Student thesis series INES nr 631

Automatic icon placement approach for improved association & walkability on city wayfinding maps

Sophia Apostolidou

2023
Department of
Physical Geography and Ecosystem Science
Lund University
Sölvegatan 12
S-223 62 Lund
Sweden

Sophia Apostolidou (2023).
Automatic icon placement approach for improved association & walkability on city wayfinding maps

Master degree thesis, 30 credits in *Subject of degree*

Department of Physical Geography and Ecosystem Science, Lund University

Level: Master of Science (MSc)

Course duration: *January* 2023 until *June* 2023

Disclaimer

Automatic icon placement approach for improved association & walkability on city wayfinding maps

Sophia Apostolidou

Master thesis, 30 credits, in *Geomatics*

Supervisor:
Rachid Oucheikh
Dep. of Physical Geography and Ecosystem Science, Lund University

Exam committee:
Lars Harrie
Dep. of Physical Geography and Ecosystem Science, Lund University

Pengxiang Zhao
Dep. of Physical Geography and Ecosystem Science, Lund University

Acknowledgments

First and foremost, gratitude is due to my supervisor Rachid Oucheikh for his invaluable guidance, rigorous feedback, and unwavering support throughout this course, especially when the research encountered unexpected obstacles. His depth of knowledge and commitment to academic excellence significantly shaped this work.

Appreciation must first be extended to my family and friends, whose enduring love and support have been a beacon, particularly during challenging phases of this research journey. Their understanding, patience, faith in my capabilities and the continuous encouragement have been the driving forces behind the completion of this research.

Building on that foundation, I would also like to express my deepest gratitude to my partner, who stood by my side through both the joyous milestones and the challenging troughs of this journey. His genuine willingness to help, his ever-present support during both highs and lows, and his unwavering belief in my dedication have been invaluable. His patience and care, especially during those intense research nights, have been nothing short of a blessing. In the quiet moments and the chaotic ones, his presence has been a constant source of strength and comfort.

Lastly, a special note of thanks to the contributors of the Pymoo framework. Their unwavering dedication to the advancement of optimization shines through in the framework's robust features, which encompass a wide range of multi-objective optimization facets. Their commitment to continuous development, ensuring the framework remains aligned with industry needs, has been instrumental. The tools provided by Pymoo, have been invaluable to this research and remain a significant asset to the broader scientific community.

Abstract

Over the years, cartography has undergone a significant transformation, evolving from hand-drawn maps to sophisticated digital ones. A crucial task in map production is label placement, which can be time-consuming when done manually. The definition of label includes the text and icons placed on a map. This study focuses on the challenges of automatic icon placement on city wayfinding maps. Despite advancements in automated label placement methods, challenges persist, especially in the field of icon placement in high-density areas. This research aims to optimize icon placement on city Wayfinding maps, addressing challenges of placing icons in high-density areas and maintaining a strong association between icons and their actual locations. A two-stage method was proposed: initially, a grid search algorithm was developed to place the icons in the least disturbing positions on the map. Quality metrics related to legibility, disturbance, and association were also defined to evaluate these placements positions. Furthermore, constraints set by T-Kartor ware considered during the developing process. Then, to optimize the positions, a multi-objective optimization algorithm (NSGA-II) was implemented to optimize simultaneously all the quality metrics. The outcome was a Pareto front comprising a set of non-dominant solutions where all three objectives were optimized. As the primary focus was on the association aspect, the optimal solution chosen among those non-dominant solutions was the one excelling better in terms of association. However, the user can choose the optimal solution that optimizes all the three metrics. In that way, high quality urban wayfinding maps were produced and proved to meet at large extent the real-world production needs and align with cartographic guidelines set by T-Kartor. As a result of enhanced association, both walkability and the overall navigational experience of users were enriched. Future work can focus on improving computational efficiency, considering varied icon shapes, and integrating a plugin that allows the user to customize the level of optimization according to his own preferences.

List of Figures

List of Tables

Contents

1. Introduction

1.1 Background

Cartography, the art and science of map-making, has been an essential tool for centuries, helping people to understand and navigate the world around them. Throughout history, the field has evolved from simple hand-drawn maps to sophisticated digital maps that are widely accessible to the public. Initially, cartographers were primarily focused on employing computer technology to automate map production, as noted by Visvalingam in 1990. Today, with the advent of technology, cartography has become increasingly digitized, resulting in faster map production and greater accessibility for a wider audience.

Label placement on digital maps is an essential part of map production, with a long history of being performed manually. This task can occupy up to 50% of the total time dedicated to designing a map (van Dijk et al., 2002). Digital maps can be broadly classified into two categories: static and dynamic maps. Static maps are designed as fixed images, while dynamic maps are characterized by a higher degree of interactivity, enabling continuous zooming and panning (Been et al., 2006). City wayfinding maps are considered as static, large-scale maps which provide directional information for pedestrians and cyclists in complex urban environments (Harrie et al., 2022). They are usually displayed as posters on pre-existing infrastructure such as bus shelters or station poster frames (City Wayfinding, n.d.) and are an invaluable tool for improving mobility and accessibility within cities.

Labels on maps like these serve to identify and describe various map features, aiding both the user and producer in understanding and interpreting the map's information. The term "label" is commonly employed to refer to texts and icons on a map. The current state of research suggests that there has been limited investigation into the topic of automated icon placement. Thus, the present study seeks to address this gap by focusing specifically on icon placement.

Label placement can be done manually, but it is often time-consuming and error prone. To improve the accuracy and efficiency of map design, this process can be automated through the use of algorithms. Since the 1970s and 1980s, researchers have proposed several methods to address the automatic label placement problem, including combinatorial optimization (van Dijk 2001; Harrie et al. 2005), sequential placement (Rylov and Reimer 2017; Klute et al. 2019), and slider model (van Kreveld et al. 1999; Strijk & van Kreveld 2002). However, the automatic placement of labels poses several challenges, which cause much interactive work in the map production. To ensure clear and effective communication, icon must address these challenges by following general rules such as legibility, association, readability, and aesthetics (Imhof 1975; Wood 2000; van Dijk 2002; Rylov and Reimer 2015). To fulfil these requirements there are specific rules for placing labels on different types of features, including points, lines, and areas.

Despite the improvements that have been proposed by researchers for all the methods, there is still a knowledge gap in the field of automatic label placement, with significant challenges yet to be addressed. Harrie et al. (2022) identified five issues in the placement of labels on City wayfinding maps, including label placement in high-density areas (regions on a map with many features, within a small area), use of true geometries in automated methods, association between icons and text labels, adjustment of text labels to fit within the available space, and road label positioning. These remain important research problems that need to be addressed in order to further improve the efficiency and effectiveness of label placement, including the icons.

1.2 Problem statement

The task of automatically determining the optimal label positions for point features on a map, known as the automatic point feature label placement problem (PFLP), is a challenging process that has garnered the attention of several research studies. The PFLP problem requires placement of labels adjacent to point features in such a way that overlap of labels is minimized or equals zero (Rylov & Reimer, 2014). The problem is defined by (Klute et al., 2019) as follows:

Given a set P of n points in $\mathbb{R}^2$ with a set of label candidates L and a weight function w, find a conflict-free set $S \subseteq L$ of label candidates for P such that the weight $W(S) = \sum_{l \in S} w(l)$ is maximized.

In simple terms, in the PFLP labels are assigned to a set of points on a 2D plane. Each point has a set of label candidates, and each label candidate has an associated weight. The task is to select a subset of label candidates, represented as S, such that each point is assigned at most one label from S, and there are no conflicts (overlaps) between the labels assigned to the same point. The objective is to maximize the total weight of the selected labels in S, to ensure that the most important and informative labels are chosen for each point.

The complexity analysis has shown that the basic PFLP is a non-deterministic polynomial-time hard (NP-hard) problem (Kato and Imai 1988; Marks and Shieber 1991). According to computational complexity theory, a problem H is considered NP-hard if any problem L that belongs to NP can be reduced into H in polynomial time (Knuth, 1974). That means that is very difficult to find an optimal solution to the problem in a reasonable amount of time as it requires significant computational resources, and is potentially costly (Klute et al., 2019).

1.3 Aim of the study

The aim of this study is to find the best placement of icons on a city wayfinding map, address the challenges of placing icons in high-density areas while maintaining good association with their actual locations. To achieve this goal, a two-stage icon placement method is proposed. First, a grid search algorithm places the icons in the least disturbing positions, where they obscure background information as little as possible (Harrie et al., 2004). Then, to improve the found solution, a multi-objective optimization algorithm is implemented to optimize simultaneously all the criteria, namely legibility, disturbance and association. The hypothesis suggests that employing a grid algorithm to search approximate icon positions and subsequently refining them through multi-objective optimization in accordance with city wayfinding guidelines and constraints outlined by T-Kartor[1] will lead to the creation of high-quality maps.

1.3.1 Main objectives

In order to achieve the goal, the main objectives of the study are as follows:

- Address the lack of walkability and readability in commercial online map services such as Google Maps through enhanced icon placement.
- Develop a two-stage algorithm that initially places icons sequentially at the least disturbing positions. Subsequently, it employs a multi-objective optimization of the placement to fine-tune the placement by optimizing various objectives such as disturbance, legibility, association, and potentially other user-defined constraints, rendering this stage adaptable to customization.
- Create high-quality City Wayfinding maps that are effective at guiding users to their destinations and align with the real production needs set by T-Kartor

1.3.2 Research question

The research question addressed in this study is:

[1] https://www.t-kartor.com/

1. How to efficiently automate the icon placement in high-density city wayfinding maps, using an optimization method that fulfils the cartographic rules and enhances the association between icons and the place that they refer to?

1.4 Delimitations

The present study is delimited to the investigation of icons placement, as this is the primary focus of the research. It is important to note that all icons are treated as square in shape. Regarding the text labels, only their bounding boxes that describe landmark building labels and labels of road names are used, rather than the full text of the label. This is due to the fact that the text labels are not visually present on the map, but they are utilized only during evaluation. Furthermore, it is assumed that the text labels are placed before icons. The alignment of icons in relation to text labels is not addressed in this study.

2. Literature overview

The process of placing labels on a map is considered as one of the most difficult and complex problems in cartography. In order to handle the problem efficiently, it is usually split up into smaller independent subtasks such as candidate position generation, position evaluation and position selection (Rylov and Reimer, 2017)

2.1 Candidate positions generation for point features

This task involves generating a set of label candidates for each map feature by considering its type (point, line and polygon), shape, and established cartographic rules. For point features, several methods have been proposed to generate potential label positions, including with the most important ones to be presented as follows.

2.1.1 Fixed position models

Fixed position models, as proposed by Yoeli (1972) and Hirsch (1982), generate a single fixed position for labelling point-like objects, such as cities or landmarks. These models are suitable when the label should always be placed at the same position, such as on top of the point feature. One of the earliest and most widely used models is the 4-Position Model (Freeman, 1988). In this model, four potential label positions are generated around the point feature, each aligned with one of the cardinal directions (north, south, east, west). A preferred label position according to Imhof (1975) is above and to the right of the point.

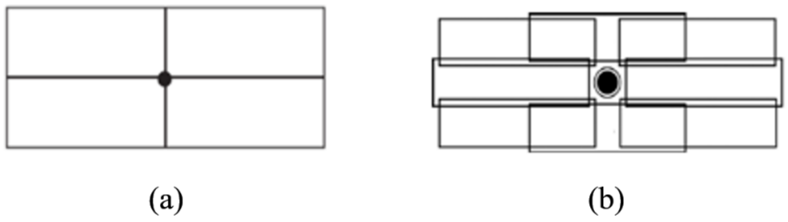


Figure 1: An example of a 4-Position Model (a) and an 8-Position Model (b) for a point feature label. Modified after Lu et al. (2019)

Later, the 8-Position Model was proposed (Imhof 1975; Yoeli 1972) as an extension of the 4-Position Model. In this model, eight potential label positions are generated, including the four positions in the 4-Position Model, as well as positions halfway between the cardinal directions (northeast, southeast, southwest, northwest). Robinson et al. (1995) states that a preferred label position generated from this model can be directly above or below the point. Label placement to the left or right of the point is not ideal, as it can decrease legibility (Slocum et al., 2008) since both the point and the label align on a single line.

2.1.2 Slider models

In contrast, slider label models (van Kreveld et al. 1999; Strijk and van Kreveld 2002; Klau & Mutzel 2003) allow the label to slide along the point feature, enabling the label to be placed at any position within defined boxes (Figure 2), as opposed to being restricted to fixed positions. This creates a higher number of possible positions for each label.

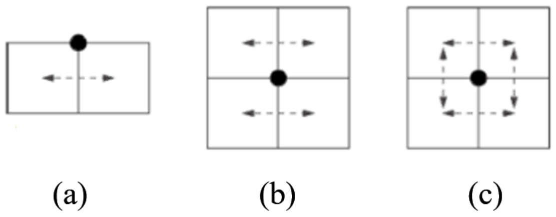


Figure 2: Examples of sliding models for point feature labelling: a) Horizontal label slide within the box. b) Same as a) within an additional box below. c) Both horizontal and vertical label slide. Modified after van Kreveld et al. (1999)

van Kreveld et al. (1999) used this method for point text labelling and found that it produced higher labelling quality compared to methods with a fixed number of positions per label. However, when it comes to labels with large font text, the slider model may not always produce satisfactory results, as it can be challenging to find a good placement for the label within the defined boxes (Strijk & van Kreveld, 2002). To address this issue, Strijk & van Kreveld (2002) developed an efficient algorithm that considers background information and handles labels of different font sizes by adjusting their width. As a whole, it has been proven that this method produces higher labelling quality than methods where labels are not able to slide and there is only a fixed number of positions per label.

2.1.3 Sequential placement

Labels can also be automatically placed in a sequence, and this method is known as sequential placement. It was used by Rylov and Reimer (2017) to label area features outside of their polygonal boundaries. The process involved constructing a bounding box around the original polygon, finding intersection points as an imaginary horizontal line moves over the plane, and eventually identify potential candidate positions around each intersection point. The goodness of each candidate position was then evaluated based on the degree of spatial relationship between the label and the feature it tagged.

However, a primary limitation of this approach is its susceptibility to early misplacements: if a label is positioned poorly early in the sequence, it can adversely affect the entire label placement. This method can also be time-consuming for large datasets. To improve efficiency, several approaches have been proposed, including using optimization algorithms and semi-automatic labelling. The semi-automatic labelling method overcomes the time-consuming issue by initially computing a labelling that has well-placed labels, even though it may not be perfect. The user can then modify the labelling interactively and locally as needed based on his own criteria. (Klute et al., 2019)

2.1.4 Grid algorithm

Harrie et al. (2004) developed an algorithm for placing icons in the least-disturbing position. That indicates that the icon is placed in such a position that it obscures as little information as possible. Algorithmically is defined as the place that returns the minimum sum of weights for the cartographic points, which are points that form the cartographic objects. In general, the algorithm operates by moving the icon to each possible position and checking which cartographic points are covered by it for each position. The computational complexity of this simple algorithm is O (n^2 * *numPoints*), where n^2 represents the total number of possible positions and *numPoints* is the number of cartographic points. Such complexity indicates the drawback of a lower speed when more cartographic points are involved.

Thus, the authors suggested an approach called grid algorithm for improving the computational complexity to O *(numPoints* + $n^2 * p$), where p is the distance between two neighbouring search positions. The grid was used to divide the search area for the icon into smaller sections, so the algorithm could evaluate each of these sections individually.

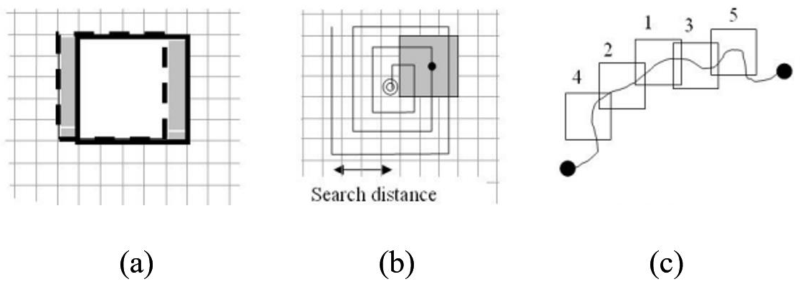


Figure 3: Grid algorithm process. Illustration of a window transitioning from its initial position (dashed lines) to a new position (solid lines)(a). Only the grid values within the shaded grey regions require adjustment to compute the disturbance value for the new icon position. The search for the optimal icon position, starting from the original location (indicated by a double circle) and progresses in a spiral pattern (b). For positioning icons representing line objectects (c) the algorithm can perform repeated spiral searches along the line around the midpoint (box 1). Created by Harrie et al. (2004)

Figure 3 shows the workflow of the grid algorithm process, which is described by Harrie et al. (2004). The algorithm takes as input the original position and the size of the icon as well as the permitted movement of the icon from its original position, the resolution of the search space, all the cartographic objects on the map and the importance (weight) of each cartographic object type. The weight is given based on how important it is to keep the icon away from that specific object type.

For each feature in the dataset, the algorithm checks if it overlaps with the original point of the icon within a certain search distance. If it does, the weight of the feature is retrieved and added to a list. The search area is later divided into a grid of cells. For each point in the list of weights, its weight is assigned to the cell in the grid that contains it and the total value of the weights in each cell is calculated. When the window is moving from one position (dashed lines) to a new position (solid lines, Figure 3a), only the grid values in the grey areas need to be adjusted to determine the disturbance value for the new icon position. The search for the optimal position begins at the original position of the icon (marked with a double circle) and progresses in a spiral pattern (Figure 3b). Finally, the icon's position with the lowest disturbance value is returned by integrating the grid values and returning the coordinates of the corresponding cell.

However, this method appears to have some limitations. Firstly, it is only designed to be used for positioning a single icon. While it can be adjusted to place multiple icons, the code must be extended to determine their positions sequentially. A subsequent icon is not possible to overlap an already positioned icon. Additionally, the algorithm does not consider text and is specifically designed for square icons. Nevertheless, it can be adjusted for placing rectangular icons by using a search space that is also rectangular.

Despite these limitations, the grid algorithm offers several benefits. For normal values of *numPoints* and *p*, it is faster than the simple algorithm. This makes it particularly useful for real-time maps that require fast performance. Furthermore, the algorithm can be used to position icons that represent line objects by performing repeated spiral searches along the line around the midpoint (box 1) as shown in Figure 3c. Overall, this method has been proven to produce good quality visual results and its computational complexity makes it easy to use for real-time maps.

2.2 Position evaluation

The evaluation of potential label positions is a task that involves measuring the quality of each position based on how well the label is positioned with respect to the feature it tags and to the rest of the map content (van Dijk et al., 2002; Hong et al., 2005). The quality of labelling is quantified by an objective function, which is normally formed by the weighted sum of single metrics (quality functions) (Rylov and Reimer 2014; van Dijk et al., 2002; Zhang and Harry 2006). The total cost of the summed factors with weights defines the objective function value of a label placement solution. The choice of maximizing or minimizing an objective function depends on the specific labelling problem and the preferences of the map maker. Therefore, the final labelling solution would be the one either with the highest or with the lowest objective function.

The construction of the objective function is important since it directly affects which solution is finally selected by the algorithm. Each quality function in the objective function serves a specific rule or criterion for label placement. There are several ways to define an objective function. Lu et al. (2019) in their work gave a simple definition of the objective function as presented in equation 1. In their case the quality functions (S_j) where four: (1) label conflict, (2) label-feature conflict, (3) label non-ambiguity, and (4) label priority. The weight of each function (W_j) is scaled to the interval of 0–1 such that their sum is 1. Generally, the weights are set as $W_1 > W_2 > W_3 > W_4$. The label conflict factor to have the greatest weight, meaning that it has the greatest influence on the quality of label placement.

$$S = \sum_{j=i}^{4} S_j * W_j \qquad (1)$$

Rylov et al. (2015) in their objective function (equation 2) defined a quality function for measuring the amount of the map information that is hidden by a label. They also quantified the degree of visual contrast between a label and other features in the map background, which format was raster. Their refined measure created by the substitution of $f_{feat-visibility}(l)$ and

$f_{label-visibility}(l)$ in the following equation and is based on information obtained from the map background after implemented an image segmentation algorithm (Haralick & Shapiro 1985; Pal & Pal 1993).

$$F(L) = \sum_{l \in L} \Big(w_1 \cdot f_{priority}(l) + w_2 \cdot f_{aesthetics}(l) + \\ + w_3 \cdot f_{association}(l) + w_4 \cdot f_{label-visibility}(l) + w_5 \cdot f_{feat-visibility}(l) \Big) \quad (2)$$

where $L=(l_1, \dots, l_n)$: set of n labels on the map, w_i, $i=1,2,..,5$: weights and $f^*(l)$: partial quality functions.

The label placement approach involves several quality metrics, including the importance, classification, and hierarchy of a labelled object ($f_{priority}(l)$). Additionally, the aesthetic quality of a label, such as its position and shape with respect to the feature it annotates, is evaluated using $f_{aesthetics}(l)$. The degree of association between a particular feature and its label is measured by $f_{association}(l)$, while $f_{label-visibility}(l)$ considers the visibility of a label in relation to other features and labels on the map. Lastly, the $f_{feat-visibility}(l)$ evaluates the visibility of a feature on the map with respect to other features and labels.

Both Lu et al. (2019) and Rylov et al. (2015) employed an objective function to evaluate the quality of label placement solutions. Although the specific quality metrics used in each study vary, there are some common factors like prioritization and overlapping. However, the second function includes a quality metric for evaluating aesthetics aiming to create a visually pleasing and informative map.

2.2.1 General cartographic requirements

As stated above the quality of map labelling is evaluated using a weighted quality function for each one of the established cartographic requirements. In previous research cartographers (Imhof 1975; Yoeli 1972) listed a number of rules for a high-quality result. Here are summarized the four main categories of general rules (as described by van Dijk et al., 2002) that must be obeyed when comes to label placement.

- Legibility: Avoid overlapping between labels
- Readability (Not disturbing the map content): Labels should not cover important information of the map background when it is needed to be placed on top of map objects.

Ideally, they should overlap only homogenous areas and less important objects. Regarding the map objects, they should allow a clear interpretation of labels.

- Association (Unambiguity): Labels should be close to their corresponding objects in order to be easy to interpret in which map objects they refer to.
- Aesthetics: The placement of labels on the map should enhance its aesthetic quality as a whole

For the purpose of this study, the primary focus lies on the first three rules—legibility, readability, and association. Aesthetics, while important, are not the central concern of this research.

2.2.2 Challenges in city wayfinding maps

In the production of city wayfinding maps, Harrie et al. (2022) identified, among others, two important challenges in icon placement that need to be evaluated and solved before deciding about. The first challenge is finding a way to effectively place labels in high-density areas on a map. These are regions that have many map features, i.e., points, lines, or polygons, within a small area. Because of the limited space for both map features and labels, it is difficult to define priorities between the labels. For instance, one label type always should be in priority over another label type. Automated labelling tools such as those implemented in QGIS and ArcGIS (i.e., Maplex) can provide satisfying results in terms of readability, but they do not follow an important requirement of city wayfinding maps, which states that all the labels should be present on the map.

The second challenge involves finding an effective way to create a good relationship between text labels and icons on the map. That means that icons are properly associated with the corresponding map features they describe. This is particularly important in city wayfinding maps, where the labels are often used to provide directional and other important information to pedestrians and cyclists (City Wayfinding, n.d). Currently, text labels and icons are treated as separate objects, which can lead to problems as it implies that their placement is independent of each other. So far there is no automated way to handle this challenge, but a manual solution has been proposed by Harrie et al. (2022), which involved combining the text labels and icons into a single label (icon), which is later placed interactively.

2.2.3 Production needs

In addition to the general cartographic requirements and challenges in city wayfinding maps, there are real production needs, that cartographers in T-Kartor defined, when it comes to icon placement. These needs, as summarized below, are crucial in ensuring the accuracy, legibility, and overall quality of the map and promote walkability in cities.

- **Location Accuracy:** Icons should be placed accurately to represent their real-world locations. For example, a TicketStop point should be placed closer to the road or pavement where it is actually located (Figure 4).
- **Avoiding Overlap:** Icons should be placed in a way that they do not overlap with other important elements on the map. For example, a landmark building's icon or text label should be placed within the building without crossing the road or text (Figure 5).
- **Icon Rotation:** Icons, such as Busstops, can be rotated 90 degrees away from the road if there is room. This prevents a text change direction on the icon but still on buildings (Figure 6).
- **Flexibility in Symbol Placement:** Some flexibility is allowed in symbol placement. For example, some overlap is allowed with icons and a road text, and certain symbols may need to be moved if there is space.
- **Avoiding Obstruction:** If a symbol is still obstructing, it may need to be moved or adjusted.

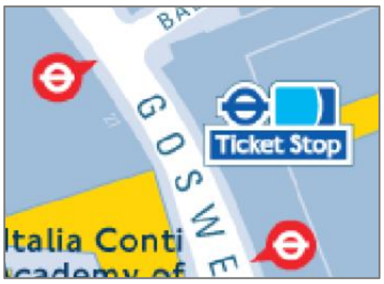


Figure 4: Adjust the position of TicketStop symbol to the pavement. Avoid placing it centered on the point it represents, so the user can easily interpret on which side of the road it is in reality © Copyright Transport for London

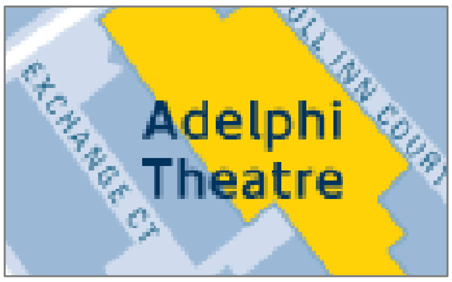


Figure 5: Placement of label with its largest part inside the associated building, without obscurring other information © Copyright Transport for London

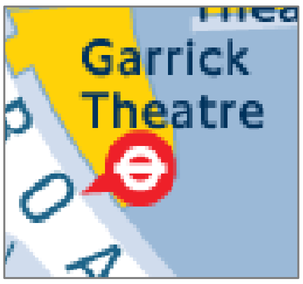


Figure 6: Example of 90 degree icon rotation © Copyright Transport for London

The above needs highlight the complexity and precision required in cartography, especially when dealing with icon placement, and also underline the importance of developing automated tools to effectively address them.

2.3 Position selection

The aim of map labelling is to find a good placement for labels according to a set of requirements. Most researchers (e.g., Zoraster 1997; van Dijk 2001) consider point labelling as a combinatorial optimization problem (Schrijver, 2003). Combinatorial optimization involves finding the best combination of label positions based on specific requirements and an objective function that is minimized or maximized using an optimization algorithm. Thus, this kind of a problem can be solved using optimization algorithms (Edmondson et al., 1996). These algorithms require two components to be defined: a discrete search space consisting of elements representing candidate label positions and an objective function. (Rylov et al., 2015).

Finding an optimal solution to the problem would mean considering every possible combination of label placement position and selecting the one that results in the best label placement, based on an objective function. The problem of finding the optimal label placement becomes even more complex when there are many features to be labelled, as the number of possible combinations grows rapidly. Therefore, researchers applied heuristic optimization algorithms like simulated annealing (Zoraster 1997; Edmondson et al. 1996), greedy (Yoeli 1972; Christensen, Marks, and Shieber 1995), tabu-search (Yamamoto et al., 2002), and genetic algorithms (Lu et al, 2019) to solve it. In a general sense, heuristic algorithms as they provide quick and practical solutions in a reasonable amount of time that are not guaranteed to be optimal, but are often satisfying, without requiring a large memory space to operate (Wang & Chen, 2013).

Addressing the intricacies of point feature labelling, particularly concerning icons and finding their best placement on the map, has been a significant area of exploration. Harrie et al. (2005) in their study were trying to evaluate methods to select candidate positions for both the icons and text labels in real-time maps before the combinatorial optimization step. The method for real-time placement of icons and text labels is divided into four main steps: compute search space for text label placement, compute search space for icon placement, perform combinatorial optimization and remove the conflict labels. The paper emphasized that a good selection of candidate positions not only saves computational time during the combinatorial optimization but also yields a cartographically satisfactory result.

Later, Zhang and Harrie (2006) continued this research following the above method of combining the placement of icons and labels on maps in real-time applications, with an aim to find an optimal solution by reducing the search space. Two strategies were evaluated: random reduction of the search space and reduction based on quadrant selection. The experiments conducted to compare the effectiveness of the two strategies involved labelling different features on maps, such as named locations and roads, using a simulated annealing algorithm. The quality of the labelling was assessed based on factors like label overlap and cartographic disturbance. The study found that both search-space-reduction strategies improved label quality compared to not reducing the search space at all. However, the strategy of reducing the search space randomly yielded better label quality than quadrant-based selection and it was also more efficient in terms of computational time. The study also highlighted the potential for integrating this method into a system architecture for real-time map services.

2.3.1 Optimization algorithm

For finding optimal for labels on a map, different optimization algorithms can be used as stated above. In this study the focus is given to the Non-dominated Sorting Genetic Algorithm II (NSGA-II), which is a popular multi-objective optimization algorithm that handles complex problems effectively (Deb et al., 2002). NSGA-II operates by maintaining a population of potential solutions and iteratively improving them through genetic operators (like crossover, mutation), as well as a non-dominated sorting and crowding (Manhatten distance in the objective space) approach to choose the best solutions (Blank & Deb, 2020). The algorithm operates quite fast, having computational complexity of O (MN2), where M is the number of objectives and N is the population size (Deb et al., 2002). The algorithm promotes solutions that are better in at least one objective without being worse in others (Yusoff et al., 2011), preserving diversity in solutions, which is essential for multi-objective problems.

The general steps of the NSGA-II algorithm as described by Yusoff et al. (2011), include:

1. **Population Initialization**: Start by initializing the population based on the problem's range and constraints. This involves creating a set of potential solutions (individuals) with random or predetermined values for the decision variables.
2. **Non-Dominant Sort:** Perform a non-dominated sorting process on the initialized population (Figure 7a). This sorting process determines the dominance relationship between individuals based on their objective function values. Individuals that are not dominated by any other individual are assigned to different fronts or levels of dominance.

3. **Crowding Distance:** Once the non-dominated sorting is complete, assign a crowding distance value to each individual in the population (Figure 7b). The crowding distance measures how crowded an individual is within its front. It helps to maintain diversity by favoring individuals that are located in less dense regions of the front.
4. **Selection:** Select individuals for the next generation based on their rank and crowding distance. This is typically done using a binary tournament selection with a crowded-comparison operator. The individuals with better ranks or greater crowding distance are more likely to be selected.
5. **Genetic Operators:** Apply genetic operators to create offspring for the next generation. The NSGA-II algorithm uses real-coded genetic operators such as simulated binary crossover (SBX) and polynomial mutation. SBX performs crossover by combining the genetic material of two parents, while polynomial mutation introduces random modifications to diversify the population.
6. **Recombination and Selection:** After applying the genetic operators, recombine the selected individuals with the offspring to form the new population for the next generation. The selection process is then repeated to choose individuals for the subsequent iterations, ensuring the preservation of the fitter individuals and diversity.
7. **Iterative Process:** Repeat the above steps until a termination condition is met.

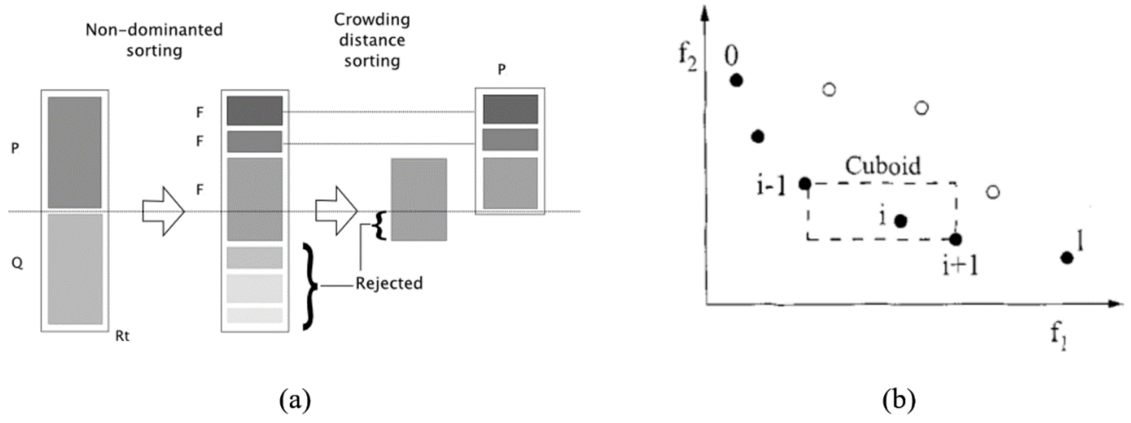


Figure 7: Non - dominated sorting process on the initialized population (a) and after the non - dominated soring is complete a crowding distance value is assigned to each individual in the population (b). Created by: Deb et al. (2002)

2.4 Lack of walkability and readability in commercial online map services

The primary objective of placing icons on city wayfinding maps is to is to enhance the navigational experiences of pedestrians and cyclists. While these maps offer static guidance at designated city locations, a significant number of people turn to online map services for real-time assistance. However, studies have revealed that such online services do not always adhere to fundamental cartographic principles, leading to more complex and confusing navigation. This section delves into various dimensions of this discourse, exploring walkability in urban settings, the concept of legibility in cartography, and the limitations inherent in popular online map services in terms of icons placement.

2.4.1 Walkability definition

The urban environment significantly affects the experience of walking in terms of comfort, safety, and satisfaction. The concept of "walkability" refers to how accommodating and conducive the built environment is for pedestrians, allowing them to walk easily and comfortably (Habibian & Hosseinzadeh, 2018). This aligns with the definition provided by (Transport & Litman, 2009), which describes "walkability" as a measure of how friendly an area is to walk.

The evaluation of walkability commonly involves assessing particular attributes of the built environment, such as the variety and concentration of land use, as well as the features of pedestrian facilities (Fonseca et al., 2021). To provide a more comprehensive analysis, Bartzokas-Tsiompras et al. (2021) have introduced a set of 17 pre-processed and microscale walkability indicators in spatial and tabular data files. The dataset covers 26 European countries (59 central areas) and aims to bridge the existing gap in European-level indicators and promote sustainable communities. Specifically, the indicators cover the seventeen themes related to micro-level characteristics of neighbourhood design, such as sidewalks, crosswalks, lights, curb cuts, street furniture, parks, transit, aesthetics.

Using the web-GIS platform developed during the above study, an illustrative example is provided in Figure 8, showcasing a performance comparison of five European cities across six indicators. The findings reveal that Copenhagen excels in well-maintained sidewalks and an abundance of pedestrian walk signals at crossings, while Warsaw demonstrates efficient pedestrian crosswalks.

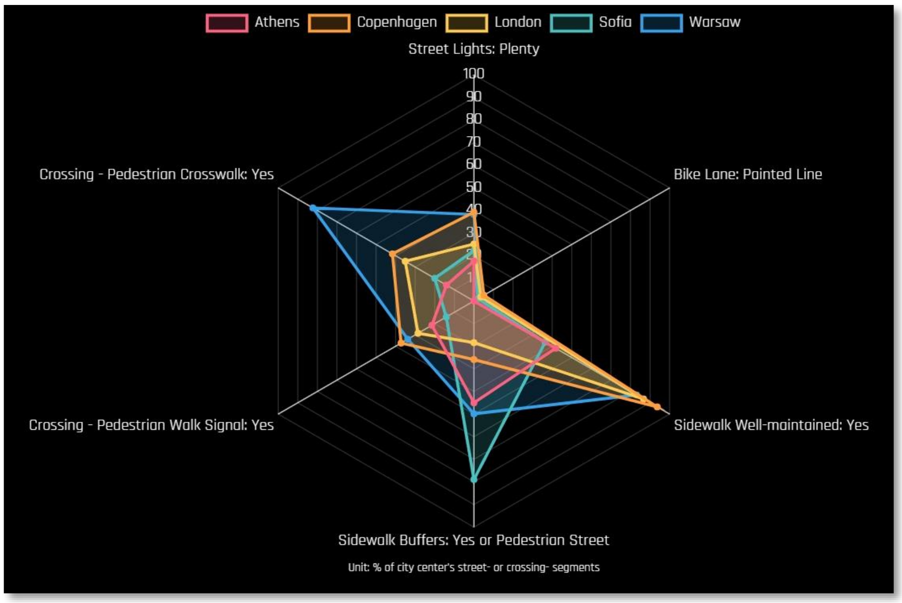
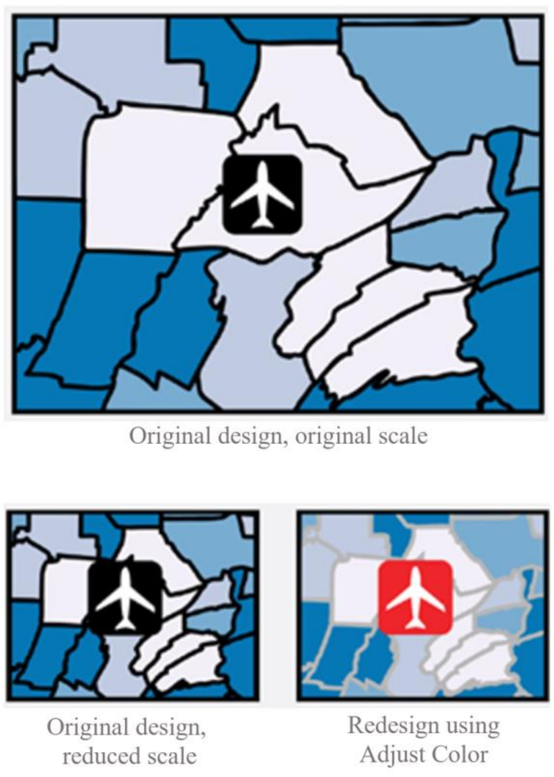


Figure 8: Comparison of microscale environmental factors and its indicators for five central urban areas (Athens, Copenhagen, London, Sofia, Warsaw. The indicators are Street lights: Plenty, Bike lane:Pointed line, Sidewalk well-maintained: Yes, Sidewalk buffers: Yes or pedestrian street, Crossing – Pedestrian walk signal: Yes and Crossing – Pedestrian crosswalk: Yes. Created in Walk & the City Center website, n.d.

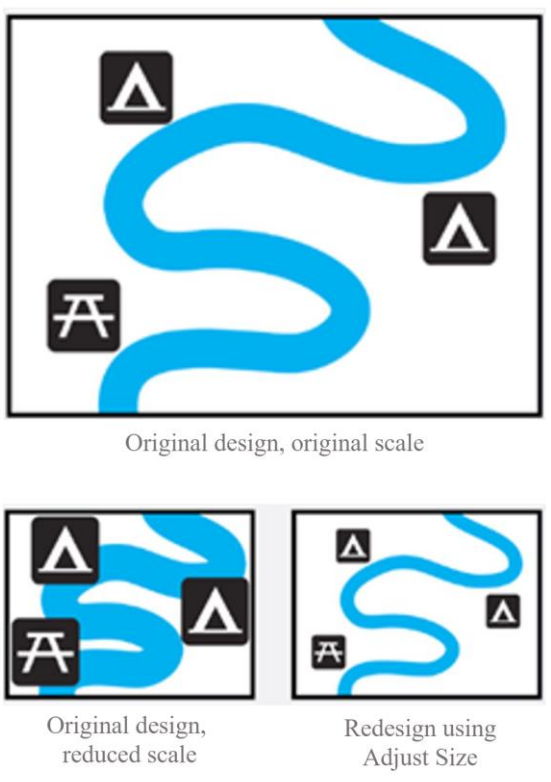
2.4.2 Readability definition

In addition to walkability, another important concept in the realm of cartography is the readability of maps. Readability in cartography plays a vital role in facilitating navigation, decision-making, and spatial understanding. It is particularly relevant in urban environments where maps are often used for wayfinding, urban planning, and transportation purposes. An early definition of readability, proposed by Robinson in 1986, refers to the ease with which map readers can interpret and understand the information presented on a map, such as readability of icons and labels and clarity of patterns. Robinson's work significantly contributed to the field of cartography, and his definition of readabiliy remains widely recognized and referenced in the study and practice of map design.

However, in practical terms, many research studies primarily focus on readability as the avoidance of labels and icons overlap (Imhof 1975; van Dijk et al., 2002). Consequently, the readability of a map heavily relies on the careful selection of symbols. Opting for familiar symbols that are appropriately sized enhances their visibility and ease of understanding. For instance, geometric symbols are easier to read at smaller sizes, while more complex symbols require more space to be legible (Buckley, 2012). Some examples of improved readability on maps are given in Figure 9 (a and b).


Original design, original scale
Original design, reduced scale
Redesign using Adjust Color

(a)


Original design, original scale
Original design, reduced scale
Redesign using Adjust Size

(b)

Figure 9: Examples of improved readability on maps by adjusting the symbol colour (i.e., hue, lightness, or saturation) to ensure legibility of the surrounding features (a) and (b) by adjusting the size of the symbol without changing feature dimensionality, so it remains readable when transitioning to a smaller scale. Credit: Adapted from (Roth et al. 2011) Cartographic Perspectives, (CC BY-NC-ND 3.0).

Map design must follow specific rules and be compliant with the principles of design and graphic communication (i.e., legibility, visual contrast, hierarchical organization etc.). Online map services offer simplified maps known as general reference maps, which depict key natural and man-made features (Skopeliti & Stamou, 2019). In terms of geographic coverage, online map services can either offer global accessibility, like Google Maps, Wikimapia, Apple Maps, HERE, Bing maps, OpenStreetMap (OSM), Yahoo! Maps, and others, or focus on specific regions, such as Baidu Maps in China, 2gis.ru in Russia, and Mapy.cz in the Czech Republic. According to data from Similarweb, as of May 2023, Google Maps stands out as the most widely recognized online map service worldwide.

Pedestrian navigation has become an essential service for people living in cities. With the rise of smartphones and navigation apps, individuals now heavily rely on pedestrian navigation to find their way around urban areas. In their study titled "Pedestrian Navigation Aids, Spatial Knowledge, and Walkability" (2016), Wang and Worboys conducted a pilot wayfinding experiment to examine the effects of two major pedestrian navigation aids, Legible London, and Google Maps, on users' acquired spatial knowledge. The study compared the impact of these aids with the direct experience of navigating routes in London. The Legible London system, developed as part of T-Kartor's city wayfinding project, is a comprehensive signage system for pedestrian wayfinding throughout the city. Initiated by Transport for London in 2007, it aims to provide clear and user-friendly navigation guidance for pedestrians across various areas of the city.

Specifically, the project offers legible and user-friendly information to assist people in navigating the city, seamlessly connecting public transportation with walking, and cycling environments. The project includes innovative features such as "Heads-up" map signs that rotate in the direction of travel for quicker and easier map reading. Additionally, a 5-minute walking circle around the "You Are Here" position highlights nearby amenities. Illustrations of landmark buildings aid in creating a mental map of the surroundings, reinforcing the connection between the map and the real world. Maps at two complementary scales provide both detailed street and footpath information for precise journey planning, as well as an overview of the broader network. By providing improved information, this initiative encourages active mobility within the city.

The results of the study mentioned above showed that the type of navigation aids people employed to guide themselves, can significantly affect their ability to understand their surroundings and the ease with which they navigate on foot. Specifically, users of both services, faced difficulties in gaining a clear understanding of how streets fit together, making it challenging for them to navigate and choose the best route to their destination. Legible London's "heads up" style made it easy to understand the immediate environment, but mentally aligning the signage maps with reality took time when the direction differed from the walking direction. Moreover, Legible London users had limited opportunities to learn about their surroundings, while Google Maps users showed less attention to their routes and surroundings, leading to poorer orientation during navigation (Wang & Worboys, 2016). These findings highlight the need for improved pedestrian navigation aids that enhance spatial knowledge and promote walkability in cities.

2.4.3 Limitations of commercial online map services

Despite their widespread use, commercial online map services have their limitations:

- Prioritization of vehicular traffic, neglecting pedestrian needs.
- Lack of detailed information about pedestrian pathways, sidewalk conditions, and crossing safety, crucial for walkability.

- Excessive details clutter the maps, making it challenging to identify important elements in urban areas.
- Poor icon placement obscures crucial information and causes confusion.

To address these limitations, it is crucial to employ clear and concise symbology that effectively communicates information. Generally, symbolization in on-line map services is in accordance with the cartographic practice. Skopeliti and Stamou (2019) in their paper "Online Map Services: Contemporary Cartography or a New Cartographic Culture?" reviewed online map services from a cartographic point of view and their findings regarding symbology, for Google Maps, OSM, HERE and Wikimapia are summarized below:

- They differ in terms of generalization degree and information density, with OSM offering the highest density and diversity compared to Google and HERE.
- Areal and linear symbols cover most of the map area, while point symbols are rare at the examined scale (zoom levels 13-14).
- Symbol sizes are well selected to support legibility, and at larger scales, Points of Interest (POI) are portrayed using pictorial symbols based on shape and colour.
- Google Maps uses the familiar pin symbol as a background with a pictorial symbol on top for point representations.
- Good visual contrast is observed in Google Maps, OSM, and HERE maps, while Wikimapia's vivid contrast is unpleasant.
- Symbolization of the road network, which is the dominant thematic layer, is crucial for the aesthetic quality of the map. Wikimapia's use of a vivid yellow hue creates an unbalanced result, while OSM achieves a balanced result through symbolization choices (Figure 10).
- Despite its information density, OSM manages to create a balanced result due to effective symbolization choices.

(Skopeliti & Stamou, 2019)

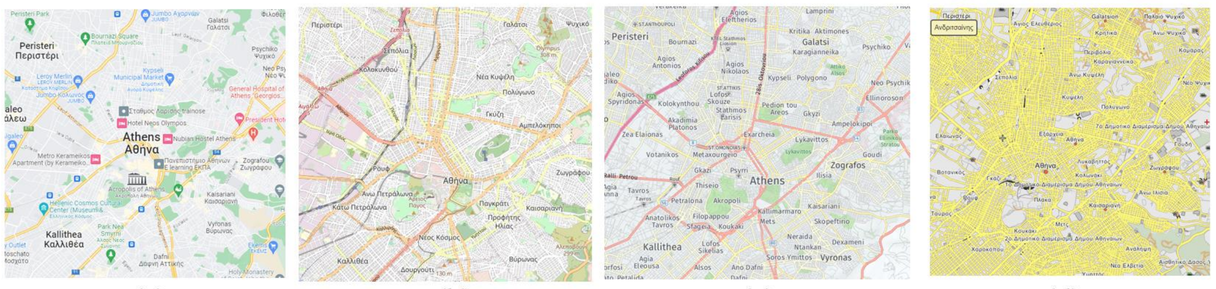


Figure 10: Images showing Athens city in Greece and the sourounding areas, of the home pages of (a) Google Maps, (b) OSM, (3) HERE, (4) Wikimapia. Source: Google Maps (https://www.google.com/maps/@37.989609,23.7156318,13z), OpenStreetMap (https://www.openstreetmap.org/#map=13/37.9861/23.7160), HERE WeGo (https://wego.here.com/?map=37.98775,23.74855,13,normal), Wikimapia (http://wikimapia.org/#lang=el&lat=37.988993&lon=23.715363&z=13&m=w)

So, it is important for online map services to consider these insights and make appropriate symbolization choices that effectively convey information while maintaining readability and walkability.

3. Methods

3.1 DRM framework

The design research methodology (DRM) is a problem-solving approach that involves a series of steps including problem identification, hypothesis generation, solution design and testing, solution evaluation, and communication and dissemination of results (Horváth, 2007). The focus is on developing practical solutions that can be implemented in real-world settings. The present study adopts the DMR framework to find solution to the problem of icon placement on city wayfinding map and to produce a high-quality output for real word applications, by evaluating and optimizing the placement based on defined criteria.

3.2 Available datasets

Data from the London city map project provided by T-Kartor was used for the current study. These maps have been produced in an ESRI ArcGIS environment using the Maplex label engine and undergone substantial manual label editing in both the ArcGIS environment and in the publishing editing tool Adobe Illustrator. Furthermore, the initial code for the grid algorithm as developed by Harrie et al. (2004), was provided in Java from the supervisor of this thesis. A detailed table with all the data employed in this study is given below.

Table 1: *Information about the data used in the study. The name, type, geometry and number of features (for SHP files) are given. For easier inderpretation the data are divided in categories such as main project, text labels, background, icon layers, symbols and code.*

	Name	Type	Geometry	Number of features (for study area)
Main Project	London20211006	QGIS Project		
Text Labels	clip_LMF_Landmark_Building_T	SHP File	Polygon (MultiPolygon)	265
	clip_L_ LMF_Road_Names_T			564
Background	Landmark	SHP File	Polygon (MultiPolygon)	322
	BuiltUp			525
	Green			233
	Road			290
	Railway			0
	Water			2
	L_ LMF_CH_Pavement_A			595
	Pavement			564
	L_ LMF_LONDON_BACKGROUND_A			1
	clip_layer_points	SHP File	Point	51021
	clip_layer_lines		Line (MultiLineString)	90771
Icon layers	L_ LMF_Building_Entrances_P	SHP File	Point	24
	L_ LMF_Station_Entrance_P			22
	L_ LMF_CP_Entrances_P			3
	L_ LMF_Subways_P			4
	L_ LMF_Bus_Stops_P			100
	L_ LMF_CH_Stations_P			55
	L_ LMF_Car_Parks_P			18
	L_ LMF_Car_Parks_P			0
	L_ LMF_Car_Parks_P			54
	L_ LMF_Police_Stations_P			0
	L_ LMF_Post_Offices_P			8
	L_ LMF_Post_Offices_P			8
	L_ LMF_Post_Offices_P			0
	L_ LMF_View_Points_P			0
Symbols	Building_entrance	SVG		
	CHF_Dark_Inner_Station_UG			
	Building_entrance			
	CHF_Dark_Inner_Station_A_UG			
	Bus_Stops			
	CH_Stations			
	CHF_Dark_Inner_CarPark1			
	CHF_Dark_Inner_Retail9			
	CHF_Inner_IconTaxi			
	CHF_Inner_IconPolice			
	CHF_Inner_Dark_IconPost			
	CHF_Dark_Inner_Toilet2			
	CHF_Dark_Inner_Station_D			
	CHF_Dark_Inner_Retail9			
Code	IconPlacementAlgorithm	Java Source File		

3.3 Methodology Overview

The general steps that were followed through the current study are presented in Figure 11.

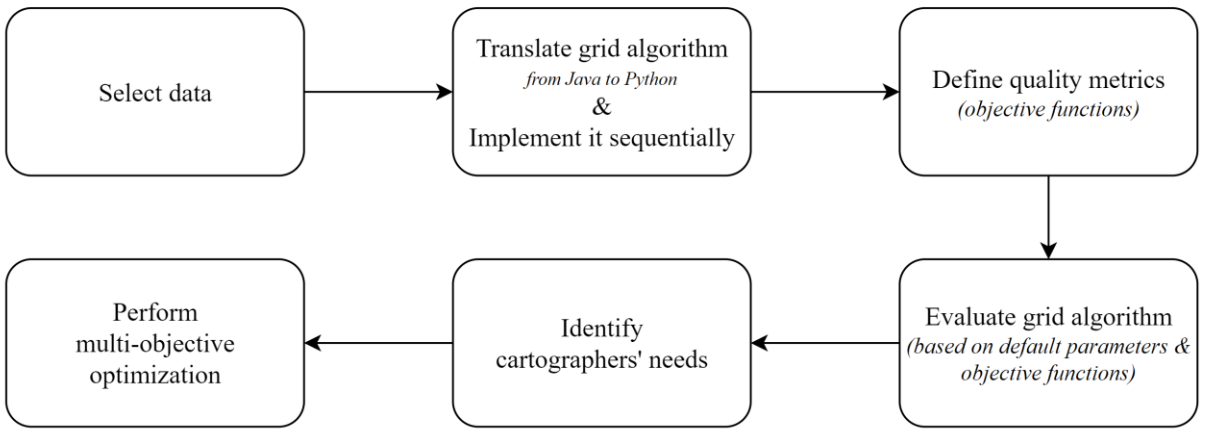

Select data
Translate grid algorithm
from Java to Python
&
Implement it sequentially
Define quality metrics
(objective functions)
Evaluate grid algorithm
(based on default parameters & objective functions)
Identify cartographers' needs
Perform multi-objective optimization

Figure 11: General workflow of the study

First step was to crop the original vector layers to the extend of the study area. Subsequently the original grid algorithm was translated from java to python to understand the logic behind it and a new version was developed in Python to position icons sequentially in the least disturbing positions, considering factors like background layer importance and icon movability. Next step was to define the objective functions for legibility, disturbance, and association to evaluate the quality of icon placements. Then the grid algorithm was texting by changing default parameters and evaluated based on the defined objective functions, to select the combination of parameters that performs better in the association metric. In parallel, T-Kartor's guidelines for city wayfinding maps were defined. The study concludes with the implementation of a multi-objective optimization process, applied after the grid algorithm, in order to optimize the positions generated by the grid algorithm by improving association with respect to the other two metrics.

3.3.1 Data selection

The workflow of this study involves several steps to achieve the objectives of the research. Firstly, all the original vector layers were cropped using the “Clip multiple layers” plugin in QGIS. The reason behind that is that the data cover the whole greater region of London which is around 1,600 km^2 and contains around 28,000 icons. The crop represents the study area (Figure 12), covering an area of 3.4 km^2 and containing 243 icons.

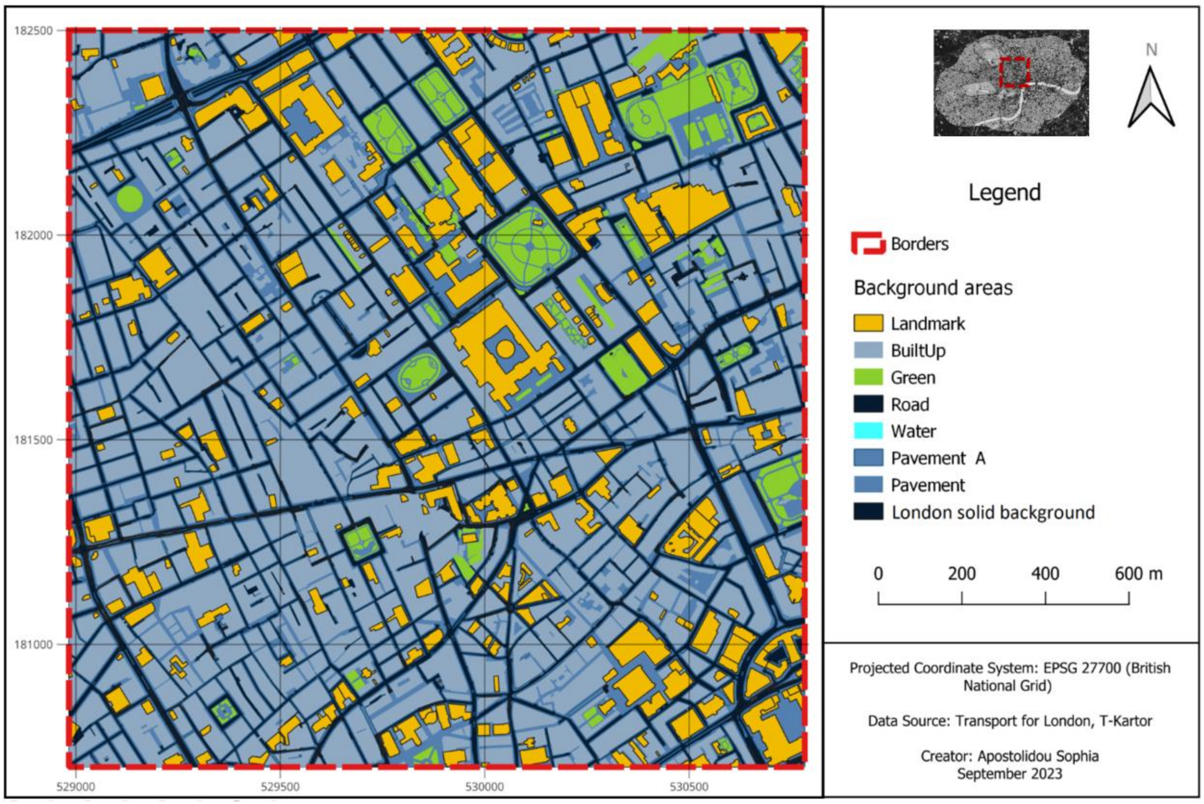


Figure 12: Map of the study area, Data source: © Copyright Transport for London

3.3.2 Grid algorithm implementation

In the subsequent stage, the Python script was used to develop the grid algorithm presented in section 2.1.4, utilizing pseudocode (see Harrie et al., 2004) and pre-existing script in Java. However, the initial grid algorithm did not address the issue of positioning icons sequentially. That means when an icon is placed the next one is not permitted to overlap it. The logic behind the grid algorithm that was developed in the current study is briefly described as follows:

In general, the implemented grid algorithm aims to position icons in the least disturbing positions by taking into account the existed elements on the map. It considers factors such as the importance of background layers, the movability of icons based on layer categories, and priority for placing icons. To achieve this a grid-based approach is utilized. A grid is applied to the search window to divide it into smaller cells or pixels. The use of a grid allows the algorithm to efficiently evaluate multiple possible positions for the icon in a systematic manner. Each cell in the grid represents a potential position for the icon. The size of the grid cells is determined based on the step size, which depends on the size of the icons and the number of possible positions in each direction.

Weights are also assigned to the different layers of the background, to reflect the relative importance of each layer. The higher the weight the most important it is for the algorithm to avoid placing icons close to its features. In the current study weights range from 1 to 3. A weight of 1 is given to the *Green, Railway, Water* and *Pavement layers,* indicating that icons are relatively free to be placed close to these features without much restriction. On the other hand, the *Landmark* layer is assigned a weight of 2, suggesting a moderate level of importance. While icons should not be placed close to landmark features, there's a degree of flexibility allowing them to be situated within a landmark feature rather than crossing a road feature (See section 2.2.3). Conversely, the highest weight of 3, emphasizing the utmost importance of keeping away from these features, is reserved for the *Road* and *BuiltUp* layers.

Moreover, icons are categorized into different layers based on their movability. The first category includes stationary icons such as *Building Entrances* which should not be moved by the algorithm, since it is important that they have an exact location. The second category includes icons that point to a location and can be moved around that point. The last category consists of icons that can be placed inside a polygon feature or on top of a line feature. Table 2 provides a categorization of layers and their symbolization.

Table 2: Divide layers in three categories based on their ability to move

	Stationary	Size (m)		Around point	Size(m)		Inside polygon or on line	Size (m)
						P	Car Parks	15
						H	Info Centers	15
⬆	Building Entrances	20	⊖	Bus Stops	18	🚕	Taxi ranks	14
⊖	Station Entrance	12	🚲	CH Stations	23	POLICE	Police Stations	20
⬆	CP Entrances	10				POST OFFICE	Post Offices	20
⊖	Subways	2				🚻	Toilets	20
						✈	Airports	15
						H	View Points	15

The algorithm begins by utilizing a spiral search pattern (Figure 3b). Starting from a central position, it systematically explores the available grid cells for icon placement in an outward spiral manner. This structured search approach ensures that all potential positions are considered and that no other satisfactory positions can be found closer to the original one.

To determine the best placement, the algorithm evaluates each cell in the grid, calculating a disturbance score. This score quantifies the extent to which placing an icon in a particular cell would disrupt the overall layout. Several factors contribute to the disturbance score, including the proximity to existing icons and the balance of the entire arrangement.

In details, the algorithm iterates through the collected background features and their corresponding weights. For each feature, it detects the spot on the grid where the feature is located. By aligning the grid with the search area, it ensures accurate placement of the icons. The weights of the background cartographic features are then added to the corresponding cell on the grid. The algorithm proceeds to sum the weights of the cells that the icon would cover for each possible icon position. This sum represents the disturbance score for each position. The preferred position is the one that has the minimum sum of the weights.

A priority system has also been incorporated to ensure icons of each category are placed strategically. Because icons that belong to the first category are not permitted to change their initial positions, they automatically added as background information. So, the algorithm starts by placing first the icons that belong to the second category followed by those of the third. The implementation is done sequentially for all the icons in the dataset. That practically means that when the first icon is placed by the algorithm, it has automatically been added to the background layers. Thus, for placing the next icon, the algorithm considers all the features that belong to the background and places it to the appropriate position to avoid overlapping. This process continues until all icons are positioned.

Finally, the algorithm evaluates all the spots on the grid and determines the one that fulfils all the above steps. This spot is considered the least disturbing position for the new icon since it minimizes disruptions to background information. The algorithm returns the *x* and *y* coordinates of this spot, providing the final solution for the placement of the new icon on the map.

3.3.3 Quality metrics definition

The three metrics that were designed were based on the work of Oucheikh and Harrie (2023). They were implemented for both the icon and the layers of the background map area. The legibility metric also considers the text labels for landmarks and roads objects. Furthermore, they were all normalized to have a range of values between 0 and 1.

Legibility

This metric evaluates the legibility of the icon l_j in relation to all the other icons and also the legibility between the icon and the text labels. The bounding box of road and landmark labels are used as text labels. The metric is defined as the degree of overlap between the icons to each other and between icons and text labels and it is calculated based on the total intersection area and the total area of all the icons. The objective is to ensure that the icons are visible and easy to interpret. The quality function is defined as follows (equation 3) and its output ranges from 0 to 1 with higher value indicating the best result, the one with a lower degree of overlapping.

$$Legibility_l = 1 - \frac{\sum_{l_j \in L, l_j \neq l_l} area(l_i \cap l_j)}{area(l_i)} \tag{3}$$

Where *Legibility*$_l$ is the legibility factor between a specific text label and all the icons, L is the set of all labels (icons and text labels) geometries, l_l is a text label, l_j is an icon, l_i are all the rectangles that surround the text labels and *area (G)* is area of geometry *G*. When legibility between a specific icon and all the other icons needs to be calculated, l_l represents that icon ($l_l \neq l_j$)

Disturbance

The disturbance metric is related to readability of the map. As mentioned in section 2.2.1 labels should be allowed to be placed in homogeneous areas. The metric takes values from 0 to 1, with higher value occurring when the icon hides any relevant information of the background features, which are represented by break points and lines. The definition of disturbance is expressed as follows:

$$Disturbance_l = w_1 * bp + w_2 * t_l \tag{4}$$

where *bp* is the number of feature's break points overlapped by the label *l*, t_l is the total length (in map units) of overlapped line segments by the label *l* and w_1, w_2: are the weight factors. In this study, after experimental testing the weights were set to 0.01 and 0.0037, respectively.

Association

To ensure a clear association between icons and background objects on a map, it is important for the icons to be located close to their corresponding objects. Two different association metrics are defined for the second and third category layers (see Table 2). Layers in the first category are not evaluated as their position is fixed and they are considered as background information.

For the second category icons, the approach involves creating a defined window around each point feature. This window acts as a permissible placement area for the four potential icons that point to this feature. To account for any potential errors and provide some flexibility, a buffer zone of 20 meters around this window is being incorporated. The association is then evaluated based on two criteria: proximity to the original icon and containment within the buffered bounding box. The metric is calculated using the Formula (5). The values range from 0 to 1 with 1 meaning that the new icon is inside the buffered bounding box and close to the original position.

$$Association_l = 1 - (w * distance + (1 - w) * int\ (not\ is\_inside)) \qquad (5)$$

where *l* is the icon to be evaluated, *distance* is the shortest distance between the new and the original placed icon (in map units), *is_inside* is a Boolean variable that is True if the bounding box of the new icon is inside the buffered bounding box. The *not* is a logical operator that inverts the truth value of the operand, so if is_inside is True, not is_inside will be False, and vice versa. The *int()* function converts this to an integer, so True becomes 1 and False becomes 0. *W* is a weight factor set to 0.5, giving equally importance to *distance* and *not is_inside* componets. This balance ensures that both the proximity of the new icon to the original and its positioning relative to the buffered bounding box are considered when evaluating the association of the icon *l*.

For the third category icons, the approach involves two basic steps. First it is checked if the icon is inside the background area and if there is overlap between the bounding box of the icon feature and the background polygon feature. For icons placed on top or close to line objects the degree of overlap with the line is considered. Then the distance between the new placed icon feature and the original placed one is calculated. The metric ranges from 0 to 1 with 1 indicating that the icon is inside the polygon of the background feature and close to the original icon. The function is defined as:

$$Association_l = 1 - (w_1 * overlap + w_2 * distance) \qquad (6)$$

where *overlap* is the percentage of bounding box of the icon being covered by the background feature object, *distance* is the shortest distance between the new and the original icon (in map units) and w_1, w_2 are the weight factors, set after experimental testing to 0.1 and 0.9 ($w_1 < w_2$ as overlap is less important than distance).

3.3.4 Grid algorithm evaluation

Initially the grid algorithm is tested by changing two important parameters (*searchDistanceGround* and *p.orgX*, *p.orgY*. See section 4.1.1). The results produced after each combination of the parameters are stored and evaluated based on the three metrics of legibility, disturbance, and association (see Appendix 2). The goal is to evaluate the placement based on the most interesting cartographic requirements, rather than only considering the number of placed icons. The combination that performs better on the evaluation of the association metric are chosen as the definitive one. The positions of icons generated using this combination are optimized later using multi-objective optimization.

3.3.5 Cartographers' needs

The placement and evaluation of icons should comply with T-Kartor's guidelines for positioning icons on city wayfinding maps as defined in the project Transport for London[2]. The following are the general guidelines divided into rules [R] and constraints [C] adapted to the purposes of this study. Rules act as soft constraints, providing guidelines that ideally should be adhered to for optimal results. However, there is some flexibility, and under certain conditions, deviations from these rules might be acceptable if they lead to better overall solutions. Constraints, on the other hand, are non-negotiable and must be strictly followed.

[R1] Upper limit for the number of icons attached to a single point: No more than four icons for each location

[R2] Icons should be placed off the streets and are permitted to overlap buildings

[R3] Cycle hire, bus stops and taxi symbols should be preferably placed in a 90-degree angle to their corresponding road. Other angles are permitted if needed. Can not be placed over roads (only in certain situations)

[2] Transport for London (2011) Street map design standard - Issue 2

[R4] Bike and traffic roundel symbols, e.g., the roundel of the bus stop symbol, must always be horizontally placed, but the circle that encloses them may be rotated as needed.

[C1] No texts symbols or icons may be removed or suppressed

[C2] Icons may sometimes be allowed to cover text labels if it is the best solution

[C3] Icons must not overlap or being placed closed with each other. A balanced distribution across the map should be kept.

[C4] Icons should not overlap other significant background features, or if overlap is necessary, should be kept to minimum

3.3.6 Multi-objective optimization

The grid algorithm includes explicitly the disturbance quality function through the inherent disturbance metric used in the search. However, it does not consider the other quality functions or the specific cartographic rules. To achieve the best solution, the algorithm is iteratively optimized. For a problem to be considered as multi-objective optimization problem, needs at least two objective functions to be simultaneously optimized (Salian, 2022). In this study, multi-objective optimization is used to find the best trade-off between the three different objectives (metrics). This involves optimizing the quality functions simultaneously to find a set of solutions that represent the best compromise between the different objectives, along with different constraints to satisfy. To address a multi-objective problem, a reasonable solution is to explore a set of solutions that meet the objectives adequately without being outperformed by any other solution (Konak et al., 2006). The selected multi-objective optimization algorithm tries to find icon placements that minimize the values of disturbance, while maximizing legibility and association. In that way a high-quality output is produced.

In the context of this problem, NSGA-II algorithm used as optimization algorithm, is expected to work by gradually adjusting the coordinates of the features in the given space to improve the objectives while satisfying the constraints. Each solution in the population is an arrangement of all the icon features on the map. The algorithm generates new solutions in each iteration by combining and modifying existing solutions, aiming to improve the objective values. To execute the algorithm, an instance of NSGA-II was created with a population size of 40 and the customized sampling method. This method was created to force the algorithm start the exploration from the already good solution, obtained by the grid algorithm. Each generation of

NSGA-II was processed using the parameters set in the 'MyProblem' class (see section 3.4.3). After running the optimization for 10 generations, the best solution was found in terms of *x* and *y* coordinates for the features and their corresponding objective function values. The total elapsed time was 5 hours for a given number of 243 icons.

3.4 Code implementation

The implementation of the study was completed using Python as the programming language and PyQGIS as the library. Several scripts were used for developing the grid algorithm, the objective functions, and the multi-objective optimization process. The code developed in this study is distributed on GitHub[3] MIT licence. The amount of python files created are presented in Appendix 1, Figure 28, Appendix 1. Below is presented briefly the main structure of each script and the important details.

3.4.1 Grid algorithm

This code was designed to work with QGIS, and it was executed in the QGIS Python console. It is an implementation of the grid algorithm to place icons sequentially on the map in a way that minimizes disturbance with background map features. It uses a disturbance matrix to determine the best position for each icon and then updates the map with the new icon positions. Below is given the main structure of the script developed.

1. **Initialization:**
   - Import necessary QGIS modules.
   - Initialize a QGIS application.
   - Load a specific QGIS project.
2. **Utility Functions:**
   - getLayer(group_name): Returns all layers within a specified group in the QGIS project.
   - getFeatures(group_name): Returns all features within layers of a specified group.
   - getSize(theFeature, defWidth, defHeight): Determines the size of a feature's symbol.
   - getBoundingBox(icon_feature, theFeature): Calculates the bounding box of an icon.
3. **Parameters Class:**

   A class that holds various variables used in the algorithm, such as the original coordinates, width, height, number of steps and others.
4. **Matrix Update Functions:**

[3] https://github.com/SPHAPST/Master-Thesis

The functions (addToNumMatrixPoint, addToNumMatrixLineString, addToNumMatrixPolygon, addSymbolToMatrixPolygon) update a matrix that represents the disturbance of icons on the map. The matrix helps in determining the best position.

5. **Distance and Coordinate Functions:**
   - distance(c1, c2): Computes the Euclidean distance between two points.
   - computeNewCoordinate(c1, c2, distLimit): Calculates a new coordinate based on a distance limit.
6. **Icon Placement Value Class:**

   A class that represents potential icon placements and their associated disturbance values.
7. **Window Localization Functions:**
   - localiseWindow(p): Determines the best position for an icon based on the disturbance matrix.
   - addTextWindow(p): Calculates the position for the search window.
8. **Main Icon Placement Function:**
   - placeIcon(layers, icon_feature, iconlayer_name, searchDistanceGround): The main function that places an icon on the map. It calculates the best position for an icon based on the disturbance matrix and other parameters.
9. **Execution:**
   - Retrieve all original layers, iterates through each icon in the original group, and places them on the map using the placeIcon function.
   - Each icon placed is added to the background group of layers (as new layer seq_layer) before placing the next one, to avoid overlapping.
   - The new positions of the icons are then added to a new layer.
10. **Visualization**

    For each icon layer:
    - Retrieve its symbology.
    - Create a new in-memory layer (output_labels) for storing newly placed icons.
    - Add layers to the project.

3.4.2 Objective functions

Disturbance

The code is designed to compute the disturbance metric for a group of layers in a QGIS project. It is based on the overlap of the icon with the lines and points that describe the background features. The more overlap there is, the higher the disturbance. The final output is an average disturbance factor for the entire group of layers. The following steps describe the main structure of the code for implementing this metric.

1. **Initialization:**
   - Import necessary QGIS modules.
   - Initialize a QGIS application.
   - Load a specific QGIS project.
2. **Utility Functions:**
   - getLayer(group_name): Returns all layers within a specified group in the QGIS project.
   - getSize(theFeature): Determines the size of a feature's symbol.
   - getBoundingBox(point, theFeature): Calculates the bounding box of an icon.
   - load_lines(): Loads all line segments from a specific layer.
   - load_points(): Loads all point features from a specific layer.
   - getpositions(layergroup): Returns the x and y positions of icons in a group of layers.
3. **Disturbance Calculation Functions:**
   - disturbance_factor(bp, tl, w1=0.01, w2=0.0037): Computes the disturbance factor based on overlapping points and total length of overlapping lines.
   - readability_lines(x_coords, y_coords, i_layer): Calculates the total length of lines that intersect with a feature's bounding box.
   - readability_points(x_coords, y_coords, i_layer): Counts the number of points that intersect with a feature's bounding box.
4. **Result Aggregation Functions:**
   - f1_single(x_coords, y_coords, i_layer): For a single layer, calculates the average disturbance factor by combining results from readability_lines and readability_points.
   - f1(group_name, x_coords, y_coords): For a group of layers, calculates the average disturbance factor by invoking f1_single for each layer in the group.
5. **Execution**
   - Compute the average disturbance factor for a group of layers.

Legibility

The code is designed to compute the legibility metric for a group of layers in a QGIS project. It is based on the overlap of icons with text labels and with other icons. The more overlap there is, the lower the legibility. The final output is an average legibility factor for the entire group of layers. The main steps of the code are:

1. **Initialization:**

   Same as above

2. **Utility Functions:**
   - getLayer(group_name): Returns all layers within a specified group in the QGIS project.
   - getSize(theFeature): Determines the size of a feature's symbol.
   - getBoundingBox(point, theFeature): & getBoundingBox2(geometry, theFeature): Calculates the bounding box of a feature based on its size. The second version handles different types of geometry.
   - getpositions(layergroup): Returns the x and y positions of icons in a group of layers.

3. **Legibility between icons and text labels:**

   legibility_text(x, y, i_layer, t_layer): For each icon in a layer (i_layer), this function calculates the legibility metric based on the overlap with text features from another layer (t_layer). The legibility metric is based on the area of overlap between the bounding box of the icon and the text features. The function returns the average legibility factor for the i_layer.

4. **Legibility between icons:**

   legibility_icons(x, y, i_layer, i_layer2): For each icon in a layer (i_layer), this function calculates the legibility metric based on the overlap with icons from another layer (i_layer2). The legibility metric is based on the area of overlap between the bounding boxes of the two icons. The function returns the average legibility factor for the i_layer.

5. **Main Legibility Functions:**
   - f2t(x_coords, y_coords, group_name, t_layers): Calculates the average legibility factor for a group of icon layers (group_name) with respect to a list of text layers (t_layers). It calls the legibility_text function for each combination of icon layer and text layer and then averages the results.
   - f2i(x_coords, y_coords, group_name): Ccalculates the average legibility factor for a group of icon layers (group_name) with respect to other icon layers in the same group. It calls the legibility_icons function for each combination of icon layers and then averages the results.

6. **Execution**
   - Compute the average legibility factor for a group of layers.

Association

The code is designed to compute the association metric for a group of layers in a QGIS project. The association metric is calculated differently for the different categories of layers (see section 3.3.3). The final output is the average association factor for the entire group of layers.

7. **Initialization:**

   Same as above
8. **Utility Functions:**
   - getpositions(layergroup): Returns the x and y positions of icons in a group of layers.
9. **Association for 2nd category:**
   - f3_2_G (x_coords, y_coords, g_layers, o_layers, n_features): Calculates the average association factor for the 2nd category of layers. It calls the f3_2 function (from the imported Association_2ndC_xy module) for each combination of the current icon layer and other icon layer and then averages the results.
10. **Total association:**
    - f3(x_coords, y_coords, g_layers, o_layers, b_layers): Calculates the average association factor for the entire group of layers. It separates the layers into the 2nd and 3rd categories. For the 2nd category, it calls the f3_2_G. For the 3rd category, it calls the f3_3 function (from the imported Association_3rdC_xy module) for each layer and averages the results. Finally, it averages the results from the 2nd and 3rd categories to get the overall average association factor for the group.
11. **Execution:**
    - Compute the average association factor for a group of layers.

3.4.3 Multi-objective optimization

The optimization step is carried out in Python utilizing the Pymoo framework (Blank & Deb, 2020) which is available on the Python Package Index (PyPI). This is a central repository to make Python software package easily accessible (PyPI - the Python Package Index, n.d). It provides algorithms for single- and multi-objective optimization, complemented by additional features like visual tools and decision-making capabilities for multi-objective optimization (Blank & Deb, 2020). The steps when using Pymoo are simple. Firstly, the problem and the quality functions are specified together with the constraints. Then an appropriate algorithm available in Pymoo is applied to find a set of optimal solutions to the problem. An overview of the general structure of the script developed for this process is given as follows:

1. **Problem initialization and constraints**

To initialize the problem, existing geographic coordinates, generated using the grid algorithm, were retrieved from the project layers, and set as the initial positions of features. This was done using the 'MySampling' class, which starts the optimization from the point of the pre-existing solution by reusing these coordinates.
An instance of the 'MyProblem' class was then created, which encapsulates the problem definition. Within the 'MyProblem' class, various attributes and methods are defined to represent the characteristics and behaviour of the problem. The attributes include the initial *x* and *y* coordinates of the features, the group name of the group that includes all the layers, and the layers representing the background information, the original positioned icons, and the text labels in the project. These attributes are essential for evaluating the objective functions (used as arguments) and constraints during the optimization process (Appendix 1, Figure 28).

Additionally, the 'MyProblem' class defines:

- **Number of Decision Variables:** The decision variables in this problem are the *x* and *y* coordinates of the features. Since there are 'len(x_coords)' features, there are '2 * len(x_coords)' decision variables, where '2' accounts for the *x* and *y* coordinates for each feature.
- **Number of Objective Functions:** As mentioned in section 3.3.3 the quality functions are 4 (1 for legibility, 1 for disturbance and 2 for association). However, in the current code the functions of association were merged into one that used a different function to calculate the association metric for each category of layers. On the other hand, the legibility as coded in two distinct manners: one for determining the legibility factor between icons and text labels (roads and landmarks) and another for assessing the legibility between the icons themselves (section 3.4.2). But the last one was not reasonable to be optimized as it relays on calculating distances between the new placed icons. Given the algorithm's sequential nature, many of the new icons may not yet be placed. So, the objectives being optimized are, f1 stand for disturbance, f2t for legibility between icons and text labels, and f3 for association. Therefore, the number of objective functions is set to 3.
- **Lower and Upper Bounds for Decision Variables:** To define the search space for the optimization process, lower and upper bounds are specified for each decision variable

(*x* and *y* coordinates). The lower bound for both *x* and *y* is set as the corresponding initial coordinate minus 40 (map units), allowing for a range of movement. Similarly, the upper bound is set as the initial coordinate plus 40, defining an acceptable range of exploration around the initial position during the optimization.

- **Number of constraints:** Constraints were built into the problem definition to account for certain limitations. To avoid confusion, it is worth mentioning that they are not related to constraints mention in section 3.3.5. These constraints included upper bounds for all the function values to account for acceptable error margins (0.05). The goal is to improve what obtained so far from the grid algorithm, but if this is not possible, code should at least keep the original values and ensure that they do not exceed the initial values by more than 0.05.

The '_evaluate' method within the 'MyProblem' class is responsible for evaluating the objective functions and constraints based on the given decision variables (Appendix 1, Figure 30). It calculates the values of the objective functions by calling the relevant functions (f1, f2t, and f3) with the current *x* and *y* coordinates. Similarly, it evaluates the constraints for functions by comparing the calculated values with the upper limits defined for each constraint.

2. **Algorithm Initialization**

   Set up the NSGA2 algorithm for optimization with a population size of 30 and the custom sampling method.
3. **Execution**
   - Run the optimization for 10 generations, using the ‘MyCallback’ class to print information after each generation.
   - Print the best solution found, check if the solutions are within bounds, and save the results to text files.
4. **Visualization**
   - Plot the Pareto front in both 3D and 2D to visualize the trade-offs between the objectives.
   - Visualize the solutions on the QGIS map project

4. Results

This section begins by showing the results from testing different parameter combinations, in order to check how the developed grid algorithm performed in each metric (disturbance, legibility and association) for all the icons in the project (Table 3). Then it is discussed why the specific parameters have been chosen as the ones to proceed with in this study and the produced new locations of the icons are presented on a map (Figure 14). A comparison between the original positions of the icons and the new ones is shown on a map to underline the differences (Figure 15). Instances where the algorithm successfully adhered to the rules and constraints set by T-Kartor are also showcased in Figure 16. Later is presented the performance of metrics before the optimization process and finally, the results of the multi-objective optimization method are examined.

4.1 Grid algorithm evaluation

4.1.1 Parameters

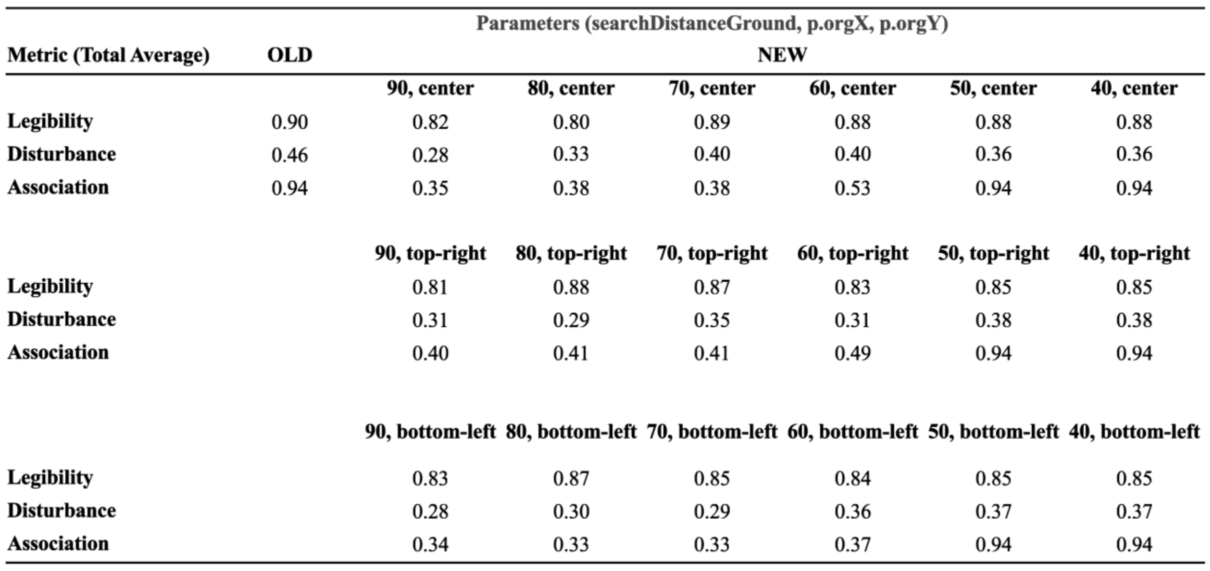
During the first test of the developed grid algorithm, it had been noticed that there are two important parameters which influence its performance. The first one is the *searchDistanceGround* variable which is the size of the grid cell in the ground space (step size) and controls the movement of the icon from its original position. The actual grid cell size (in ground space) is influenced by the combination of the icon's width and the value of *searchDistanceGround*. When a low value to *searchDistanceGround* is assigned, it results to a denser grid with smaller grid cells, reducing the search area and making it more likely for the algorithm to choose the same placement for multiple icon features. On the other hand, a larger value results in a coarser grid providing more potential positions for icon placement. As a result, the algorithm can find different positions for each icon, based on the weights and disturbances in the area.

p.orgx and *p.orgY* are the next variables that influence the position of icons when placed on the map. They serve as the coordinates of the reference point (bottom left, center, top right) of the bounding box that surrounds the icon feature. This bounding box is the search window. Using *p.orgX* and *p.orgY*, the code further defines a bigger window (or a matrix) to evaluate where the icon can be placed with the least disturbance to other features. These bigger boundaries of the window are later centred on *p.orgX* and *p.orgY*. This is then used to determine the optimal position for the icon. Adjusting these values affects where the function starts its evaluation and can bias the result towards certain directions. For instance, if the *p.orgX* and *p.orgY* are redefined to represent the bottom left of the bounding box, the center of the evaluation window will be shifted to the top right of the bounding box of the icon. In general, centring on the bounding box provides a balanced, neutral evaluation.

4.1.2 Experiments

Below are presented the statistics derived from the objective functions for the metrics of disturbance, legibility, and association, prior to optimization. These results were achieved by executing the grid algorithm using varied parameters. In total, eighteen experiments were conducted, with each representing a unique parameter combination. The *searchDistanceGround* variable was set at values of 90, 80, 70, 60, 50, and 40 (map units). Meanwhile, the *p.orgX* and *p.orgY* variables were tested in three positions: centred, top-right, and bottom-left within the bounding box. Result values span from 0 to 1, where 1 represents the optimal solution for legibility and association metrics. On the other hand, for disturbance metric 0 indicates the best result (less degree of overlapping). A summary of the statistics is given in Table 3 for a quick interpretation. However, in Appendix 2 the detailed table of experiments is presented, where the average of each metric is calculated for each layer and each category of layers. Additionally, calculations for the legibility metric are detailed separately for road, landmark labels, and icons, offering a deeper analysis. The following table presents the legibility metric as average between icons - text labels and icons themselves.

Table 3: Summary statistics of the average value (of all icons) for each metric across the eighteen experiments. "Old" indicates the values for metrics for the manually placed icons, while "new" the values for metrics after the placement using the grid algorithm.



		Parameters (searchDistanceGround, p.orgX, p.orgY)					
Metric (Total Average)	**OLD**	**NEW**					
		90, center	**80, center**	**70, center**	**60, center**	**50, center**	**40, center**
Legibility	0.90	0.82	0.80	0.89	0.88	0.88	0.88
Disturbance	0.46	0.28	0.33	0.40	0.40	0.36	0.36
Association	0.94	0.35	0.38	0.38	0.53	0.94	0.94
		90, top-right	**80, top-right**	**70, top-right**	**60, top-right**	**50, top-right**	**40, top-right**
Legibility		0.81	0.88	0.87	0.83	0.85	0.85
Disturbance		0.31	0.29	0.35	0.31	0.38	0.38
Association		0.40	0.41	0.41	0.49	0.94	0.94
		90, bottom-left	**80, bottom-left**	**70, bottom-left**	**60, bottom-left**	**50, bottom-left**	**40, bottom-left**
Legibility		0.83	0.87	0.85	0.84	0.85	0.85
Disturbance		0.28	0.30	0.29	0.36	0.37	0.37
Association		0.34	0.33	0.33	0.37	0.94	0.94

The table presents the total average value for each metric for all the icons in the project. Originally, the legibility, disturbance, and association metrics for the pre-placed icons were 0.90, 0.46, and 0.94, respectively. The experiments aimed to discern the most effective parameter combination across all metrics, intending to identify a strong (albeit not optimal) result for further study application.

Regarding the metric of disturbance, all the experiments produced better results (lower to zero values), which proves that the grid algorithm really improves the readability of the map. The combinations of 90 center and 90bottom-left notably outperformed others in this domain, registering a value of 0.28. On the other hand, the legibility does not seem to be improved, which means that the overlapping with text labels and other icons remains almost the same or in some cases (80 center and 90 top-right) is getting worst. Only the value of the third experiment (70, center) is close to the original one, but not better.

With a deeper look into the calculations for legibility metric for the road text labels, the landmark text labels and the icons (see Appendix 2) the grid algorithm improves slightly the legibility of road labels for the experiments 60 center, 50 center and 40 center. Also, it shows improvement in the legibility of icons, with many combinations (70 center, 60 center, 80 top-right, 70 top-right, 80 bottom-left) to have better outcome than the original result.

For the association metric a value close to 1 indicates that the newly placed icon is very close to its original position and inside the background feature (or the defined bounding box for 2nd category icons). As seen in Figure 13, there is a noticeable trend: as the value of *searchDistanceGround* increases, the metric gets lower values, suggesting poorer performance due to the deviation of the icon from its initial placement. The 60 center combination emerges as the most optimal in this scenario, with a value of 0.53.

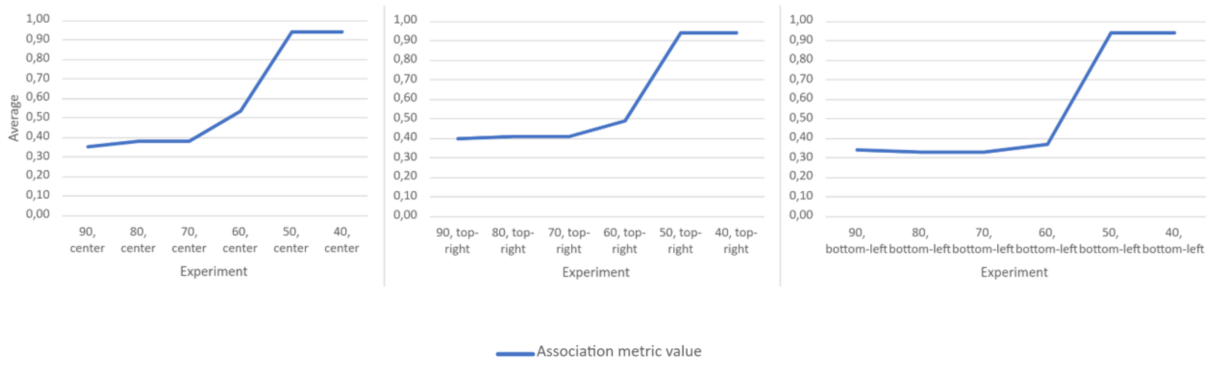


Figure 13: Trends in the experiments for the association metric when the position (p.orgX, p.orgY) remains the same and the searchDistanceGround values change

Overall, experiments with searchDistanceGround lower than 50 produce same results for the same p.orgX and p.orgY. Hence, combinations with these particular values are deemed unreliable and have been excluded from further exploration. Given the consistent improvement in the disturbance metric across all experiments, the 60 center combination is preferred to continue with mainly due to its notable performance in the association metric.

4.1.3 Visualization of icon placement

An example of how icons are placed in the area after running the adjusted grid algorithm with the defined parameters is presented in Figure 14. However, to understand their relative positions to the original placed ones, Figure 15 gives a deeper insight.

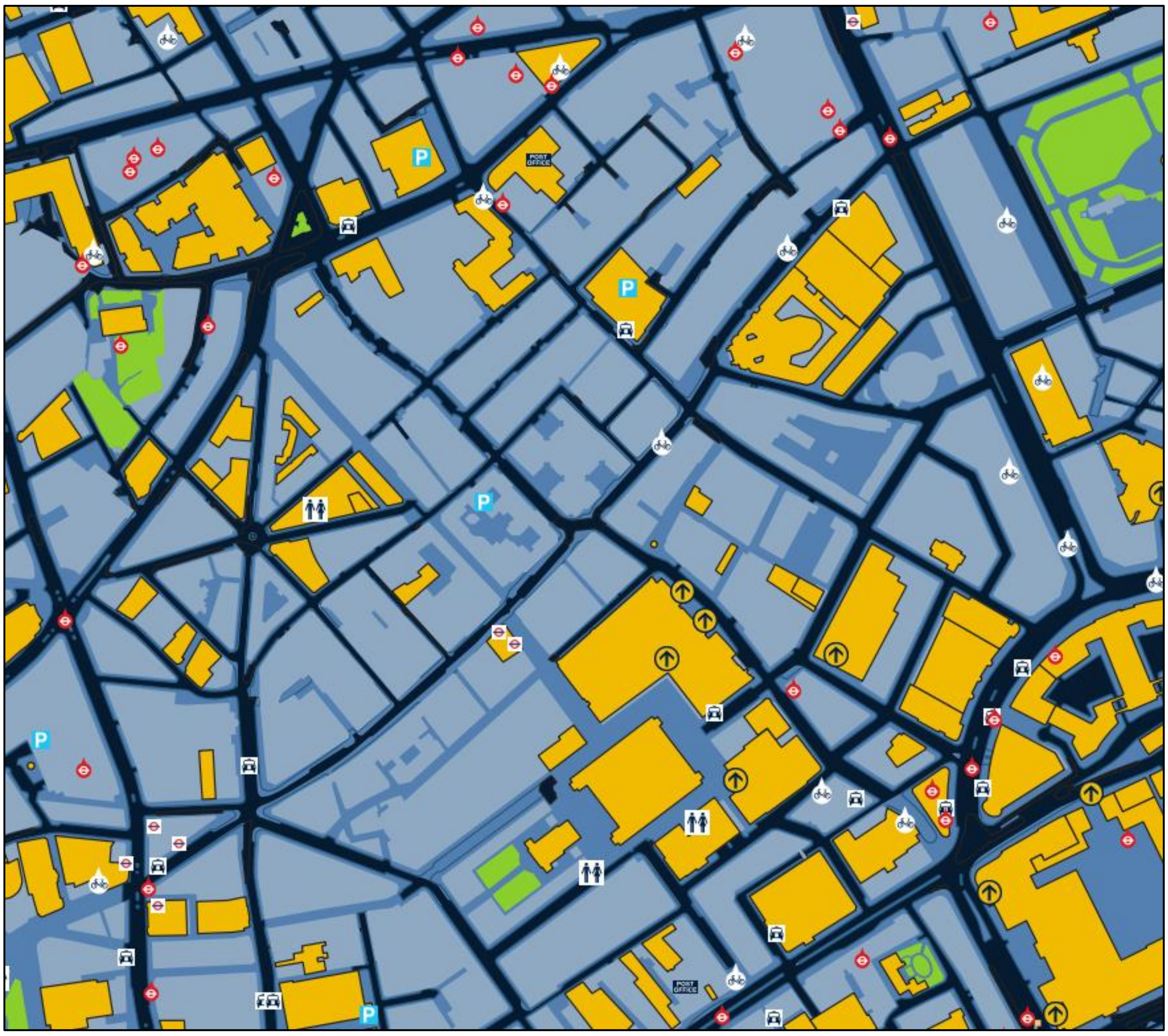


Figure 14: Zoom in to the bottom right quarter of the study area. Icons are placed using the adjusted grid algorithm

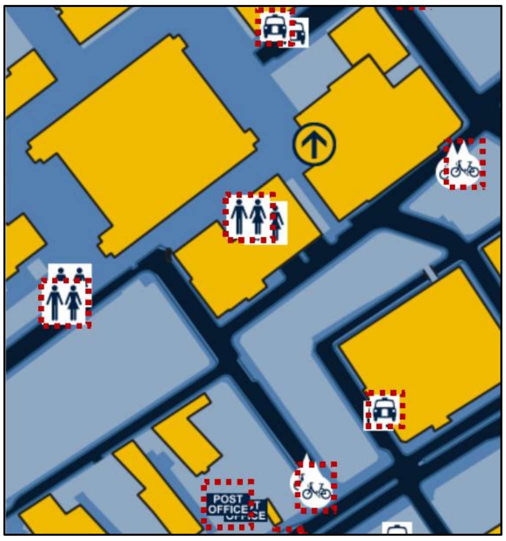


Figure 15: Comparison between the original manually placed icons and the new placed icons positioned by the grid algorithm. The new icons are enclosed within a red dashed square. The area is located in the south-east part of Figure 14.

The algorithm produced a good result, concerning the metrics of legibility and association, as depicted in Figure 14. Nonetheless, some overlaps are present, possibly due to the existence of overlapping in the original data. Moreover, Figure 15 demonstrates that the icons are positioned in close proximity to their original locations, reflecting a high degree of location accuracy. This aligns with the primary production requirement stipulated by T-Kartor (refer to section 2.2.3).

As illustrated in Figure 16, the grid algorithm effectively adheres to numerous of the rules and constraints established by T-Kartor for icon positioning on city wayfinding maps. The displayed examples highlight instances where the performance of the algorithm excels. Specifically compared to the manually placed icons, the first example aligns with the third rule, which states icons should be set away from streets. This is evident as the new placed taxi icon is distinctly placed away from streets. Also, the ability of the algorithm to place icons in the least disturbing position is verified by the second example where the icon is situated in a location that has the least overlap with the segments of the background polygon. Rule number four is also met in the third example where Cycle hire symbol is placed away of the road. Regarding the constraints it is obvious that the last example complies with C2 as it shows the case where an icon is allowed to be placed on top of a text label when the position is better than the prior one (less overlapping).

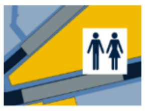
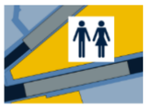
OLD	NEW	DESCRIPTION
		Placed off the streets [R3]
		Not obscuring background inf ormation **Red line is the segment of the background polygon*
		Cycle hire symbol away from road [R4]
		Minimize overlap with text label [C2] **Grey polygon describes the road text label*

Figure 16: Examples of successful icon placement (using the grid algorithm), that comply with cartographers' guidelines as described in section 3.3.5

4.1.4 Metrics values before optimization

The legibility metric was calculated based on a total of 243 icons, 265 text labels for landmarks and 564 for roads that were included in the test area. The analysis revealed that 43 icons overlapped by landmark labels and 61 by road labels. Furthermore, the total number of overlaps between icons was 46. Table 4 displays some examples of the legibility results between the new placed icons and the text labels as well as between the new placed icons themselves. Values range from 0 to 1 with low values indicating high degree of overlapping and thus poorer legibility.

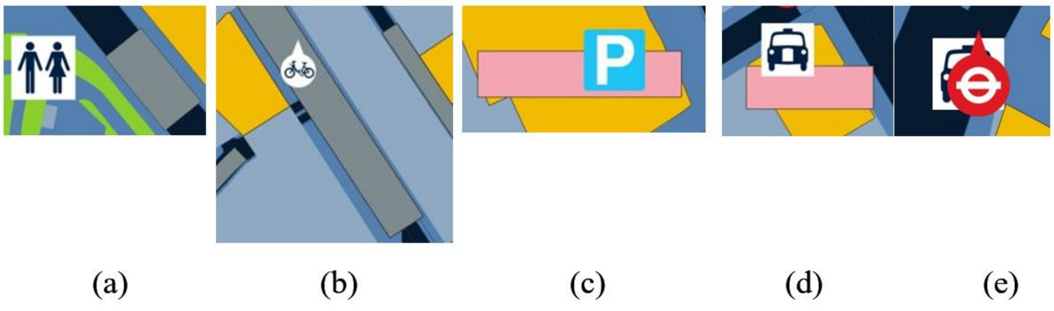


Figure 17: The grey bounding boxes (a, b) represent road labels and the pink (c, d) landmark text labels. For visualization purposes the icons are presented to overlap the text labels.

Table 4: Legibility values for the samples depicted on Figure 17 (a, b, c, d). Values close to 1 indicate a more legible placement, while those to 0 a less legible positioning where icons obscure text labels and other icons.

Figure 17	a	b	c	d	e
Legibility	**1**	**0.2**	**0.4**	**0.8**	**0.02**

For the calculations of the disturbance metric 51,021 break points and 90,771 line segments were utilized. The results showed that 1,200 break points and 3,652 line segments were overlapped by the icons. Figure 18a illustrates examples of icons with low disturbance value (0.1) and Figure 18b with high (0.6). In the first example the icon covers 9 line segments and no break points, while in the second it overlaps with 294 line segments and 21 break points.

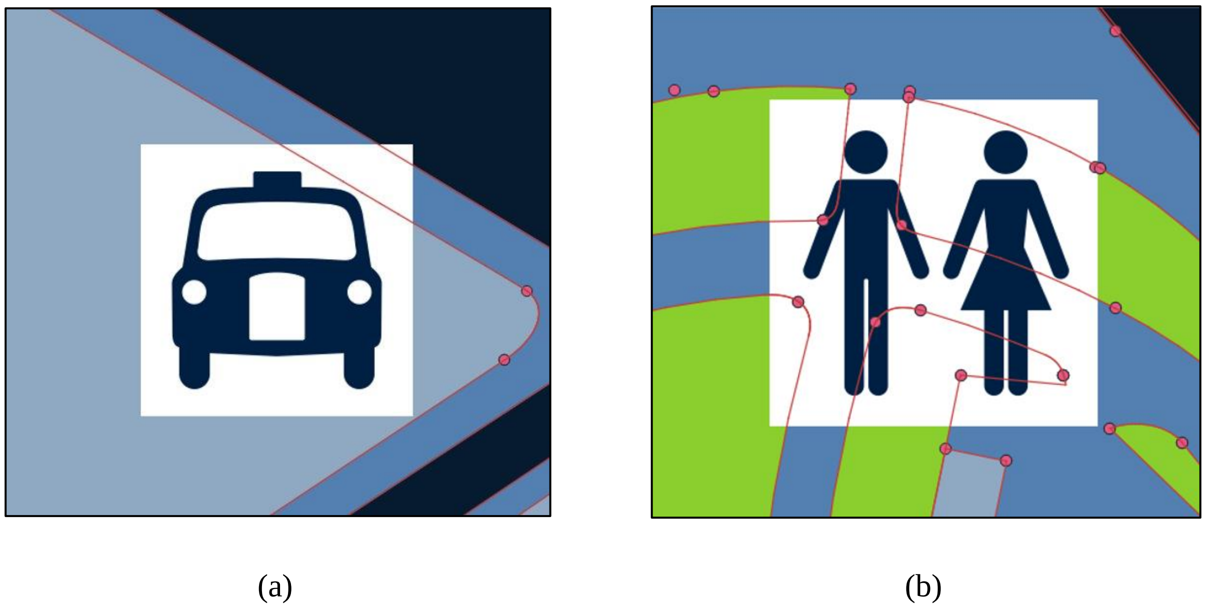


Figure 18: Examples of icons overlapping line segments and break points of background features.

As detailed in section 3.3.3, the association metric was computed differently for icons in the second category compared to those in the third category. Figure 19 provides an illustration of an icon in second category with a good association value of 0.9. This icon is situated closely to its original position and is encompassed partly within its bounding box, designed to contain any of the four potential icons (four positions of the icon while it is rotated around the point it describes) representing that point.

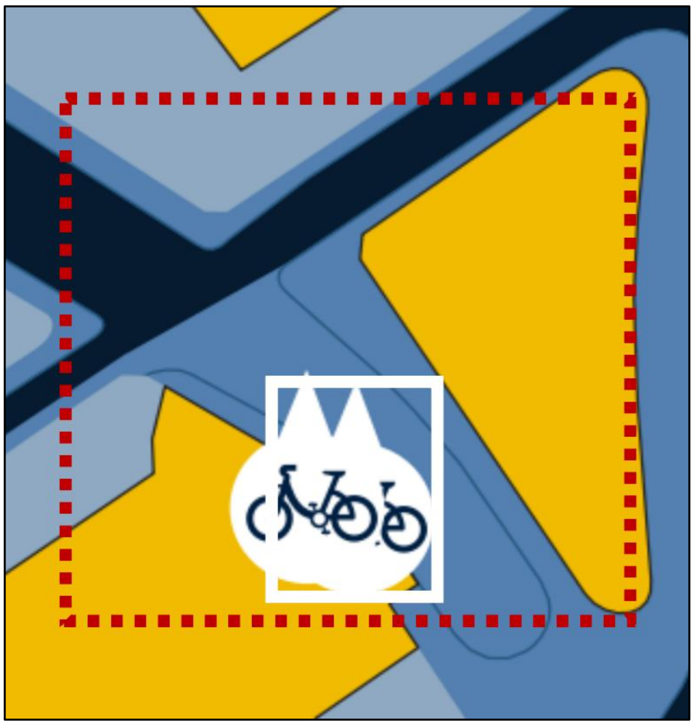


Figure 19: Example of icon highly associated to its original one. With red dushed line is described the outline of the bounding box that surrounds the point of interest and in a white rectangle is the original icon (the one in the right).

4.2 Multi-objective optimization

After running the optimization algorithm (NSGA-II) for 10 generations the results obtained for the quality functions are given in Table 5. The results prove the efforts of algorithm to improve the metrics after each generation. Overall, the algorithm seems to be improving or maintaining performance for the three metrics over time. There is not a consistent trend of worsening for any metric, which is an encouraging observation. In detail, for the disturbance metric, the algorithm has significantly improved (minimize) the values from the first generation to the last. For association metric, there is a noticeable increase, though there are fluctuations in the values, while the legibility metric remains constant.

Table 5: Results from the multi-objective optimization across 10 generations on the three objectives: Disturbance, legibility (text labels), and association. The value represents the total average for all the icons in the dataset.

	Gen 1	Gen 2	Gen 3	Gen 4	Gen 5	Gen 6	Gen 7	Gen 8	Gen 9	Gen 10
Disturbance	0.24	0.27	0.27	0.27	0.27	0.23	0.28	0.28	0.26	0.2
Legibility (text labels)	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96
Association	0.52	0.585	0.585	0.59	0.585	0.57	0.62	0.62	0.61	0.59

The complete table detailing each generation, as provided by the algorithm, can be found in Appendix 3, Table 10.

The definition of the optimal solution in multi-objective optimization, as done by the NSGA-II algorithm, is different from single-objective optimization. There is usually not one single best solution that optimizes all objectives, but rather a set of solutions that represent the trade-offs between the objectives (Petchrompo et al., 2022). These solutions are called Pareto-optimal solutions and form a Pareto front. As seen in Table 6, six Pareto optimal solutions were identified for the multi-objective optimization problem of this study. However, solutions 1 and 2, as well as solutions 3 and 6, are identical, leading to their overlap in the Figure 20 and resulting in visualizing only four distinct dots. Each blue dot on the graph represents a solution that is not dominated by any other solution in the population with respect to the three objectives. These solutions represent trade-offs among the objectives, and no solution on the Pareto front is universally better than the others; the best solution depends on the specific preferences or priorities regarding the objectives. For the specific study, the solution that improves the association metric was chosen as the best one.

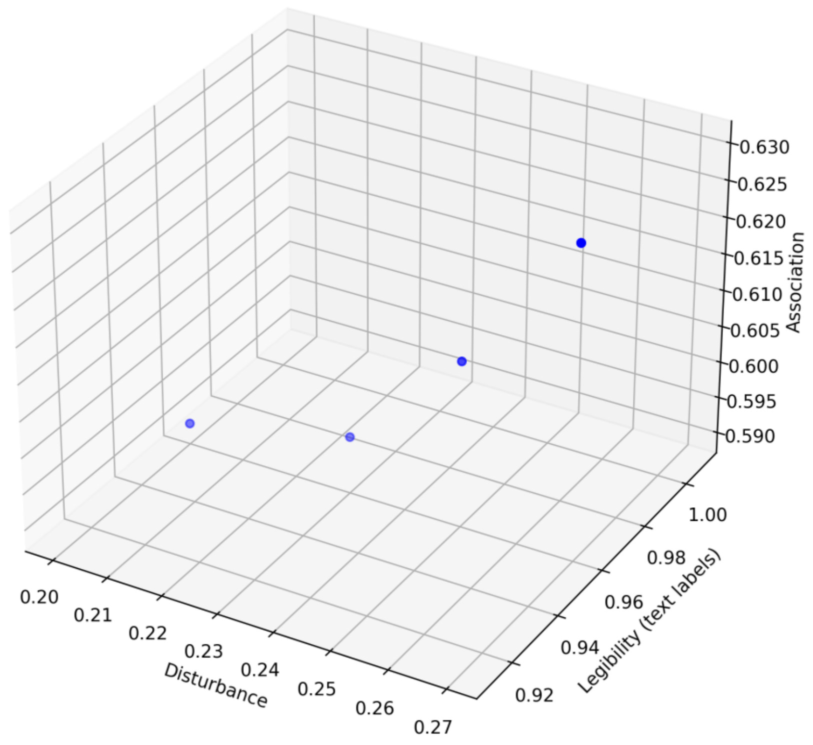
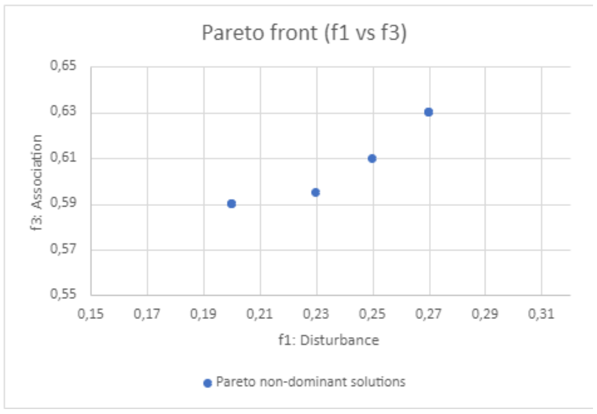
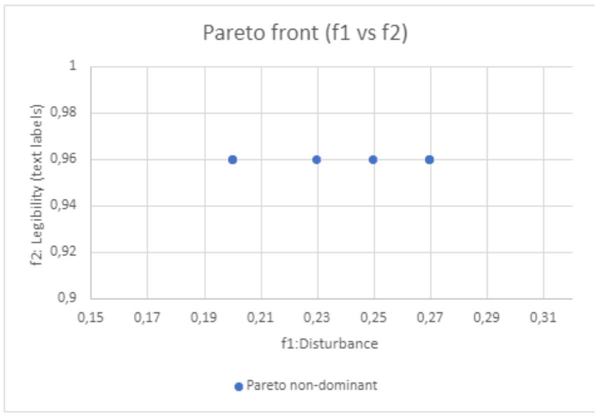


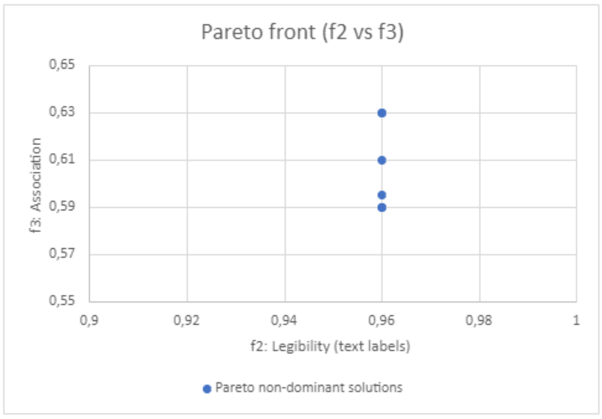
Figure 20: Pareto front of the multi-objective optimization problem in a 3D space, where each axis corresponds to one of the objectives: f1: Disturbance, f2t: Legibility (text labels), and f3: Association. Each blue dot on this graph represents a solution from the Pareto front. Specifically: The x-coordinate of the blue dot represents the value of the objective f1 for that solution, the y-coordinate represents the value of the objective f2t for that solution and the z-coordinate represents the value of the objective f3 for that solution.

For a more straightforward interpretation, the Pareto front is presented as a 2D scatterplot, comparing each metric against the others. In Figure 21a and Figure 21c, it becomes evident that the chosen optimal solution, which maximizes association between the labels and their corresponding location on the map, is the one with a value of 0.63.





(a) (b)



(c)

Figure 21: Pareto front non dominant solutions between objectives f1 and f3, f1 and f2, as well as f2 and f3, visuallized in a 2d scatterplot.

Given the fact that the legibility value is not further optimized, selecting the value that maximizes association inadvertently leads to a value that maximizes disturbance (0.27) rather than minimizing it, as highlighted in Table 6 below. However, compared to the original data (Table 7) it is still considered a better performance.

Table 6: Pareto non-dominant solutions for each metric across 10 generation of the multi-objective optimization algorithm

Solution	f1: Disturbance	f2: Legibility (text labels)	f3: Association
1	0.2	0.96	0.59
2	0.2	0.96	0.59
3	0.27	0.96	0.63
4	0.23	0.96	0.595
5	0.25	0.96	0.61
6	0.27	0.96	0.63

Overall, the algorithm worked as expected with respect to constraints initially set. The initial results from the grid algorithm indicated good icon placement. Hence, the constraints were designed to either enhance the quality metrics or, if improvements were not feasible, to maintain the existing values. In case of maximization of values compared to the ones obtained by the grid algorithm, the constraints also ensure that the values do not exceed the initial ones by more than 0.05. As illustrated in the Table 7 below, the optimization process improved the disturbance and association metrics, while the legibility metric remained unchanged.

Table 7: Metrics values before (using grid algorithm) and after optimization

Metrics	Value before optimization	Value after optimization
Disturbance	0.40	0.27
Legibility (text labels)	0.96	0.96
Association	0.53	0.63

The new positions of the icons are depicted in Figure 22, where a comparison is made on a scatterplot, between the optimized points and the ones generated by the grid algorithm (original).

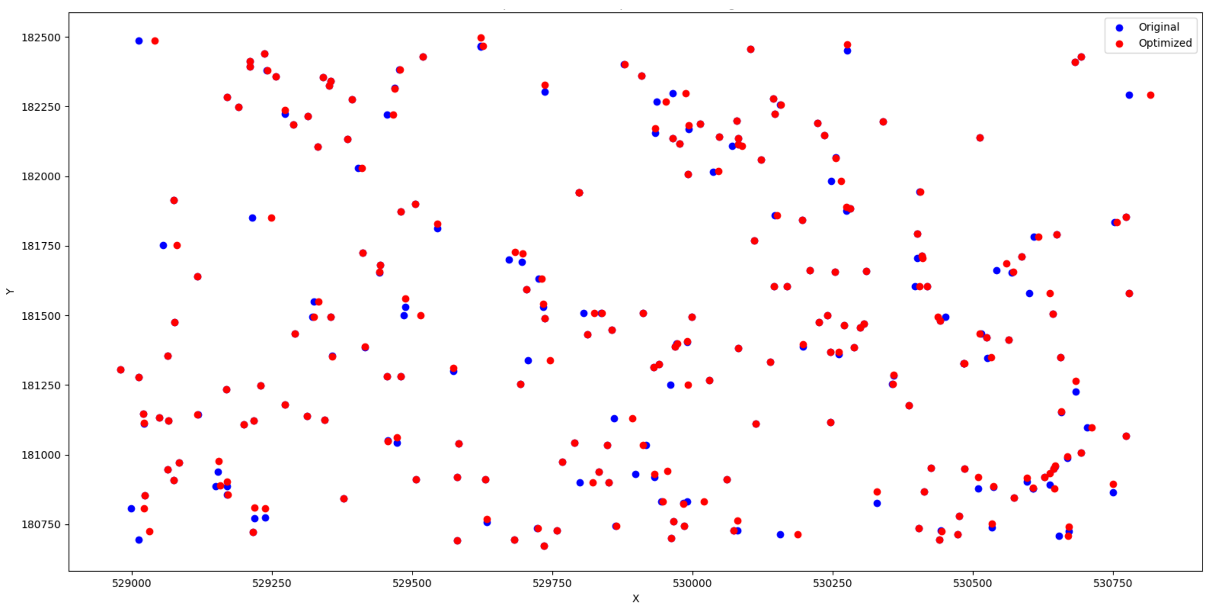


Figure 22: Scatterplot of icons' positions generated by the grid algorithm(in blue color) and new positions (in red color) after the optimization process. The points represent the center of each icon.

4.2.1 Visualization of optimization

For further investigation of the results a visualization of the chosen optimized positions of icons on the map (Figure 23) was created. Compared to Figure 16 it is notable that the distribution of icons in the space changed and that the overlapping with some icons was avoided. That shows how the optimization process managed to solve previous overlapping issues that the grid algorithm did not address successfully (Figure 24).

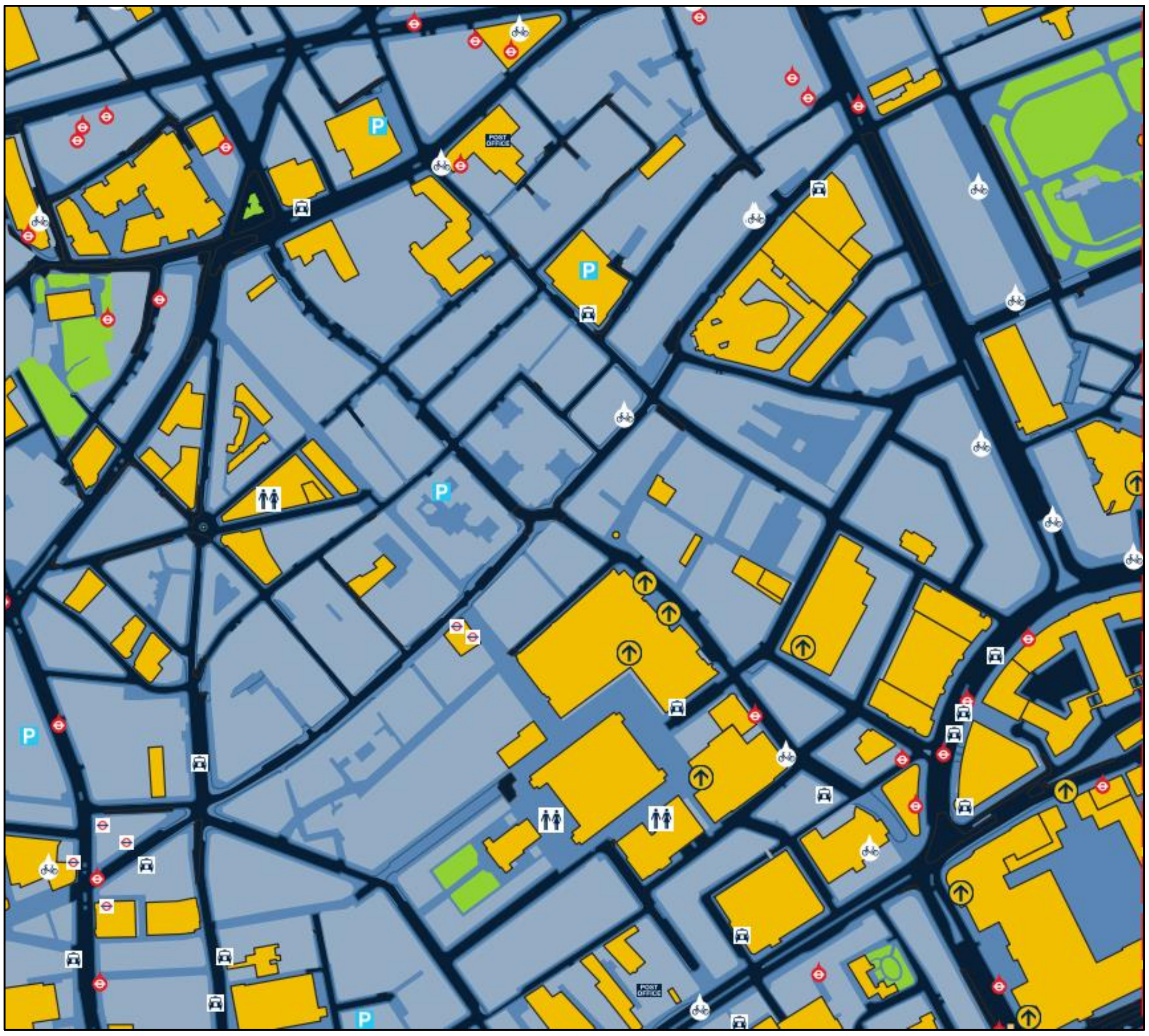


Figure 23: Bottom right quarter of the test area. Icons are placed after optimization.

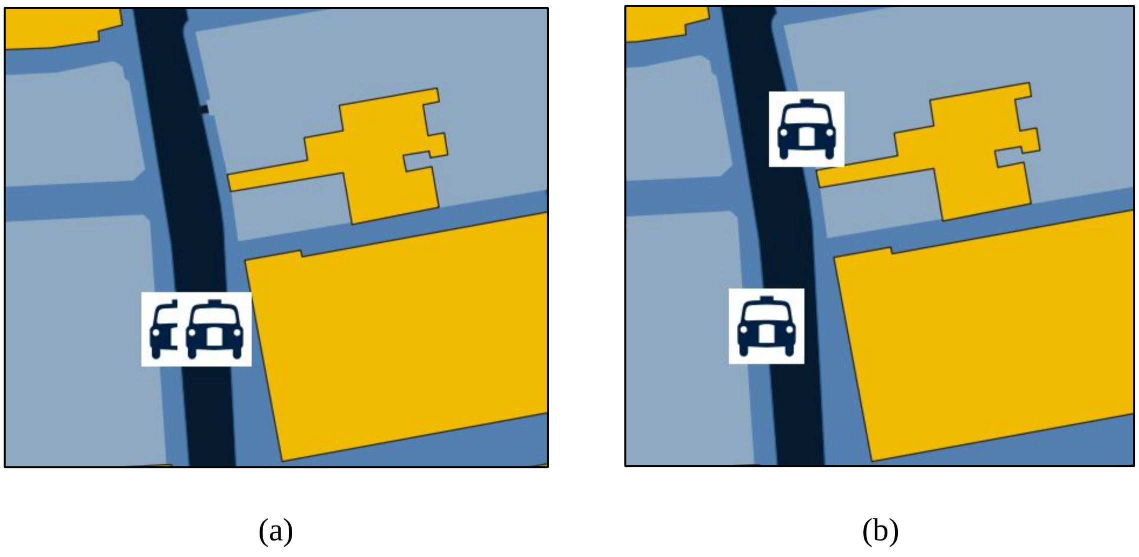


Figure 24: Example of icon placement after the grid algorithm (a) and after the optimization process (b).

Specifically, as illustrated in the Table 8 below, when compared to manually placed icons, the chosen optimized result demonstrated a 7.2% reduction in overlaps between icons and text labels, resulting in 103 overlaps instead of the 111 observed with manual placement. Furthermore, the optimization outperformed the grid algorithm by marginally reducing the overlaps between both icons and text labels (from 104 to 103). Additionally, there was a notable 21.4% decrease in overlaps among the icons themselves, dropping from 28 to 22.

Table 8: Number of overlaps between icons and text labels and icons themselves across different placement methods.

	Original	**Grid Algorithm**	**Optimization**
	Number of overlaps		
Icons and Text labels	111	104	103
Icons themselves	22	28	22

Cases where the overlapping could not be avoided, were primarily observed in areas with high density, having approximately 10 points within 20 square meters (Figure 25). In that case the algorithm could not eliminate the original overlaps but succeeded in distributing the icons such that both the disturbance and association metrics showed improvement.

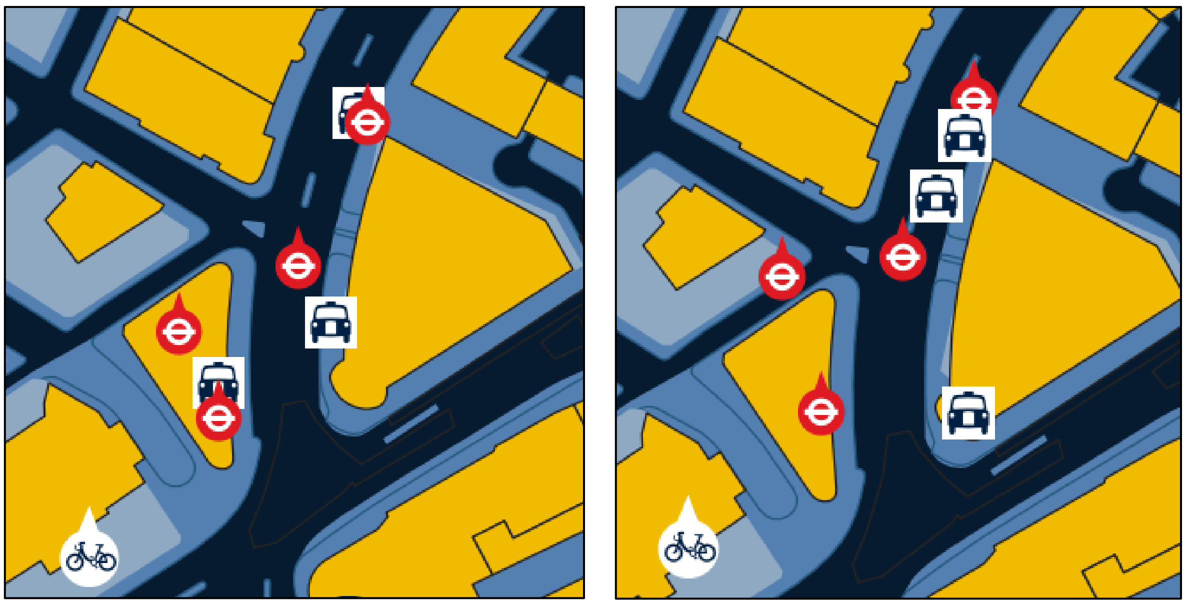

(a) (b)

Figure 25: Comparison of icon placement in a high-density 20 sq. meters area – using grid algorithm (a) and after (b) optimization.

Moreover, Figure 26 below demonstrates that the icons were positioned near their original locations and any disturbance with the background line segments of cartographic features was minimized to enhance the disturbance metric.

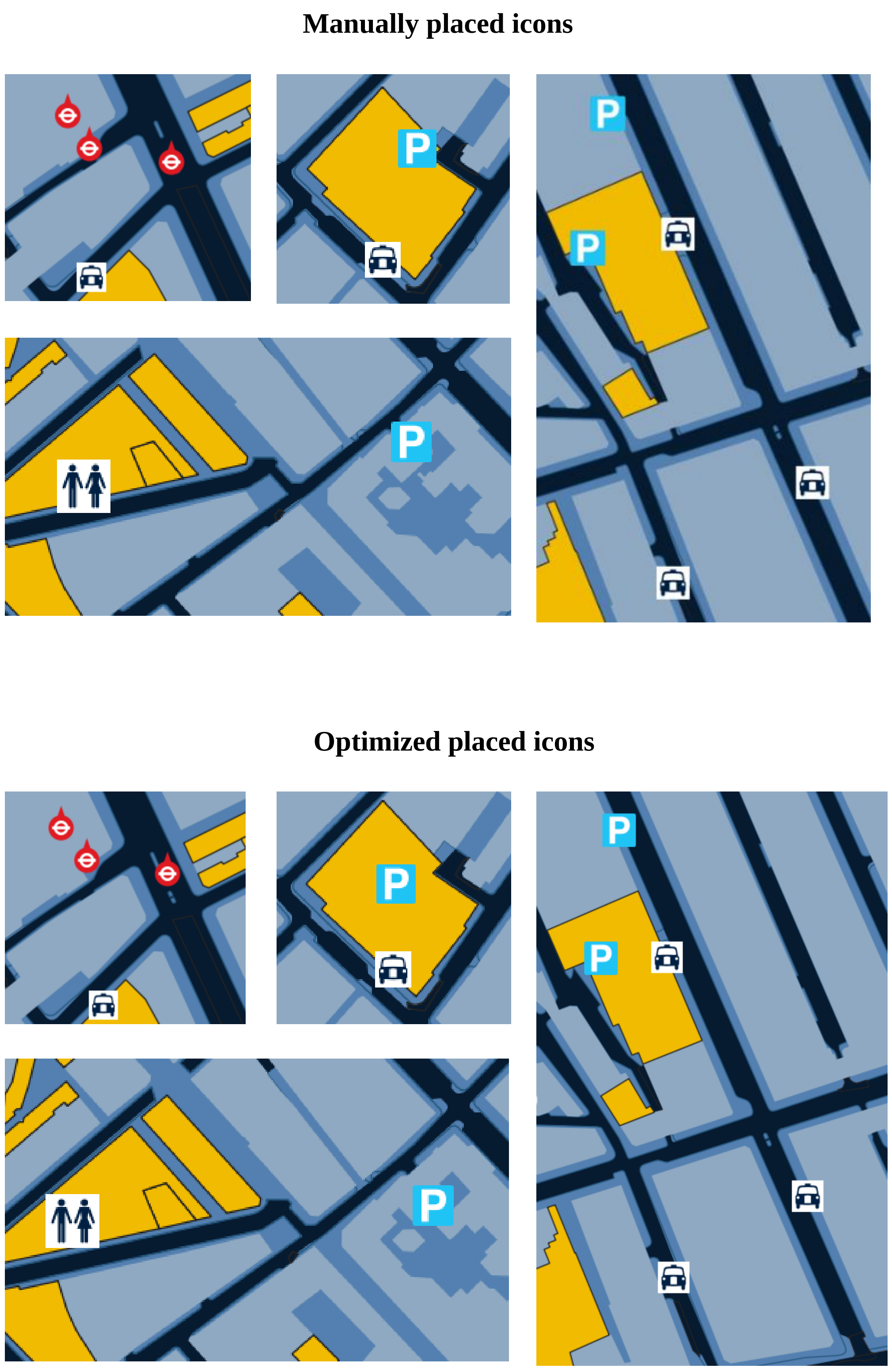


Figure 26: Examples of placement of icons using a manual process and using an optimization process.

4.2.2 Optimized solutions comparison

It is evident from the Pareto front (Figure 21a) that the decision for the best solution relies completely on the map producer (or user) and is based on the needs of the specific problem. In this study the solution emphasizing maximum association was selected. Below is presented a visual comparison between the original placement, the solution that was chosen, the one that minimizes disturbance (by sacrificing association) and the one that balances between the two metrics.

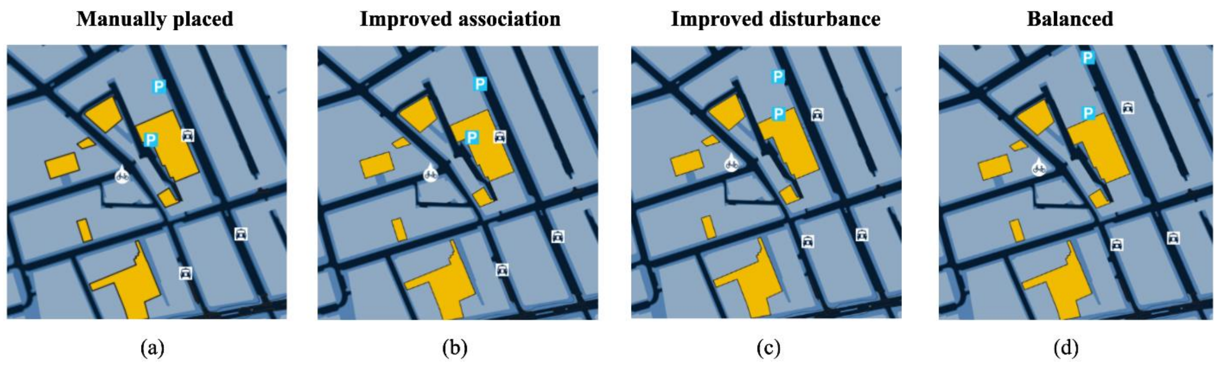


Figure 27: Example of icon placement compared to the original one (a) using different solutions from the pareto front: Solution 6 (b), solution 1 (c) and solution 4 (d).

Figure 27b displays the sixth solution of Table 6 with association value of 0.63 and disturbance 0.27. It is obvious as explained above that the icons are highly associated with the original ones and that the disturbance with background features is improved (compared to 0.46 manually). For instance, the bicycle icon was placed away from the road in a position that did not overlap line segments of the background features.

Figure 27c represents the first solution in Table 6 and indicates an overall improved disturbance (0.20) of the icons in the neighbourhood. Specifically, while the bicycle icon's placement still obscures some background details, the relocation of the top right taxi icon to a spot with fewer background line segment overlaps contributes to the overall reduction in disturbance. However, it is obvious that the icons are placed away from their actual locations, as reflected by the lower association value of 0.59.

Adopting a more balanced approach and choosing a solution (solution 4, Table 6) that maximizes association while at the same time minimizes disturbance leads to a placement as shown in Figure 27d. In this case the disturbance value (0.23) is better than the one from manually placement, which is also obvious from the position of the bottom parking icon (placed in a way that overlaps less line segments of the background polygon), but the icons appear low associated with their real locations.

5. Discussion

5.1 Positions generation

From the methods to generate potential label positions as described in section 2.1 the grid search algorithm was chosen to be used in this study. The 4 position model as or 8 position model as described by researchers could have been used for the icons of second category, whose positions are fixed around the point that they describe. Specifically, on city wayfinding maps produced by T-Kartor, these icons can be placed in four possible positions around the reference point: top, bottom, left, or right. However, this study used the positions generated manually and thus the rotation of the icon around its descriptive point was not explored.

Meanwhile, Slocum et al. (2008) stated that when the label is placed to the left or right of the point it describes can decrease legibility since both the point and the label align on a single line. While this observation holds for text labels, in this study legibility was defined in a different way and the alignment between icons and text labels was not considered. While the above method could be ideal for icons in the second category, the current study involved different kind of icons, as categorized in Table 2. Consequently, a more refined approach was necessary to be adopted.

5.2 Challenges in icon placement

The process of icon placement proved quite different challenges compared to the work of Zhang and Harrie (2006) which provided a different approach by considering both text and icon placement simultaneously. The challenge in the current study was that the icons were already placed and thus there were limitations in the space and more constraints for their placement.

Overall, the new grid search algorithm managed to successfully address a key limitation inherent in the original grid algorithm. Instead of positioning a singular icon, the code was reconfigured to sequentially place all icons in the dataset. Once an icon was positioned, it was instantaneously added to the background features. The aim was to reduce overlapping with the previously positioned icons. As evidenced in Appendix 2, the legibility metric between icons across all layers improved in 6 out of 18 experiments (70 center, 60 center, 80 top right, 70 top right, 80 top left, 70 top left). Notably, this includes the experiment that was selected for further exploration, showcasing the efficacy of the refined approach.

An additional challenge that this study successfully addressed was the one highlighted in the research of Harrie et al (2022), which involved placing labels in high-density areas. The main problem was that it was difficult to define priorities between the labels. This study managed to solve this problem by assigning priority levels to icons based on their categories and then utilizing the grid algorithm to position them sequentially.This was accomplished by dividing the icons into three distinct categories and forcing the algorithm to place them in order. Icons from the second category were placed first, followed by those from the third category. The second category icons were positioned before third category icons because their accurate position was limited as they can only be placed around the point they describe, while the others can be positioned in a larger search space. Notably, icons from the first category maintained fixed positions as detailed in section 3.3.2, with the algorithm leaving their placements unchanged. The results of the metrics were improved compared to the manually placed labelling, by also following the important requirement of city wayfinding maps, which states that all the labels should be present on the map.

Another pivotal challenge was ensuring that icons on the map accurately corresponded to the features they represented. To achieve this, the association metric was given precedence in the multi-objective optimization process, aiming for optimal alignment between newly placed icons and their corresponded features. As Figure 26 and Figure 27b show the mission was successfully addressed and the icons were positioned close to their original ones with respect also to the overall disturbance and legibility metrics. That is particularly important why as the findings of Wang and Worboys (2016) indicated accurate icon positioning fosters trust and reliability in the map's information, empowering users to make informed decisions confidently. This practice enhances spatial awareness, reducing disorientation and facilitating efficient route planning. Accurate icons enable users to align their mental maps with the physical environment, aiding quick decision-making and minimizing search times for destinations. Overall, the practice ensures that users can seamlessly bridge the gap between map representation and reality, resulting in smoother navigation, enhanced user satisfaction, and a heightened sense of familiarity with the city's layout.

5.3 Positions evaluation and selection

For the evaluation of the positions produced by the grid algorithm 4 different functions (metrics) were utilized as defined in section 3.3.3. Instead of relying on a singular objective function, a multi-objective optimization approach was opted to quantify labeling quality. The reason was that the nature of the problem addressed in the current study involved multiple objectives that need to be satisfied simultaneously. Understanding of the trade-offs between the conflicting objectives was necessary in order to decide which solution satisfies better the problem. In the context of this study the aim was to minimize the objective function of disturbance while maximizing the ones for legibility and association of the icons.

The multi-objective optimization process, executed using the NSGA-II algorithm, clearly improved the icons placement by simultaneously balancing the three key metrics. The process was trying to minimize disturbance while maximizing legibility and association. As Table 7 presents, there is a notable reduction in the disturbance metric from an initial value of 0.40 to 0.20 after optimization. This reduction underscores the efficacy of the algorithm in repositioning icons to mitigate disruptions to background cartographic elements. By strategically selecting placements that minimize overlaps and conflicts, the algorithm succeeded in achieving a more harmonious integration of icons within the map's spatial context.

While the disturbance metric demonstrated substantial improvement, the legibility metric remained constant at 0.96 from the first generation. This unchanging value suggests that the algorithm converged to a solution that effectively maximized legibility early in the optimization process. It is worth noting that this metric is improved compared to the value obtained by the grid algorithm (0.88), which was considered as relatively good. Furthermore, when comparing it to manually placed icons, which had a legibility of 0.90, the optimized result is even more impressive.

The association metric also displayed noteworthy progress, with its value increasing from 0.53 before optimization to 0.59 after optimization. This indicates that the algorithm succeeded in creating more coherent spatial relationships between icons and their original positions on the map. Given the higher importance of association, the selection of this value was based on the Pareto front (Figure 21a), and it was primarily based on the superior performance of the solution in this specific metric.

5.4 Production needs

The general production needs as described in section 2.2.3 were met by the optimized placement of icons on the map. Only icon rotation was not addressed, as it was not the main focus of this study. The original rotation of icons was retained, and rotations were not altered or optimized as part of this research. Addressing the other needs:

1. Location Accuracy: The optimized icon placement algorithm succeeded in maintaining location accuracy (Figure 26). The association metric played a pivotal role in achieving this accuracy, ensuring that icons were situated in proximity to their initial positions.

2. Avoiding Overlap: The algorithm's emphasis on disturbance minimization directly contributed to this aspect. Even if eventually as optimal solution was chosen the one that optimized association, the disturbance value was remarkably improved compared to the original data (Table 7).

4. Flexibility in Symbol Placement: Allowing some flexibility in symbol placement related to the metrics and the general rules, was a practical requirement. The algorithm's multi-objective optimization approach balanced the competing objectives of disturbance reduction, association improvement, and legibility enhancement.

5. Avoiding Obstruction: The algorithm's focus on minimizing disturbances and enhancing associations effectively contributed to this need. Icons that obstructed other features were adjusted during the optimization process, allowing for improved clarity while adhering to cartographic guidelines.

5.6 Cartographers' needs

Regarding the specific rules and constraints established by T-Kartor for icon positioning on city wayfinding maps, the grid algorithm effectively adhered to numerous of them (Figure 16). As a result, also the optimized placement of icons on the maps complied with T-Kartor's guidelines. The definition of quality metrics aimed to accommodate the general constraints outlined in section 3.3.5. It is important to note that the first constraint (C1), dictating the presence of all icons on the map, was maintained consistently throughout the study. Specifically, a total of 243 icon features were placed both before and after optimization. Constraints C2 and C3 were implemented and addresed through the legibility metric's efforts to minimize overlap between icons and text labels, as well as between icons themselves. While the third constraint strictly prohibited icon overlap, in certain scenarios, such as high-density areas or initial overlapping, complete avoidance was challenging (Figure 25). The disturbance metric effectively handled C4 by striving to reduce overlap with significant background elements.

In terms of rules, the first (R1) that described the maximum number of icons attached to a point. In the initial data, this rule was naturally satisfied because there was never more than one icon associated with any given point. This makes logical sense because a single location on a map can represent only one specific feature. For instance, a spot marked as a parking area cannot simultaneously be labeled as a bus stop at the exact same location. Moreover, to keep icons off the streets (as the second rule indicated) a higher weight was assigned to layers associated with roads, ensuring the grid algorithm refrained from placing icons on top of road features (as elaborated in section 3.3.2). Rules R3 and R4 were inherently satisfied since, as previously mentioned, icon rotations remained unchanged throughout the various processes.

5.7 Hypothesis validation

The results presented in previous sections provide a comprehensive evaluation of how well the aim of the study was achieved and whether the hypothesis was validated. The experiments conducted, showcased ability of the adopted approach to improve map readability, disturbance, and association while adhering to cartographic rules and aligning with production needs outlined by T-Kartor. The subsequent multi-objective optimization process further enhanced these qualities, albeit with trade-offs that are inherent in multi-objective optimization and resulted in high quality maps that effectively communicate the information. Overall, the findings support the hypothesis that optimizing the grid algorithm based on T-Kartor's constraints and with respect to disturbance, legibility and association can indeed yield maps of higher quality.

5.8 Improvements

Despite the successful accomplishment of the primary research objective, it is important to recognize various aspects that should be taken into account for the purpose of achieving even more refined results in the future.

Computational efficiency

More specifically, it is essential to improve the computational time of optimization algorithm as with the formulation presented in section 3.4.3, it took 30 minutes to complete one generation. The optimization algorithm was executed for a strategically chosen number of generations (10), determined through performance monitoring and empirical experimentation with various values. The computational duration was closely linked to the number of icons within the study area. In this study, 243 icons were used throughout the processes, employing an AMD Ryzen 7 3700U processor. For a more comprehensive approach it would be ideal if all the icons of the whole area were used, a total of 28000 icons.

However, the quality of map labelling cannot solely be based on the number of labels. Consideration must also be given to placement conforming to cartographic requirements such as legibility, association and disturbance. The computational efficiency could potentially be improved through refinement of the objective functions. Furthermore, spatial Indexing was used, which is a data structure designed for quicker searches, to speed up the process, by organizing the data into a search tree which can be quickly traversed to find a particular record (source). In pursuit of efficiency, parallel processing was also adopted, distributing evaluations across eight multi-core processors, which slightly reduced computational time. Nevertheless, there is room for improvement if more efficient coding practices are applied.

Klute et al. (2019) also proposed a semi-automatic labelling method which was proved to overcome the time-consuming issue by initially computing a labelling that has well-placed labels, even though it may not be perfect. Users could then make localized, interactive adjustments based on their criteria. However, as technology advanced and the demands of companies working on map production increased, there is a growing need for this process to be fully automated.

Icon shape

Furthermore, it has been noted that the shape of icons plays a pivotal role in the performance and accuracy of metrics. Throughout this study, a uniform square icon size was employed. It is noticeable, however, that certain icons, such as cycle hire stations or police offices, are not inherently square but are treated as such. This discrepancy can lead to variations in the degree of overlap observed. For instance, a cycle icon, even if it visually appears not to intersect with or overlap background features, may experience increased overlap due to the encompassing square bounding. This scenario can particularly impact the reliability and interpretation of the overlap metric, especially when dealing with a substantial number of icons of this specific shape within a layer. The potential for misinterpretation underscores the importance of understanding and addressing such nuances in metric evaluation.

A potential avenue for improvement is the use of rectangular icons, as suggested by Harrie et al. (2004), which can address the limitations of square shapes. However, it is imperative to align the search space proportionately with the geometry of the icon for optimal results. For instance, square symbols should be aligned within a square space, whereas rectangular symbols should be fitted within a rectangular space. Thus, tailoring the search space to match the icon's geometry becomes essential for high quality outcomes.

For those icons with less defined or irregular shapes, an enclosing bounding box could serve as an effective approach to delineate their spatial constraints. Also, instead of SVG symbols, the (multipolygon) polygon geometry that describe these symbols can be used to accurate shape them. However, implementing such varied geometries introduces a degree of complexity to the coding process. Adjusting the algorithm to accommodate multiple icon shapes necessitates meticulous modifications, elevating the complexity of the procedure. Nonetheless, despite these challenges, adopting a more customized approach to icon placement, tailored to their inherent shapes, holds the potential to yield map outputs that are not only more precise but also visually captivating.

Plugin development

Moreover, the current research provides the foundation for developing a user-centric plugin that can interact with users' preferences can significantly advance this work. Such a plugin would offer a more personalized and dynamic user experience, enabling individuals to adjust the level of optimization based on their unique navigation needs (Figure 27). This not only enhances the user's navigation experience but also provides a level of adaptability and customization that caters to diverse user requirements. Future research and development can focus on the design and usability of this plugin, ensuring it is intuitive and user-friendly, further bridging the gap between advanced optimization techniques and everyday map users.

In conclusion all the aforementioned considerations not only pinpoint the challenges of this study but also open avenues for future exploration in the field of icon placement on city wayfinding maps.

6. Conclusion

This study delved into the intricate challenge of automating icon placement on city wayfinding maps, a task that has historically consumed significant portions of map design time. Through a two-pronged approach involving a grid search algorithm followed by multi-objective optimization, the research aimed to enhance the placement of icons, especially in high-density urban areas.

The results were promising. The grid algorithm demonstrated its capability to sequentially place icons, reducing overlaps and enhancing disturbance. The subsequent multi-objective optimization further refined icon placement, emphasising in association improvement. This optimized placement not only adhered to the cartographic guidelines set by T-Kartor but also significantly improved the overall quality and readability of the maps.

However, like all research, this study had its limitations. The computational efficiency of the optimization algorithm and the uniform square icon size employed present areas for potential improvement. Future endeavours could explore varied icon shapes, more efficient coding practices, and the integration of a plugin more customized to the needs of users.

In essence, this research has paved the way for more refined and user-friendly city wayfinding maps that improve walkability in urban environments. By addressing the challenges of automated icon placement and aligning with cartographic rules, the study has made a significant contribution to the field. As cities continue to grow and urban navigation becomes more complex, the importance of such optimized maps will only increase, making this research both timely and impactful.

7. References

Bartzokas-Tsiompras, A., Photis, Y. N., Tsagkis, P., & Panagiotopoulos, G. (2021). Microscale walkability indicators for fifty-nine European central urban areas: An open-access tabular dataset and a geospatial web-based platform. *Data in Brief*, *36*, 107048. https://doi.org/10.1016/j.dib.2021.107048

Been, K., Daiches, E., & Yap, C. (2006). Dynamic map labeling. *IEEE Transactions on visualization and computer graphics*, *12*(5), 773-780. https://doi.org/https://doi.org/10.1109/TVCG.2006.136

Blank, J., & Deb, K. (2020). Pymoo: Multi-objective optimization in python. *IEEE Access*, *8*, 89497-89509. https://doi.org/10.1109/ACCESS.2020.2990567

Buckley, A. (2012). Make Maps People Want to Look At. Www.esri.com. https://www.esri.com/news/arcuser/0112/make-maps-people-want-to-look-at.html#:~:text=Legibility%20is%20the%20ability%20to

Christensen, J., Marks, J., & Shieber, S. (1995). An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics (TOG)*, *14*(3), 203-232. https://doi.org/https://doi.org/10.1145/212332.212334

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. https://doi.org/10.1109/4235.996017

Dijk, S. V., Kreveld, M. V., Strijk, T., & Wolff, A. (2002). Towards an evaluation of quality for names placement methods. *International Journal of Geographical Information Science*, *16*(7), 641-661. https://doi.org/https://doi.org/10.1080/13658810210138742

Edmondson, S., Christensen, J., Marks, J., & Shieber, S. (1996). A general cartographic labelling algorithm. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *33*(4), 13-24. https://doi.org/10.3138/U3N2-6363-130N-H870

Fonseca, F., Ribeiro, P. J. G., Conticelli, E., Jabbari, M., Papageorgiou, G., Tondelli, S., & Ramos, R. A. R. (2021). Built environment attributes and their influence on walkability. International Journal of Sustainable Transportation, 16(7), 660–679. https://doi.org/10.1080/15568318.2021.1914793

Freeman, H. (1988). An expert system for the automatic placement of names on a geographic map. *Information Sciences*, *45*(3), 367-378. https://doi.org/https://doi.org/10.1016/0020-0255(88)90011-4

Habibian, M., & Hosseinzadeh, A. (2018). Walkability index across trip purposes. Sustainable Cities and Society, 42, 216–225. https://doi.org/10.1016/j.scs.2018.07.005

Haralick, R. M., & Shapiro, L. G. (1985). Image segmentation techniques. *Computer vision, graphics, and image processing*, *29*(1), 100-132. https://doi.org/https://doi.org/10.1016/S0734-189X(85)90153-7

Harrie, L., Oucheikh, R., Nilsson, Å., Oxenstierna, A., Cederholm, P., Wei, L., Richter, K.-F., & Olsson, P. (2022). Label Placement Challenges in City Wayfinding Map Production—Identification and Possible Solutions. *Journal of Geovisualization and Spatial Analysis*, *6*(1), 16. https://doi.org/https://doi.org/10.1007/s41651-022-00115-z

Harrie, L., Stigmar, H., Koivula, T., & Lehto, L. (2004). An algorithm for icon labelling on a real-time map. *Developments in Spatial Data Handling*, 493-507. https://doi.org/https://doi.org/10.1007/3-540-26772-7_38

Harrie, L., Zhang, Q., & Ringberg, P. (2005). A case study of combined text and icon placement. Proceedings of the International Cartographic Conference

Hirsch, S. A. (1982). An algorithm for automatic name placement around point data. *The American Cartographer*, *9*(1), 5-17. https://doi.org/10.1559/152304082783948367

Hong, F., Zuxun, Z., & Daosheng, D. (2005). Quality evaluation model for map labeling. *Geo-spatial Information Science*, *8*(1), 72-78. https://doi.org/https://doi.org/10.1007/BF02826996

Horváth, I. (2007). Comparison of three methodological approaches of design research. DS 42: Proceedings of ICED 2007, the 16th International Conference on Engineering Design, Paris, France, 28.-31.07. 2007,

Imhof, E. (1975). Positioning names on maps. *The American Cartographer*, *2*(2), 128-144. https://doi.org/https://doi.org/10.1559/152304075784313304

Kato, T., & Imai, H. (1988). The NP-completeness of the character placement problem of 2 or 3 degrees of freedom. Record of Joint Conference of Electrical and Electronic Engineers in Kyushu,

Klau, G. W., & Mutzel, P. (2003). Optimal labeling of point features in rectangular labeling models. *Mathematical Programming*, *94*, 435-458. https://doi.org/http://dx.doi.org/10.1007/s10107-002-0327-9

Klute, F., Li, G. P., Loffler, R., Nollenburg, M., & Schmidt, M. (2019). Exploring Semi-Automatic Map Labeling. *27th ACM Sigspatial International Conference on Advances in Geographic Information Systems*, 13-22. https://doi.org/10.1145/347146.3359359

Knuth, D. E. (1974). Postscript about NP-hard problems. *ACM SIGACT News*, *6*(2), 15-16. https://dl.acm.org/doi/pdf/10.1145/1008304.1008305

Konak, A., Coit, D. W., & Smith, A. E. (2006). Multi-objective optimization using genetic algorithms: A tutorial. Reliability Engineering & System Safety, 91(9), 992–1007. https://doi.org/10.1016/j.ress.2005.11.018

Lu, F., Deng, J., Li, S., & Deng, H. (2019). A hybrid of differential evolution and genetic algorithm for the Multiple Geographical Feature Label Placement Problem. *ISPRS International Journal of Geo-Information*, *8*(5), 237. https://doi.org/https://doi.org/10.3390/ijgi8050237

Marks, J., & Shieber, S. M. (1991). *The computational complexity of cartographic label placement*. Citeseer. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f9c4ea02dd5c3e3b00a41843a62f0cbef3c65719

Oucheikh., R., Harrie., L. (2023). *A feasibility study of applying generative deep learning models for map labeling*. [Manuscript submitted for publication]

Pal, N. R., & Pal, S. K. (1993). A review on image segmentation techniques. *Pattern recognition*, *26*(9), 1277-1294. https://doi.org/https://doi.org/10.1016/0031-3203(93)90135-j

Petchrompo, S., Coit, D. W., Brintrup, A., Wannakrairot, A., & Parlikad, A. K. (2022). A review of Pareto pruning methods for multi-objective optimization. *Computers & Industrial Engineering*, 108022. https://doi.org/10.1016/j.cie.2022.108022

Robinson, A. H. (1986). The Look of Maps: An Examination of Cartographic Design. *The American Cartographer*, *13*(3), 280–280. https://doi.org/10.1559/152304086783899881

Robinson, A., Morrison, J., Muehrcke, P., Kimerling, A., & Guptill, S. (1995). Elements of Cartography. New York, Chichester, Brisbane, Toronto, Singapore, John Willey & Sons. In: Inc.

Roth, R. E., Brewer, C. A., & Stryker, M. S. (2011). A typology of operators for maintaining legible map designs at multiple scales. Cartographic Perspectives, 68, 29–64. https://doi.org/10.14714/cp68.7

Rylov, M., & Reimer, A. (2017). A Practical Algorithm for the External Annotation of Area Features. *Cartographic Journal*, *54*(1), 61-76. https://doi.org/10.1179/1743277414y.0000000091

Rylov, M. A., & Reimer, A. W. (2014). A comprehensive multi-criteria model for high cartographic quality point-feature label placement. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *49*(1), 52-68. https://doi.org/10.3138/carto.49.1.2137

Rylov, M. A., & Reimer, A. W. (2015). Improving label placement quality by considering basemap detail with a raster-based approach. *Geoinformatica*, *19*, 463-486. https://doi.org/https://doi.org/10.1007/s10707-014-0214-6

Salian, A. (2022, February 23). A Gentle Introduction to Multi-Objective Optimisation. Stories | Codemonk - Product Design & Engineering. https://codemonk.in/blog/a-gentle-introduction-to-multi-objective-optimization/

Schrijver, A. (2003). *Combinatorial optimization: polyhedra and efficiency* (Vol. 24). Springer Berlin.

SimilarWeb. (2023). Similarweb.com - Digital World Market Intelligence Platform. SimilarWeb.com. https://www.similarweb.com/

Skopeliti, A., & Stamou, L. (2019). Online Map Services: Contemporary Cartography or a New Cartographic Culture? ISPRS International Journal of Geo-Information, 8(5), 215. https://doi.org/10.3390/ijgi8050215

Slocum, T. A., McMaster, R. M., Kessler, F. C., Howard, H. H., & Mc Master, R. B. (2008). Thematic cartography and geographic visualization.

Strijk, T., & van Kreveld, M. (2002). Practical extensions of point labeling in the slider model. *Geoinformatica*, *6*(2), 181-197. https://doi.org/https://doi.org/10.1145/320134.320148

Transport, V., & Litman, T. A. (2009). Pedestrian and bicycle planning: a guide to best practices. The Institute.

van Dijk, S. (2001). *Genetic algorithms for map labeling* [Ph.D. thesis, Utrecht University, Netherlands]. https://core.ac.uk/download/pdf/39700283.pdf

van Kreveld, M., Strijk, T., & Wolff, A. (1999). Point set labeling with sliding labels. Proceedings of the fourteenth annual symposium on Computational geometry

Visvalingam, M. (1990). Trends and concerns in digital cartography. *Computer-Aided Design*, *22*(3), 115-130. https://doi.org/https://doi.org/10.1016/0010-4485(90)90070-S

Walk & the City Center. (n.d.). Geochoros.survey.ntua.gr. Retrieved August 30, 2023, from http://geochoros.survey.ntua.gr/walkandthecitycenter/chart

Wang, F.-S., & Chen, L.-H. (2013). Heuristic optimization. *Encyclopedia of Systems Biology, Springer, New York*, 885-885. https://doi.org/https://doi.org/10.1007/978-1-4419-9863-7_411

Wang, J., & Worboys, M. (2016). Pedestrian Navigation Aids, Spatial Knowledge and Walkability. International Conference on GIScience Short Paper Proceedings, 1(1). https://doi.org/10.21433/b3114b58k9tp

Wood, C. H. (2000). A descriptive and illustrated guide for type placement on small scale maps. *The Cartographic Journal*, *37*(1), 5-18. https://doi.org/https://doi.org/10.1179/caj.2000.37.1.5

Yamamoto, M., Camara, G., & Lorena, L. A. N. (2002). Tabu search heuristic for point-feature cartographic label placement. *Geoinformatica*, *6*, 77-90. https://doi.org/https://doi.org/10.1023/A:1013720231747

Yoeli, P. (1972). The logic of automated map lettering. *The Cartographic Journal*, *9*(2), 99-108. https://doi.org/10.1179/000870472787352505

Yusoff, Y., Ngadiman, M. S., & Zain, A. M. (2011). Overview of NSGA-II for Optimizing Machining Process Parameters. Procedia Engineering, 15, 3978–3983. https://doi.org/10.1016/j.proeng.2011.08.745

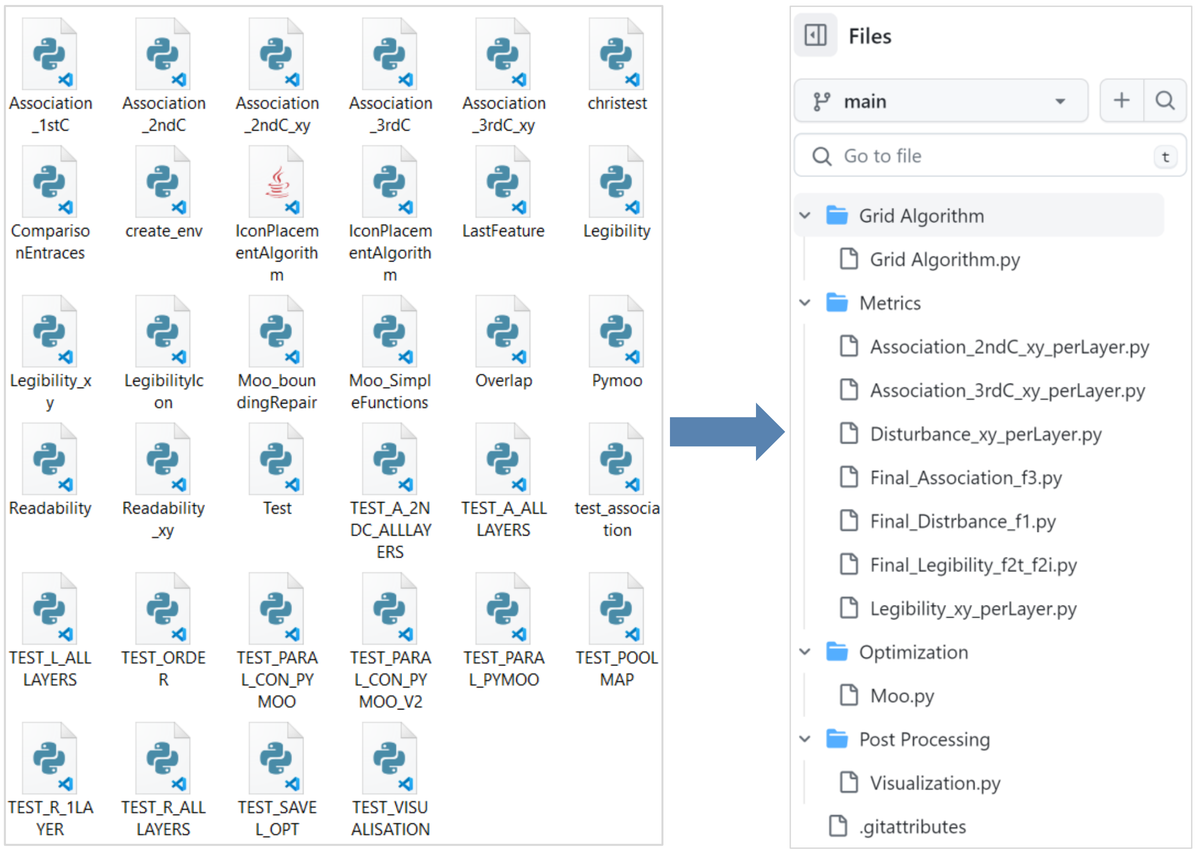
Zhang, Q., & Harrie, L. (2006). Placing text and icon labels simultaneously: A real-time method. *Cartography and Geographic Information Science*, *33*(1), 53-64. https://doi.org/https://doi.org/10.1559/152304006777323127

Zoraster, S. (1997). Practical results using simulated annealing for point feature label placement. *Cartography and Geographic Information Systems*, *24*(4), 228-238. https://doi.org/https://doi.org/10.1559/152304097782439259

8. Appendices

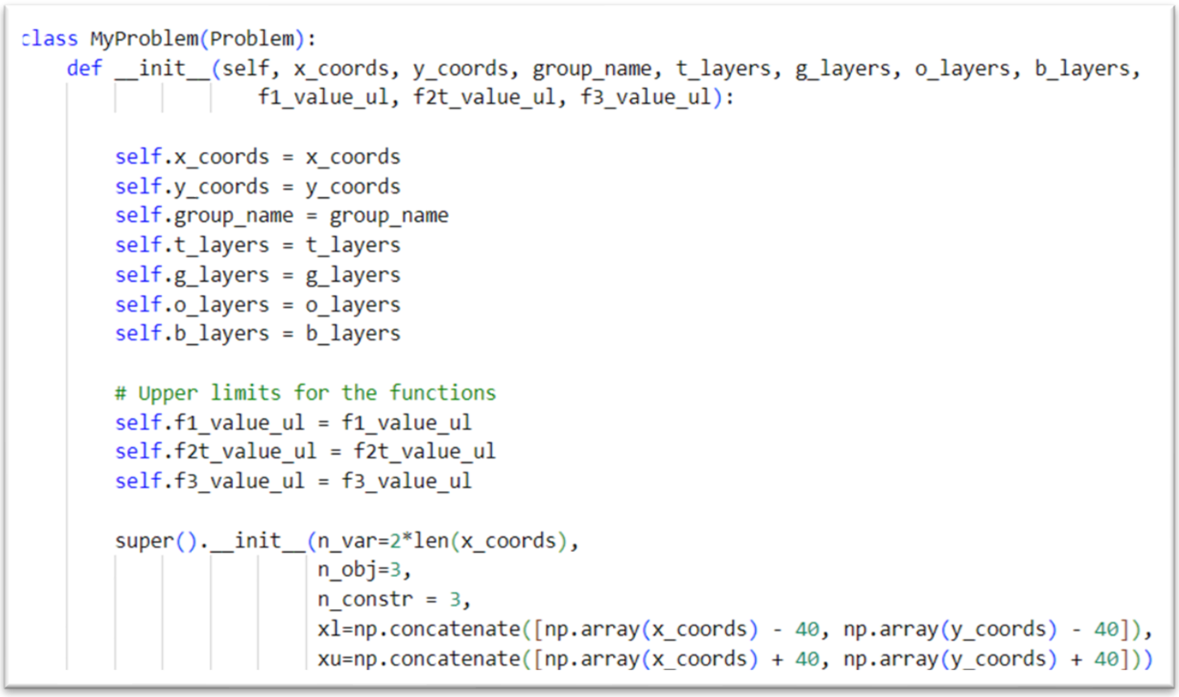
Appendix 1. Supplementary figures for section 3.4

This appendix showcases the evolution of scripts from the initial phase of this research to the final versions in the GitHub repository. Additionally, it provides code snippets from the implementation of the multi-objective optimization, with main focus on problem initialization.



(a) (b)

Figure 28: Initial batch of 34 Python files and the final scripts (10 files) developed in this research (GitHub repository). The final scripts are based on the initial attempts made. Especially for the objective functions where the functions were developed to return first the value of the metric for each icon feature until they were adjusted to get the same arguments and return the average value for the group of icon layers (all icons in the dataset). Several attempts were also made to improve the computation time by implementing different strategies (I.e.: parallel processing, poolmap)



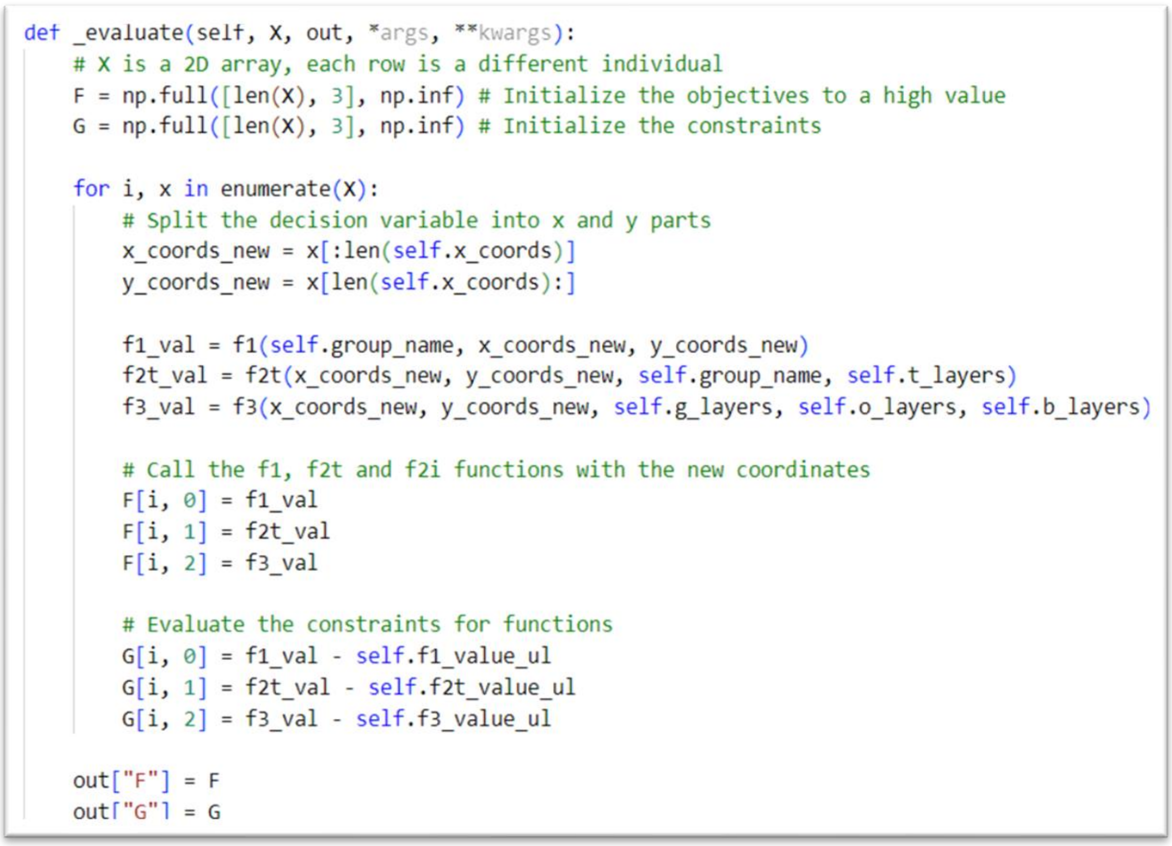
```
class MyProblem(Problem):
    def __init__(self, x_coords, y_coords, group_name, t_layers, g_layers, o_layers, b_layers,
                 f1_value_ul, f2t_value_ul, f3_value_ul):

        self.x_coords = x_coords
        self.y_coords = y_coords
        self.group_name = group_name
        self.t_layers = t_layers
        self.g_layers = g_layers
        self.o_layers = o_layers
        self.b_layers = b_layers

        # Upper limits for the functions
        self.f1_value_ul = f1_value_ul
        self.f2t_value_ul = f2t_value_ul
        self.f3_value_ul = f3_value_ul

        super().__init__(n_var=2*len(x_coords),
                         n_obj=3,
                         n_constr = 3,
                         xl=np.concatenate([np.array(x_coords) - 40, np.array(y_coords) - 40]),
                         xu=np.concatenate([np.array(x_coords) + 40, np.array(y_coords) + 40]))
```

Figure 29: Code snippet from the first function of the MyProblem class



```
def _evaluate(self, X, out, *args, **kwargs):
    # X is a 2D array, each row is a different individual
    F = np.full([len(X), 3], np.inf) # Initialize the objectives to a high value
    G = np.full([len(X), 3], np.inf) # Initialize the constraints

    for i, x in enumerate(X):
        # Split the decision variable into x and y parts
        x_coords_new = x[:len(self.x_coords)]
        y_coords_new = x[len(self.x_coords):]

        f1_val = f1(self.group_name, x_coords_new, y_coords_new)
        f2t_val = f2t(x_coords_new, y_coords_new, self.group_name, self.t_layers)
        f3_val = f3(x_coords_new, y_coords_new, self.g_layers, self.o_layers, self.b_layers)

        # Call the f1, f2t and f2i functions with the new coordinates
        F[i, 0] = f1_val
        F[i, 1] = f2t_val
        F[i, 2] = f3_val

        # Evaluate the constraints for functions
        G[i, 0] = f1_val - self.f1_value_ul
        G[i, 1] = f2t_val - self.f2t_value_ul
        G[i, 2] = f3_val - self.f3_value_ul

    out["F"] = F
    out["G"] = G
```

Figure 30: Code snippet from the evaluation function of the MyProblem class

Appendix 2. Grid algorithm evaluation

Below is given a table with all the eighteen experiments conducted using default parameters and the objective functions for evaluating the performance of the grid algorithm. The aim was to choose the best combination of parameters that performs better in the association. Moreover, the visual representation of the experiments' results is given in a figure.

Table 9: Experiments conducted using different combinations of searchDistanceGround variable and p.orgX, p.orgY. The values for disturbance, legibility and association metrics are given for each layer. The layers are also split into categories and the subaverage value of each experiment in each category is calculated. Additionally, the overall average metric value is calculated across all layers in each experiment. The legibility metric was calculated in three ways: between the new icons and road text labels, between the new icons and landmarks text labels, and among the new icons themselves. Parts of the maps created for each experiment are presented in Figure 31.

		OLD	NEW																	
Parameters (searchDistanceGround, oreX oreY):			90, center	80, center	70, center	60, center	50, center	40, center	90, top-right	80, top-right	70, top-right	60, top-right	50, top-right	40, top-right	90, botom-left	80, botom-left	70, botom-left	60, botom-left	50, botom-left	40, botom-left

DISTURBANCE METRIC

Category	Layer	Disturbance factor (avg)																		
2	L_ LMF_Bus_Stops_P_new	0.34	0.19	0.13	0.18	0.22	0.20	0.20	0.21	0.15	0.18	0.20	0.34	0.34	0.16	0.18	0.23	0.19	0.32	0.32
	L_ LMF_CH_Stations_P_new	0.35	0.28	0.26	0.36	0.34	0.29	0.29	0.41	0.35	0.27	0.26	0.41	0.41	0.30	0.37	0.24	0.26	0.29	0.29
Sub Avg		**0.35**	**0.24**	**0.20**	**0.27**	**0.28**	**0.25**	**0.25**	**0.31**	**0.25**	**0.23**	**0.23**	**0.38**	**0.38**	**0.23**	**0.28**	**0.24**	**0.23**	**0.31**	**0.31**
3	L_ LMF_Car_Parks_P_new	0.44	0.24	0.31	0.25	0.34	0.30	0.30	0.18	0.30	0.33	0.32	0.26	0.26	0.26	0.25	0.22	0.28	0.28	0.28
	L_ LMF_Post_Offices_P_new	0.53	0.30	0.52	0.62	0.50	0.42	0.42	0.34	0.36	0.54	0.31	0.34	0.34	0.47	0.46	0.49	0.61	0.61	0.61
	L_ LMF_Taxi_ranks_P_new	0.43	0.17	0.32	0.33	0.43	0.37	0.37	0.22	0.17	0.17	0.19	0.26	0.26	0.20	0.23	0.20	0.26	0.26	0.26
	L_ LMF_Toilets_P_new	0.68	0.48	0.42	0.66	0.54	0.58	0.58	0.47	0.43	0.58	0.58	0.69	0.69	0.31	0.33	0.37	0.56	0.48	0.48
Sub Avg		**0.56**	**0.30**	**0.39**	**0.47**	**0.45**	**0.42**	**0.42**	**0.30**	**0.32**	**0.41**	**0.35**	**0.39**	**0.39**	**0.31**	**0.32**	**0.32**	**0.43**	**0.41**	**0.41**
Total Avg		**0.46**	**0.28**	**0.33**	**0.40**	**0.40**	**0.36**	**0.36**	**0.31**	**0.29**	**0.35**	**0.31**	**0.38**	**0.38**	**0.28**	**0.30**	**0.29**	**0.36**	**0.37**	**0.37**

LEGIBILITY METRIC

Roads Text Labels

Category	Layer	Legibility factor (avg)																		
2	L_ LMF_Bus_Stops_P_new	0.88	0.94	0.96	0.97	0.96	0.96	0.96	0.93	0.93	0.92	0.91	0.90	0.90	0.96	0.96	0.93	0.93	0.94	0.94
	L_ LMF_CH_Stations_P_new	0.92	0.95	0.93	0.93	0.94	0.95	0.95	0.94	0.93	0.94	0.94	0.94	0.94	0.92	0.91	0.91	0.91	0.91	0.91
Sub Avg		**0.90**	**0.95**	**0.95**	**0.95**	**0.95**	**0.96**	**0.96**	**0.94**	**0.93**	**0.93**	**0.93**	**0.92**	**0.92**	**0.94**	**0.94**	**0.92**	**0.92**	**0.93**	**0.93**
3	L_ LMF_Car_Parks_P_new	0.99	0.95	0.93	0.99	0.98	1.00	1.00	0.99	0.96	0.94	0.97	0.94	0.94	0.93	0.98	0.97	0.96	0.95	0.95
	L_ LMF_Post_Offices_P_new	1.00	0.91	0.88	0.92	0.98	0.99	0.99	0.84	0.90	0.79	0.88	0.88	0.88	0.90	0.91	0.94	0.94	0.92	0.92
	L_ LMF_Taxi_ranks_P_new	0.96	1.00	0.98	0.95	0.96	0.96	0.96	0.98	0.96	0.95	0.93	0.93	0.93	0.97	0.98	0.95	0.96	0.96	0.96
	L_ LMF_Toilets_P_new	0.99	0.93	0.94	0.97	1.00	0.98	0.98	0.99	0.99	1.00	1.00	0.97	0.97	0.94	0.94	0.93	0.94	0.94	0.94
Sub Avg		**0.99**	**0.95**	**0.93**	**0.96**	**0.98**	**0.98**	**0.98**	**0.95**	**0.95**	**0.92**	**0.95**	**0.93**	**0.93**	**0.94**	**0.95**	**0.95**	**0.95**	**0.94**	**0.94**
Total Avg		**0.96**	**0.95**	**0.94**	**0.96**	**0.97**	**0.97**	**0.97**	**0.95**	**0.95**	**0.92**	**0.94**	**0.93**	**0.93**	**0.94**	**0.95**	**0.94**	**0.94**	**0.94**	**0.94**

Landmarks Text Labels

Category	Layer	Legibility factor (avg)																		
2	L_ LMF_Bus_Stops_P_new	0.96	0.89	0.90	0.93	0.93	0.93	0.93	0.94	0.94	0.94	0.94	0.93	0.93	0.90	0.92	0.92	0.93	0.94	0.94
	L_ LMF_CH_Stations_P_new	0.97	0.94	0.95	0.95	0.95	0.95	0.95	0.96	0.98	0.97	0.97	0.98	0.98	0.88	0.88	0.90	0.90	0.90	0.90
Sub Avg		**0.97**	**0.92**	**0.93**	**0.94**	**0.94**	**0.94**	**0.94**	**0.95**	**0.96**	**0.96**	**0.96**	**0.96**	**0.96**	**0.89**	**0.90**	**0.91**	**0.92**	**0.92**	**0.92**
3	L_ LMF_Car_Parks_P_new	0.99	0.89	0.97	0.93	0.93	0.96	0.96	0.89	0.91	0.96	0.98	0.98	0.98	0.87	0.89	0.86	0.88	0.90	0.90
	L_ LMF_Post_Offices_P_new	0.98	0.81	0.90	0.96	0.97	0.97	0.97	0.99	1.00	0.96	0.91	0.91	0.91	0.99	0.96	1.00	0.98	0.94	0.94
	L_ LMF_Taxi_ranks_P_new	1.00	0.83	0.88	0.94	0.98	1.00	1.00	0.83	0.84	0.86	0.83	0.86	0.86	0.83	0.83	0.86	0.89	0.92	0.92
	L_ LMF_Toilets_P_new	0.99	0.96	0.93	0.99	1.00	0.97	0.97	0.89	0.90	0.93	0.86	0.89	0.89	0.86	0.82	0.78	0.82	0.87	0.87
Sub Avg		**0.99**	**0.87**	**0.92**	**0.96**	**0.97**	**0.98**	**0.98**	**0.90**	**0.91**	**0.93**	**0.90**	**0.91**	**0.91**	**0.89**	**0.88**	**0.88**	**0.89**	**0.91**	**0.91**
Total Avg		**0.98**	**0.89**	**0.92**	**0.95**	**0.96**	**0.96**	**0.96**	**0.92**	**0.93**	**0.94**	**0.92**	**0.93**	**0.93**	**0.89**	**0.88**	**0.89**	**0.90**	**0.91**	**0.91**

Text Labels (road&landmarks)

Avg		**0.97**	**0.92**	**0.93**	**0.95**	**0.97**	**0.97**	**0.97**	**0.93**	**0.94**	**0.93**	**0.93**	**0.93**	**0.93**	**0.91**	**0.92**	**0.91**	**0.92**	**0.92**	**0.92**

Icons Labels

Category	Layer	Legibility factor (avg)																		
2	L_ LMF_Bus_Stops_P_new	0.86	0.57	0.53	0.72	0.65	0.70	0.70	0.60	0.69	0.74	0.60	0.70	0.70	0.73	0.73	0.65	0.61	0.70	0.70
	L_ LMF_CH_Stations_P_new	0.76	0.76	0.74	0.73	0.91	0.87	0.87	0.73	0.81	0.91	0.75	0.87	0.87	0.81	0.88	0.91	0.92	0.87	0.87
Sub Avg		**0.81**	**0.67**	**0.64**	**0.73**	**0.78**	**0.79**	**0.79**	**0.67**	**0.75**	**0.83**	**0.68**	**0.79**	**0.79**	**0.77**	**0.81**	**0.78**	**0.77**	**0.79**	**0.79**
3	L_ LMF_Car_Parks_P_new	1.00	0.54	0.24	0.89	0.32	0.47	0.47	0.28	0.64	0.22	0.40	0.47	0.47	0.58	0.74	0.47	0.22	0.47	0.47
	L_ LMF_Post_Offices_P_new	1.00	0.50	0.68	0.71	0.85	0.68	0.68	0.66	0.82	0.82	0.51	0.68	0.68	0.72	0.64	0.54	0.76	0.68	0.68
	L_ LMF_Taxi_ranks_P_new	0.68	0.50	0.30	0.71	0.61	0.69	0.69	0.32	0.82	0.80	0.73	0.69	0.69	0.42	0.70	0.86	0.81	0.69	0.69
	L_ LMF_Toilets_P_new	0.45	1.00	1.00	1.00	0.96	1.00	1.00	1.00	0.74	0.93	1.00	1.00	1.00	0.77	0.92	0.88	1.00	1.00	1.00
Sub Avg		**0.78**	**0.64**	**0.56**	**0.83**	**0.69**	**0.71**	**0.71**	**0.57**	**0.76**	**0.69**	**0.66**	**0.71**	**0.71**	**0.62**	**0.75**	**0.69**	**0.70**	**0.71**	**0.71**
Total Avg		**0.79**	**0.65**	**0.58**	**0.79**	**0.72**	**0.74**	**0.74**	**0.60**	**0.75**	**0.74**	**0.67**	**0.74**	**0.74**	**0.67**	**0.77**	**0.72**	**0.72**	**0.74**	**0.74**

Text & Icons Labels

Avg		**0.91**	**0.83**	**0.81**	**0.90**	**0.88**	**0.89**	**0.89**	**0.82**	**0.88**	**0.87**	**0.84**	**0.86**	**0.86**	**0.83**	**0.87**	**0.85**	**0.85**	**0.86**	**0.86**

ASSOCIATION METRIC

Category	Layer	Association factor (avg)																		
2	L_ LMF_Bus_Stops_P_new	1.00	0.44	0.47	0.49	0.55	1.00	1.00	0.49	0.49	0.52	0.74	1.00	1.00	0.24	0.25	0.25	0.27	1.00	1.00
	L_ LMF_CH_Stations_P_new	1.00	0.40	0.41	0.41	0.44	1.00	1.00	0.41	0.47	0.41	0.68	1.00	1.00	0.21	0.21	0.23	0.24	1.00	1.00
Sub Avg		**1.00**	**0.42**	**0.44**	**0.45**	**0.50**	**1.00**	**1.00**	**0.45**	**0.48**	**0.47**	**0.71**	**1.00**	**1.00**	**0.23**	**0.23**	**0.24**	**0.26**	**1.00**	**1.00**
3	L_ LMF_Car_Parks_P_new	0.91	0.35	0.36	0.38	0.62	0.91	0.91	0.39	0.41	0.33	0.37	0.91	0.91	0.41	0.36	0.36	0.37	0.91	0.91
	L_ LMF_Post_Offices_P_new	0.91	0.40	0.43	0.28	0.59	0.91	0.91	0.40	0.30	0.38	0.36	0.92	0.92	0.46	0.50	0.46	0.56	0.92	0.92
	L_ LMF_Taxi_ranks_P_new	0.93	0.26	0.27	0.35	0.54	0.92	0.92	0.35	0.36	0.36	0.34	0.91	0.91	0.34	0.33	0.35	0.39	0.91	0.91
	L_ LMF_Toilets_P_new	0.91	0.26	0.34	0.39	0.46	0.91	0.91	0.35	0.42	0.44	0.45	0.92	0.92	0.38	0.33	0.34	0.39	0.91	0.91
Sub Avg		**0.92**	**0.32**	**0.35**	**0.35**	**0.55**	**0.91**	**0.91**	**0.37**	**0.37**	**0.38**	**0.38**	**0.92**	**0.92**	**0.40**	**0.38**	**0.38**	**0.43**	**0.91**	**0.91**
Total Avg		**0.94**	**0.35**	**0.38**	**0.38**	**0.53**	**0.94**	**0.94**	**0.40**	**0.41**	**0.41**	**0.49**	**0.94**	**0.94**	**0.34**	**0.33**	**0.33**	**0.37**	**0.94**	**0.94**

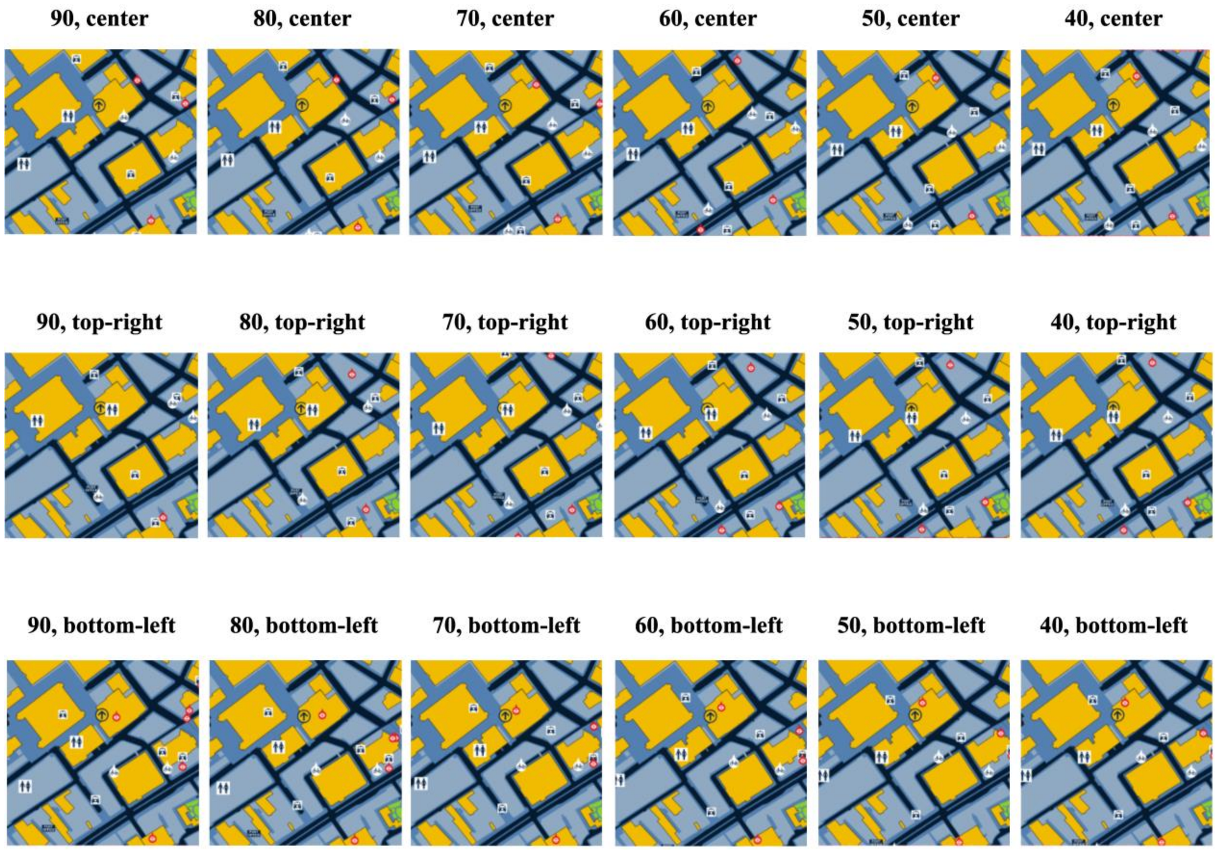
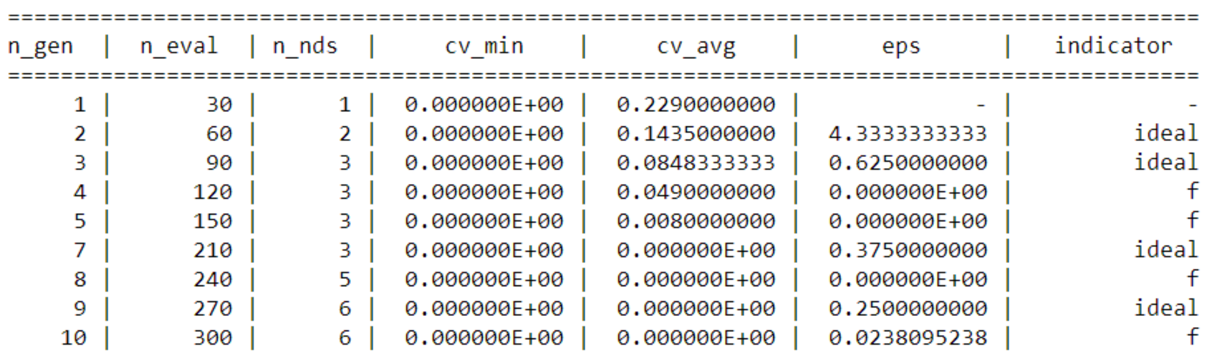


Figure 31: Maps of icons placed by the grid algorithm across the different experiments

Appendix 3. Printouts during NSGA2 execution

In this appendix is presented the table as generated from Python after running the multi-objective optimization process for ten generations. It works as a detailed snapshot of the optimization process at each generation, tracking key metrics such as constraint violations, diversity in the Pareto front, and the performance of the algorithm in terms of objective values.

Table 10: Results of the multi-objective optimization indicating the number of generation (n_gen), number of evaluations(n_eval), number of non-dominant solutions (n_nds), minimum, maximum and average constraint violation (cv_min, cv_max, cv_avg respectively), epsilon-dominance (eps) and performance indicator (indicator).



```
==========================================================================================
n_gen  |  n_eval  | n_nds  |     cv_min    |     cv_avg    |      eps      |   indicator
==========================================================================================
     1 |       30 |      1 |  0.000000E+00 |  0.2290000000 |             - |             -
     2 |       60 |      2 |  0.000000E+00 |  0.1435000000 |  4.3333333333 |         ideal
     3 |       90 |      3 |  0.000000E+00 |  0.0848333333 |  0.6250000000 |         ideal
     4 |      120 |      3 |  0.000000E+00 |  0.0490000000 |  0.000000E+00 |             f
     5 |      150 |      3 |  0.000000E+00 |  0.0080000000 |  0.000000E+00 |             f
     7 |      210 |      3 |  0.000000E+00 |  0.000000E+00 |  0.3750000000 |         ideal
     8 |      240 |      5 |  0.000000E+00 |  0.000000E+00 |  0.000000E+00 |             f
     9 |      270 |      6 |  0.000000E+00 |  0.000000E+00 |  0.2500000000 |         ideal
    10 |      300 |      6 |  0.000000E+00 |  0.000000E+00 |  0.0238095238 |             f
```