

Space Colonisation for Procedural Road Generation

Gabriel Dias Fernandes
Departamento de Informática
Universidade do Minho, Portugal
a71492@alunos.uminho.pt

António Ramires Fernandes
Centro Algoritmi
Universidade do Minho, Portugal
arf@di.uminho.pt

Abstract—Procedural road layout generation has been traditionally approached using L-Systems, with some works exploring alternative avenues. Although originally conceived for biological systems modelling, the adequacy of L-Systems as a base for road generation has been demonstrated in several works.

In this context, we present an alternative approach for procedural road layout generation that is also inspired by plant generation algorithms: space colonisation. In particular, we use the concept of attraction points introduced in space colonisation as the base of our approach to produce road layouts, both in urban and inter-city environments. As will be shown, the usage of attraction points provides an intuitive way to parameterise a road layout. Our method can also be used with flow-fields, demographics, geographical, and boundary maps to further fine tune the layout.

Keywords—procedural road generation, space colonisation, L-systems

I. INTRODUCTION

The procedural generation of content arises from the human curiosity and need to describe natural phenomena with maths. Fibonacci sequences and the golden ratio are well known nowadays by being present in grow patterns in nature.

In 1968, Aristid Lindenmayer [1], a biologist, introduced L-Systems to describe cellular algae growth. Later, Lindenmayer and Prusinkiewicz ([2]) used L-systems to generate fractals, and other geometrical patterns, as well as 3D bushes and trees. The procedural generation of trees and bushes, using L-Systems, was the focus of [3] which achieved very realistic results. Given the flexibility of L-Systems, they started to be widely used in several topics among procedural generation.

In procedural generation of roads, L-Systems were also the starting point of the early generation algorithms, with a significant amount of the subsequent works being based on this approach.

In this work, an alternative approach to procedural street layout generation is presented, combining different ideas from previous works but using a different base algorithm.

The algorithm used as the cornerstone of this work is the *Space Colonisation Algorithm* (SCA). Presented in [4] by Runions et. al., SCA was designed to procedurally generate trees. Our motivation is explore the concept of attraction points introduced in SCA to direct the growing of roads.

We opted for SCA because it is a more intuitive and predictable algorithm than L-systems. With SCA it is feasible

to imagine the final output based on the input parameters, something which is hard when considering the axioms and rules for an L-system.

By design SCA returns a tree like structure, which we need to transform into a graph like structure to be able to simulate a road layout. This is our main contribution: to extend SCA for road layout generation.

The remainder of this document is organised as follows: section II provides an overview of previous related work; in section III we present the algorithm in which we based our approach; section IV details our extension to SCA so that it is suitable for road layout generation; section V presents some results obtained with more complex settings; section VI presents our conclusions and suggests some avenues for further research.

II. RELATED WORK

Concerning the procedural generation of road networks, Parish and Müller [5] produced one of the earliest works. They presented a system called *CityEngine*, capable of generating a city road network based on L-Systems and maps as inputs. These maps contain information regarding population density and land-water boundaries. L-systems were extended to be able to consider local and global constraints. Furthermore, they present a system to procedurally generate buildings and the appropriate textures, making it a very complete system.

Sun et. al. [6] focus on highways, and use road templates to produce radial, population-based, raster configurations. These templates can be used together to produce a mixed layout. The authors also use land/water/vegetation and elevation maps.

Citygen [7] is an interactive engine, fully configurable by the user. The user defines the primary roads by creating a graph, which is superimposed in an elevation map. Roads are generated based on several algorithm variations such as a minimum elevation strategy. Inside a city, cells are found by considering enclosed regions of the primary graph. Then a variant of L-systems is used to fill the inside of those cells.

Autopolis, a system presented in [8] and [9] by Teoh, starts by asking the user to define a set of initial parameters such as the number of commercial and industrial centres, airports, among other items. The system then uses a set of heuristic to place these items and grows a network of roads to include residential centres starting with highways to connect the centers, followed by main and secondary roads. Highways are created using the shortest path weighted with cost. Vector

fields are created based on either elevation, coastlines, or existing roads, for the main and secondary roads.

In [10], by Chen et. al., the generation and modification of the road network is based on interactively created tensor fields. The roads will be created following the directions described by the tensor field. To achieve more organic results, Perlin noise [11] is used to provide some controlled randomness.

Webber et. al. in [12] propose a 4D simulation, including time to simulate the growth of cities. Regarding street generation their concern is that the street layout must make sense at any time t . Therefore, their strategy is based on expansion of the previous street layout in a two stage process: expanding major streets and filling quarters with minor streets. To determine which streets to grow at any particular time a probabilistic approach is used based on the distance to the nearest growth centre. Direction, length, and other street parameters are chosen based on the street pattern which can be radial, organic or grid based.

Aiming towards the generation of a single realistic road, Galin et. al. [13] proposed a flexible algorithm to deal with obstacles such as forests, rivers and mountains. By specifying a starting and ending point of the road to be generated, the goal is to find the shortest path, weighted by cost, between those points. The algorithm relies on the existence of a cost function that can be evaluated at an point on the map to determine the best solution. For instance the cost function may dictate that is cheaper to build a bridge than to contour the water areas, or build tunnels, or bridges to deal with steep elevation changes. Later, in [14], this work was extended to generate hierarchical road networks.

Lindorfer et. al. [15] method also starts by creating highways between highly populated centres. The regions delimited by the highways are then partitioned to create local transport roads. A modified environment sensitive L-system algorithm is used to create the street maps. As in Sun et. al [6] they also use patterns, namely the raster and radial pattern.

Yang et. al. [16] propose an interactive system where the user starts by specifying main roads. The map is then divided in cells limited by these roads. For each cell the user can then specify a template to fill the cell.

Campos [17] based his method in the procedures used in road engineering, producing road paths definitions according to design standards, current practices in road design, thus producing roads similar to those found in the real world.

Nishida et. al. [18] approach is based on the extraction of patches from real road maps, where a patch is defined as being a meaningful road structure. An interactive system was built in which the user selects areas from real-road maps from which patches are extracted and used in the new road layout. Later, Teng and Bidarra [19], extended the core patch idea to include semantics, and allow a higher level editing based on these semantics.

In conclusion, there is some diversity on the approaches taken, although a very significant number uses L-systems and/or extensions. The earliest works explore L-Systems, or variants, as the main procedural algorithm to generate both

the roads and buildings in cities. Other algorithms include the shortest path algorithm, minimum spanning trees, templates, and even tensor fields. More recent works are based on pattern extraction from real world road layouts.

III. SPACE COLONISATION FOR TREE GROWTH

The essence of the SCA is the notion of attraction points, which promote branch growth and bifurcation. This led us to consider the application of attraction points to procedural road layout generation. In this section we will provide a short introduction to the algorithm in which our work is based.

A. Space colonisation algorithm

SCA is presented by Runions et. al. in [4] and in summary consists in a tree generation algorithm that promotes space colonisation.

A tree is composed by *tree nodes* connected between them. The tree growth will be influenced by a set of *attraction points* that will dictate how the tree will grow and develop.

Initially, attraction points are scattered on the desired volume, together with an initial tree node or nodes. The tree will grow starting from the existing tree nodes, towards attraction points that influence them. An attraction point can only influence the closest tree node, being required that the distance between the two is less than a given distance called *radius of influence*.

Although an attraction point can only influence one tree node, a tree node can be influenced by many attraction points.

For any tree node under the influence of attraction points, a new node is created along the direction given by the average of all normalised directions between the current tree node and all attraction points that influence it. The new tree node will be placed at a distance D , the segment length, from the predecessor tree node.

All attraction points used to influence a new node creation that are at a distance from the node which is less than a pre-defined threshold, called the *kill distance*, are eliminated.

All these steps are repeated until all attraction points are consumed, the tree is no longer attracted by any points, or the user decides to stop the algorithm.

Figure 1 presents the algorithm graphically.

B. Leaf venation patterns

Prior to the space colonisation algorithm, Runions et. al. published "*Modeling and visualisation of leaf venation patterns*" ([20]). The leaf venation patterns can be distinguished by two types: open and closed patterns, as seen in Figure 2. The open venation pattern consists on a vein network tree like structure, where existing vein nodes do not connect to each other, except when a new node is generated. It will result in a more or less sparse network that it is not intra-connected. The algorithm that generates the open venation pattern, is essentially the space colonisation algorithm and is the method that gave rise to the latter work [4]. As for the closed venation pattern, this is a more usual pattern found in nature. A dense vein graph network may have a ladder-like pattern (*percurrent*) or a cracked/netlike pattern (*reticulate*).

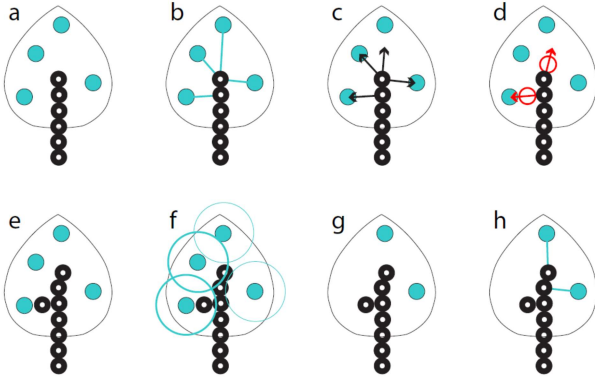


Figure 1: Space colonisation steps. Black dots - Tree nodes; Blue dots - Attraction points; Blue lines - Influence link; Black arrows - Normalised directions; Red dots - New tree nodes; Blue radius - Kill distance. Source: [4]

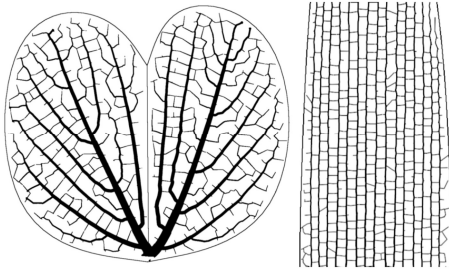


Figure 2: Left: Reticulate pattern (Orchid leaf); Right: Percurrent pattern (Grass leaf). Source: [20]

Getting into further detail about the closed venation pattern, it is based on the open venation pattern but it also allows that more than one vein may grow towards the same source. This feature is to promote the formation of anastomoses, the connection of two different veins. To be similar to nature, anastomoses should be done when the two or more veins are close enough to each other, yet they are relatively far from each other. To do so, the algorithm will use the notion of relative neighbourhood. Two points, s and v , are relative neighbours of each other if for every any other point u from the set of all points that are closer to s than v , v is closer to s than to u .

In Figure 3, we may see that v , a and b are relative neighbours of s , and we can confirm it by checking their relative distances shown by lines. The green area admits the points that are both closer to s than v , and that v is closer to them than to s , in other words, that area must be empty otherwise v would be promptly discarded as neighbour of s as said before. The blue area consists in every point that is closer to v than to s , which are promptly discarded as well.

As already seen in Figure 2, this algorithm produces extremely realistic results in terms of generating a venation network that not only tries to mimic botanical elements visually but also how do they work biologically (auxin sources,

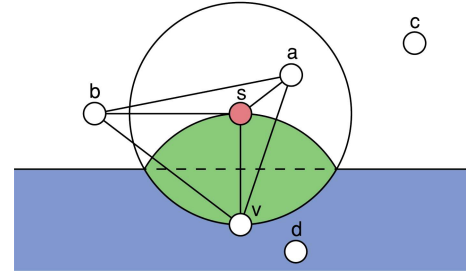


Figure 3: Relative neighbourhood example. Source: [20]

anastomoses).

This could have been the perfect algorithm to solve the problem of inter-connecting the main roads of a network, however, the relative neighbourhood method which is the main reason why the closed venation pattern is so realistic and optimal, is not flexible enough in terms of controlling its output in a road layout context.

Hence, we chose to use SCA as the base for our approach, and to further parameterise and customise to suit our needs of generating a road topology.

IV. SPACE COLONISATION IN THE CONTEXT OF ROAD LAYOUT GENERATION

This section covers our proposed extension to SCA. As in SCA [4] we use attraction points to shape the final network. The start of the process is to create a cloud of attraction points (in 2D). This cloud can be influenced by information maps (IV-C1). Attraction points will be used for road segment growth, much like in the original algorithm. We also need to specify a set of initial segments.

Similarly to many of the previous road generation methods, our approach is a two phase algorithm. The first phase creates a 2D network of roads in what is mostly a tree like structure (IV-A). The second phase further produces the connections between nodes created in the first phase (IV-B), effectively creating the required graph structure. Additional configuration features, namely a flow field and information maps are discussed next (IV-C).

A. First phase

The first phase of the algorithm aims to generate what is mostly a tree shaped road network. When the road network cannot grow further, due to consuming all the attraction points or by not being influenced by them anymore, we have two options: we add more attraction points and continue growing the initial road network, or we proceed to the second phase.

In here we explore some of the possible parameterisation for the first phase, starting with the original SCA's parameters, followed by our extensions.

1) *Capillarity*: Different topologies require different settings. For instance, the main roads of a country or a network that represents a city map differ in terms of density, or *capillarity*. Working towards these layouts can be achieved

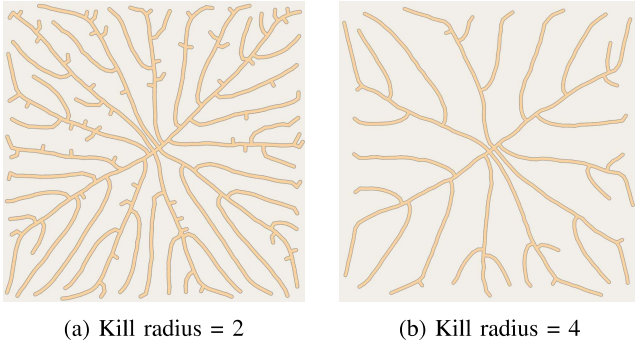


Figure 4: Capillarity influence result.

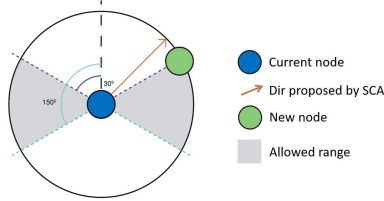


Figure 5: Angle control example.

with SCA original parameters: *radius of influence* and *kill radius*. A network can have more or less capillaries depending on the ratio between the *radius of influence* and *kill radius* parameters. As discussed in [4] the *radius of influence* should be larger than the *kill radius*. The control of this feature is very intuitive, for instance, a dense network is generated by having a large ratio between the *radius of influence* and the *kill radius*, otherwise, we have a sparse network. In other words, with a low *kill radius* an attraction point will tend to live longer than with a higher radius, causing the growth of more road segments.

Two full-grown road network results based only in capillarity are presented in Figure 4. These were both generated using the same uniform distribution of attraction points and *radius of influence* set to 5, with the first node placed in the middle of the grid and with different *kill radius*.

2) *Angle constraints*: To control the appearance of the initial road layout we extended SCA to include parameters to control the angle between segments of the road. These parameters impose a maximum and minimum angle for newly created nodes and segments. Proposed directions outside the defined range are snapped to the valid range as shown in Fig. 5. With this feature we can create a rigid grid-like city map, or a more organic look as shown in Figure 6. As an extra option we also allow the road to grow in a straight direction, even when setting angle constraints. This effect can be seen most pronounced in Figure 5 c).

3) *Road hierarchy*: In real-world road networks, there is a hierarchy that classifies roads by their relevance. An automated procedure creates this hierarchy, classifying roads based on the number of attraction points consumed by their segments. We can cope with any number of hierarchical levels based on user defined thresholds. We can further restrict road hierarchical

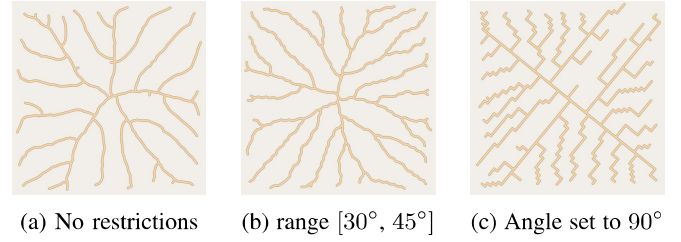


Figure 6: Angle restriction effect on the road network.



Figure 7: Automatic road hierarchy: red primary ; yellow secondary; grey:tertiary roads.

classification per phase, for instance, considering roads created in the first phase as being either primary or secondary, and relegate all roads created in the second phase to being tertiary roads, reducing the number of threshold parameters to one. Using attraction point count per segment as the base for defining a hierarchical level is an intuitive approach that is directly related to the density of attraction points in specific areas. As seen in IV-C1, the distribution of attraction points can be made to match population density providing more important roads in higher population density areas.

The road hierarchy algorithm starts by considering the initial segments defined by the user as primary roads. Road classification is performed on each new segment as follows: when the new segment implies a bifurcation, we evaluate the road relevance for the new segment depending on the number of attraction points that influenced the creation of the segment; otherwise, when the segment is just extending the previous road, we set its relevance equal to the previous segment. We also found it convenient to ensure that the relevance of a new segment can never be set higher than the previous segment. Figure 7 shows two examples of the automatic road classification in the first phase. With this approach we obtain long primary roads, and shorter roads with less relevance.

4) *Snapping and Merging*: Mainly due to the angle limitation feature it is possible that newly created nodes are too close, or even superimposed, to other nodes or segments as is the case in Fig. 6 c).

If the proposed node is too close to any other road segment or node, by a given distance, called *snapping distance*, one of two things will happen: if the nearest part of the road network to the proposed node is a previously existing node, the proposed node will be merged with this node, if the nearest

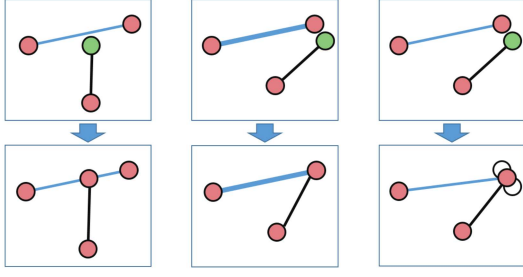


Figure 8: Snapping modes. Left: snap to segment; Middle: snap to higher relevance node; Right: snap both nodes to average position.

part is in the midst of a road segment, it will snap to it creating a new node in that position.

Node merging takes in to account the road hierarchy. When merging two nodes belonging to roads with the same relevance, the new location is the average location of both nodes. Otherwise, the new position will be the position of the node of the most relevant road to maintain the most relevant road direction.

Figure 8 illustrates this procedure, where road thickness relates to road relevance. This is similar to the local constraints defined in [5].

B. Second phase

In general, the first phase tends to produce a road network shaped mostly like a tree. To get an inter-connected road graph network we run a second phase. In here we detail our approach to achieve this.

1) *Road inter-connection*: The first step in this phase is to directly connect nodes created in the first phase. These connections will take into account two different aspects: the nodes to be connected should be close enough to each other, and the suggested connection should make an angle with the previous node connections that should be as close as possible to the *ideal angle*. The *ideal angle*, is the angle that allows the new segment to be the most distant to any of the other existing segments of the node, to promote a distributed network.

A node can have multiple ideal angles based on different scenarios depending on the cardinality of the node connections. When there is none, the ideal angles are perpendicular or parallel to the previous connection with the node. When there is at least one descendent from the node, the ideal angles are the average between the subsequent directions of the connections from and to this node.

Figure 9 presents two examples on how the connecting mechanism works. Nodes A and B are both inside the valid area, in which case the connection will be from the current node to B because it is closer to the ideal angles.

When applying this procedure as described we noted that there was a tendency to form consecutive triangular connections, something that is not observable in real roads. Therefore, we added a restriction to prevent the formation of triangular connections in this step.

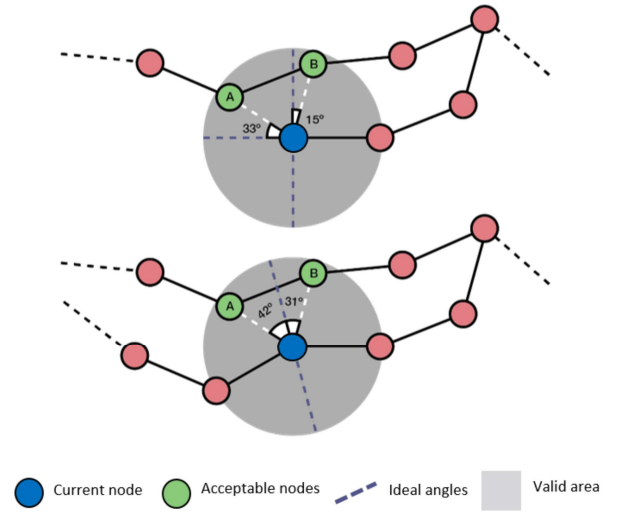


Figure 9: Node connection mechanism.

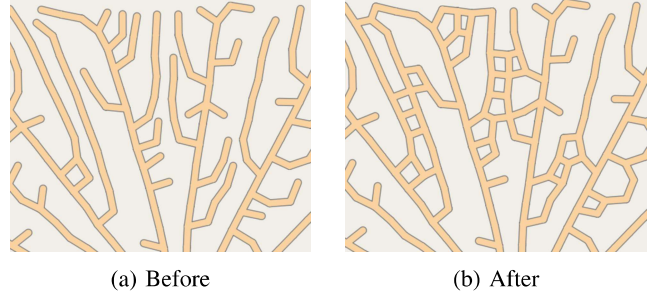


Figure 10: Road inter-connection impact on the network.

An example of the application of this step is presented in Figure 10, showing both the network at the end of the first phase and the end result of this step.

2) *Attraction point validation*: In the general case, to continue to grow the road network in the second phase we need to add more attraction points. Since there is already a formed road network, this attraction point distribution cannot be completely random, otherwise new attraction points may cause the network to grow roads that can cross the original network, or create road segments too close to existing ones.

To prevent this we've added a parameter to control the minimum required distance to accept a new attraction point. Attraction points inside this safety margin will be removed, see Figure 11.

3) *Some notes regarding attractors*: In here we present some extensions we made to SCA in order to inter-connect the roads from the first phase to make the attraction points more influential.

To achieve the desired inter-connection between the first phase nodes, these nodes will also be considered as attractors. Furthermore, attraction points will attract the N closer nodes instead of just the closest node, unlike the original SCA (we found that $N = 2$ provides more pleasing results).

To guarantee that the road network will successfully connect

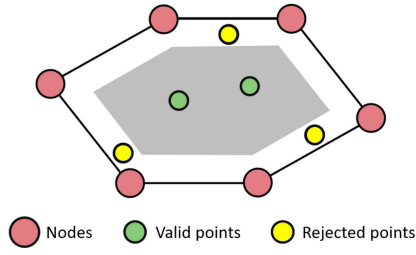


Figure 11: Attraction point validation example.

itself we must extend the *lifespan* of the attraction points relative to the first phase. This can be achieved setting a lower kill radius in this phase. By default we set it at half the value from the first phase.

C. Additional features

In addition to the features described before, it is possible to complement the results with external information. This information is given through maps representing information about population density, type of terrain and elevation. These maps will condition and change how the network will grow.

As an alternative to setting angles to condition direction, or in combination with them, it is also possible to use a flow-field to impose restrictions on the direction for new segments.

1) *Information maps*: A population density map is a grey level map where lighter areas imply higher population density. These maps will condition the placement of attraction points. When attraction points are placed they are accepted/rejected based on a probability taken from the intensity.

The border map has information about illegal areas to grow roads, such as rivers, lakes, sea, city and natural parks. This will dictate which areas are available to support the road network. It can be extended to differentiate areas so that the algorithm can behave differently when generating the road network according to those areas. Attraction points placed in illegal areas will not be considered for road generation, and road segments have to be checked to ensure they do not cross these illegal areas.

The height map contains the height of each point in the grid/terrain. Darker tones are associated with low elevations, whereas lighter tones correspond to higher altitudes. These maps are used to determine the steepness of the terrain in order to decide if a road can grow in a particular direction based on a parameter *maximum steepness*. An example is presented in sec. V-B.

2) *Flow field*: Many scientific areas use vector fields to represent some models and phenomena such as force, wind movement, gravitational force, *et cetera*. In our context, we call them *flow fields* since they influence the direction of new segments.

To build flow fields we use flow primitives, based on the work by Wejchert and Haumann [21]. The primitives will shape the field, influencing the direction of the vectors in the grid, corresponding to the building blocks of the flow field:

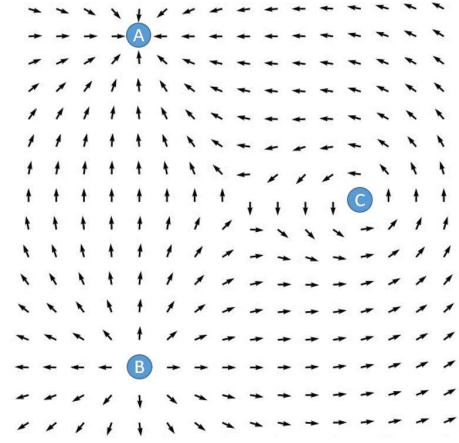


Figure 12: Flow field with primitives: A is a sink; B a source, and C a vortex.

uniform which causes a flow in straight lines, parallel to the direction specified by the primitive; *source*, which creates a flow away from the location of the primitive; *sink* which flows towards the primitive; and *vortex* which causes a circular flow around the primitive. Each primitive is described by a linear equation, and the cumulative result at each grid point can be computed as the sum of the influences for all primitives. Under our approach, a flow field is defined by placing these primitives. Figure 12 shows a flow field created with three primitives, and Figure 13 shows an example of a road layout created with the flow field.

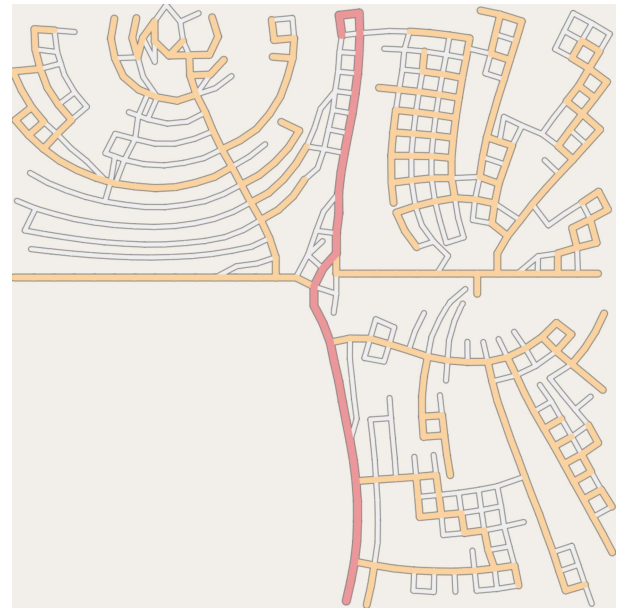


Figure 13: Road layout for flow field from Figure 12.

When using a flow field the direction of new segments is limited to either follow the flow or perpendicular to it.

The flow field can be used as a replacement of the angle restriction parameters, or in a combination of influence for

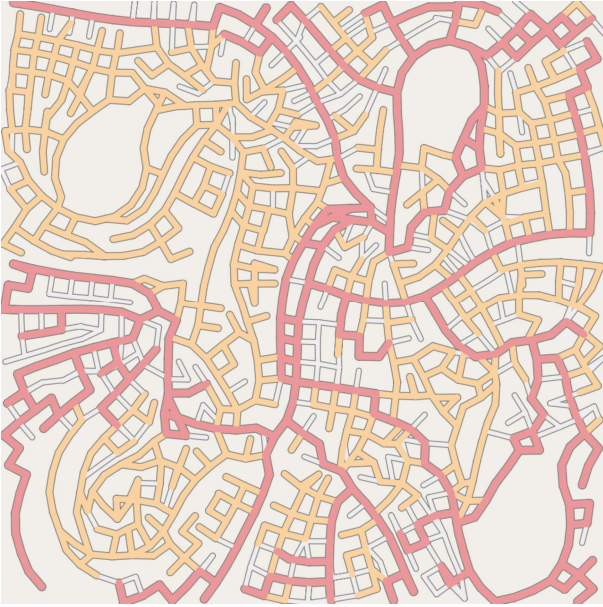


Figure 14: Town with parks.

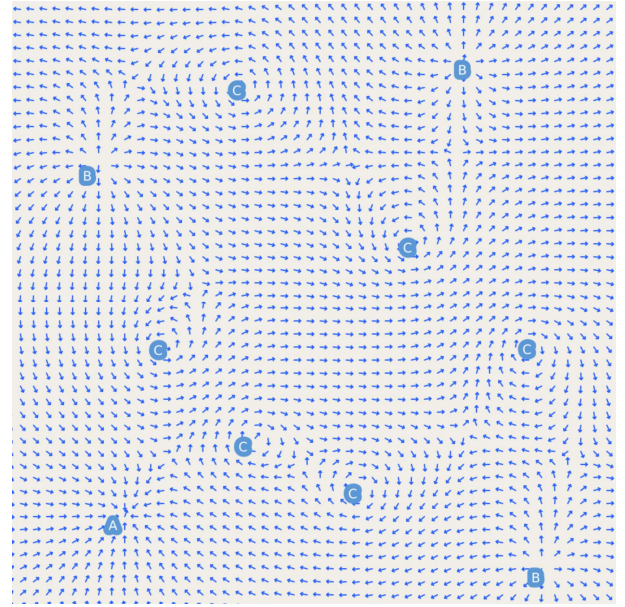


Figure 15: Flow field for road layout in Figure 14

the direction of the new road segments.

V. ADDITIONAL RESULTS

A. Flow fields

In here we present more complex road layouts based on flow fields. Figure 14 shows a layout generated with the flow field presented in Figure 15. The sources (label B) in the flow field tend to create empty spaces and vortices (label C) create curved roads. These primitives create patterns that can simulate various layout types. For instance, Figure 14 resembles a town with large parks in the empty areas, where as Figure 16 could be a more modern, grid like, coastal area.

An example where the initial segments are placed as a ring road around the relevant area and also using a flowfield can be seen in Figure 17. Attraction points were placed mainly inside the area delimited by the initial ring road.

B. Dealing with elevation data

The algorithm can also deal with height taken from elevation maps based on a parameter that controls the *maximum steepness* of generated roads. Figure 18 shows elevation information in grey scale. From the elevation and a border map of the island, we tested the growth of the road network limited by the steepness of the terrain setting *maximum steepness* to 10° . The solution found covers the whole island, still there are some connections missing. This leads to the conclusion that an extra step based on path finding might be required to perform these connections.

VI. CONCLUSION

We based our method on the concept of attraction points from SCA. An extension to SCA was presented as a two phase algorithm, both phases using a modified SCA algorithm. The first phase grows mostly a tree like structure, and the second



Figure 16: Coastal area

performs inter road connectivity, effectively transforming the initial tree structure into a road layout graph. The parameterisation of the algorithm to suit different road layouts was presented and exemplified. We find the parameter setting to be intuitive as there is a direct connection between the parameter values and the geometry of the road layout. For instance, road hierarchy is directly related to the density of the distribution of attraction points, and the kill radius also provides an intuitive way of controlling the capillarity of the roads. We also showed how our proposal could use a flow field and information maps to further tune the road layout.

Regarding future work we believe that it would be inter-

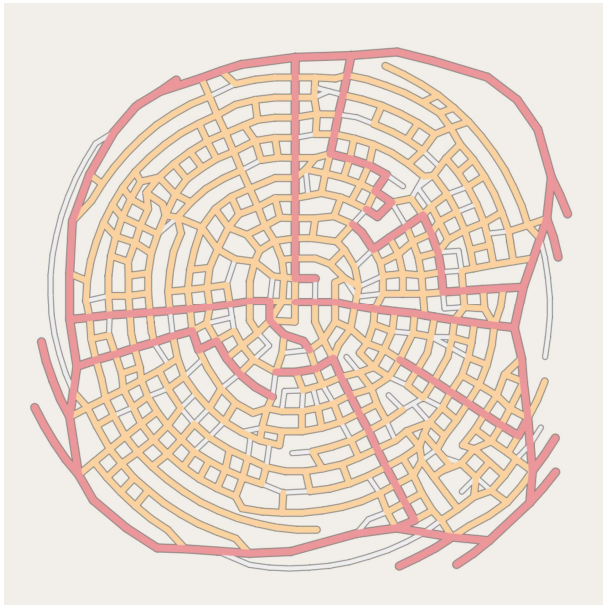


Figure 17: Defining initial segments as a ring road.

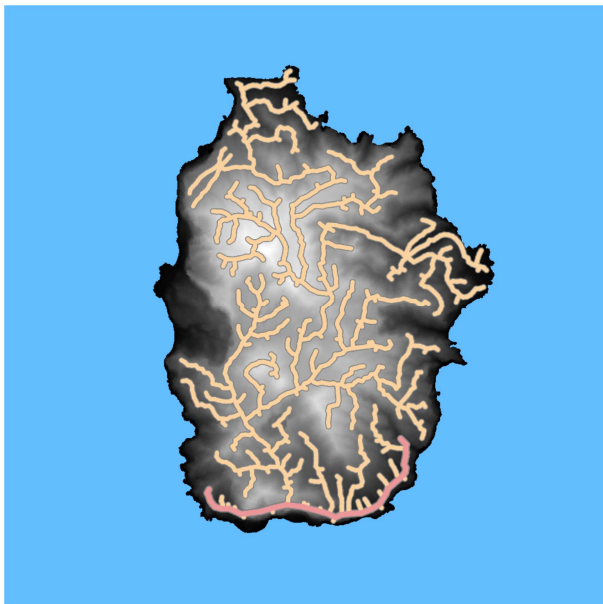


Figure 18: Road layout with elevation map.

esting to further parameterise the distribution of the attraction points taking into consideration points of interest, such as, for instance, shopping malls, train and bus stations. Allowing a radius of influence for the primitives in a flow fields, and exploring different primitives might also provide interesting road layout patterns.

ACKNOWLEDGMENTS

This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT – Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/00319/2013.

REFERENCES

- [1] A. Lindenmayer, "Mathematical Models for Cellular Interaction in Development," *Journal of Theoretical Biology*, vol. 18, pp. 280–315, 1968.
- [2] P. Prusinkiewicz and A. Lindenmayer, "Graphical modeling using L-systems," in *The Algorithmic Beauty of Plants*. Springer New York, 1990, pp. 1–50.
- [3] J. Weber and J. Penn, "Creation and rendering of realistic trees," *SIGGRAPH '95 - Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 119–128, 1995.
- [4] A. Runions, B. Lane, and P. Prusinkiewicz, "Modeling trees with a space colonization algorithm," in *Natural Phenomena*, ser. NPH'07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 63–70.
- [5] Y. I. H. Parish and P. Müller, "Procedural modeling of cities," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 301–308.
- [6] J. Sun, X. Yu, G. Baci, and M. Green, "Template-based generation of road networks for virtual city modeling," *Proceedings of the ACM symposium on Virtual reality software and technology - VRST '02*, p. 33, 2002.
- [7] G. Kelly and H. McCabe, "Citygen: An interactive system for procedural city generation," *Fifth International Conference on Game Design and Technology*, pp. 8–16, 2007.
- [8] S. Teoh, "Autopolis: Allowing user influence in the automatic creation of realistic cities," *Advances in Visual Computing*, pp. 118–129, 2007.
- [9] S. T. Teoh and T. Soon Tee, "Algorithms for the automatic generation of urban streets and buildings," *Proceedings of the 2008 International Conference on Computer Graphics and Virtual Reality (CGVR 2008)*, vol. 1, pp. 122–128, 2008.
- [10] G. Chen, G. Esch, P. Wonka, P. Müller, and E. Zhang, "Interactive procedural street modeling," in *ACM SIGGRAPH 2008*, vol. 27, 2008.
- [11] K. Perlin, "An image synthesizer," *ACM SIGGRAPH Computer Graphics*, vol. 19, no. 3, pp. 287–296, jul 1985.
- [12] B. Weber, P. Müller, P. Wonka, and M. Gross, "Interactive geometric simulation of 4D cities," *Computer Graphics Forum*, vol. 28, no. 2, pp. 481–492, 2009.
- [13] E. Galin, A. Peytavie, N. Maréchal, and E. Guérin, "Procedural generation of roads," *Computer Graphics Forum*, vol. 29, no. 2, pp. 429–438, 2010.
- [14] E. Galin, A. Peytavie, E. Guérin, and B. Beneš, "Authoring hierarchical road networks," *Computer Graphics Forum*, vol. 30, no. 7, pp. 2021–2030, 2011.
- [15] M. Lindorfer, C. Backfrieder, C. Kieslich, J. Krösche, and G. Ostermayer, "Environmental-sensitive generation of street networks for traffic simulations," in *Proceedings - UKSim-AMSS 7th European Modelling Symposium on Computer Modelling and Simulation, EMS 2013*, 2013, pp. 457–462.
- [16] Y.-L. Yang, J. Wang, E. Vouga, and P. Wonka, "Urban pattern: Layout design by hierarchical domain splitting," *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013)*, vol. 32, p. Article No. xx, 2013.
- [17] C. Campos, "Modelação procedimental de ambientes rodoviários para simulação de condução," Ph.D. dissertation, Faculdade de Engenharia da Universidade do Porto, 2015.
- [18] G. Nishida, I. Garcia-Dorado, and D. G. Aliaga, "Example-Driven Procedural Urban Roads," *Computer Graphics Forum*, vol. 35, no. 6, pp. 5–17, 2016.
- [19] E. Teng and R. Bidarra, "A semantic approach to patch-based procedural generation of urban road networks," in *Proceedings of PCG 2017 - Workshop on Procedural Content Generation for Games, co-located with the Twelfth International Conference on the Foundations of Digital Games*, 2017. [Online]. Available: <http://graphics.tudelft.nl/Publications-new/2017/TB17>
- [20] A. Runions, M. Fuhrer, B. Lane, P. Federl, A.-G. Rolland-Lagan, and P. Prusinkiewicz, "Modeling and visualization of leaf venation patterns," *ACM Transactions on Graphics*, vol. 24, no. 3, p. 702, 2005.
- [21] J. Wejchert and D. Haumann, "Animation aerodynamics," *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 19–22, Jul. 1991. [Online]. Available: <http://doi.acm.org/10.1145/127719.122719>