretical perspective," *International Journal of Robotics Research*, 2015, in press.
[8] Google, "Just press go: designing a self-driving vehicle." Tech. Rep., 2014.
[9] Ligier Group, "Launch of EasyMile: "the mobility of the last mile even easier"." Tech. Rep., 2014.
[10] S. L. Smith, M. Pavone, M. Schwager, E. Frazzoli, and D. Rus, "Rebalancing the rebalancers: Optimally routing vehicles and drivers in mobility-on-demand systems," in *American Control Conference*, Washington, DC, Jun. 2013, pp. 2362–2367.
[11] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, and M. Pavone, "Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems: A case study in Singapore," in *Lecture Notes in Mobility*, ser. Lecture Notes in Mobility. Springer, Jun. 2014, pp. 229–245.
[12] R. Zhang and M. Pavone, "A queueing network approach to the analysis and control of mobility-on-demand systems," in *American Control Conference*, Chicago, IL, Jul. 2015, pp. 4702–4709.
[13] M. Volkov, J. Aslam, and D. Rus, "Markov-based redistribution policy model for future urban mobility networks," in *IEEE Conference on Intelligent Transportation Systems*. IEEE, 2012, pp. 1906–1911.
[14] K. T. Seow, N. H. Dang, and D. H. Lee, "A collaborative multiagent taxi-dispatch system," *IEEE Transactions on Automation Sciences and Engineering*, vol. 7, no. 3, pp. 607–616, 2010.
[15] A. Alshamsi, S. Abdallah, and I. Rahwan, "Multiagent self-organization for a taxi dispatch system," in *8th International Conference on Autonomous Agents and Multiagent Systems*, 2009, pp. 21–28.
[16] N. Kliewer, T. Mellouli, and L. Suhl, "A time–space network based exact optimization model for multi-depot bus scheduling," *European Journal of Operational Research*, vol. 175, no. 3, pp. 1616–1627, 2006.
[17] D. Jorge, G. H. A. Correia, and C. Barnhart, "Comparing optimal relocation operations with simulated relocation policies in one-way carsharing systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1667–1675, 2014.
[18] A. G. H. Kek, R. L. Cheu, Q. Meng, and C. H. Fung, "A decision support system for vehicle relocation operations in carsharing systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 45, no. 1, pp. 149–158, 2009.
[19] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn, "Efficient network-wide model-based predictive control for urban traffic networks," *Transportation Research Part C: Emerging Technologies*, vol. 24, pp. 122–140, 2012.
[20] R. R. Negenborn, B. De Schutter, and J. Hellendoorn, "Multi-agent model predictive control for transportation networks: Serial versus parallel schemes," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 3, pp. 353–366, 2008.
[21] C. Danielson, F. Borrelli, D. Oliver, D. Anderson, M. Kuang, and T. Phillips, "Balancing of battery networks via constrained optimal control," in *American Control Conference*, 2012, pp. 4293–4298.
[22] E. Franco, L. Magni, T. Parisini, M. M. Polycarpou, and D. M. Raimondo, "Cooperative constrained control of distributed agents with nonlinear dynamics and delayed information exchange: A stabilizing receding-horizon approach," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 324–338, 2008.
[23] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.
[24] SuperShuttle, "How It Works," 2015.
[25] R. Zhang, F. Rossi, and M. Pavone, "Model predictive control of autonomous mobility-on-demand systems (extended version)," Sep. 2015, available at http://arxiv.org/abs/1509.03985.
[26] *ILOG CPLEX User's guide*, ILOG, Mountain View, CA, 1999.