nature nanotechnology

ARTICLES

PUBLISHED ONLINE: 28 NOVEMBER 2016 | DOI: 10.1038/NNANO.2016.256

Programmable disorder in random DNA tilings

Grigory Tikhomirov[1†], Philip Petersen[2†] and Lulu Qian[1,3*]

Scaling up the complexity and diversity of synthetic molecular structures will require strategies that exploit the inherent stochasticity of molecular systems in a controlled fashion. Here we demonstrate a framework for programming random DNA tilings and show how to control the properties of global patterns through simple, local rules. We constructed three general forms of planar network—random loops, mazes and trees—on the surface of self-assembled DNA origami arrays on the micrometre scale with nanometre resolution. Using simple molecular building blocks and robust experimental conditions, we demonstrate control of a wide range of properties of the random networks, including the branching rules, the growth directions, the proximity between adjacent networks and the size distribution. Much as combinatorial approaches for generating random one-dimensional chains of polymers have been used to revolutionize chemical synthesis and the selection of functional nucleic acids, our strategy extends these principles to random two-dimensional networks of molecules and creates new opportunities for fabricating more complex molecular devices that are organized by DNA nanostructures.

Understanding the function of stochasticity in biological systems has been a long-standing challenge[1–3]. Two main principles have been recognized. First, stochastic algorithms allow simpler genetic programmes compared with deterministic algorithms, as seen in randomly expressed genes in olfactory receptor neurons[4], for example. Second, stochasticity can give rise to combinatorial diversity, such as that seen in the transmembrane domain proteins essential for neural wiring[5]. Taking inspiration from the stochasticity in biological systems and applying the principles in engineered cellular and molecular systems has enabled substantial technological advances and deepened our understanding of natural algorithms. One example is the 'brainbow' technique for visualizing complex biological neural networks[6] and another is the original demonstration of DNA molecules performing non-trivial computation[7]. In both examples, stochasticity was the key to creating combinatorial diversity and to providing simple algorithms for complex tasks.

Here we demonstrate that besides molecular information processing, in structural DNA nanotechnology, stochastic algorithms can also provide simple solutions for creating complex patterns in molecular systems that have combinatorial diversity and programmable features. We refer to the random-yet-controlled properties of these molecular patterns as programmable disorder.

From the first unnatural DNA nanostructure[8] to DNA origami[9], rationally designed DNA nanostructures have been increasingly used as scaffolds to organize diverse molecules such as proteins[10], organic dyes[11], metal nanoparticles[12,13] and polymers[14]. Because of the high programmability and spatial precision of the DNA nanostructures, they hold promise for arranging functional components into molecular devices and for advancing a variety of technologies, including molecular electronics, plasmonics and photonics[15]. However, to move from functional components to complex networks of molecular devices that can actually carry out sophisticated tasks, it is necessary to assemble DNA nanostructures at a much larger scale, using DNA tilings.

There are so far two strategies for constructing large-scale arrays of DNA tiles: periodic arrays and algorithmic arrays. Periodic arrays are easier to construct and have been demonstrated with both small DNA tiles[16–20] and DNA origami tiles[21–23], but the pattern complexity is limited. Algorithmic arrays can create complex global patterns by using DNA tiles that encode a specific set of rules defined by a cellular automaton[24–26]. Each tile can interact with other tiles with two inputs and two outputs. The growth has to start from a seed and continue to follow the input-to-output direction. The correct growth of an array relies on competition among multiple types of tiles. It is critical that tiles with one matching input do not attach successfully but tiles with two matching inputs do. Because of these constraints, algorithmic arrays require delicate experimental conditions: the desired growth only takes place at a very specific temperature, and any change in sequences or modifications to the DNA strands could disrupt the growth. Although algorithmic arrays have been constructed with small DNA tiles, arrays of DNA origami tiles have not yet been demonstrated.

There are several distinct advantages of origami arrays compared with small DNA tile arrays. Because of the larger size, origami tiles have more programmable interactions, including multiple sticky ends[21] and geometries[27]. Geometry could also give rise to more sophisticated control of structural reconfiguration[28]. Most importantly, origami tiles are large enough to accommodate complex internal patterns for organizing molecules, and thus arrays of origami tiles can be used to construct networks of sophisticated molecular components. Here, internal patterns will play yet another role.

To scale up the complexity and diversity of origami arrays, we present a strategy using random DNA tilings, which are as easy to construct as periodic arrays but with substantially more control over the complexity of the global patterns. The strategy is guided by stochastic Truchet tiling[29,30]. A Truchet tile has a rotationally asymmetric pattern that is designed to continue into neighbouring tiles. Complex patterns can be easily generated by allowing a random orientation of a tile at each location in an array. Generalizing the theory of classic Truchet tiling, we developed a framework for creating programmable disorder in random DNA tilings (Fig. 1). We chose three example tasks: creating random loops (that is, closed lines with no branches), mazes (lines with loops and branches) and trees (lines with branches but not loops), to demonstrate how the properties of global patterns can be controlled by local rules. Loop[31], maze[32–34] and tree[35–37] structures widely exist in natural systems across multiple scales. We defined a set of rules for programming the pattern on a

[1]Department of Bioengineering, California Institute of Technology, Pasadena, California 91125, USA. [2]Department of Biology, California Institute of Technology, Pasadena, California 91125, USA. [3]Department of Computer Science, California Institute of Technology, Pasadena, California 91125, USA. †These authors contributed equally to this work. *e-mail: luluqian@caltech.edu

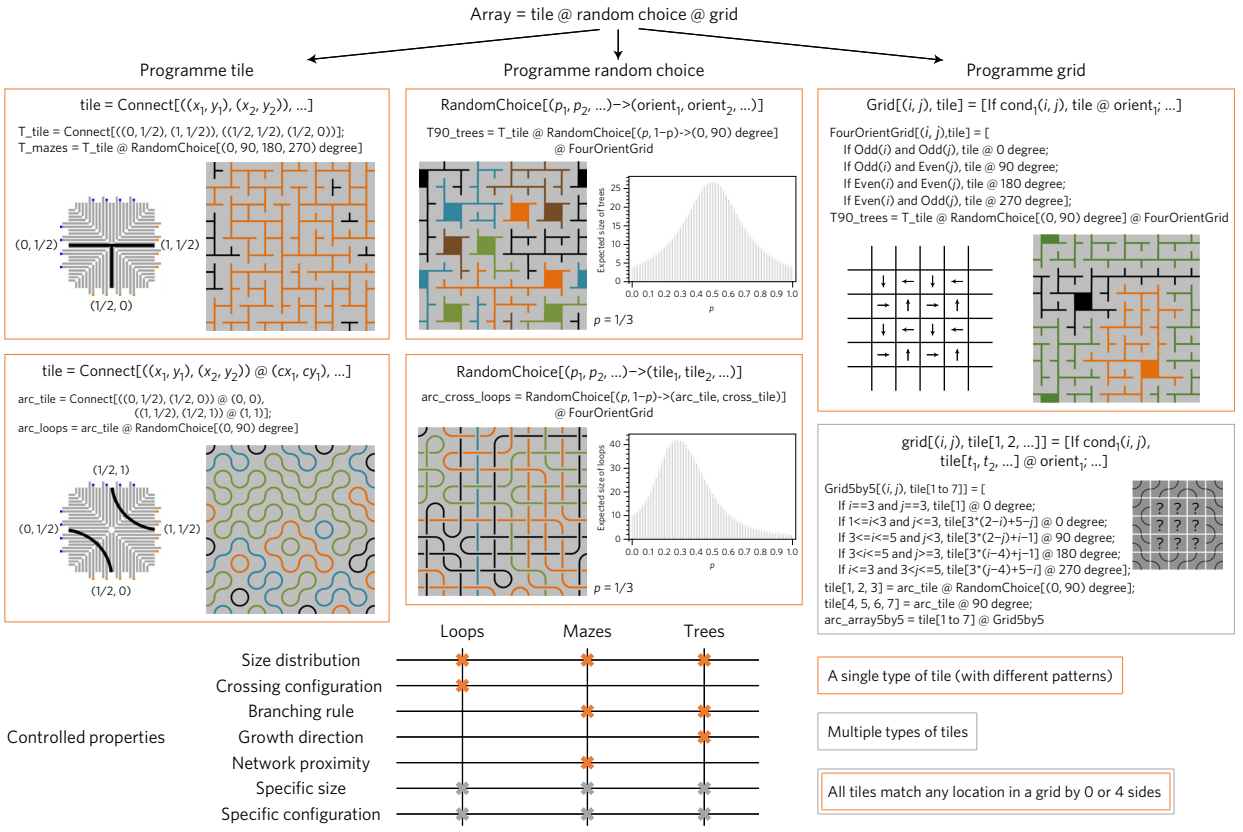

Figure 1 | Summary of programmable disorder in random DNA tilings. The rules for programming random DNA tilings (the syntax is explained in Methods and the automated design steps are illustrated in Supplementary Fig. 1), the properties of patterns that can be controlled in self-assembled DNA origami arrays and the properties of the DNA origami tiles used in experimental implementations. The sizes of loops, mazes and trees are defined on the basis of the area that they cover (see Methods). The orange or grey box around each rule corresponds to the properties of the DNA origami tiles required for implementing the rule, which also corresponds to the properties of patterns that can be controlled using the rules.

tile, the random choice among different tiles and the different choices at specific locations on a grid. Many of these rules can be implemented using DNA origami tiles with just a single type of edge design and different surface patterns. In the process of tiles self-assembling into arrays, four errors (that is, four unmatched sides) are required for an incorrect tile attachment to be locked in place, which ensures that the implementation is experimentally robust with a high barrier for errors.

Implementation of Truchet arrays

A classic Truchet tile (Fig. 2a), which we refer to as the arc tile, has two possible orientations created by any 90° rotation. In an array of arc tiles with a random choice of the two orientations at each location, the lines will continue through tiles, either becoming loops of varying sizes or exiting from an edge of the array. Interestingly, the arc tile is related to a fully packed loop model, which has been studied extensively in theoretical physics, with applications to the quantum phase transitions in magnets and to Anderson localization problems[38–40].

To create Truchet arrays, we designed a square DNA origami tile with maximally continuous surface area for implementing a wide range of patterns on a single tile (Fig. 2b). The tile is composed of four isosceles triangles (Supplementary Note 2.1 and Supplementary Fig. 2) and has a four-fold rotational symmetry, such that it can adopt an unbiased choice of any of the four orientations when self-assembled into an array. Short single-stranded domains were used in staples that bridge the seams between the four triangles (Fig. 2b, top inset), and in scaffolds at the locations where the scaffold makes a turn from one helix to another near the interior edge of the triangles (Fig. 2b, bottom inset). To calculate lengths of these single-stranded domains that should not strain the DNA structure, we developed a three-dimensional (3D) model at the level of each base pair (Fig. 2c; Supplementary Fig. 3 and Supplementary Note 2.2). We explored two designs of bridge staples and chose the one that provided more structural stability (Supplementary Figs 4–6). To verify the formation of tiles as monomers, we modified the edge staples to minimize the interactions between tiles (Supplementary Figs 7–9). High-resolution atomic force microscope (AFM) images confirmed that the square tile was created with over 94% yield, clearly showing four triangles joined together (Fig. 2d; Supplementary Figs 10 and 11).

To encourage the formation of large 2D DNA origami arrays in solution, we tuned tile–tile interactions by programming the eleven staples along each edge. It is desirable to have a weak binding energy at the lower temperatures at which the tiles are structurally stable. The weak binding energy allows the tiles to rearrange themselves and avoid kinetic traps at undesired configurations during self-assembly. It is also desirable to have sufficiently high specificity, such that perfect alignment between tiles is much preferred over any misalignment that would encourage aggregation rather than crystal growth. DNA blunt-end stacking, which is the interaction between the ends of two double helices, has been shown to provide a weak binding energy for programming the interactions between 2D and 3D DNA origami structures[27,28] with specificity encoded in the geometry. We used both geometry and very short sticky ends to encode specificity, and explored several edge designs to achieve a sufficiently weak binding energy and a sufficiently high specificity simultaneously (Supplementary Figs 12–14).

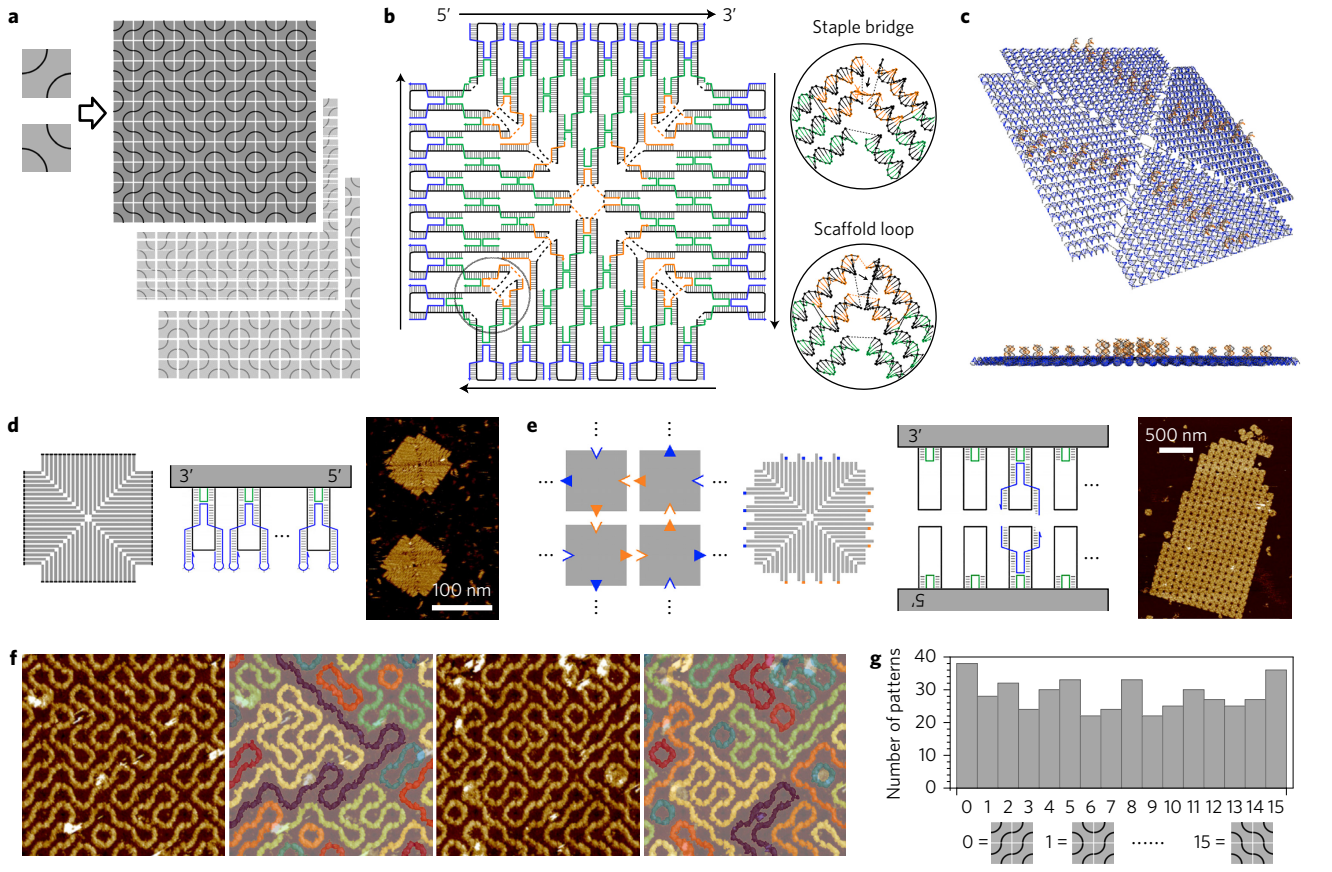

Figure 2 | Implementation of Truchet arrays. a, A classic Truchet tile design with two arcs. Example arrays are shown with a random choice of the two possible tile orientations at each location. **b**, Design of a symmetric square DNA origami tile composed of four identical triangles. A mini version (12 × 12 helices) of the actual origami tile (22 × 22 helices) is shown with a circular scaffold (black), edge staples (blue) that can be programmed for specific binding between one tile and another, interior staples (green) that fold the scaffold into four triangles and bridge staples (orange) that zip up the seams between the triangles. The orientation of each edge is defined as the orientation of the scaffold path, and labelled as a line pointing from the 5′ end to the 3′ end. The single-stranded domains of the staple bridges and scaffold loops are shown as dotted lines, with one example of each shown as bases in the circular insets. **c**, 3D model of the square DNA origami tile (full size, 22 × 22 helices) with double-stranded staple extensions in a pattern of two arcs. With this model, we calculated the lengths of all single-stranded domains that should not strain the DNA structure. **d**, Tile abstraction (left), edge design (middle) and AFM image (right) of square DNA origami tiles as monomers. All edge staples were modified with double hairpins to prevent interactions between tiles. **e**, Array abstraction, tile abstraction, edge design and AFM image of unbounded square DNA origami arrays (left to right). The edge design consisted of four edge staples, each having a stacking bond and a two-nucleotide sticky end. Other locations on the edge were left as scaffold loops. Most of the unbounded arrays were created with an overnight anneal, and a few of the largest ones were annealed for two days to a week (Methods and Supplementary Fig. 25). **f**, AFM images of random Truchet arrays of DNA origami tiles. Each original image (left) was coloured to show continuous paths on the double-stranded staple extensions (right). The method of colouring is shown in Supplementary Fig. 28. All images here and in Figs 3–5 show an area of 10 × 10 tiles, which is approximately 880 × 880 nm^2. **g**, Analysis of arc orientations in random Truchet arrays. The numbers of all 16 possible patterns in a 2 × 2 neighbourhood of tiles were calculated on the basis of a single crystal domain in a 2.8 μm × 2.8 μm AFM image (Supplementary Fig. 30). With a total of 456 neighbourhoods, each of the 16 patterns appeared 28.5 ± 4.9 times (6.25% ± 1.07%).

We observed that if the binding energy was too strong or if the tiles had curvature, the assemblies tended to form tubes instead of 2D arrays (Supplementary Figs 15–17 and 22). Similar to the 'corrugated' design introduced in self-assembled arrays of cross-shaped DNA tiles[17,18,21], we used a global self-correction mechanism that forced the tiles to attach to each other with a 90° rotation and prevented the accumulation of curvature during self-assembly (Supplementary Figs 18 and 19). With the edge design shown in Fig. 2e, we demonstrated robust crystallization that is more reliable and consistently produces larger crystals than previous techniques[21]. Unbounded arrays up to 16 μm × 16 μm, consisting of tens of thousands of tiles, were self-assembled (Supplementary Figs 20 and 21) with a melting temperature close to 35°C (Supplementary Figs 23 and 24).

To create the classic Truchet arrays at the nanometre scale, we constructed square origami tiles with double-stranded staple extensions in the pattern of two arcs (Supplementary Figs 26 and 27). Because the edge design defined a fixed relative orientation of tiles, two types of tiles were annealed separately with distinct sets of extended staples for the two orientations of the arc pattern, and then mixed together to self-assemble into arrays. Arrays with a variety of fully packed loops were created (Fig. 2f; Supplementary Fig. 29), and the choice of arc orientation at all locations was close to random (Fig. 2g; Supplementary Fig. 30).

The use of multiple tiles for multiple pattern orientations is an acceptable but inefficient approach for implementing Truchet arrays. We further designed extended edges to remove the sequence dependence between edge staples and the scaffold, and created random arc arrays with a single tile that was truly rotatable with identical sequences on four edges (Supplementary Fig. 31). However, extended edges increased the distance between staple extensions on adjacent tiles and resulted in less continuous patterns.

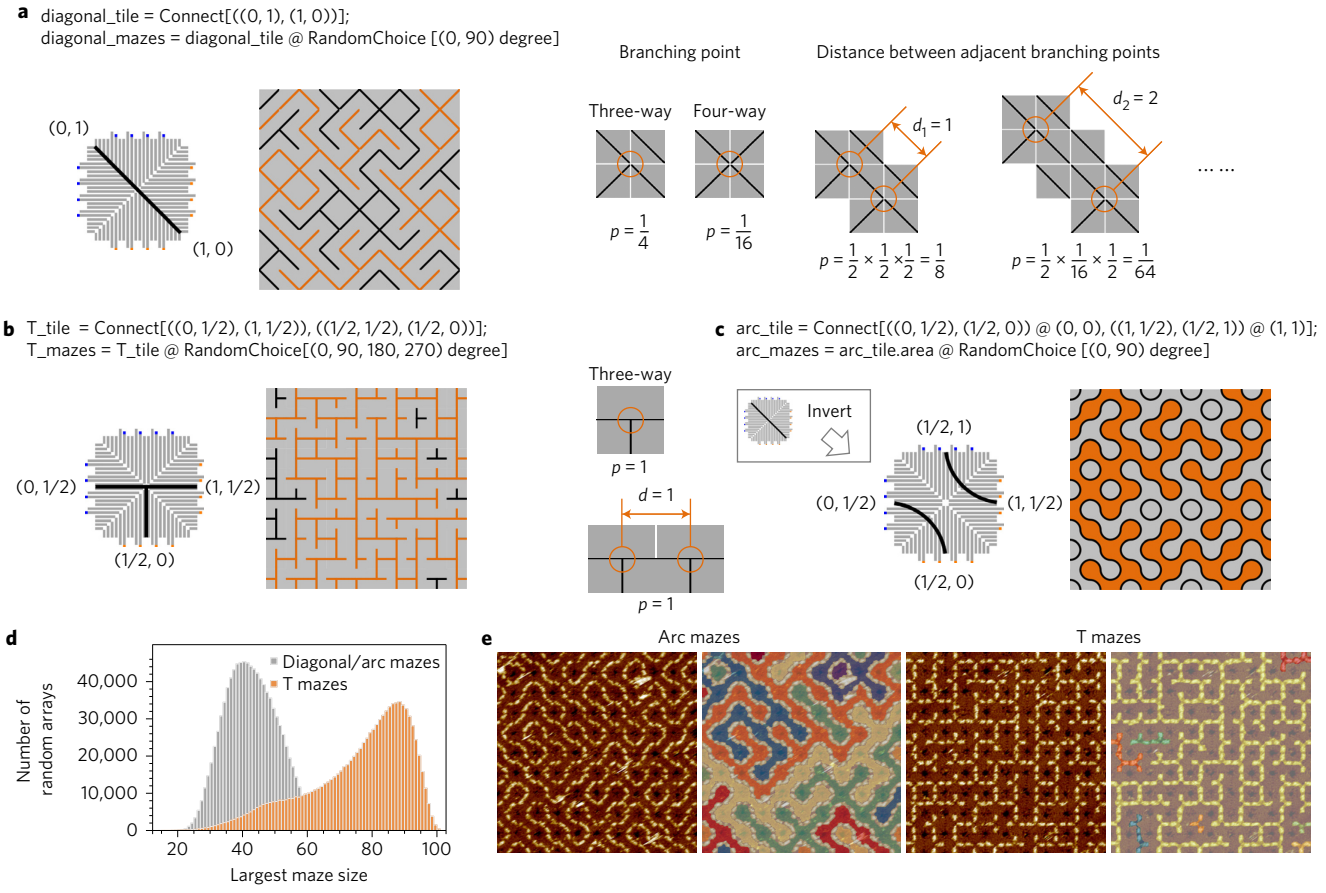

Figure 3 | Programming the tile. a,**b**, Diagonal tile (**a**) and T tile (**b**) designs that both generate random maze-like patterns but with different branching rules (the syntax is explained in Methods). **c**, As an inverse of the diagonal tile, the arc tile can be used to create area mazes instead of line mazes. The largest maze in each example random array is highlighted in orange. **d**, Histogram of the largest maze size on random 10 × 10 arrays, generated from numerical simulations with one million independent trials. The size of a diagonal maze or a T maze is determined as the number of tiles in the maze. The size of an arc maze is determined as the size of its equivalent diagonal maze. **e**, AFM images of random mazes on arc and T tile arrays. Each original AFM image (left) was coloured to show continuous paths on or between the staple extensions (right).

Programming the tile

To demonstrate the programmability of random arrays, we explored how to design the local tile patterns that control the properties of the global array pattern. The classic Truchet tile creates loops, but to create mazes, branching points are needed. This can be accomplished by designing how the lines connect within a tile and among neighbouring tiles. For example, a diagonal tile will introduce branching points at the centre of a 2 × 2 neighbourhood (Fig. 3a). In these arrays, both three-way and four-way junctions are possible. The distances between adjacent junctions vary, with shorter distances being more likely than longer distances.

When branching points are introduced within a tile instead of across tiles, it is possible to remove four-way junctions and allow a fixed distance between adjacent junctions (for example, with a 'T' tile, Fig. 3b). In random T tile arrays, there are four possible tile orientations, but all junctions occur at the centre of a tile so there can only be three-way junctions. All tiles have a junction at the centre and thus the adjacent junctions on connected branches are always one tile apart.

There are two ways of looking at the global patterns created on random arrays: following the lines or following the areas between the lines, which can exhibit distinct properties. By inverting a tile design, area patterns of one tile that have the same properties as line patterns of another tile are created, and vice versa. For example, the arc tile can be seen as an inverse of the diagonal tile, and vice versa (Fig. 3c; Supplementary Fig. 32a,c).

Besides the branching rules, the proximity between mazes are also qualitatively different: in random arrays of diagonal and arc tiles, the adjacent mazes are interwoven; in random arrays of T tiles, it is easy to find a large rectangular area occupied by a single maze. Very small mazes occur within a large maze, but mazes with comparable sizes are separated.

Quantitatively, the size distribution of diagonal and arc mazes is significantly different from the T mazes. Numerical simulations suggested that the size of the largest maze on random arrays of T tiles is expected to be larger than that of diagonal and arc tiles, with a wider distribution (Fig. 3d). Unlike the diagonal tile arrays, the area patterns on T tile arrays are also mazes, but with a smaller expected size (Supplementary Fig. 32b,d,e).

We constructed random T tile arrays using the same tile shown in Fig. 2e but with four types of surface modifications for the four orientations of the T pattern. AFM images showed maze-like patterns in both arc and T tile arrays (Fig. 3e; Supplementary Figs 33–35). The expected branching properties, network proximity and difference in maze size were observed (Supplementary Figs 36 and 37): the arc mazes had three-way and four-way junctions that were one to six tiles apart, and the T mazes had only three-way junctions that were all one tile apart. The arc mazes were interwoven and the T mazes were mostly separated. The largest arc maze in a 10 × 10 tile area was 39.4 ± 4.5 and the largest T maze was 77.8 ± 9.5, agreeing with the expected sizes of 42.3 and 75.3.

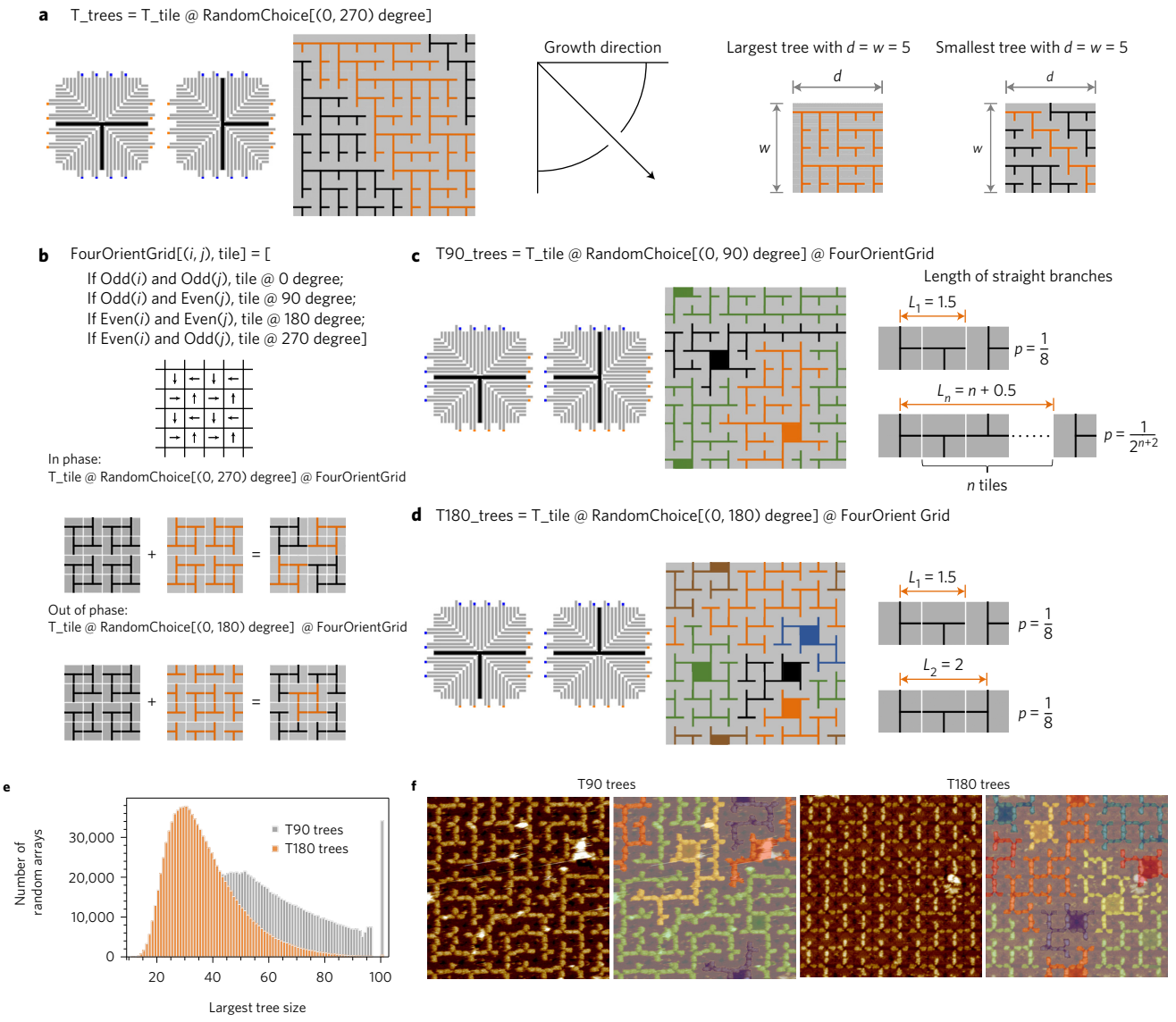

Figure 4 | Programming the grid. a, A T tile design that generates random tree-like patterns with a specific growth direction. *d* and *w* are the depth and width of a tree, respectively. **b**, A four-orientation grid that allows trees to grow in all directions (the syntax is explained in Methods). Arrows indicate the specific orientations of a tile at given locations (for example, the down arrow indicates the orientation of a tile as it is shown, the left arrow indicates the orientation of a tile that is rotated 90° clockwise from what it is shown and so on). One example random array is shown when two types of T tiles (in black and orange) are alone (left of =) and mixed together (right of =), either in phase (not a tree) or out of phase (a tree with a square root). **c**,**d**, Two T tile designs on the four-orientation grid that both generate random tree-like patterns but with different lengths of straight branches. For the convenience of analysing these trees, we assume that each array is on a torus. Individual trees in each example random array are shown in distinct colours. The root of each tree is filled with the same colour as the branches. **e**, Histogram of the largest tree size on random 10 × 10 arrays, generated from numerical simulations with one million independent trials. The size of a tree is determined by the number of tiles in the tree. The peak at size 100 of T90 trees indicates arrays with a single tree, which is a result of the array size being limited to 10 × 10. A smooth and complete tail of the size distribution occurred with simulations of 16 × 16 arrays (Supplementary Fig. 39f). **f**, AFM images of random trees on T90 and T180 tile arrays. Each original AFM image (left) was coloured to show continuous paths on the staple extensions (right).

In the maze examples, only straight lines are needed to define the branching rules (for example, the arc tile can be replaced by two straight lines that connect the same points and the configuration of the mazes would stay the same). But if we also consider how the lines are connected in terms of the exact curvature, more complex geometries can be created (Supplementary Fig. 38).

Programming the grid

The next task was to remove the loops from the mazes and create trees. Taking T mazes as an example, the simplest loops can be created in a 2 × 2 neighbourhood, using T tiles of all four orientations or just two orientations that are rotated 180° from each other. Allowing globally only two T tile orientations that are 90° rotated from each other will prohibit the formation of loops (Fig. 4a), but the direction of growth of the trees is always along the diagonal of the arrays, considering the corner tile as the root.

Is it possible to create trees that grow in all directions? Yes, and it can be accomplished by designing a grid that allows the tiles to adopt specific orientations at different locations. For example, on a grid with four distinct orientations of tiles in any 2 × 2 neighbourhood (Fig. 4b), any single type of T tile will create square loops of size 4. Any two types of T tile are either in phase (that is, alternating

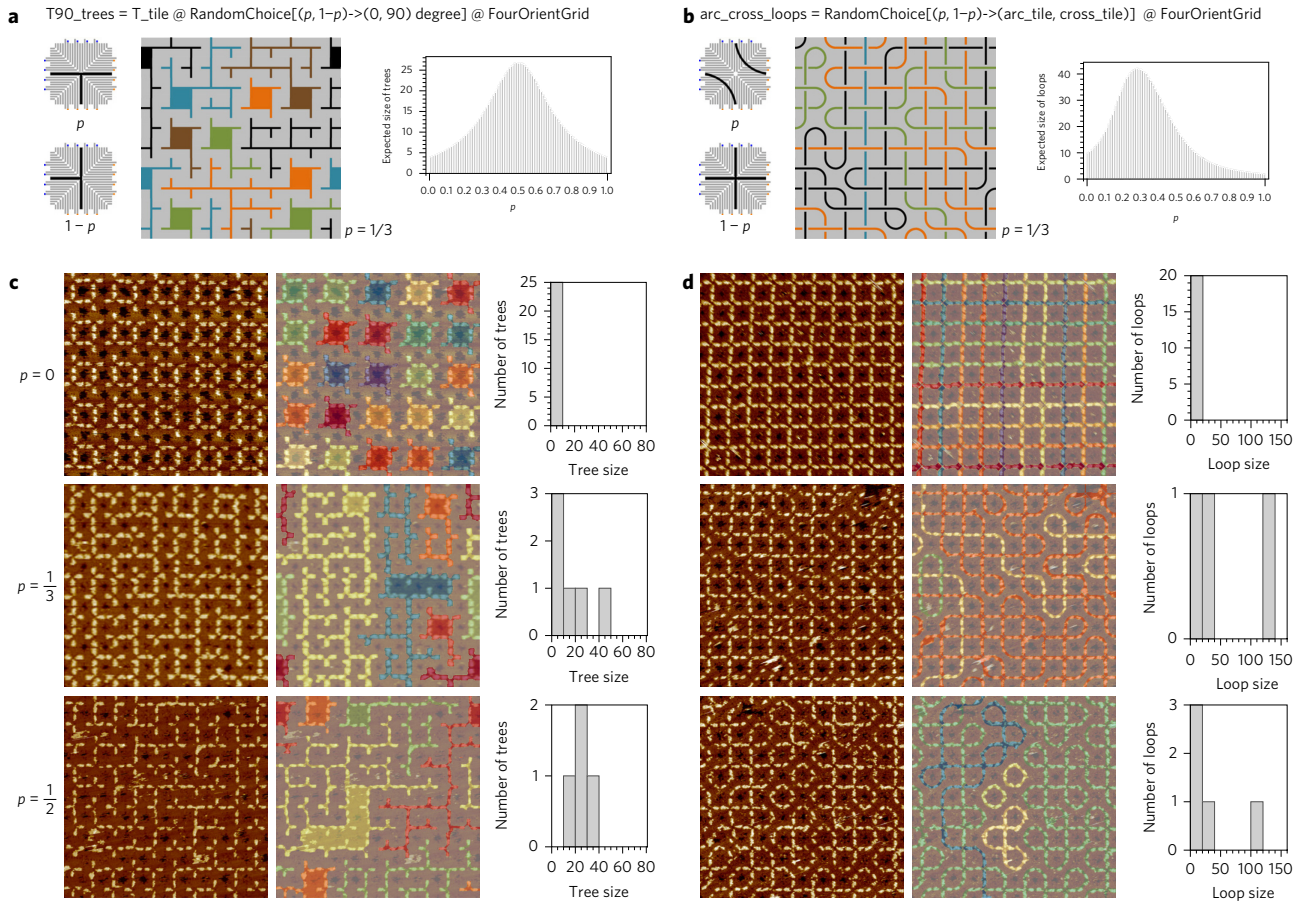

Figure 5 | Programming the random choice. a, Random trees with the size controlled by the probabilities of two tiles with distinct pattern orientations (the syntax is explained in Methods). **b**, Random loops with the size of loops and number of crossings controlled by the probabilities of two tiles with distinct patterns (the syntax is explained in Methods). For the convenience of analysing the trees and loops, we assume that each array is on a torus. Expected size of trees and loops on random 10 × 10 arrays were generated from numerical simulations with ten thousand independent trials for each value of p from 0 to 1 with 0.01 increments. The length of an arc on each tile is 0.5, and the length of a straight line on each tile is 1. The size of the loops increases with the size of random arrays at $p = 0$ but remains constant at $p = 1$, and thus the maximum point of expected loop size shifts more toward $p = 0$ with larger arrays. **c**,**d**, AFM images and analysis of trees and loops on random arrays with $p = 0$, 1/3 and 1/2.

choices in adjacent 2 × 2 neighbourhoods exist to connect all of the loops together) or out of phase (that is, preserving one loop would necessarily break the adjacent loops).

Allowing each tree to have a single loop, considered as the root, random trees that grow in all directions can be created on the four-orientation grid with two types of T tile that are out of phase, rotated by either 90° or 180° from each other (Fig. 4c,d). The sizes of the roots will vary but larger roots will be less likely than smaller roots. Numerical simulations of random 10 × 10 arrays suggested that more than 80% of the T90 trees and more than 99% of the T180 trees will have a 'square root', which is the smallest possible loop (Supplementary Fig. 39d).

Other than the difference in the percentage of square roots, the two types of trees also have distinct branching properties. In T90 trees, the lengths of straight branches vary, with shorter branches more likely than longer branches (Fig. 4c). In T180 trees, starting from a junction, there are only two possible lengths for straight branches (Fig. 4d). The sizes of the trees are different as well: the size of the largest trees on random T90 arrays is expected to be larger than that on random T180 arrays, with a wider distribution (Fig. 4e).

The edge design of the tile shown in Fig. 2e naturally implements the four-orientation grid, because the two pairs of matching sticky ends on the adjacent sides of the square force the tiles to attach to each other with the same relative orientations as specified in Fig. 4b. Using this tile with two orientations of surface modifications in a T pattern, we constructed two types of random trees on T90 and T180 arrays (Fig. 4f; Supplementary Figs 40 and 41). They grew in all directions, each demonstrating the expected branching properties: the T90 trees in a 10 × 10 tile area had straight branches of varying lengths from 1.5 to 8.5, and the T180 trees only had straight branches of length 1.5 and 2. The expected difference in size was also observed (Supplementary Fig. 42): the largest T90 tree was 54.7 ± 11.7 and the largest T180 tree was 33.4 ± 11.9, agreeing with the expected sizes of 58.0 and 36.7.

To gain greater control of the global patterns, more complex grids can be defined using multiple types of tile with distinct edge designs, at the cost of increased design and experimental challenges, which we will discuss in a later section.

Programming the random choice

In previous sections, we achieved equal probabilities of pattern orientations by mixing the corresponding tiles in an equimolar ratio. Now, simply by changing this tile ratio, we can tune the size distributions of global patterns more precisely (Supplementary Fig. 43). For example, the expected size of T90 trees depends on the probability of the two T pattern orientations (Fig. 5a). When $p = 0$ or 1 (where p is the probability of the illustrated configuration of branching points or distance between adjacent branching points) all trees will have just a square root with four leafs, and thus the expected size is exactly 4. When p increases from 0 or decreases

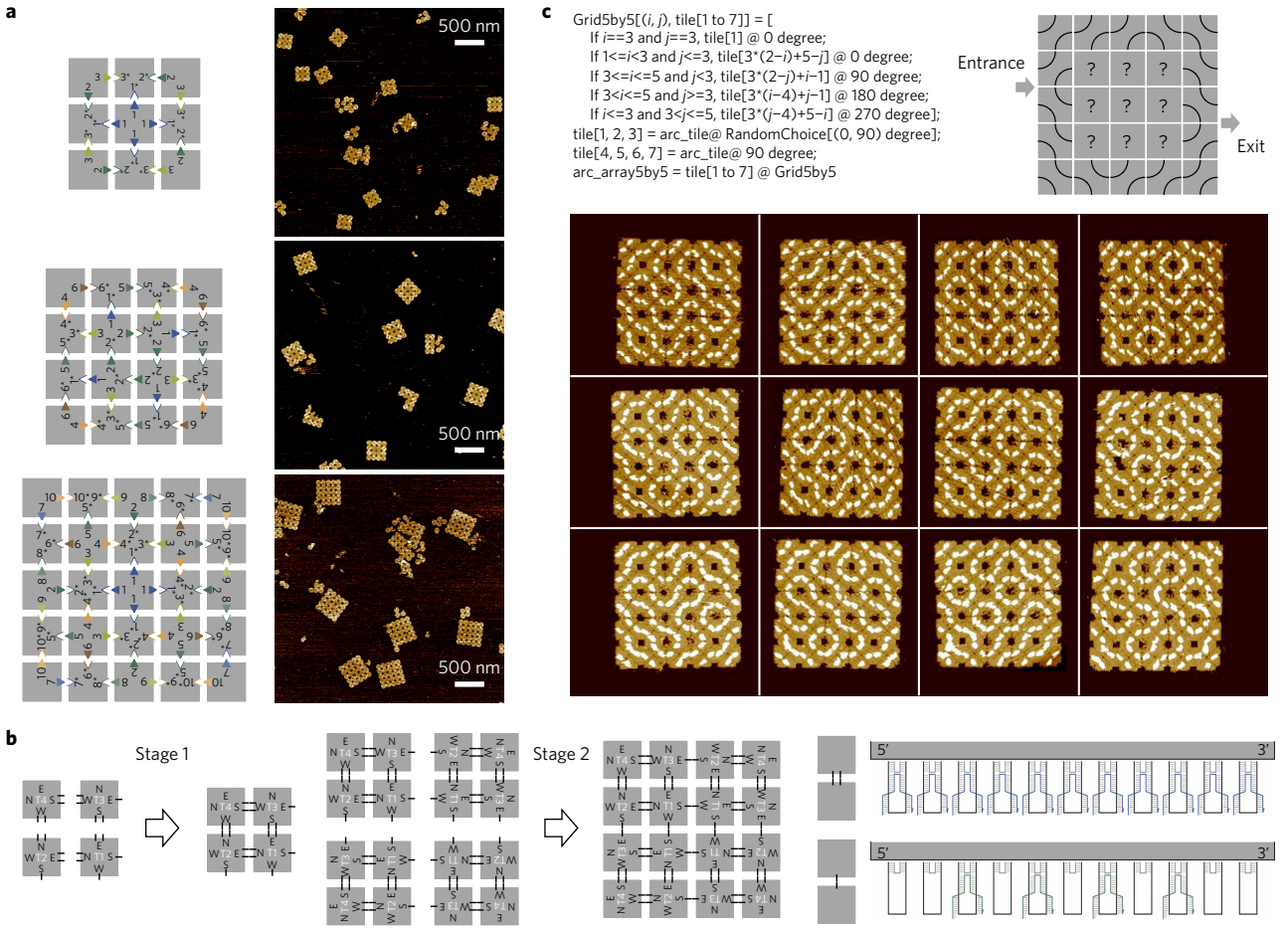

Figure 6 | Programming a finite grid. a, Abstract design diagrams and AFM images of finite DNA origami arrays with designed size. Three, four and seven distinct types of tiles were used in the 3 × 3, 4 × 4 and 5 × 5 arrays, respectively. Giving edge with extended staples and receiving edge with truncated staples are indicated by solid triangles facing outwards and hollow triangles facing inwards, respectively. **b**, One-pot staged self-assembly in finite DNA origami arrays. Four distinct types of tiles used in the 4 × 4 array are labelled as T1 to T4. Four sides of each tile are labelled as N, E, S and W. Two types of edge designs with weaker (4 edge staples that each have a stacking bond and a two-nucleotide sticky end, resulting in 16 stacking bonds total) and stronger (11 edge staples that each have a stacking bond and a one-nucleotide sticky end, resulting in 33 stacking bonds total) binding energies are indicated as one and two bars between the tiles, respectively. In the 4 × 4 array with four distinct types of tiles, we expect that one copy of each tile type will first cooperatively self-assemble into a 2 × 2 array, and then four copies of the same 2 × 2 array will self-assemble into a 4 × 4 array. **c**, Design (syntax explained in Methods) and AFM images of 5 × 5 arrays with partially random arc patterns. Mazes with designed entrances and exits can be created by fixing the orientation of the exterior tiles, but allowing the interior tiles to have random orientations. The image of 12 different mazes is composed from several independent AFM images.

from 1, the trees will grow larger, and the expected size will reach a maximum at $p = 0.5$.

Moreover, by introducing tiles with multiple patterns and varying their probabilities, we can gain control over other features of global patterns (Supplementary Fig. 44). For example, the probability of a cross tile introduces a controlled number of crossings in random loops (Fig. 5b). When $p = 1$, all loops will be circles on the four-orientation grid. When p decreases, longer loops will appear with more crossings. However, when p continues to decrease, loops will become shorter as the number of turns decreases. When $p = 0$, crossings will be on every tile and all loops will be straight lines of length n on all n by n arrays.

We constructed random trees with the two types of T tile, and random loops with the arc tiles and cross tiles, mixed together at ratios of p and $1 - p$, with three probabilities (Fig. 5c,d; Supplementary Figs 45 and 47). As expected, when $p = 0$, all trees were square roots with four leafs. When $p = 1/3$ and $1/2$, larger trees were observed with decreasing number of trees on each array. The average tree size was 17.6 ± 5.8 when $p = 1/3$ and 25.9 ± 8.0 when $p = 1/2$, agreeing with the expected sizes 16.7 and 26.7, respectively (Supplementary Fig. 46). Also as expected, all loops were straight lines when $p = 0$. Compared with $p = 1/2$, longer and fewer loops were observed when $p = 1/3$. The total number of crossings was 70.3 ± 7.4 when $p = 1/3$ and 48.7 ± 4.4 when $p = 1/2$, also agreeing with the expected numbers 67 and 50, respectively (Supplementary Fig. 48).

Programming a finite grid

As we have shown, with just a single type of tile in terms of the edge design, and with different types of surface modifications, complex patterns with desired properties can be created on unbounded DNA origami arrays. A fundamentally different approach for controlling the complexity of global patterns is to control the size of the grid by creating finite DNA origami arrays. Because a finite grid requires multiple types of tile with distinct edge designs, it also allows specific pattern configurations at some locations that differ from other locations.

We explored two designs for creating finite origami arrays, each exhibiting an important feature for scaling up in size: one encourages complete assemblies over incomplete ones, and

another allows an asymptotically smaller number of distinct tiles and edges (Supplementary Figs 49–51). The tiles are fully connected in the first design, and connected like a comb structure in the second design, similar to previously proposed designs for self-assembled squares[41]. Both array designs have a four-fold rotational symmetry.

We established four criteria for designing a large set of distinct edges (Supplementary Note 8.2 and Supplementary Fig. 52), and constructed finite arrays of sizes 3×3, 4×4 and 5×5 (Fig. 6a; Supplementary Figs 53–58). The yield (Supplementary Fig. 59) of fully connected arrays was determined to be 15.6, 15.0 and 32.4%, respectively (Supplementary Figs 60–62), and the yield of comb-connected arrays was determined to be 8.0, 6.7 and 1.3% (Supplementary Figs 63–65). Our results suggest that the advantage of complete over incomplete assemblies is crucial for scaling up the size of finite origami arrays.

An important design principle, which we refer to as one-pot staged self-assembly (Fig. 6b), contributed to the high yield of the finite arrays. We used different strengths of the binding energy for specific edge interactions to encourage sequential stages of self-assembly during annealing of all types of tile in one pot. This strategy divides a more complex self-assembly process into multiple simpler stages, and thus reduces the potential spurious interactions that could occur at any given time. Compared with previous methods for constructing finite shapes using origami tile–tile interactions[9,42], our approach enabled a substantially higher efficiency of the conversion from origami monomers into target assemblies. These structures were also larger and demonstrated better scalability than arrays created with other methods, including organizing origami tiles using scaffold frames[43].

Using the fully connected design for 5×5 arrays, we created arc mazes of $440 \times 440\ nm^2$ in size (Supplementary Fig. 66). Theoretically, if we look at the mazes from a fixed view point, over 30 million (that is, $2^{25} = 33{,}554{,}432$) distinct mazes can be found on the arrays in one test tube. The average number of circles was 1.0 ± 0.6, agreeing with the expected number $(m-1) \times (n-1) \times (1^4/2)$ in a random $m \times n$ array of the arc tiles[44].

To demonstrate not only controlled size of a finite grid, but also controlled pattern configurations at specific locations, we created random mazes with designed entrances and exits (Fig. 6c; Supplementary Fig. 67). In these arrays, the exterior tiles had a fixed arc orientation, while the interior tiles remained a random choice between the two arc orientations. Taking rotational symmetry into consideration, $2^8 = 256$ distinct mazes exist, all of which should have a path from the designed entrance to exit—such a path was found in all 5×5 arrays shown in the AFM images.

The rule that we introduced for programming a finite grid is not limited to a square-shaped grid. In general, finite grids with more complex shapes can be used to create patterns with well-defined boundaries (Supplementary Fig. 68).

Conclusions

We developed a framework for creating programmable disorder in random DNA tilings and created DNA nanostructures with a combinatorial diversity of complex patterns, including random loops, mazes and trees. These self-assembled patterns exploit both deterministic and random features, trading some amount of the control offered by deterministic processes for the diversity and simple algorithms offered by random processes. We demonstrated that simple rules in random tilings are still powerful enough to control important global properties, including the formation of loops (for example, the difference between mazes and trees), the branching rules (such as the configuration of junctions, the distances between adjacent branching points and the lengths of straight branches), the growth direction of trees, the proximity between adjacent networks (interwoven versus separated) and the size distribution (different means and standard deviations).

The principle of programmable disorder offers solutions that neither deterministic nor random processes alone can offer. DNA origami can be used to create arbitrary complex patterns on the scale of 100 nm. Using scaffold strands that are longer than the M13 viral DNA[45] or arrays of addressable DNA origami tiles[42,43], it is possible to create complex patterns that are up to nine times larger than a single origami folded from M13. However, if only deterministic assembly processes are used to create any of the loops, mazes and trees that we showed here, one would have to further increase the complexity of uniquely addressable DNA nanostructures by 10 to 100 times over what current techniques are capable of. Moreover, to create all possible mazes on 5×5 arrays with designed entrances and exits as we demonstrated, it would be 256 times more expensive if the assembly processes are fully deterministic. On the other hand, if only random processes are involved and the global features are not properly controlled by local rules, it would be impossible to create complex patterns with desired properties, without simultaneously generating a large fraction of molecules that are wasted.

One example application of random DNA origami arrays is for molecular robots (Supplementary Note 9.1). The sophisticated functions of biological motors[46,47] have inspired the development of synthetic DNA motors using DNA origami as a 2D playground[48–50]. However, compared with the operating environments of biological motors, tracks that can be built on a single DNA origami are far less complex. The random arrays that we created could be used to provide DNA robots with diverse operating and testing environments that are much more complex than a single DNA origami. This would be critical for enabling the development of DNA robots that perform increasingly sophisticated tasks such as maze-solving, and for improving our understanding of how to build robust DNA robots that operate well under a wide range of conditions. Compared with deterministic approaches, random arrays have the distinct advantage of performing massively parallel experiments in one test tube.

From a broader perspective, random DNA origami arrays are not only useful for creating patterns that are geometrically interesting, like the examples that we showed here, but they could also be applied to create combinatorial circuits and devices with desired functions (Supplementary Fig. 69). A single DNA origami can be used to organize carbon nanotubes[51] and polymers[14] to build functional components of molecular electronics, and to organize metal nanoparticles[12], nanorods[13,52] and organic dyes[11] to build functional components of molecular plasmonics and photonics. Using these approaches as building blocks, and using the principle of programmable disorder in random DNA origami arrays, it would be possible to create complex networks of molecular devices with controlled size distributions, branching properties and circuit functions (Supplementary Note 9.3). Much as combinatorial approaches for generating random 1D chains of polymers have been used to revolutionize chemical synthesis[53] and the selection of functional nucleic acids[54], programmable disorder that extends the principle to random 2D networks of molecules now creates new opportunities for fabricating more complex molecular devices organized by DNA nanostructures.

Methods

Methods and any associated references are available in the online version of the paper.

Received 7 July 2016; accepted 18 October 2016;
published online 28 November 2016

References

1. Rao, C. V., Wolf, D. M. & Arkin, A. P. Control, exploitation and tolerance of intracellular noise. *Nature* **420,** 231–237 (2002).

2. Kærn, M., Elston, T. C., Blake, W. J. & Collins, J. J. Stochasticity in gene expression: from theories to phenotypes. *Nat. Rev. Genet.* **6,** 451–464 (2005).
3. Eldar, A. & Elowitz, M. B. Functional roles for noise in genetic circuits. *Nature* **467,** 167–173 (2010).
4. Losick, R. & Desplan, C. Stochasticity and cell fate. *Science* **320,** 65–68 (2008).
5. Zipursky, S. L. & Sanes, J. R. Chemoaffinity revisited: dscams, protocadherins, and neural circuit assembly. *Cell* **143,** 343–353 (2010).
6. Livet, J. *et al.* Transgenic strategies for combinatorial expression of fluorescent proteins in the nervous system. *Nature* **450,** 56–62 (2007).
7. Adleman, L. M. Molecular computation of solutions to combinatorial problems. *Science* **266,** 1021–1024 (1994).
8. Seeman, N. C. Nucleic acid junctions and lattices. *J. Theor. Biol.* **99,** 237–247 (1982).
9. Rothemund, P. W. K. Folding DNA to create nanoscale shapes and patterns. *Nature* **440,** 297–302 (2006).
10. Rinker, S., Ke, Y., Liu, Y., Chhabra, R. & Yan, H. Self-assembled DNA nanostructures for distance-dependent multivalent ligand–protein binding. *Nat. Nanotech.* **3,** 418–422 (2008).
11. Schreiber, R. *et al.* Hierarchical assembly of metal nanoparticles, quantum dots and organic dyes using DNA origami scaffolds. *Nat. Nanotech.* **9,** 74–78 (2014).
12. Pal, S., Deng, Z., Ding, B., Yan, H. & Liu, Y. DNA-origami-directed self-assembly of discrete silver-nanoparticle architectures. *Angew. Chem.* **122,** 2760–2764 (2010).
13. Pal, S. *et al.* DNA directed self-assembly of anisotropic plasmonic nanostructures. *J. Am. Chem. Soc.* **133,** 17606–17609 (2011).
14. Knudsen, J. B. *et al.* Routing of individual polymers in designed patterns. *Nat. Nanotech.* **10,** 892–898 (2015).
15. Pinheiro, A. V., Han, D., Shih, W. M. & Yan, H. Challenges and opportunities for structural DNA nanotechnology. *Nat. Nanotech.* **6,** 763–772 (2011).
16. Winfree, E., Liu, F., Wenzler, L. A. & Seeman, N. C. Design and self-assembly of two-dimensional DNA crystals. *Nature* **394,** 539–544 (1998).
17. Yan, H., Park, S. H., Finkelstein, G., Reif, J. H. & LaBean, T. H. DNA-templated self-assembly of protein arrays and highly conductive nanowires. *Science* **301,** 1882–1884 (2003).
18. Malo, J. *et al.* Engineering a 2D protein–DNA crystal. *Angew. Chem. Int. Ed.* **44,** 3057–3061 (2005).
19. Zheng, J. *et al.* Two-dimensional nanoparticle arrays show the organizational power of robust DNA motifs. *Nano Letters* **6,** 1502–1504 (2006).
20. Zheng, J. *et al.* From molecular to macroscopic via the rational design of a self-assembled 3D DNA crystal. *Nature* **461,** 74–77 (2009).
21. Liu, W., Zhong, H., Wang, R. & Seeman, N. C. Crystalline two-dimensional DNA-origami arrays. *Angew. Chem.* **123,** 278–281 (2011).
22. Woo, S. & Rothemund, P. W. K. Self-assembly of two-dimensional DNA origami lattices using cation-controlled surface diffusion. *Nat. Commun.* **5,** 4889 (2014).
23. Aghebat Rafat, A., Pirzer, T. & Scheible, M. B., Kostina, A. & Simmel, F. C. Surface-assisted large-scale ordering of DNA origami tiles. *Angew. Chem. Int. Ed.* **53,** 7665–7668 (2014).
24. Rothemund, P. W. K., Papadakis, N. & Winfree, E. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biol.* **2,** e424 (2004).
25. Barish, R. D., Schulman, R., Rothemund, P. W. K. & Winfree, E. An information-bearing seed for nucleating algorithmic self-assembly. *Proc. Natl Acad. Sci. USA* **106,** 6054–6059 (2009).
26. Schulman, R., Yurke, B. & Winfree, E. Robust self-replication of combinatorial information via crystal growth and scission. *Proc. Natl Acad. Sci. USA* **109,** 6405–6410 (2012).
27. Woo, S. & Rothemund, P. W. K. Programmable molecular recognition based on the geometry of DNA nanostructures. *Nat. Chem.* **3,** 620–627 (2011).
28. Gerling, T., Wagenbauer, K. F., Neuner, A. M. & Dietz, H. Dynamic DNA devices and assemblies formed by shape-complementary, non–base pairing 3D components. *Science* **347,** 1446–1452 (2015).
29. Truchet, S. Mémoire sur les combinaisons. *Mém. Acad. R. Sci.* **1704,** 363–372 (1704).
30. Smith, C. S. & Boucher, P. The tiling patterns of Sebastien Truchet and the topology of structural hierarchy. *Leonardo* **20,** 373–385 (1987).
31. Murphy, C. J. *et al.* Structure and energetics of hydrogen-bonded networks of methanol on close packed transition metal surfaces. *J. Chem. Phys.* **141,** 014701 (2014).
32. Less, J. R., Skalak, T. C., Sevick, E. M. & Jain, R. K. Microvascular architecture in a mammary carcinoma: branching patterns and vessel dimensions. *Cancer Res.* **51,** 265–273 (1991).
33. Portmann, O., Vaterlaus, A. & Pescia, D. An inverse transition of magnetic domain patterns in ultrathin films. *Nature* **422,** 701–704 (2003).
34. Blagodatski, A., Sergeev, A., Kryuchkov, M., Lopatina, Y. & Katanaev, V. L. Diverse set of Turing nanopatterns coat corneae across insect lineages. *Proc. Natl Acad. Sci. USA* **112,** 10750–10755 (2015).
35. Metzger, R. J., Klein, O. D., Martin, G. R. & Krasnow, M. A. The branching programme of mouse lung development. *Nature* **453,** 745–750 (2008).
36. Lefebvre, J. L., Kostadinov, D., Chen, W. V., Maniatis, T. & Sanes, J. R. Protocadherins mediate dendritic self-avoidance in the mammalian nervous system. *Nature* **488,** 517–521 (2012).
37. Svitkina, T. M. & Borisy, G. G. Arp2/3 complex and actin depolymerizing factor/cofilin in dendritic organization and treadmilling of actin filament array in lamellipodia. *J. Cell Biol.* **145,** 1009–1026 (1999).
38. Temperley, H. N. V. & Lieb, E. H. Relations between the 'percolation' and 'colouring' problem and other graph-theoretical problems associated with regular planar lattices: some exact results for the 'percolation' problem. *Proc. R. Soc. Lond. A* **322,** 251–280 (1971).
39. Blöte, H. W. J. & Nienhuis, B. Fully packed loop model on the honeycomb lattice. *Phys. Rev. Lett.* **72,** 1372–1375 (1994).
40. Nahum, A., Chalker, J. T., Serna, P., Ortuño, M. & Somoza, A. M. 3D loop models and the $CP^{(n-1)}$ sigma model. *Phys. Rev. Lett.* **107,** 110601 (2011).
41. Rothemund, P. W. K. & Winfree, E. The program-size complexity of self-assembled squares. In *Proc. 32nd Annual ACM Symp. Theory Computing* 459–468 (Association for Computing Machinery, 2000).
42. Rajendran, A., Endo, M., Katsuda, Y., Hidaka, K. & Sugiyama, H. Programmed two-dimensional self-assembly of multiple DNA origami jigsaw pieces. *ACS Nano* **5,** 665–671 (2011).
43. Zhao, Z., Liu, Y. & Yan, H. Organizing DNA origami tiles into larger structures using preformed scaffold frames. *Nano Letters* **11,** 2997–3002 (2011).
44. Browne, C. Truchet curves and surfaces. *Comput. Graph.* **32,** 268–281 (2008).
45. Marchi, A. N., Saaem, I., Vogen, B. N., Brown, S. & LaBean, T. H. Toward larger DNA origami. *Nano Lett.* **14,** 5740–5747 (2014).
46. Hirokawa, N. Kinesin and dynein superfamily proteins and the mechanism of organelle transport. *Science* **279,** 519–526 (1998).
47. Vale, R. D. The molecular motor toolbox for intracellular transport. *Cell* **112,** 467–480 (2003).
48. Lund, K. *et al.* Molecular robots guided by prescriptive landscapes. *Nature* **465,** 206–210 (2010).
49. Gu, H., Chao, J., Xiao, S. -J. & Seeman, N. C. A proximity-based programmable DNA nanoscale assembly line. *Nature* **465,** 202–205 (2010).
50. Wickham, S. F. J. *et al.* A DNA-based molecular motor that can navigate a network of tracks. *Nat. Nanotech.* **7,** 169–173 (2012).
51. Maune, H. T. *et al.* Self-assembly of carbon nanotubes into two-dimensional geometries using DNA origami templates. *Nat. Nanotech.* **5,** 61–66 (2010).
52. Zhou, C., Duan, X. & Liu, N. A plasmonic nanorod that walks on DNA origami. *Nat. Commun.* **6,** 8102 (2015).
53. Gordon, E. M., Barrett, R. W., Dower, W. J., Fodor, S. P. A. & Gallop, M. A. Applications of combinatorial technologies to drug discovery. 2. Combinatorial organic synthesis, library screening strategies, and future directions. *J. Med. Chem.* **37,** 1385–1401 (1994).
54. Ellington, A. D. & Szostak, J. W. In vitro selection of RNA molecules that bind specific ligands. *Nature* **346,** 818–822 (1990).

Acknowledgements

We thank S. Wang for designing the fish and gear tile (Supplementary Fig. 38) and J. Parkin, A. Karan and S. Wang for designing and creating a heart shape that is self-assembled from square DNA origami tiles (Supplementary Fig. 68). We thank E. Winfree, P. Rothemund, D. Soloveichik and A. Condon for critique on the manuscript. G.T. was supported by an NSF grant (1317694). P.P. was supported by a NIH/NRSA training grant (5 T32 GM07616). L.Q. was supported by a Career Award at the Scientific Interface from the Burroughs Wellcome Fund (1010684) and a Faculty Early Career Development Award from the NSF (1351081).

Author contributions

G.T. and P.P. performed the experiments and analysed the data. P.P. performed the simulations and developed the software tools. G.T., P.P. and L.Q. designed the systems and wrote the manuscript. L.Q. initiated and guided the project.

Additional information

Supplementary information is available in the online version of the paper. Reprints and permissions information is available online at www.nature.com/reprints. Correspondence and requests for materials should be addressed to L.Q.

Competing financial interests

The authors declare no competing financial interests.

Methods

Syntax of the programming language.

(1) Defining patterns on a tile:
tile = ***Connect***[((x_1, y_1), (x_2, y_2)) @ (cx_1, cy_1), …]
(x_i, y_i) defines the start and end points of a line. (cx_i, cy_i) defines the centre of an arc. When @ is missing, the points are connected by a straight line.

(2) Defining a grid:
grid[(***i***, ***j***), ***tile***] = [If $cond_1$(***i***, ***j***), ***tile*** @ $orient_1$;…]
(i, j) indicates a location on the grid. $cond_i$ defines a set of specific locations on the grid, as a function of (i, j). $orient_i$ defines the orientation of a tile at cond$_i$ on the grid. The default grid is $grid$[(i, j), $tile$] = [$tile$ @ 0 degree], which has the same orientation of tiles at all locations. tile can be replaced by a set of tiles ***tile***[**1, 2, …, *n***], in which case each $cond_i$ will be associated with a subset of tiles ***tile***[t_1, t_2, …], $t_i \in \{1, 2, \ldots, n\}$.

(3) Defining a random choice of tile orientations:
tile @ RandomChoice[(p_1, p_2, …) → ($orient_1$, $orient_2$, …)]
$orient_i$ defines an orientation of the tile. p_i is the probability of $orient_i$ ($\sum p_i = 1$). The default probability is $1/n$, where n is the total number of choices.

(4) Defining a random choice of tile types:
RandomChoice[(p_1, p_2, …) → ($tile_1$ @ $orient_1$, $tile_2$ @ $orient_2$, …)]
$orient_i$ defines an orientation of $tile_i$. p_i is the probability of $tile_i$ @ $orient_i$ ($\sum p_i = 1$). The default probability is $1/n$, where n is the total number of choices. The default orientation is 0 degree.

(5) Defining a random array:
array = ***tile @ RandomChoice @ grid***

Sample preparation. Single-stranded M13mp18 DNA (scaffold strand) was purchased from Bayou Biolabs (Catalog # P-107) at 1 g l^{-1} in 1 × TE buffer (10 mM Tris-HCl, 1 mM EDTA, pH 8.0). The concentration of scaffold strand was calculated on the basis of DNA ultraviolet absorbance measurements at 260 nm using NanoDrop2000 (Thermo Scientific). Staple strands were purchased unpurified from Integrated DNA Technologies in 1 × TE buffer (pH 8.0) at 100 μM each.

Individual DNA origami tiles for creating unbounded arrays were prepared with 50 nM scaffold strand and 75 nM staple strands in 1 × TE/Mg^{2+} (1 × TE buffer containing 12.5 mM magnesium acetate). Individual DNA origami tiles for creating finite arrays with designed size were prepared with 10 nM scaffold strand and 75 nM staples in 1 × TE/Mg^{2+} buffer. In both protocols the scaffold and staple mixtures were kept at 90°C for 2 min and annealed from 90°C to 20°C at 6 sec per 0.1°C.

Unbounded arrays were constructed using: (1) an overnight anneal from 40 to 30°C at 5 min per 0.1°C and then from 30 to 20°C at 10 sec per 0.1°C (examples include the arrays shown in Figs 2– 5); (2) a two-day anneal from 40 to 30°C at 25 min per 0.1°C and then from 30 to 20°C at 10 sec per 0.1°C (examples include the array shown in Supplementary Fig. 19); or (3) a one-week anneal from 40 to 30°C at 60 min per 0.1°C and then from 30 to 20°C at 30 min per 0.1°C (examples include the arrays shown in Supplementary Figs 20 and 21).

Before mixing different types of tiles for creating finite arrays with designed size, a tenfold excess (relative to the concentration of staple strands) of a full set of 44 negation strands (sequences listed in Supplementary Table 6) were added to each type of DNA origami tile and quickly cooled down from 50°C to 20°C at 2 sec per 0.1°C. Different types of tiles were then mixed together and annealed from 50°C to 20°C at 2 min per 0.1°C.

AFM imaging. Samples for AFM imaging of unbounded arrays were prepared by diluting the origami to 5 nM (monomer concentration) in 1 × TE/Mg^{2+} buffer. After dilution, 40 μl of the sample was deposited onto freshly cleaved mica (SPI Supplies, 9.5 mm diameter, LOT # 1170203). After 30 s the solution was removed by sucking up all the liquid that comes off in a single thumb-up movement while keeping the pipette attached to and almost perpendicular to the mica surface. After that, 80 μl of a 1 × TE/Mg^{2+} buffer was added onto the mica and the sample was imaged.

Samples for AFM imaging of the individual DNA origami tiles and finite arrays with designed sizes were prepared by diluting the origami to 1 nM (single-tile or target finite-shape concentration) in a 1 × TE/Mg^{2+} buffer. The following steps were the same as for unbounded arrays, except after removing the solution, the mica surface was washed three times with 40 μl TE buffer containing 10 mM $MgCl_2$ and 10 mM NaCl, by performing 10 down-and-up thumb movements for each wash. Compared with unbounded arrays, the finite arrays had a much larger excess of short strands (including a 5× higher ratio of staples to scaffold and an addition of negations strands at 10× the concentration of the staples), and thus the washing step was used to remove the short strands and provide a cleaner background for imaging.

AFM images were taken in tapping mode in fluid on a Dimension FastScan Bio (Bruker) using FastScan-D tips (Bruker). Typical scanning parameters were: scan rate = 5 Hz, lines = 512, amplitude set point = 30–50 mV, drive amplitude = 180–240 mV, drive frequency = 110 Hz, integral gain = 1, proportional gain = 2.

Size analysis. The size of a diagonal maze or a T maze was determined as the number of tiles in the maze. The size of an arc maze was determined as the size of its equivalent diagonal maze. The size of a tree was determined as the number of tiles in the tree. For calculating the size of loops in random arrays of arc and cross tiles, the length of an arc on each tile was 0.5, and the length of a straight line on each tile was 1. To analyse trees and loops, we assumed that each array was on a torus. All of the expected sizes of mazes, trees and loops were computed from numerical simulations with ten thousand to one million independent trials. All data analysis of the sizes was calculated as $\mu \pm 2\sigma$ in a 10 × 10 tile area from three to ten independent AFM images, where μ is the mean, σ is the standard error of the mean and $\pm 2\sigma$ corresponds to 95% confidence.

Yield calculation. The yield of finite DNA origami arrays was calculated using high-resolution 30 μm × 30 μm AFM images, each containing 3,000 to 12,000 DNA origami tiles. The calculation was aided by a custom software tool, which determines the yield as the total pixels in isolated complete assemblies of the designed size divided by the total pixels above the threshold of background (Supplementary Fig. 59).

Code availability. The code for the yield calculations can be accessed online at http://qianlab.caltech.edu/YieldCalculator/.