Designing the game "Lasers!"

Sergio Ricossa

August 13, 2021

Contents

1 The Problem

You are targeted by a laser in a rectangular room with four mirror-walls. The laser can bounce off the mirrors. You have up to sixteen point-like obstacles that will block the laser. Your goal is to find a configuration in which the laser can no longer hit you.

Both you and the shooter are point-like objects, and the laser can go through the shooter. If the laser hits a corner of the room, it bounces right back 180deg.

2 The Solution

The idea is to consider the equivalent problem of an infinite square lattice, made of infinitely many copies of the original rectangular room, with a target point in each. The laser is always a straight line from the shooter to any of the targets on the lattice.

When an obstacle is placed, infinitely many copies are placed in the lattice. The laser has to avoid all of them.

We will denote by l and L the width and length of the room, respectively.

$\boldsymbol{x_0} = (x_0, y_0)$, is the position of the shooter and, $\boldsymbol{x_1} = (x_1, y_1)$, that of the target.

Because of the mirrors, our lattice has to be divided into four sub-lattices. One of them is made of copies of the original room. Another one is made of copies flipped upside down, yet another one is flipped left to right. The last one is flipped both left to right and upside down. They all have periodicity $(2l, 2L)$.

These can be encoded by two bits, $s_x, s_y \in \{+1, -1\}$, which are negative only if the lattice is flipped along the specified axis.

The positions of the targets are labelled by a choice of $\boldsymbol{s} = (s_x, s_y)$, as well as two integers, $\boldsymbol{n} = (n_x, n_y) \in \mathbb{Z}$, that denote the position in the chosen sub-lattice:

$$\boldsymbol{x}_1^{\boldsymbol{s},\boldsymbol{n}} = (2ln_x + s_x x_1, 2Ln_y + s_y y_1) \tag{1}$$

A valid laser trajectory goes from $\boldsymbol{x}_0$ to any of the $\boldsymbol{x}_1^{\boldsymbol{s},\boldsymbol{n}}$. We can write it as,

$$\boldsymbol{x}_{\text{laser}}(\boldsymbol{s}, \boldsymbol{n}; t) = t\boldsymbol{x}_1^{\boldsymbol{s},\boldsymbol{n}} + (1-t)\boldsymbol{x}_0. \tag{2}$$

Thanks to the peculiar $(2l, 2L)$ periodicity, it is possible to block every trajectory with $\boldsymbol{s}$ fixed with only 4 obstacles. In addition, this solution is unique. Since there are 4 values of $\boldsymbol{s}$, at most $4 \times 4 = 16$ obstacles are needed to secure every target on the lattice.

We show this now. Two points of the lattice are equivelent if they are equal to each other $\mod (2l, 2L)$. This criterion is sufficient, but not necessary, since it only checks equivalence on the same sub-lattice.

Taking $\boldsymbol{x}_{\text{laser}}(\boldsymbol{s}, \boldsymbol{n}; t) \mod (2l, 2L)$, we look for a value of t that yields values from a finite set of points, independent of $\boldsymbol{n}$. This is equivalent to asking which values of t are such that $\{2ln_x t \mod 2l | n_x \in \mathbb{Z}\}$ and $\{2Ln_y t \mod 2L | n_y \in \mathbb{Z}\}$ are both finite.

This is only possible if t is a rational number. And the minimal number of obstacles is obtained when $t = 0$ and $t = 1$, but that is obviously not permitted by the problem. It would amount to placing one obstacle on the shooter, or the target, and calling it done.

Instead, the next best option is $t = 1/2$, for which we get 4 points, depending on whether n_x and n_y are even or odd.

Two or more of these points could still be equivalent. But we can just check for all 4 points (16, counting all choices of $\boldsymbol{s}$) to see if they are the same or different. They are given by the formula,

$$\boldsymbol{x}_{\text{solution}}(\boldsymbol{s}, \boldsymbol{s}') = \left(s_x' \frac{x_0 + s_x x_1}{2} \mod l, s_y' \frac{y_0 + s_y y_1}{2} \mod L \right), \tag{3}$$

where we introduced a new set of bits, $\boldsymbol{s}' = (s'_x, s'_y)$. $s'_x = 1$ if n_x is even, and $s'_x = -1$ if n_x is odd. The same goes for s'_y and n_y.

The way we check that there are no other solutions with 16 obstacles or less is by noticing that $\boldsymbol{s}'$ corresponds to the sub-lattice where the midpoint $\boldsymbol{x}(\boldsymbol{s}, \boldsymbol{n}; 1/2)$ lies.

Indeed, increasing either n_x or n_y by 2 moves the midpoint $2l$ or $2L$ ahead on the lattice, which means it stays in the same sub-lattice. If n_x and n_y are set as either even or odd, then they can only be changed by a multiple of two. And setting $\boldsymbol{s}'$ amounts to the same constraint.

For two or more points to be equivalent on a *sub-lattice*, it is in fact necessary that they be congruent $\mod (2l, 2L)$. Thus, each of the points in eq.3 is the only one that covers all laser paths, if the target is on the sub-lattice $\boldsymbol{s}$, and the midpoint is on the sub-lattice $\boldsymbol{s}'$.

Programmatically, we can encode $\boldsymbol{s}$ and $\boldsymbol{s}'$ inside a single 4 bit number. There are also a few (quite litteral) edge cases that need to be checked:

```
std::set<std::pair<int,int>> solution;

for(unsigned i{0} ; i < 16 ; i++)
{
    int x{( (i&1 ? 1 : -1) * (x0 + (i&2 ? 1 : -1)*x1 )/2 )
        % m_width};
    int y{( (i&4 ? 1 : -1) * (y0 + (i&8 ? 1 : -1)*y1 )/2 )
        % m_height};
    if(x < 0) x = m_width + x;
    else if(x == 0) x = i&1 ? 0 : m_width;
    if(y < 0) y = m_height + y;
    else if(y == 0) y = i&4 ? 0 : m_height;
    solution.insert(std::pair<int,int>{x,y});
}
```

3 Implementation details