A Combinatorial Maximum Cover Approach to 2D Translational Geometric Covering

K. Daniels A. Mathur* R. Grinde†

1 Introduction

The goal of this research is to provide practical solutions for covering problems. Covering problems arise in a variety of practical settings such as manufacturing, telecommunications, spatial query optimization, publish/subscribe middleware, graphics, molecular modeling, medical treatment and mobile military sensor coverage and targeting. For example, in the mobile sensor case a region is associated with each sensor representing the extent of its coverage. The goal is to find placements of the sensors so that, together, they cover a collection of target regions. This section specifies the covering problem addressed here, surveys related work and gives an overview of the remainder of the paper.

1.1 Problem Specification

A finite collection of shapes is specified along with the allowable motions for the shapes. Each shape is a semi-algebraic subset of $\mathbb{R}^n$. The shapes are rigid. The allowable motions are a subgroup $\mathcal{G}$ of $SO(n,\mathbb{R}) \ltimes \mathbb{R}^n$, the group of rigid motions of $\mathbb{R}^n$ of the first type. We seek only one solution instead of all possible solutions.

Rigid Exact Covering: Let $Q = \{Q_1, Q_2, ..., Q_m\}$ be a set of rigid covering items and $P = \{P_1, P_2, ..., P_l\}$ be a set of rigid target items. Let $\mathcal{G}$ be a subgroup of $SO(n,\mathbb{R}) \ltimes \mathbb{R}^n$. Our covering problem seeks $\gamma = \{\gamma_1, \dots, \gamma_j, \dots, \gamma_m\}$, where $\gamma_j \in \mathcal{G}$, such that:

$$P \subseteq \bigcup_{1 \le j \le m} \gamma_j(Q_j). \tag{1.1}$$

This paper addresses 2D rigid, exact, covering for polygonal shapes with translational motion of the items in Q. The items in P are not allowed to move. Polygonal items may be nonconvex. Rigid, exact, translational covering is NP-hard [DI01b, DI01a].

*K.D. and A.M. are with the Computer Science Department, University of Massachusetts Lowell, Lowell, MA, 01854.

†R.G. is with the Decision Sciences Department, The Whittemore School of Business and Economics, The University of New Hampshire, Durham, NH, 03824.

1.2 Related Work

Algorithmic[1] 2D covering work has primarily focused on specialized forms of the covering shapes and the target shapes, often addressed the optimization problem of minimizing the number of covering polygons, and identified many NP-complete covering problems. [Tot97] surveys results for covering the plane with congruent convex shapes. [JCR88] summarizes early hardness results for coverings that are also decompositions. [DI01a] surveys some work on covering and closely related problems.

The first known work on NP-hard 2D translational covering for arbitrary polygonal covering and target items is [DI01b] and [DI01a]. In [DI01a] an assignment constrains each covering polygon to cover a particular point in the target set. An initial subset of target points is selected, assignments are generated, and the result is tested to see if the assignments guarantee coverage of the entire target set. If not, the subset of target points is augmented and the process repeats. The convex hull of the target set is selected as the initial subset of target points. Assignments for a small collection of target points can sometimes guarantee coverage of the entire target set using a convexity coverage property. [DI01b] builds on [DI01a] using intersection graphs. Given polygonal P, Q, and a translation vector for Q, let R be the partition of P induced by the boundaries of the translated items of Q. An intersection graph[2] for a cover is an undirected graph containing one node for each region of R and an edge connecting each pair of nodes whose regions of R share an edge in R. [DI01b] seeks covers having particular intersection graph topologies.

In Grinde and Daniels [GD99] a combinatorial covering optimization problem is reformulated along the lines of a maximum cover location problem. [GD99] addresses a two-phase layout problem in apparel manufacturing. The first phase places large pattern pieces and the second (trim placement) places small pieces.

[1]Nonalgorithmic results on covering and packing, such as density bounds, are surveyed in [Tot97].

[2]Intersection graphs have been used previously in the context of covering (see [FPT81], for example).

[GD99] views the trim placement problem as a collection of many small containment problems followed by a combinatorial covering problem. Each containment problem identifies a group of trim pieces that fit within a container delimited by large pattern pieces. The combinatorial problem maximizes a weighted sum of placed trim pieces by selecting one group for each container. An integer programming (IP) formulation of the covering problem is solved using a Lagrangian Heuristic.

1.3 Overview

The approaches of [DI01a] and [DI01b] work well for problem instances in which the number of vertices of the convex hull of P is small, the entire convex hull can be covered by the covering shapes, and the number of faces of a convex decomposition of Q is small. In such cases these algorithms can often find a cover by examining only a small number of candidate assignments. However, these methods lack a strong mechanism for deciding which covering polygon should cover which parts of P. The goal of this paper is to reduce this limitation. The new heuristic uses the combinatorial covering method of [GD99] to maximize coverage of P. Section 2 describes the heuristic. Section 3 discusses implementation and results. Results of the new technique are compared with the polygonal covering algorithm of [DI01b], which builds on [DI01a].

2 Covering Heuristic

For each polygon of Q the new heuristic creates a list of groups of triangular regions of P that it can cover. It then selects one group for each polygon of Q. The selection process uses the Lagrangian Heuristic of [GD99] to try to maximize the number of triangles covered. If the union of the covered triangles covers all of P, then the covering heuristic terminates successfully. Otherwise, it subdivides an uncovered triangle of P, the group list is augmented, and the group selection process repeats. A subdivision tolerance provides a stopping criterion. The success of this approach relies on the strength of the group generation process and the method for selecting one group for each polygon of Q. Section 2.1 provides high-level pseudocode. Section 2.2 describes the combinatorial covering technique that performs group selection.

2.1 Pseudocode

COVER (P, Q, τ)
  $T \leftarrow$ TRIANGULATE(P)
  $G \leftarrow \varnothing$
  $ToleranceReached \leftarrow$ FALSE
  while not $ToleranceReached$
    $G \leftarrow$ ADD-GROUPS(T, P, Q, G)
    $\gamma \leftarrow$LAGRANGIAN-COVER(G, T)
    if $(P \setminus \bigcup_j \gamma_j(Q_j)) = \varnothing$
     then return γ
     else $T' \leftarrow$ SUBDIVIDE-TRI(T, τ)
       if $T' = T$
        then $ToleranceReached \leftarrow$ TRUE
        else $T \leftarrow T'$
  report failure

ADD-GROUPS (T, P, Q, G)
  for each triangle $t \in T$
   if $\{t\} \notin G_S$
    then for each $Q_j \in Q$
     $K \leftarrow t \oplus -\overline{Q_j}$
     if $K \neq \varnothing$
      then $\vec{v} \leftarrow$ first vertex of K
       $G_S \leftarrow G_S \cup \{t\}$,$G_Q \leftarrow G_Q \cup \{Q_j, \{t\}, \vec{v}\}$
       for each vertex $\vec{v}$ of K
        $C \leftarrow P \cap \vec{v}(Q_j)$
        $g \leftarrow \{t' \in T | t' \subseteq C\}$
        $G_S \leftarrow G_S \cup g$, $G_Q \leftarrow G_Q \cup \{Q_j, g, \vec{v}\}$
  return G

The group structure G consists of two types of information[3] : 1) G_S: a list of all groups and, for each group g, a list of triangles of T in g, and 2) G_Q: for each Q_j in Q, a list of all groups that fit into Q_j and, for each group g, a translation vector for Q_j that allows it to simultaneously cover all the triangles in g. The subdivision tolerance τ specifies the maximum number of triangles in T. If the subdivision tolerance has not been reached, SUBDIVIDE-TRI subdivides into several triangles the largest triangle of T that is not covered by the current group selections. LAGRANGIAN-COVER uses the method described in the Section 2.2 below to attempt to maximize coverage of the triangles in T.

ADD-GROUPS uses the translational intersection and containment properties of the Minkowski sum[4].

[3] Although G is not a set, we use set notation to manipulate it. The meaning should be clear from context.

[4] The symbol $\oplus$ is a Minkowski sum operator: $A \oplus B = \{a + b | a \in A, b \in B\}$.

For each triangle t that fits within a particular Q_j, ADD-GROUPS finds the polygon K representing all translations of Q_j for which t fits into Q_j. It then attempts to identify large groups of triangles that Q_j can simultaneously cover by successively placing Q_j at vertices of K. Each vertex of K represents a two-contact placement[5] of t within Q_j. For each such placement, ADD-GROUPS determines which triangles of T fit into Q_j; these form a group.

2.2 LAGRANGIAN-COVER

LAGRANGIAN-COVER uses a Lagrangian Heuristic to solve the following IP formulation. The formulation has two groups of integer decision variables. Assignment variables that show if a given group is assigned to a particular Q_j and usage variables that reflect whether or not a given group is used in the optimal solution are given in Eqns. 2.1 and 2.2 respectively for $1 \leq i \leq |T|, 1 \leq k \leq |G_S|, 1 \leq j \leq m$.

$$g_{kj} = \begin{cases} 1 if\ group\ k\ is\ assigned\ to\ Q_j \\ 0 \quad\quad\quad otherwise \end{cases} \tag{2.1}$$

$$t_i \ = \begin{cases} 1 if\ triangle\ i\ \in group\ k\ \ni g_{kj} = 1 \\ 0 \quad\quad\quad\quad otherwise \end{cases} \tag{2.2}$$

Two types of parameters are given below in Eqns. 2.3 and 2.4; these provide coefficients for the constraints of the IP formulation.

$$a_{ik} = \begin{cases} 1 if\ triangle\ i\ \in group\ k \\ 0 \quad\quad otherwise \end{cases} \tag{2.3}$$

$$b_{kj} = \begin{cases} 1 if\ group\ k\ fits\ in\ Q_j \\ 0 \quad\quad otherwise \end{cases} \tag{2.4}$$

The IP model is stated as follows:

$$maximize \sum_{i=1}^{|T|} t_i \tag{2.5}$$

$$subject\ to \sum_{k \ni b_{kj}=1} g_{jk} = 1, j = 1, \ldots, m \tag{2.6}$$

$$t_i \leq \sum_{j=1}^{m} \sum_{k \ni b_{kj}=1} a_{ik} g_{kj}, i = 1, \ldots, |T| \tag{2.7}$$

The objective function in Eqn. 2.5 maximizes the number of triangles covered. The m constraints of Eqn. 2.6 ensure that exactly one group is selected for each Q_j. The constraints of Eqn. 2.7 cause a value of 1 to be contributed to the objective function for each triangle covered by a Q_j, where that triangle is in a group selected for that Q_j.

This IP formulation is a special case of the more general one in [GD99]. In [GD99], groups of trim pieces fit into containers and a group is selected for each container. A triangle of P in the new heuristic corresponds to a trim piece in [GD99]. A Q_j maps to a container. The formulation of [GD99] allows a trim piece to have a weight and it accommodates categories of trim pieces; these features are not used here.

Execution times can vary substantially when IP software is used to solve the IP model. For this reason the constraint set of Eqn. 2.7 is relaxed, bringing it into the objective function so that the optimization-based technique Lagrangian Relaxation can be used in conjunction with a local improvement heuristic. The resulting Lagrangian Heuristic does not guarantee optimality, but tests in [GD99] showed it was successful in finding the optimum most of the time, with shorter solution times than traditional branch-and-bound methods. Details of this Lagrangian Heuristic applied to trim placement appear in [GD99]; the same heuristic is used here in LAGRANGIAN-COVER.

3 Implementation and Results

The implementation uses the LEDA and CGAL C++ algorithms libraries. It also uses the Lagrangian Heuristic code of [GD99], with small modifications. In Table 1 results of the new method are compared with the polygonal covering algorithm of [DI01b], which builds on [DI01a]. The first column gives the test case number. The second column shows the number m of polygons in Q. The third column shows the number η of vertices of P. The value of l is 1 for the examples shown, but the algorithm is designed to handle $l \geq 1$. In columns 4-7, the new heuristic is numbered 1 and the algorithm of [DI01b] is numbered 2. Columns 4 and 5 show the number of points of P examined in order to find a cover. For the new heuristic this is the total number of triangle vertices. For the previous algorithm this is the number of points of P involved in assignments. Columns 6 and 7 show execution time in seconds on a 450 MHz SPARC Ultra[6]. The stopping criterion τ for the new heuristic is 300 triangles. Cases in which the new heuristic reached this limit without finding a cover are denoted by $*$. A time bound of 10 minutes was imposed on the previous algorithm. Cases in which it reached this limit without finding a cover

[5] A two-contact placement removes both translational degrees of freedom.

[6] SPARC Ultra is a trademark of Sun Microsystems Corporation.

#	m	η	# Pts 1	# Pts 2	Time 1	Time 2
1	3	9	24	**	25	**
2	7	20	20	**	23	**
3	2	12	12	18	2	2
4	2	6	*	11	*	3
5	2	12	39	14	71	2
6	2	12	12	12	3	1
7	2	5	11	6	3	4
8	2	6	12	6	5	1
9	3	6	6	6	1	3
10	3	12	12	**	3	**
11	3	12	12	**	3	**
12	5	9	54	**	258	**
13	4	12	24	**	33	**

Table 1: Covering Comparison

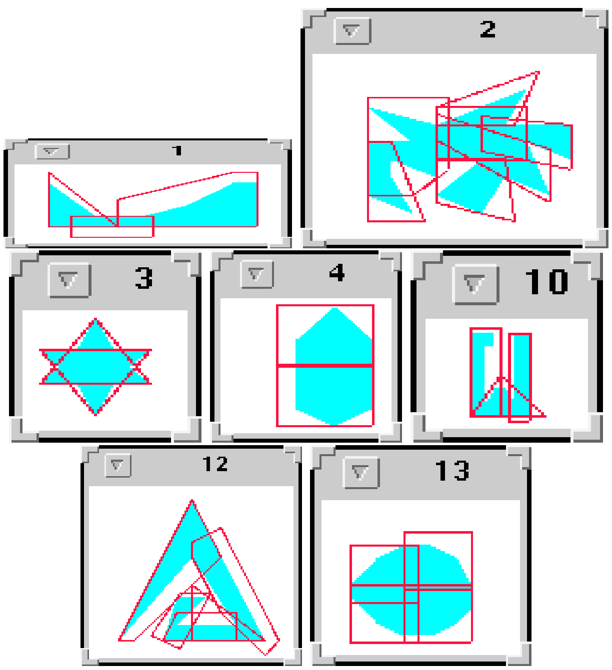

Figure 1: Covers for Selected Rows of Table 1

are denoted by $**$. [DI01b] uses a LEDA randomized convex decomposition function. Due to randomization the results reported for that algorithm are the best results obtained in 10 test runs. Figure 1 depicts covers obtained for selected rows of Table 1. For row 4 the cover was obtained from the algorithm of [DI01b]; the remaining figures are from the new heuristic. Although in each example each $P \cap \vec{v}(Q_j)$ consists of a single connected component, this need not be the case.

The algorithm of [DI01b] works well for problem instances in which the number of points in the convex hull of P is small, the number of faces of a convex decomposition of Q is small and the entire convex hull of P can be covered. In such cases it can often find a cover by examining a small number of candidate assignments of covering polygons to points of P and seeking covers with particular intersection graph topologies. However, it is limited by the lack of a strong method for deciding which covering polygon should cover which parts of the target set. The new heuristic uses a group generation technique based on Minkowski sums and the combinatorial covering approach of [GD99] to attempt to maximize coverage of P. This allows it to find covers in many cases where the algorithm of [DI01b] fails (see, for example, rows 2, 10, 12 and 13 of the table and their corresponding figures). In particular, covering the convex hull of P is not a limitation.

The new heuristic is limited by the NP-hardness of the covering problem itself. For this reason it is only expected to be able to handle a small number of covering shapes in Q. Another limitation arises from the choice of triangle subdivision method. In some cases in which a cover must be tight and have small overlaps among the shapes in Q, many small triangles are generated within and near the overlaps, causing the heuristic to reach its limit on the number of triangles. Future work will design a subdivision technique that decomposes the target polygon based on properties of the covering shapes. With this improvement the heuristic is expected to be strong enough to be used as a building block in 2D covering applications.

References

[DI01a] K. Daniels and R. Inkulu. An incremental algorithm for translational polygon covering. Technical report, Dept. of Computer Science, UMass Lowell, Lowell, MA, 2001.

[DI01b] K. Daniels and R. Inkulu. Translational polygon covering using intersection graphs. In *Proceedings of the Thirteenth Canadian Conference on Computational Geometry*, page 61:64, 2001.

[FPT81] R. Fowler, M. Paterson, and L. Tanimoto. Optimal packing and covering in the plane are np-complete. *Information Processing Letters*, 12(3):133:137, 1981.

[GD99] R. Grinde and K. Daniels. Solving an apparel trim placement problem using a maximum cover problem approach. *Institute of Industrial Engineers (IIE) Transactions*, 31(8):763:769, 1999.

[JCR88] J. J. Culberson and R. Reckhow. Covering polygons is hard. In *Proceedings of 29th IEEE Conference on Foundations of Computer Science*, 1988.

[Tot97] G.F. Toth. Packing and covering. In J. Goodman and J. ORourke, editors, *Handbook of Discrete and Computational Geometry*. CRC Press, New York, 1997.