

Unit Tilt Row Relocation in a Square

David Caballero¹, Angel A. Cantu², Timothy Gomez¹, Austin Luchsinger³, Robert Schweller¹, and Tim Wylie¹

¹Department of Computer Science, University of Texas Rio Grande Valley, Edinburg, TX USA

²Southwest Research Institute, San Antonio, TX USA

³Electrical and Computer Engineering Department, University of Texas Austin, Austin, TX USA

Abstract

Relocation in the square is the most basic open question related to the tilt model with unit moves under global uniform movement. It is open even with limited directions. We prove that deciding if a piece can move to the bottom row with only two directions is in P and gives the minimum column.

Keywords: tilt model, robot motion planning, relocation, uniform control, external forces.

2010 MSC: Primary 52C99; Secondary 68R05, 52B20, 68W01.

1 Introduction

Tilt assembly with uniform global instructions is currently a popular area of research since individually controlling nanobots may be infeasible or impractical at certain scales. One of the most basic open questions in the single step tilt model with uniform global movements is whether a tile can be relocated to specific location on the board (Figure 1a) even without any blocking geometry. We solve a simpler version of this problem asking whether, with only two directions, a tile can reach the bottom row.

Given a tilt system with an $n \times n$ board $\mathbb{B}$, a set of tiles $\mathbb{T} = \{t_1, \dots, t_m\}$ where each $t_l = (i, j)$ s.t. $1 \leq l \leq m$, $1 \leq i \leq n$, and $1 \leq j \leq n$. Given $t_i, t_j \in \mathbb{T}$, $t_i \neq t_j$ if $i \neq j$. For shorthand, we use $t_{i,j}$ for $t_l = (i, j)$. For just the row or column, we may use t_{l_r} and t_{l_c} . See [1–3] for model and related work.

Definition 1.1 (First Row Relocation). Given a specific tile to relocate t_R at location $(r, c) = (t_{R_r}, t_{R_c})$, the first row relocation problem asks whether a series of transformations can translate t_R s.t. $t_{R_r} = 1$.

2 High Level Overview

In order for tile t_R to reach the bottom row, there must not be any other tiles beneath it. We first find the empty column needed, and then, through a process termed *knitting*, we attempt to move the empty column beneath t_R . We also label the board with sections (Figure 1a). See Figure 1b for examples of the empty column. We then label the rows with *counts* of tiles that may affect relocation (Figure 1c).

Definition 2.1 (Empty Column E_c). Given a tilt system board configuration $\mathbb{S} = (\mathbb{B}, \mathbb{T})$ and tile $t_R = (r, c)$ where $t_R \in \mathbb{T}$, define $E_c = \min\{k : c \leq k \leq n + 1 \text{ s.t. } |\{t_{i,j} : t_{i,j} \in \mathbb{T}, i < r, j = k\}| = 0\}$. If all columns in the BR section have tiles, $E_c = n + 1$ and enters the board after a west, $\langle W \rangle$, movement.

Definition 2.2 (Knitting). The row between the BL and TL section is the knitting row. Knitting is the act of performing $\langle W \rangle$ movements when every position of the knitting area is occupied by a tile. Thus, t_R maintains its position.

Definition 2.3 (Counts). Given a tilt system board configuration $\mathbb{S} = (\mathbb{B}, \mathbb{T})$, a tile $t_R \in \mathbb{T}$ with location (r, c) , and the target empty column E_c . Define the count of a row k as follows.

$$Count(k, t_{r,c}) = \begin{cases} |\{t_{i,j} : t_{i,j} \in \mathbb{T}, i = k, 1 \leq j < c\}|, & \text{if } k > r \text{ (rows above } t_R) \\ |\{t_{i,j} : t_{i,j} \in \mathbb{T}, i = k, 1 \leq j \leq c\}|, & \text{if } k = r \text{ (} t_R \text{ row),} \\ |\{t_{i,j} : t_{i,j} \in \mathbb{T}, i = k, 1 \leq j < E_c\}|, & \text{if } k < r \text{ (rows below } t_R) \end{cases}$$

This basic framework leads to two important lemmas (2.5 2.6). The algorithm will ensure that one of these is eventually met. The basic idea is Algorithm 1.

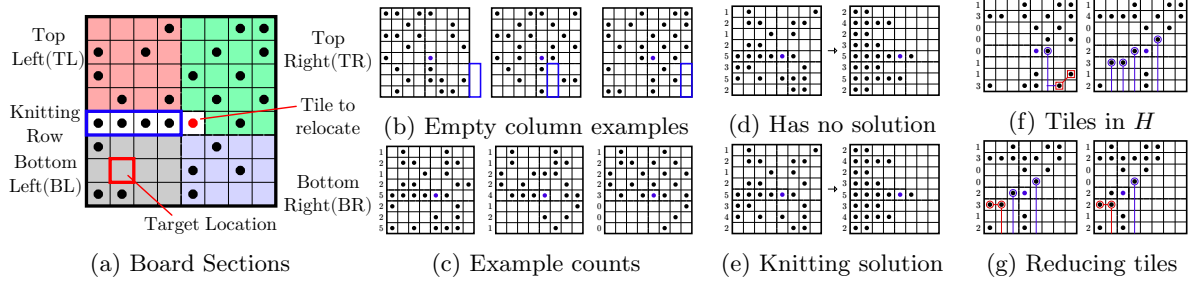


Figure 1: (a) The board is an $n \times n$ square divided into 6 sections based on the tile to relocate. The four large sections are the Top Left (TL), Top Right (TR), Bottom Left (BL), and Bottom Right (BR) sections. There is also the knitting area (outlined in blue) and the tile to relocate (red dot). When we move to general relocation, there will also be a target spot (red square). (f) Tiles that enter the BR section after D movements are included in N_R . If the empty column is occupied by a tile after D movement (i.e., it becomes non-empty), then every tile between the empty column and the next empty column (shown in red) are also added to N_R . Reducing the number of tiles in H is done by $\langle W \rangle$ movement. (g) Tiles in H that lead towards another tile beneath it are removed from H .

2.1 Changing the Knitting Row

If neither condition in the lemmas is satisfied, we cannot knit with the tiles in the knitting row. Define the *candidate row* as the closest row in the TL section with a count higher than the knitting row.

Definition 2.4 (Candidate Row). Given a tilt system board configuration $\mathbb{S} = (\mathbb{B}, \mathbb{T})$ and a tile $t_R \in \mathbb{T}$ at location (r, c) . The knitting row may contain up to $c - 1$ tiles. The closest row (fewest south, $\langle S \rangle$, movements) in the TL section with a count higher than the knitting row is the candidate row.

Lemma 2.5. [Existence] If the count of the knitting area is greater than the counts of the lower section, then first row relocation is possible.

Lemma 2.6. [Nonexistence] If there exist a count in the bottom sections that is larger than every count in the TL section and knitting area, then first row relocation is impossible.

Result: First Row Relocation

while Lemmas 2.5, 2.6 are unsatisfied **do**
 Determine E_c and counts for each row;
 if Lemma 2.5 is satisfied **then** accept;
 if Lemma 2.6 is satisfied **then** reject;
 Set candidate row CR ;
 Move CR to be the new knitting row;
Algorithm 1: High Level Idea

If we make any movements, we may introduce new tiles that must be considered. We look at new tiles that may enter the BL section (N_L) and tiles that may enter in the BR section (N_R).

We include the set of tiles in TR that are at most D distance (with $\langle S \rangle$ moves) from the BR that increase the counts of the bottom rows (Figure 1f). The BR and TR sections may change if t_R moves. If E_c has tiles after D movements, then we change E_c to the correct column. All tiles that enter the last row after D movements to the left of the new empty column are included in N_R . Let H be the set of all tiles in either the TL or BL section that are within D $\langle S \rangle$ movements from the first row. With $\langle W \rangle$ moves, we may be able to reduce H (Figure 1g). When first row relocation is possible, the sequence is $W^u S^v W^x S^y$ and u, v, x, y must be determined, but are bounded by n . There are $\leq c - 2$ possible candidate rows, and at most $n - r$ moves to bring any CR to the knitting row.

Theorem 2.7. First row relocation of tile $t_{r,c} \in \mathbb{T}$ on an $n \times n$ board can be solved in $\mathcal{O}(cn + c|\mathbb{T}|)$ time.

Funding. This research was supported in part by National Science Foundation Grant CCF-1817602.

References

- [1] David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert Schweller, and Tim Wylie, *Building patterned shapes in robot swarms with uniform control signals*, Proceedings of the 32nd Canadian Conference on Computational Geometry, CCCG'20, 2020, pp. 59–62.
- [2] ———, *Hardness of reconfiguring robot swarms with uniform external control in limited directions*, Journal of Information Processing (2020).
- [3] ———, *Relocating units in robot swarms with uniform control signals is pspace-complete*, Proceedings of the 32nd Canadian Conference on Computational Geometry, CCCG'20, 2020, pp. 49–55.