2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Balancing Workloads of Service Vehicles over a Geographic Territory

John Gunnar Carlsson[1], Erik Carlsson[2] and Raghuveer Devulapalli[1]

***Abstract*— Autonomous vehicles (or drones) are very frequently used for servicing a geographic region in numerous applications. Given a geographic territory and a set of n fixed vehicle depots, we consider the problem of designing service districts so as to balance the workload of a collection of vehicles which service this region. We assume that the territory is a connected polygonal region, i.e. a simply connected polygon containing a set of simply connected obstacles. We give a fast algorithm, based on an infinite-dimensional optimization formulation, that divides the territory into compact, connected sub-regions, each of which contains a vehicle depot, such that all regions have equal area. We also show how we can use this algorithm to find better locations of the vehicle depots.**

I. Introduction

Efficiently dividing a workload among a collection of service vehicles, facilities, or agents is a common objective in many disciplines. The most natural objective is to distribute workloads in such a way as to minimize the total amount of work done by all agents. In practice, it is also often desirable to find a *balanced* assignment so as not to overload any particular group and to ensure uniform service levels. Such assignment policies are commonly encountered in robotics [20], [28], queueing theory [4], [19], [21], vehicle routing [12], [18], [29], and facility location [2], [5], [7], [13], among others.

The primary application we are interested in is balancing workloads of unmanned aerial vehicles (or drones) in a service region. Drones are extensively used for aerial surveillance, remote sensing, transportation, combat and scientific purposes. Although they are predominantly used in military applications, with the *FAA Modernization and Reform Act of 2012* [27], there is a rapid demand to integrate drones into civil airspace. With such increased usage, the task of efficiently allocating workload to these vehicles becomes critical. Assigning service districts by partitioning a given region into sub-regions is a natural and obvious strategy for assigning workloads to autonomous vehicles.

With this as our motivation, we consider the problem of dividing a given geographic territory among a set of vehicles in such a way as to minimize the total workload imposed on them while simultaneously ensuring that all vehicles service the same amount of territory. More precisely, we are given a planar region R and a set of n fixed landmark points ("vehicle depots") $P = \{p_1, \ldots, p_n\}$, and our objective is to design sub-regions $\{R_1, \ldots, R_n\}$ of equal area that minimize the workloads of the vehicles p_i in providing service to their respective sub-regions R_i. The set $\{R_1, \ldots, R_n\}$ should obviously be a *partition* of R, that is, we should require that $\bigcup_i R_i = R$ and that the interiors do not overlap, $\text{int}(R_i) \cap \text{int}(R_j) = \emptyset$ for all distinct pairs i, j. The input region R over which the drones travel could be complicated. It need not always be a convex region; we will consider a more general case in which the region R contains a collection of impenetrable obstacles, such as tall buildings or mountains. To incorporate a realistic ground environment, we consider R to be represented by a simple polygon that contains a set of *obstacles* $\{O_1, \ldots, O_m\}$ (also simple polygons) as shown in Figure 1a.

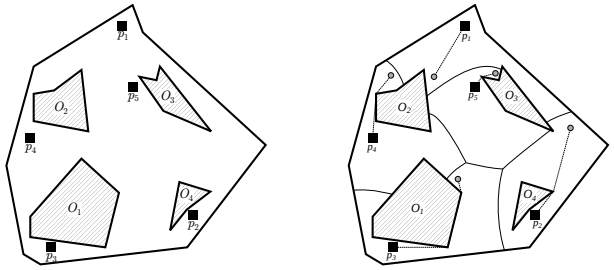


Fig. 1: Inputs (1a) and outputs (1b) to our partitioning problem; we are given a region R containing $m = 4$ obstacles and $n = 5$ points p_i, and we design $n = 5$ sub-regions of equal area corresponding to the vehicles that minimize the workloads as defined in (1). It turns out that our partition is *relatively star-convex* to R (1b).

To model the workload of vehicle i in providing service to region R_i, we find it useful to introduce what we call the *Fermat-Weber* function $\text{FW}(R_i, p_i)$, defined as

$$\text{FW}(R_i, p_i) := \iint_{R_i} d(x, p_i)\, dx \tag{1}$$

where $d(\cdot, \cdot)$ is the length of the shortest path between a pair of points when obstacles are taken into account. A vehicle i travels a distance of $d(x, p_i)$ to service a point x. In practice, drones have less fuel holding capacity or short lived batteries and hence they are often required to return to their depots periodically for re-fueling or updating information. The function $\text{FW}(R_i, p_i)$ thus defines the workload of the i^{th} vehicle and it is simply proportional to the average distance between p_i and a point uniformly randomly sampled in its service region R_i.

It is obvious that, by seeking to minimize the total vehicle workloads, our partition will be *compact*, in the sense that a vehicle should not be assigned to service points that are far away from it. In fact, the partitions that we produce turn out to be *connected*, so that all districts are contiguous; see Figure 1. In addition, we may require that these service

[1]Industrial and Systems Engineering, University of Minnesota.
[2]P. Simons Center for Geometry and Physics, SUNY Stony Brook.

sub-regions be computable in a *decentralized* fashion [26]: that is, the sub-region assigned to vehicle i should be computable using only "local" information to vehicle i (such as nearby neighbors or information about its surroundings), and the optimal boundary between two sub-regions should be computable using only knowledge available to those two vehicles.

Our contribution

Designing boundaries between sub-regions is an infinite dimensional problem and is usually extremely complicated. The traditional way to solve it is to break the region into small discrete "pixels" and assign binary variables to each pixel. This essentially becomes a large combinatorial problem which is computationally intractable for large problem instances. In this paper, we show how we transform our partitioning problem into a simple n-dimensional convex optimization problem that determines the optimal partition boundaries using a sequence of *cutting planes*. Our algorithm can be readily applied to any planar connected region that admits a representation via poly-lines. Our algorithm also has the advantage of *decentralization*, where each vehicle depot can compute its sub-region by obeying a simple control law and communicating with its nearby neighboring depot points. A side consequence of our analysis is a collection of simple and immediate proofs of established results in equitable partitioning [2], [3], [28] and an affirmative answer to a question posed in the concluding remarks of [1], [2].

We have previously considered the problem of dividing a region into smaller pieces when the input region R is convex or simply connected. Specifically, in [11], [12], we give a fast algorithm that takes as input a *convex* polygon C and a point set $P = \{p_1, \ldots, p_n\}$ and divides C into n sub-regions $\{C_1, \ldots, C_n\}$ such that each C_i is convex, each C_i contains a point, and all sub-regions have equal area. In [10] we extend this algorithm to the case where the input region is a simply connected polygon S (i.e. a connected polygon and devoid of any obstacles) and we have an arbitrary probability density $f(\cdot)$ defined on S. Rather than producing equal-area sub-regions $\{S_1, \ldots, S_n\}$, our objective is to find a partition such that $\iint_{S_i} f(x)\,dA$ is equal for all sub-regions (the previous paper is thus a special case of this problem in which $f(\cdot)$ is a uniform distribution). Since it is not always possible to divide S into *convex* pieces in this case, our algorithm divides S into sub-regions that are *relatively convex* to S: for any two points x, y contained in a sub-region S_i, the shortest path in S from x to y (which may not be a straight line segment) is itself contained in S_i (see Figures 2, 3, and 12 of [10] for examples). This is clearly a natural generalization of convexity, because when S happens to be convex, the shortest path from x to y in S is indeed a straight line segment, and therefore all sub-regions S_i will be convex.

The algorithm of this paper presents two clear advantages over our previous methods: first, it is easy to show that when the input region R contains obstacles (and is thus no longer simply connected), an equitable relatively convex partition of the type just described may not exist (see Figure 1 of [6]), and thus our prior work is of little utility. Second, the algorithms of [11], [12] and [10] do not force regions to be *compact*: it is possible for the output regions to be extremely long and skinny, which is clearly undesirable from a practical standpoint (see Figure 10 of [11] and Figure 14 of [10]).

Here we tackle these two problems in the following way: first, as we will show, our algorithm produces sub-regions $\{R_1, \ldots, R_n\}$ that are *relatively star-convex* to R: for any point x in sub-region R_i, the shortest path in R from x to p_i is itself contained in R_i (see Figure 1b). This is a weaker condition than relative convexity but still offers considerable practical utility; for example, it automatically produces connected sub-regions. Second, we are able to explicitly enforce our sub-regions to be compact; in fact, one interpretation of our problem is that we are trying to find the partition that is as compact as possible; we will make this explicit in Section II. The algorithm in this paper is also novel in that our prior work generally uses well-established principles of discrete geometry such as ham sandwich cuts and binary and ternary space partitions, whereas here we use techniques from vector space optimization and linear programming complementarity.

Related work

In addition to our own previous work that we have already mentioned, the general problem of equitably partitioning a region has been studied from many perspectives. They are of particular interest to the robotics community. For example, the paper [28] considers the problem of dividing a convex region into convex equal-area sub-regions using power diagrams. The authors give a decentralized control policy that provably converges to the desired partition. More recently, [15] considers a hybrid problem in which the objective is to simultaneously place the facilities P and design coverage regions associated with the facilities using only asynchronous (and possibly noisy) pairwise communication between facilities. The authors give an algorithm that provably converges to a *centroidal Voronoi partition*, that is, a Voronoi diagram $\{V_1, \ldots, V_n\}$ such that p_i is the center of mass of each sub-region V_i. Other papers making use of power diagrams as a partitioning policy include [24] and [30].

Closely related papers [1], [2] considers the problem of partitioning a *convex* region C so as to minimize the aggregate workload over all facilities while imposing an equal-area constraint. The authors describe a constant-factor approximation algorithm for dividing C into equal-area convex pieces to maximize the minimum "fatness" of any piece. This in turn gives an approximation algorithm for the problem of minimizing the aggregate workload over all facilities when facility placement is variable, as well as the sub-region boundaries. The authors also give a constant-factor approximation algorithm for solving the convex case of precisely the problem that we address here, which will be introduced in the following section. The paper [3] considers the problem of dividing a convex region into convex sub-regions according using *power diagrams*, a natural extension of *Voronoi diagrams*. The authors develop a rich and elegant theory relating such partitions to the infinite-dimensional

least-squares problem, which we will touch on in Remark 1 of this paper.

Notational conventions and technical assumptions

Our notational conventions in this paper will be as follows: in addition to the Fermat-Weber function $\mathrm{FW}(\cdot,\cdot)$ and the distance function we have introduced, we also remark that, when we refer to the input region R, we implicitly take the obstacles into account (so that $R = S \setminus \bigcup_i O_i$ for some simply connected polygon S). We will assume throughout this paper that $\mathrm{Area}(R) = 1$.

II. Problem formulation

We begin by formally stating our optimization problem for designing the service districts R_i. Suppose that $\mathrm{Area}(R) = 1$ and that the cost of service between a demand point x and depot i is simply $d(x, p_i)$, where the points p_i are fixed and given as inputs. If demand is uniformly distributed over R, the average workload on vehicle at depot i, when assigned to provide service to region R_i,is precisely

$$\mathrm{FW}(R_i, p_i) = \iint_{R_i} d(x, p_i)\, dx$$

as stated previously. It is obvious that the total workload on all vehicles is minimized when each point x is merely assigned to its nearest depot, i.e. when $\{R_1, \ldots, R_n\}$ is a Voronoi partition of P in R. In order to balance the workloads of the vehicles, we will impose the constraint that $\mathrm{Area}(R_i) = \mathrm{Area}(R)/n = 1/n$ for each i. We can thus write our optimization problem as

$$\begin{aligned} \underset{R_1,\ldots,R_n}{\text{minimize}} \sum_{i=1}^{n} \mathrm{FW}(R_i, p_i) \qquad & s.t. \qquad (2)\\ \mathrm{Area}(R_i) &= 1/n \\ \bigcup_{i=1}^{n} R_i &= R\,. \end{aligned}$$

It is easy to see that the objective function of (2) forces regions to be compact because it minimizes the average distance between demand points and their assigned vehicle depots. It is also worth mentioning that (2) is a special case of the famous *Monge-Kantorovich transportation problem* [31] in the plane in which one Radon measure is the uniform distribution and the other is atomic: our objective is to "transport" the continuously distributed demand to the finite collection of facilities, while obeying capacity constraints and minimizing the aggregate transportation cost.

III. Optimal partitioning

For ease of exposition, we find it helpful to give an outline of this section:

- We first show how to formulate our problem (2) as an infinite-dimensional optimization problem (3), which we then relax to an infinite-dimensional linear program (4).
- We next take the *dual* (in the sense of linear programming) of problem (4), and show that this dual (7) can be expressed in terms of n Lagrange multipliers (Theorem 1).
- Finally we show that the optimal solution to problem (4) can be recovered via the optimal solution to (7) (Theorem 2), and that this optimal solution to (4) is also a solution to problem (3) and therefore to our original problem (2) (Theorem 3).

It is clear that we can re-write our initial formulation (2) as an infinite-dimensional optimization problem by introducing indicator function variables $I_1(\cdot), \ldots, I_n(\cdot)$ defined on R in the following fashion:

$$\begin{aligned} \underset{I_1(\cdot),\ldots,I_n(\cdot)}{\text{minimize}} \iint_R \sum_{i=1}^{n} d(x, p_i) I_i(x)\, dx \qquad & s.t. \qquad (3)\\ \iint_R I_i(x)\, dx &= 1/n \quad \forall i \\ \sum_{i=1}^{n} I_i(x) &= 1 \quad \forall x \in R \\ I_i(x) &\in \{0,1\} \quad \forall i, x\,. \end{aligned}$$

Thus, R_i consists of precisely those points x for which $I_i(x) = 1$. The above problem is, of course, an infinite-dimensional *integer* program which we expect to be difficult to solve. We can relax the integrality constraint that $I_i(x) \in \{0,1\}$ to obtain a related infinite-dimensional *linear* program:

$$\begin{aligned} \underset{I_1(\cdot),\ldots,I_n(\cdot)}{\text{minimize}} \iint_R \sum_{i=1}^{n} d(x, p_i) I_i(x)\, dx \qquad & s.t. \qquad (4)\\ \iint_R I_i(x)\, dx &= 1/n \quad \forall i \\ \sum_{i=1}^{n} I_i(x) &= 1 \quad \forall x \in R \\ I_i(x) &\geq 0 \quad \forall i, x\,. \end{aligned}$$

We can discretize the above problem into N grid cells $\square_j$ of area $\epsilon = 1/N$, where d_{ij} denotes the distance between p_i and the center of cell $\square_j$ and z_{ij} denotes the fraction of cell $\square_j$ assigned to p_i, to obtain the approximate formulation

$$\begin{aligned} \underset{Z=[z_{ij}]}{\text{minimize}} \sum_{i=1}^{n} \sum_{j=1}^{N} \epsilon d_{ij} z_{ij} \qquad & s.t. \qquad (5)\\ \sum_{j=1}^{N} \epsilon z_{ij} &= 1/n \quad \forall i \\ \sum_{i=1}^{n} z_{ij} &= 1 \quad \forall j \\ z_{ij} &\geq 0 \quad \forall i, j\,. \end{aligned}$$

It is a standard exercise in linear programming to verify that the dual problem to (5), which has Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^n$ and $\boldsymbol{\sigma} \in \mathbb{R}^N$, is

$$\begin{aligned} \underset{\lambda,\sigma}{\text{maximize}} \frac{1}{n} \sum_{i=1}^{n} \lambda_i + \sum_{j=1}^{N} \epsilon \sigma_j \qquad & s.t. \\ \lambda_i + \sigma_j &\leq d_{ij} \quad \forall i, j\,. \end{aligned}$$

which is a discretization of the problem

$$\begin{aligned} \underset{\lambda,\sigma(\cdot)}{\text{maximize}}\ \frac{1}{n}\sum_{i=1}^{n}\lambda_i + \iint_R \sigma(x)\,dx \qquad & s.t. \qquad (6)\\ \sigma(x) \quad &\leq \quad d(x,p_i) - \lambda_i \quad \forall i, x \end{aligned}$$

which is itself equivalent to the unconstrained problem

$$\underset{\lambda}{\text{maximize}}\ \frac{1}{n}\sum_{i=1}^{n}\lambda_i + \iint_R \min_i \{d(x,p_i) - \lambda_i\}\,dx\,.$$

Finally, we note that the above problem is translation-invariant in $\boldsymbol{\lambda}$ because we have assumed that Area(R) = 1 and thus we obtain the convex, n-dimensional dual problem

$$\begin{aligned} \underset{\lambda}{\text{maximize}} \iint_R \min_i \{d(x,p_i) - \lambda_i\}\,dx \qquad & s.t. \qquad (7)\\ \sum_{i=1}^{n}\lambda_i \quad &= \quad 0 \end{aligned}$$

so that we can state our first result:

Theorem 1: The dual problem of the infinite-dimensional linear program (4) is the finite-dimensional convex problem (7).

Proof: There is nothing more to do here, except to provide some justification for the discretization that we introduced to obtain (5) from (4). This can be made entirely rigorous using established principles of vector space optimization [23]. We defer this proof to a later version of this paper due to space constraints. ∎

We will next show that the optimal solution to (4), $I_1^*(\cdot), \ldots, I_n^*(\cdot)$, can be recovered from the optimal solution to (7), $\boldsymbol{\lambda}^*$: consider any point $x \in R$ and the optimal solution $\boldsymbol{\lambda}^*$ to (7). Suppose $\bar{i}$ is the index such that $d(x,p_i) - \lambda_i^*$ is minimal (assuming such an index is unique). From basic linear programming theory, we know that the *complementary slackness* conditions of problem (6) stipulate that $I_i^*(x) = 0$ for all indices i other than $\bar{i}$, and consequently that $I_{\bar{i}}^*(x) = 1$. From this we have proven our second result:

Theorem 2: The optimal solution $I_1^*(\cdot), \ldots, I_n^*(\cdot)$ to the infinite-dimensional linear program (4) must satisfy

$$I_i^*(x) = \begin{cases} 0 & \text{if } d(x,p_i) - \lambda_i^* > d(x,p_j) - \lambda_j^* \text{ for some } j \\ 1 & \text{if } d(x,p_i) - \lambda_i^* < d(x,p_j) - \lambda_j^* \text{ for all } j \neq i \end{cases}$$

where $\boldsymbol{\lambda}^*$ is the optimal solution to (7). If neither of the two cases above holds at a point x, then if $I_i^*(x) > 0$ at a point x, it must be the case that $d(x,p_i) - \lambda_i^* \leq d(x,p_j) - \lambda_j^*$ for all j, i.e. the index i is among the minimal indices. In other words, the optimal solution to 3 is an *additively weighted Voronoi diagram*.

Note that Theorem 2 tells us that the optimal solution to the linear relaxation (4) has the useful property that $I_i^*(x) \in \{0,1\}$, except possibly for those points x such that $d(x,p_i) - \lambda_i^* = d(x,p_j) - \lambda_j^*$ for two indices i, j. We will next show that there actually exists an optimal solution $I_1^*(\cdot), \ldots, I_n^*(\cdot)$ to the linear relaxation (4) in which $I_i^*(x) \in \{0,1\}$ for *all* $x \in R$. This must, therefore, also be a solution to the problem (3) and therefore to our original partitioning problem (2).

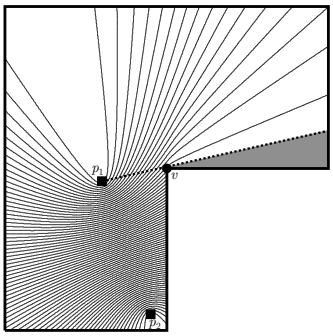


Fig. 2: As λ_1 and λ_2 vary, the induced sub-regions change as indicated above. The points x in the shaded region do not have a unique minimal index $i_{\min}$ because $\lambda_1 - \lambda_2 = d(v,p_1) - d(v,p_2)$. The point v is therefore the *reflex vertex* associated with the shaded region.

Theorem 3: There exists an optimal solution $I_1^*(\cdot), \ldots, I_n^*(\cdot)$ to the infinite-dimensional linear program (4) such that $I_i^*(x) \in \{0,1\}$ for all $x \in R$, which is therefore also an optimal solution to (3). Thus, the optimal solution to problem (2) is an additively weighted Voronoi partition of R with respect to P.

Proof: Given the optimal Lagrange muliplier vector $\boldsymbol{\lambda}^*$ from (7), we define the "strict dominance regions" $R_i^+ \subseteq R_i^*$ as

$$R_i^+ = \left\{x \in R : d(x,p_i) - \lambda_i^* < d(x,p_j) - \lambda_j^* \text{ for all } j \neq i\right\}\,.$$

It is easy to see by construction that each R_i^+ is *relatively star-convex* to R: if $x \in R_i^+$, the shortest path (or paths) in R from x to p_i is contained in R_i^+. From basic Euclidean geometry it is also clear that the boundary between two strict dominance regions R_i^+ and R_j^+ consists of a collection of hyperbolic arcs, since a hyperbola is the locus of points where the absolute value of the difference of the distances to two foci is constant (if we measure point-to-point distances using the ℓ_1 or ℓ_∞ norms, while still taking obstacles into account, the boundary between two strict dominance regions consists instead of a collection of line segments). Given $\boldsymbol{\lambda}^*$, we can construct these arcs for all R_i^+ efficiently using the *continuous Dijkstra paradigm* [25]. The question remains of how to allocate the remaining area of R that does not lie in a strict dominance region (see Figure 2).

It is not hard to show that each such region is polygonal (as opposed to being bounded by hyperbolic arcs), and indeed, determining a proper assignment is a fairly straightforward procedure. Since each such "ambiguous dominance region" is potentially associated with more than one point p_i, we associate with each such region R_k^- (not indexing by i because these regions are not yet associated with individual facilities) an index set $\mathcal{I}_k \subseteq \{1, \ldots, n\}$. By construction, each R_k^- must have a *reflex vertex* r_k, that is, a point in R_k^- such that $d(x,p_i) = d(x,r_k) + d(r_k,p_i)$ for all $x \in R_k^-$ and $i \in \mathcal{I}_k$; an example of this is the point marked v in Figure 2. Consider

the functions $I_i^*(\cdot)$ (which we have not yet defined on R_k^-) for $i \in \mathcal{I}_k$. The total cost due to R_k^- is

$$\begin{aligned}
&\iint_{R_k^-} \sum_{i\in\mathcal{I}_k} d(x,p_i) I_i^*(x)\, dx \\
= &\iint_{R_k^-} \sum_{i\in\mathcal{I}_k} (d(x,r_k) + d(r_k,p_i)) I_i^*(x)\, dx \\
= &\iint_{R_k^-} d(x,r_k) \underbrace{\left(\sum_{i\in\mathcal{I}_k} I_i^*(x)\right)}_{=1} dx + \iint_{R_k^-} \sum_{i\in\mathcal{I}_k} d(r_k,p_i) I_i^*(x)\, dx \\
= &\underbrace{\iint_{R_k^-} d(x,r_k)\, dx}_{\text{constant}} + \sum_{i\in\mathcal{I}_k} d(r_k,p_i) \iint_{R_k^-} I_i^*(x)\, dx
\end{aligned}$$

which we observe only depends on $\iint_{R_k^-} \sum_{i\in\mathcal{I}_k} I_i^*(x)\, dx$, that is, the *areas* of region R_k^- assigned to the facilities $i \in \mathcal{I}_k$, as opposed to the particular assignment patterns $I_i^*(x)$. Thus, in order to assign these amounts to the facilities, we can simply construct a matrix A such that

$$a_{ik} = \begin{cases} 1 & \text{if } i \in \mathcal{I}_k \\ 0 & \text{otherwise} \end{cases}$$

and then consider the problem of constructing a matrix Z that gives the areas assigned to each facility from each R_k^-, i.e. satisfying the constraints

$$\begin{aligned}
\sum_i a_{ik} z_{ik} &= \iint_{R_k^-} dx \quad \forall k \qquad (8)\\
\sum_k a_{ik} z_{ik} &= c_i - \iint_{R_i^+} dx \quad \forall i \\
z_{ik} &\geq 0 \quad \forall i,k\,.
\end{aligned}$$

It follows from the argument above that the total cost due to these allocations is then $\sum_{i,k} b_{ik} z_{ik}$, where B is a matrix such that $b_{ik} = d(r_k, p_i)$. In fact, it is not hard to show that all feasible solutions to (8) have the same objective value $\sum_{i,k} b_{ik} z_{ik}$: this is precisely because, for any ambiguous dominance region R_k^-, we must have $d(r_k,p_i) - d(r_k,p_j) = \lambda_i - \lambda_j$ for all $i,j \in \mathcal{I}_k$. Thus, the columns of B are simply the vector $\boldsymbol{\lambda}^*$, translated by a scalar, and with all indices $j \notin \mathcal{I}_k$ set to 0:

$$B = \begin{pmatrix} | & & | & & | \\ \boldsymbol{\lambda}^*_{\mathcal{I}_1} + \beta_1, & \cdots, & \boldsymbol{\lambda}^*_{\mathcal{I}_k} + \beta_k, & \cdots, & \boldsymbol{\lambda}^*_{\mathcal{I}_K} + \beta_K \\ | & & | & & | \end{pmatrix}$$

where K is the number of ambiguous dominance regions. After finding a feasible (and therefore optimal) set of allocations Z^*, we merely have to find a way to assign the areas z_{ik}^* so that the final regions R_i^* are relatively star-convex. This is trivial as shown in Figure 3. ■

Some further commentary is in order at this time:

Remark 1: The special case of Theorem 3 where $d(x,y) = \|x-y\|_2$ (i.e. where R is convex and without obstacles) was already proven in [1], [2] (whose analysis extends to our case with obstacles without incident), and the special case where $d(x,y) = \|x-y\|_2^2$ was proven in [3]. However, our proof of Theorem 3 is novel in two ways: first, it generalizes these prior results by establishing that both are simple and immediate consequences of duality theory in linear programming (since our proofs carry forward for *any* cost function $d(x,p_i)$). Second, as we will demonstrate in the following section, Theorem 3 is actually *constructive* because one can find the optimal Lagrange multipliers $\boldsymbol{\lambda}^*$ (and thereby the optimal solution to our original partitioning problem (2)) efficiently, for any cost function $d(x,p_i)$.

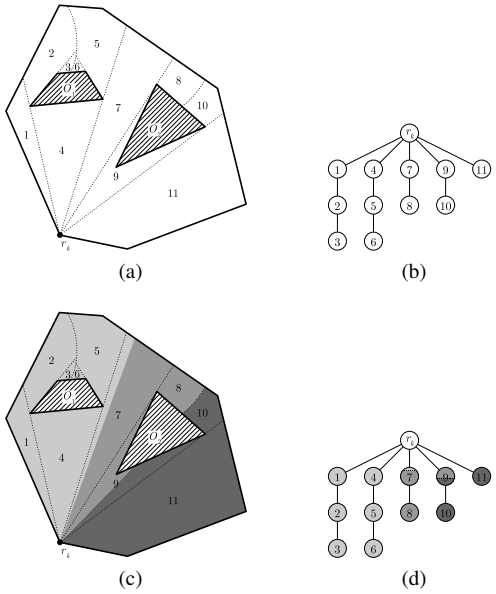


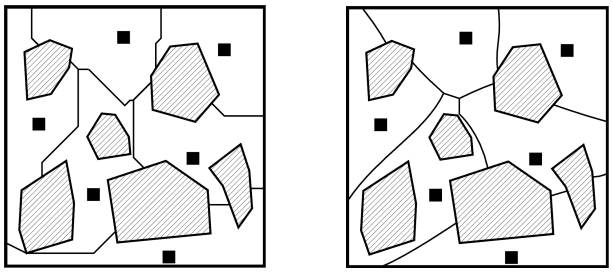
Fig. 3: Given an ambiguous dominance region R_k^- (3a) and a set of allocations Z^*, it is straightforward to divide R_k^- into pieces that are relatively star-convex by constructing a shortest-path tree (3a and 3b) and then assigning regions based on (for example) a depth-first search of the tree (3c and 3d).

Remark 2: If we desire regions with *line segments* as boundaries instead of hyperbolas (perhaps for computational efficiency), we can measure point-to-point distances using the ℓ_1 or ℓ_∞ norms (while still taking obstacles into account). Figure 4 shows the difference between partitions induced by using such a distance metric.

Remark 3: It is straightforward to verify that the optimal solution to problem (2) is still an additively weighted Voronoi diagram when we require that all sub-regions have the same mass of some *probability density* $f(\cdot)$ (instead of equal areas). Note also that we have not made explicit use of the requirement that all sub-regions have *equal* area (or mass); indeed Theorem 3 remains true for any arbitrary set of assignments of areas or masses to the sub-regions (provided, of course, that the desired areas or masses sum to Area(R)).

IV. Solving problem (7)

In the previous section, we showed that our partitioning problem can be reduced to the n-dimensional convex optimization problem (7). In this section, we describe how to



(a) (b)

Fig. 4: Equitable partitioning under the ℓ_1 norm (4a) versus the ℓ_2 norm (4b).

solve problem (7) efficiently. Our key tool in doing so is to show how to quickly construct a *supergradient* vector [8] of (7), defined as follows:

Definition 1: A *supergradient* of a function $h(\boldsymbol{\lambda})$ is a vector $\mathbf{g}$ such that $h(\boldsymbol{\lambda}') \leq h(\boldsymbol{\lambda}) + \mathbf{g}^T(\boldsymbol{\lambda}' - \boldsymbol{\lambda})$ for all $\boldsymbol{\lambda}'$.

This is useful for us because we can then solve (7) efficiently using a standard *analytic center cutting plane method* (ACCPM) to determine $\boldsymbol{\lambda}^*$ within arbitrary precision [9]. For the moment, for a given weight vector $\boldsymbol{\lambda}$, we let $R_i(\boldsymbol{\lambda})$ denote the additively weighted Voronoi cell corresponding to point p_i.

Theorem 4: The vector $\mathbf{g}$ defined by

$$g_i := \text{Area}(R_i(\boldsymbol{\lambda}))$$

is a supergradient for the concave function

$$h(\boldsymbol{\lambda}) = \iint_R \min_i \{d(x, p_i) - \lambda_i\}\, dx\,.$$

Proof: Consider two vectors $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}'$ and the corresponding additively weighted Voronoi partitions $\{R_1, \ldots, R_n\}$ and $\{R_1', \ldots, R_n'\}$ (we omit the dependence of the partitions on $\boldsymbol{\lambda}$ for brevity). We want to show that $h(\boldsymbol{\lambda}') \leq h(\boldsymbol{\lambda}) + \mathbf{g}^T(\boldsymbol{\lambda}' - \boldsymbol{\lambda})$, i.e. that

$$\iint_R \min_i\{d(x, p_i) - \lambda_i'\}\, dx \leq \iint_R \min_i\{d(x, p_i) - \lambda_i\}\, dx - \mathbf{g}^T(\boldsymbol{\lambda}' - \boldsymbol{\lambda})$$

or equivalently that

$$\sum_{i=1}^n \iint_{R_i'} d(x, p_i) - \lambda_i'\, dx \leq \sum_{i=1}^n \iint_{R_i} d(x, p_i) - \lambda_i\, dx - g_i(\lambda_i' - \lambda_i)\,.$$

Consider the right-hand side of the above; for each i, we have

$$\begin{aligned} & \iint_{R_i} d(x, p_i) - \lambda_i\, dx - g_i(\lambda_i' - \lambda_i) \\ = & \iint_{R_i} d(x, p_i) - \lambda_i\, dx - (\lambda_i' - \lambda_i) \iint_{R_i} dx \\ = & \iint_{R_i} d(x, p_i) - \lambda_i'\, dx \end{aligned}$$

and therefore we have

$$\sum_{i=1}^n \iint_{R_i} d(x, p_i) - \lambda_i'\, dx \geq \sum_{i=1}^n \iint_{R_i'} d(x, p_i) - \lambda_i'\, dx$$

since by construction, the sub-regions of the partition $\{R_1', \ldots, R_n'\}$ are defined by looking at the minimal value of $d(x, p_i) - \lambda_i'$ and are therefore minimal over all partitions. This completes the proof. ∎

We thus have in hand a fast method for computing supergradients for problem (7). We can therefore solve this problem quickly using an *analytic-center cutting plane method* (ACCPM), as given in Algorithm 1. The basic idea of this algorithm is to identify a polyhedron that is guaranteed to contain the optimal mulitplier λ^* and to subsequently reduce the size of this polyhedron at each iteration of the algorithm. The supergradient given by Theorem 4 provides a *cutting plane* which helps reduce the size of the polyhedron at every iteration step.

Input: A polygonal region R with area 1 containing a collection of obstacles $O_1, \ldots, O_m$, a collection of points $P = \{p_1, \ldots, p_n\} \subset R$, and a threshold ϵ.
Output: A partition of R into n compact relatively star-convex regions $R_1, \ldots, R_n$, such that $p_i \in R_i$ and $|\,\text{Area}(R_i) - \epsilon| \in O(\epsilon)$ for all i.
Note: this is simply a standard analytic center cutting plane method applied to problem (7).

Define the initial polyhedron by
$\Lambda = \left\{\boldsymbol{\lambda} \in \mathbb{R}^n : \mathbf{1}^T\boldsymbol{\lambda} = 0 \text{ and } \left|\lambda_i - \lambda_j\right| \leq d(p_i, p_j)\right\}$;
while $\text{vol}(\Lambda) > \epsilon$ **do**
  Let $\boldsymbol{\lambda}^0$ be the analytic center of Λ;
  Construct the strict dominance regions R_i^+ with multiplier vector $\boldsymbol{\lambda}^0$;
  Allocate the remaining area in ambiguous dominance regions lexicographically; that is, assign region R_k^- to the facility i with minimal index $i \in \mathcal{I}_k$;
  `/* This lexicographic allocation will not generally be feasible for the original partitioning problem. */`
  for $i \in \{1, \ldots, n\}$ **do**
    Set $g_i := \text{Area}(R_i)$;
  end
  Set $\Lambda := \Lambda \cap \{\boldsymbol{\lambda} : \mathbf{g}^T\boldsymbol{\lambda} \geq \mathbf{g}^T\boldsymbol{\lambda}^0\}$;
end
Let $\boldsymbol{\lambda}^0$ be the analytic center of Λ;
Construct the strict dominance regions R_i^+ with multiplier vector $\boldsymbol{\lambda}^0$;
Allocate the remaining area in ambiguous dominance regions so that $\text{Area}(R_i) = 1/n$ for all i;
return $\{R_1, \ldots, R_n\}$;

Algorithm 1: Algorithm ObstaclePartition(R, O, P) partitions a region with obstacles into compact relatively star-convex sub-regions.

Since Algorithm 1 is simply a standard analytic center cutting plane method (whose complexity is well-understood [17]) for solving the convex problem (7), it will suffice to discuss the computational complexity of each iteration. For any given vector $\boldsymbol{\lambda}$, we can construct the corresponding strict and ambiguous dominance regions (i.e. the weighted Voronoi diagram) using the *continuous Dijkstra paradigm* in $O(N^{5/3})$ steps, where N is the total number of vertices of R and the obstacles [25]. We can also compute the areas of these cells in $O(N^{5/3})$ steps and therefore each iteration requires only $O(N^{5/3})$ computations.

Remark 4: It is mentioned in the concluding remarks of [1], [2] that, for the special case where R is a convex polygon,

> "[I]t would be interesting to devise heuristics to approximate the additive-weighted Voronoi diagram that induces an optimal subdivision."

Algorithm 1 thus gives an affirmative answer to precisely this question as well as giving a general solution for non-convex regions.

V. Computational results

In this section, we present the results of numerical simulation based on a practical military application. We consider a military conflict zone which is located in a hilly region. Unmanned aerial vehicles (UAV's) are used for aerial surveillance to monitor activities in such regions. A fixed number of UAV's are available for this purpose. Our task is to strategically divide the area into sub-regions such that the total work load for all the UAV's is minimized, i.e. such that all regions have equal area, as in the second example of Section II. Typical low-cost UAV's used by the military for aerial surveillance have a *service ceiling*, or maximum usable altitude, of 5000 meters. As a sample dataset we use the troubled region of Jammu & Kashmir close to the India-Pakistan border in the Himalayan mountains, whose high altitude peaks pose an obstruction for the UAV's. Using a contour map of the Drass Sector obtained from Google Maps, we set all regions with an altitude exceeding 5000m as "obstacles" for the UAV's and built polygonal approximations for them. We choose six UAV launch sites randomly in the basin of the region since one generally prefers a flat terrain to launch such vehicles. The setup to this simulation is shown in Figure 5. In Figure 6, we show the equal-area partitions as computed by Algorithm 1. Figure 7 shows performance of the ACCPM method used in the algorithm which converges in 64 iteration steps. The figure in the left shows the normalized areas of the regions that are produced in the various iterations of our algorithm and the figure in the right shows the primal and dual objective function values.

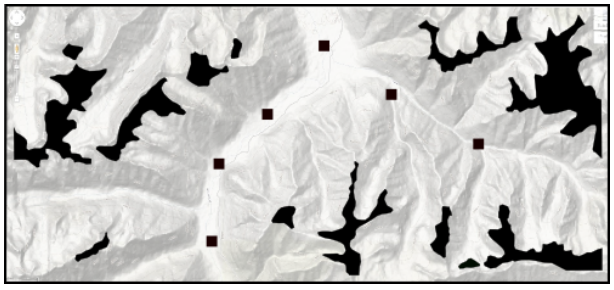


Fig. 5: Input to our simulation. The black shaded areas correspond to obstacles, i.e. those regions with an altitude exceeding 5000m.

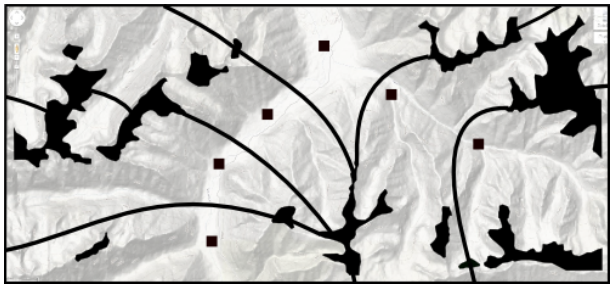


Fig. 6: Output of our simulation.

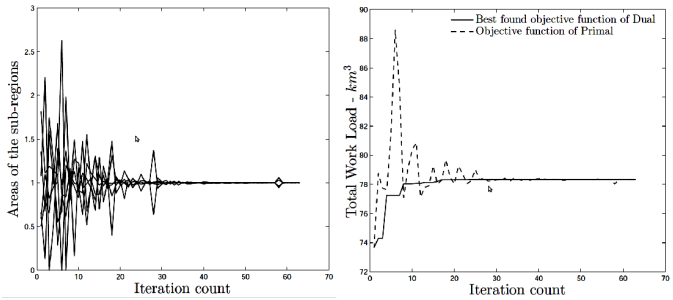


Fig. 7: Performance of ACCPM in Algorithm 1for the Drass map.

A. Simultaneous depot placement

Our objective so far was to partition the region to minimize workload for set of vehicles with fixed depots. By computing optimal depot locations, one can reduce the workload further. In this section we present the results of a simulation in which depots positions are also allowed to vary, in addition to the partitions. Since determining the optimal placement of depots is a non-convex problem, we are only able to look for locally optimal solutions. To this end, we use a natural variation of the well-known *Lloyd algorithm* for computing centroidal Voronoi partitions [22], [32]. The variation is straightforward: for a given initial set of depots P (Figure 5), we compute the optimal partitions using Algorithm 1 as before. Then, we relocate each depot p_i to the geometric median (the Fermat-Weber center) of its associated region R_i. The new optimal partitions are then re-computed, and so on and so forth. This algorithm is guaranteed to converge to a solution because the objective function value (the total weighted distance between depots and their assigned sub-regions) decreases at each of the two steps. A detailed proof of convergence can be found in [14]. The problem of relocating each depot p_i to the geometric median of its associated region R_i is called the continuous Fermat-Weber problem and was studied in [16]. For our purposes, we merely estimated the geometric median by discretizing R_i and selecting the best point in the discretization. The locally optimal depot locations and their corresponding partitions for the Drass map is shown in Figure 8. The algorithm takes fewer than 15 iterations to converge to the locally optimum.

B. Comparison with Voronoi Partitions

If we were to minimize the Fermat-Weber cost of (2) without the equal area constraints, then it is clear that we

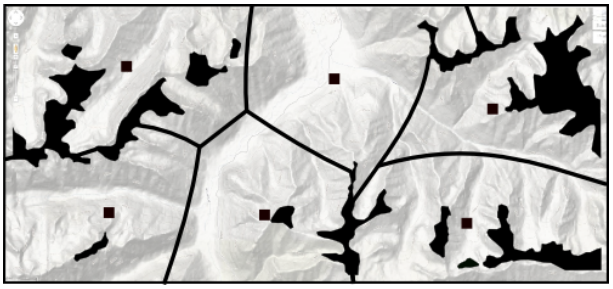


Fig. 8: Output of our simulation when Algorithm 1 is used as a sub-routine in a Lloyd algorithm.

would assign every demand point in the region to its nearest depot which results in a simple Voronoi decomposition. Hence, the Fermat-Weber cost of a Voronoi partition of a given set of depots is a lower bound for the objective function in (2) and we use this to compare the performance of partitions produced by our algorithm. We also compare the maximum distance of demand point assigned to a vehicle in our partitions with the corresponding value in Voronoi partitions. We compare these parameters for outputs shown in Figures 6 and 8 with their corresponding Voronoi partitions and the results are presented in Table I (values are in terms of percentage increase).

	Figure 6	Figure 8
Fermat-Weber cost	6.35%	0.43%
Maximum distance	2.89%	0%

TABLE I

ACKNOWLEDGMENT

We gratefully acknowledge the support of the Boeing Company, DARPA YFA grant N66001-12-1-4218, NSF award CMMI-1234585, and ONR grant N000141210719. We also thank Ravi Janardan and his research group, Yinyu Ye, and Shuzhong Zhang for helpful discussions.

REFERENCES

[1] B. Aronov, P. Carmi, and M. J. Katz. Minimum-cost load-balancing partitions. In *Proceedings of the twenty-second annual symposium on Computational geometry*, SoCG '06, pages 301–308, New York, NY, USA, 2006. ACM.

[2] B. Aronov, P. Carmi, and M.J. Katz. Minimum-cost load-balancing partitions. *Algorithmica*, 54(3):318–336, July 2009.

[3] F. Aurenhammer, F. Hoffmann, and B. Aronov. Minkowski-type theorems and least-squares partitioning. In *Proceedings of the eighth annual symposium on Computational geometry*, SCG '92, pages 350–357, New York, NY, USA, 1992. ACM.

[4] Y. Azar. On-line load balancing. In *Online Algorithms*, volume 1442 of *Lecture Notes in Computer Science*, pages 178–195. Springer Berlin / Heidelberg, 1998.

[5] O. Baron, O. Berman, D. Krass, and Q. Wang. The equitable location problem on the plane. *European Journal of Operational Research*, 183:578–590, 2007.

[6] S. Bereg, P. Bose, and D. Kirkpatrick. Equitable subdivisions within polygonal regions. *Computational Geometry*, 34(1):20 – 27, 2006. Special Issue on the Japan Conference on Discrete and Computational Geometry 2004.

[7] Oded Berman, Zvi Drezner, Arie Tamir, and George O. Wesolowsky. Optimal location with equitable loads. *Annals of Operations Research*, 167(1):307–325, March 2009.

[8] S. Boyd. Subgradients. Lecture slides for EE364b, Stanford University, 2011. See `http://www.stanford.edu/class/ee364b/lectures/subgradients_slides.pdf`.

[9] S. Boyd. Subgradients. Lecture slides for EE364b, Stanford University, 2011. See `http://see.stanford.edu/materials/lsocoee364b/05-localization_methods_slides.pdf`.

[10] J.G. Carlsson. Dividing a territory among several vehicles. *INFORMS Journal on Computing*, To appear, 2011.

[11] J.G. Carlsson, B. Armbruster, and Y. Ye. Finding equitable convex partitions of points in a polygon efficiently. *ACM Transactions on Algorithms*, 6, September 2010.

[12] J.G. Carlsson, D. Ge, A. Subramaniam, and Y. Ye. Solving the min-max multi-depot vehicle routing problem. In *Proceedings of the FIELDS Workshop on Global Optimization*, 2007.

[13] Z. Drezner and A. Suzuki. Covering continuous demand in the plane. *Journal of the Operational Research Society*, 61(5):878–881, 2010.

[14] Qiang Du, Maria Emelianenko, and Lili Ju. Convergence of the lloyd algorithm for computing centroidal voronoi tessellations. *SIAM journal on numerical analysis*, 44(1):102–119, 2006.

[15] J.W. Durham, R. Carli, P. Frasca, and F. Bullo. Discrete partitioning and coverage control for gossiping robots. *IEEE Transactions on Robotics*, 99:1 – 15, 2011.

[16] Sándor P. Fekete, Joseph S. B. Mitchell, and Karin Beurer. On the continuous Fermat-Weber problem. *Oper. Res.*, 53:61–76, January 2005.

[17] J.-L. Goffin, Z.-Q. Luo, and Y. Ye. Complexity analysis of an interior cutting plane method for convex feasibility problems. *SIAM J. on Optimization*, 6:638–652, March 1996.

[18] D. Haugland, S. C. Ho, and G. Laporte. Designing delivery districts for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 180(3):997 – 1010, 2007.

[19] Y. He and Z. Tan. Ordinal on-line scheduling for maximizing the minimum machine completion time. *Journal of Combinatorial Optimization*, 6(2):199–206, 2002.

[20] M. Jäger and B. Nebel. Dynamic decentralized area partitioning for cooperating cleaning robots. In *ICRA 2002*, pages 3577–3582, 2002.

[21] H. Kellerer, V. Kotov, M. Grazia Speranza, and Z. Tuza. Semi on-line algorithms for the partition problem. *Operations Research Letters*, 21(5):235 – 242, 1997.

[22] S. Lloyd. Least squares quantization in PCM. *Information Theory, IEEE Transactions on*, 28(2):129 – 137, 1982.

[23] D. G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, Inc., 1st edition, 1997.

[24] Q. Mérigot. A multiscale approach to optimal transport. *Computer Graphics Forum*, 30(5):1583–1592, 2011.

[25] J.S.B. Mitchell. Shortest paths among obstacles in the plane. In *Proceedings of the ninth annual symposium on Computational geometry*, SCG '93, pages 308–317, New York, NY, USA, 1993. ACM.

[26] Office of Naval Research. Basic research challenge: Decentralized online optimization, 2012. [See: `http://www.onr.navy.mil/~/media/Files/Funding-Announcements/Special-Notice/2012/12-SN-0006.ashx`, Online; accessed 13-Mar-2013].

[27] U.S Government Printing Office. Faa modernization and reform act of 2012, 2012. [See: `http://www.gpo.gov/fdsys/pkg/CRPT-112hrpt381/pdf/CRPT-112hrpt381.pdf`, Online; accessed 13-Mar-2013].

[28] M. Pavone, A. Arsie, E. Frazzoli, and F. Bullo. Distributed algorithms for environment partitioning in mobile robotic networks. *Automatic Control, IEEE Transactions on*, 56(8):1834–1848, 2011.

[29] M. Pavone, N. Bisnik, E. Frazzoli, and V. Isler. Decentralized vehicle routing in a stochastic and dynamic environment with customer impatience. In *RoboComm '07: Proceedings of the 1st international conference on Robot communication and coordination*, pages 1–8. IEEE Press, 2007.

[30] L. C. A. Pimenta, V. Kumar, R. C. Mesquita, and G. A. S. Pereira. Sensing and coverage for a network of heterogeneous robots. In *IEEE CDC'08*, pages 3947–3952, 2008.

[31] C. Villani. *Topics in optimal transportation*. Graduate studies in mathematics. American Mathematical Society, 2003.

[32] Wikipedia. Lloyd's algorithm, 2012. [Online; accessed 05-Feb-2012].