Urban Fabric Generation
A comparative analysis of multiple vector field methods

F. Peter Ortner[1], Zebin Chen[2], Peng Song[3], Pengyun Qiu[4]
[1,2,3,4] Singapore University of Technology and Design
[1]{peter_ortner |zebin_chen| peng_song| pengyun_qiu}@sutd.edu.sg

This paper provides a comparative analysis of vector field methods for automatically generating urban fabric geometries and introduces the Rhino/Grasshopper plugin UrbanFab which implements these methods. Automated generation of urban design is a growing area of research addressing the complex challenge of building sustainable and economically viable cities. Within this research area vector fields are used to generate non-uniform urban fabric geometries. Similar to simulation of magnetic fields or moving fluids, vector field algorithms can be adapted to simulate urban sites, resolving complex site constraints and generating geometry used in the design of roads, land parcels or buildings. Vector field methods for urban fabric generation, however, are not well represented in computational tools serving urban designers, or in description of algorithms and evaluations in computational urban design literature. To address these challenges, this paper describes multiple urban vector field algorithms, with accompanying streamline visualization and evaluation methods. A comparative analysis of the results generated by these methods provides a means for designers to make informed decisions on which method is appropriate to their requirements and supports discussion of future work in urban fabric generation.

Keywords: *Urban Planning, Urban Design, Generative Design, Vector Field, Optimization.*

INTRODUCTION

Automated urban fabric generation is an area of growing interest in computer graphics and computational design, driven by the advent of planner-oriented design tools like ESRI's CityEngine (Parish et al., 2001; Vanegas et al., 2012) as well as new developer-oriented webtools like Delve (Sidewalk Labs) and Digital Blue Foam. The motivation behind the increasing prevalence of these tools is not only the democratization of their development and use via the web, but also growing disciplinary concern for the grand challenge of the design of the sustainable and resilient city to which they contribute.

Urban fabric, which is used here to refer to the geometric antecedents and constituent elements of road networks and urban parcels, exerts a disproportionate influence on the performance of a city plan. Urban fabric impacts vehicular and pedestrian mobility, wayfinding, placemaking, as well as density and orientation dependent sustainability concerns like the urban heat island effect. Optimization of these performative factors is necessary for achieving a sustainable and resilient city, implying many rounds of quantitative evaluation and redesign of any proposed solution.

Generative tools for urban fabrics must meet a dual challenge of extensiveness and explicitness. Extensive refers to the ability to generate a wide-

enough variety of meaningful options to meet the exploratory needs of the design team. Explicitness supports intentionality and iterative refinement by human designers and associated stakeholders.

To address the dual challenge of an extensive and explicit design tool for urban fabrics this paper presents multiple methods of vector field generation with the capability of producing extensive options for streamline based urban road network and parcel generation. To support designer understanding and iterative control of outcomes from the tool, the paper presents explicit descriptions of the algorithms involved and methods for evaluation of generated geometries. The combined vector field, streamline generation and evaluation tools are packaged in a Rhino/Grasshopper plugin *UrbanFab* presented for the first time in this paper (Figure 1).

The presented methods are introduced in the context of related work from computer graphics and artificial intelligence (AI) on vector fields and urban fabric generation. Results are presented from testing the methods on multiple urban sites, and the plugin interface is presented in a discussion of relevant applications and future work.

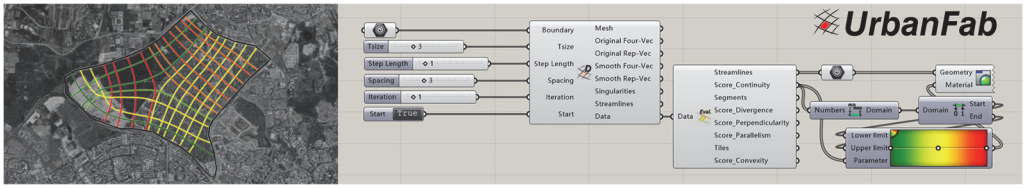


Figure 1
UrbanFab plugin components and output

RELATED WORK

While this paper puts forward vector-field based approaches to urban fabric generation, recent work in automated generation of urban fabrics has expanded to include generative AI. The AI approaches present different capabilities from the vector-field based approach. Neural Turtle Graphics, based on a recurrent neural network, generates a variety of networks that imitate and remix characteristics of existing city road networks with good accuracy and diversity (Chu et al., 2019). DeepStreet (Zhou et al., 2020) presents a raster-based method which takes into account road network and topography and uses a deep learning approach to infill patches of road networks within test examples with good accuracy particularly for city center/ gridded areas. These approaches produce results that are both accurate enough to be directly implemented in urban design process and are arguably diverse enough to permit new design directions. The challenge of explicitness is more difficult to meet with AI urban fabric tools as the generated results are controlled based on statistical similarity to existing city datasets but without an explainable procedural logic for the generation of each element. Practically, we have found that vector fields provide a valid contrast to AI urban fabric generation tools due to their ability to rapidly subdivide complex site boundary conditions with a wide variety of novel options that are valid in real-world applications while maintaining explicitness of generative logic.

In computer graphics and geometry processing, several vector field types have been defined (Vaxman et al., 2016) with multiple optimization methods applied to urban fabric generation in recent literature (Chen et al., 2008; Yang et al., 2013; Zhang et al., 2022). Vector fields can automatically generate solutions for urban street/parcel lines in complex sites where boundary curves, concavities or other constraints or contextual features make straight grids unsuitable.

Vector field outputs are similar to recent urban developments in Singapore where regular urban grids are locally adapted to boundary constraints. The case studies of Tengah New Town, Punggol New Town and Jurong Lake District (Figure 3) represent recent or proposed developments in Singapore where comparison between designed urban fabric and vector field outputs demonstrate the value of

Figure 3
3 case study sites from Singapore: Tengah, Punggol, and Jurong Lake District (Left to Right)

the proposed methods as well as areas of potential improvement.

Clear methodological explication and comparative analysis of the several vector field types relevant to urban fabric generation are limited in the literature of the computational design field and often presented with confusingly vague or synonymous terms (tensor field vs. vector field vs. distance field). Access to vector field tools for urban fabric is also currently limited, with most publications referring only to custom tools that are not publicly released. Together these challenges limit the ability of designers to apply these techniques in practice. To address these challenges, this paper provides an overview of vector field types relevant to urban fabric generation and matches the presented terms with algorithms. Further supporting designer understanding are the presented evaluation methods: divergence, continuity, boundary perpendicularity, parallelism and singularity identification.

FIELD DESIGN METHODS

This section covers the set-up for 3 types of 4-vector fields: 1) distance field; 2) boundary field; 3) harmonic field. Based on initial workshop sessions with Singapore's Urban Redevelopment Authority (URA) this research prioritizes the 4-vector field as it produces streamlines that intersect at approximately right angles, ensuring that subdivided land parcels are approximately rectilinear. We build upon these techniques, published notably by Yang et al. (2013), adding more detail on algorithms and proposing additional methods of evaluation relevant to urban designers.

Distance field

The distance field method calculates the closest point Cp of all vertices V to the site boundary ∂M, identifying the vector d oriented to the closest point from which the four-vector Df is derived, constructing the full 4-vector field DF. For all algorithms shown, the 4 vectors Df $(d_i = (\sin(u_i), \cos(u_i)), i = 0,1,2,3)$ at each vertex v can be uniquely represented as a representation vector $D = (sin(u), cos(u))$. This distance field presents visible discontinuities on the medial axis. To minimize this, seedpoints on the medial axis can be excluded in this method. Some discontinuities can still be observed in Figure 2 near the medial axis. In addition, the field can be updated by calculating the average value of D on incident vertices IV, making streamlines generated by the field smoother (Algorithm 1).

Algorithm 1: Distance Field

```
∂M ← Get the boundary of Mesh M
V ← Get the set of vertices of Mesh M
for all v ∈ V do
    Cp ← Find closest point of v on ∂M
    d ← Create vector by v and Cp
    Df ← Rotate d by 90, 180 and 270 degrees
    D ← Create representing vector with d
    if smooth then
        While n < iteration do
            IV ← Find incident vertices of v
            D ← Average D of IV
            Df ← Encode D and return 4 vectors
            n+=1
        end while
    end if
end for
DF ← Get all the Df from v ∈ V
return DF // 4-vector field
```

Boundary field

The boundary field method allows the user to set constraints with weights. This paper demonstrates boundary constraints: for a vertex v on the boundary ∂M, the representation vector $\tilde{D}$ as the constraint can

be computed by using the perpendicular vector $\tilde{d}$ to the boundary. The field is obtained by solving the quadratic variational problem with D_i's constraint (Equation 1, where *i* is the vertex and *j* is an incident vertex) and uses the method of Lagrange multipliers to solve the problem (Algorithm 2). Equation 1 can be extended to introduce more constraints/weights $\{C_1, C_2 \ldots\ldots C_n\}$ to control the result of the field; for example including constraints based on additional topographic features (Equation 2).

$$\min_{D} \sum_{i,j} \|D_i - D_j\|^2 + \omega \sum_{i \in bdry} \|D_i - \widetilde{D}_i\|^2 \quad s.t\ \|D\| = 1 \tag{1}$$

$$\min_{D} \sum_{i,j} \|D_i - D_j\|^2 + \sum_{k=1}^{n} \sum_{D_{k_i} \in C_k} \omega_k \|D_{k_i} - \widetilde{D}_{k_i}\|^2 \quad s.t\ \|D\| = 1 \tag{2}$$

Algorithm 2: Boundary Field

```
∂M ← Get the boundary of Mesh M
V ← Get the set of vertices of Mesh M
for all v ∈ V do
    d ← Initialize a vector randomly
    Df ← Rotate d by 90, 180 and 270 degrees
end for
for all v ∈ ∂M do
    d̃ ← Boundary perpendicular vector
end for
while n < iteration do
    for all v ∈ V do
        IV ← One ring incident vertices of v
        D ← Get representation vector with d
        if v ∈ ∂M do
            D ← Solve Equation 1
        else
            D ← Solve the first half of Equation 1
        end if
    end for
end while
DF ← Get all the Df from v ∈ V
return DF // 4-vector field
```

Harmonic field

The harmonic vector field method builds on the mathematical concept of a conformal map. In the discrete triangulation mesh, because the constructed discrete 4-vector field is required to satisfy the condition that the 4-direction field of vertices on the mesh boundary is either perpendicular or parallel to the boundary edges, the interior vertices should satisfy the discrete Laplacian equation with the specific boundary conditions (Algorithm 3). This approach is implemented to find the 4-vector field (Yang et al.,2013) by computing and minimizing the energy of the field (Equation 3).

$$E(D) = \int_R (\nabla \cdot d)^2 \, dA \tag{3}$$

Algorithm 3: Harmonic Field

```
∂M ← Get the boundary of Mesh M
V ← Get the set of vertices of Mesh M
repeat
    for all v ∈ ∂M do
        d̃ ← Get the vectors that are perpendicular
            to the boundary
        D̃ ← Create representation vector with d̃
    end for
    for all MeshEdge ∈ ∂M do
        EdgeConst ← Assign a random constant
    end for
// To solve the discretization Lz = b, get the
Hessian of z and calculate the eigenvectors
    z ← Solve Lz = b
    for all v ∉ ∂M do
        Hessian of z ← Solve with v's coordinate and
            its two − ring neighbors
        Df ← Get Df by calculating the eigenvectors
        d ← Df
    end for
    Energy ← Calculate the current field energy
        (Equation 3)
until ∇ Energy < ε
DF ← Get all the Df from v ∈ V

    return DF // 4-vector field
```

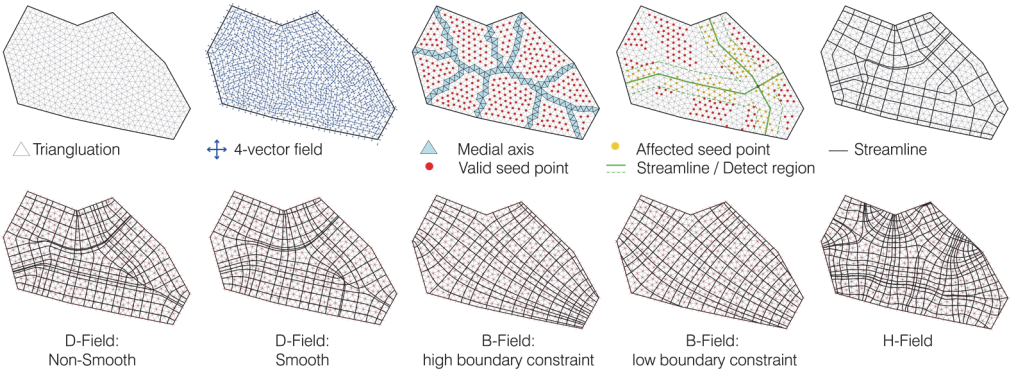


Figure 2
Field initialization and streamline generation for all presented field optimization methods

Field and streamlines visualization

After obtaining the field, a 2D placement algorithm is used to visualize streamlines using a set distance value to ensure reasonable spacing. As shown in Figure 2, the streamlines generated for each field option present distinct geometric characteristics, supporting multiple options for urban fabric generation.

FIELD EVALUATION CRITERIA

To visualize geometric differences relevant to urban fabric generation between the types of vector field, several methods of evaluation are put forward. A first set of criteria is used to evaluate streamline geometry: divergence, continuity, boundary perpendicularity and parallelism. Convexity is applied to evaluate the closed polygonal tiles generated between field streamlines. Finally, singularity analysis identifies distinct breakpoints within the vector field orientation that have strong implications for urban fabric (Figure 4). Each evaluation is presented as a minimization problem, with better results closer to 0.

Divergence

Divergence measures directional change in a streamline at a given point relative to the overall vector field: low divergence streamlines are visibly more parallel to their neighbours (Yang et al., 2013).

Continuity

Continuity measures the extent to which a streamline changes direction over its entire length: a streamline with fewer turns is more continuous (Yang et al., 2013). Each streamline is divided into equal segments and a detect region is set. By checking whether the polyline's next point falls within a range formed by a predetermined angle, this criterion evaluates the number of turns in the streamline.

Boundary Parallelism

Boundary parallelism identifies for each streamline segment, the boundary edge split by its adjacent streamlines for comparison. For interior streamline segments, the algorithm matching a streamline segment and a boundary segment produces less significant results.

Boundary Perpendicularity

Boundary perpendicularity identifies the nearest site boundary edge touched by the streamline as the comparison object, and assesses the degree of perpendicularity between each streamline segment and the designated boundary edge. Similar to parallelism, this selection method becomes less significant further from the boundary.

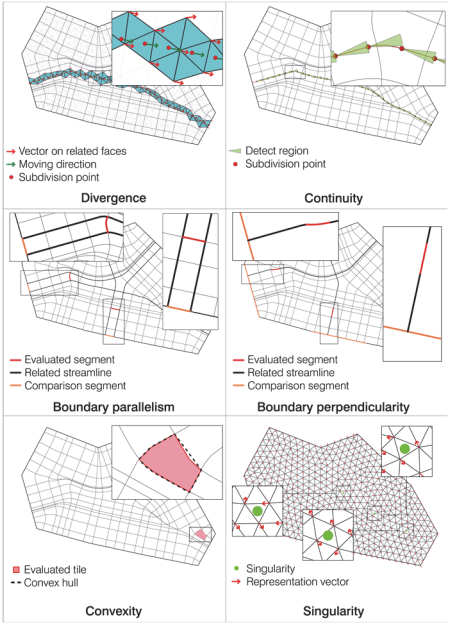
Convexity

Convexity refers to how much a polygonal tile generated between streamlines deviates from its convex hull. This is calculated as the ratio between the perimeter of the tile's convex hull and its perimeter.

Singularity

The interpolation method is used to calculate the vector value of a specific point in the field. If the result of the interpolated vector value is 0 at a certain point, we define it as a singularity. More specifically, if the system of equations shown in Equation 4 (where α, β, γ are the coefficients of interpolation for a triangle) has a solution, there is a singularity point.

$$\begin{cases} \alpha + \beta + \gamma = 1 \\ \alpha D_i + \beta D_j + \gamma D_k = 0 \\ \alpha, \beta, \gamma \geq 0 \end{cases} \quad (4)$$



COMPARATIVE STUDY

To evaluate the streamline networks obtained by the presented methods they are applied to three recent or planned development sites in Singapore.

The existing plans for these sites demonstrate grid-like urban fabrics which are adapted to the site boundary and other geographic constraints to differing degrees. In complement to the existing plans, the streamline results produced by the presented vector field methods illustrate the extensiveness of options possible. In the analysis presented below, tradeoffs between the vector methods are emphasized with the urban implications of the different geometries discussed. Results diagrams are not drawn to scale, as they are used for shape evaluation (Figures 5-9). For the sake of brevity, the H-field is excluded from the presented results.

Results: Low divergence vs. boundary parallelism

A first design-relevant tradeoff in the results is observed between options with very low divergence (B-field) and options with higher levels of boundary parallelism (D-field). As solutions attain lower levels of divergence, the streamlines become closer to parallel with their neighbours: a result that produces regular tiles and approaches becoming a regular grid after many iterations (Figures 5-6). These low divergence fields may provide benefit when used to generate road networks, for example by permitting easier way-finding through consistency of orientation and grid like layout. However, they attain this regularity at the expense of parallelism with the site boundary – and thus produce in more extreme examples rows of triangular tiles and short, poorly connected streamlines at site edges which may be inappropriate for urban networks or development parcels. The D-Field presents an alternative which results in tiles that are closer to rectilinear at the site boundary. When generated with a complex site boundary, however, the D- Field will generate many orientations and may be poorly suited to efficient way-finding if converted to a mobility network.

Figure 4
Upper left: Divergence

Upper Right: Continuity

Middle left: Boundary parallelism

Middle right: Boundary perpendicularity

Bottom left: Convexity

Bottom right: Singularity

Continuity evaluation for the D-Field also shows its streamlines have many more turns per their length in comparison with other options presented in this study. Lack of continuity is a potential way-finding and traffic engineering challenge. As concerns for continuity are scale dependent, the designer would need to choose a strategy of lower divergence or improved boundary parallelism and adjust the iteration number to achieve an acceptable result.

Results: Field Singularities

An extension of the tradeoff observed between low divergence and boundary parallelism is the prevalence of singularities observed within the field, and the resulting wedge-shaped parcels that appear at complex corner conditions within the sites. These singularities are marked by dots (Figure 5). Low-divergence solutions (D-Field with smoothing and B-Field) progressively eliminate singularities especially

Figure 5
Divergence and singularity

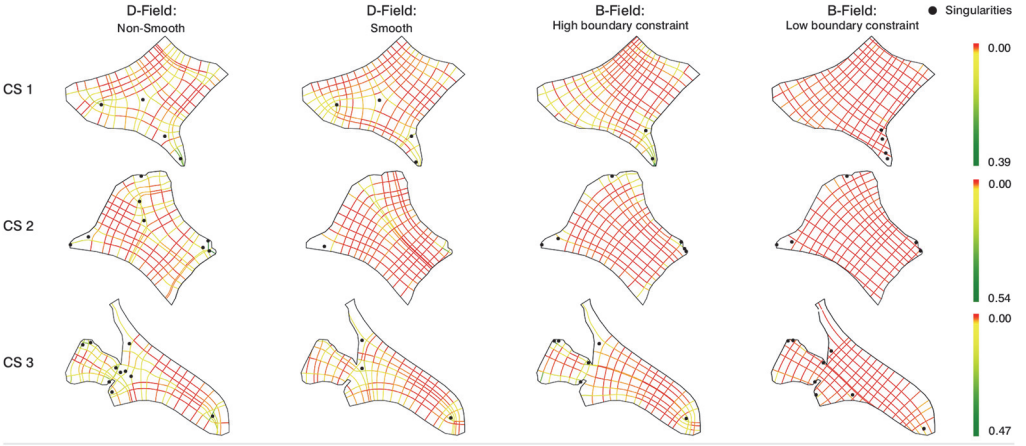
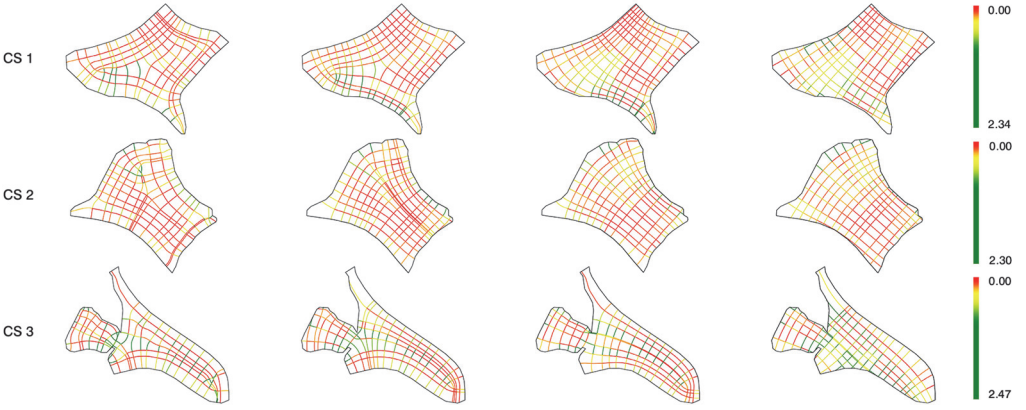


Figure 6
Boundary parallelism



after many iterations, whereas the D-field without smoothing generally has more singularities. The presence of the singularity is not necessarily to be avoided but represents a design choice for the user. The wedge-shaped sites produced by singularities in the D-field are similar to the triangular 'central places' found in the organic plans of medieval European cities, evident for example in the City of London district or the 5th arrondissement in Paris. The existing plan for Singapore's Tengah New Town (Case Study 1, Figure 3) also includes several central wedge-shaped sites.

While smoothing and reduction of boundary weighting will eliminate many singularities, a highly complex site like Jurong Lake District (Case Study 3) with peninsular protrusions will almost invariably generate singularities in the field if site boundaries are used for generative purposes (Figure 5). In the masterplan proposed by KCAP for Case Study 3, the designers have opted for a straight grid without distortion that produces parcel clipping at the boundary (Figure 3).

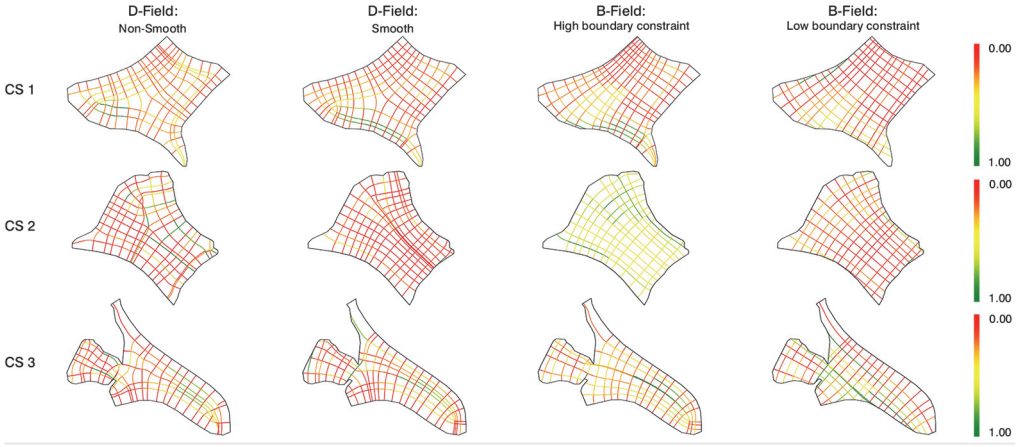


Figure 7
Boundary perpendicularity

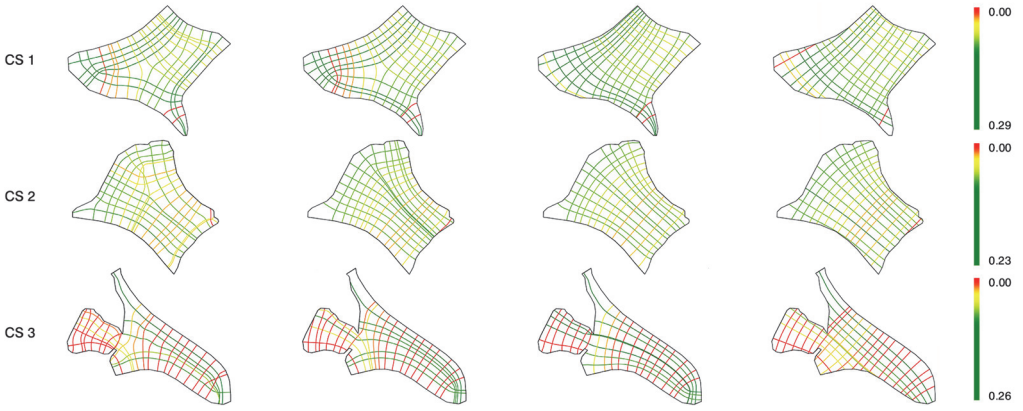


Figure 8
Continuity

Figure 9
Convexity

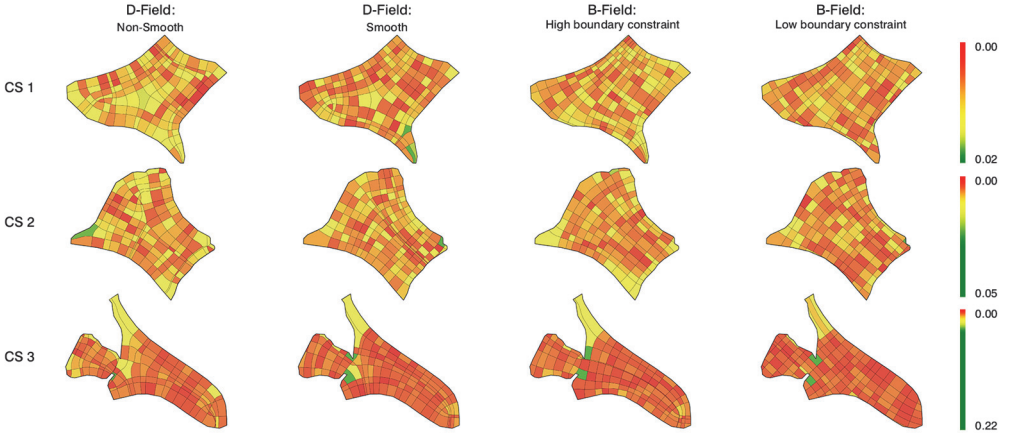


Figure 10
UrbanFab plugin components

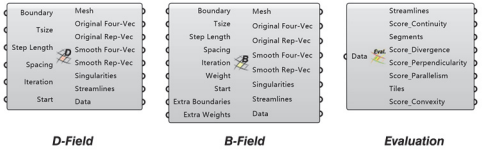
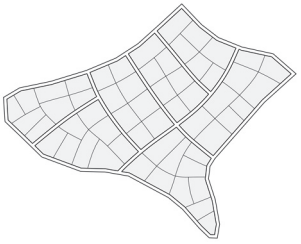


Figure 11
On-going work: application of *UrbanFab* within a generative urban design tool



Results: Tile Convexity

The regularity of the tiles, as measured by convexity, produced by the fields across all four case studies generally improves from left to right in Figure 9, with smoothing and low boundary constraints improving results. This improvement reflects the consolidation of the streamlines into organizations that are closer to regular grids. An extension of this effect is consistency of distribution of convexity. The D-Field shows a wider distribution of convexity results, with local areas of high convexity bordered by areas of low convexity where the field orientation is shifting. This uneven distribution of convexity is most clearly shown in D-Field results for Case Study 1 (Figure 9). The B-Field with low boundary constraint represents the opposite end of this trade-off where variation in convexity is more evenly distributed across all tiles generated.

FURTHER WORK: URBANFAB PLUGIN AND APPLICATIONS

The *UrbanFab* plugin (Figure 10) for Rhino/Grasshopper makes available vector field methods for urban fabric generation to the wider design community. In its first iteration this plugin provides the methods of field and streamline generation, as well as evaluation presented in this paper. With these components the user may make informed decisions on how to implement vector field geometry as part of their urban design/planning workflow.

On-going work by the research team, shown in Figure 11, will apply *UrbanFab* within a generative urban design tool: streamlines are implemented in a breadth-first binary splitting algorithm to create parcels and an associated road network. Splitting is optimized evaluation criteria. This work will allow

the user to customize road topology, parcel orientation and elongation as well as other parameters.

CONCLUSION

The methods presented in this paper make vector field applications for urban fabric generation better available to urban designers and planners. The results of the presented vector fields methods have been shown to be extensive: able to produce a variety of results which a designer can deploy to achieve specific geometries and urban performances. The function and output of the algorithms have been made explicit for design users through the comparative presentation of algorithms and accompanying evaluation of case study results. By releasing the plugin *UrbanFab*, the presented methods are made widely available to support urban design and planning practice and to bring added specificity to discourse on the role of computation and automation in the planning of cities.

While the scope of this paper is devoted to supporting user understanding and access to vector field methods for urban fabric generation, future work will focus on the implementation of algorithmic methods for automatic generation of urban plans with appropriate inputs and breakpoints for extensive and explicit use by designers and stakeholder groups.

ACKNOWLEDGEMENTS

This work was supported by the Singapore URA grant (RS-INDUS-00095: Computational modelling for optimisation of planning and urban design parameters).

REFERENCES

Chen, G., Esch, G., Wonka, P., Müller, P., Zhang, E. (2008). Interactive Procedural Street Modeling. ACM Trans. Graph, vol. 27, no. 3. Available at: https://dl.acm.org/doi/10.1145/1360612.1360702 (Accessed: 22 March 2023)

Chu, H., Li, D., Acuna, D., Kar, A., Shugrina, M., Wei, X., Liu, M., Torralba, A., Fidler,S.(2019). Neural Turtle Graphics for Modeling City Road Layouts. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 4521 -4529, Available at: https://doi.org/10.48550/arXiv.1910.02055 (Accessed: 22 March 2023)

Fang, Z., Yang, T., & Jin, Y. (2020). DeepStreet: A deep learning powered urban street network generation module. arXiv preprint. Available at: https://arxiv.org/ftp/arxiv/papers/2010/2010.04365.pdf (Accessed: 22 March 2023)

Parish, YIH., Müller, P. (2001). Procedural Modeling of Cities. ACM SIGGRAPH. Available at: https://doi.org/10.1145/383259.383292 (Accessed: 22 March 2023)

Vanegas, CA., Kelly, T., Weber, B., Halatsch, J., Aliaga, DG., Müller, P. (2012). Procedural Generation of Parcels in Urban Modeling. Computer Graphics Forum, 31, pp. 681–690. Available at: https://doi.org/10.1111/j.1467-8659.2012.03047.x (Accessed: 22 March 2023)

Vaxman, A., Campen, M., Diamanti, O., Panozzo, D., Bommes, D., Hildebrandt, K., Ben-Chen, M. (2016). Directional Field Synthesis, Design, and Processing. Computer Graphics Forum, vol. 35, no. 2, pp. 545-572. Available at: https://doi.org/10.1111/cgf.12864 Accessed: 22 March 2023)

Yang, Y., Wang, J., Vouga, E., Wonka, P. (2013). Urban Pattern: Layout Design by Hierarchical Domain Splitting. ACM Trans. Graph. Available at: https://doi.org/10.1145/2508363.2508405 (Accessed: 22 March 2023)

Zhang,Q., Li B., Mo Y., Chen Y., Tang P.(2022). A WEB-BASED INTERACTIVE TOOL FOR URBAN FABRIC GENERATION: A Case Study of Chinese Rural Context. CAADRIA. Available at: https://caadria2022.org/wp-content/uploads/2022/04/291-1.pdf (Accessed: 22 March 2023)