Footprint Placement for Mosaic Imaging by Sampling and Optimization

Scott A. Mitchell,[*] **Christopher G. Valicka,** [†]
Stephen Rowe,[‡] **Simon X. Zou**[‡]

Abstract

We consider the problem of selecting a small set (mosaic) of sensor images (footprints) whose union covers a two-dimensional Region Of Interest (ROI) on Earth. We take the approach of modeling the mosaic problem as a Mixed-Integer Linear Program (MILP). This allows solutions to this subproblem to feed into a larger remote-sensor collection-scheduling MILP. This enables the scheduler to dynamically consider alternative mosaics, without having to perform any new geometric computations. Our approach to set up the optimization problem uses maximal disk sampling and point-in-polygon geometric calculations. Footprints may be of any shape, even non-convex, and we show examples using a variety of shapes that may occur in practice. The general integer optimization problem can become computationally expensive for large problems. In practice, the number of placed footprints is within an order of magnitude of ten, making the time to solve to optimality on the order of minutes. This is fast enough to make the approach relevant for near real-time mission applications. We provide open source software for all our methods, "GeoPlace."

Optimization of Spatial Coverage

There are many spatial optimization problems, and several subcategories have been well studied in the planning and scheduling communities. For example, a common problem is where to place supply facilities to maximize utility. Subcategories include considering distances between suppliers and consumers modeled as points (Ghosh and Varakantham 2016), or when the consumers are moving along paths (Lowalekar et al. 2017; Funke, Nusser, and Storandt 2016). Path planning and monitoring (guarding or searching, "art-gallery" problems) in spatial domains is also common (Nussbaum and Yörükçü 2015; Deshpande et al. 2007). This sometimes involves coordinating a group of agents, and centralized planning benefits from significantly different techniques than distributed planning of semi-autonomous agents (Torreño, Sapena, and Onaindia 2015).

Similarly, optimizing the use of Earth-observing sensors and satellites is a fairly well studied optimization problem category, but there are many ways to model the problem, and within a model there are many aspects such as communication (Pralet et al. 2014), periodic or one-time image acquisition, storage, coordination between a constellation of heterogenous or homogeneous UAVs (Mersheeva and Friedrich 2015), scheduling, and stochastics. Solutions to some models and aspects are much more mature than others. A few recent optimization approaches have focused on scheduling communication and storage for periodic acquisition, where satellites travel quickly along one-dimensional paths with limited side-viewing capabilities (Frank, Do, and Tran 2016; Laborie and Messaoudi 2017). Constellation scheduling usually uses a centralized planner; Valicka (Valicka et al. 2017) and the references reported therein give a general overview. Often a single satellite image can cover an entire ROI, so collection is commonly modeled using discrete, isolated points. Sometimes collection requirements are that a single image covers an entire ROI. Other times a set of multiple sensor images are needed to cover a larger ROI.

Problem Definition

We focus on the spatial aspects of acquiring a *mosaic* of overlapping and adjacent images, called footprints herein. The problem is to find a mosaic that covers a continuous, two-dimensional region of the Earth's surface, called an *ROI* for Region Of Interest. Mosaics are common to aerial and space-based imaging and cover a much larger area of Earth than is possible with single images. Ensuring that there are no gaps between constituent images is important to generating high-fidelity maps and composite images. In some settings, significant image overlap is desired because it helps align adjacent images, and because the sensors may lose fidelity in their periphery. But each sensor image is expensive, so we must avoid wasteful overlap.

The ROI and sensor footprints are modeled as polygons, sometimes with holes. For exposition we start with circular footprints, but do not specialize our algorithms for these, and instead address the broader class of footprint shapes that are nearly convex and have good aspect ratios when projected on Earth. (describes a continuous quadratic optimization solution for the special case of circular footprints.)

[*]Center for Computing Research, samitch@sandia.gov

[†]Center for Information & Systems Research & Analysis,

[‡]Systems Mission Engineering, Sandia National Laboratories.

Applied Techniques

Solving the mosaic problem optimally is challenging, and for this and other reasons it remains much less studied within the optimization community than some other remote sensing problems. However, it is an important enough application, with high costs and consequences, that some applied communities have developed techniques for it. We highlight two solutions that are based on recursive subdivision, a common technique in computational geometry.

Merritt (Merritt 2011) considers a problem very similar to our mosaic problem, motivated by radar coverage of the U.S. The solution is an adaptation of the algorithm of Kazazakis (Kazazakis and Argyros 2002), based on recursive subdivision. If a polygon (subregion of the ROI) is sufficiently covered by one footprint, it is considered covered and discarded. Otherwise, it is subdivided. Merritt considers limited resources, covering less than the entire ROI, and experimental performance.

Daniels (Daniels, Mathur, and Grinde 2003) considers whether a set of polygonal shapes can cover a set of polygonal target items, an NP-hard problem. As in our problem, the covering shapes may overlap and extend beyond the domain. Daniels's method partitions the target items into triangles, and maximizes the number of triangles covered. Daniels iteratively uses a Lagrangian heuristic and recursive triangle subdivision.

Computational Geometry Approaches

When viewed from a purely computational geometry standpoint, the mosaic problem lies in the family of geometric coverage problems, so techniques from that community are relevant. Many approaches to geometric coverage exploit the structure of the covering shape. For example, circles lead to some elegant equations and gradient descent optimization (Stoyan and Patsuk 2010). Specialized solutions exist for axis-aligned rectangles. Typically, for general shapes it is theoretically hard to find optimal solutions. Hence approximation heuristics are common. While squares and circles are common examples of camera images, their projection from orbit are not circular or square (Zolnay 1971); see . Alternatively, one may retain circular or square images by projecting the ROI onto the camera image plane. However, the ROI becomes more complicated, and it was unclear how this would work in practice with time-varying sensor positions and coupling to the scheduling problem.

One category of practical covering algorithms is the "shifting technique" (Har-Peled 2011; 2008). One starts with a set of footprints covering the ROI, then makes local changes to try to reduce the size of the set. The initial covering may be a "greedy cover" by area, or a GreedyK-Center where one places footprints as far as possible from prior footprints, or a regular packing. The replace-with-fewer-disks technique of throwing away k (e.g. 2 or 3) footprints, then attempting to reinsert $k - 1$ regions without introducing gaps, works for disks (Mustafa, Raman, and Ray 2014; Ebeida et al. 2013) but is challenging for arbitrary shapes (Har-Peled 2011). Variations maintain coverage while improving other properties (Abdelkader, Mitchell, and Ebeida 2014).

Our Optimization Approach

We reformulate the continuous problem P as a discrete problem Q, amenable to Mixed-Integer linear Programming (MIP). This is an example of a relaxation technique, reformulating P into Q so it is possible to find an optimal solution to Q that also solves (but is suboptimal) for P. We reformulate because the original problem P is in general hard to solve using exact techniques. In our case, Q is a geometric version of the classical "min set cover" problem which is NP-Complete (Karp 1972). So, even though Q is easier than P, it is still hard to solve.

In a blog post, Har-Peled (Har-Peled 2011) sketches an approach very similar to ours, turning a cover-a-polygon-by-disks problem into a discrete optimization problem by first sampling the polygon, then covering the samples by disks, and expanding their radii to ensure polygon coverage. To our knowledge, Har-Peled has not published this approach or applied it to create the types of models that we study in this paper.

Algorithm

The footprint problem P of finding the minimum number of *footprints* and their placements that can cover a *Region Of Interest (ROI)* can be converted into a discrete optimization problem Q that approximates P. A footprint is a simply-connected region, and a placement of it is a translation of its shape. The ROI is a planar region, with perhaps isolated lines and points. The idea is to create a set of discrete coverage points within the ROI, and a set of nearby discrete placement points for footprints. We then solve a discrete optimization problem of where to place the footprints such that all of the coverage points are far enough inside at least one footprint, so that all of the ROI is inside the union of footprints.

In particular, we sample the ROI with well-spaced points. These must be covered by a footprint. Conceptually, there is a disk around each point with radius equal to a sample spacing parameter, and the union of disks covers the ROI. We only count points as "covered" if they are strictly inside a footprint by the spacing parameter, to ensure that if all samples are thus covered, the ROI space between them is truly covered. We have a second set of well-spaced samples, that cover both the ROI and some buffer region around it. Footprints may be placed at any of the second samples.

The procedure begins with the user choosing the sample spacing parameter ϵ. It controls how well the discrete points approximate the geometric ROI. The smaller the ϵ, the less overlap between footprints and the more likely we can cover the ROI with fewer footprints. The downside to using a small ϵ is an increase in the size of the MIP and a corresponding increase in solution time. See Figure 5 for an example of these trends. Fortunately, in practice, some overlap in the sensor images is desired and a fairly large ϵ is preferred.

The ROI is covered by some set of ϵ-radius circles (not the footprint), whose centers are *coverage points*. Hence the distance from any point of the ROI to the nearest coverage point is at most ϵ. We generate a second set of *placement points*, using a larger spacing to keep the number of points small. We consider placing a footprint at each placement point, and compute the set of coverage points within the footprint by at least ϵ distance. For each placement point, we have a Boolean 0–1 decision variable to decide whether to place a footprint there. The optimization problem Q is to minimize the sum of 0–1 decision variables (number of footprints). For each coverage point, we have a constraint that it is covered at least once. Any solution to Q is a solution to P, meaning the ROI is covered by footprints, but the optimal solution to Q may be suboptimal for P, in that it may have been possible to cover the ROI with fewer footprints. In this conversion the combinatorial descriptions of the boundary of the ROI and footprint are lost. The runtime of solving Q dominates. The method is insensitive to the number of sides, holes, curvature, and convexity of the ROI and footprint.

Sampling for Coverage Points

We generate the coverage points through a variant of "Simple MPS" (Ebeida et al. 2012), which was designed for square-shaped domains. Here we have generalized it to ROIs of arbitrary shape, but specialized it to 2D. See Algorithm 1 for the pseudocode and Figure 1 for an example run. It places random points at least r apart until every ROI point is within r of a sample, where $r = \epsilon$. Simple MPS divides the ROI into squares, and samples points uniformly from the squares. Accepted samples are the centers of disks of radius ϵ. A sample is rejected if it lies outside the ROI or inside a prior disk. After some constant number of sample attempts, such as 3 times the number of squares, all the squares are divided into 4. Squares are discarded if they lie completely outside the ROI or inside a disk. We repeat until the remaining squares are covered or smaller than machine precision. In practice, Simple MPS complexity is linear in the number of output samples, for both time and memory, and is easy to code because it only relies on simple and reliable geometric primitives such as is-point-in-circle and is-point-in-square. Alternatives include other Maximal Poisson-disk Sampling (MPS) algorithms, Delaunay refinement, or even structured patterns.

Algorithm 1 Simple MPS adapted from (Ebeida et al. 2012)

```
initialize uniform grid of squares
remove squares outside the ROI (plus buffer)
while $\exists$ empty squares, larger than machine precision do
  Randomly Sample:
  for 3 times the number of squares do
    select a square at random
    select a point in the square at random
    if the point is far enough ($> r$) from all prior samples
    and the point is inside the ROI (plus buffer) then
      accept the point as a sample
      remove the square
    end if
  end for
  Refine:
  for all squares do
    subdivide the square into 4 smaller squares
    remove squares too close ($< r$) to prior samples
    remove squares outside of the ROI (plus buffer)
  end for
end while
```

Sampling for Placement Points

The main difference between this step and sampling for coverage points is that placement points may be outside the ROI. We restrict placement points to the convex hull of the ROI to keep the size of the subsequent optimization problem small; depending on the footprint shape this restriction may exclude valuable solutions. See Figure 2 for an example.

We introduce the concept of an *anchor* point in a footprint. The position of a footprint is described by the position of its anchor point. Anchors are placed at placement points. The *footprint radius* is the farthest distance from a footprint point to its anchor point. A natural choice for the anchor is the centroid of the footprint, but there is no perfect choice except for circular footprints. To minimize the radius, one could use the center of the smallest enclosing circle of the footprint.

We rerun Simple MPS, but add a buffer region to the ROI. We consider the domain to be the region within both the maximum footprint-radius of the ROI, and the convex hull of the ROI. To determine if a sample point is inside this buffered domain, we use the same kind of distance calculations as described in the next subsection. For simplicity we initialize the set of placement points to be a subset of coverage points, then fill in the rest of the domain, rather than generating them independently from scratch. However, we use a larger sampling radius, e.g. $r = \sqrt{2}\epsilon$, in order to keep the MIP Q small enough to be solved quickly.

Computing Coverage

Our method is not restricted to polygonal (straight sided) ROIs and footprints. It is straightforward to convert a curved footprint into a polygonal one inside it. Conversely, it is easy to find a polygon enclosing an ROI or its convex hull. Conceptually, one may consider a footprint shrunk by distance ϵ. In practice, instead we compute the distance to the footprint boundary because it is robust, and simpler than computing an offset polygon. To determine if a coverage point is at least ϵ inside a footprint, we determine whether it is inside the polygon using the winding number. If so, then we determine its distance to the boundary by projecting to each line segment of the boundary. Figure 3 shows an example for our square annulus ROI.

These same operations are used for the buffered region around the ROI for placement points. The winding number also provides a robust answer to determining if a point lies in the convex hull.

Optimization Problem Q

The optimal set of footprints for Q is an epsilon approximation to P, in that any coverage of the same ROI with footprints that are smaller by epsilon must use at least the same number of footprints. The converted problem Q has a number of rows (constraints and objectives) linear in the total number of samples, the sum of the number of coverage and

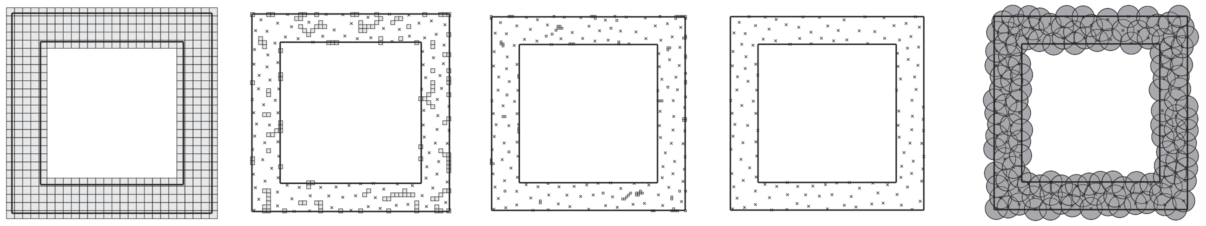
(a) Square grid containing the ROI. (b) End of iteration 1 (c) End of iteration 2 (d) Coverage points. (e) ROI lies inside ϵ-radius coverage circles.

Figure 1: Creating *coverage* samples using ϵ spacing. Coverage points are "x" and the small squares are the ones that are not yet covered by an ϵ-radius circle centered at a coverage point.

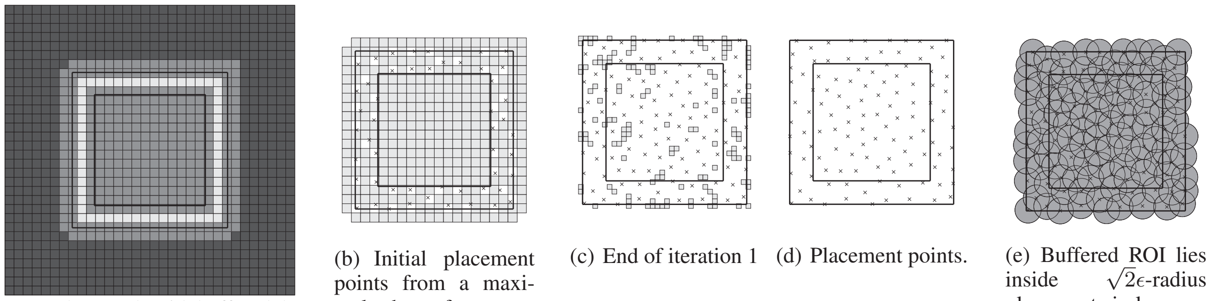
(a) Background grid buffered by the (large) footprint radius. (b) Initial placement points from a maximal subset of coverage points. (c) End of iteration 1 (d) Placement points. (e) Buffered ROI lies inside $\sqrt{2}\epsilon$-radius placement circles.

Figure 2: Creating *placement* samples using $\sqrt{2}\epsilon$ spacing. Placement points may be outside the ROI, but are limited to its convex hull to keep the problem size small, so the black squares in (a) are discarded.

placement points. Each row has at most a linear number of columns (variables with non-zero coefficients), but usually much less depending on the choice of epsilon. The number of rows and columns of Q are each $O(1/\epsilon^2)$, leading to the problem size being $O(1/\epsilon^4)$.

Describing and Solving the MIP

Let T denote the (potential) footprint placement points and C the (required) coverage points. Let $\delta_{t\in T} \in \{0,1\}$ represent the decision to place a footprint at t. The set of placements $j \in J$ that would cover $c \in C$ is denoted $J(c)$. The MIP to minimize the number of placed footprints follows:

$$\begin{aligned} &\min \sum_{t}^{T} \delta_t \\ &\text{s.t.} \sum_{j}^{J(c)} \delta_j \geq 1 \qquad \forall c \in C, \\ &\delta_t \in \{0,1\} \qquad \forall t \in T. \end{aligned} \tag{1}$$

Spreading Out Footprints

It is often desirable to cover area outside the ROI if it does not require more footprints, i.e., it can be done "for free." We explored a variation that discouraged placing footprints that strongly overlap, by adding constraints and objectives to the mixed-integer linear program of Equation (1). A natural choice would be maximize the sum of pairwise distances, but in practice this did not work well because it overly penalized the beneficial overlap, and resulted in odd placements for irregularly shaped ROIs. What worked better was to minimize the number of coverage points that two footprints have in common. Our approach has the advantage of capturing the geometry of the ROI. For efficiency, we only introduced variables for placement pairs with non-zero overlap. We define a placement pair (i, j) where $i, j \in T$, over the set D of non-zero overlapping placement pairs, where $|D|$ depends on placement point locations and footprint shape. Let $f(i, j) : D \to [0, 1]$ be the mapping of placement pairs to their respective overlap. We then define $\phi_{i,j} \in \{0, 1\}$ with $(i, j) \in D$ to represent whether or not the placement pair (i, j) has footprints placed at both placement pair locations $i, j \in T$. To enforce that $\phi_{i,j} = 1$ if and only if $\delta_i = \delta_j = 1$, we construct the following three constraints to form a logical AND, and add them to Equation (1).

$$\begin{aligned} \phi_{i,j} &\geq \delta_i + \delta_j - 1, \forall (i,\, j) \in D, \\ \phi_{i,j} &\leq \delta_i, \forall (i,\, j) \in D, \\ \phi_{i,j} &\leq \delta_j, \forall (i,\, j) \in D. \end{aligned} \tag{2}$$

To penalize overlap we add a term to the previous objec-

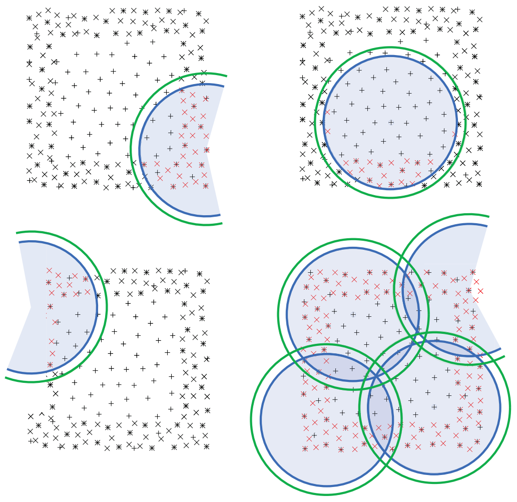

Figure 3: Placement ("+") and coverage points("x"). The footprint is the green circle. The footprint radius is reduced by ϵ to the blue circle, ensuring that any gaps between the coverage points are covered by some green circle. Coverage points inside the blue circle are drawn in red and included in the MIP Q to model "if a footprint is placed here, these coverage points are covered." The optimal solution uses four footprints: two are centered outside the ROI.

tive, resulting in the following mixed-integer linear program:

$$\begin{aligned} &\min \sum_{t}^{T} \delta_t + \sigma \sum_{(i,j)}^{D} f(i,j)\phi_{i,j}, \\ &\text{s.t.} \sum_{j}^{J(c)} \delta_j \geq 1 && \forall c \in C, \\ &\delta_t \leq 1 && \forall t \in T, \\ &\delta_t \in \{0,1\} && \forall t \in T, \qquad (3) \\ &\phi_{i,j} \geq \delta_i + \delta_j - 1, && \forall (i,\, j) \in D, \\ &\phi_{i,j} \leq \delta_i, && \forall (i,\, j) \in D, \\ &\phi_{i,j} \leq \delta_j, && \forall (i,\, j) \in D, \end{aligned}$$

where σ is a constant. In practice, we chose σ to be quite small, $\sigma \approx 0.001$ to ensure that minimizing the number of footprints dominated the desire for non-overlap.

But Keep Footprints Near the Domain

Unfortunately, Equation (3) tended to place footprints that were mostly outside the ROI. Hence, we countered the spreading objective by a small term rewarding footprint placements that covered more coverage points. The objective function of Equation (3) then became

$$\sum_{t}^{T} \delta_t(1 - \sigma_1 h(t)) + \sigma_2 \sum_{(i,j)}^{D} f(i,j)\phi_{i,j}, \qquad (4)$$

where $h(t) : T \to \mathbb{N}$ is the number of coverage points covered by a footprint at t, and depends on the geometry. As before, both σ_1 and σ_2 are chosen to be small so the number of placed footprints dominates the objective.

Examples

Figure 4 includes examples of non-circular and non-convex polygonal footprints. For these, the offset polygons are not shown because we used the simpler distance-to-boundary calculations. The runtime of sampling and computing coverage is insignificant. The "Right Triangle" example illustrates the well-known observation that a right-triangle, without rotations, is an inefficient covering shape. Figure 5 includes instance and performance statistics for the example ROI at the top of Figure 4, using varying sample spacing ϵ, and with the optimality tolerance set to 0.01%.

Realistic Shapes

The footprint in the bottom row of Figure 4 is the shape of a part of the globe seen by a square camera lens aimed at a glancing angle.

GeoPlace Open Source Software

We provide the GeoPlace open-source software on GitHub[1]. It includes libraries for sampling domains that are polygons with holes. It provides coverage computations for footprints that are arbitrary simple polygons. It also has routines for visualizing solutions. Besides providing all the algorithms in this paper, it also solves the related "sub-footprint placement" problem of where to focus the sensor within a footprint (Rowe et al. 2017). Sub-footprint variations include constrained bandwidth placement and an objective to maximize average bandwidth for the placed sub-footprints.

The development environment was Linux and problems were solved on a 64-core workstation with 2.4GHz AMD processors and 512GB of RAM. Source code is primarily Python 2.7 and C++ with optimization models expressed using the Pyomo (Hart et al. 2012) optimization modeling libraries. Instances of the optimization problem Q were solved using both Gurobi[2] (free academic use license) and CPLEX[3] (commercial) solvers.

Extensions

Other Objectives

An advantage of the MIP modeling approach over a purely geometric one is that it is easy to adapt to other goals and constraints. For example, other contexts desire maximum overlap to increase redundant imaging in important regions or to help stitch separate camera images into the mosaic.

Multiple Footprint Shapes

It is straightforward to extend to multiple footprint shapes. For example, the same camera at different locations and view angles of the earth will project different footprints.

[1]github.com/cgvalic/GeoPlace

[2]www.gurobi.com

[3]www.ibm.com/analytics/data-science/prescriptive-analytics/cplex-optimizer

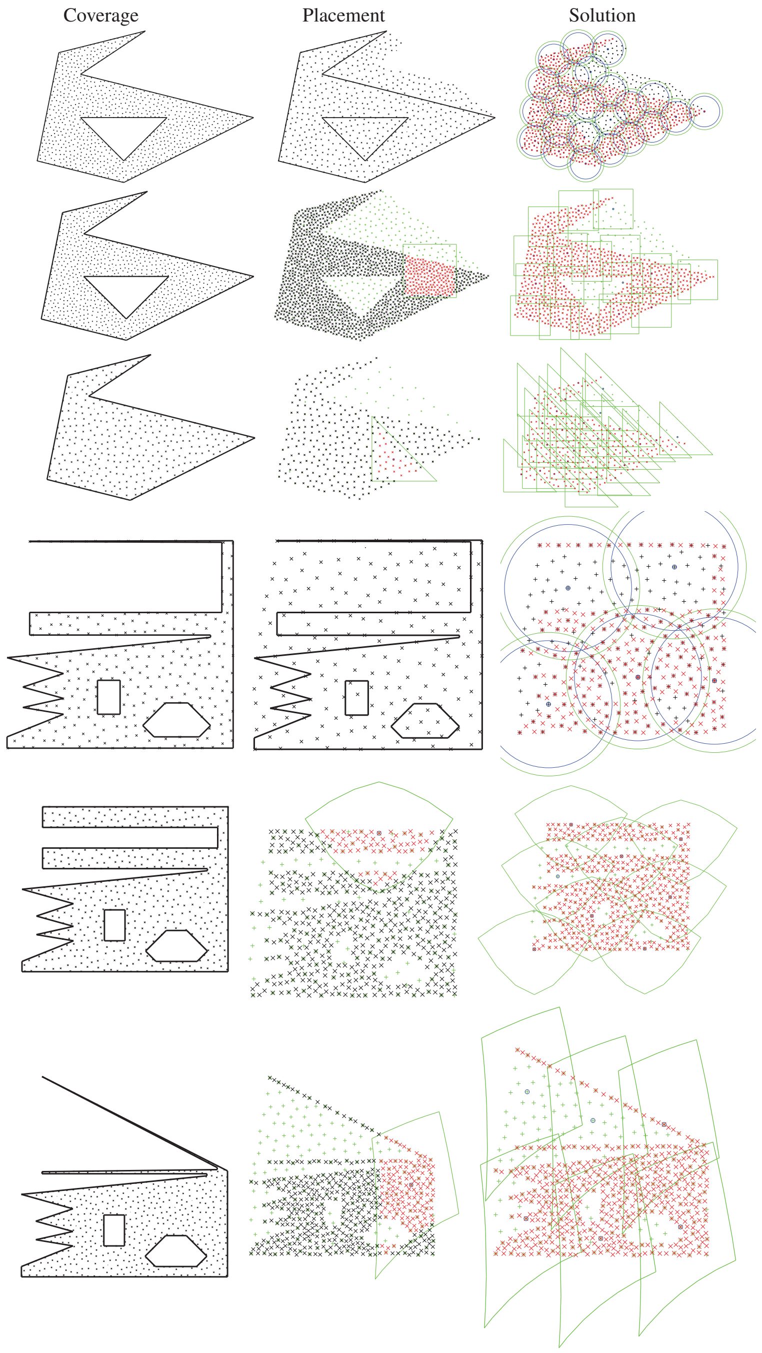

Figure 4: Examples of different ROI and footprint shapes and resultant coverings.

Multiple footprints can also come from different cameras with different aperture sizes or shapes. We may include weighted objectives, and constraints on the number of times we use each one.

Faster Coverage Computation and Smarter Placement Points

In our setting, it was adequate for each placement point to compute its coverage points independently by brute force. However, it is possible to simultaneously compute a reasonable set of placement points, and which coverage points are covered for each of them, and this efficiency may be helpful for other settings.

The reference footprint is shrunk by ϵ and inverted through each of the coverage points: the inversion shape is the set of locations of the shrunk footprint's anchor point such that the shrunk footprint will cover that coverage point. The arrangement of intersections of all inversions are pro-

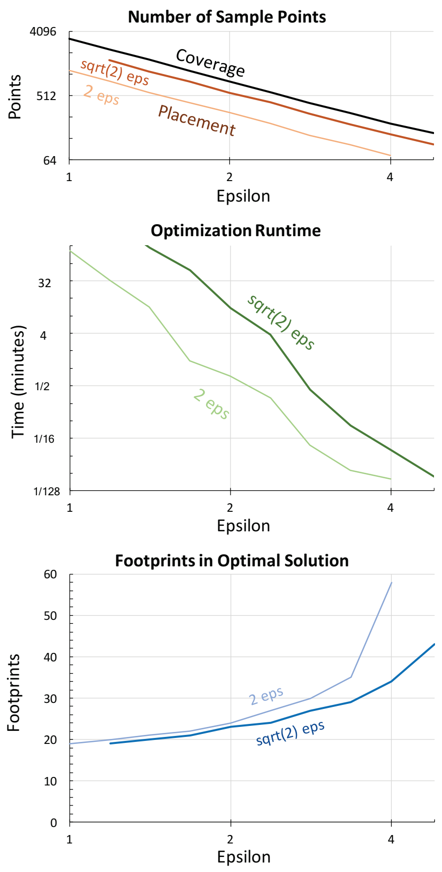

Figure 5: Example performance showing more footprints and shorter runtime for larger ϵ. All instances were solved to optimality. To further explore the effect of the spacing parameters, we report performance for both 2ϵ and $\sqrt{2}\epsilon$ spacing for the placement points.

cessed by a plane sweep, to find the cells that cover a maximal set of coverage points, the maximal canonical sets. For each such set, we may select a placement point.

Using inversion and plane sweep to create the intersection cells and the maximal canonical sets is described in an arXiv paper, "Discretization of Planar Geometric Cover Problems" (Jang and Choi 2014). Inversion is reflection of the footprint through the anchor point, which preserves convexity. Sweep the arrangement and identify the cells. The maximal canonical sets are the cells who have no adjacent cells that cover more points. This can be discovered by a directed graph from low to high adjacent cells: maximal canonical cells have no outgoing edges.

Alternative Continuous Quadratic Optimization

There is a continuous optimization solution for circular footprints. The following determines whether there exists a placement of k footprints that covers all the sample points. So, the overall problem would be to find the smallest k for which the optimization problem is feasible. The continuous decision variables are the (x, y) coordinates of each of the k footprints' anchor points, the circle centers. The basic optimization problem has no objective, and just the constraint that for each sample point, the distance between it and the closest footprint is less than zero, i.e., it lies inside a footprint. For circular footprints, the distance is just the distance to the center minus the radius, which is a simple quadratic if we use the square of the distance instead. However, for more complicated footprints, even something as simple as a convex square, the distance is not even a quadratic, and the distance to the footprint boundary can not be described by only the Euclidean distance to the footprint center.

Conclusions

We have demonstrated a MILP solution to the problem of finding a small set of sensor images that collectively cover a two-dimensional region. Solutions for problems of typical size for remote sensing missions can be found at mission-relevant time scales. By using best practices from the computational geometry community to set up the discrete optimization problem, it is simple to alter the formulation to solve a number of related mosaic selection problems. Software implementing these techniques are publicly available from the GeoPlace GitHub repository. By using a MILP formulation, the mosaic selection subproblem can be coupled with larger MILP sensor scheduling problems. We wish to explore this coupling as well as a coupling with the subfootprint problem (Rowe et al. 2017).

There are many opportunities for future research. We wish to consider a single footprint as its shape varies in time according to the sensor's orbit. This is related, but not identical, to exploring problems involving the simultaneous placement of different footprint shapes. To date, we have used off-the-shelf commercial solver parameters and used no optimality gap tolerance. We also wish to explore the benefits of developing heuristics and exploiting the benefits of seeding a MIP solver with an initial, feasible solution. This could come from a greedy cover, the shifting heuristic, etc., and perhaps enable our approach to be fast enough to work in real-time or on larger problems.

Acknowledgements

We thank Karen Daniels and the authors of SimpleMPS for kindly sharing their source code.

This work was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research (ASCR), Applied Mathematics Program, and the Laboratory Directed Research and Development program at Sandia National Laboratories. Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

References

Abdelkader, A.; Mitchell, S. A.; and Ebeida, M. S. 2014. Steiner point reduction in planar Delaunay meshes. In *Symp. on Comp. Geom., YRF*.

Daniels, K.; Mathur, A.; and Grinde, R. 2003. A combinatorial maximum cover approach to 2D translational geometric covering. In *CCCG*, 2–5.

Deshpande, A.; Kim, T.; Demaine, E. D.; and Sarma, S. E. 2007. A pseudopolynomial time $O(\log n)$-approximation algorithm for art gallery problems. In *WADS*, 163–174. Springer.

Ebeida, M. S.; Mitchell, S. A.; Patney, A.; Davidson, A. A.; and Owens, J. D. 2012. A simple algorithm for maximal Poisson-disk sampling in high dimensions. *Computer Graphics Forum* 31(2):785–794.

Ebeida, M. S.; Mahmoud, A. H.; Awad, M. A.; Mohammed, M. A.; Mitchell, S. A.; Rand, A.; and Owens, J. D. 2013. Sifted disks. *Computer Graphics Forum* 32(2pt4):509–518.

Frank, J.; Do, M.; and Tran, T. 2016. Scheduling ocean color observations for a GEO-stationary satellite. In *Int. Conf. on Automated Planning and Scheduling*.

Funke, S.; Nusser, A.; and Storandt, S. 2016. Placement of loading stations for electric vehicles: Allowing small detours. In *Int. Conf. on Automated Planning and Scheduling*.

Ghosh, S., and Varakantham, P. 2016. Strategic planning for setting up base stations in emergency medical systems. In *Int. Conf. on Automated Planning and Scheduling*.

Har-Peled, S. 2008. *Geometric approximation algorithms*, volume 173. Amer. Math. Soc.

Har-Peled, S. 2011. Covering a simple polygon with circles. http://cstheory.stackexchange.com/questions/8799/covering-a-simple-poly.

Hart, W. E.; Laird, C.; Watson, J.-P.; and Woodruff, D. L. 2012. *Pyomo–optimization modeling in Python*, volume 67. Springer Science & Business Media.

Jang, D., and Choi, H. 2014. Discretization of planar geometric cover problems. *CoRR* abs/1411.6810.

Karp, R. 1972. Reducibility among combinatorial problems. In Miller, R., and Thatcher, J., eds., *Complexity of Computer Computations*. Plenum Press. 85–103.

Kazazakis, G. D., and Argyros, A. A. 2002. Fast positioning of limited-visibility guards for the inspection of 2D workspaces. In *IROS*, volume 3, 2843 – 2848. IEEE.

Laborie, P., and Messaoudi, B. 2017. New results for the GEO-CAPE observation scheduling problem. In *Int. Conf. on Automated Planning and Scheduling*.

Lowalekar, M.; Varakantham, P.; Ghosh, S.; Jena, S.; and Jaillet, P. 2017. Online repositioning in bike sharing systems. In *Int. Conf. on Automated Planning and Scheduling*.

Merritt, K. M. 2011. Coverage of continuous regions in Euclidean space using homogeneous resources with application to the allocation of the phased array radar systems. Master's thesis, Air Force Inst. of Tech.

Mersheeva, V., and Friedrich, G. 2015. Multi-UAV monitoring with priorities and limited energy resources. In *Int. Conf. on Automated Planning and Scheduling*.

Mustafa, N. H.; Raman, R.; and Ray, S. 2014. QPTAS for geometric set-cover problems via optimal separators. *CoRR* abs/1403.0835.

Nussbaum, D., and Yörükçü, A. 2015. Moving target search with subgoal graphs. In *Int. Conf. on Automated Planning and Scheduling*.

Pralet, C.; Verfaillie, G.; Maillard, A.; Hébrard, E.; Jozefowiez, N.; Huguet, M.-J.; Desmousceaux, T.; Blanc-Paques, P.; and Jaubert, J. 2014. Satellite data download management with uncertainty about the generated volumes. In *Int. Conf. on Automated Planning and Scheduling*.

Rowe, S.; Valicka, C. G.; Mitchell, S. A.; and Zou, S. X. 2017. Nonoverlapping grid-aligned rectangle placement for high value areas. In *29th Canadian Conference on Computational Geometry, CCCG 2017*, 132–137. http://2017.cccg.ca/proceedings/CCCG2017.pdf.

Stoyan, Y. G., and Patsuk, V. M. 2010. Covering a compact polygonal set by identical circles. *Computational Optimization and Applications* 46(1):75–92.

Torreño, A.; Sapena, Ó.; and Onaindia, E. 2015. Global heuristics for distributed cooperative multi-agent planning. In *Int. Conf. on Automated Planning and Scheduling*.

Valicka, C.; Garcia, D.; Staid, A.; Watson, J.-P.; Hackebeil, G.; Rathinam, S.; and Ntaimo, L. 2017. Models for optimal constellation scheduling under weather uncertainty. Under review.

Zolnay, S. L. 1971. Earth coverage (”footprint”) of a satellite-borne antenna. Technical report, DTIC Document.