Eurographics Symposium on Rendering 2013
Nicolas Holzschuch and Szymon Rusinkiewicz
(Guest Editors)

Volume 32 (2013), Number 4

Supplemental Material For "A Shape-Aware Model for Discrete Texture Synthesis"

Pierre-Edouard Landes[1,3], Bruno Galerne[2,3] and Thomas Hurtut[1,3]

[1]LIPADE [2]Laboratoire MAP5 (UMR CNRS 8145) [3]Université Paris Descartes, Sorbonne Paris Cité

1. Conditional Intensities

Our shape process model is fully described by its density function given by the paper's Eq.(4). However, in both the parameter fitting procedure and the model sampling algorithm, this intensity f only appears through its associated (Papangelou) conditional intensity λ defined by

$$\lambda((u,s);\mathbf{x}) = \begin{cases} \frac{f(\mathbf{x})}{f(\mathbf{x}\setminus(u,s))} & \text{if } (u,s) \text{ is one of the elements of } \mathbf{x}, \\ \frac{f(\mathbf{x}\cup(u,s))}{f(\mathbf{x})} & \text{otherwise.} \end{cases} \tag{A}$$

Evaluating this conditional intensity λ efficiently is thus a critical requirement. One should stress that the ratios $\frac{f(\mathbf{x}\setminus(u,s))}{f(\mathbf{x})}$ and $\frac{f(\mathbf{x}\cup(u,s))}{f(\mathbf{x})}$ are of course not computed directly but after simplification of the many common terms that are on both sides of the fractions. More explicitly, if $(u,s) \notin \mathbf{x}$

$$\lambda((u,s);\mathbf{x}) = \beta \prod_{\mathbf{e}_i} \left(\prod_{[\delta_{i,k},\delta_{i,k+1})} \gamma_{i,k}^{n_{i,k}(u,s)} \right), \tag{B}$$

where $n_{i,k}(u,s)$ is the number of elements $x_n \in \mathbf{x}$ such that the pair $((u,s),x_n)$ is $\mathbf{e}_i$-oriented and has its distance $d_{\text{shp}}((u,s),x_m)$ lying in the interval $[\delta_{i,k},\delta_{i,k+1})$. In particular, all elements $x_n \in \mathbf{x}$ such that $d_{\text{shp}}(x_n,x_m) > \max_i \delta_{i,K}$ do not intervene.

Numerically, these ratios are computed by scanning successively each element lying in the spatial neighborhood of (u,s), computing its orientation and distance to (u,s), and multiplying by the corresponding $\gamma_{i,k}$. Hence, evaluating the conditional intensity $\lambda((u,s);\mathbf{x})$ is only linearly proportional to the number of elements surrounding the element (u,s).

2. Log-Pseudolikelihood Maximization

The parameter fitting algorithm we use tends to find the parameters that maximize the log-pseudolikelihood of the model evaluated on the input discrete texture $\mathbf{z}$. Following [BT00], it consists in two main steps: (i) Approximate the log-pseudolikelihood with a quadrature method, (ii) Maximize this approximate log-pseudolikelihood using a Poisson regression.

The log-pseudolikelihood in the paper's Equation (5) can be rewritten as

$$\log \text{PL} = \sum_{(u_n,s_n)\in\mathbf{z}} \log \lambda((u_n,s_n);\mathbf{z}) - \iint_{\mathcal{P}\times\mathcal{S}} \lambda((u,s);\mathbf{z}) \mathrm{d}u\,\mathrm{d}s$$

where conditional intensity $\lambda((u,s);\mathbf{z})$ is given by Eq. (A). Using Eq. (B), the discrete sum in the above equation can be computed explicitly. Dealing with the double integral is however more involving. First, note that in our context the set of possible shapes $\mathcal{S}$ is discrete, and each shape in the exemplar is considered to be equiprobable. Hence the ds measure is the uniform distribution on $\mathcal{S}$, that is $\Pr(s = s_n) = 1/N$ where N is the total number of elements in the exemplar $\mathbf{z}$. Integrating over the continuous domain $\mathcal{P}$ necessarily involves a quadrature method. As proposed by [BT00], this quadrature method is obtained using a discrete regular cell grid to which the positions $\{u_n\}$ of the elements of $\mathbf{z}$ are added. The resulting set of points is denoted by $\mathcal{G} = (u_j)$, and each of these points has a corresponding weight w_j that compensates for the irregularity of the grid. In the end the double integral is approximated by

$$\iint_{\mathcal{P}\times\mathcal{S}} \lambda((u,s);\mathbf{z}) \mathrm{d}u\,\mathrm{d}s \approx \sum_{u_j\in\mathcal{G}} \sum_{s\in\mathcal{S}} \lambda((u_j,s);\mathbf{z}) \frac{w_j}{N}.$$

Since all the elements $(u_n,s_n) \in \mathbf{z}$ correspond to some couple $(u_j,s) \in \mathcal{G}\times\mathcal{S}$, the resulting approximation of $\log \text{PL}$ can be written in the factored form

$$\log \text{PL} \approx \sum_{u_j\in\mathcal{G}} \sum_{s\in\mathcal{S}} \left(y_{j,s} \log \lambda((u_j,s);\mathbf{z}) - \lambda((u_j,s);\mathbf{z}) \right) \frac{w_j}{N},$$

where $y_{j,s} = \frac{N}{w_j} \mathbb{1}\left((u_j,s) \in \mathbf{z}\right)$. Maximizing this approximation corresponds to fit a generalized linear model [Dob02] whose optimal parameters to be estimated are, when f is given by the paper's Equation (4), the vector $(\log\beta, (\log\gamma_{i,k}))$.

Implementation Following classical Poisson regression solvers, we maximize the above approximation of the log-pseudolikelihood using Broyden-Fletcher-Goldfarb-Shanno method. Some parameters $\gamma_{i,k}$ are *unobserved* in the sense that no pair of exemplar elements has its relative direction in the i-th quadrant and their distance within $[\delta_{i,k}, \delta_{i,k+1})$ (this is inevitably the case of the parameters $\gamma_{i,0}$ due to the hard-core condition). As suggested in [BT00], these unobserved parameters need to be set to zero (their optimal value) and removed from the regression beforehand in order to keep a tractable Poisson regression.

3. Metropolis-Hastings Sampling

In this section we present the Metropolis-Hastings sampling algorithm we implemented to generate all our results. The algorithm relies only on birth and death proposals and is known to converge towards our target distribution [GM94].

The algorithm to simulate an output discrete texture $\mathbf{x}$ in a toroidal window W is as follows (below $\#\mathbf{x}$ and $|W|$ denotes respectively the number of elements of $\mathbf{x}$ and the area or volume of W):

• Initialize $\mathbf{x}$ as an empty set.

• For iteration $t = 1$ to $t = T$, apply equiprobably one of the following perturbation:

⋆ **Birth:**

1. Draw uniformly a candidate location u in the output window.

2. Draw uniformly a candidate shape s among the set of shapes of the input $\mathbf{z}$.

3. Compute the acceptance rate $R_b = \lambda((u,s);\mathbf{x})\frac{|W|}{\#\mathbf{x}+1}$ for adding (u,s) to $\mathbf{x}$.

4. Add the element (u,s) to $\mathbf{x}$ with probability $\min(R_b, 1)$.

⋆ **Death:**

1. Select uniformly an element $(u,s) \in \mathbf{x}$ (if $\mathbf{x}$ is empty, do nothing).

2. Compute the acceptance rate $R_d = \frac{1}{\lambda((u,s);\mathbf{x})}\frac{\#\mathbf{x}}{|W|}$ for removing (u,s) from $\mathbf{x}$.

3. Remove (u,s) from $\mathbf{x}$ with probability $\min(R_d, 1)$.

Note that each birth or death iteration has in average a constant cost. Indeed it consists mainly in the evaluation of the conditional intensity $\lambda((u,s);\mathbf{x})$ which only depends on the number of elements in the spatial neighborhood of (u,s) (see Section 1 of the supplemental material). As reported in the paper's Table 1, in all our experiments we used either $T = 10,000$ or $T = 50,000$ for the number of iterations.

4. Additional Editing Results

Figure 1 exposes three more editing results: boundary handling with uniform texturing (a), intensity modulation allowing stippling effects (b), and a parameter transfer example (c). In the latter, the parameters are first fitted on the *snakes* input, giving the interaction function displayed at the bottom. We then use this interaction function parameters on a model sampling that uses the set of ant elements (taken from the input (e) from the paper comparative figure) instead of the snakes. Note that this parameter transfer is not equivalent to a simple element interchange, which would lead to an output with different statistics since the shapes are different. Here, the output shares the same interaction statistics as the snakes input.

5. Intermediary Representations and Fitting Results

This section includes figures giving more insight on the computational internals of our discrete texture synthesis:

- Figure 2 reveals the proxy geometries that our method automatically computes for the exemplars displayed in the comparative figure of our submitted article;
- Figure 3 displays the directional density functions of the pairwise element interactions as well as the fitted interaction function for each texture exemplar.

6. Details on [IMIM08] and [MWT11]

For the sake of fair comparison, we further detail in this section the results we obtained with our implementations of [IMIM08]'s and [MWT11]'s techniques:

- Figure 4 displays the entire output textures and associated triangulations synthesized by [IMIM08]'s seeding algorithm;
- Figure 5 exposes the element sampling scheme we used for describing the element shapes of the exemplar textures to our implementation of [MWT11]'s method;
- Figure 6 shows the patch-based initialization of [MWT11]'s Expectation-Maximization procedure, and the three element reconstruction methods mentioned in the paper.

References

[BT00] BADDELEY A., TURNER R.: Practical maximum pseudolikelihood for spatial point patterns. *Australian & New Zealand Journal of Statistics 42*, 3 (2000), 283–322. 1, 2

[Dob02] DOBSON A.: *An introduction to generalized linear models*. CRC press, 2002. 1

[GM94] GEYER C., MØLLER J.: Simulation procedures and likelihood inference for spatial point processes. *Scandinavian Journal of Statistics* (1994), 359–373. 2

[IMIM08] IJIRI T., MECH R., IGARASHI T., MILLER G.: An Example-based Procedural System for Element Arrangement. In *Eurographics* (2008), vol. 27, pp. 429–436. 2

[MWT11] MA C., WEI L.-Y., TONG X.: Discrete element textures. In *SIGGRAPH* (2011), vol. 30, pp. 1–10. 2

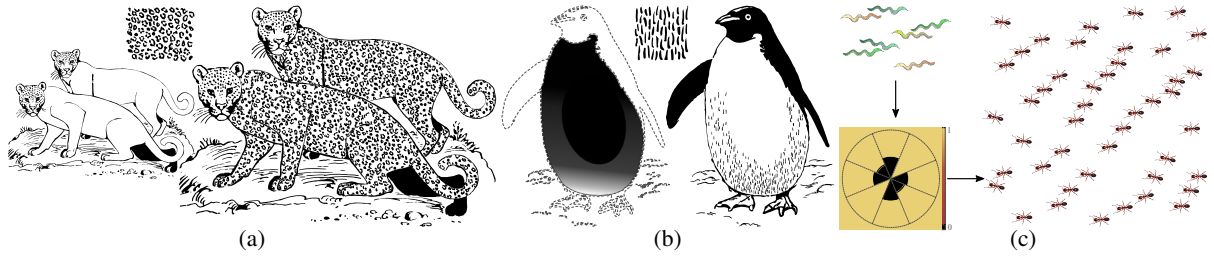

Figure 1: ***Complex Texture Editing Based on Our Shape Process.*** *We demonstrate here the versatility of the editing that our texture modeling provides. Not only is it possible to have our output textures account for possibly complex domain boundaries during their synthesis (a), but we can also modulate (b) or even transfer (c) the values of the parameters fitted to our exemplars.*

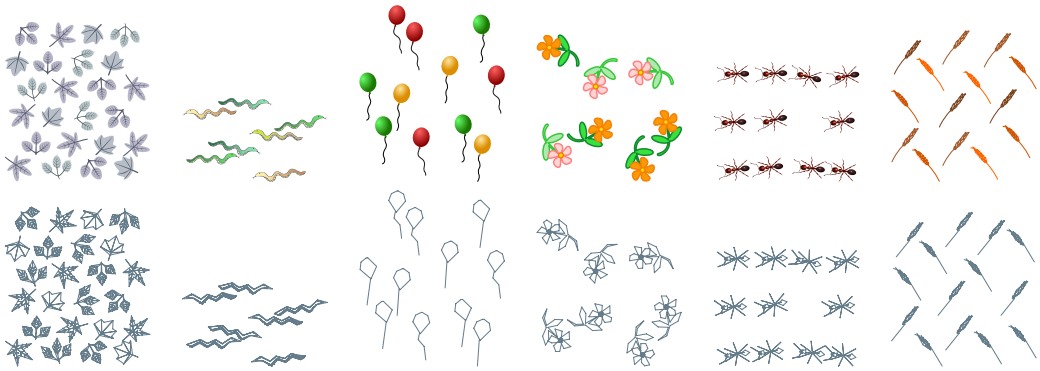

Figure 2: ***2D Proxy.*** *For each of the examplar from the paper comparative figure, we present here the proxy geometry being computed and used by our model. These proxy show how a geometric simplification is achieved, yet preserving a rather faithful and detailed shape geometry.*

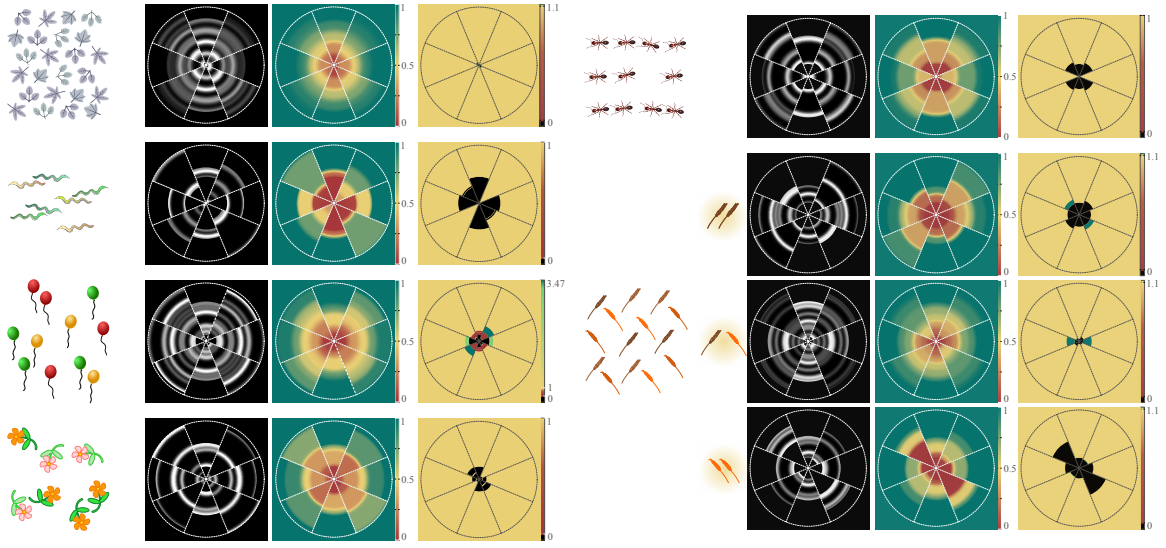

Figure 3: ***2D Distance Wheels.*** *For each input used in the paper, this figure shows, from left to right wheels, the directional Probability Density Function of observed pairwise distances (grayscale), the directional Cumulative Density Function (colorscale), and the fitted parameter values obtained by likelihood maximization. For the ears of wheat examplar, as mentioned in the paper, we make use of two element categories: brown* vs *yellow wheat ears. We therefore show three series of wheels, two for intra-category interactions (top and bottom), and one serie for the inter-category interactions (middle).*

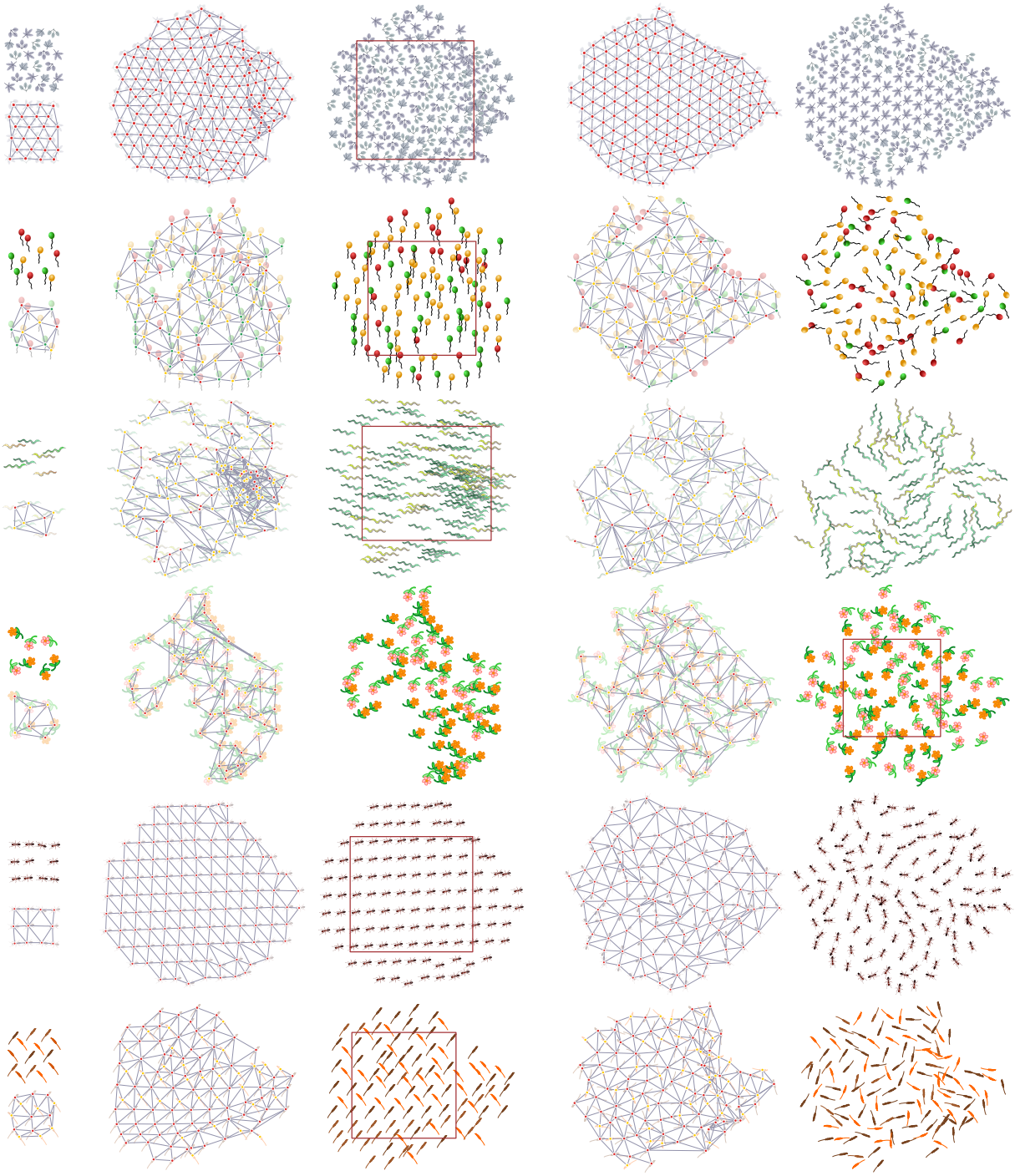

Figure 4: ***Extended Outputs for Ijiri* et al..** *Since Ijiri* et al.*'s paper is a growing greedy algorithm, we show here the complete outputs we generated to fill the rectangular window we used to compare all methods. The input (a) is shown along with its Delaunay triangulation on which the algorithm rely to build neighborhoods. The dot colors on the triangulation indicate the element ids. The columns (b) and (c) show both the output triangulations and pinned element arrangements without allowing elements to rotate, whereas the columns (d) and (e) show the triangulations and arrangements with allowed element rotations. The dotted rectangular windows represent the best output crops we chose for the comparaison Figure in the paper. In general, a limited search space strongly impacts synthesis quality, but for each presented result we chose the best outcome between a rotation-free and a rotation-fixed neighborhood matching.*

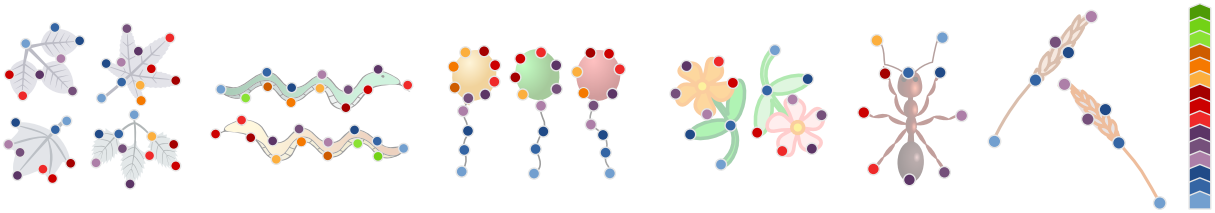

Figure 5: ***Sampled Inputs for Ma* et al..** *This figure shows the sample sets we used in our pqper comparative figure for each exemplar element. The sample color indicate the sample ids, showing in which order we did sample the shape borders, basically following the outer contour.*

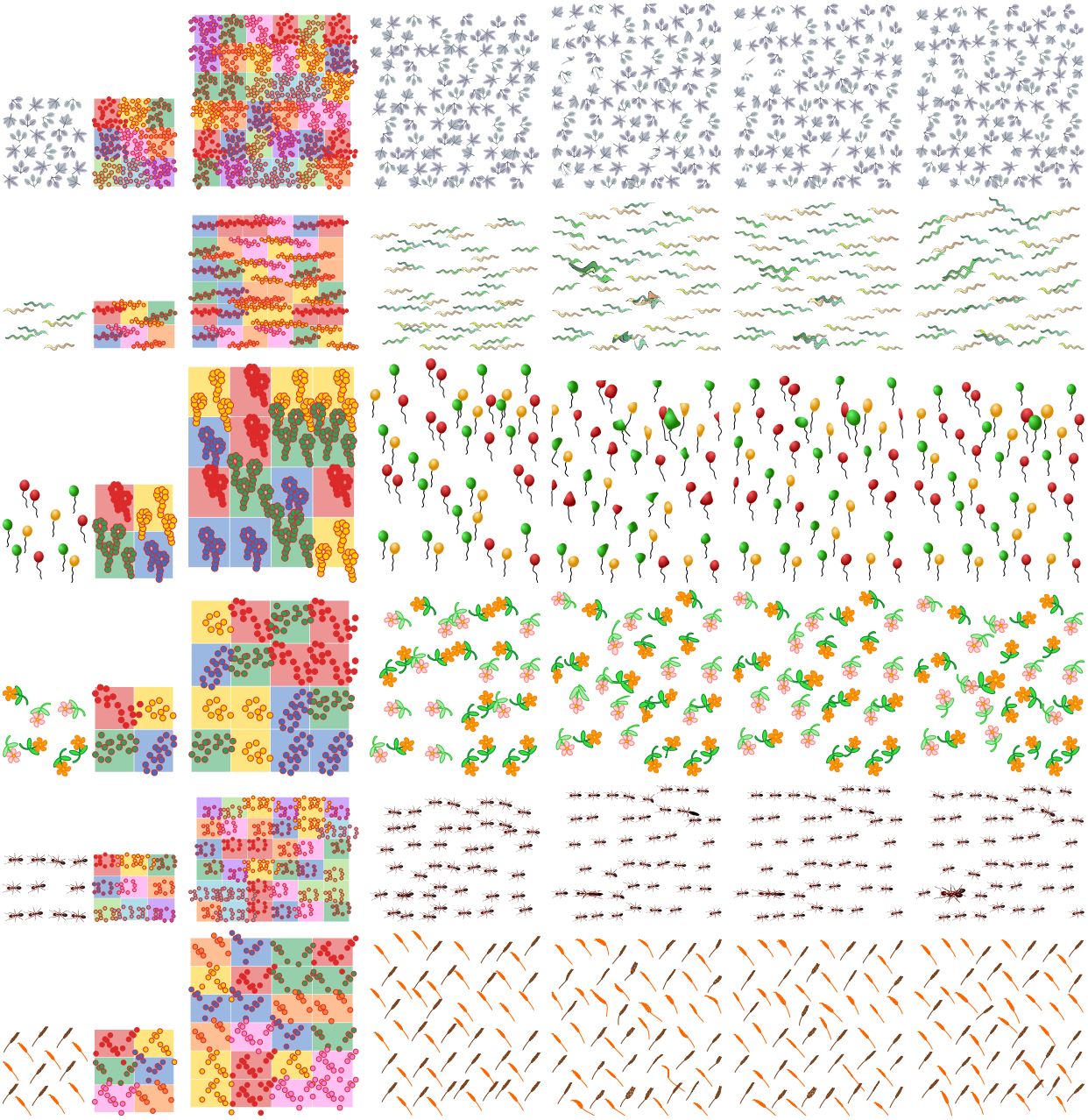

Figure 6: ***Ma* et al. *Output Details.*** *This figure gives more details about Ma* et al. *synthesized outputs used in the paper comparative figure. Exemplars (a) and their underlying patches (b) are first shown. The element dots color corresponds to the underlying patch it belongs to. As mentionned by Ma* et al.*, due to their discrete nature, elements often partially straddle over some surrounding patches, this provokes unwanted interlockings in the output patch initialization (c) and the resulting element initialization given to the EM algorithm (d). The last three columns show the final outputs (after* 20 *EM iterations), reconstructed using the three tradeoffs presented in our submitted paper: An interpolating and approximating thin plate spline in (e) and (f) respectively; and the best fitted similarity mapping (g). In the paper we chose the latter as giving the mean best results.*