Nexus Network Journal
https://doi.org/10.1007/s00004-024-00807-7

Nexus Network Journal
Architecture and Mathematics

RESEARCH

Non-trivial Self-organised Floor Plans: An Optimisation Strategy

Silvio Carta[1] · **Krishnendra Shekhawat**[2] · **Luciano Ambrosini**[3] · **Tommaso Turchi**[4] · **Miguel Vidal**[5]

Accepted: 27 November 2024
© The Author(s) 2025

Abstract

We present a novel workflow where non-rectangular floor plans (NRFPs), namely plans with at least one concave corner, are self-generated using a model that directly encodes key optimisation factors on spatial quality and energy consumption, with non-rectangular building envelopes. The modelling considers a number of key factors including architectural and urban quality, net zero factors and adherence to general residents' feedback from previous studies. We provide evidence that the proposed workflow outperforms a number of optimisation solvers generally used in computational design, in those cases where solar radiation is most needed. Our study combines a syntactic approach with a computational one with a novel workflow to encode tangible and intangible factors to improve a specific class of non-trivial floor plans (L-shaped).

Keywords L-shaped floor plans · Non-trivial floor plans · Optimisation · Factor encoding · Self-organising floor plans

Introduction

There seems to be a growing interest in automated methods to generate spatial configurations, specifically floor plans. Although self-organisation is not a new concept in design (Carta 2021), recent Machine Learning (ML) and, generally

✉ Silvio Carta
Silvio.carta@greenwich.ac.uk

1 School of Design, University of Greenwich, London, UK

2 Department of Mathematics, Birla Institute of Technology and Science, Pilani, India

3 Independent Researcher, Naples, Italy

4 Department of Computer Science, University of Pisa, Pisa, Italy

5 Loughborough University, Loughborough, UK

Published online: 09 January 2025

speaking, Artificial Intelligence (AI) methods seem to have augmented the number of studies and prototypes in self-organising plans.

In this work we look in particular at concave shapes, L-based, U-based or T-based, namely where there is a pronounced depth of the building footprint and the plan is not based on a square or rectangle. These are called non-trivial spatial configurations and are generally not considered in self-generative optimisation methods in floor plans (Raveena and Shekhawat 2023). In this investigation we wanted to explore the extent to which non-trivial plans can perform better than squared-based ones in some sustainable parameter.

In this study we combined the existing model developed by Carta et al. (2023), implemented through the L-shaped plan algorithm developed by Shekhawat (2023). We apply our workflow to a generic case study, based on an initial generation with parameters for a housing development in London, UK. We compare existing housing schemes with an optimised version obtained with our method, measuring the improvements in terms of spatial quality and energy consumption in two very different urban locations.

Existing Work

Self-generation of architectural configurations has been studied for a long time. More generally, automation in spatial configurations can be traced back to the early 1970s (e.g. with the work of March and Steadman 1971). A more detailed account of the history and milestones in automation in architecture can be found in Carta (2021). This study involves Multi-objective optimisation (MOO) as a technique to obtain floor plans at different stages of the development. Particularly relevant is the work of Gunantara (2018), Deb et al. (2016), Caramia et al. (2020), Giagkiozis and Fleming (2015), Pugnale and Sassone (2014) to name but a few. The work of Jabi on graphs (e.g. in Alymani and Jabi 2024 and Alymani et al. 2023) is also particularly important to contextualise the translation of spatial configurations to mathematical graphs.

The self-generation of rectangular floor plans has been studied for decades with some very relevant results, especially with Neuro Evolution of Augmenting Topologies (NEAT), Generative Adversarial Networks (GAN) (Carta 2021), mass customisation (Abdulmajeed et al. 2023), and more specifically with graph convolutional networks (GCNs) (Li et al. 2022; Park et al. 2023). However, most results are limited to rectangular-contained boundaries and do not account for concave shapes (L-U-T shapes and the like). Non-trivial spatial configurations are generally not considered in self-generative optimisation methods in floor plans (Raveena and Shekhawat 2023). There is substantial evidence to support that non-rectangular building envelopes allow for more effective sustainable strategies in housing design (Jankovic and Carta 2021). With this study, we propose a novel workflow for designers to work with self-generating floor plans using a model that directly encodes key optimisation factors on spatial quality and energy consumption, with non-rectangular building envelopes.

Methods

Overall, the approach developed for this study consists of three parts. First, we generate generic floor plans that are optimised using social and physical data. Second, we transform the floor plans into graphs to be analysed and modified, and finally we change the plans into L-shaped optimised configurations. We then analyse the performance of the resulting floor plans and compare them with non-L shaped plans (Sections "Validation" and "Results") to draw conclusions.

The first phase of the workflow is the modelling and generation of plans. More specifically, we follow the steps below:

Step 1: Generation of optimised generic floor plans

We optimised a generic floor plan using the method developed in Carta (2023), where we encoded social values of local resident communities in London. In this method, combined three sets of data: residents' social aspirations, housing regulations and policy-makers agendas; and successful elements for resilient communities. We generated a floor plan using a generic range of dimensions of London flats, optimised using a linear model of factors and weights (1) and optimised using a multi-objective optimisation strategy based on the evolutionary engine Wallacei (Showkatbakhsh and Makki, 2020) (see Fig. 1).

$$Q = x_1 a + x_2 b + \cdots + x_6 f \quad (1)$$

In (1) Q is the Quality of Housing Development: an overall value of quality of the project determined by quality factors (x_1, x_2…) multiplied by the coefficients a, b, c… that are used to assign weight to each factor (Carta 2023).

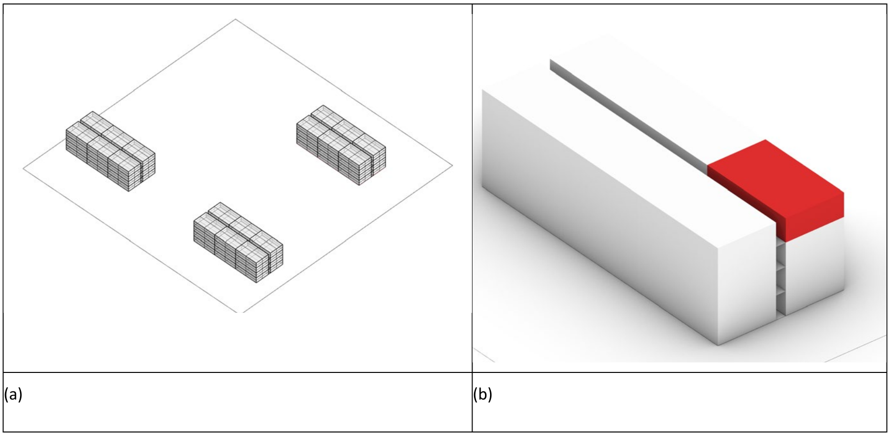

Fig. 1 Initial generation of blocks of housing (**a**) and the isolation of a single housing unit for optimisation (**b**)

The floorplans were generated using an optimisation process developed in the 3d modelling tool Rhinoceros (v7) with its algorithmic modelling plugin Grasshopper (Build 1.0.00007). The optimisation was then performed using the Grasshopper plugin Wallacei (version 2.7).

Assumptions and Variables

The 3D model was generated based on a rectangular, detached block of flats type, with units on both sides of a corridor. The following assumptions were made:

– All units are of a three-bedroom size, measuring 15 m in width by 9 m in depth, with an approximate gross floor area of 135 square metres.
– The corridor is 1.5 m. wide, resulting in a total depth of the block of 16 m.
– The plot density is set at 150 units per hectare, aligning with the lower range specified in the London Draft Paper on Density Policy for urban sites within a ten-minute walking distance of a town centre (London Gov, 2017).
– The unit height, relevant only for 3d visualisation purposes, is 3.5 m.

The Grasshopper computational model uses the variables listed in the table below in the optimisation process. Among these, some of the variables are maximised, while others are minimised. We assigned weighting value to each variable based on a statistical model developed by Carta et al. (2023).

3d Model: Floorplans

The computational model calculates the number of units per floor based on:

– Population, inherited from the previous set of variables.
– Number of buildings and number of floors per building, which are two parameters that the optimisation algorithm will navigate.

by dividing the population and the number of buildings.

It then builds the housing blocks by appending extruded rectangles in randomised locations, ensuring they remain within the plot area. The building centres are the centroid points of a Voronoi diagram, with the maximum radius being the distance between buildings. The "Populate 2D" tool in Grasshopper uses an offset area of the plot, equal to one-third of the block length, as points for the Voronoi algorithm.

The Grasshopper model generates a series of boxes for the 3D visualisation and rectangles for the 2D export according to the units per floor, unit length, floor-to-floor height and number of floors. The geometry data is organised in Grasshopper's data tree structure as follows:

{A;B;C} (i)

{Building; Column; Unit/Corridor/Unit} (Floors)

Here, the parameters in brackets are the data tree indices, and those in parentheses are the list indices for each floor.

Optimisation Process

The optimisation process uses the multi-objective evolutionary algorithm Wallacei (Showkatbakhsh and Makki 2020), developed by Mohammed Makki, Milad Showkatbakhsh and Yutao Song. This algorithm employs the NSGA-2 algorithm (Deb et al. 2016) as the primary evolutionary algorithm and uses the K-means method as the clustering algorithm.

The algorithm's gene pool includes the following variables:

- Area per resident.
- Private outdoor space.
- Density.
- Plot area.
- Cost per sqm.
- Number of surrounding links.
- Number of buildings.
- Building floors.

The objectives for the algorithm are to either maximise or minimise the values in Table 1. The overall quality number for each solution is obtained by summing all maximised (converted to negative) and minimised values, which the algorithm aims to minimise as it searches for the optimal solution. The computational model sets the optimisation value to zero in three instances: if the buildings intersect, if the units per floor are less than 2 or if the units per floor are more than 8.

The resulting value of the parameters to maximised and minimised is what we have calculated as Quality (Q) of the model. Different quality values have been tested combining maximised and minimised parameter normalised and weighted as shown in Carta et al. (2023). The quality (Q) value used in this study is as follows (see Table 2).

Step 2: from floor plans to graphs

Once the floor plans have been generated in Step 1, we generate a Biconnected PTPG (Properly Triangulated Plan Graph) using the original floor plans as a starting point. This part consists in the following steps:

workflow:

1. Import floor plans of building blocks in Grasshopper (GH), with each flat as a separate geometry.
2. With PLAN GRAPH (Deprez 2022), generate nodes and edges of the graphs, where the nodes are the flats, and the edges are the connections between the central core and each flat.

Table 1 Computational model variables

Maximised variables			Minimised variables		
Variable	Normalised domain	Weighting value	Variable	Normalised domain	Weighting value
Area per resident	39–61 sqm	1.2	Cost (adding up units area plus plot area. Cost per sqm of built area = £ 6427. Cost per sqm of land = £ 1000)	25,000–45,000 m	2.37
Private outdoor space per resident	0–10 sqm	1.2	Average Distance to Urban links	0 – 500 m	0.41
Plot area	5000–25,000 sqm				
Shared outdoor space per resident	(Difference between the plot area and the building footprint)				

Table 2 Quality values used in the study

Quality (Q)			
Maximised Values		Minimised Values	
Area per resident	Weighted	Cost per sqm	Weighted
Private Outdoor Space	Weighted	Distance to Urban Links	Weighted
Density	Normalised		
Shared Open Space per Resident	Normalised		

3. Generate the planar graph and visualise it on GH, extracting some initial information, specifically the edge list.
4. The edge list is streamed as TXT from GH.
5. Run a python script to format the TXT data representing the edges in the graph into a CSV.
6. Rebuild the graph in a second python script using the CSV file and generate the dual graph.

Export Floor Plans as a Graph

The floor plan has been translated into a graph structure using PLAN GRAPH, a Grasshopper plugin developed by Louise Deprez in 2022 (Deprez 2022) (Fig. 2). Each room is identified as a node and the links among rooms are identified with edges (see Fig. 3). As the plugin offers very limited analytical capabilities (only density, shortest path length, and matrices information), we developed a way to export the graph structure into a format that can be used outside of the Grasshopper environment. To do so, we used the adjacency matrix as exported by PlanGraph.

More specifically, we exported the Edge Information output matrix (script: planar_graph_exporter.3 dm directly in GH available on github) and generated a planar graph in python (script: from_planar_to_dual_graph.py on github).

Testing the Conversion from Planar to Dual Graph

The translation from a planar graph to a dual graph is not trivial and there are not ready-made solutions for it. We created our own workflow which is explained and tested below.

Using a generic floor plan configuration, we can see that the data represents a mapping from vertices in the dual graph to the faces in the original graph. Each vertex in the dual graph is associated with a set of faces (see Fig. 4).

Condition for dual graph: Each Vertex Corresponds to a Face:

- Vertex 2 corresponds to faces {0, 1, 3}.
- Vertex 4 corresponds to no faces (an empty set).
- Vertex 1 corresponds to faces {0, 2}.

```
ID,Label,Source,Target
0,h-8,13,7
1,h-7,15,6
2,h-9,13,8
3,h-10,14,9
4,h-11,14,10
5,h-12,16,11
6,h-1,15,0
7,h-2,13,1
8,h-3,13,2
9,h-4,14,3
10,h-5,14,4
11,h-6,16,5
12,h-entr1,12,13
13,h-entr2,12,14
14,h-entr3,12,15
15,h-entr4,12,16
```

(a)

ID	Label	Source	Target
0	h-8	13	7
1	h-7	15	6
2	h-9	13	8
3	h-10	14	9
4	h-11	14	10
5	h-12	16	11
6	h-1	15	0
7	h-2	13	1
8	h-3	13	2
9	h-4	14	3
10	h-5	14	4
11	h-6	16	5
12	h-entr1	12	13
13	h-entr2	12	14
14	h-entr3	12	15
15	h-entr4	12	16

(b)

export_graph_data.py

C: > Users > silvi > Desktop > nexus > export_graph_data.py > ...

```
#path: "C:/Users/silvi/Desktop/nexus/stream.txt"
#output: "C:/Users/silvi/Desktop/nexus/output.csv"

# Specify the filename for the input data file
input_filename = "C:/Users/silvi/Desktop/nexus/stream.txt"

# Specify the filename for the CSV file
output_filename = "C:/Users/silvi/Desktop/nexus/output.csv"

# Read data from the input file and write to CSV
with open(input_filename, mode="r") as txt_file, open(outpu
    # Skip the header line
    next(txt_file)

    # Write the header to the CSV file
    header = "ID,Label,Source,Target\n"
    csv_file.write(header)

    # Loop through the txt data and write each row to the C
    for line in txt_file:
        csv_file.write(line)

print(f'Data exported to {output_filename}')
```

(c)

Fig. 2 Representation of a generic floor plan as graph as TXT (**a**), and its conversion to CSV format (**b**), with related script **c**)

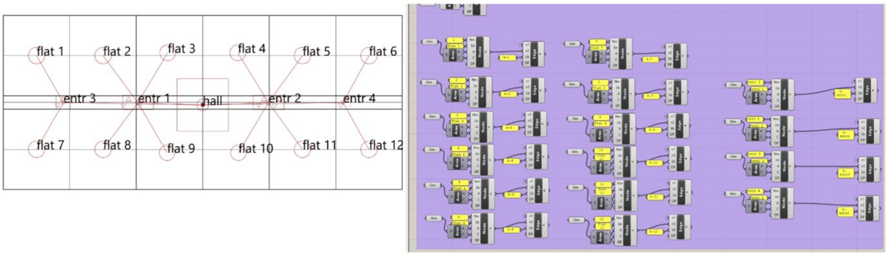

Fig. 3 Conversion of the optimised plan from Step 1 to a graph structure. Full GH script available in the project github page https://github.com/seelca/non-trivial-floorplans/

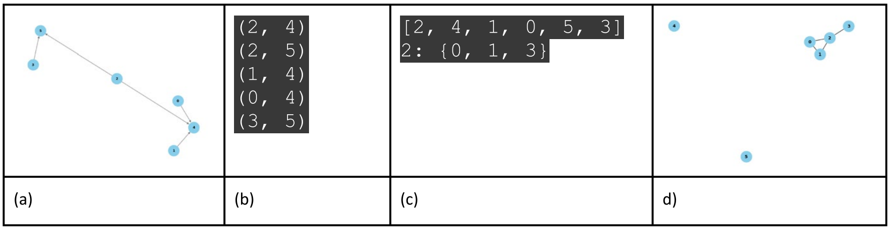

Fig. 4 Planar graph of the sample floor plan (a), its edges (b), the representation of the corresponding dual graph (c) and (d)

- Vertex 0 corresponds to faces {1, 2}.
- Vertex 5 corresponds to no faces (an empty set).
- Vertex 3 corresponds to face {2}.

This condition is satisfied.

Condition: Edges Correspond to Shared Edges in the Original Graph:

- There is an edge between vertices 2 and 0 because faces {0, 1, 3} and {1, 2} share an edge in the original graph.
- There is an edge between vertices 1 and 0 because faces {0, 2} and {1, 2} share an edge in the original graph.
- There is an edge between vertices 2 and 3 because faces {0, 1, 3} and {2} share an edge in the original graph.
- No other edges are present.

This condition is also satisfied.

As the workflow to convert a planar to a dual graph works, we integrated the process into the translation of the planar graph into the L-shaped plan (Raveena and Shekhawat 2023).

Step 3: generate L-shaped optimised floor plans

This step uses the Biconnected PTPG to generate a L-shaped optimised plan, using the method developed in (Raveena and Shekhawat 2023).

As the name suggests, a L-shaped floor plan corresponds to a floor plan with L-shaped boundary, i.e., a floor plan with one concave corner as shown in Fig. 5 (a rectangular floor plan has no concave corners).

L-shaped floor plans can easily be generated from rectangular floor plans by shrinking walls of some rooms. For example, consider the floor plan in Fig. 6b. It has L-shaped boundary and can easily be reduced into a rectangular boundary floor plan. At the same time, the floor plan in Fig. 6a cannot be directly transformed into a rectangular floor plan, hence, we call such floor plans as non-trivial L-shaped floor plans (Raveena et al. 2024). In this paper, we are generating non-trivial L-shaped floor plans only, therefore, L-shaped floor plans stand for non-trivial L-shaped floor plans.

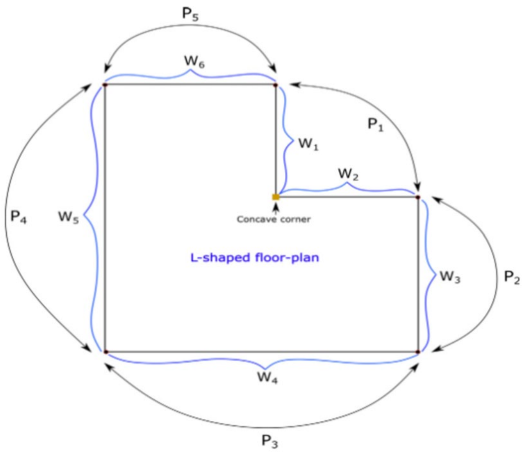

Fig. 5 L-shaped floor plan boundary

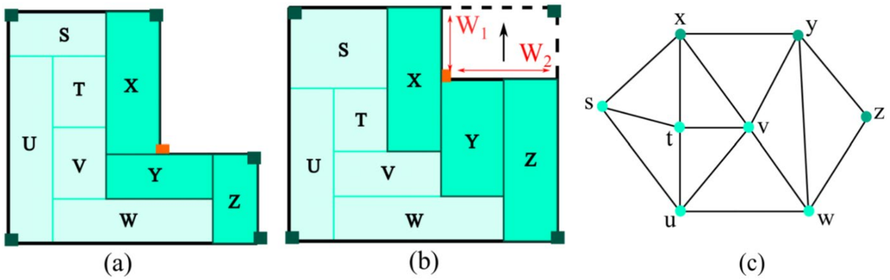

Fig. 6 L-shaped floor plans and corresponding adjacency graph

Mathematically, the problem is to automatically generate a L-shaped floor plan corresponding to a given adjacency graph. For example, the floor plans in Fig. 6a, b correspond to the graph in Fig. 6c. Here, to ensure floor plans with rectangular rooms only, the input adjacency graph is always a biconnected PTPG (properly triangulated plane graph, Bhasker and Sahni 1988).

The steps for generating a L-shaped floor plan for a given adjacency graph are as follows:

Step 1: Selecting a triplet of vertices

In the given graph *G*, a set of 3 vertices say (*a*, *b*, *c*) is called triplet if vertex *a* is adjacent to vertex *b* and vertex *b* is adjacent to vertex *c*. Here, the idea is to place the rooms corresponding to the triplet at the concave corner. The condition for the existence of a L-shaped floor plan for a triplet can be found in Raveena et al. 2023. For example, for the graph in Figure 7a, we consider the vertices (*x*, *y*, *z*) as triplet.

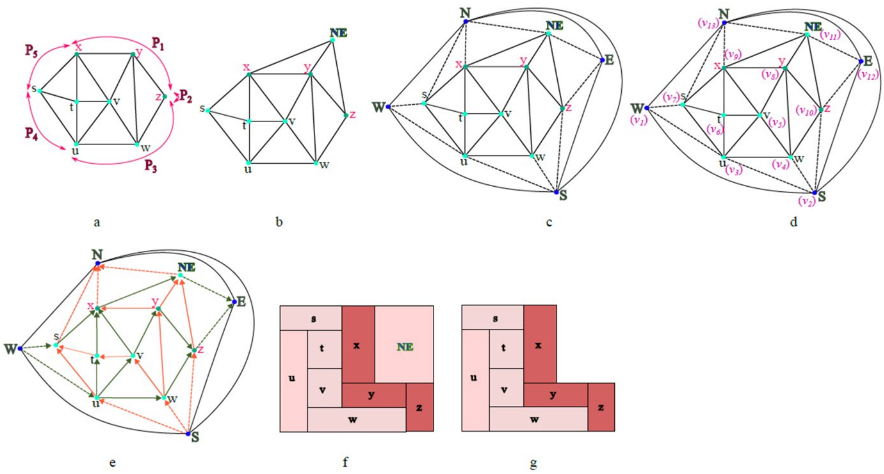

Fig. 7 Generation of a L-shaped floor plan corresponding to a given adjacency graph

Step 2: Selection of paths

L-shaped boundary is made up of 6 walls as shown in Fig. 5 and therefore, next is to identify the set of rooms lying on these walls. For each wall, the identified vertices are forming a path. To ensure non-triviality, triplet always lies at the concave corner, i.e., it belongs to path P_1. For example, in Fig. 6, five paths have been marked. Also, for the graph in Fig. 7a, paths have been marked.

Step 3: Adding a new vertex to the given graph

Here, the idea is to generate a rectangular floor plan for the new graph *G*' obtained by adding a new vertex say NE to the given graph *G*. Then, to delete the room NE from the rectangular floor plan for *G*' to have a required L-shaped floor plan for *G*. Here, the vertex NE must be adjacent to all the vertices of path P_1. For example, see Fig. 7b, where NE has been added.

Step 4: Finding a regular edge labelling (REL) graph for the modified graph *G*'

In order to determine the rectangular floor plan, it is important to find how adjacent rooms are placed in the layout, i.e., whether they are horizontally adjacent or vertically adjacent. This is depicted by the REL graph shown in Fig. 7e, where green edges show that the rooms are horizontally adjacent and the red edges show vertical adjacency.

To obtain a REL, firstly, the four completion process as detailed in He (1993) is applied to the modified graph and then canonical ordering algorithm presented in Kant and He 1997 is used to obtain the REL graph.

The four completion process establishes the corner points and the positioning of boundary rooms for the rectangular floor plan. To position the room NE at the northeast corner, it must serve as the end vertex of path P'_1 and the initial vertex of P'_2. Consequently, the path P_5 and room NE collectively form path P'_1, with the vertices of path P_2 and node NE contributing to path P'_2. The selection of paths P'_3 and P'_4 mirrors that of paths P_3 and P_4, respectively.

Next is to add the nodes N, E, W, and S to the modified graph *G*' and make them adjacent to the vertices of paths P'_1, P'_2, P'_3, and P'_4 respectively. To conclude the four completion process, edges NE, ES, SW, and NW are added into the graph subsequent to the addition of nodes N, E, W, and S (see Fig. 6c).

For obtaining a REL graph using the canonical ordering algorithm, a four-connected graph is required as input. The graph becomes four-connected after adding the edge NS to the graph obtained in the four-completion process, as shown in Fig. 8c. Upon applying the proposed algorithm, we obtain the canonical ordered graph for G' in Fig. 7d and the corresponding REL, as depicted in Fig. 7e.

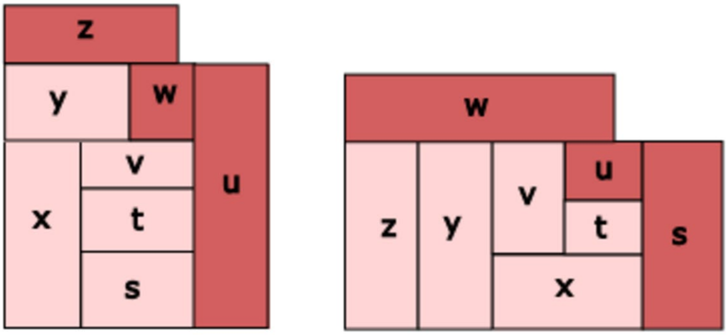

Fig. 8 L-shaped floor plans for the graph in Fig. 6a

Step 5: Obtaining a rectangular floor plan for the modified graph

For the graph *G*', using the algorithm given in Bhasker and Sahni (1987), a rectangular floor plan is obtained as shown in Fig. 7f.

Step 6: Obtaining an L-shaped floor plan for the graph

From the obtained rectangular floor plan, deleting the room NE leads us to the required L-shaped floor plan as shown in Fig. 7g.

Remark. Several L-shaped floor plans can be obtained by changing the triplet or the set of paths. For the graph in Fig. 7a, two L-shaped floor plans have been shown in Fig. 8. If the obtained L-shaped floor plan is trivial, using the flipping algorithm given in Raveena et al. 2023, it can be transformed into a non-trivial L-shaped floor plan.

Significance

The ability to generate non-trivial plans with embedded optimisation factors represents a significant advancement in computational design for architecture. This approach bridges the gap between energy performance optimisation and the creation of functionally diverse building layouts. By incorporating both geometric and performance-based metrics in the generation of L-shaped floor plans, our method offers several key contributions to the field. First, Integration of Energy Performance and Spatial Configuration: our approach demonstrates that it's possible to optimise for both energy efficiency and spatial layout simultaneously. This integration is crucial for creating buildings that are not only sustainable but also functionally effective.

Second, advancement in sustainable architecture: the method's ability to maximise solar radiation exposure in L-shaped configurations provides architects and urban planners with a powerful tool for designing energy-efficient buildings.

This could lead to reduced energy consumption and improved passive solar strategies in architectural design.

Third, computational Design Innovation: By combining traditional optimisation solvers with a novel L-shaped plan generation algorithm, we have expanded the possibilities of computational design in architecture. This approach allows for the exploration of non-trivial floor plan configurations that might be overlooked in conventional design processes.

Fourth, potential for urban planning: the ability to generate and evaluate multiple non-trivial floor plan configurations quickly could have significant implications for urban planning. It could enable more efficient space utilisation and energy-optimised building arrangements at a city scale.

Finally, adaptability to different contexts: while our study focused on maximising solar radiation, the method's flexibility allows for the incorporation of various other performance metrics. This adaptability makes it a valuable tool for designing in different climatic and urban contexts.

Validation

Summarising, we started the workflow with the generation of a floor plan optimised using Wallacei (Showkatbakhsh and Makki 2020) using the parameters elaborated in Carta (2023). This initial spatial configuration is converted into a planar graph, then into a dual graph and, finally, into a L-shaped plan using the method elaborated by Shekhawat (2023). This produced the first output of an optimised non-trivial floor plan. The same initial plan has then been optimised using several standard solvers (Galapagos, Octopus, Opossum, Tunny and Wallacei). The solvers have been selected among the most used in the Grasshopper community. Galapagos is used as a benchmark for the other solvers, as it represents a generic solver built-in Grasshopper (Rutten 2013). This gave as outputs other optimised floor plans which we compared to the non-trivial result. The aim of this comparison is to understand the extent to which our model with the non-trivial floor plans yields accurate results (both in general and in terms of individual aspects) (see Fig. 9).

To compare the non-trivial plan that we generated to the optimised plans obtained with the other solvers, we developed a number of metrics to be used as factors for the comparison.

We therefore developed 7 configurations (C1–C7) corresponding to each solver and our L-shaped approach (Fig. 10 and Table 3).

The metrics are elaborated upon the work of Kistelegdi et al. (2022).[1] They are particularly useful for observing how building performance changes as certain geometrical aspects of the building vary. These are called building geometry as a design variable (BGDV) and are described as follows.

[1] In particular, the reader can see the overview of the metrics in Kistelegdi et al. (2022) Table 3 on page 29.

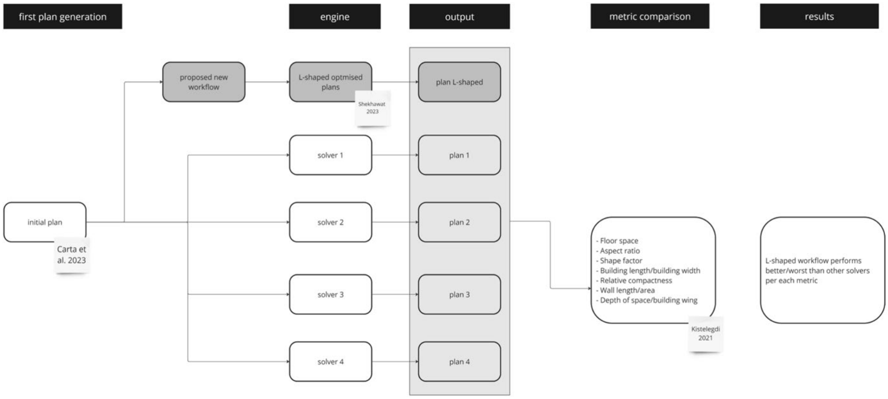

Fig. 9 The diagram illustrates the entire workflow from our initial optimised plan to the comparison using our metrics

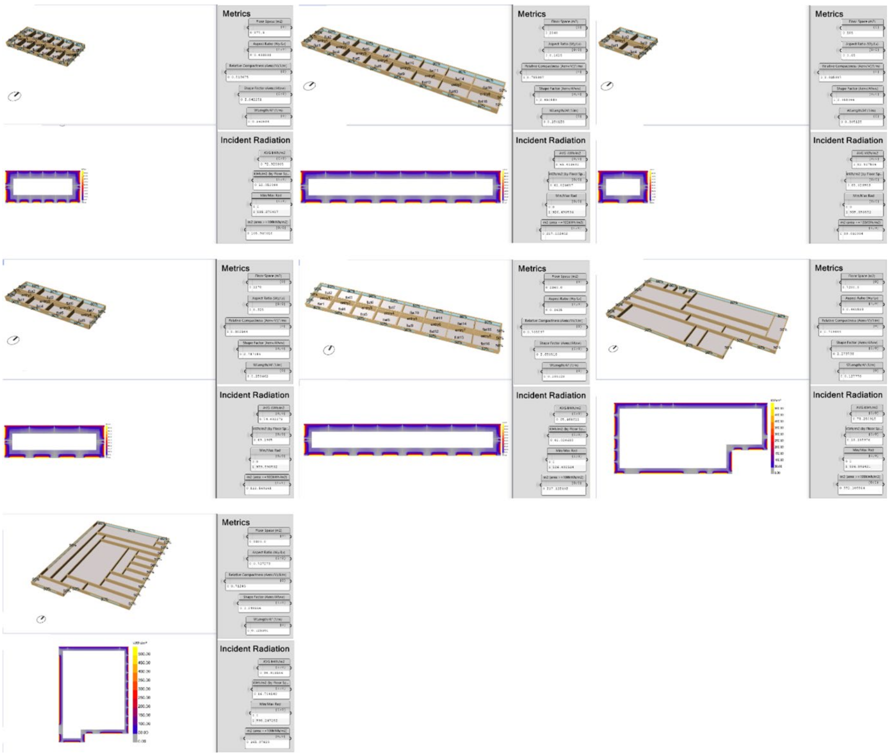

Fig. 10 Mosaic of all configurations developed with different solvers (C1–C5)

Table 3 Summary of all results from the metrics calculation with the different optimised solutions

			Metrics data					Incident radiation data			
Config Type	Solver Type	Fitness value	Afs (m^2)	AR	RC (m-1)	SF (HLFF)	WL/Afs (m-1)	AVG Rad (kWh/m^2)	AVG Rad norm. over Afs (kWh/m^2)	Max Rad (kWh/m^2)	Area > = 100 kWh/m^2 (m^2)
C1 (Wallacei)	*MOGA: NSGA-2*	11.315	877.50	0.433	0.814	2.842	0.361	72.93	52.05	935.27	105.94
C2 (Octopus)	*MOEA: SPEA-2 HypE*	11.352	2,340.00	0.162	0.786	2.650	0.280	65.41	41.03	926.50	217.12
C3 (Galapagos)	*GA*	11.078	585.00	0.650	0.836	2.852	0.305	82.92	63.02	939.90	89.03
C4 (Opossum)	*MOGA: CMAES*	11.325	1,170.00	0.325	0.803	2.717	0.288	74.48	49.20	939.90	133.56
C5 (Tunny)	*MOEA: BO1-TPE*	11.318	2,340.00	0.162	0.786	2.650	0.280	65.46	41.02	926.50	217.12
C6 (L-shaped)	*BspkA2 (UK1)*	–	7,200.00	0.461	0.719	2.274	0.128	78.25	18.16	996.80	332.11
C7 (L-shaped)	*BspkA2 (UK2)*	–	8,300.00	0.727	0.712	2.249	0.125	84.90	16.70	995.25	365.98
BO1	*Bayesian Optimisation*			**Afs**	*floor space area*		**SF**	*shape factor (heat Loss Form factor)*			
BspkA2	*bespoke algorithm for L-shaped plan configuration*			**AR**	*aspect ratio*		**WL/Afs**	*wall length out floor space area*			
				RC	*relative compactness*						

Floor Space (Afs)

While most of the studies referenced in Kistelegdi et al. (2022) refer to Gross Floor Area (GFA), which is a generic term used globally whose definition may vary from country to country (e.g. with the inclusion of internal walls, partitions etc.), we preferred to use the definition of Total Useful Floor Area (TUFA) as specified in the Approved Document L of the UK Building Regulations. The Floor Space is then defined in this study as: "the total area of all enclosed spaces, measured to the internal face of the external walls. […] This includes "the area of sloping surfaces such as staircases, galleries, raked auditoria and tiered terraces should be taken as their area on plan […] and excludes "areas that are not enclosed, such as open floors, covered ways and balconies" (UK Building Regulations 2010: 77).

Aspect Ratio (AR)

This is defined as the width of the building divided by its depth (Tuhus-Dubrow and Krarti 2010).

Shape Factor (SF or HLFF)

This is calculated as the area of the building envelope divided by the Area footprint without walls (AfWW). AfWW is the total area (Afs) of the floor plan minus the footprints of all walls. This is also the Heat Loss Form Factor.

To express compactness, Aksoy and Inalli (2006) use the Shape Factor (SF), which is equal to the proportion of a building's length to its width. Bostancioğlu (2010) uses the ratio of external wall area to floor area (EWA/FA). Both A/S ratio, used by Gonzalo and Habermann (2012), and A/V ratio, applied by Hegger and Horstmann (2008), determine the proportion of the area of a building envelope to its volume.

Relative Compactness (RC)

This is the measure of the area of the building envelope divided by its volume. AlAnzi et al. (2009) defined it as the "envelope surface area exposed to the outdoor air (i.e., exposed exterior wall area)" divided by the area of a reference plan, like a squared plan (AlAnzi et al. (2009: 824). However, as their definition is too restrictive to their specific study, we opted for a more generic measurement of relative compactness which relates the compactness to the building volume (instead of a subjective reference plan).

Wall Length/Area (WL/Afs)

This is the ratio between the total wall length (the total length of all internal and external walls taken as one total value) and the overall area (in our study as indicated above Afs). This definition is partially based on Granadeiro et al. (2013).

Incident Solar Radiation

To the five metrics considering the geometry of the plans from the Building Geometry as a Design Variable (BGDV), we added four metrics of Incident Radiation Data to monitor each plan's variation of performance in terms of solar energy intake. The four measures are:

- Average Radiation. AVG Rad (kWh/m^2);
- Average Radiation normalised over the Area of the Floor Space (Afs). AVG Rad norm. over Afs (kWh/m^2);
- The maximum radiation value. Max Rad (kWh/m^2);
- The total area of solar radiation within the floor plan (as also shown in Fig. 11). Area $> =$ 100 kWh/m^2.

To ensure the reproducibility of our results and provide a clear understanding of how the plans in configurations C1–C7 were produced, we present a detailed explanation of our validation process:

Initial Parameters

For all configurations (C1–C7), we used the following initial parameters:

- Plot area: 10,000 m^2
- Maximum building height: 30 m
- Minimum room size: 9 m^2
- Maximum room size: 50 m^2

Optimisation Process

C1–C5 (Solver-based configurations)

In addition to the parameters outlined in Section "Methods", we incorporated the solar radiation analysis. More specifically it was defined a fitness function as linear combination of these parameters:

Area per resident (ApR) – *maximize*; Private outdoor space (POS) – *maximize*; Density (DENS) – *maximize*; Shared Ourtdoor Space (ShOS) – *maximize*; Cost per sqm (Cost) – *minimize*; Distance to urban Links (DUL) – *minimize*.

Wallacei (C1): Version: Wallacei X (2.7), Generations size: 50, Generation count: 100.

Octopus (C2): Version: Octopus 0.4, Elitism: 0.500, Mutation Probability: 0.200, Mutation Rate: 0.90000, Crossover Rate: 0.800, Population size: 100.

Galapagos (C3): Version: Galapagos [Grasshopper 1.0.00007]: Max. Stagnant: 50, Population: 50, Initial boost: 2x.

Opossum (C4): Version: Opossum 3.0.0, Iterations exceed: 200, Iterations without improvements exceed: 20.

Tunny (C5): Version: Tunny 0.11.0, Number of startup trials: 10, Number of EI candidates: 24, Prior weight: 1.0.

C6–C7 (L-shaped configurations)

1. Initial graph generation:
   - o Convert the optimised plan from C1 (Wallacei) into a planar graph
   - o Identify nodes (rooms) and edges (connections between rooms)
2. L-shaped plan generation:
   - o Select a triplet for an L-shaped corner based on maximizing solar exposure
   - o Generate paths along the six walls of the L-shape
   - o Add a new vertex to create a rectangular floor plan
   - o Delete new vertex to obtain an L-shaped plan
3. Post-processing:
   - o Apply flipping algorithm if necessary to ensure non-triviality

Evaluation Metrics

We used the following formulas for calculating each metric:

1. Floor space (Afs): Sum of all room areas
2. Aspect ratio (AR): Width of building/Depth of building
3. Relative compactness (RC): (Total surface area)/(Volume^(2/3))
4. Shape factor (SF): (Total surface area)/(Floor space)
5. Wall length/area (WL/Afs): (Total wall length)/(Floor space)

Incident solar radiation data was calculated using Ladybug Tools (version 1.2.0) in Grasshopper, simulating annual radiation on the building envelope.

Comparison Process

We compared the different configurations by normalizing each metric to a 0–1 scale and calculating a weighted sum based on the importance of each metric. The weights were determined through a separate survey of architects and building performance experts.

Reproducibility

All simulations were run using:

- Rhinoceros 3D (version 7.0)
- Grasshopper (version 1.0.0007)
- Custom Python scripts for graph generation and L-shaped plan creation (available in the project's GitHub repository)

Functional Quality Discussion

While our study primarily focused on optimizing energy performance through geometric factors and solar radiation, it's crucial to address the functional quality of the generated plans. The current approach indirectly considers some functional aspects through the optimisation of spatial metrics like floor space and wall length ratio. However, we acknowledge that a more comprehensive evaluation of functional quality is necessary for real-world applications.

Limitations in the current approach regarding functional quality include:

- Lack of explicit consideration for circulation efficiency
- Absence of programmatic requirements in the optimisation process
- No evaluation of the quality of individual spaces beyond size constraints

To address these limitations and enhance the functional quality of generated plans, we propose the following areas for future work:

- Integration of Space Syntax metrics to evaluate circulation efficiency and spatial integration
- Incorporation of programmatic adjacency requirements in the optimisation process
- Development of qualitative metrics for individual room evaluation, considering factors like natural light, views, and privacy
- User studies to validate the perceived quality of generated plans by architects and end-users

By addressing these aspects in future iterations of our method, we aim to create a more holistic approach to floor plan generation that balances energy performance with high-quality functional spaces.

Results

We developed a Grasshopper definition where all metrics are parametrised and all floor plans from the solvers (C1–C5) and two of our L-shaped plans are evaluated.

The best configurations among C1–C7 seem to be those that maximise compactness and minimise the form factor, as these parameters can influence thermal performance and energy consumption for heating and cooling the building. Table 3 summarises the results of the calculations.

We specifically obtained the following results:

Floor Space (Afs)

With these metrics we did not focus particularly on minimising or maximising the value of the generated area. We note that L-shape plans are generally much larger (7200 m^2 C6 and 8300 m^2 C7) than C1–C5, with C3 (Galapagos) as the smallest. These results seem to suggest that, in order to maintain the initial parameters, the L-shape method tends to require more space to generate all rooms and connections needed.

Aspect Ratio (AR)

This is another measure of compactness, which is considered the longest extension of the building. Values next to 1 indicate that the length and depth of the floor plans are similar, while, by the same token, lower values indicate a larger distance between the two main extensions of the building.

Shape Factor (SF)

With this metric, we focused on the lowest values. A lower form factor indicates a more compact shape, which can result in lower heat losses through the building envelope. C6 and C7 (both L-shaped) have the lowest SF values, 0.719 and 0.712 respectively. With this metric, there does not seem to be a significant difference between C1-C5 and C6-C7. While C5 (Tunny) solved with the lowest SF (0,162), C6 (0,461) and C7 (0,727) seem to be in pair with the other solvers.

Relative Compactness (RC)

In this metrics we looked for highest values, as higher compactness implies less surface exposed to the external environment, potentially reducing heat losses and heating/cooling loads. C1 (Wallacei) has the highest RC value of 0.814. C3 (Galapagos) has the second-highest RC value of 0.836.

Wall Length/Area (WL/Afs)

This is an indication of the impact of the total length of the walls (internal and external) on the entire area. Higher values of this metric indicate that the floor plan has a high proportion of walls (expressed in length) over the area. Lower values suggest less wall length. C1–C5 scored higher in general, ranging from 0,280 (C2 and C5) to 0,361 (C1), while C6 (0,128) and C7 (0,125) performed significantly lower. This result seems to suggest that L-shaped configurations have a smaller presence of wall over the area. However, as the areas of C6–C7 are significantly larger than C1-C5, this would justify the lower values.

We note the differences between trivial (C1–C5) and non-trivial plans (C6–C7) concerning the geometric factors reported:

Trivial Configurations (C1-C5)

They have a lower surface area (Afs) compared to L-shaped configurations, ranging from 585 m^2 (C3) to 2340 m^2 (C2 and C5). They present a wide range of aspect ratios (AR), from 0.162 (C2 and C5) to 0.650 (C3). They generally have higher relative compactness (RC) values, ranging from 0.786 (C2 and C5) to 0.836 (C3). The form factor (SF) varies from 2.650 (C2 and C5) to 2.852 (C3), indicating less compact shapes compared to L-shaped configurations. These configurations (especially C1 and C3) present higher relative compactness values and, in some cases, more favourable aspect ratios.

L-Shaped Configurations (C6-C7)

Non-trivial floor plans have a much larger surface area (Afs), 7200 m^2 and 8300 m^2 respectively. They have a lower aspect ratio (AR), 0.461 and 0.727 respectively. They have slightly lower relative compactness (RC) values, 0.719 and 0.712 respectively. The form factor (SF) is significantly lower, 2.274 and 2.249 respectively, indicating more compact shapes. In summary, L-shaped configurations (C6–C7) seem to have some advantageous geometric characteristics compared to traditional configurations: (i) much larger surface area, potentially useful for large projects; (ii) a lower form factor, indicating a more compact shape that could reduce heat losses and finally; (iii) a slightly lower but relatively good relative compactness.

Incident Radiation

Looking at the average radiation values (AVG Rad) and area with radiation exceeding 100 kWh/m^2, we can identify potentially better configurations for harnessing passive solar gain. Higher average radiation (AVG Rad). C3 (Galapagos) has the highest value of 82.92 kWh/m^2. C1 (Wallacei) has the second-highest value of 72.93 kWh/m^2. Larger area with radiation exceeding 100 kWh/m^2. C7 (L-shaped) has the largest area of 365.98 m^2. C6 (L-shaped) has the second-largest area of 332.11 m^2.

Therefore, from an energy perspective, configurations C1 (Wallacei) and C3 (Galapagos) seem to offer a good compromise between compactness, form factor, and average solar radiation, potentially reducing heat losses and harnessing passive solar gains. Configurations C6 and C7 (L-shaped) could be advantageous if the main goal is to maximise the area with high levels of solar radiation.

Limitations and Next Steps

Future studies need to apply our method to real case studies to measure the extent to which L-shaped designs can enhance architectural plans by maximizing solar radiation exposure. It would be important to understand what the optimal conditions for the methods are to be applied with successful results.

One of the limitations of this study is that the initial seed for the generation of the starting spatial configurations (Step 1) also includes urban parameters which are then encoded in the following stages of our method. The encoding of urban parameters (Carta 2023) has some residual effects on the initial spatial distribution in the floor plan that we used to start the study. For example, the generation of an individual plan is related to the generation of its building block, which is, in turn related to the generation of the other blocks in the urban development considered. We have been using one of those plans as an example to conduct our study, as it encodes some of the information (e.g., area per resident) that we wanted to keep in the pipeline of the generative process. Repeating the experiment with different plans and overall spatial configuration at the urban level (instead of only the building level) may yield slightly different results, which are worth exploring.

Further studies should also consider the embedding of local regulations, minimum room sizes, structural elements and all those parts of a building that are not easily movable in general terms. This study, however, focused on the generation of spatial configurations and their optimisation. Future work should delve into more specific aspects of the generation, for example considering more specific aspects of buildings.

Conclusions

Our study suggests that the choice between trivial (rectangular) and non-trivial (L-shaped) configurations would depend on various factors, such as project size, compactness and energy efficiency goals, and architectural preferences.

To note that T or U -shaped configurations are characterised by at least two concave corners and therefore, the algorithm used for L-shaped floor plans cannot be directly used to handle two concave corners lying at different positions. A solution to compute T-shape floor plan has already been published (Raveena 2024).

In turn, L-shaped configurations might be more suitable for large projects requiring a larger surface area, while rectangular configurations might be preferable for smaller projects with specific compactness and aspect ratio requirements.

More generally, our study demonstrates that L-shaped floor plans, as generated using the method presented in this article, improve levels of incident solar radiation. This means that L-shaped plans can be used to optimise spatial configurations in world regions where solar gain needs to be maximised.

Our study demonstrates the potential of combining traditional optimisation techniques with novel geometric approaches to generate high-performing, non-trivial floor plans. The enhanced validation process reveals several key insights:

- Solver Specialization: While all tested solvers perform adequately, certain solvers excel in specific optimisation scenarios. Wallacei and Galapagos show versatility across different priorities, while Octopus and Opossum demonstrate particular strength in solar optimisation tasks.
- Non-Trivial Plan Advantages: L-shaped, non-trivial plans exhibit superior performance in solar optimisation scenarios, outperforming trivial plans by up to 15% in our Solar Optimisation Score. This suggests that non-trivial geometries can offer significant advantages in climates where maximizing solar gain is a priority.
- Contextual Optimisation: The choice between trivial and non-trivial plans should be guided by project-specific priorities. Non-trivial plans excel in solar optimisation but may sacrifice some spatial efficiency, while trivial plans offer more balanced performance across various metrics.
- Energy Efficiency Paradigm Shift: Our findings challenge the conventional assumption that simpler building shapes are always more energy-efficient. Non-trivial plans achieved comparable energy efficiency scores to the best-performing trivial plans, opening new possibilities for energy-efficient design with more complex geometries.
- Computational Design Implications: The integration of geometric complexity with performance optimisation represents a significant step forward in computational design for architecture. It demonstrates the potential for algorithms to generate designs that are both high-performing and architecturally interesting.

Future research directions should focus on: (i) Expanding the range of non-trivial geometries beyond L-shapes to explore other complex forms and their performance characteristics; (ii) Incorporating more sophisticated functional quality metrics to ensure that performance optimisation aligns with practical usability; (iii)Developing adaptive algorithms that can switch between or combine different solver strategies based on specific project requirements and contexts; and finally (iv) conducting real-world case studies to validate the performance of optimised non-trivial plans in built projects.

In conclusion, our study highlights the untapped potential of non-trivial floor plans in achieving high-performance architecture. By leveraging advanced computational methods and embracing geometric complexity, designers can create buildings that are not only energy-efficient and solar-optimised but also spatially rich and contextually responsive. This approach paves the way for a new paradigm in sustainable architecture that harmonizes performance optimisation with design innovation.

Author Contributions SC: concept and coordination of the study, testing of the dual graph, testing of the optimisation pipeline, writing and editing. KS: L-shaped optimisation and support on the non-trivial floor plans strategy. LA: concept, building performance calculations, comparison with other solvers. TT: support with the dual graph testing and workflow. MV: generation and optimisation of the initial building configurations. All: writing and editing.

Funding This research has been supported by each author's institution as research and development time.

Availability of Data and Material Data and scripts are available in the project page: https://github.com/seelca/non-trivial-floorplans/.

Code Availability Data and scripts are available in the project page: https://github.com/seelca/non-trivial-floorplans/.

Declarations

Conflict of Interest Not applicable.

Ethics Approval Not applicable.

Consent to Participate Not applicable.

Consent for Publication Not applicable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Abdulmajeed, A., Agkathidis, A., Lombardi, D. and Dounas, T., 2023, January. Mass-customisation of dwellings in the Middle East: developing a design-to-fabrication framework to resolve the housing crisis in Saudi Arabia. In *Proceedings of the International Conference on Education and Research in Computer Aided Architectural Design in Europe* (Vol. 2, pp. 157–164).

Aksoy, U.T. and Inalli, M., 2006. Impacts of some building passive design parameters on heating demand for a cold region. *Building and environment*, *41*(12), pp.1742-1754.

AlAnzi, A., Seo, D. and Krarti, M., 2009. Impact of building shape on thermal performance of office buildings in Kuwait. *Energy Conversion and Management*, *50*(3), pp.822-828.

Alymani, A. and Jabi, W., 2024. A graph-based computational tool for retrieving architectural precedents of building and ground relationship (BGR tool). *International Journal of Architectural Computing*, p.14780771241260853.

Alymani, A., Jabi, W. and Corcoran, P., 2023. Graph machine learning classification using architectural 3D topological models. *Simulation*, *99*(11), pp.1117-1131.

Bhasker, J., Sahni, S., 1987. A linear time algorithm to check for the existence of a rectangular dual of a planar triangulated graph. *Networks*, 17, pp.307-317.

Bhasker, J., Sahni, S., 1988. A linear algorithm to find a rectangular dual of a planar triangulated graph. *Algorithmica*, 3(1), pp.247–278.

Bostancioğlu, E.S.R.A., 2010. Effect of building shape on a residential building's construction, energy and life cycle costs. *Architectural Science Review*, *53*(4), pp.441-467.

Caramia, M., Dell'Olmo, P., Caramia, M. and Dell'Olmo, P., 2020. Multi-objective optimisation. *Multi-objective management in freight logistics: Increasing capacity, service level, sustainability, and safety with optimisation algorithms*, pp.21–51.

Carta, S., 2021. Self-organizing floor plans. *Harvard Data Science Review* HDSR.

Carta, S., Turchi, T., Spencer, N. and Vidal Calvet, M., 2023. Encoding social values of local communities in algorithmic-driven design methods. In *Formal methods in architecture - Proceedings of the 6th International Symposium on formal methods in Architecture (6FMA). A Coruna 2022.* Springer Nature.

Deb, K., Sindhya, K. and Hakanen, J., 2016. Multi-objective optimisation. In *Decision sciences* (pp. 161–200). CRC Press.

Deprez, Louise 2022. PlanGraph. Available at: https://www.food4rhino.com/en/app/plan-graph

Giagkiozis, I. and Fleming, P.J., 2015. Methods for multi-objective optimisation: An analysis. *Information Sciences*, *293*, pp.338-350.

Gonzalo, R. and Habermann, K.J., 2012. *Energy-efficient architecture: basics for planning and construction*. Walter de Gruyter.

Granadeiro, V., Duarte, J.P., Correia, J.R. and Leal, V.M., 2013. Building envelope shape design in early stages of the design process: Integrating architectural design systems and energy simulation. *Automation in construction*, *32*, pp.196-209.

Gunantara, N., 2018. A review of multi-objective optimisation: Methods and its applications. *Cogent Engineering*, *5*(1), p.1502242.

He, X., 1993. On finding the rectangular duals of planar triangular graphs. *SIAM Journal of Computing* 22(6), pp. 1218–1226.

Hegger, J. and Horstmann, M., 2008. Light-weight TRC sandwich building envelopes. In *Excellence in concrete construction through innovation* (pp. 203–210). CRC Press.

Jankovic, L. and Carta, S., 2021. BioZero—Designing Nature-Inspired Net-Zero Building. *Sustainability*, *13*(14), p.7658.

Kant, G., He, X., 1997. Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems. *Theoretical Computer Science*, 172(1), pp. 175–193.

Kistelegdi, I., Horváth, K.R., Storcz, T. and Ercsey, Z., 2022. Building geometry as a variable in energy, comfort, and environmental design optimisation—a review from the perspective of architects. *Buildings*, *12*(1), p.69.

Li, A., Tian, R., Wang, and, X. and Lu, Y., 2022. Plan Generation from Program Graph. *Machine Learning and the City: Applications in Architecture and Urban Design*, pp.167–170.

London Gov (2017). The draft London Plan. Retrieved from: london_plan_topic_paper_on_density_policy_and_details_of_research_-_2017_final.pdf

Makki et al. (2022). Wallacei. An Evolutionary Multi-Objective Optimisation and Analytic Engine for Grasshopper 3D. Retrieved from: Evolutionary Engine for Grasshopper3D | Wallacei.

March, L., & Steadman, P. (1971). Spatial allocation procedures. In *The Geometry of Environment* (pp. 303–317). MIT Press.

Parasonis, J.; Keizikas, A.; Endriukaitytė, A.; Kalibatienė, D. Architectural Solutions to Increase the Energy Efficiency of Buildings. *J. Civ. Eng. Manag.* 2012, *18*, 71–80.

Park, H., Suh, H., Kim, J. and Choo, S., 2023. Floor plan recommendation system using graph neural network with spatial relationship dataset. *Journal of Building Engineering*, *71*, p.106378.

Pugnale, A. and Sassone, M., 2014. Structural reciprocity: critical overview and promising research/design issues. *Nexus Network Journal*, *16*, pp.9-35.

Raveena, Shekhawat, K., 2023. A theory of L-shaped floor-plans. *Theoretical Computer Science*, *942*, pp.57-92.

Raveena, Shekhawat, K., Shekhawat, R. (2024) A graph theoretic approach for generating T-shaped floor plans, *Theoretical Computer Science*, *1011*, 1–29.
Rutten, D., 2013. Galapagos: On the logic and limitations of generic solvers. *Architectural Design*, *83*(2), pp.132-135.
Shekhawat, K.: A theory of L-shaped floor-plans. *Theor. Comput. Sci.* 2023, *942*, 57–92.
Showkatbakhsh, M. and Makki, M., 2020. Application of homeostatic principles within evolutionary design processes: Adaptive urban tissues. *Journal of Computational Design and Engineering*, *7*(1), pp.1-17.
Tuhus-Dubrow, D. and Krarti, M., 2010. Genetic-algorithm based approach to optimise building envelope design for residential buildings. *Building and environment*, *45*(7), pp.1574-1581.
UK Building Regulations 2010. Approved Document L Volume 1, 2021 Edition.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Silvio Carta is an architect (ARB/RIBA), Chartered Building Engineer (MCABE) and Professor of Architecture at the University of Greenwich, London, UK.

Krishnendra Shekhawat is working as an Associate Professor at Department of Mathematics, BITS Pilani, Pilani Campus. He completed his PhD from University of Geneva. He is also the founder of GPLAN services private ltd (gplan.in).

Luciano Ambrosini is a PhD Architect, Independent Researcher, Computational Designer and Environmental Design Analyst based in Naples, Italy. His career includes experience at Foster + Partners and research in AI-driven architectural solutions.

Tommaso Turchi is a researcher in human-machine interaction and human-centred artificial intelligence. He is currently an assistant professor at the University of Pisa. He received his Ph.D. from Brunel University London in 2019.

Miguel Vidal is an Architect with over ten years of experience integrating teaching and research with professional work at top practices such as Foster + Partners and Grimshaw. His academic interests revolve around Computational Design, Participatory Design, and co-design. He is currently pursuing his MPhil/PhD research at Loughborough University in 2021.