2015 IEEE 18th International Conference on Intelligent Transportation Systems

Dynamic Patrolling Policy for Optimizing Urban Mobility Networks

Gavin Chase Hall[1], Mikhail Volkov[2], and Daniela Rus[2]

***Abstract*— A dynamic patrolling policy is presented for a fleet of service vehicles operating in response to incident requests in an urban transportation network. We modify an existing adaptive, informative path controller so that the fleet of vehicles is driven to locally optimal service configurations within the environment. These configurations, called patrolling loops, minimize the distance between the instantaneous vehicle position and incident customer request. Our patrolling algorithm is trained using one month of data from a fleet of 16,000 vehicles. This historical dataset is used to learn the parameters required to set up a representative urban mobility model. Using this model we conduct large-scale simulations to show the global stability of the patrolling policy and evaluate the performance of our system by comparing it against a greedy service policy and historical data.**

I. Introduction

We are interested in developing optimized task allocation algorithms for Mobility-on-Demand (MOD). In our previous work [14], we showed how autonomous driving can be used to mitigate the rebalancing problem current MOD systems face. In this paper we consider the task allocation problem in a MOD scenario. In MOD transportation we assume historical knowledge of passenger arrival at discrete sets of locations. The goal is to minimize the waiting time of the passengers and the amount of time the vehicles in the system drive empty. The critical question is where should each vehicle go once a delivery is complete? In this work we present a dynamic patrolling policy that allocates vehicles to pickup and delivery tasks. Using historical arrival distributions, we compute patrolling loops that minimize the distance driven by the vehicles to get to the next request. These loops are used to redistribute the vehicles along stationary virtual taxi stand locations on the loop. The algorithm was trained using one month of data from a fleet of 16,000 taxis. We compare the policy computed by our algorithm against a greedy policy as well as against the ground truth redistribution of taxis observed on the same dates, and show an improvement with respect to three key evaluation criteria: minimizing the number of vehicles in the system, quality of service, and distance traveled empty. We show that our policy is robust by evaluating it on previous unseen test data.

*Support for this research has been provided by the Future Urban Mobility project of the Singapore-MIT Alliance for Research and Technology (SMART) and ONR grant N00014-09-11051. We are grateful for this support.

[1]G. Hall is with the Department of Mechanical Engineering, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, MA 02139, United States gvnhall at mit.edu

[2]M. Volkov and D. Rus are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, MA 02139, United States {mikhail,daniela} at csail.mit.edu

The main contributions of this paper are:

- patrolling loop and redistribution model of an unmanaged fleet operation using historical data,
- provably stable dynamic redistribution policy for a large number of vehicles using informative paths,
- centralized scheduling algorithm for request allocation and vehicle redistribution,
- large-scale simulations and evaluations using real data from a fleet of 16,000 vehicles.

A. Related Work

The dynamic traffic assignment problem (DTA) considers the optimization of traffic flow while accounting for congestion effects. The models often differ significantly in their representation of the request arrival and service processes. Work on this problem dates back to [9] and [4]. A thorough review of DTA results can be found in [15]. Mobility-on-demand (MOD) is a similar paradigm for dealing with increasing urban congestion. Generally speaking, the objective of MOD problems is to provide on-demand rental facilities of convenient and efficient modes of transportation [10]. For a thorough survey of studies to date see [2], [11] and the references therein. Load balancing in DTA problems essentially reduces to the Pickup and Delivery problem (PDP), whereby passengers arriving into a network are transported to a delivery site by vehicles. Autonomous load balancing in MOD systems has recently been studied in [12] and [13], where a fluid model was used to represent supply and demand. In this work we employ a PDP roblem formulation to model an urban transportation network.

Socially-motivated optimization criteria have also been considered in prior work. In [16], [23], social optimum planning models were used to compute vehicle paths. Optimization of driving routes subject to congestion was considered in [8]. In a broader context, [7] observed the effect that multiple service policies had on logistic taxi optimization. More recently, in [22] we studied both system-level and social optimization criteria, showing a relationship between urban planning, fuel consumption, and quality of service metrics. In this work we consider similar evaluation models, showing how we can achieve an improvement with respect to all three of these aforementioned points of interest.

Adaptive path planning considers adapting paths to unknown or dynamic environment states in continuous-time systems. For example, in [3] the authors present a path planning algorithm for deploying unmanned aerial vehicle systems. In [21], an optimization for path planning was presented for the case of partially known environments. Informative sensing extends adaptive path planning algorithms

978-1-4673-6596-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ITSC.2015.305

with an emphasis on efficiently measuring and monitoring a dynamic environment. Such a method for computing paths that provide the most information about an environment was presented in [18], with the aim of adaptively learning and traversing through regions of interest with multiple robots. Informative sensing while maintaining periodic connectivity for the robots to share information and synchronize was examined in [6]. Our work considers adaptive path planning and informative sensing in a similar context, by using sensor measurements to compute informative paths for a network of autonomous vehicles in a dynamic environment.

The controller used in this work generates informative paths by optimizing the location of waypoints according to a Voronoi-based coverage criterion in an unknown environment. This controller builds upon [20], [17] which defined a robot's path by a Voronoi decomposition, and used a parameter adaptation law to learn where and how fast the environment is changing.

II. Problem Formulation

We consider a pickup and delivery problem (PDP) in a convex bounded planar area $Q \subseteq \mathbb{R}^2$. This area or *service environment* is subject to incident request arrivals at continuous points $\mathbf{q} \in Q$. The environment is patrolled by N vehicles that drive along closed loops at constant speed. Vehicles are assigned to service requests by a centralized server, which is assumed to know the locations of all vehicles at any time. A vehicle v_i that has been assigned a request $\mathbf{q}_j$ will travel in a straight line to $\mathbf{q}_j$ to pick up the request, and then deliver it to its destination $\mathbf{s}_j \in Q$. We assume a continuous time model, i.e. time $t \in \mathbb{R}_{\geq 0}$.

A. Informative Paths

An informative path control algorithm takes as input sensory information over a dynamically changing environment and outputs locally optimal paths so that the trajectory of a patrolling agent along each path covers the areas in the environment where the sensory information is important.

In this work we modify an existing informative path controller [20] and extend it to a pickup and delivery. We observe that the regions of dynamic change are analogous to regions of pickup demand, and the act of sampling to reduce uncertainty in coverage is analogous to the act of picking up delivering incident requests. The difference is that in PDP problems we have discrete rather than continuous events, and delivery is different to sampling in that the vehicle has to deliver the request and return to the informative path.

In our model, the resulting informative path is a patrolling loop whose route is locally optimized such that it traverses along or very near the areas where pickup requests originate. By utilizing waypoints along a patrolling loop, informative paths can be visualized as method to locally optimize the location of virtual taxi stands across the environment. The goal is to compute the path and placement of the patrolling loops so that the distance from the patrolling loop to the requests is optimized. A mathematical description of this algorithm for multiple agents follows.

B. Multi-Agent Controller

There are $X \in \mathbb{R}_{>0}$ agents identified by $r \in \{1,\ldots,X\}$ in a compact, convex environment $Q \subset \mathbb{R}^2$. In this derivation, note that the number of agents represents the number of patrolling loops that are computed by our patrolling policy. A point in Q is denoted $\mathbf{q}$. Agent r is positioned at $p_r \in Q$ and travels along its closed path $\ell_r : [0,1] \mapsto \mathbb{R}^2$, consisting of a finite number n of waypoints. The ith waypoint on ℓ_r is located at p_i^r, $i \in \{1,\ldots,n\}$. Define a vector $P \in \mathscr{P}^{Xn} \subset \mathbb{R}^{dim(P)}$ as the vector obtained by making an array of the agents' waypoint positions, $P = \{p_i^r,\ldots,p_n^r\}$, where $\mathscr{P}^{Xn}$ is the state space of the waypoints for all agents. Note that the controller for the single agent is derived by setting $X = 1$.

Let V_i^r be a Voronoi partition of Q, for the ith waypoint position in agent r's path, defined as

$$\begin{aligned} V_i^r \quad = \quad & \{\mathbf{q} \in Q : \|\mathbf{q} - p_i^r\| \leq \|\mathbf{q} - p_{i'}^{r'}\|,\ \forall (r',i') \neq (r,i)\}, \\ & \text{where } r, r' \in \{1,\ldots,X\},\ i \in \{1,\ldots,n\} \\ & \text{and } i' \in \{1,\ldots,n'\}. \end{aligned} \tag{1}$$

Agents can compute the Voronoi partitions based on their waypoint positions. Because each path is closed, $\ell_r(0) = \ell_r(1)$, each waypoint i along the path has a corresponding previous waypoint $i-1$ and next waypoint $i+1$. An agent travels between sequential waypoints in a straight line.

A scalar sensory function, defined as a map $\phi : Q \mapsto \mathbb{R}_{\geq 0}$ determines the rate of change of the environment at point $\mathbf{q} \in Q$. The sensory function $\phi(q)$ is updated every fifteen minutes over the course of 24 hours in order to reflect the change in sensory information throughout the day. The agent knows $\phi(\mathbf{q})$; however, it is equipped with a sensor with sensing radius ρ, to make a point measurement of $\phi(p_r)$ at its position p_r so that it ensures the stability criterion for the persistent task [19].

C. Local Optimality

The cost incurred by the multi-agent system over the environment Q is given as

$$H = \sum_{r=1}^{X} \sum_{i=1}^{n} \int_{V_i^r} \frac{W_s}{2} \|\mathbf{q} - p_i^r\|^2 \phi(\mathbf{q}) d\mathbf{q} + \sum_{r=1}^{X} \sum_{i=1}^{n} \frac{W_n}{2} \|p_i^r - p_{i+1}^r\|^2 \tag{2}$$

where $\|\mathbf{q} - p_i\|$ is a strictly increasing function that gives the unreliability of the sensory function value $\phi(\mathbf{q})$ when the agent is at p_i, and $\|p_{i,i+1}\|$ is the cost of the distance between two sequential waypoints. $W_s \in \mathbb{Z}^+$ and $W_n \in \mathbb{Z}^+$ are weights assigned to the sensing task and waypoint neighbor distance, respectively. Note that unreliable sensing and distance between neighboring waypoints are expensive. The cost function is differentiable everywhere on $\mathscr{P}^{Xn}$ so that its partial derivative with respect to every waypoint's position is well-defined. Additionally, $\partial H / \partial p_i^r$ is locally Lipschitz which guarantees the existence of solutions in our system. A formal definition of informative paths for multiple agents follows.

We define a collection of informative paths for a multi-agent system corresponds as a set of waypoint locations for

each agent that locally minimize (2). The mass, first mass-moment, and centroid of the Voronoi partitions for the system are given by $M_i = \int_{V_i} W_s \phi(\mathbf{q}) d\mathbf{q}$, $L_i = \int_{V_i} W_s \mathbf{q} \phi(\mathbf{q}) d\mathbf{q}$, and $C_i = L_i/M_i$, respectively [17]. Note that $f(\|\mathbf{q} - p_i\|)$ strictly increasing and $\phi(q)$ strictly positive imply both $M_i V_i > 0 \ \forall \ V_i \neq \emptyset$ and $C_i V_i \in V_i / \partial V_i$ ($C_i V_i$ is in the interior of V_i). Thus $M_i V_i$ and $C_i V_i$ have properties intrinsic to physical masses and centroids. Letting $e_i^r = C_i^r - p_i^r$ and, following the same procedure in [20], we have

$$\frac{\partial H}{\partial p_i^r} = -M_i^r e_i^r - W_n(p_{i+1}^r + p_{i-1}^r - 2p_i^r). \tag{3}$$

An equilibrium is reached when $\frac{\partial H}{\partial p_i^r} = 0$. Assigning to each waypoint integrator dynamics of the form $\dot{p}_i^r = u_i^r$ where u_i^r is the control input, we propose the following gradient descent control law for the waypoints to converge to an equilibrium configuration:

$$u_i^r = \frac{-K_i^r}{\beta_i^r} \frac{\partial H}{\partial p_i^r} = \frac{K_i(M_i^r e_i^r + \alpha_i^r)}{\beta_i^r}, \tag{4}$$

where $\alpha_i^r = W_n(p_{i+1}^r + p_{i-1}^r - 2p_i^r), \beta_i^r = M_i^r + 2W_n > 0, K_i^r$ is a uniformly positive definite matrix. Note that $\beta_i^r > 0$ normalizes the weight distribution between sensing and staying close to neighboring waypoints.

Theorem 1 (Convergence Theorem for Multiple Agents) *The path will reach a locally optimal configuration for sensing, as defined by $\partial H / \partial p_i^r = 0$.*

Proof: We define a Lyapunov-like function based on the agent's path and environment measurement. Because the system is autonomous, we use LaSalle's Invariance Principle to prove asymptotic stability of the system to a locally optimal equilibrium.

Let H be the Lyapunov function candidate. H is positive definite, radially unbounded, and has continuous first partial derivatives. Domain Q is bounded, therefore the state space $\mathscr{P}^{Xn}$ is bounded, as is the Voronoi state space for all agents $P_v^r = \{P = [(p_1^r)^T \dots (p_n^r)^T]^T \mid p_i^r \neq p_j^r \ \forall i \neq j\} \subset \mathscr{P}^{Xn}$. Let $\Omega = \{P^* \mid \dot{H}_{P^*} = 0\} \subset P_v^r$. Let $\Omega = \{P^* \mid \dot{H}_{P^*} = 0\}$ be the set of all critical points of H over $\mathscr{P}^{Xn}$. Taking the time derivative of H, we obtain $\dot{H} = \sum_{r=1}^{X} \sum_{i=1}^{n} -\frac{1}{\beta_i^r} (M_i^r e_i^r + \alpha_i^r)^T K_i^r (M_i^r e_i^r + \alpha_i^r) \leq 0$.

Therefore Ω is defined by the set of solutions of $\sum_{r=1}^{X} \sum_{i=1}^{n} M_i^r e_i^r + \alpha_i^r = 0, \forall i$. Let S be the largest invariant set within Ω. By definition $\dot{p}_i = \partial H / \partial p_i^r = K_i(M_i^r e_i^r + \alpha_i^r)/\beta_i^r$, from which it follows that $S = \Omega$, the set of all critical points of H. Thus Ω itself is an invariant set, and all trajectories converge to Ω as $t \to \infty$ using LaSalle's Invariance Principle. From (4), $M_i^r e_i^r + \alpha_i^r \to 0$ implies $\partial H / \partial p_i^r = 0$. ■

D. Computational Complexity

At each iteration the controller must compute the Voronoi cell for each waypoint and the spatial integrals over the region. Thus the parameters affecting the computation time are the number of agents X, the number of waypoints n, and the number of grid squares in the integral computation m.

A decentralized algorithm for a single agent to compute its Voronoi cell [5] runs in $O(n)$ time. The time complexity for computing a discretized integral is linear in the number of grid squares, and at each grid square requires a check if the center point is within the Voronoi cell, which is $O(n)$. Therefore the time complexity of the integral is in $O(nm)$. If the Voronoi cell is computed first, followed by the discretized integral, the total time complexity $O(n(m+1))$ at each step of the control loop. Therefore in the multi-agent case a single Voronoi controller has time complexity $O(n(Xm+1))$.

E. Operational Stability

In addition to controller stability, a necessary condition for a functional deployment of any fleet of service vehicles is operational stability. Informally, we understand operational stability to mean the condition whereby the number of outstanding requests remains bounded in steady state. Formally, we define operational stability as the condition

$$\int_0^\tau \lambda(t)\, dt < k\lambda, \ \forall \ t \geq 0, \ k < \infty \ . \tag{5}$$

To motivate this requirement, consider events arriving into a queue according to a standard Poisson process with rate parameter λ. Then the integral gives us the total number of arrivals from the beginning of time until the current time t. If the events are also being serviced at a sufficient rate according to some other process then the number of events in the queue will be less than some constant times the rate parameter for any time window.

The service rate $\mu(t)$ is defined as the rate at which incident customer requests are being serviced by vehicles. In steady state, the stability requirement in (5) is satisfied by the simplified expression $\overline{\mu(t)} > \overline{\lambda(t)}$, where $\overline{\lambda(t)}$ and $\overline{\mu(t)}$ denote the average arrival and service rates, respectively, over the open interval $(0, \infty)$.

Lemma 1 *Let C be a closed curve in Euclidean space, $Q \subseteq \mathbb{R}^2$, composed of n waypoints, $\{w_1, w_2, ..., w_n\}$, that are connected by straight lines. Let x and y be any two points on C. Then $d(x,y) \leq \frac{L(C)}{2}$, where $L(C)$ is the arc length of C and $d(x,y)$ is the Euclidean distance between x and y .*

Proof: The arc length of C is given by $L(C) = \sum_{i=1}^{n} d(w_i, w_i + 1)$, where waypoint $w_{n+1} = w_1$, and $d(w_i, w_{i+1})$ is the Euclidean distance between consecutive waypoints. Let A and B be unique segments of curve C that connect point x to point y, such that $L(A) + L(B) = L(C)$. Without loss of generality, assume $L(A) \leq L(B)$, which implies that $L(A) < \frac{L(C)}{2}$.

Assume for the sake of contradiction, $d(x,y) \geq \frac{L(C)}{2}$. Because $d(x,y)$ is by definition the minimum distance between any two points in Q, there exists no path $P \in Q$, such that $P \leq \frac{L(C)}{2}$. However, $L(A) \leq \frac{L(C)}{2}$, thus giving the contradiction. ■

Theorem 2 (Steady State Stability) *An informative path service policy gives $\overline{\mu(t)} > \overline{\lambda(t)}$ if and only if $X > \lambda \left(\frac{L(C)}{2} + \rho + 2\sqrt{2}l \right) / v$, where X is the number of service*

vehicles, ρ is the sensor radius of a service vehicle, l is the dimension of the square environment Q, v is the constant speed of the service vehicle.

Proof: ($\Leftarrow$) $X > \lambda(\frac{L(C)}{2} + \rho + 2\sqrt{2}l)/v \Rightarrow Xv/(\frac{L(C)}{2} + \rho + 2\sqrt{2}l) > \lambda$. Let $\alpha = v/(\frac{L(C)}{2} + \rho + 2\sqrt{2}l)$, which has units s^{-1}, assuming the time step τ of the experiment is $1s$. Using the convergence stability of the informative path controller from Sections II-C and II-E , no incident customer request is located further than ρ from the informative path. By Lemma 1, $\sup\{d : d = \|x-y\|,\ \forall x,y \in Q\} = L(C)/2$. Thus, the maximum distance any vehicle on the path is from an incident customer request located at $\mathbf{q}$ is $L(C)/2 + \rho$. Because Q is a square environment, the maximum distance it takes a service vehicle to drive the customer to its destination, and return to the informative path is $2\sqrt{2}l$. Thus the maximum distance traveled by a service vehicle whose trip originates on the informative path to service a request is $L(C)/2 + \rho + 2\sqrt{2}l$. Therefore α is now the smallest rate at which a vehicle can service a request, and hence $X\alpha$ is the smallest $\overline{\mu(t)}$ that satisfies $\overline{\mu(t)} > \overline{\lambda(t)}$.

($\Rightarrow$) By the contrapositve law, $X < \lambda(\frac{L(C)}{2} + \rho + 2\sqrt{2}l)/v \Rightarrow Xv/(\frac{L(C)}{2} + \rho + 2\sqrt{2}l) < \lambda$. Again, by letting α denote $v/(L(C)2 + \rho + 2\sqrt{2}l)$, the service rate for a single service vehicle, we have $X\alpha < \lambda$, which implies $\overline{\mu(t)} < \overline{\lambda(t)}$. ■

This result shows that our approach to task allocation for PDP in MOD systems is stable. Next we present a patrolling policy for the delivery vehicles.

III. Dynamic Patrolling Policy

Our case study for this work is a PDP in a MOD system and uses real data provided by a fleet of 16,000 taxis in Singapore. We will refer to vehicles as taxis for the rest of the paper. We illustrate the operation of our algorithm for the Central Business District (CBD) and extend it for the entire island of Singapore. We evaluate how effective our solution is at minimizing the amount of time taxis drive empty by comparing against a greedy policy as well as what actual taxi drivers do based on historical data.

A. Solution Outline

The service region is subject to incident customer requests located at points $\mathbf{q} \in Q$ and is patrolled by N taxis whose task is to service these requests in a manner that will minimize the distance driven to every request. Requests arrive at a rate λ, representing the sensory function $\phi(\mathbf{q})$. Each patrol loop is defined by a fixed number of waypoints whose positions are using historical customer request distributions. Our patrolling policy is adaptive in time and benefits from a finer discretization time period. In this work we use 15-minute time periods to compute 96 patrol loops for simulations over a 24-hour period (Figures 1b, 1c).

The simplest scheduling and path planning protocols are used to ensure that performance is attributable to the patrolling policy only. We emphasize that neither of these elements are crucial to the operation of the patrolling policy.

Algorithm 1 Informative Path Controller Pseudocode

Parameters: arrival distribution Z_α, patrol loop ℓ, waypoints $W^\ell = \{w_1, w_2, \ldots, w_n\}$ located at $P^\ell = \{p_1, p_2, \ldots, p_n\}$, vector of taxis $S^\ell = \{s_1, s_2, \ldots, s_m\}$

loop
  while $\left(\partial H/\partial P^\ell > 0\right)$ **do**
    Compute Voronoi partition V_w^ℓ from Z_α
    Compute centroid C_w^ℓ by integrating over V_w^ℓ
    Compute neighbor waypoint locations p_{w-1}^ℓ, p_{w+1}^ℓ
    Compute control input u_w according to (4)
    Update waypoint position p_w^ℓ according to $\dot{p}_w^\ell = u_w^\ell$
  end while
  Check for incoming requests $R = \{r_1, r_2, \ldots, r_k\}$
  Assign requests $R^\ell \subseteq R$ within $\cup V_w^\ell$ to ℓ
  Assign nearest taxis $S^* \subseteq S^\ell$ to requests R^ℓ
  Rebalance remaining $S^\ell \setminus S^*$ taxis s.t. $\sum_{i=1}^n \bar{\Phi}(w_i) = 0$
end loop

In Section V-A we present control experiments that evaluate our policy against another with identical scheduling and path planning, but using greedy redistribution.

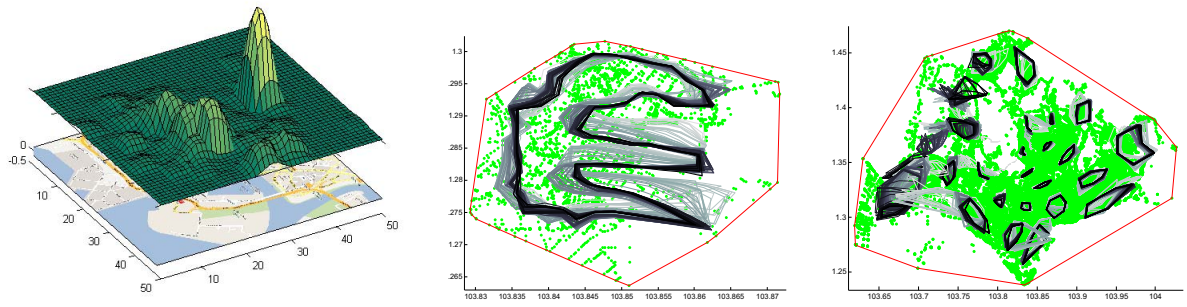
B. Algorithm Description

Algorithm 1 contains the pseudocode for the informative path patrolling policy.

The first stage of the patrolling algorithm describes how the patrol loop waypoints reposition themselves in locally optimal locations. The algorithm calculates the Voronoi region for each waypoint, computes the centroid of the region based on Z_α, and subsequently repositions the waypoints based on (4). This algorithm can be implemented in a distributed way such that it can be computed for each waypoint independently, only sharing information with their neighboring waypoints and enough information for all waypoints to compute their Voronoi regions.

Once the waypoints have converged to a locally optimum configuration, the second stage of the algorithm assigns requests to taxis and rebalances taxis within each patrol loop. Each taxi is initially assigned a home patrol loop in round robin manner. Taxis cycle through through four successive modes of operation: FREE, ONCALL, POB ("passenger on board") and RETURN. The service model assumes a dispatch center that controls all incoming requests. Scheduling is performed by matching incoming requests with the nearest available taxis. Once assigned a request, the taxi picks up the customer (ONCALL), delivers them to their destination (POB), and returns to its home loop (RETURN).

Taxis flagged FREE use an intra-loop redistribution policy that is analogous to flow equilibrium at waypoints. Taxis patrol around their home loop until such time that every waypoint is serviced by a taxi. Thereafter taxis remain stationed at their waypoints (treating them as virtual taxi stands), with any remainder of taxis (modulo the number of waypoints) continuing to patrol around the loop. This ensures that all waypoints receive equal service in steady state, while also ensuring that taxis do not waste fuel unnecessarily.



(a) Arrival distribution Z_α (b) CBD loops (c) Singapore loops

Fig. 1: Figure 1a shows a surface plot of an example arrival distribution Z_α for the CBD, overlayed on a map of the region; the destination distribution Z_δ has a similar format. Figures 1b-1c show the temporal progression of the patrol loops, overlayed on longitude/latitude plots of the GPS coordinates that form the service region. Patrol loops are shown changing dynamically in 15-minute time periods throughout the day (for a total of 96 iterations for each loop), with a darker shade indicating the most recent configuration.

Assuming we have N_ℓ taxis and n waypoints along loop ℓ, we require that the net flux (rate of inflow and outflow) Φ of taxis at waypoints converge to zero over the entire loop, i.e. $\sum_{i=1}^{n} \bar{\Phi}_i = 0$. The following three scenarios serve as the basis for all possible taxi dynamics along loop ℓ:

1) $N_\ell < n$: each taxi continuously patrols along the loop.
2) $N_\ell = n$: taxis redistribute to the nearest waypoint until there are N_ℓ/n taxis at each waypoint, and remain stationary queueing for a customer request.
3) $N_\ell > n$: $N_\ell - (N_\ell \bmod n)$ taxis queue at their respective waypoints waiting for a customer, while the remaining $N_\ell \bmod n$ taxis continue to patrol around the loop ensuring that $\sum_{i=1}^{n} \bar{\Phi}_i = 0$.

Our service policy ensures that each patrolling loop is in a locally optimal configuration within the environment while bounding the number of outstanding customer requests. In this work we do not allow taxis to exchange home loops, as this could lead to scenarios whereby a patrolling loop loses all of its taxis to neighboring loops. We solve this by ensuring that requests are assigned to the loop whose Voronoi region they originate in.

IV. Data

We use data collected by a fleet of 16,000 taxis in Singapore. The dataset is one month (August 2010) of trips, consisting of millions data points at thousands of GPS locations. Each entry records time, location, ID, etc., as well as the status (FREE, ONCALL, POB, etc.). The data serves several purposes. First, we use a subset of the data to train our dynamic patrolling algorithm. Second, we use two subsets of the dataset as test data for conducting simulations. We use the same day (Monday, August 16) for real-time data simulations which do not require training and the same day of the following week (Monday, August 23) for unseen data simulations. Finally, we use the data to quantify ground truth redistribution of taxis in Singapore. Since the actual taxi operation is unmanaged, there is no direct comparison against an existing policy. Instead, we analyze the distribution of the fleet throughout the day and record statistics such as odometry, status of operation, etc. that can be used as quantifiable metrics in our analysis.

A. Arrival and Destination Distributions

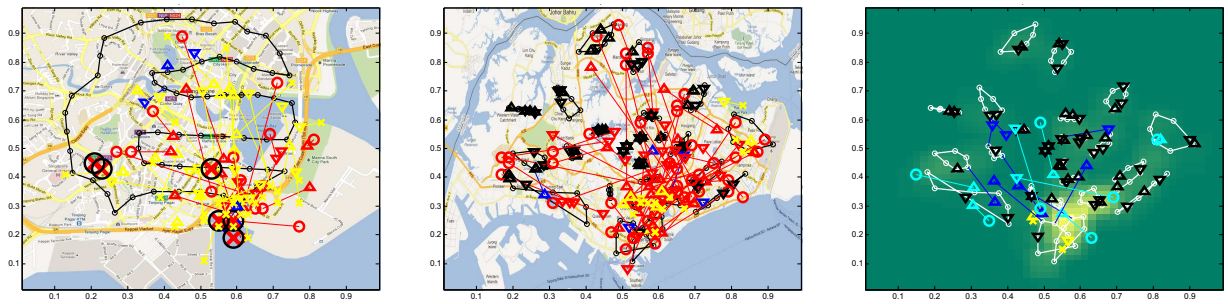
Training the policy and conducting simulations both require knowledge of customer arrivals and destinations. Historical data is used to compute spatial arrival and destination *distribution surfaces*, denoted by Z_α and Z_δ respectively. The region is discretized into a 50×50 grid, with the height of the surface at each location representing the probability of either a customer arrival (α) or request destination (δ). We use a 15-minute discretization to construct the surfaces. Figure 1a shows an example of the arrival surface Z_α.).

Data sparsity is almost always an issue in statistical modeling. Considering that the one month dataset spans a $50 \times 50 \times 96 \times 31$ space, we see that even a large amount of data will be very sparse. Two stages of smoothing were used to improve performance of our model. First, each 24-hour dataset was smoothed temporally using a simple averaging filter to reduce noise caused by temporal discretization. The resulting surfaces for each time window were then normalized and smoothed using a Gaussian filter to reduce noise caused by the 50×50 spatial discretization.

V. Experiments

A simulation framework was implemented in MATLAB. The model implements the spatial PDP formulation presented in Section II. Customer requests arrive in a Poisson process with rate parameter $\lambda(t)$ and are distributed according to $Z_\alpha(t)$. Taxis traverse the space in straight lines and at constant speed. As the simulation evolves, customers are serviced by $N(t)$ taxis that respond to the incoming pickup requests. Customer destinations are distributed according to the destination distribution surface $Z_\delta(t)$. Figure 2 shows annotated screenshots of a typical simulation.

Our simulation engine can incorporate any path planning mechanism that maps the locomotion of the taxi onto the road network. This additional complexity was omitted in this work in order to evaluate the effect of the informative



(a) CBD simulation (b) Singapore simulation (c) Singapore simulation (overlayed on Z_α)

Fig. 2: Figures 2a-2c show screenshots of the simulator in action. Taxis are indicated by a colored triangle. A taxi can be in one of four states: traveling along patrol loop (FREE, black), servicing a pickup request (ONCALL, yellow), driving a passenger to their destination (POB, red or cyan), returning to the patrol loop (RETURN, blue). Pending requests are shown with a yellow $\otimes$, and outstanding requests are shown with a circled red $\otimes$.

path policy in isolation. Since the recorded data from the Singapore taxi fleet use the underlying road network, there is clearly a cost associated with assuming a straight line path planner. We evaluate this cost by leveraging Google's extensive geocoding API. For a given pair of coordinates p_1 and p_2 we can calculate the driving distance from p_1 to p_2, taking into account the time, day, road conditions and road directionality. Generally, distances recorded in simulation incur a cost function $\beta(p_1,p_2,t) = \|p_1,p_2\|/d(p_1,p_2,t)_{drive}$ by which the recorded distance scales the true driving distance. A series of Monte Carlo simulations were carried out to approximate β under steady-state conditions by sampling points and computing the driving distance. The average value was calculated to be $\beta = 1.974$. Thus on average a taxi would drive approximatley twice the distance quoted by the simulation if it had made the same journey along the Singapore road network.

Three different types of simulation were conducted: (A) greedy policy simulations establish the benefit of the patrolling policy as a benchmark against a simplistic redistribution strategy, (B) ground truth simulations, which use the same number of taxis N that were recorded for a corresponding scenario from historical data, and (C) stability-based simulations, which aim to find the minimum number of taxis N_{min} that ensure a k-tight stability guarantee defined by (5). A stability margin of $k = 0.01$ (i.e. 1 percent) was chosen for our experiments.

A. Greedy Policy Experiments

We first show the utility of our policy by comparing it against a simplistic patrolling policy implemented in the same simulation framework. This "like to like" comparison is important for any simulation-based work if the assumptions made in the simulation engine can affect the generality of the results. Further, these simulations serve as control experiments: if a greedy policy performs poorly then we can conclude that the improvement was due to the informative path redistribution (the only difference between the two policies) and not due to path-planning or scheduling simplifications, which are trivial by comparison. Greedy policy simulations were carried out for the CBD, and for the whole of Singapore. Different numbers of loops (5,10,15,20,25) were used for multi-loop experiments as a benchmark for the best number of loops to use in the main experiments.

B. Single Loop Experiments

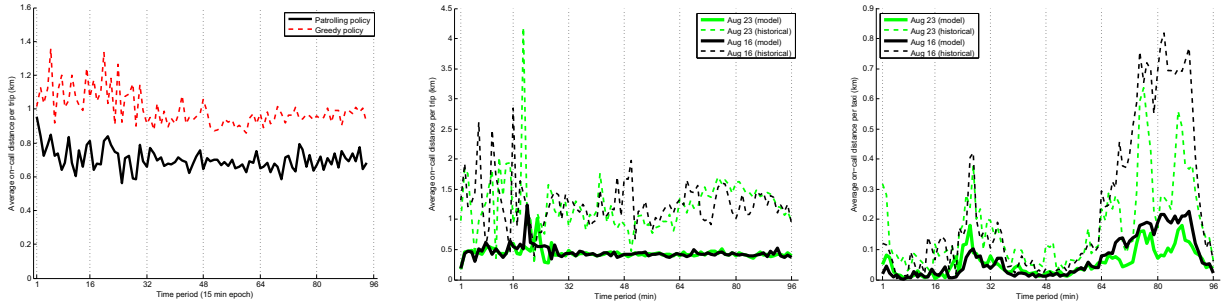
The CBD was chosen for the single-loop experiments because (1) it has a high volume of customer requests throughout the day and thus presents a lot of scope for optimization, (2) the CBD is representative of Singapore as a whole in terms of request arrival and destination flow throughout the day, and (3) an area the size of the CBD is an appropriate service region to consider for a single loop.

Both ground truth and stability-based simulations were carried out for the CBD. We use a 15-minute discretization epoch both for updating the patrol loops, and for determining the corresponding number of taxis N and arrival rate λ. Due to a fine discretization and a relatively small arrival rate, we add smoothing to reduce noise in the results.

C. Multi-Loop Experiments

Our multi-loop experiments scale up to consider the whole of Singapore. We use a larger time discretization for arrival rate λ and for the number of taxis in ground truth simulations. This gives us 6 four-hour epochs to consider throughout the course of the day, which are representative of different time periods throughout a typical day (night, early morning, morning, afternoon, evening, late evening). We maintain the 15-minute discretization for updating the patrol loops to ensure that the loops can adapt to changing arrival rate (also discretized into 15-minute time steps).

Both ground truth and stability-based simulations were carried out for the whole of Singapore. Based on preliminary results we consider 25 loops as the optimal choice for our experiments. There are 28 postal districts in Singapore, so the 25-loop case gives us an approximation for scaling up a single-loop policy in the case of the CBD (which covers around 1-2 postal regions).

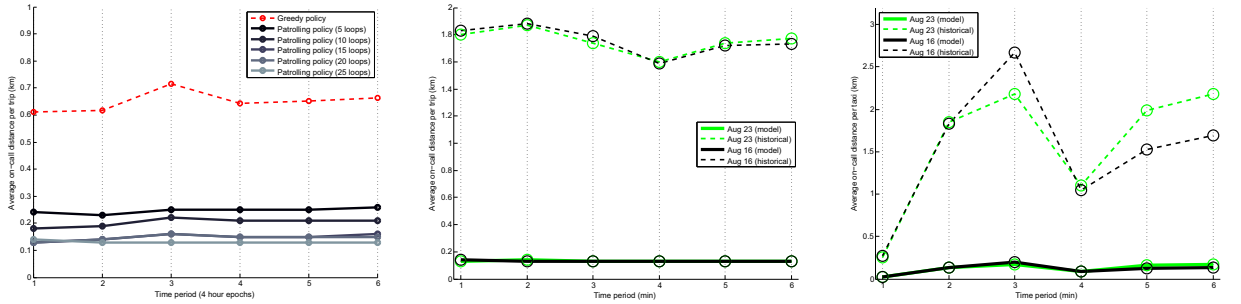


(a) CBD average on-call distance per trip (patrolling policy vs greedy policy)

(b) CBD average on-call distance per trip (patrolling policy vs historical data)

(c) CBD average on-call distance per taxi (patrolling policy vs historical data)

Fig. 3: Single-loop (CBD) simulation results



(a) Singapore average on-call distance per trip (patrolling policy vs greedy policy)

(b) Singapore average on-call distance per trip (patrolling policy vs historical data)

(c) Singapore average on-call distance per taxi (patrolling policy vs historical data)

Fig. 4: Multi-loop (Singapore-wide) simulation results

D. Unseen Test Data

In previous work [1] we demonstrate how to accurately infer traffic volume from historical data collected on different days. To determine if our policy is still useful in the absence of real-time data we conduct simulations using unseen historical test data. Our algorithm was first trained by pre-constructing dynamic patrolling loops using historical data from the same day (August 16). Experiments were then carried out using historical data from the same day in the following week (August 23). A full replication of all the preceding experiments was conducted for both the CBD and Singapore-wide multi-loop scenarios.

VI. Results

First, we consider the implications of the greedy policy simulations. Figures 3a and 4a show the greedy simulation results. In the case of the CBD we observe an overall increase in ONCALL distance per trip by a factor of 1.42. For the entire region of Singapore, the greedy policy performs even worse, increasing the overall ONCALL distance by a factor of 3.71. This supports our intuition that our policy is useful: since the control experiments employing a simple policy performed much worse using the same simulation engine, the increase in performance is due to patrolling policy only.

We are interested in examining solutions from three different points of view: customer, taxi driver and urban planning.

1) Customer (quality of service): Quality of service is represented by customer waiting time, which is equivalent to the ONCALL distance of individual taxi trips. Figure 3b shows the average ONCALL per trip distance for our single-loop policy in the CBD. The average ONCALL per trip distance for August 16 is 0.45km, as compared to 1.2km from the historical data. Thus our single-loop policy reduces the total customer waiting time by a factor of 2.66. Figure 4b shows the ONCALL per trip distance for the multi-loop case. The total average ONCALL per trip distance computed using our model is 0.13 km as compared to 1.7 km from the historical data. With a distance cost factor $\beta \approx 2$, we see that our 25-loop patrolling policy in Singapore reduces the total customer waiting time by a factor of 6.84.

2) Taxi Driver (distance traveled empty): We assume that the goal of the taxi driver is to minimize the amount of time driving empty. Figure 3c shows the ONCALL per taxi distance for our single-loop policy in the CBD. The average total ONCALL per taxi distance for August 16 is 0.06 km, as compared to 0.18 km from the historical data. Our single-loop policy reduces the average total distance driven empty by a factor of 3. Figure 4c shows the ONCALL per taxi distance for the multi-loop case. The total average ONCALL per taxi distance computed using our model is 0.12 km as compared to 1.6 km from the historical data. With a distance cost factor $\beta \approx 2$, we see that our 25-loop patrolling policy

in Singapore reduces the average total distance driven empty by a factor of 6.74.

3) Urban Planning (reducing congestion): We assume that the goal of the municipal authority is to reduce congestion by reducing the number of taxis on the road. The minimum number of taxis N_{min} that are necessary to maintain stability is given by (5) for some stability margin k. For any number of taxis $N > N_{min}$ we define the utilization factor as

$$\eta = N_{min}/N. \tag{6}$$

The utilization factor η is the fraction of those N taxis that can service all requests while maintaining stability for a given λ. The total utilization factor on August 16th in the CBD using our model is $\eta = 0.05$, thus our model requires only 5 percent of the total taxis available throughout the day to maintain stability. The total utilization factor for the multi-loop case is $\eta = 0.14$, similarly implying that our the taxi network is over-utilized by an order of magnitude.

	CBD		Singapore	
	Aug. 16	Aug. 23	Aug. 16	Aug. 23
On-call per taxi (*km*)	0.06	0.06	0.12	0.13
On-call per trip (*km*)	0.45	0.44	0.13	0.13
Utilization factor η	0.05	0.07	0.14	0.14

TABLE I: Total patrolling policy ONCALL distance ratios and utilization factor over 24-hour simulations.

A. Unseen Test Data Results

All of the preceding experiments were conducted on unseen test data, as described in Section V-D. By evaluating it on previously unseen test data from August 23, we see that our model performs well and maintains its robustness in the absence of real-time data. Figures 3 and 4 show the corresponding results for August 23 overlayed in green. We see that the ONCALL distance results maintain nearly the same magnitude and provide the same caliber of improvement over the historical data as described in Section V-D. We conclude that our policy results in a comparable improvement in performance in the absence of real-time data.

VII. Conclusions

In this paper we presented a novel patrolling policy for a fleet of service vehicles responding to requests in a PDP scenario. Our policy uses patrol loops based on informative path planning to minimize the distance driven by the vehicles to an incident request. We formalized the notion of stability in our problem context, and proved guarantees for our policy. We used historical data from a fleet 16,000 taxis in Singapore to (1) infer the current ground truth behavior of the unmanaged taxi fleet, (2) to train our algorithm, and (3) to conduct simulations using both real-time and unseen test data. We evaluated the performance of our policy by evaluating customer waiting time, distance driven emtpty, and congestion. The experiments show that we can achieve substantial improvement in customer waiting time and expected distance driving empty. Further, we observe that the taxi network is over-utilized by showing that a similar level of service is possible with much fewer taxis. Finally, we show that our policy generalizes well to unseen test data, offering an improvement in performance that is on par with results from real-time simulations.

References

[1] J. Aslam, S. Lim, X. Pan, and D. Rus. City-scale traffic estimation from a roving sensor network. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, pages 141–154. ACM, 2012.

[2] G. Berbeglia, J.F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, 2010.

[3] C.T. Cunningham and R.S. Roberts. An adaptive path planning algorithm for cooperating unmanned air vehicles. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 3981–3986. IEEE, 2001.

[4] T.L. Friesz, J. Luque, R.L. Tobin, and B.W. Wie. Dynamic network traffic assignment considered as a continuous time optimal control problem. *Operations Research*, pages 893–901, 1989.

[5] R. Graham and J. Cortés. Adaptive information collection by robotic sensor networks for spatial estimation. *Automatic Control, IEEE Transactions on*, 57(6):1404–1419, 2012.

[6] G. Hollinger and S. Singh. Multi-robot coordination with periodic connectivity. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 4457–4462. IEEE, 2010.

[7] S. Li. *Multi-attribute taxi logistics optimization*. PhD thesis, Massachusetts Institute of Technology, 2006.

[8] S. Lim, H. Balakrishnan, D. Gifford, S. Madden, and D. Rus. Stochastic motion planning and applications to traffic. *The International Journal of Robotics Research*, 30(6):699–712, 2011.

[9] D.K. Merchant and G.L. Nemhauser. Optimality conditions for a dynamic traffic assignment model. *Transportation Science*, 12(3):200–207, 1978.

[10] W.J. Mitchell, C.E. Borroni-Bird, and L.D. Burns. Reinventing the automobile. *Personal Urban Mobility for the 21st Century. Cambridge/I\/IA*, 2010.

[11] S.N. Parragh, K.F. Doerner, and R.F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(2):81–117, 2008.

[12] M. Pavone, S.L. Smith, E. Frazzoli, and D. Rus. Load balancing for mobility-on-demand systems. *Robotics: Science and Systems, Los Angeles, CA*, 2011.

[13] M. Pavone, S.L. Smith, E. Frazzoli, and D. Rus. Robotic load balancing for mobility-on-demand systems. 2011.

[14] M. Pavone, S.L. Smith, E. Frazzoli, and D. Rus. Robotic load balancing for mobility-on-demand systems. *The International Journal of Robotics Research*, 31(7):839–854, 2012.

[15] S. Peeta and A.K. Ziliaskopoulos. Foundations of dynamic traffic assignment: The past, the present and the future. *Networks and Spatial Economics*, 1(3):233–265, 2001.

[16] T. Roughgarden and É. Tardos. How bad is selfish routing? *Journal of the ACM (JACM)*, 49(2):236–259, 2002.

[17] M. Schwager, D. Rus, and J.J. Slotine. Decentralized, adaptive coverage control for networked robots. *The International Journal of Robotics Research*, 28(3):357–375, 2009.

[18] A. Singh, A. Krause, and W.J. Guestrin, C.and Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34(2):707, 2009.

[19] S. L. Smith, M. Schwager, and D. Rus. Persistent robotic tasks: Monitoring and sweeping in changing environments. *Robotics, IEEE Transactions on*, 28(2):410–426, 2012.

[20] D.E. Soltero, M. Schwager, and D. Rus. Generating informative paths for persistent sensing in unknown environments. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2172–2179. IEEE, 2012.

[21] A. Stentz. Optimal and efficient path planning for partially-known environments. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 3310–3317. IEEE, 1994.

[22] M. Volkov, J. Aslam, and D. Rus. Markov-based redistribution policy model for future urban mobility networks. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 1906–1911. IEEE, 2012.

[23] J.G. Wardrop. Some theoretical aspects of road traffic research. In *Inst Civil Engineers Proc London/UK/*, number 0, 1900.