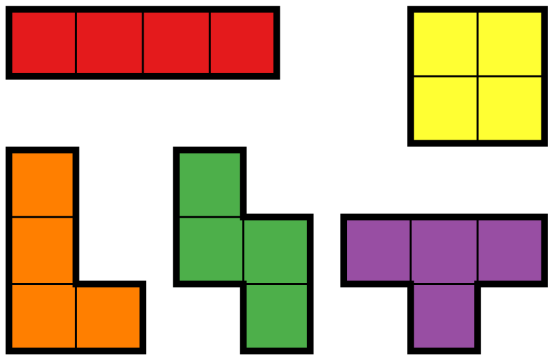
INF219 – COWI Research



Jakob Sverre Alexandersen
Thone Anlaug Stordal Støa
Hanna Søndenaa Rasmussen
Håvard Kattem Lunde
Hermann Holstad Walaunet

April 2025

GitHub Repository

Contents

1 Summary

Click me: for the full repo

This project considers techniques for placing light fixtures in any given room, minimizing the amount of lights used whilst still being compliant with regulatory and aesthetic requirements.

There are many different approaches to such a challenge. For instance, COWI presented a genetic algorithm/reinforcement learning ML-based solution. Their approach entailed maximizing coverage area while maintaining a pattern by penalizing variations in x/y coordinates.

We chose to take a different approach, using Linear Programming instead. The main reason behind this choice is the guaranteed optimality involved in such a solution. Using a solver, we are able to find the single most mathematically optimal solution that satisfies all constraints, given that a solution exists.

In addition to the optimality, we are also presented with several other advantages using the LP approach. The deterministic nature of our solution ensures that given the same inputs and constraints, we always get identical, reproducible results — eliminating the randomness inherent in genetic algorithms. This consistency is particularly valuable in professional settings where repeatability is essential.

Our LP formulation also offers greater transparency and explainability, making it clear exactly what we're optimizing for and what constraints we're satisfying. For typical room dimensions, the LP solver finds optimal solutions relatively quickly, without requiring the multiple generations of evolution that genetic algorithms need. This efficiency translates to faster design iterations and project delivery. The precise constraint handling in our approach allows for explicit modeling of critical requirements like minimum light levels and fixture spacing. Rather than approximating these through penalty functions as in the genetic algorithm approach, we incorporate them directly into the mathematical model. Finally, our solution offers straightforward adaptability to changing requirements. The LP model parameters can be easily adjusted to meet different regulatory standards or design preferences without needing to retrain an entire ML model. While the ML-based approach might offer advantages in terms of continuous learning from historical data, our LP solution provides immediate, verifiable results with mathematical guarantees—a critical factor when dealing with safety-related regulatory compliance in building design. This becomes a bigger deal if we choose to also incorporate positioning of a sprinkler system.

2 Motivation and Introduction

The goal of this project is to optimize the placement of light sources within a room in order to achieve the best possible light coverage, while ensuring that the lighting stays within a defined range of intensity (measured in LUX). Rather than maximizing the number of lights, the focus is on maximizing the illuminated area and ensuring an even distribution of light, avoiding both underlit and overlit zones.

Each light source is modeled as a red dot, and it produces a distribution of light visualized using color gradients. Brighter areas are shown in green, indicating stronger illumination, while darker areas appear more blue, representing lower light intensity. Corners and edge zones, where lighting is less critical, are deprioritized in the optimization process, allowing for a more efficient allocation of light coverage in central and more important areas.

This project was conducted in collaboration with the engineering consultancy firm COWI, and guided academically by out course supervisor, Dag. We are a team of five students with diverse academic backgrounds. Throughout the spring semester, we have worked closely together, holding regular internal meetings as well as joint sessions with COWI and our academic advisor, to define requirements, develop solutions, and refine our approach.

3 Mathematics

In this section, we will discuss the mathematical model behind the program.

3.1 Sets and Indices

$$G = \{(x, y) \mid (x, y) \in \mathbb{R}^2\} \quad \text{(Grid points for potential circle placement)} \tag{1}$$
$$A = \{(x, y) \mid (x, y) \in \mathbb{R}^2\} \quad \text{(Area cells for measuring coverage)} \tag{2}$$

3.2 Decision Variables

$$z_{x,y} \in \{0, 1\} \quad \forall (x, y) \in G \quad \text{(Binary variable for circle placement)} \tag{3}$$
$$c_{x,y} \geq 0 \quad \forall (x, y) \in A \quad \text{(Continuous variable for cell light level)} \tag{4}$$

3.3 Parameters

$$\alpha_{(x_g,y_g),(x_a,y_a)} \geq 0 \quad \text{(Coverage value from grid point to area cell)} \tag{5}$$
$$L_{\text{min}} \geq 0 \quad \text{(Minimum required average light level)} \tag{6}$$
$$S_{\text{min}} \geq 0 \quad \text{(Minimum required spacing between circles)} \tag{7}$$
$$A_g \subset A \quad \text{(Set of area cells within grid cell } g \in G\text{)} \tag{8}$$

3.4 Objective Function

$$\min \sum_{(x,y) \in G} z_{x,y} \quad \text{(Minimize total number of circles)} \tag{9}$$

3.5 Constraints

$$c_{x_a,y_a} = \sum_{(x_g,y_g) \in G} \alpha_{(x_g,y_g),(x_a,y_a)} \cdot z_{x_g,y_g} \quad \forall (x_a, y_a) \in A \tag{10}$$

$$\sum_{(x_a,y_a) \in A_g} c_{x_a,y_a} \geq L_{\text{min}} \cdot |A_g| \quad \forall g \in G \text{ where } |A_g| > 0 \tag{11}$$

$$z_{x_1,y_1} + z_{x_2,y_2} \leq 1 \quad \forall (x_1, y_1), (x_2, y_2) \in G \text{ where } \max\left(\frac{|x_1 - x_2|}{\text{grid\_size}}, \frac{|y_1 - y_2|}{\text{grid\_size}}\right) \leq S_{\text{min}} \tag{12}$$

3.6 Constraints explainations

10. Defines how the light level for each area cell is calculated
11. Ensures that each grid cell receives adequate light
12. Prevents lights from being placed too close to each other
13. TODO add neigh constraint

4 Execution

In this section we will shine some light on how we chose to solve the main problem – finding the optimal solution via a generic algorithm for placing light fixtures in any room of any size.

4.1 Method

As previously mentioned, a genetic algorithm/RL approach was presented to us. Although this can be a good solution in some cases, we found out that this method does not provide heuristics that are sufficiently effective or efficient for approximating the optimal solution.

Our approach to optimizing light fixture placement employs linear programming (LP), a powerful mathematical optimization technique that allows us to find the exact minimum number of fixtures required while satisfying all regulatory and aesthetic constraints.

The problem is formulated as a binary integer linear program where each potential fixture location is represented by a binary decision variable. The objective function minimizes the total number of fixtures, subject to a set of constraints that ensure adequate light coverage and appropriate spacing between fixtures.

The room is first resized into a smaller polygon. The reason for this is so that we can avoid considering not placing lights near the walls. This is a relatively easy and efficient way of minimizing computational cost as we don't need to define this in the constraints. Next up, we discretize the room into a grid, where each grid point represents a potential fixture location. A finer grid of area cells is then used to evaluate light coverage with precision. For each area cell, we calculate the light contribution from all potential fixture locations, creating a coverage model that accounts for light falloff with distance.

Two primary constraint types govern our solution; coverage constraints ensure that each area maintains at least the minimum required light level (lux) specified by regulations, while spacing constraints enforce a minimum distance between fixtures to create aesthetically pleasing and practically functional layouts.

Unlike heuristic approaches such as genetic algorithms that approximate solutions, our LP formulation guarantees finding the mathematically optimal solution whenever one exists. This mathematical certainty is particularly valuable in contexts where regulatory compliance must be definitively demonstrated.

The model's flexibility allows for straightforward parameter adjustments to accomodate different regulatory standards or design preferences. By simply modifying the minimum lux parameter or the fixture spacing requirement, the model can be adapted to various building types and usage scenarios without requiring findamental changes to the underlying optimization approach.

Implementation is achieved using the PuLP library in Python, which interfaces with powerful LP solvers capable of handling the binary integer programming challenges in the problem space.

4.2 What we tried:

4.2.1 Pattern and symmetry

One thing we spent some time exploring was the idea of creating a pattern in the chaos. The first challenge was defining what a "nice" pattern actually means. There are many ways to place lights that can look good and maintain a cohesive feel in a room, but putting that into a concrete definition is difficult.

We started experimenting with a grid-like structure of rows and columns, where the goal was to optimize both the spacing between light sources and their distance from the walls. This approach turned out to be very time-consuming and limiting. It excluded a variety of potentially good patterns, some of which might be better suited to certain types of rooms—and didn't necessarily optimize the number of light sources used.

During this process, we also tested a constraint where no two light sources could be placed directly next to each other. Interestingly, this often led to patterns that felt more balanced and comfortable, even without a precise definition of what we were aiming for.

Although this idea didn't make it into the final implementation, it helped us better understand how certain constraints can naturally lead to more visually appealing layouts.

4.2.2 Light distribution

Another area we spent some time on was distribution function for light. With so many different types of lamps, designing a distribution function that accurately reflects real-world lighting is a complex task. A fully accurate model would require a lot of computation and a many constraints to adapt to all possible light sources.

In the end, we chose a simple light distribution function that gives a rough idea of how light spreads and how much of a room is covered. While not perfectly precise, it's effective enough for visualizing light placement and coverage, and it allowed us to focus on other core aspects of the system.

4.2.3 Movable light sources

We also attempted to implement the ability to move light sources within the GUI after the optimization phase. The idea was to give users more control, allowing them to adjust placements manually if needed.

However, this feature turned out to be more complex than expected. It involved not just updating the visual placement of the lights, but also syncing those changes with the backend data and ensuring consistency with the optimized layout. Implementing this properly would have required significant frontend development, which we hadn't prioritized for this version of the project.

Because of the time constraints and complexity, we decided to shift our focus to improving the back-end functionality. This is a feature we believe has potential and could be explored further in future development.

5 Discussion

5.1 Potential Improvements

At the first meeting with COWI, we were given a short explanation of the problem, as well as insight into what they had tried prior to our involvement. They attempted to implement a genetic algorithm to find a solution, without success.

In our project, we chose to model the problem using linear programming (LP), which offered a transparent, structured, and computationally efficient method to reach an optimal solution under given constraints. The only problem was crafting these constraints in a way that would yield a feasible solution. Defining whether something is optimal or not is a subjective matter, dependent on . While LP proved to be effective for the problem at hand, there are several areas where our approach, and the overall process, could be improved or extended.

5.1.1 Clearity

Early in the project, we realized that more detailed input from the company regarding practical constraints and priorities would have helped in developing a more realistic model. Certain business rules or preferences may not have been fully captured by the LP formulation, due to either simplifications made on our side or limitations in the initial communication. For future projects, closer collaboration in the problem-formulation phase could enhance model relevance and accuracy.

5.1.2 Caching

As the model was supplemented with more constraints, sets, and parameters, we noticed that the solver's performance declined. For further improvements, we suggest implementing a caching mechanism. This could reduce redundant computations and significantly improve responsiveness, particularly when working with larger datasets or when users repeatedly test different scenarios. This is especially important for deploying the tool in a real-world environment where several users may interact with it simultaneously.

5.1.3 Scalability

While our model worked well for the current dataset and problem size, we identified potential issues with scalability. Linear programming solvers can handle reasonably large problems, but as the number of decision variables and constraints grows, the performance and clarity of the solution may degrade. For very large-scale instances, alternative approaches such as decomposition methods or heuristics/metaheuristics might be needed to find near-optimal solutions within a reasonable timeframe.

6 Conclusion

In this project, we have explored two approaches to lighting optimization using the PuLP library, a mixed-integer programming solver. The first model uses a simple mathematical grid patterning approach that places lights based on fixed spacing rules, independent of actual lighting values. This results in visually good, regular arrangements, but lacks adaptability to lighting requirements.

The second model is a constraint-based optimization solution that mathematically ensures compliance with lighting thresholds, fixture spacing, and other constraints. This model provides precise results and guarantees that the lighting meets the minimum lux requirement throughout the room. However, the output can sometimes appear less pleasing, especially in irregular spaces, as the focus is purely on optimization rather than visual symmetry.

Through our experiments, we observed that the optimization model performs particularly well when the required lighting level is higher, such as 300 lux or more. In such cases, the denser fixture placement leaves more room for the solver to balance aesthetics and coverage. Conversely, at low lux requirements, the model tends to use fewer lights and may produce more irregular patterns.

The project was carried out through structured collaboration, with group meetings and division of tasks ensuring even distribution of work and collective problem-solving. Together, we have demonstrated how linear programming can be a powerful tool in practical engineering contexts, offering both adaptability and mathematical rigor.