Multi-agent space planning: a literature review 2008-2017

Author 1 and Author 2
University, ABC

This paper presents a literature review on the emerging field of multi-agent space planning (MASP) in the period of 2008-2017. By MASP we refer to space planning (SP) methods in which the model itself is based on online agents that build distributed spatial representation. The agents interweave local computation of the problem and action on the environment, characterizing a distributed information process. The paper is structured in precedents, literature review, analysis and conclusion. After presenting two early precedent for MASP, it reviews 17 papers that propose novel MASP systems. Finally, it analyses the methods of representation, objectives, and solution procedures of the proposed MASP systems. It concludes presenting the research gaps and challenged for future MASP research.

Multi-agents in SP

Designing spatial arrangement is in the core of architecture and has traditionally been solved by human-centered methods. Since the early 1960s, a main branch of the CAAD research called automated space planning (SP) consisted of methods for allocating architectural spaces based on computational data-structures, algorithms and techniques from Artificial Intelligence (AI). Design theorists and researchers dislocated the activity of design from the creative exploration of educated individuals and groups in favor of a framework of rational problem-solving activity.

Traditionally automated SP methods adopts a well-structured interpretation of design problems, which makes them tractable by canonical AI techniques [1]. For example, SP can be formulated as the finding of the optimal locations for a set of discrete interrelated objects in a discrete space, packing pre-defined shapes in a given boundary, or dividing a boundary to satisfy the sizes and neighborhood relationships of certain rooms.

By simplifying design constraints and goals, the task environment becomes observable, deterministic, static, and known. The computation of the solution can be isolated from the problem formulation, resulting in a sequential offline search from an initial state to an optimal solution. This can be noticed in recent SP trend, which focuses on evolutionary optimization [2]. The solution is generated by an offline agent that not only understands the environment and how its actions affect it (it has a world-model), but also has a metric to evaluate its performance.

A subset of recent SP research addresses the ill-defined aspects of design problems and processes that have been simplified by this working hypothesis of high rationality. Human-centered design operates with multiple constraints by adopting non-linear models of design exploration. It involves strategies such as feedback, problem restructuring, and results in a dynamic and unreliable task environment for automation. To work in this environment, these researches formulate SP as model based on online multi-agents that can navigate in unreliable and noisy task environments to build a distributed spatial representation. These agents interweave local computation of the problem and action on the environment, characterizing a distributed information process that is potentially adaptable and robust. The focus shift from using AI agents to solve a well-defined problem to the use of multi-agents to explore spatial arrangements.

Given the potential of MASP in design, and the secondary role it had in previous SP reviews [1, 3–6], this paper presents a specific and structured literature review and exposes the challenges for future research.

Two precedents

One of the most interesting precedents for MASP is the work of Guy Weinzapfel and associated researchers in the late 1960s and 1970s at MIT.

a) IMAGE [7]

b) IMAGE [8]

IMAGE is based on the formulation of the SP as a multicriteria, design exploration problem. Each spatial unit is represented as a custom cuboid and the spatial arrangements aim at satisfying the objectives and constraints defined by the users. The program incorporates a series of relevant spatial constraints to be satisfied – adjacent, keep out, overlap, relative position, ratio, width / depth / height, x / y / z position, distance [near, far], shared wall, enclose, next, on top of, floor, above, align, and visual access. A constraint graph stores the cells as the main nodes and relations as intermediate labelled nodes. It is possible to create more complex boundaries by establishing a relative position constraint between component rectangles.

Each space can be evaluated in relation to the satisfaction of the constraints both by specific queries or by a general ranking. The generation procedure can be done by the user and/or by the computer. In the case of the computer, an iterative routine traverses the list of existing spatial units. It fixes all the other units of the system and looks for changes in the current state. The improvement for one constraint can conflict with another constraint or other units' constraint.

A Least Mean Squares Fit is used as a non-destructive optimization, changing the descriptors of a unit (dimension, location and rotation) to achieve the best local improvement – i.e. the change that results in minimum error regarding all the linear equations of the constraints. Applied sequentially to all the spaces, this solution procedure incrementally satisfies the constraints and converges to an optimal state – which may be local. As a result, the agents react to the constraints around it, improving the state of the whole arrangement.

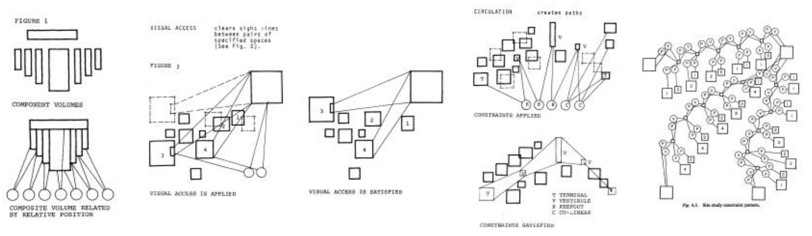


FIGURE 1
COMPONENT VOLUMES
COMPOSITE VOLUME RELATED BY RELATIVE POSITION
VISUAL ACCESS
clears sight lines between pairs of specified spaces (See Fig. 2).
FIGURE 2
VISUAL ACCESS IS APPLIED
VISUAL ACCESS IS SATISFIED
CIRCULATION
creates paths
CONSTRAINTS APPLIED
T TERMINAL
V VESTIBULE
K KEEPOUT
C CO-LINEAR
CONSTRAINTS SATISFIED
Fig. 4.5. Site study constraint pattern.

Fig. 1 IMAGE [7, 8].

c) **Architecture-by-yourself** [9]

YONA aims to empower non-expert users by providing a method to produce residential design. The name YONA is not only a tribute to the influent utopian architect Yona Friedman but also references Friedman's theoretical approach published one year earlier in [10]. Weinzapfel and Negroponte adopted Friedman's graph-based SP framework to develop a three-stage representation of the residential layout problem with incremental design information: graph embedding, bubble-diagram, and schematic plan.

Initially, the user defines the spaces of the residence with specifications of area and connections. The program's routine tests the graph to ensure its planarity, then uses a heuristic approach to generate an embedding that satisfies the user's spatial specifications. The user can rearrange the graph sliding each node to a new position. After a satisfactory arrangement, the program draws an offset boundary and a dual-graph, creating multiple polygonal partitions around the original nodes. Then, it generates b-spline curves inside these partitions, forming a bubble-diagram. In the last step, YONA does not try to generate the floor plan automatically, but it provides drawing tools for the user to sketch the shape of the rooms over the bubble.

Compared to IMAGE, YONA upgrades the human-machine interaction component of the system, aiming to empower non-expert designers. However, it comes with the cost of breaking the design feedback in three stages, and of reducing the number of design constraints.

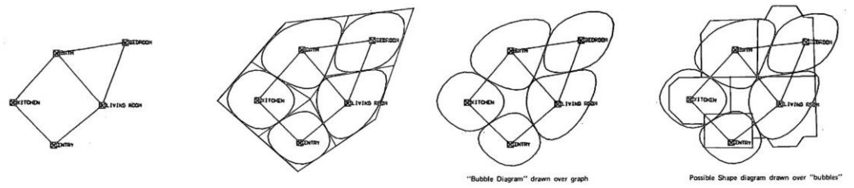

"Bubble Diagram" drawn over graph
Possible Shape diagram drawn over "bubbles"

Fig. 2 stages of YONA: top: graph embedding, dual-graph connected to offset boundary to generate bubble diagram, bubble-diagram, and plan [9].

Multi-agent SP 2008-2017

This section contains short descriptions of seventeen papers, emphasizing details about the MASP system, its scope, concept and algorithms. Different databases were used to look for papers, but the main source is the CumIn-CAD, which provides access to abstracts and papers of ACADIA, CAADRIA, eCAADe, SIGraDi, ASCAAD, CAAD futures, and DCC.

1) Material & Science [11]

Inspired in the operators of evolutionary developmental biology, Sean Ahlquist and Moritz Fleischmann created a set of local geometric rules for a rule-based SP system. Particularly, the authors translate the process of specialization of organisms by folding to geometrical subdivision operations. Each unit of the system is a cuboid that has the capacity to divide itself into smaller volumes that are appropriate for different bits of the program activities. The subdivision of a volume is based on local plane, that is translated and rotated according to a sample from a pre-defined list.

The paper is neither clear about the process of distribution of the program nor of the union of the volumes. Apparently, the resulting volumes can be combined with the neighbors, generating complex packing of activities. This algorithm is divided in two steps. The step "program by volume" occurs iteratively in the outer loop, selecting a volume that most closely fits to a particular program. The second step, “program by proximity”, looks for the nearest neighbor volume to connect and insert a program. Supposedly, the program is distributed in the available bits generated by volume subdivision, resulting in a polyhedral packing.

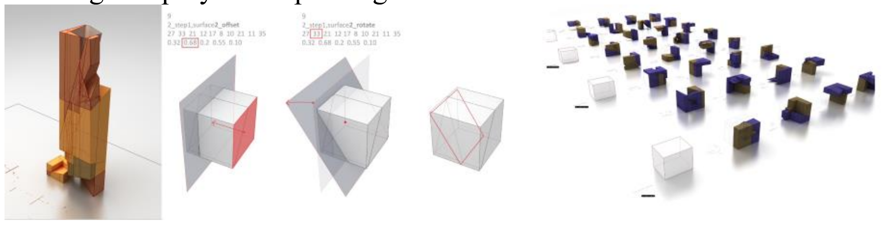


Fig. 3 Generation of envelopes based on local subdivision and distribution of the program [11].

2) Space diagrams [12]

3) Emergent Space Diagrams [13]

In Space diagrams [12], Ireland used consolidated ant pheromone routing algorithms to develop the Sniffing Space, a model for the exploration of routing alternatives between multiple points in a 2d-grid. In [13], Ireland extends the model Sniffing Space to generate diagrams of architectural space in early-stages of design. The extended Sniffing Space is based on the interaction between an array of ant colonies. Each colony represents a space/activity in the architectural brief and they form a graph with edges indicating association or adversity.

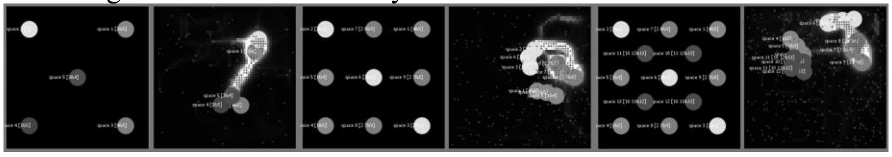


Fig. 4 Sniffing Space: spatial association generating zones [13]

The colonies produce ants, which are soldiers that search for associated nest sites. After discovering an associate nest, the soldiers return to the nest with a specific alternative pheromone marking a trail between the associate nest and its home nest. Each ant will do the same search and return process until it finds all the associated colonies; then it dies. The evaporation of the pheromone ensures that ineffective trails or dead ends will disappear.

The nest will gradually follow the associated returning ants with slower steps, following the dominant pheromone path to the associated nests. The nests keep reproducing soldiers until they are clustered with all the associated nests. The spatial organization emerges from the clustering of all the associated colonies. The associations can be asymmetrical, which may result in non-adjacent zones connected by pheromone paths [13].

Ireland developed an user interface for the model [13] to control parameters such as diffusion rate, evaporation rate, splodge rate at destination, amount of pheromone, splodge trail, and max smelliness of trail.

4) Sniffing Space II [14]

Ireland customized the Sniffing model presented in Space Diagrams [12] to represent the flow in a specific architectural space. He extends Sniffing Space by incorporating more features to the model and to the agents.

Firstly, he encoded a goal to each agent, simulating a passenger routing in an airport, where there is a clear arrival and departure sequencing. Each ant has an agenda with a sequence of destinations or hubs, that represent specific spaces of the airport.

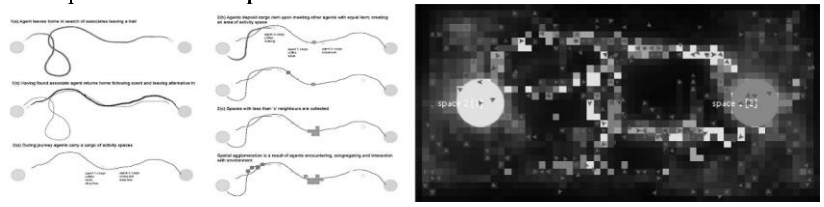


Fig. 5 Routing between destination points and Spatial arrangement from cargo clustering [14].

Ants not only follow pheromones and navigate between destination points but also respond to other agents sharing similar qualities. These qualities are encoded as random activity cargos that are assigned to each agent. As agents with similar items meet in a trail, they deposit the cargo in their current location. Every time another agent with similar cargo passes by this location, it also drops its cargo, reinforcing a cluster of cargo space. In contrast, a passing agent might collect dropped cargos that have less than n neighbors, preventing scattering of activities. As a result, the model generates a grid with activity cells organized between destination points.

5) Stigmergic Planning [15]

Ireland customize the Sniffing Space II [14] to address the dimensioning of architectural spaces. He extends this model to generate spatial arrangements of loosely packed rectangles [15]. Each nest is initialized with a rectangular boundary with random dimensions x and y that satisfy its required area. There are two variations in the process of an ant detecting a nest, which influence the development and result of the rectangular packing. In the first variation, the ant must go to the core of the nest. In the second, the ants only must touch the nest's rectangular boundary. Moreover, each nest changes its dimensions when it interacts with neighbor nests. It has three behaviors: adapts x and y when it nestles, moves away from overlapping adversary nests, and overlies with associated nests. It results in a loose packing of rectangles, defined by the connections.

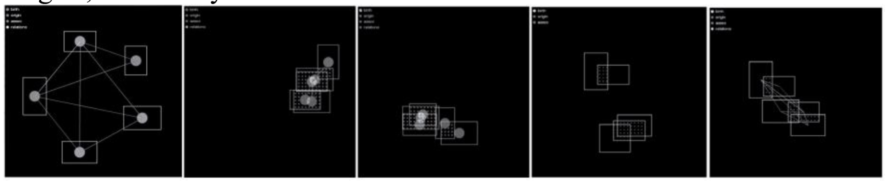


Fig. 6 G&T algorithm. 1st: Nest relation between 5 colonies; 2nd and 3rd: model where nest is identified by the nucleus; 4th and 5th: model where the nest is identified by the rectangular boundary [15].

6) Associative Spatial Networks in Architectural Design [16]

Inspired by Cybernetics, John Harding and Christian Derix propose an exhibition hall layout that adapts according to the topology of the exhibition and potentially to the users' feedback. The system is divided in 3 parts: the generation of a spatial plan that relates all the exhibition; the generation of the specific spatial plan for the individual exhibits; the spatialization of the plans over a modular layout arrangement.

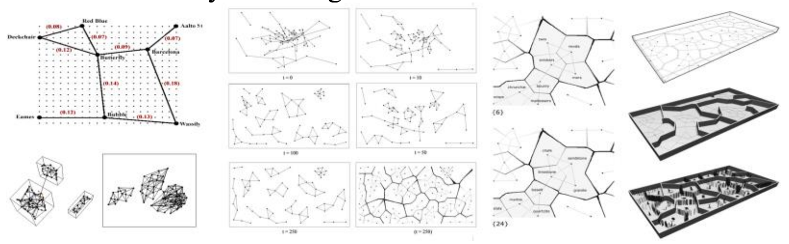


Fig. 7 Top-left: table with features from an exhibit; top-left: weighted graph generated by SOM; bottom-left: growing neural gas mapping in 3d and real-time embedding using a particle spring system in 2d. middle: repulsion algorithm distributes graph evenly over the exhibition space; right: translating a graph into the space of the exhibition with a 2d Voronoi diagram [16].

For the first stage, each individual exhibit is described in a table with its qualities. A self-organizing map reduces the multidimension feature space of this representation to $\mathbb{R}^2$, preserving most of its topological associations. Then they connect the closest neighbors, creating a planar graph with edges weighted by the normalized length of edge distance.

The second stage involves finding the difference between multiple exhibits using the spectrum for the Laplacian matrix of the graph. The multiple spectra have different connectivity and dimensionality. The authors developed a growing neural network able to adapt – by insertion and deletion of neurons - to the varying dimensions of the feature space and to cluster the graphs according to similar topologies.

The graphs with the spectrum closest to the average in each cluster are selected and organized on the exhibition hall layout by a repulsion algorithm. Using a Voronoi diagram, each node becomes a polygonal exhibition cell. Where there is no connection between nodes, the edge of the Voronoi cell becomes a wall. The length of the connection between nodes is translated inversely as a permeability in the boundary between the respective cells - reaching a lower threshold value where there is no wall.

7) Floating Bubbles [17]

Inspired in the agent behavior of IMAGE program [7, 8], Hua Hao and Jia Ting-Li developed the program Floating Bubbles, which is an agent-based model based on bubble diagrams. The bubbles are represented as circles connected by edges. The objective of the system is to solve the adjacency of bubbles connected by edges. To solve the diagram, the system assigns two basic forces to the agent: attraction and repulsion.

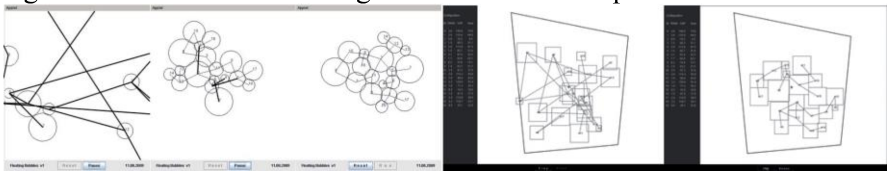


Fig. 8 *Floating Bubbles. left: three stages in the basic floating bubble system; right: two stages of the generative process* [17].

Attraction is proportional to the length of the vector between the boundaries of two connected bubbles. Repulsion is proportional to the overlapping area between two bubbles, and pushes each bubble away. The final vector for each bubble is the sum of the attraction vectors with all its neighbors.

Only with attraction and repulsion forces, the bubbles get stuck in local optimum. To avoid these situations, a heuristic moves this bubble towards a connected bubble when an adjacent requirement stays unsatisfied for too long. As the heuristic restarts the interactions in a slightly different state, eventually it reaches the equilibrium and satisfy all the adjacencies.

To display the potential for real applications, the authors used Floating bubbles as a generative tool for a project of a two-story museum called Clouds Collective. The system was expanded with two layers of bubbles, to represent the different floors of the museum. It was also integrated with custom spatial elements in a BIM model, using squares instead of circles as the shape of the bubbles. After a solution was achieved, the BIM model generated architectural representations such floor plans and 3d models.

8) Stigmergic Space [18]

In this paper AnnaLisa Meyboom and Dave Rees describe the Stigmergic Space Adjacency Software (SSAS). It is a multi-agent system based on the collective behavior of insects regulated by pheromones, such as ants or termites. The different agents represent activities of architecture brief connected by adjacency relationships. Each agent has a specific value in a RGB color space and is compatible with agents within a close color range – meaning that they should be adjacent. The agents dispute the territory in a three-dimensional array composed of nodes.

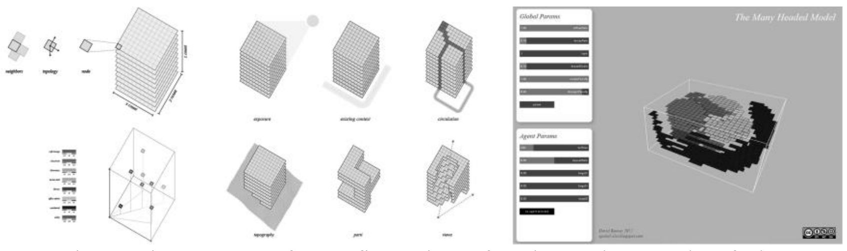


Fig. 9 Stigmergic Space. Left: configuration of nodes and example of pheromone value for programs; middle column: sources create templates representing external influence, and node masking; Right: a solution [18]

The space is composed of nodes, i.e. discrete spatial unit of the grid. They can be occupied by the agents and each is connected to the three-dimensional Von Neumann neighbors. Nodes can carry a pheromone value or can even produce it. In the former case, as an agent occupies this node, its pheromone value tends towards the agent's value, "gradually saturating the local environment with the same concentration through diffusion" [18]. In the latter, its value can be pre-defined to stimulate spatial templates over which the agents will dispute the territory – such as circulation systems, boundary areas closer to the light exposure, etc.

Besides customizing the initial state with pre-defined pheromone patterns, the user also can select and deactivate a set of nodes, sculpting the three-dimensional array where the agents look for pheromones. In this case, conditions such as the topography, a pre-defined *parti* or voids to preserve views, can be embedded in the model.

The user can tune all these different parameters of the system to look for a satisfactory spatial configuration. However, the authors report that the system gets stuck in scenarios with limited space and large population.

9) Fuzzy Layout Planner [19]

Bayraktar and Çağdaş describe the Fuzzy Layout Planner, an editor to create and edit dynamic bubbles diagram for early-stages of design. There is an extensive list of commands, such as [create] bubble, select, move, copy, cut and join, group/ungroup, name, rotate, resize, paint, layer, import image, export image, pan and zoom, view options, adjust overlap spaces, and adjust repulsion forces. Each bubble is a rectangle with filleted corners. The reason why the editor is called fuzzy is because the boundaries of the rectangles have a “slighty fuzzy” motion.

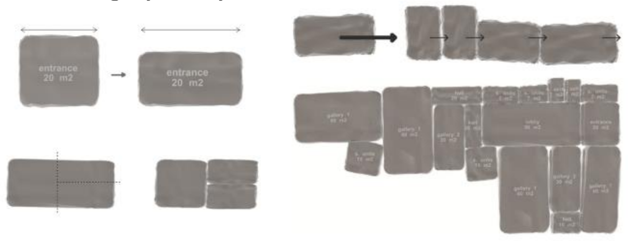


Fig. 10 Fuzzy Layout Planner. Top left: dimensions of bubble adapt to preserve area; bottom-left: bubbles can be cut into smaller bubbles; top-right: bubbles behave according to adjustable repulsion forces; bottom-right: bubbles can be grouped and moved together [19].

Each bubble is associated with a name and an area when it is created. As the user changes one of its dimensions, the other dimension automatically adjusts to preserve the assigned area. The interaction between bubbles happens mainly manually, but there is a repulsion force that ensures the packing of a cluster of bubbles.

10) Exploring the Bubble Diagram [20]

The prototype Interactive Bubbles was developed as an agent-based system in a 2D environment with event-based interactive tools.

Each agent in the system is represented a circular bubble. The agents are connected to the neighbors and these connections are interpreted as a force of attraction. To avoid overlapping between bubbles and to be able to constrain a system of bubbles inside a custom area, there are also repulsion forces. The behavior of each agent is a result of the sum of the vectors of the forces of attraction and repulsion.

However, with local forces guiding the agents, the system gets stuck and agglomerates without achieving an optimal solution. Inspired by Simulated Annealing, a non-destructive heuristic uses a varying temperature to prevent getting stuck in dead-ends. Whenever an agent is far from the success, its temperature is higher, reducing its area and the effect of the forces of attraction and repulsion. Therefore, when a bubble is in a good position, it stabilizes and reaches its real area, but when it is far from a good position, it is in an exploratory mode, which means that it is smaller and can move between bubbles.

Spatial arrangements are not only governed by connections between spaces but also by environmental conditions or pre-determined structures. In this regard, the system has a top-down function "create boundary", which restricts drastically the solution space to the interior of a user-defined polygon.

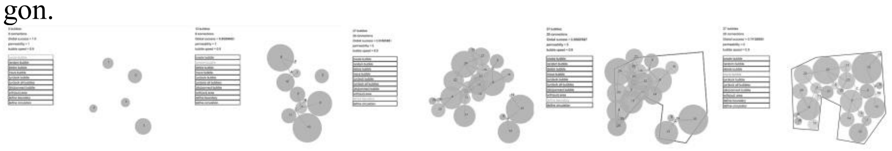


Fig. 11 Interactive Bubbles [20].

11) SP and Preliminary Design using Artificial Life [21]

Ruwan Fernando developed a SP prototype for early stages of design exploration. The system combines two distinct parts to promote the interaction of a human designer with the computer.

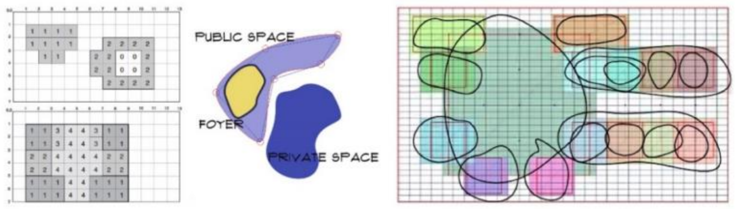


Fig. 12 left: grid-planning system; center: blob region; right; integration of grid-planning system and blob regions [21].

The first part is a 2d bubble diagram. Each bubble is defined by a set of control points that forms a polyline and controls its boundary. The physics simulation is restricted to an individual blob to provide an "intuitive feel" for the movement triggered by the user.

The second part of the system consists of a grid of cells containing numerical values and an operator with five descriptors (row, col, width, length, and state) to draws a rectangle on the grid. The solution procedure is a genetic algorithm (GA) with the genome encoding a list of rectangles, each one represented by the descriptor as a five bit-string. The mutation of the GA is defined by bit flips according to a probability distribution and a single point crossover. The fitness function is based on the area and connections of the spaces, as configured by the blobs.

The SP system works by oscillating between these two parts. The user defines a set of blobs, which results in areas and adjacencies for the activities. Then the GA looks for an optimal set of rectangles on the grid that satisfy these requirements. In the paper, it is not clear if the bubbles adapt to the grid after an optimization.

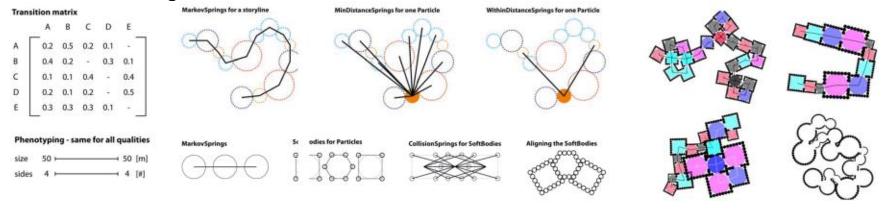
12) The Generation of Possible Space Layouts [22]

Influenced by Gordon Cullen’s concept of serial vision, Christensen proposes a method to generate spatial arrangements based not only on the satisfaction of functional requirements but also on spatial qualities. Initially, the user defines all spatial phenotypes by its quality, such as narrow, dark, low, high, etc. Then, these spaces are associated with a range of values for its geometric properties (width, openings, height, length, size, sides, walls, etc.). Each shape is a convex polygon represented by a soft body composed of a central particle and surrounded by children particles on the border.

The generative process starts with a random space. At each time step, the arrangement grows with a Markov chain, which generates a new space based on the last space in the sequence an in an user-defined transition matrix.

The behaviors of the system are defined by a variety of different springs. A class MarkovSpring connects the central particles of the two adjacent spaces. A CollisionSprings attaches part of the peripheral children particles of adjacent spaces. The class minDistanceSprings connects non-adjacent spaces with different qualities, preserving a minimum distance in the arrangement. In contrast, the class WithinDistanceSpring connects non-adjacent spaces with the same qualities. It is a spring that can attract and merge the pair of similar spaces as they are within a certain distance, establishing clusters with three or more connections.

Different transition matrices and spatial properties will generate different storylines, which can be developed in parallel and eventually merge if they share similar qualities.


Transition matrix

	A	B	C	D	E
A	0.2	0.5	0.2	0.1	-
B	0.4	0.2	-	0.3	0.1
C	0.1	0.1	0.4	-	0.4
D	0.2	0.1	0.2	-	0.5
E	0.3	0.3	0.3	0.1	-

Phenotyping - same for all qualities

size	50	50	[m]
sides	4	4	[#]

MarkovSprings for a storyline
MinDistanceSprings for one Particle
WithinDistanceSprings for one Particle
MarkovSprings
odies for Particles
CollisionSprings for SoftBodies
Aligning the SoftBodies

Fig. 13 left: transition matrix and phenotype; center: classes for springs connecting particles; right: possible results [22].

13) Agent-based Models for Computing Circulation [23]

14) Spatial Agglomerates [24]

Renee Puusepp presents a model inspired by the work of Bill Hillier and Paul Coates: the Dwelling Agglomerator. It addresses the morphogenesis of settlement geometry based on the local interactions of discrete spatial units.

It is composed of three cyclic modules: dwelling locations, generation of geometry, and analysis of satisfaction of resulting dwellings. Based on Reynolds’ Boids, the first module sets the movement of the individual agents as a flocking simulation. Instead of the canonical Boid behaviors, Puusepp establishes a set of problem-specific flocking rules for the agents:

- Further than a given range, it moves towards the closest neighbor n_1.
- Closer than a given range, it moves away from the closest neighbor n_1.
- Within the given range of the closest neighbor n_1 and with less than 60 degrees in relation to the edge of the two neighbors n_1 and n_2, it orbits around n_1 to reach zero degrees on this edge.
- Within the given range of the closest neighbor n_1 and with between 60 and 120 degrees in relation to the edge of the two neighbors n_1 and n_2, it orbits around n_1 to reach 180 degrees.
- Within the given range of the closest neighbor n_1 and with an angle smaller than 60 degrees in relation to the edge of the two neighbors n_1 and n_2, it orbits around n_1 to reach 90 degrees.

The second module spatializes the position of the agents by associating it with the cells of a Voronoi Diagram. The remaining cells of the diagram are treated as an open space between the dwelling units.

The third module analyses the resulting spatial pattern, based on some criteria, such as area, direct sunlight or accessibility – defined as the inverse of total topological distance to all units [24]. This module calculates a score for each unit, defining if its current configuration is satisfactory or not. This information is sent back to the first module so unsatisfied units will become active and will look for a better performance.

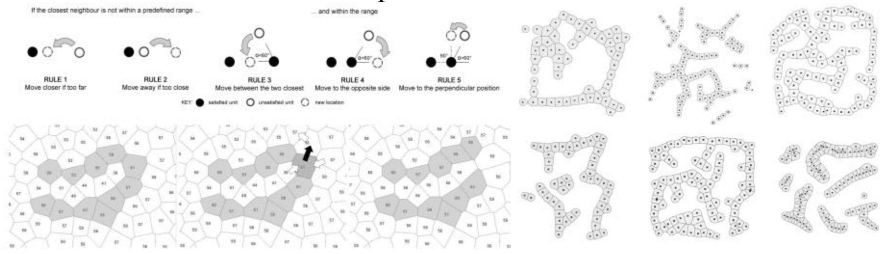

If the closest neighbour is not within a predefined range ...
... and within the range
RULE 1
Move closer if too far
RULE 2
Move away if too close
RULE 3
Move between the two closest
RULE 4
Move to the opposite side
RULE 5
Move to the perpendicular position
KEY: satisfied unit unsatisfied unit new location

Fig. 14 Spatial Agglomerates. Left: flocking rules; second row: creation of new access routes; right: agglomeration layouts [23]

15) A Cell-inspired Model of Configuration [25]

16) An Artificial Life Approach to Configuring Architectural Space [26]

Ireland proposes a MASP system based on a set of discrete spaces/activities inspired by Eukaryotic cells. This autonomous artificial cell is called an actant and is composed of an internal central node (or nucleus), multiple nodes representing the vertices of the polygon, the boundary link between vertices forming a polygon, and its internal region. The model is based on adjacency and can express arrangements with different relations - from containment (one cell inside another) to disjunction (two isolated cells).

The sensors and actuators of the actants are its nodes. The boundary nodes and the nucleus maintain the consistency of the shape by attraction-repel forces. The external nodes move the actant in the environment, triggering a search in the local vicinity for neighbor actants. As the boundary-receptors detect differences in the environment, the actant keeps moving. If no actant is found, one of the boundary nodes is selected to be a hunter and explore beyond the vicinity. When the actant finds a neighbor, their boundary nodes react, following the pre-defined behavior.

The sensor nodes of the actant not only detect other nodes but also emit and track pheromone. The pheromone functions as a gradient that might attract or repulse the actant – according to its relation to the pheromone producer. The actant only produces the pheromone when it wants to be found by its associates. When it is evading or seeking, it does not produce the pheromone, to prevent the agglomeration of non-associate actants. Therefore, the spatial diagramming is generated by a bottom-up approach based on a hunter-prey dynamic between different cells.

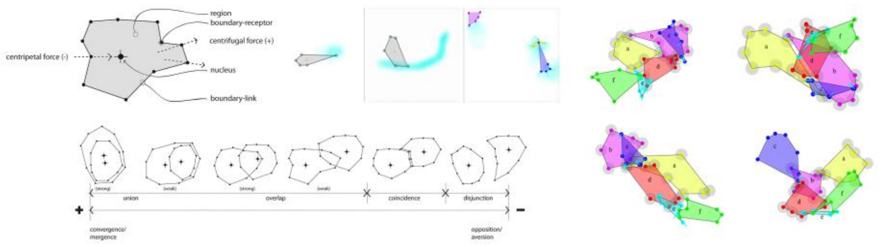


Fig. 15 top-left: an actant and its components; ; bottom-left: scale of consolidation top-center: actants' vertices reacting to pheromone; right: actants settled in different configurations according to their associations [26].

17) Responsive Algorithms [27]

Frano Bazalo and Tane Moleta explored two SP workflows using algorithms from the available components and plug-ins inside a parametric modelling editor. Their investigation starts by defining a workflow that uses physics simulation to solve an adjacency graph and the packing circles, preserving the control of the arrangement by the user. A Voronoi algorithm partitions the resulting packing, results in the dynamic arrangement of Voronoi cells according to an embedded adjacency graph.

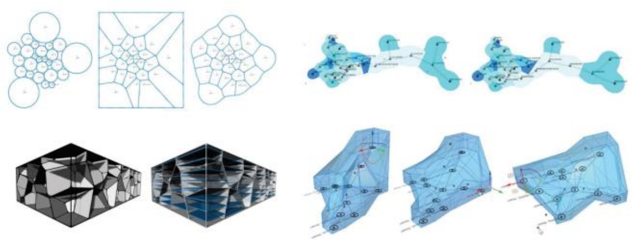


Fig. 16 top-left: circle packing, Voronoi diagram and Voronoi diagram with radii; top-right: force-directed graph over Voronoi diagram with radii (the author refers space syntax); bottom-left: 3d Voronoi cells; bottom-right: three-dimensional packing with a skin [27].

Then, this workflow is extended to the three-dimensional space. The nodes of the adjacency graph are the center of packed spheres. A three-dimensional Voronoi partition generates a polyhedral boundary, but the authors also cite geometric entities and algorithms that can extend the volumetric output, such as cubes, cuboids, and meta-balls.

Analysis

This section provides a structured outline of the selected papers based on the categories representation, objective, and solution procedure.

Table 1 Spatial representation

#	ref	Year	nickname	**Graph**	**Bubble diagram / packing**	**Polygon Polyhedron**	**Grid**	**Voronoi diagram**
a b	[7] [8]	1971 1975	IMAGE	■		■		
c	[9]	1976	YONA	■	■	■		
1	[11]	2008	Evo-dev			■		
2 3	[12] [13]	2008 2009a	Sniffing Space				■	
4	[14]	2009b	Sniffing Space II				■	
5	[15]	2010	Stigmergic Planning				■	
6	[16]	2010	Spectral graphs	■				■
7	[17]	2010	Floating Bubbles		■			
8	[18]	2013	Stigmergic Space				■	
9	[19]	2013	Fuzzy Layout planner		■			
10	[20]	2014	Interactive bubbles	■	■			
11	[21]	2014	AL bubbles		■		■	
12	[22]	2014	Spatial narratives	■		■		
13 14	[23] [24]	2014a 2014b	Dwelling Agglomerator					■
15 16	[25] [26]	2015a 2015b	Actants			■		
17	[27]	2015	Responsive algorithms	■	■	■		■

While very specific systems for representation such as dimensionless representations or Smith diagrams are absent, all other categories presented by [1] – regular grids, polygonal representations, and graph - are present in the contemporary researches.

- The graph is generally used to represent topological connections of the spaces, so it is combined with a complementary geometry to represent spatial units.
- The use of bubble diagrams for interactive systems is still consistent. With the use of physics simulation, the online packing of circles enables an easy interaction with the user (#7, 9, 10, 11 and 17).
- Polygonal representations are recently associated with physics simulation, resulting in adaptable irregular shapes.
- Grids are generally used for cellular-based simulations of complex systems, such as in ant foraging (#2, 3, 4, 5, and 8). Only one example in the review (#11) does not use pheromone-based algorithms.
- The use of Voronoi diagrams to generate polygonal or polyhedral cells is present in cases where it is necessary to translate from a population of points in space to a set of packed polygons. In general, the design problems presented in these examples are very specific (#6 and 13) or the results are intentionally diagrammatic (#17).

Table 2 objectives

#	ref	year	nickname	Adjacency	Area Volume	Shape	Visual access	Relative position	Accessibility	Containment	Exposure
a b	[7] [8]	1971 1975	IMAGE	■	■	■	■	■	■	■	■
c	[9]	1976	YONA	■	■	■					
1	[11]	2008	Evo-dev		■						
2 3	[12] [13]	2008 2009a	Sniffing Space	■							
4	[14]	2009b	Sniffing Space II	■							
5	[15]	2010	Stigmergic Planning	■	■						
6	[16]	2010	Spectral graphs	■							
7	[17]	2010	Floating Bubbles	■	■						
8	[18]	2013	Stigmergic Space	■						■	■
9	[19]	2013	Fuzzy Layout planner	■	■	■					
10	[20]	2014	Interactive bubbles	■	■					■	
11	[21]	2014	AL bubbles	■	■	■					
12	[22]	2014	Spatial narratives	■	■	■					
13 14	[23] [24]	2014a 2014b	Dwelling Agglomerator		■				■		■
15 16	[25] [26]	2015a 2015b	Actants	■	■						
17	[27]	2015	Responsive algorithms	■	■						

In terms of objectives, there is a huge contrast between IMAGE and current proposals. IMAGE contained a list of seventeen constraints that could be combined to guide the improvement of the arrangement. Current systems are more abstract as they focus on basic constraints such as adjacency and area/volume. In some of these systems, relations such as exposure or visual access can be customized by using the tools for adjacency. For example, (#8) allows the use of pheromone markers to promote the exposure or privacy of a part of the activities to the exterior of the building. Some systems have an explicit implementation of different objectives, such as the containment of the agents in a custom grid (#8) or area (#10), or an objective function to evaluate accessibility and exposure (#13 and 14).

Table 3 Solution procedure. LMSE: Least Mean-Square Error; H: Heuristic, Ph: pheromone; Fl: flocking; PF: physical forces; Sub: subdivision; GA: genetic algorithm; ML: Machine Learning; MC: Markov chain; DC: direct control by the user.

#	ref	year	nickname	LMSE	H	Ph	Fl	PF	Sub	GA	ML	MC	DC
a b	[7] [8]	1971 1975	IMAGE	■				■					■
c	[9]	1976	YONA		■			■					■
1	[11]	2008	Evo-dev						■				
2 3	[12] [13]	2008 2009 a	Sniffing Space			■							
4	[14]	2009b	Sniffing Space II			■							
5	[15]	2010	Stigmergic Planning			■							
6	[16]	2010	Spectral graphs					■			■		
7	[17]	2010	Floating Bubbles		■			■					
8	[18]	2013	Stigmergic Space			■							
9	[19]	2013	Fuzzy Layout planner					■	■				■
10	[20]	2014	Interactive bubbles		■			■					■
11	[21]	2010	AL bubbles					■		■			■
12	[22]	2014	Spatial narratives					■				■	■
13 14	[23] [24]	2014a 2014b	Dwelling Agglomera-tor				■						
15 16	[25] [26]	2015a 2015b	Actants			■		■					
17	[27]	2015	Responsive algorithms					■					■

The topic of solution procedures presents a remarkable difference between current systems and systems from decades ago.

Three agent-based algorithms - ant-foraging, flocking and physics simulation - are present in almost every recent SP system of the literature review (#2-17). The most dominant algorithm here is the physics simulation, which was introduced to SP by [23]. Some works implement a custom version of attraction and/or repulsion forces (#a-c, 6, 7, 9 and 10), while others use more complete implementations of particle-spring systems (#11, 12, 15, 16 and 17). It has recently been incorporated in real world practices [28, 29].

The second most common agent-based algorithm is pheromone ant-foraging (mostly by the same researcher). Initially, it was customized and used as an independent SP solution procedure (#2-5 and 9). The agents traverse the cells of a grid following the stimuli of pheromones cast by moving or static entities. In a recent hybrid application (#15 and 16), it uses the pheromones to guide spatial unit structured with physical springs.

The last agent-based algorithm – flocking – has only one occurrence (#13-14). In this example, the author used a flocking algorithm inspired by Reynold's Boids, but with customized rules for SP.

Some of the papers present hybrid methods that incorporate techniques outside of the scope of agent-based modeling, such as Least-square optimization to guide the agent (#a), GA to generate solutions in parallel to the agent (#11), Machine Learning networks (#6) and Markov Chain (#12) to generate the agents.

Direct user interaction seems to be a dominant trend, except in implementations that strictly use ant-foraging algorithms (#2-5, and 8). In these cases, the user controls the systems by setting the grid or controlling the parameters related to the pheromone.

Final considerations

MASP seems to be a prominent area with a common ground of ideas. Despite the diversity of solution procedures in the papers, the idea of a design cycle is pervasively used to reference the interaction of the user with the program or the internal data flow of the program. Some papers even organize the cycle as a diagram (#6, 10, 11, 13, 14, 15, 16).

However, the open-ended character of the works also results in a lack of scientific rigor. While most papers reference general ideas that inspired their approach – such as cybernetics, semiotics, biology, etc. –, few of them (#a-b and 7) provide some formal description of the method for the solution procedure, which is generally not enough to reproduce the algorithms. Only one paper presents a quantitative evaluation of the convergence of the solution procedure (#7). Three papers have an application of the design method to concrete architectural problems (#a, b, and 7), but without a structured analysis of the result.

These are some challenges in the field:

- Develop agency beyond physics simulation to solve spatial conflicts.
- Incorporate multicriteria for design exploration in a single model.

- Address geometric representation beyond the level of diagrams, considering the integration with important spatial elements, such as the circulation networks. Some papers suggest a direct circulation between spaces and one paper (#13) suggests a separated system to generate circulation.
- Provide a solid scientific ground to MASP research, incorporating an evaluation of the interaction with the designer, the capacity to support problem reformulation, and robustness to changes in the design process.

References

1. Mitchell WJ (1977) Computer-aided architectural design. Mason Charter Pub, New York
2. Calixto V, Celani G (2015) A literature review for space planning optimization using an evolutionary algorithm approach: 1992-2014. In: Project Information for Interaction: Proceedings of 19th SIGraDi conference. Blucher, Santa Catarina, pp 662–671
3. Liggett RS (2000) Automated facilities layout: past, present and future. Autom Constr 9:197–215
4. Homayouni H (2000) A Survey of Computational Approaches to Space Layout Planning (1965-2000). Dep Archit Urban Plan Univ Wash
5. Hsu Y-C, Krawczyk RJ (2003) New generation of computer aided design in space planning methods: A survey and a proposal. In: Proceedings of the 8th CAADRIA conference. Bangkok, pp 101–116
6. Lobos D, Donath D (2010) The problem of space layout in architecture: A survey and reflections. arquiteturarevista 6: . doi: 10.4013/arq.2010.62.05
7. Weinzapfel G, Johnson TE, Perkins J (1971) IMAGE: an interactive computer system for multi-constrained spatial synthesis. In: Proceedings of the 8th Design Automation Workshop. ACM, pp 101–108
8. Weinzapfel G, Handel S (1975) IMAGE: computer assistant for architectural design. Spat Synth Comput-Aided Build Des 61–68
9. Weinzapfel G, Negroponte N (1976) Architecture-by-yourself: an experiment with computer graphics for house design. In: ACM SIGGRAPH Computer Graphics. ACM, pp 74–78
10. Friedman Y (1975) Toward a Scientific Architecture, 1st ed. MIT press
11. Ahlquist S, Fleischmann M (2008) Material & Space Synthesis Strategies based on Evolutionary Developmental Biology. In: Proceedings of the 28th ACADIA Conference. Minneapolis, USA, pp 66–71
12. Ireland T (2008) Space diagrams: The problem of spatial arrangement and the automatic generation of architectural plans. In: Architecture in computro - Integrating Methods and Techniques: Proceedings of the 26th eCAADe Conference. Antwerpen, pp 91–98

13. Ireland T (2009) Emergent space diagrams. In: Joining Languages, Cultures and Visions: Proceedings of the 13th CAAD Futures Conference. pp 245–258
14. Ireland T (2009) SNIFFING SPACE II. In: Joining Languages, Cultures and Visions: CAADFutures. pp 214–227
15. Ireland T (2010) Stigmergic planning. Proc 33rd Annu Conf Assoc Comput Aided Des Archit ACADIA 183–189
16. Harding J, Derix C (2011) Associative spatial networks in architectural design: Artificial cognition of space using neural networks with spectral graph theory. Des Comput Cogn 305–323
17. Hao H, Ting-Li J (2010) Floating bubbles. In: New frontiers: proceedings of the 15th CAADRIA conference. pp 175–183
18. Meyboom A, Reeves D (2013) Stigmergic Space. In: Adaptive Architecture: Proceedings of the 33rd ACADIA conference. pp 200–206
19. Bayraktar ME, Çağdaş G (2013) Fuzzy Layout Planner: A simple layout planning tool for early stages of design. In: Proceedings of the 31st eCAADe Conference. Delft, The Netherlands, pp 375–381
20. [omitted for blind review]
21. Fernando R (2014) Space Planning And Preliminary Design Using Artificial Life. In: Rethinking Comprehensive Design: Speculative Counterculture: Proceedings of the 19th CAADRIA conference. Kyoto
22. Christensen JT (2014) The generation of possible space layouts. In: Fusion: Data integration at its best.: Proceedings of eCAADe conference. eCAADe, pp 239–246
23. Puusepp R (2014) Agent-based Models for Computing Circulation. In: Design Agency: Proceedings of the 34th ACADIA conference. pp 43–52
24. Puusepp R (2014) Spatial Agglomerates. In: Rethinking Comprehensive Design: Speculative Counterculture: Proceedings of the 19th CAADRIA conference. pp 585–594
25. Ireland T (2015) A cell inspired model of configuration. In: Computational Ecologies: Design in the Anthropocene: Proceedings of 35th ACADIA conference. Association for Computer Aided Design in Architecture (ACADIA).
26. Ireland T (2015) An Artificial Life Approach to Configuring Architectural Space. Real Time: Proceedings of 33rd eCAADe conference, Wien
27. Bazalo F, Moleta TJ (2015) Responsive Algorithms. In: Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia. pp 209–218
28. Derix C, Izaki A (2013) Spatial computing for the new organic. Archit Des 83:42–47
29. Abu Dhabi Education Council | Office & Workplace | AHR | Architects and Building Consultants. http://www.ahr-global.com/Abu-Dhabi-Education-Council. Accessed 10 Nov 2017