Exploring Simple Triangular and Hexagonal Grid Polygons Online

Daniel Herrmann* Tom Kamphans** Elmar Langetepe*

Abstract

We examine online grid covering with hexagonal and triangular grids. For arbitrary environments without obstacles we provide a strategy that produces tours of length $S \leq C + \frac{1}{4}E - \frac{5}{2}$ for hexagonal grids, and $S \leq C + E - 4$ for triangular grids. C is the number of cells—the area—, E is the number of boundary edges—the perimeter—of the environment. We show that our strategy is $\frac{4}{3}$-competitive, and give lower bounds of $\frac{14}{13}$ ($\frac{7}{6}$) for hexagonal (triangular) grids.

Keywords: Exploration, covering, grid graphs

1 Introduction

Exploring an unknown environment is one of the basic tasks of autonomous mobile robots. For some applications such as robots with limited vision or robots that have to visit every part of the environment (lawn mowers, cleaners), it is convenient to subdivide the given environment by a regular grid into *cells*. The robot's position is always given by the cell currently occupied by the robot. From its current position, it knows the neighboring cells and it can move to a free one. The robot's task is to visit every free cell and to return to the start.

There are three regular tilings: square, hexagonal, or triangular subdivisions. We call the subdivisions of the given environment a square polygon (hexagonal polygon, triangular polygon; respectively). Hexagonal cells are a matter of particular interest for robots that are equipped with a circular tool such as lawn mowers, because hexagonal grids provide a better approximation for the tool than square grids [1].

In a square polygon with obstacles, the offline problem (i.e., finding a minimum length tour that visits every cell) is NP-hard [13], but there are $1 + \epsilon$ approximation schemes (e.g., [3]). For square polygons there is a $\frac{53}{40}$ approximation by Arkin et al. [1].

In a square polygon without obstacles, the complexity of constructing offline a minimum length tour is still open. There are approximations with factors $\frac{4}{3}$ [14] and $\frac{6}{5}$ [1]. There is an $O(C^4)$ algorithm for deciding the existence of Hamiltonian cycles in a square grid. Hamiltonian paths were considered by Everett [6]. HCs on triangular and hexagonal grids were studied by Arkin et al. [2], and Islam et al. [12].

Our interest is in the *online* version of the cell exploration problem for *hexagonal* and *triangular* polygons. Square polygons with holes were considered by Gabriely and Rimon [7] and Icking et al. [11]. Our exploration strategy is based on the $\frac{4}{3}$-competitive strategy *SmartDFS* by Icking et al. [10], which needs at most $\#\text{Cells} + \frac{\#\text{Edges}}{2} - 3$ steps from cell to cell. Also, there is a lower bound of $\frac{7}{6}$ on the competitive factor.

Subdividing the robot's environment into grid cells is used also in the robotics community (e.g., [4]). See also the survey by Choset [5].

*University of Bonn, Institute of Computer Science I, 53117 Bonn, Germany.

**Braunschweig University of Technology, Computer Science, Algorithms Group, 38106 Braunschweig, Germany

2 Preliminaries

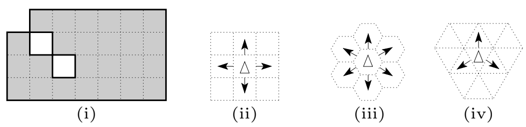

Figure 1: (i) A simple grid polygon, (ii)-(iv) neighboring (arrows) and touching cells.

We consider polygons that are subdivided by a regular grid into *cells*. A cell is *free* if it can be visited by the agent, otherwise *blocked*. We call two cells *neighboring* if they share a common edge, *touching* if they share a common corner. A *path* is a sequence of consecutively neighboring cells, a *grid polygon* is a path-connected set of free cells. A polygon without blocked cells inside its boundary is called *simple*. From its current position, the agent can find out which neighbor is free and which one is blocked, and it can move in one *step* to a free neighbor, see Fig. 1. The agent has enough memory to store a map of known cells.

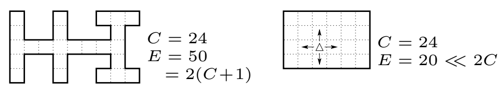

Figure 2: E distinguishes *thin* and *thick* polygons.

We analyze the performance of an online exploration strategy using the *area*, C (the number of cells), and the *perimeter*, E (the number of boundary edges). E is adequate to distinguish between thin environments that have many corridors of width 1, and thick environments that have wider areas; see Fig. 2.

3 Lower Bounds

First, our interest is in the best competitive factor we can expect for an online strategy that visits every cell and returns to the start cell.

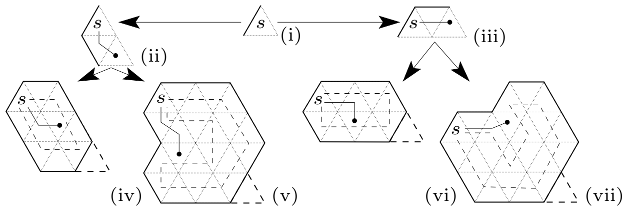

Figure 3: A lower bound on the exploration of simple triangular polygons (thin dashed: optimal solution, bold dashed: next block).

Theorem 1 *There is no online strategy for the exploration of simple triangular (hexagonal) grid polygons with a competitive factor better than* $\frac{7}{6}$ ($\frac{14}{13}$, *resp.).*

Proof. For triangles, we start in a cell with two neighbors, see Fig 3(i). If the first step is to the south, we add a cell such that the only possible step is to the southwest (3(ii)). Otherwise, we force a step to the east (3(iii)). In both cases, the agent has the choice to leave or to follow the polygon's boundary. In either case we close the polygon. In the first case, the agent needs at least 12 steps while the optimal path has 10 steps (3(iv),(vi)). In the second case, the agent needs ≥ 26 steps; the optimal path needs 22 steps (3(v), (vii)). To construct arbitrarily large polygons, we use more of these blocks and glue them together, the cell shown with bold dashed lines is the next start cell. Unfortunately, both the online strategy and the optimal path need two additional steps for the transition. For n blocks we have in the best case a ratio of $\frac{26+28(n-1)}{22+24(n-1)}$, which goes to $\frac{7}{6}$ if n goes to infinity.

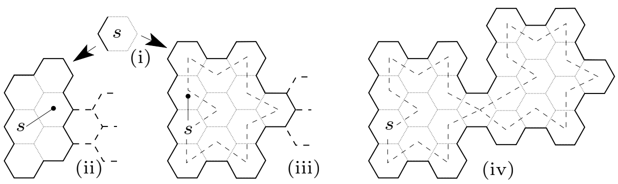

Figure 4: A lower bound for the exploration of simple polygons. The dashed lines show the optimal solution.

For hexagons, we start in a cell with four neighbors, see Fig. 4(i). The agent may leave the polygon's boundary with a step NW or SW, or follow the boundary by walking north or south. In either case we fix the polygon (Fig. 4(ii),(iii)), yielding a ratio of $\frac{7}{6}$ or $\frac{13}{12}$, respectively. The subsequent block attaches using the cell(s) shown with bold dashed lines. Again, we need one or two additional steps for the transition (4(iv)), yielding a best-case ratio of $\frac{13+14(n-1)}{12+13(n-1)} \longrightarrow \frac{14}{13}$. □

4 Exploring Simple Polygons

In this section, we briefly describe the exploration strategy *SmartDFS* [11]. As a first approach, we can apply a depth-first search (DFS): We explore the polygon following the left-hand rule; that is, for every entered cell the agent tries to continue its path to a neighboring unexplored cell in clockwise order. This results in a complete exploration, but takes $2C-2$ steps. We introduce two improvements.

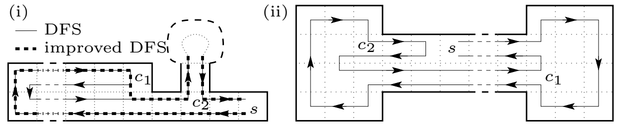

Figure 5: Improvements to DFS: (i) optimize return path, (ii) detect polygon splits.

First, we return directly to those cells that have unexplored neighbors. See Fig. 5(i): DFS walks from c_1 to c_2 through the completely explored corridor. A more efficient strategy walks on a shortest path (on known cells) from c_1 to c_2.

In Fig. 5(ii), DFS walks four times through the corridor. A more clever solution explores the right part immediately after the first visit of c_1, and continues with the left part, resulting in only two visits. When c_1 is explored, the graph of unvisited cells splits into two components. We call cells with this property *split cells*. The second improvement is to handle split cells.

SmartDFS(*P*, *start*):
  Choose *dir*, such that reverse(*dir*) is blocked;
  ExploreCell(*dir*);
  Walk on the shortest path to *start*;

ExploreCell(*dir*):
  base := current position;
  if not isSplitCell(*base*) **then**
    forall neighbors *c* of *base*, in clockwise order
      ExploreStep(*base*, direction towards *c*);
  else Choose different order, see Sect. 5.

ExploreStep(*base*, *dir*):
  if unexplored(*base*, *dir*) **then**
    Walk on shortest path to *base*;
    move(*dir*);
    ExploreCell(*dir*);
  end if

5 The Analysis of SmartDFS

In this section, we briefly analyze the performance of our strategy in hexagonal and triangular grids. See our full version [9, 8] for the complete proofs.

To show our main theorem, we need an upper bound on the length of a path inside a grid polygon.

Lemma 2 *The length of a shortest path,* Π*, between two cells in a grid polygon is bounded by*
$|\Pi| \leq \frac{1}{4}E(P) - \frac{3}{2}$ *for hexagonal polygons, and*
$|\Pi| \leq E(P) - 3$ *for triangular polygons.*

First, we give an upper bound on the number of steps needed by our strategy. The basic idea is an induction on the number of split cells, so let P^* be the part of P that is currently explored. When we meet a split cell, c, P^* divides into three parts: $P^* = K_1 \dot{\cup} K_2 \dot{\cup} \{$ visited cells of $P^* \}$, where K_1 and K_2 denote the connected components of the set of unvisited cells of P^*, see Figs. 6 and 7.

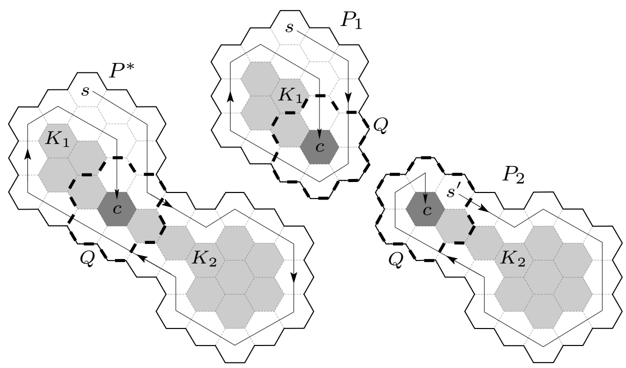

Figure 6: A split: K_1 is of type C, K_2 type A.

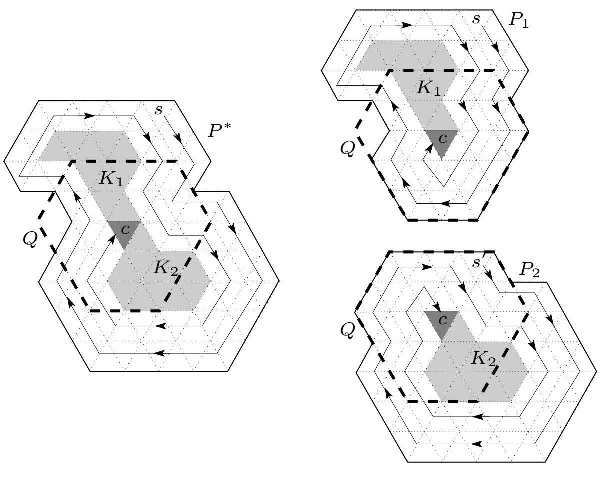

Figure 7: A split: K_1 is of type C, K_2 type B.

Now, we divide P^* into two parts, P_1 and P_2, such that each of them is an extension of the two components. Both polygons overlap in the area around the split cell c. If there is a polygon that does *not* contain s, we explore the corresponding component first, expecting that in this part the path from the last visited cell back to s is the shorter than in the other part. In the following, let K_2 denote the component that is explored first.

We can decide which component we have to visit first using the following definition:

Definition 1 *The boundary cells of* P *uniquely define the* first layer. P *without its first layer is the* 1-offset. *The* ℓ*th layer and the* ℓ*-offset are defined successively. When a split cell occurs in layer* ℓ*, every component is one of the following types:*
A. the part of layer ℓ *that surrounds* K_i *is* not *visited*
B. ... completely *visited*
C. ... partially *visited*

If a component of type C exists, it is reasonable to explore it at last. Otherwise, we proceed using the left-hand rule, but omit the first possible step [9].

Now, let Q be the polygon made of c extended by q layers, where $q := \ell$ if K_2 is of type B, and $q := \ell - 1$ if K_2 is of type A. We choose $P_2 \subset P^* \cup Q$ such that $K_2 \cup \{c\}$ is the q-offset of P_2, and $P_1 := ((P^* \backslash P_2) \cup Q) \cap P^*$. Further, we require $P^* \cup Q = P_1 \cup P_2$ and $P_1 \cap P_2 \subseteq Q$. The choice of P_1, P_2, and Q ensures that the paths in $P_1 \backslash Q$ and $P_2 \backslash Q$ do not change compared to P^*. The parts of the path that lead from P_1 to P_2 and back are fully contained in Q. Just the parts inside Q are bended to connect the appropriate paths inside P_1 and P_2; see Figs. 6, 7.

Theorem 3 *Let* P *be a simple grid polygon with* $C(P)$ *cells and* $E(P)$ *edges.* P *can be explored with*
$S(P) \leq C(P) + \frac{1}{4}E(P) - \frac{5}{2}$ *(hexagonal polygons)*
$S(P) \leq C(P) + E(P) - 4$ *(triangular polygons)*
steps. This bound is tight.

Proof. (sketch for hexagonal polygons)
We show by an induction on the number of split cells that the number of *additional* cell visits, $ex(P^*)$, is smaller than $\frac{1}{4}E(P^*) - \frac{5}{2}$.

In the induction base we have no split cell: SmartDFS visits every cell and returns to the start cell. Without a split, all cells of P can be visited by a path of length $C(P) - 1$. By Lemma 2, the shortest path back to s is smaller than $\frac{1}{4}E(P^*) - \frac{3}{2}$.

Now, let c be the first split cell in P^*. Two new components, K_1 and K_2, occur. Let P_1, P_2, and Q defined as earlier. There is no split in $P_2 \backslash (K_2 \cup \{c\})$, but c is visited twice. Thus, we have

$$ex(P^*) \leq ex(P_1) + ex(K_2 \cup \{c\}) + 1.$$

Now, we apply the induction hypothesis to P_1 and $K_2 \cup \{c\}$ and get

$$ex(P^*) \leq \frac{1}{4}E(P_1) - \frac{5}{2} + \frac{1}{4}E(K_2 \cup \{c\}) - \frac{5}{2} + 1.$$

The ℓ-offset, P', of P fulfills $E(P') \leq E(P) - 12\ell$ [9]:

$$ex(P^*) \leq \frac{1}{4}E(P_1) + \frac{1}{4}E(P_2) - 3q - 4.$$

With $E(P_1) + E(P_2) = E(P^*) + E(Q)$ and $E(Q) = 12q + 6$ we have

$$ex(P^*) \leq \frac{1}{4}E(P^*) + \frac{3}{2} - 4 = \frac{1}{4}E(P^*) - \frac{5}{2}\,.$$

It is easy to see that this bound is exactly achieved in corridors of width 1. The exploration of such a corridor needs $2(C(P) - 1)$ steps. On the other hand, the number of edges is $E(P) = 4C(P) + 2$. □

Theorem 4 *The strategy SmartDFS is $\frac{4}{3}$-competitive in hexagonal and triangular polygons. This factor is tight.*

Proof. (sketch for hexagonal polygons)
Corridors of width 1 or 2 play a crucial role in the following, so we refer to them as *narrow passages*. More precisely, a cell, c, belongs to a narrow passage, if c can be removed without changing the layer number of any other cell. It is easy to see that narrow passages are explored optimally: In corridors of width 1 both SmartDFS and the optimal strategy visit every cell twice, and in the other case both strategies visit every cell exactly once. Thus, we can assume that P is a polygon without narrow passages. If P has no split cell in the first layer, we can show that $E(P) \leq \frac{4}{3}\,C(P) + \frac{26}{3}$ holds ($E(P) \leq \frac{1}{3}\,C(P) + \frac{14}{3}$ for triangular polygons). Further, in these type of polygons SmartDFS needs two steps fewer than shown in Theorem 3.

Now, we show by induction on the number of split cells in the first layer that $S(P) \leq \frac{4}{3}C(P) - \frac{7}{3}$ holds.

For the induction base we can apply the observations from above: $S(P) \leq C(P) + \frac{1}{4}\,E(P) - \frac{9}{2} \leq C(P) + \frac{1}{4}\left(\frac{4}{3}\,C(P) + \frac{26}{3}\right) - \frac{9}{2} = \frac{4}{3}\,C(P) - \frac{7}{3}$.

Two cases occur if we meet a split cell, c, in the first layer: Either the new component was never visited before (type A), or we meet a visited cell, c', that touches the current cell (type B). In the first case let $Q := \{c\}$, in the second case $Q := \{c, c'\}$.

Similar to the proof of Theorem 3, we split the polygon P into two parts, both including Q. Let P'' denote the part that includes the component of type A or B, P' the other part. For $|Q| = 1$ we conclude $S(P) = S(P') + S(P'')$ and $C(P) = C(P') + C(P'') - 1$. Applying the induction hypothesis to P' and P'' yields $S(P) = S(P') + S(P'') < \frac{4}{3}\,C(P) - \frac{7}{3}$.

For $|Q| = 2$ we gain some steps by merging the polygons. If we consider P' and P'' separately, we count the steps from c' to c—or vice versa—in both polygons, but in P the path from c' to c is replaced by the exploration path in P''. Thus, we have $S(P) = S(P') + S(P'') - 2$ and $C(P) = C(P') + C(P'') - 2$. Applying the induction hypothesis yields our claim.

OPT needs at least C steps, which, altogether, yields a competitive factor of $\frac{4}{3}$. This factor is achieved in a corridor of width 3, see Fig. 8. $\square$

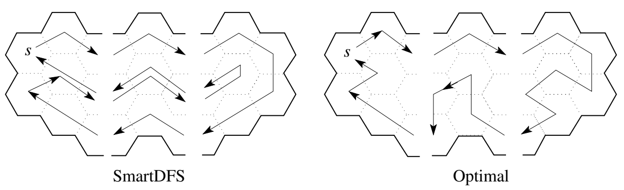

Figure 8: $S(P) = \frac{4}{3}S_{\text{Opt}}(P) - \frac{7}{3}$ holds.

6 Summary

We considered the online exploration of hexagonal and triangular grid polygons with SmartDFS.

For hexagonal polygons we gave a lower bound of $\frac{14}{13}$ and showed that SmartDFS explores polygons with C cells and E edges using no more than $C + \frac{1}{4}E - \frac{5}{2}$ steps. For triangular polygons we have a lower bound of $\frac{7}{6}$ (matching the lower bound for square polygons) and an upper bound of $C + E - 4$ on the number of steps. Further, we showed that both strategies are $\frac{4}{3}$-competitive. An interesting open problem is how to close the gap between the upper bound and the lower bound on the competitive factors.

References

[1] E. M. Arkin, S. P. Fekete, and J. S. B. Mitchell. Approximation algorithms for lawn mowing and milling. *Comput. Geom. Theory Appl.*, 17:25–50, 2000.

[2] E. M. Arkin, J. S. B. Mitchell, and V. Polishchuk. Two new classes of hamiltonian graphs. In *Proc. Europ. Conf. Comb. Graph Theo. Appl.*, volume 29C of *e-Notes Discr. Math*, pages 565–569, 2007.

[3] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 2–11, 1996.

[4] A. M. Bruckstein, M. Lindenbaum, and I. A. Wagner. Mac vs. pc - determinism and randomness as complementary approaches to robotic exploration of continuous unknown domains. *Internat. J. Robotics Res.*, 19(1):12–31, 2000.

[5] H. Choset. Coverage for robotics - A survey of recent results. *Ann. Math. Artif. Intell.*, 31:113–126, 2001.

[6] H. Everett. Hamiltonian paths in non-rectangular grid graphs. Report 86-1, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, 1986.

[7] Y. Gabriely and E. Rimon. Competitive on-line coverage of grid environments by a mobile robot. *Comput. Geom. Theory Appl.*, 24:197–224, 2003.

[8] D. Herrmann. Exploration von Dreiecks– und Sechsecksgittern. Diplomarbeit, University of Bonn, January 2008.

[9] D. Herrmann, T. Kamphans, and E. Langetepe. Exploring simple triangular and hexagonal grid polygons online. Technical Report 007, Department of Computer Science I, University of Bonn, 2007. http://web.informatik.uni-bonn.de/I/publications/hkl-esthg-07.pdf.

[10] C. Icking, T. Kamphans, R. Klein, and E. Langetepe. Exploring grid polygons online. Technical Report 001, Department of Computer Science I, University of Bonn, December 2005. http://web.informatik.uni-bonn.de/I/publications/ikkl-egpol-05.pdf.

[11] C. Icking, T. Kamphans, R. Klein, and E. Langetepe. Exploring simple grid polygons. In *11th Internat. Comput. Combin. Conf.*, volume 3595 of *Lecture Notes Comput. Sci.*, pages 524–533. Springer, 2005.

[12] K. Islam, H. Meijer, Y. N. Rodríguez, D. Rappaport, and H. Xiao. Hamilton circuits in hexagonal grid graphs. In *Proc. 19th Canad. Conf. Comput. Geom.*, pages 85–88, 2007.

[13] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter. Hamilton paths in grid graphs. *SIAM J. Comput.*, 11:676–686, 1982.

[14] S. Ntafos. Watchman routes under limited visibility. *Comput. Geom. Theory Appl.*, 1(3):149–170, 1992.