Computational topology - group project

The Evasion Problem

The evasion problem asks if there is a way for an intruder to escape detection in an area covered with moving sensors. We will assume that the area X is a subset of the plane $\mathbb{R}^2$ and that the movement of the sensors is periodic. This means that there exists some time p such that the positions and directions of movement of the sensors at time $t + p$ are identical to the situation at time t for all times $t \in \mathbb{R}$. If we combine the snapshots at each point in time we get a three-dimensional object $X \times \mathbb{R}$. Because the movement is periodic, we can limit ourselves to the times $[0, p]$ and consider $X \times [0, p]$ instead with the assumption that the slices $X \times \{0\}$ and $X \times \{p\}$ are glued together.

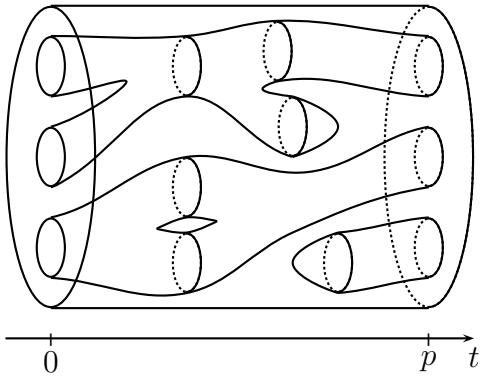

Figure 1: With passing time the sensors move around and covered areas can merge or split apart. If the movement of the sensors is periodic, the situation will repeat from the left side after it reaches the right side as if the left and the right were glued together.

We will simplify the situation by assuming that:

- the room is a rectangle $a \times b$ units in size,
- the sensors move along the given straight-line segments (rails) at a constant speed, turning around instantly when they reach the end,
- all sensors move at the same speed of one unit of distance per one unit of time,
- each sensor covers a square area two units of distance across oriented parallel to the edges of the room they protect.

Therefore, the behaviour of the system is completely determined by the initial situation. At each integer time $t \in \{0, 1, \ldots, p - 1\}$ the area covered by the sensors is the union of all the unit squares that have one of the sensors at one of the vertices.

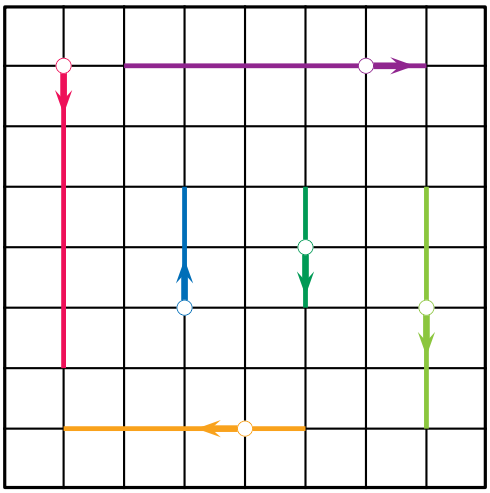

Figure 2: A possible initial configuration. This system will return to the initial configuration after 40 time steps.

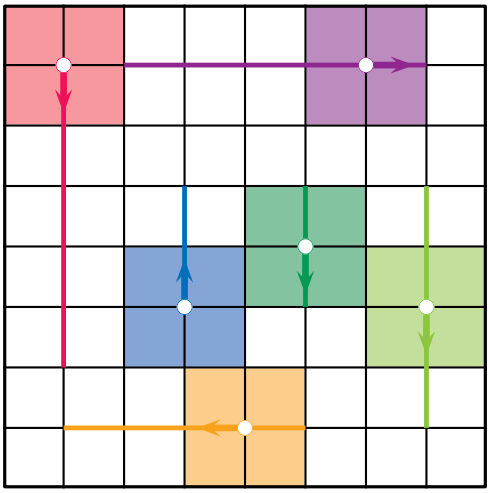

Figure 3: Covered area corresponding to the given initial configuration.

Project goal: Given the initial state of the system:

- determine the period p,
- construct the planar slices $X \times \{t\}$ for all $t \in \{0, 1, \ldots, p-1\}$,
- construct the complex $X \times [0, p]$ by connecting the slices from the previous step and subdividing as necessary,
- construct the observed (covered) complex C and the unobserved (free) subcomplex $F = (X \times [0, p]) \setminus C$,

- compute $H_1(F, \mathbb{Z})$ and determine which generators of $H_1(F, \mathbb{Z})$ represent paths a thief can take to avoid detection.

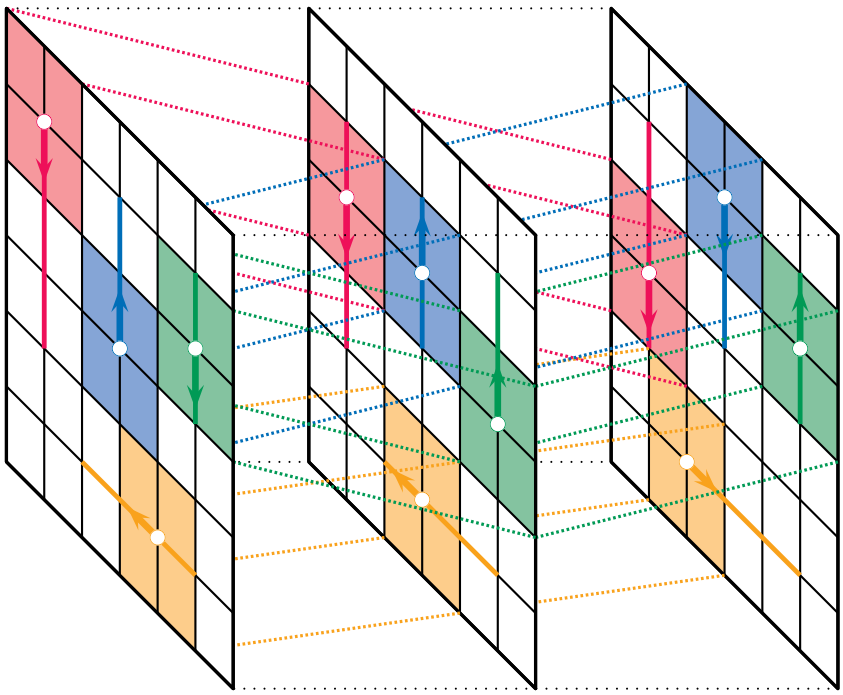

Figure 4: Changes after one and two time units. Between each consecutive pair of discrete snapshots the covered area changes continuously. This creates prisms that combine to form the covered complex C. The complement $F = (X \times [0, p]) \setminus C$ is the free space where the thief can move without being noticed. If the thief ends up in a wedge-shaped piece, he will get caught, so you can simplify the structure by considering the subspace of F consisting of cubes. In this case you might consider using cubical homology instead of subdividing the cubical complex into a simplicial complex.

Simplifications: You can begin with the examples where the period p and the room size $a \times b$ are small, so that the number of simplices in $X \times [0, p]$ remains small as well. You can further simplify the situation by positioning the rails so that the cover areas never intersect or even touch, but in this case the topology of the free space is very uninteresting, so you will definitely want to drop these restrictions later. Finally, start with a small number of sensors that move in such a way that the solution to avoiding them is obvious and test your algorithm with those examples first. Does it work as expected?

Bonus goal 1: Create a room generator that creates a room of random size with some randomly placed rails and initial positions/orientations of the sensors. Then use the generated rooms to test your free-path finding algorithm. Can you make sure that every point in the room is covered by at least one sensor at least part of the time? How can homology help in detecting this?

Bonus goal 2: Create an animation that shows how the sensors move. If your algorithm finds a path along which the thief can avoid detection, simulate the thief moving along this path.