Author(s): Coates, Paul S.; Derix, Christian; Lau, T.; Parvin, T.; Puusepp, R.
Title: Topological Approximations for Spatial Representations
Year of publication: 2005
Citation: Coates, P.S. et al. (2005) 'Topological Approximations for Spatial Representations', *Proceedings of the 8th Generative Art Conference (GA2005)*, Milan: Generative Design Lab Milan Polytechnic University, Italy.
Publisher link: http://www.generativeart.com

Topological Approximations for Spatial Representations

P. Coates AA Dipl, C. Derix MSc DiplArch, T. Lau MSc BScArch,
T. Parvin MSc DiplArch and R. Puusepp, MSc MArch
Centre for Evolutionary Computing in Architecture
School of Architecture and Visual Arts, University of East London, London, UK
e-mail: p.s.coates@uel.ac.uk + c.derix@uel.ac.uk

Marshall McLuhan once said in his book Understanding Media that *'Environments are invisible. Their ground rules evade easy perception.'* Evasive perception leads to fuzzy representations as shown through Kevin Lynch's mental maps and the Situationists' psycho-geographies. Eventually, spatial representations have to be described through abstractions based on some embedded rules of environmental interaction. These rules and methods of abstraction serve to understand cognition of space.

The Centre for Evolutionary Computing in Architecture (CECA) at the University of East London has focused for the last 5 years on methods of cognitive spatial descriptions, based largely on either behavioural patterns (i.e., Miranda 2000 or Raafat 2004) or topological machines (i.e., Derix 2001, Ireland 2002 or Benoudjit 2004). The former being agent based, the latter (neural) network based.
This year's selection of student work constitutes a combination of cognitive agents + perceptive networks, and comprises three theses:

Tahmina Parvin (section 1, *p.2*): an attempt to apply a **Growing Neural Gas** algorithm (GNG) - an extension of the traditional Kohonen SOM (self-organizing feature map) - to spatial representation. This follows as a direct consequence from previous work (Derix 2001 and Benoudjit 2004) where the rigidity of the simple SOM topology during learning became problematic for representation.

Renee Puusepp (section 2, *p.10*): an experiment to create a genuinely intelligent agent to **synthesize perception**. This work also uses the connectionist model as a basis for weighted learning. Agents learn to direct themselves through environments by distinguishing features over time using a type of weighted behavioural memory.

Timothy Lau (section 3, *p.16*): a **route finding** method as outcome from an ant-colony algorithm constraint through way-finding operations and an environmentally generated topology. This work was stipulated by 'live' architectural briefs to solve health-care lay-outs.

All three projects are in development and promise to be explored further. Their relevance cannot be underestimated at a time when so called 'smart' technologies seem to have been exploited and the demand for self-regulating 'intelligent' media is growing. This means for architects that they will need to understand the occupants' perception and their behaviours in more depth by using such simulation methods like the ones presented below:

+ section 1 +

GROWING NEURAL GAS. SELF-ORGANIZING TOPOLOGIES FOR SPATIAL DESCRIPTION

Abstract

Space can cognise itself from elements of social system and can be self-generative at the same time. Architectural space can be visualized as an autonomous**,** adaptive and intelligent-complex system coupled with other autonomous complex system. Adaptive behaviour of autonomous artificial system of growing neural gas is exploited as a model of cognitive process for built environment. Cognition can be defined in terms of ability to respond to the environmental events, and the 'stimulus' is the part of environment that is absorbed by the structure of the model. Space can be expanded, deformed, fragmented, decayed and above all self-organized by learning from constituent of environment.
Programs or events can be constructed by combination of actions and can be represented as point intensity on probability space. Space is constructed by intentional actions of individuals. Growing neural gas can be viewed as a 'distributed representation technique for the spatial description' of space, can learn and adapt itself according to point intensity on space.
Different space – time conditions can create topological displacement and variation in spatial character.
In general, the discussion is about the morphology of growing neural gas as an artificial autopoietic system and how it can couple to relevant features of the environment, as a two-way learning mechanism. The model's synergetic construction of topology is a mental construction towards a theory of space.
Space can be viewed as open, complex and self-organizing system, which exhibits phenomena of non-linearity and phase-transition.

The notion of spatial cognition

Spatiality refers to mental perception of space, mapping environment and behaviour.

Spatial analysis is about the interrelationships between behaviour and properties of the man-made environment. It is about the techniques that objectively describe environments. This description can be related to specific local level phenomena of human behaviour such as vision, movement, perception, action or particular global phenomena of complex system.

Cognition or 'knowing' is the perception of environmental change. It is the design process in nature and artificially intelligent system. Pointed out by Herbert Simon, *any entity, be it natural or artificial, that devises courses of action aimed at changing existing situations into preferred ones* (whatever they might be), *can be said to engage in design activity* [Simon, 1981].

Cognition can be modelled as the embodied, evolving, and interaction of a self-organizing system with its environment as a design process in nature. Time-dependent cognition is an essential component of life or artificial life. All living systems or artificially living systems are cognitive systems. Cognition is simply the process of maintaining itself by acting in the environment.
Maturana and Varela define 'cognition' in terms of basic ability to respond to environmental events. Design is a cognitive activity that all humans, other living beings, and the intelligent machines, are engaged in. The nature of design activity is highly diversified, as the underlying cognitive processes involved in different domains are fundamentally different.
Space is a container, and action is what we do within it. A kind of natural geometry is generated by actions and movements on space. As an example, a group collectively define a space in which people can see each other, mathematician defines it as 'convexity of space', and represents it by points on space.
Formal and spatial aspects of architectural and urban design are known as 'configurational' research techniques, which can easily be turned round and can be used to support experimentation and simulation in design. This way is one kind of attempt to subject 'the pattern aspect of elements in architecture' to rational analysis, and to test the analysis by embodying them in real designs.
Spatial scales can be local and global. The local scale deals with immediate neighbours and global scale deals with overall spatial configuration.

Graph-based topological model in archiectural spatial analysis

Graph-based operationalizations of space, which are used for mental representations of environment in the field of architecture and cognitive science, describe environment by means of nodes and edges, roughly represent to spaces and their spatial relations. Edges implement relationship between nodes. Graph-based model is flexible, extendable, and its' generic framework with straightforward algorithms help to solve specific higher-level questions regarding architectural spatial issues.

Mental representations of space cannot be seen independently from the formal and configerational properties of the corresponding environments. At the same time, formal description of a space as used in architecture gain rationality by incorporating results from cognitive research, which give the prediction and explanation of actual behaviour of the system. Graph- based diagrammatic analysis deals with theory of space gives quantitative expression to 'elusive pattern aspect' of architectural and urban design. It is a 'neutral techniques' for the description and analysis of the 'non-discursive' aspects of space and form. Graph based models are used for quantitative comparisons between spatial configurations in order to identify the essential properties in terms of function or usage. It is considered as strictly formalized description system of space. Beside applied research, graph investigations in architecture particularly concentrated on methodological issues such as the analysis techniques on arbitrarily shaped environments on variable scale levels and on the formalization and automation of the graph generation process within certain complex spatial system.Turner, Dexa, O'Sullivan, & Penn (2001) have proposed visibility graphs in different way to optimise the computational graph analysis. Visibility graphs replace the isovist as node content by inter-visibility information translated into edges to other nodes. Visibility graphs are useful to predict spatial behaviour and affective qualities of indoor spaces.

Neural networks and graph-based topological representation

Knowledge is acquired by the network from its environment through a learning process and Interneuron connection strengths are used to represent the acquired knowledge. The every single unit could be the metaphoric representation of an individual entity and lines could be communication links. A neural network can be represented as a graph consisting of nodes with interconnecting synaptic and activation links.

Self-organizing neural network: Growing Neural Gas

Self –organizing neural network models were first proposed by Willshaw & vonder Mlalsburg (1976) and Kohonen (1982). The networks generate mappings from high-dimensional signal spaces to lower dimensional topological structures.
The "growing neural gas" algorithm of Martinetz & Schulten (1994) seems to produce compact networks, which preserve the neighbourhood relations extremely well. The network has a flexible as well as compact structure, a variable number of elements, and a K-dimensional topology. The algorithm is considered as spetio-temporal Event Mapping (STEM) architecture. It can be viewed as decentralized model of human brain learning process or decentralized model of cognition.
Martinetz (1994) introduced a class of Topology Representing Networks (TRN), which belongs to **Dynamic Cell Structures** (DCS), build perfectly topology preserving feature map. DCS idea applied to the **Growing Cell Structure** algorithm (Fritzke 93) is a more efficient and elegant algorithm than conventional models on similar task. **Growing Neural Gas** (GNG) is a class of Growing Cell Structure. It differs from Growing Cell Structure through its less rigid topological definition.
GNG only uses the parameters that are constant in time. It employs a modified Kohonen Learning rule (unsupervised) combine with competitive Hebbian learning. The kohonen type learning rule serves to adjust the synaptic weight vectors and Hebbian learning establishes a dynamic lateral connection structure between the units. The new model of growing neural gas can be said the extension of Kohonen's feature map with respect to various important criteria (Fritzke, 1993a).

Two basic differences with Kohonen SOM:

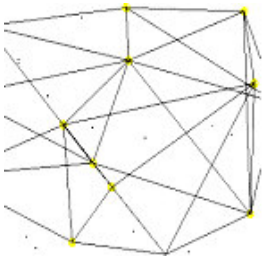
1 The adaptation strength is constant over time. Specifically used constant adaptation parameters e_b and e_n for the best matching unit and the neighbouring cells, respectively.
2 Only the best-matching unit and its direct topological neighbours are adapted.

Neural gas as a connectionist model (Donald Hebb 1949)

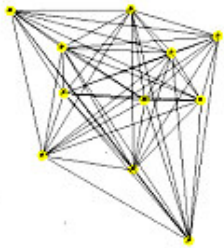
Brain operation can be modelled by means of a network of interconnected nodes. Each Node takes the sum of its inputs and generates an output. The output is determined by the transfer function of the node, which has to be non-linear. The connection (synapse) between any two nodes has certain 'weight'. The information flows only in one direction. Complex behaviour emerges from the interaction between many simple processors that respond in a non-linear way to local information.

The characters of connectionist models are:
1 High level of interconnectivity
2 Recurrence
3 Distributedness (described algorithmically)

sparsely connected



fully connected



richly connected

GNG connectionist model consists of a large number of units, richly interconnected with feedback loops, but responding only to local information.

The process of autopoiesis

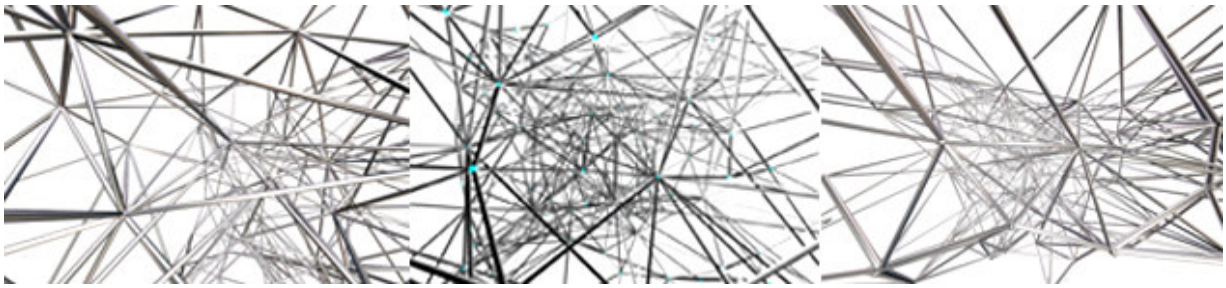
An answer lies in the pattern characteristic of all living systems is autopoiesis, a network pattern in which each node is a production process that produces or transforms other nodes. The entire network continually produces itself. The pattern & process of interaction are called autopoiesis & cognition. Thus, autopoiesis is the pattern by which life emerges in dissipative systems and cognition is the very process of life itself.

Virtually autopoietic system can be understood in terms of networks or systems with very simple rules of interaction. These simple rules can lead towards immensely complex sequences. Networks are systems composed of interconnected parts. Relationships between parts are circular (closed loop). This causes feedback because the actions of each component "feeds back" on itself. Both positive and negative feedback is possible.

Autopoiesis generates a structural coupling with the environment: the structure of the artificial system generates patterns of activity that are triggered by perturbations from the environment and that contribute to the continuing autopoiesis of the system. Maturana argues that the relation with the environment moulds the "configuration" of a cognitive system. Autopoiesis is the process by which an organism continuously re-organizes its own structure. Adaptation consists in regenerating the organism's structure so that its relationship to the environment remains constant. An organism is therefore a structure capable to respond to the environment. Autopoiesis is a self-generating, self-bounded and self-perpetuating process.

Growing neural gas as a autopoietic representation of space

Growing Neural Gas can be an advance tool for visualization of emergent spatio-temporal correlations between different entities on space. The point intensity of signal space can represent activity intensity on a three dimensional space. More program intensity will generate more signals on space, which means more stimuli to enhance the system operation.

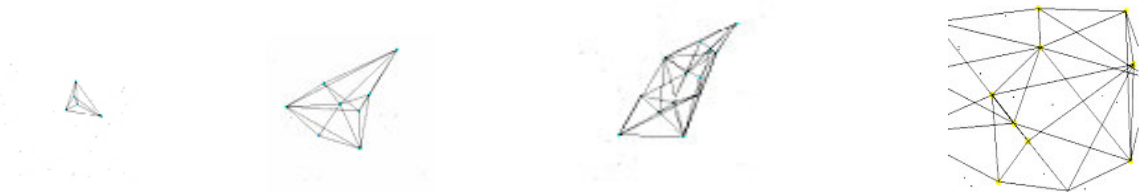


Self-organizing growing neural gas

Model description

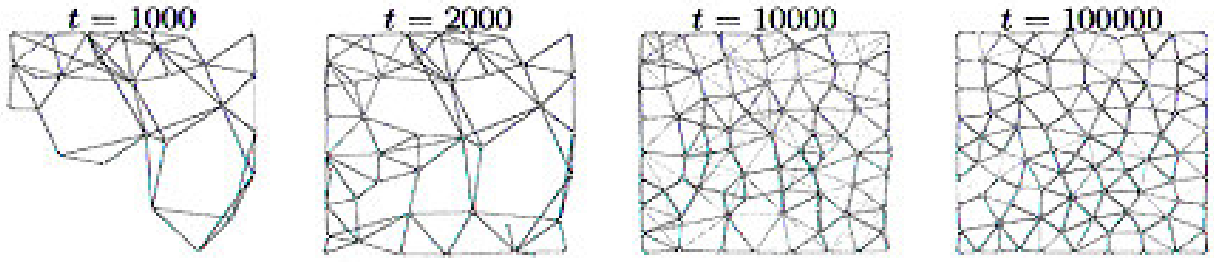
Current work

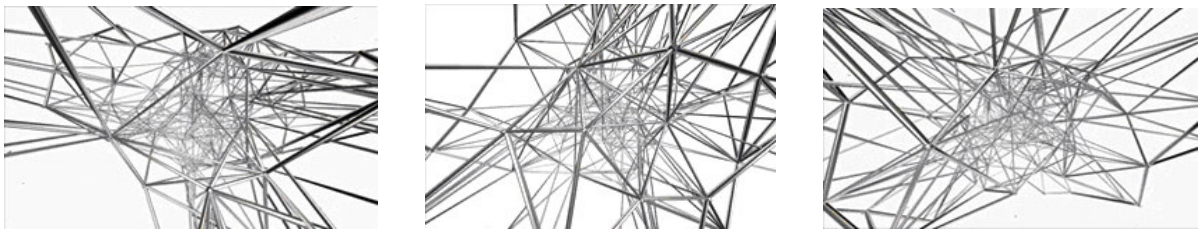
Initially, the model is an unsupervised self-organizing growing neural gas algorithm. It has some randomly moving points on three dimensional spaces, which are denoting the signal intensity on the space. The signals movements are based on Brownian motion.
The network start with more than three units (k > 3), that means with connecting edges it forms a hyper tetrahedron, which ultimately lead towards k-dimensional simplex.



k-dimensional simplex

At first the system shows a chaotic way of growth and decay and topological displacement. But after a certain period of time, after it has reached to the maximum growth level, self-organized itself within the three-dimensional signal-space. It gives a topology preserving networks of nodes. In this self-organization, every unit deals with its compact perceptive field with its' nearest neighbours, which ultimately gave a compact volumetric topological expression of the space.



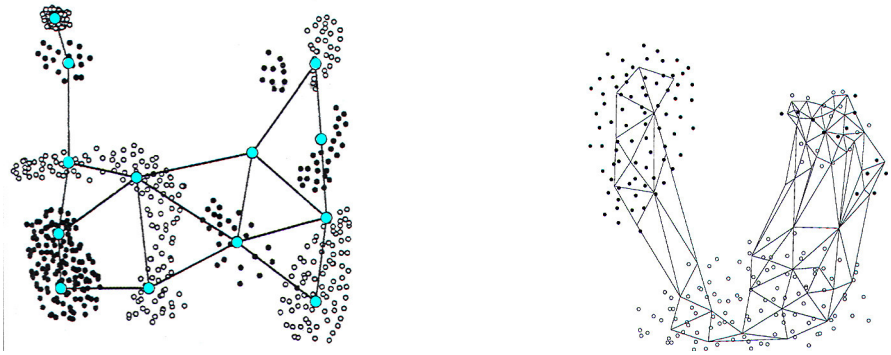


GNG Topologies after thousands iterations

Projected work

Further work will combine the GNG with a supervised Radial Basis function (RBF) which will give more potential flexibility to the model and to the clustering of elements.

Otherwise, the Ant-based clustering algorithm and growing neural gas networks can be combined to produce an unsupervised classification algorithm that can exploit the strengths of both techniques and can be more flexible to describe autopoietic space. The ant-based clustering algorithm detects existing classes on a training data set, and at the same time, trains growing neural gas networks.



GNG with Radial Basis Function

In later stages, these networks are used to classify previously unseen input vectors into the classes detected by the ant-based algorithm. The advantage of this model above GNG model is that the dimensions of the environment don't need to be changed when dealing with large databases.

CONCLUSION

In this discussion, the goal was to establish self generating growing neural gas as an tool to visualize autopoietic complexity of architectural and urban space. The model is exceptional for its' growth process through cognition, the occasional death of units and self-generation; all of which belong to the core concept of life. The initial model is a simplest form of growing neural gas model and an abstract representation of design space.

REFERENCES

[1] Coates P., Derix C., Appels T. and Simon C, *Dust, Plates & Blobs,* Centre for Environment and Computing in Architecture CECA, Generative Art 2001 Conference, Milan 2001

[2] Derix C., *Building a Synthetic Cognizer*, Design Computational Cognition 2004 Conference, MIT, Boston, USA, 2004

[3] Cilliers, Paul. 'Complexity and Postmodernism, Understanding Complex systems.'

[4] Bill Hillier, 'Space is the machine', Cambridge University Press, Cambridge, 1996

[5] Luhmann, N. ' The Autopoiesis of social system', 1972

[6] Maturana H. & Varela, F. (1980) *Autopoesis and cognition: The realization of the living*.

[7] Yehuda E. Kaley, Architectures New Media. Principle, Theory, and Methods of Computer-Aided Design

[8] Lionel March and Philip Steadman, 'The geometry of environment'. An introduction to spatial organization in design, RIBA, 1971

[9] Teklenburg, Jan A.F., Zacharias, John, Heitor, Teresa V. 'Spatial analysis in environment-behaviour studies: Topics and trends', 1996

[10] Turner A., O'Sullivan D., Penn A., Doxa M, 'From isovists to visibility graphs: a methodology for the analysis of architectural space'.

[11] Gerald Franz, Hanspeter A. Mallot, & Jan M. Wiener. 'Graph-based models of space in architecture and cognitive science-a Comparative analyses'.

[12] Peponis, John, Karadima Chryssoula, Bafna Sonit. 'On the formulation of spatial meaning in architectural design'.

[13] Fritzke, B " Unsupervised Ontogenic Network", Handbook of neural Computation, 1997

[14] Gerald Tesauro, David Touretzky, Todd Leen. 'Advances in neural information processing system'. MIT press,Vol. 7, 1995

[15] Andrew, Adamatzky. 'Computing in Nonlinear Media and Automata Collectives'. 2001

[16] Stephen Judd, 'Neural Network Design and the Complexity of Learning.' MIT Press, 1990

[17] Fritzke, B. 'Fast learning with incremental RBF networks'. *Neural Processing Letters,* 1994

[18] Fritzke, B. A growing neural gas network learns topologies. 1995

[19] Fritzke, B. Growing Cell Structures – A self-organizing Network for Unsupervised and Supervised Learning, 1993

[20] Gerald Tesauro, David Touretzky, Todd Leen. ' A growing neural gas network learns topologies'. 'Advances in neural information processing system'. MIT press,Vol. 7, 1995

[21] Martinetz and Schulten, A "neural-gas" network learns topologies, 1991

[22] Martinetz and Schulten, 'Topology representing networks', 1994

[23] Jim Holmtrom,'Growing Neural Gas'. Experiments with GNG, GNG with Utility and Supervised GNG

[24] J. Handl, J. Knowles and M. Dorigo, 'Ant-based clustering: a comparative study of its relative performance with respect to k-means, average link and 1d-som'.

[25] Conroy, Ruth. 'Spatial Intelligibility & Design of VEs'

[26] Toussaint, Marc. ' Learning maps for planning'

[27] X. Cao, P.N. Suganthan, Video shot motion characterization based on hierarchical overlapped growing neural gas networks

[28] Randall D. Beer, 'Autopoiesis and Cognition in The Game of Life'.

[29] Gershenson, Carlos. Francis Heylighen.'How can we think the complex?'

[30] Wendy X. Wu , Werner Dubitzky. 'Discovering Relevant Knowledge for Clustering Through Incremental Growing Cell Structures'.

[31] K.A.J. Doherty, R.g. Adams, N. Davey. Hierarchical Growing Neural Gas

[32] http://www.neuroinformatik.ruhr-uni-bochum.de_
/ini/VDM/research/gsn/DemoGNG/GNG.html

+ section 2 +

Synthetic perception. Processing spatial environments

Abstract

The way we perceive the environment around us is inseparable from our decision-making and spatial behaviour. Learning to find one's way around in a foreign city does not rely only on one's cognitive skills, but it is also dependent on the clarity of urban layout. Computational models have proven to be highly useful in simulating various aspects of sensory-motor conduct, and help to explore the fundamental principles of environmental behaviour. The proposed paper employs algorithms to assess intelligibility of spatial arrangements, and suitability of these algorithms to elucidate the processes of cognitive mapping. During the exploration of digital models, semi-autonomous agents develop a set of formal rules to interact with the environment.

Introduction

Architectural space is full of cues. These cues, either intentionally designed or randomly established, determine the way people use the space. If features of space happen to give misguiding signals, space becomes inefficiently used. One can argue that environment has some sort of intelligence that communicates with its inhabitants; one can even claim that people are capable of magnifying this intelligence. There can be a certain kind of embedded intelligence in the environment – an intelligence that makes the environment literally speak or automatically adjust itself. However, speaking of traditional urban environment, there can be only embedded intelligibility. The intelligibility of space determines the capability of being understood by an intelligent inhabitant. In other words, the space does not contain any self-explanatory information *per se*; it is the observer who carries the meaning with him. From that point of view, the space is unconscious medium like a piece of paper. Merleau-Pointy [1] sees intelligible space as an explicit expression of oriented space that looses it meaning without the latter.

Spatial behaviour is very much affected by one's knowledge of the space and vice versa. The perception of space is never static but evolves over time. Perception is locked in a reciprocal dialogue with the movement of the inhabitant. The way the environment coerces us to move, can be more or less in conflict with our natural movement, which is defined by the proportions of our body and by the speed of the locomotion.

An organism's ability to learn is strongly related to its position in the evolutional timeline. Higher organisms have less intrinsic behavioural patterns, and depend more on their experience. Environmental learning involves obtaining appropriate sensory-motor conduct, and storing spatial information for future use. In literature, such stored environmental data structures have been referred as cognitive maps [2] and environmental images [3].

This paper considers cognitive mapping – also known as mental mapping – as a method to collect, restore and retrieve environmental information. Maps are representations of the environment, acquired through direct or mediated perception. Psychologists and geographers have used the term 'cognitive map' for decades in context of way-finding and environmental

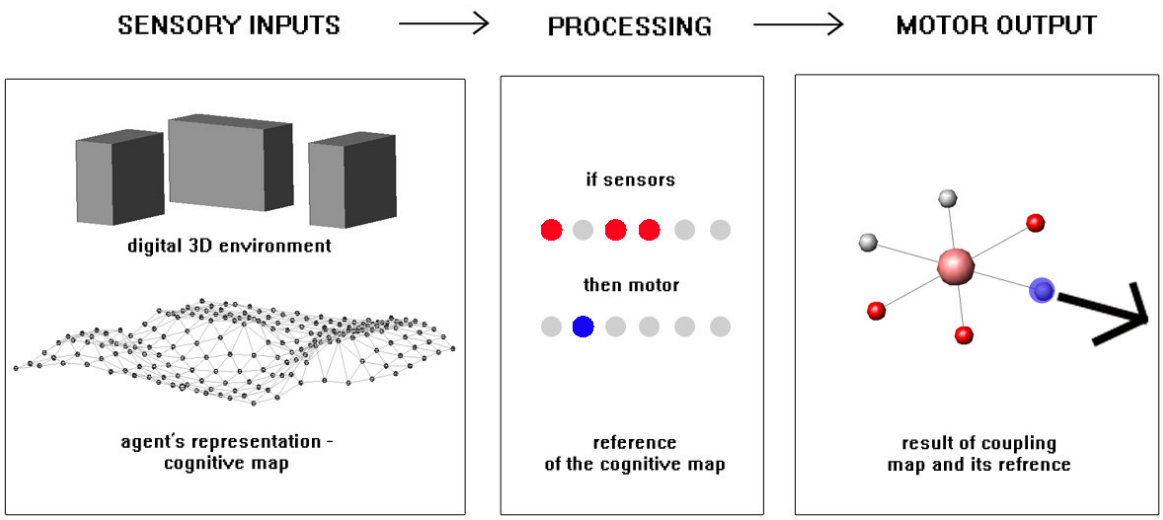
behaviour [4].
The cognitive map possesses a kind of continuity; a certain procedural or algorithmic sequence that we follow in order to get from one point to another. The map is more a routine-like description, a behavioural description of an environment rather than a structural description of surroundings. From architects' and city planners' point of view, it is worth studying the process of cognitive mapping and its major shapers, because it could reveal the fundamental principles behind environmental decisions.

Artificial perception of space

This paper explores a contingent way to approach processes of cognitive mapping using computational methods. It is impossible to explain the full range of natural cognitive mapping with computational models, but these models still provide some helpful insights. The usefulness of these maps is best displayed in unravelling wayfinding difficulties within multi-agent environment. The proposed algorithm tries to exhibit essential parts of cognitive mappings such as collecting, storing, arranging, editing and retrieving environmental data.

Flow of the algorithm

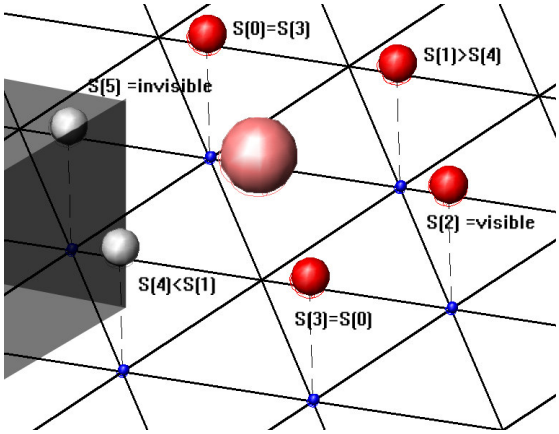
The algorithm under review benefits from several well-known concepts. The goal of the agents is to find a way to an assigned target point while interpreting sensory input and leaving some trails into the environment. Successful agents are awarded by upgrading their value; the value of a non-successful agent is reduced to minimum.
Agents are using a combination of computational *cognitive map*, which is agent's representation of the environment, and syntactic instructions that determine how that map is interpreted. An agent is 'born' *tabula rasa* – the evolution of the map and the reference to it happens during its interaction with the environment. Conception and fade-out of cognitive data is simulated using pheromone trail algorithm – last visited spots contain more information than formerly visited ones. Syntactic instructions have to develop and change according to this dynamic environment. The pursuit of targets is facilitated by trivial vision. If the visibility line between the agent and its target is clear, the agent takes an automatic step towards the goal.
Besides individual learning, the development of an agents' syntactical instructions also takes place at a phylogenetic level. Evolution of agents is similar to evolution of Braitenberg's vehicle type 6 [5]. A single agent is chosen for reproduction. However, in the process of reproduction only 75% of syntactical instructions of the agent have been transmitted to its *offspring*. The selection of the *parent* agent is partly up to chance – the best (by value – see above) of a subset of randomly selected individuals gets the honour to be copied.



1. Input-output coupling. Agent obtains input from digital environment and from the *cognitive map*. Output is generated interpreting input according to syntactical rules (reference of the *cognitive map*).

Design of the agent

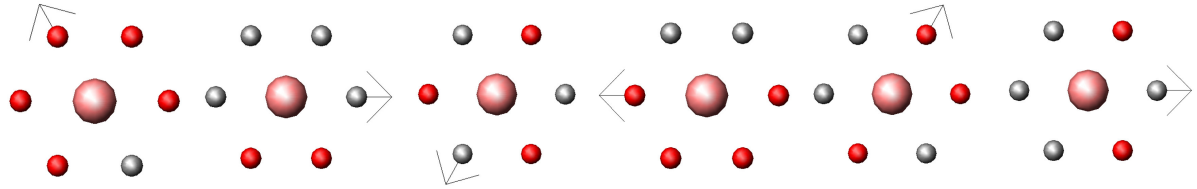
The design of the agent is fairly simple. The visible entity of the agent consists of the central 'body' and six sensors attached to it to form three symmetry axes. The consideration behind the hexagonal design was to give agents sufficient liberty of motion retaining the symmetry and thus leaving undefined the *front* and the *back*. Sensors are combined into three identical axes which have a major influence over the activation function (see image no 2). Each sensor-axis has four possible states: two polarized states (only one sensor active), both sensors active, and both sensors passive. All three sensor-axes together yield to a sensory space with 64 possible input combinations. Sensor morphology is invariant, which means that the body plan of the agent is not capable of evolving over time.



2. Sensory input. Sensors acquire their value from the closest nodes. The value is then compared to the value of diametrically opposite sensor. Red sensors are activated.

The output space, or the motor space, is much smaller containing only six possible directions. The agent can take only one step at a time, and cannot combine different directions. Although the motor space is relatively small, the mapping of inputs to outputs results in 384 different combinations. The agent's behaviour is simply controlled by a list of input-output matches, and is completely undefined when the agent comes into being. The agent has to create the controller itself by interacting with the digital environment constantly comparing the input-output mapping (henceforth: *syntactic memory*) to environmental affordances. The *syntactic*

memory consists of two layers: long-term and short-term *memory*, where the latter acts as the verification layer to buffer to the long-term *memory*. In addition to the controller, the agent possesses some persistence in its action – if the sensory input appears to be unfamiliar, the agent continues in the previously chosen direction.



3. Syntactic instructions. Agents gain different syntactical instructions to map inputs to output. 75% of these instructions are being passed to 'offspring' to maintain diversity and explorative ability.

Design of the environment

The term 'environment' obtains ambiguous meaning within the context of digital computers. In most cases, digital environments are just representations, and therefore leave some room for human interpretation. Although, in order for true intelligence to emerge, people in the field of embodied cognitive science strictly insist on using the world as its own model [6]. The complexity of cognitive mapping, however, encourages exploitation of virtual worlds.

This paper addresses the digital environment from the agents' point of view, thus omitting the observer's perspective. The three-dimensional data ceases to be a representation for agents, and becomes adequate *environment* for them. Agents, in turn, create a representation of this *environment*, which can be interpreted as a *cognitive map*. The *map* is laid over the *environment* only for observational considerations, but it can actually be an integral part of the agent.

The *cognitive map* is a hexagonal network of interconnected nodes. Each node contains one or more values (or *vectors*). If stepped on by an agent, these vectors are adjusted according to the success of the agent. A node is just a passive piece of information – agents gain meaningful knowledge by comparing adjacent nodes. Nodes also have a tendency to forget their values gradually causing the information on the *cognitive map* to fade away.

Observational notes

The progress in agents' behaviour and development of the *syntactical memory* and the *cognitive map* tends to follow a standard pattern. As the first meandering agent finds a way to its target, all nodes of the *cognitive map* the agent has stepped on are positively adjusted. When the agent tries to repeat the same route, it may turn out that instructions in the *syntactic memory* do not match the modified *cognitive map*. Thus, new instructions have to be invented to 'read' the trail left behind by the first agent. If the next agent is capable of finding the target in slightly changed situation (with positively adjusted nodes), it reinforces the trail on the *cognitive map*, and appropriate reference have been generated to interpret this map.

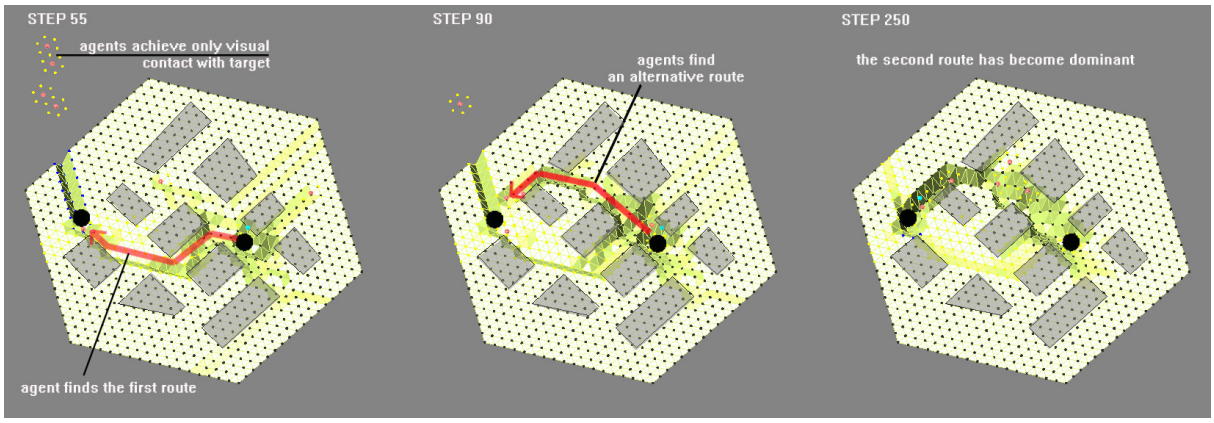
Certain features of the *environment* facilitate wayfinding, especially those that suit the agents' 'anatomy' and gestural traits most. It is not always the shortest route that becomes most popular – it is usually the most suitable route for particular kind of agents.

Few interesting behavioural phenomena can be pointed out:

a) Agents acquire different techniques to move around in the *environment*. Some of them try to keep away from environmental obstacles, others, in contrast, obtain 'wall-following' tactic.

b) Some agents tend to travel to locations where they have clear visual contact with their target, without actually getting closer to the target.

c) If a route to the target has been found, some agents still keep exploring and finding other ways than the established one. The firstly discovered route does not necessarily become the most used one.
d) Unplanned competition between different 'species' occasionally takes place. The nature of pheromone trail algorithm prevents agents to use the same trail in both directions. One way 'traffic' tends to force the other way out.



4. In search for targets. Environmental features have crucial role in competition between popular routes. It is not always the shortest route that is preferred by agents.

Conclusion

The results of the undertaken computational experiment are not comparable to behaviour of biological organisms and organizations, which are vastly more complex and beyond the reach of the present work. However, even simplified and abstract simulations can guide us to towards a deeper understanding of some mechanisms behind complex systems. In that light, simulated artificial agents become a simplistic synthetic species.
Synthetic perception of architectural space is potentially a powerful concept. Besides the speed and the accuracy, it could help us explicitly define essential design parameters, and critically assess our own methods of work. Although computational models are far from displaying similar performance to human perception, we can learn a lot about ourselves by simulating some aspects of spatial behaviour. Computational experiments could reveal some fundamental rules of interaction between inhabitants and their environment, which are likely to hold true also in more complex context. As Kelly [7] points out, complexity has to be build bottom-up using simple comprehensible units, rather than deconstructing entities that perhaps have evolved over million of years.
Perception is a prerequisite of any intelligent action. If an architect desires to employ a semi-autonomous agent to automate some design tasks, artificial perceptual mechanisms obtain crucial importance. The agent has no use in generating spatial organizations if it is not capable of reacting to spatial stimuli

Possible alterations and future orientation:
· Evolution of agents' sensor morphology and body plan
· Agents with neural network controllers to 'learn' intricate concepts
· Agents generating spatial constructions

References

[1] Merleau-Pointy, M. 1979. Phenomenology of Perception. Suffolk: St Edmundsbury Press Ltd

[2] Downs, R.M., Stea, D. 1977. Maps in Minds: Reflections of Cognitive Mapping. New York: Harper & Row

[3] Lynch, K. 1960. The Image of the City. Massachusetts: MIT Press

[4] Stea, D. 1974. Architecture in the Head: Cognitive Mapping. In: J. Lang, ed. Designing for Human Behavior. Stroudsburg, Dowden, Hutchinson & Ross, Inc. 157-168.

[5] Pfeifer, R., Scheier, C. 2001. Understanding Intelligence. Massachusetts: MIT Press

[6] Brooks, R.A. 1987. Intelligence without Representation [online]. Cambridge (USA), MIT Artificial Intelligence Laboratory. Available from: people.csail.mit.edu/brooks/papers/representation.pdf [Accessed 17 October 2005]

[7] Kelly, K.1994. Out of Control: The Biology of Machines. London: Fourth Estate

+ section 2 +

Route finding for the Built Environment

Abstract

The task of designing a well-planned group of spaces, still possess a fundamental problem for designers of the built environment. In our cities and buildings, the chaos of multiply users and their routes through the built environment seems to be getting worse, not better. Much of the problems with poor route planning logically fall on the heads of those who design and allocate the uses for each of the spaces in the first place. But how much of the blame can be put upon the designers? Many such problems only become apparent when the building are built and put into use. How can we predict when and where the interactions of users moving through the proposed spaces will result in congestion and problems, especially when the spaces that we can create become more and more complex?

Introduction

There has been a great deal of research into wayfinding and perception of the built environment done in the past 10 years. But, as Ruth Dalton observes in the second chapter of her paper on such studies…

"There has been little research undertaken into direct relation wayfinding performance back to the design of the Environment."[1]

Due to the heavy amount of information that has to be embedded into the plans, or the crippling computational expense for programs of this nature, most wayfinding programs are tested after plans and spaces have been finalised. This means that the results of the experiments have little or no effect on the spaces tested. If a tool can be developed that doesn't require the user to spend larger amounts of time embedding information into the plan, and has agents that use little computational processing power, then I believe that testing can happen much earlier on in the design process of the building. Proposed plans could be tested for the efficiency of important routes, adjusted according to the issues that the tool highlighted and then tested again.

How do I plan to achieve effective results whilst avoiding the same problems and pitfalls that are common to similar programs? I hope that, by drawing on emergent phenomena found in nature, I can find a system that gives me solid and replicable results, whilst minimising computational costs. By the term *emergent*, I patterns and outcomes that happen without any apparent cause or leader (One classic example of decentralized, emergent phenomena would be the way that flocks of birds file together in unison. Until the last 15 yrs, the accepted explanation to this was that there was some kind of leader bird in the flock. Now, however, this approach has given way to a decentralised explanation that is based on interactions between individual birds within the flock, Resnick M. 1994).

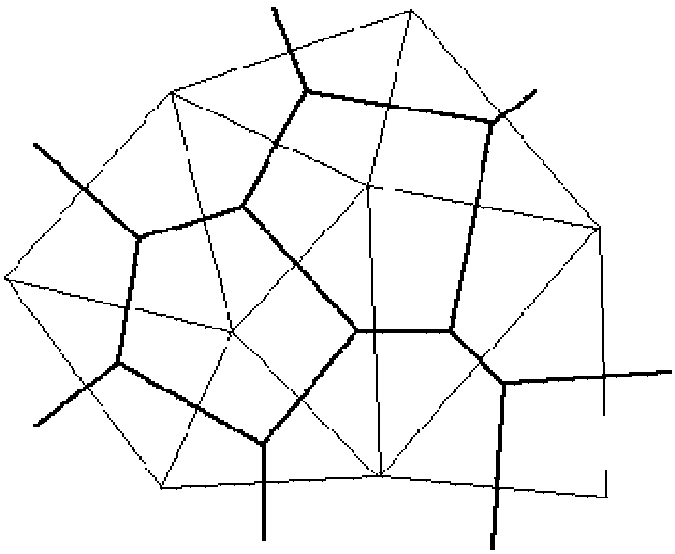
Flock of birds in flight

Plan recognition

First, I need to find a way of forming a grid of patches that can hold its location, and whether or not it was a walkable or non-walkable surface (i.e. a wall or part of an open space). My solution stems from my observations of bubbles. A soap bubbles film is a perfect example of a minimum spanning surface. An important characteristic of a minimal spanning surface is that they are

"Surfaces of minimal surface area for given boundary conditions" [2]

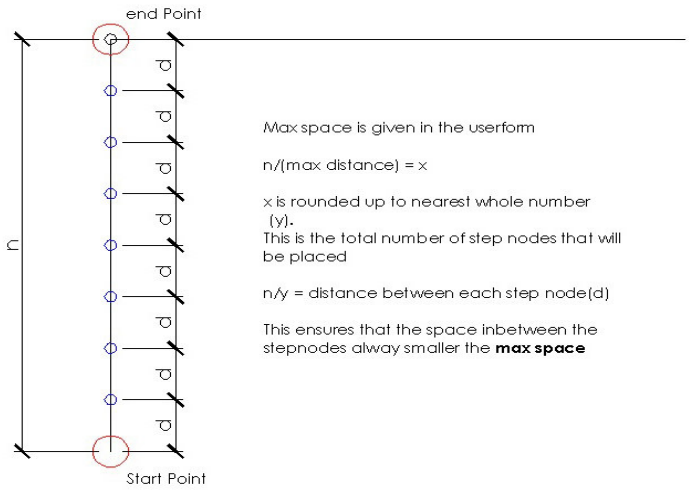
In general terms we can say that the molecules in the bubble skin are evenly spaced throughout the film. This would make logical sense, as the solution is the same throughout, therefore the modules and bonds would also be the same. Each modulus could be seen as seeking an equal distance with its neighbouring molecules. These properties are seen in the tessellations found in Voronoi fields.



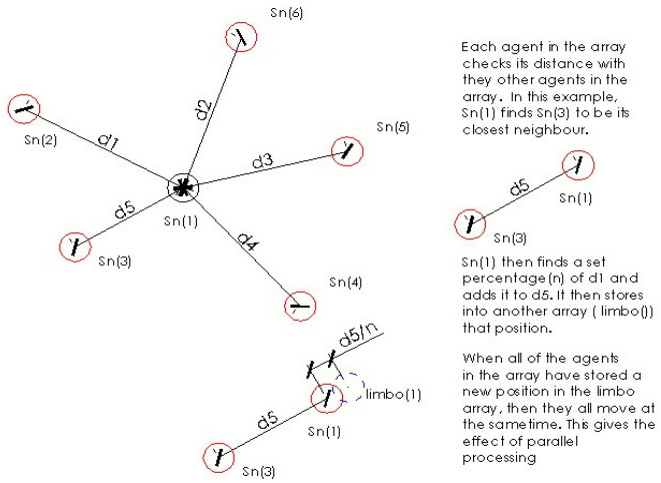
a Voronoi field

With an array of agents that have the similar properties to soap film molecules, I hope to find a quick and effective way of creating a grid, or *field,* of patches. By having either wall properties (places that can not be walk on or through) and space properties (places where one can walk), these agents will form the stepping stone for the next array of agents that will move through the spaces.

The key part of this section of the program lies in the agents creating the even tessellation throughout the plan. This solution to this problem turned out to be theoretically, very simple. After marking out the walls in the plan and dropping groups of agents within the spaces each agent starts by takes a step of a random length in a random direction.



From there, they have a simple goal: to be of equal distance with his neighbours. To get the agents to spread through the plan, they are repelled from one another and the wall-nodes. To break down the processes even more, each agent need only be concerned with its closest neighbour. When its closest neighbour is identified, it moves n% of the total distance between itself and that neighbour.



In order for this to work, each agent needs to perform the search and move at the same time. As true parallel processing can not be achieved on conventional, single processing computer, we mimic the parallel computation by get each agent in turn to find its close neighbour, and store its next move in a "limbo" array of positions. Only when all the agents have found their next position do the agents then move and update their positions visually. Over a period of time, we get the even tessellation of agents.

Route finding

Now that the program has "read" the plan, how does it move through the spaces? The idea of looking at the behaviour of insects that function in large colonies, such as ants, termites and bees, when tackling wayfinding problems, is something that has been widely researched and well developed.
The analogy in social insects to our "routing" problem, would be the way that some species of insects *self organise* to exploit a food source that is closest to their nest. As the authors of "Swarm Intelligence" point out,

"Many ant species have trail-laying trail-following behaviour when foraging" [3]

This process of trail-laying and trail-following can be distilled into a program. It stands to reason that if social insects can finding "better" routes to food sources in an uncalculatable variation of situations and terrains, a program based on these emergent phenomena could be just as effective and flexible.
The bases of an A.C.O (ant colony optimisation) algorithm is that it takes each ant as in individual entity. This entity has very simple properties, like the ability to move, lay a scent (pheromone) and also pick up a scent in its local environment. As the ants move away from their nest, they move in a more or less a random sweep until they find a food source. When they have found a food source, they starts dropping a scent and head back to the nest. Over time more ants pick up the scent and follow it thus reinforcing the scent. The evaporation of this scent is just as important to the algorithm as its reinforcement as explained by Bonabeau, Dorigo and Theraulaz.

"*In the field of ant colony optimisation and swam intelligence, common wisdom holds that pheromone evaporation, when occurring on a sufficiently short time scale allows ant colonies to avoid being trapped on a "suboptimal" solution, as is the case for Linepithema humile (a specie of ant found in south America).* ***This is because maintaining a well-marked pheromone trail is more difficult on a longer trail****"* [4]

A.C.O algorithms have been used to solve problems like

- *Quadratic Assignment Problems*, the problem of assigning a given number of facilities to a given number of locations so that the costs of the assignment are minimized
- *Job/shop scheduling Problem.* If there are a set number of machines, and a set number of jobs, how can operations be organised so that no two jobs are being processed by the same machine at the same time, thus minimising the time it takes for all the jobs to be completed?
- *Graph colouring problems.* What is the colouring scheme for a graph that needs the least number of colours?

There is also the TSP (travelling salesperson problem) that has become the benchmark for A.C.O algorithms. If a salesperson has n number of cites, what is the shortest route he can take to go through every city only once. The solutions to this problem have been used in telecommunication networks to great effect.
To read more about A.C.O (ant colony optimisation) algorithms and other such phenomena, please take a look at the excellent book, "swam intelligence" that I have referenced at the end of this paper

Here are the rules that I use in my program:

- the number of agents start at a minimum of 50 agents
- initially, our agents have to take random steps on any available step-points (all nodes that Indicate an open space)
- when an agent has found the goal (food source) it then beings its' way back to the nest.
- when the agent gets back to the nest, if it has made it back within a required numbered of steps, then the route is show and the step nodes that are in that route are weighted (this acts the same way as the pheromone scent)
- the required number of steps back to the nest from the goal will decrease with each successful return so that, overtime, the routes shown will tend towards the shortest.
- the weight of a node with a scent on it will decrease over time. This allows the program to get itself out of a "sub-optimal" route.

To make this process of wayfinding clearer, lets break the program down into steps. A start point, end point and any via points (points that have to be touched whilst going from start point to end point) are placed on the plan. Then the searcher agents follow these steps:

1 Finding their 5 closest neighbours.
2 Each step node has an agent-type value that distinguishes the nodes into wall nodes, step-nodes, and start point, goal point and via point. The searcher agent checks each one and all nodes, apart from the wall nodes, are collected into an array.
3 The array of eligible candidates for the searcher agents' next step is then effectively put onto a chance wheel. The different step-nodes could be weight differently (how the step-nodes are weighted will be discussed later), giving them a higher chance of being stepped on.
4 A random number is chosen and the step node with that number in its segment of the wheel is the winner. The agent moves to that step-nodes position.

When an agent has found its way to the goal it then starts to find its way back to the start point. From this point it keep track of how many steps it takes and the route it takes back. If it makes it back to the start point while also checking all the via points and doing so within a given number of steps, then the route it has taken is show on the screen.

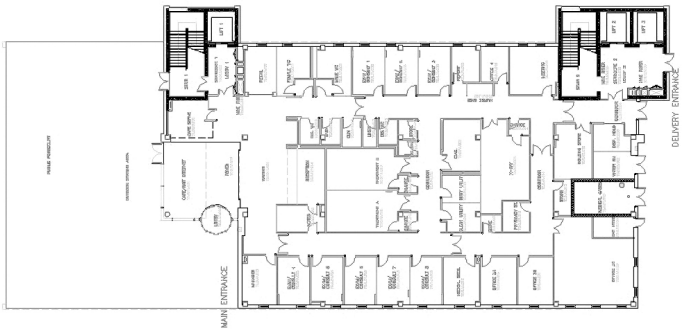
The step-nodes along the "winning" route are given a scent that gives them a heavier weight when they are placed in the chance wheel. (This acts as the pheromone scent*).* Each step-node also has a timer, and its weight will continue to decrease to zero after a while if it does not continue to be part of a winning route. (This forms the evaporation of the pheromones over

time). As the step counter decreases, each route that wins becomes shorter, and the route that the agents take between A to B also decreases. The weighting of the routes allows the chance of the agents to follow the shortest current route to increase whilst also allowing the agents to have a chance of shortening the route via altering the route with random steps.

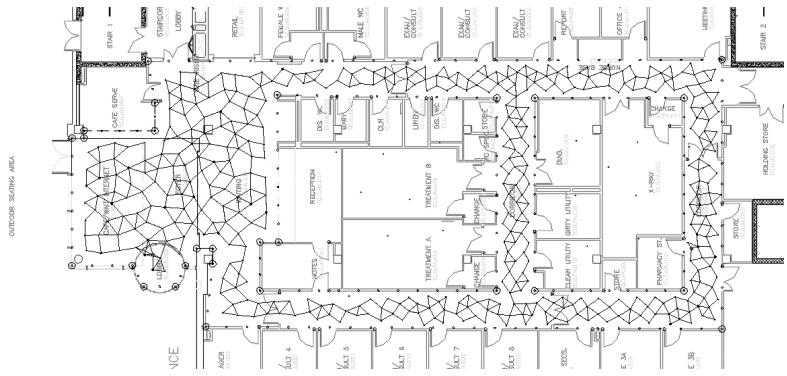
After one route has been successfully found, it can be saved. Any areas in the plan that the user would like to block out to prevent other agents moving through particular areas in the plan can be done so (this is useful if you want to give your first route sole access to a given corridor. The following routes found on the plan would then be found without interfering with this primary route). A new set of agents can then be let loose with a new starting point, end point and via points.

Test Cases

With the main components of the program in place, I thought that it would be a good time to test my program in what could be seen as a real project. An opportunity was opened to me to run my program on the plan of a real project that is being worked on by Aedas' research and development department. The project is to do with a new hospital in London. Here is an image of the ground floor of the hospital.

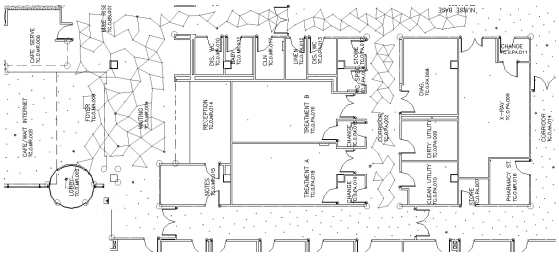


Firstly, we simplify the plan and place wall nodes to define the area that we want to test. We then seed the first array of agents that create the tessellation throughout the plan. To make sure that all the areas that I wanted to test can be reached, I ran my program, but turned the layer with the agents' paths on. This shows me if all the areas could be reached. With this done, I could start testing.

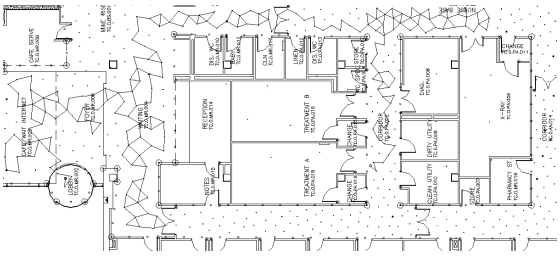


I decided to do a simple test that consists of choosing two destination points from the main foyer. The first route would be the primary route. I would then block out the corridors that the first route used and then run the second route twice: once without the blocked out corridor, and then again with the blocked out corridors.

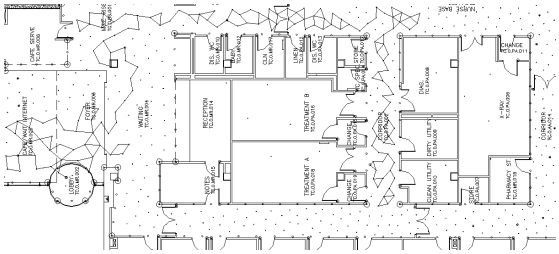
Firstly I asked the program to find a route from the main entrance to "Treatment Room A". Below are different routes the program took to get to its destination:



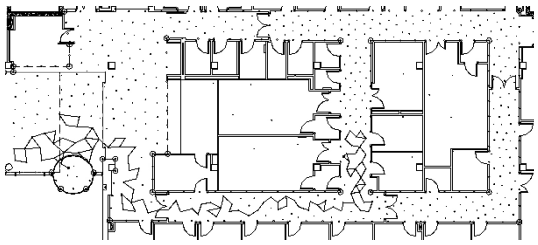
route A



route B

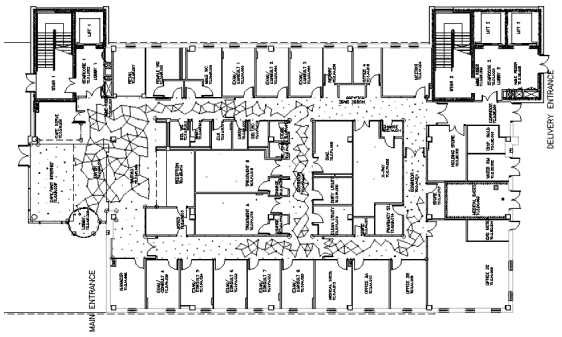


route C

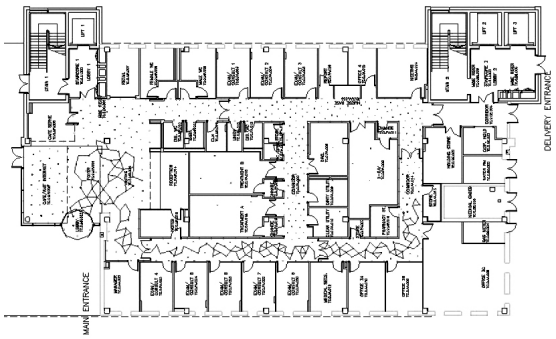


final route

In the next test, the program should find a route from Entrance to "Pharmacy store":



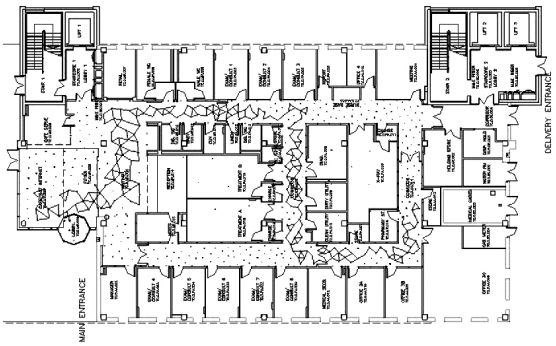
route *n*



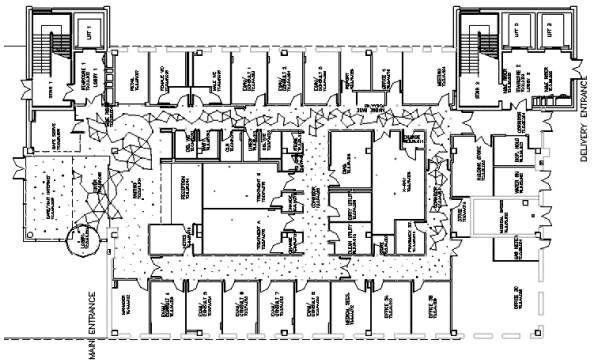
final route

Notice how the final route also uses the lower corridor to get to its destination.

Finally, same destination was chosen again, but this time the route along the lower corridor that the first test established was blocked off:



route *n*



final route

We can see that because the left half of the lower corridor is not available to it anymore, it changes its route to find the next best thing. Although this has been a simple experiment, we can see that the program can read the plan and find a good route between given points in the plan.

Conclusion

The progress that I have made in the short time that I have worked on this program has been promising. I have been able to cover the main goals that I set my self at the beginning of my research:

1 Read a plan quickly and easily.
2 Implement the A.C.O to help find routes between a numbers of points.
3 Save the plan and reuse it again to compare different routes.

Further Development

By being able have multiply routes running simultaneously could result on seeing areas of congestion "as it happens". I would like to start to refine the selection process of the agent's next step. Perhaps by introduction a simple system of sight and distance, the movements of the agents can progress towards being less erratic.

References

[1] Dalton R.A., *Spatial Navigation in Immersive Virtual Environments*, PhD Thesis, University of London, 2001

[2] www.mathworld.wolfram.com/MinimalSurface.html

[3] Bonabeau E., Dorigo M. and Theraulaz G , *Swarm Intelligence*, Oxford University Press, 1999

[4] Resnick M., *Turtles, Termites and traffic jams: Explorations in massively Parallel Microworlds*, MIT Press, 1997