nature computational science

Article

https://doi.org/10.1038/s43588-023-00448-9

An additive framework for kirigami design

In the format provided by the authors and unedited

Supplementary information for "An additive framework for kirigami design"

Levi H. Dudte, Gary P. T. Choi, Kaitlyn P. Becker, L. Mahadevan

Contents

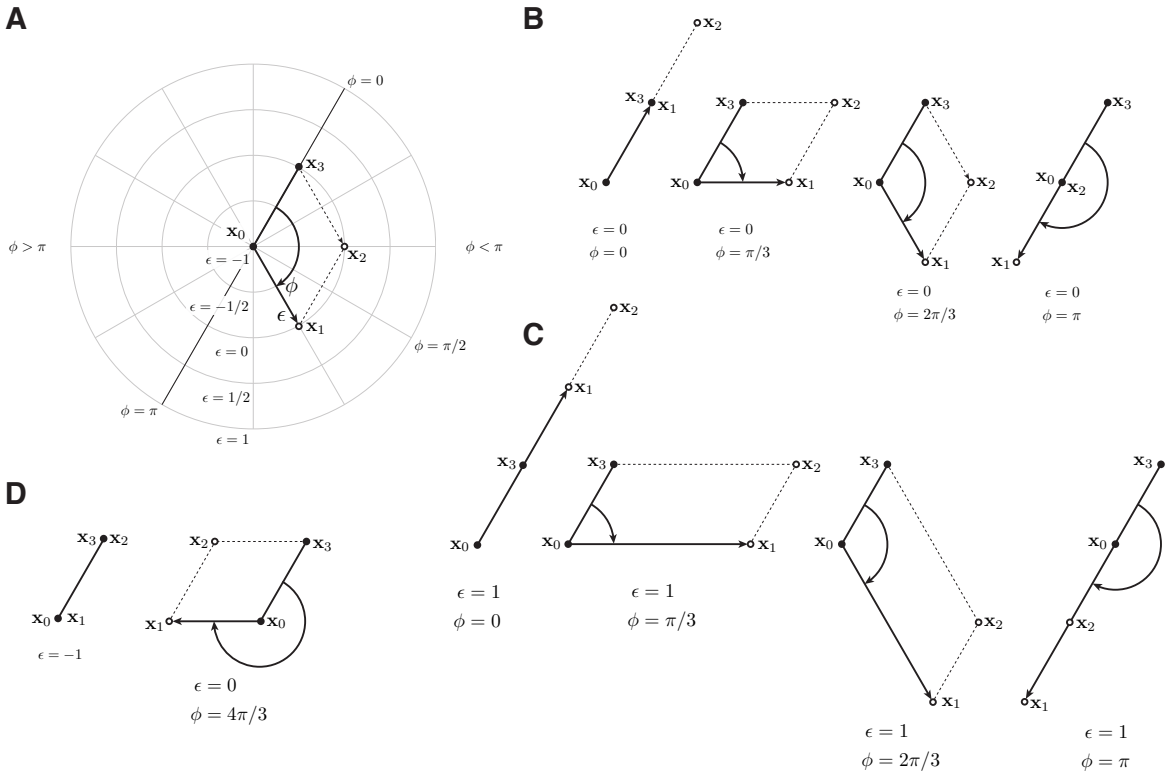
1 Linear kirigami design

1.1 Linkage design

Here we give a more detailed description of the planar parallelogram four-bar linkage design. Recall that the four-bar linkage is parameterized by the offset ϵ and the deployment angle ϕ as shown in Supplementary Figure 1A. Specifically, for the four points $\mathbf{x}_k = (x_k, y_k)$, $k \in \{0, 1, 2, 3\}$ that form the parallelogram, we have

$$D \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} I & 0 \\ I - Q & Q \\ -Q & I + Q \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix}, \tag{1}$$

where $Q = (1+\epsilon)R(-\phi)$ is a scaled rotation matrix. When $\epsilon = 0$, the linkage is a rhombus (see Supplementary Figure 1B). When $\epsilon = 1$, the linkage is a parallelogram with the side length ratio $2:1$ (see Supplementary Figure 1C). When $\epsilon = -1$, the parallelogram degenerates to two equal collinear line segments with $\mathbf{x}_0 = \mathbf{x}_1$ and $\mathbf{x}_2 = \mathbf{x}_3$ (see Supplementary Figure 1D). Otherwise, so long as $\phi \neq 0, \pi$ the generated points $\mathbf{x}$ form a parallelogram in $\mathbb{R}^2$ for all values of ϵ. The points are ordered counter-clockwise when $\phi < \pi$, clockwise when $\phi > \pi$ and are collinear when $\phi = \pi$. Holding ϵ constant keeps each edge length in the parallelogram constant and varying ϕ casts the parallelogram as a four-bar linkage, giving its one-dimensional deployment path. During deployment, two opposite interior angles in the linkage each measure ϕ and the other two each



Supplementary Figure 1: **Planar parallelogram four-bar linkage design. (A)** The parameterization of a four-bar linkage. Given two points $\mathbf{x}_0$ and $\mathbf{x}_3$, the four-bar linkage is parameterized by the deployment angle ϕ and an offset ϵ, from which the other two points $\mathbf{x}_2, \mathbf{x}_4$ can be obtained. **(B)** The deployment of a rhombus ($\epsilon = 0$). **(C)** The deployment of a rhombus ($\epsilon = 1$). **(D)** Some singular/illegal cases.

measure $\pi - \phi$.

Now we can take a closer look at each of the 2×2 blocks that comprise the linkage design matrix D, in particular their rank. The identity components I are trivially rank two and the zero components are trivially rank zero. The $\pm Q$ blocks are scalar multiples of rotation matrices $(1+\epsilon)R(-\phi)$ and are therefore rank two when $\epsilon \neq -1$ and rank zero when $\epsilon = -1$ (a degenerate case described above). Lastly, the $I \pm Q$ blocks are generically rank two and only become rank zero when $I = \mp Q$, respectively. The first case

$$I = Q \Leftrightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = (1+\epsilon) \begin{bmatrix} \cos(-\phi) & -\sin(-\phi) \\ \sin(-\phi) & \cos(-\phi) \end{bmatrix} \tag{2}$$

occurs when $(1+\epsilon)\cos(-\phi) = 1$, so when $\epsilon = 0, \phi = 0$ or $\epsilon = -2, \phi = \pi$. The second case

$$I = -Q \Leftrightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = -(1+\epsilon) \begin{bmatrix} \cos(-\phi) & -\sin(-\phi) \\ \sin(-\phi) & \cos(-\phi) \end{bmatrix} \tag{3}$$

clearly occurs when $-(1+\epsilon)\cos(-\phi) = 1$, so when $\epsilon = -2, \phi = 0$ or $\epsilon = 0, \phi = \pi$. These cases have geometric significance which can be interpreted by inspecting Equation (1). The first case gives a linkage with coincident $\mathbf{x}_3 = \mathbf{x}_1$ and collinear $-\mathbf{x}_0 + 2\mathbf{x}_3 = \mathbf{x}_2$. The second case gives a linkage with coincident $\mathbf{x}_0 = \mathbf{x}_2$ and $2\mathbf{x}_0 - \mathbf{x}_3 = \mathbf{x}_1$.

Finally, we can use Equation (1) to prove the following result:

Proposition 1. *Choosing the coordinates of any two points in a planar parallelogram four-bar linkage along with its offset ϵ and deployment angle ϕ determines the spatial configuration of the full linkage, generically.*

Proof. To prove this statement we will calculate the ranks of the six possible 4×4 block submatrices $d_{i,j}$ of D (i.e. a 4×4 matrix comprised of 2×2 of the 2×2 blocks of D):

$$d_{i,j} = \begin{bmatrix} D_i \\ D_j \end{bmatrix}, \tag{4}$$

where $(i, j) = (0, 1), (0, 2), (0, 3), (1, 2), (1, 3), (2, 3)$, which correspond to the six possible ways to choose a pair i, j from a set of four points. If a square submatrix is full rank, then it is invertible and the linear inverse solution

$$D d_{i,j}^{-1} \begin{bmatrix} \mathbf{x}_i \\ \mathbf{x}_j \end{bmatrix} = D \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix} \tag{5}$$

yields the input positions of points $\mathbf{x}_0$ and $\mathbf{x}_3$ to Equation (1) that produce a full linkage with the prescribed positions of points $\mathbf{x}_i$ and $\mathbf{x}_j$ satisfied.

$$d_{0,1} = \begin{bmatrix} D_0 \\ D_1 \end{bmatrix} = \begin{bmatrix} I & 0 \\ I - Q & Q \end{bmatrix}, \tag{6}$$

$$d_{0,2} = \begin{bmatrix} D_0 \\ D_2 \end{bmatrix} = \begin{bmatrix} I & 0 \\ -Q & I + Q \end{bmatrix}, \tag{7}$$

$$d_{0,3} = \begin{bmatrix} D_0 \\ D_3 \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}, \tag{8}$$

$$d_{1,2} = \begin{bmatrix} D_1 \\ D_2 \end{bmatrix} = \begin{bmatrix} I - Q & Q \\ -Q & I + Q \end{bmatrix}, \tag{9}$$

$$d_{1,3} = \begin{bmatrix} D_1 \\ D_3 \end{bmatrix} = \begin{bmatrix} I - Q & Q \\ 0 & I \end{bmatrix}, \tag{10}$$

$$d_{2,3} = \begin{bmatrix} D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} -Q & I + Q \\ 0 & I \end{bmatrix}. \tag{11}$$

Four of these submatrices ($d_{0,1}$, $d_{0,2}$, $d_{1,3}$ and $d_{2,3}$) are block triangular with an identity matrix and a 2×2 matrix analyzed above on the main diagonal. These four matrices are therefore rank four (full) generically and become rank two when one of the non-identity main diagonal 2×2 entries (Q, $I + Q$, $I - Q$ and $-Q$, respectively) loses rank. The submatrix $d_{0,3}$ is a 4×4 identity matrix and therefore trivially full rank. Lastly, the submatrix $d_{1,2}$ can be block row-reduced to upper triangular form in two steps:

$$d_{1,2} - \begin{bmatrix} 0 & 0 \\ I - Q & Q \end{bmatrix} + \begin{bmatrix} Q & 0 \\ I & 0 \end{bmatrix} = \begin{bmatrix} I & Q \\ 0 & I \end{bmatrix} \tag{12}$$

and is therefore full rank generically, i.e. when $\text{rank}(I \pm Q) = 2$. □

As analyzed above, each of these submatrix rank loss cases corresponds to a geometrically indeterminate linkage where two coincident points have been prescribed, leaving an additional rotational degree of freedom to determine the other two points in the linkage. Note also that we have restricted ourselves to prescribing both coordinates of a given point and therefore working only with the 2×2 block components of D. This is not necessary, however, and more submatrices of D comprised of collections of four individual (not block) rows could be shown to be full rank and therefore invertible as well, should the designer wish to prescribe a mix of coordinates across a set of points in the linkage.

To conclude our analysis of planar parallelogram four-bar linkage design, we look in more detail at the endpoints of a deployment path, i.e. when $\phi = 0, \pi$:

$$\phi = 0 \Rightarrow Q = (1+\epsilon)I \Rightarrow D = \begin{bmatrix} I & 0 \\ -\epsilon I & (1+\epsilon)I \\ -(1+\epsilon)I & (2+\epsilon)I \\ 0 & I \end{bmatrix}, \tag{13}$$

$$\phi = \pi \Rightarrow Q = -(1+\epsilon)I \Rightarrow D = \begin{bmatrix} I & 0 \\ (2+\epsilon)I & -(1+\epsilon)I \\ (1+\epsilon)I & -\epsilon I \\ 0 & I \end{bmatrix}. \tag{14}$$

Because each of the blocks becomes either a zero matrix or a scaled identity matrix, the effects of D on the respective x and y coordinates of $\mathbf{x}_0$ and $\mathbf{x}_3$ are identical, i.e. the 4×4 submatrix of D given by its odd rows of its odd columns (corresponding to x coordinates) is equal to the submatrix given by the even rows of the even columns (corresponding to y coordinates). In these cases, we can write reduced versions of Equation (1) with 4×2 design matrices D that take a 2×2 input matrix with coordinates of a point as a row vector, rather than stacked column vectors as in Equation (1):

$$\phi = 0 \Rightarrow D \begin{bmatrix} \mathbf{x}_0^T \\ \mathbf{x}_3^T \end{bmatrix} = \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \end{bmatrix} \begin{bmatrix} x_0 & y_0 \\ x_3 & y_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\epsilon & 1+\epsilon \\ -(1+\epsilon) & 2+\epsilon \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 & y_0 \\ x_3 & y_3 \end{bmatrix} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \end{bmatrix}, \tag{15}$$

$$\phi = \pi \Rightarrow D \begin{bmatrix} \mathbf{x}_0^T \\ \mathbf{x}_3^T \end{bmatrix} = \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \end{bmatrix} \begin{bmatrix} x_0 & y_0 \\ x_3 & y_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 2+\epsilon & -(1+\epsilon) \\ 1+\epsilon & -\epsilon \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 & y_0 \\ x_3 & y_3 \end{bmatrix} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \end{bmatrix}, \tag{16}$$

where $\mathbf{x}^T$ denotes the transpose of $\mathbf{x}$. This simplifies the linear inverse design of linkages in "closure" configurations as both coordinates are handled simultaneously and indistinguishably. And, just as we showed for deployed configurations, we would like to prove that:

Proposition 2. *Prescribing any two points along with an offset ϵ in a closure linkage determines the full closure linkage.*

Proof. Again, we break each D into the six possible submatrices $d_{i,j}$ and calculate the rank of each.

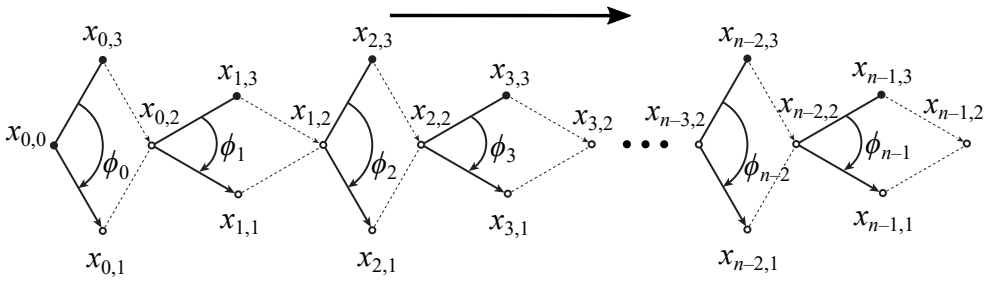
When $\phi = 0$, four of its submatrices ($d_{0,1}$, $d_{0,2}$, $d_{1,3}$ and $d_{2,3}$) are triangular and are generically full rank with rank loss occurring when $\epsilon = -1, -2, 0, -1$, respectively. The identity submatrix $d_{0,3}$ is trivially full rank. Lastly, the submatrix $d_{1,2}$ is full rank for all values of ϵ, which can be shown by the linear independence of its rows, as we have

$$k \begin{bmatrix} -\epsilon & 1+\epsilon \end{bmatrix} \neq \begin{bmatrix} -(1+\epsilon) & 2+\epsilon \end{bmatrix} \tag{17}$$

for any real number k. Similarly, when $\phi = \pi$, four of its submatrices ($d_{0,1}$, $d_{0,2}$, $d_{1,3}$ and $d_{2,3}$) are triangular and are generically full rank with rank loss occurring when $\epsilon = -1, 0, -2, -1$, respectively. The identity submatrix $d_{0,3}$ is trivially full rank. Lastly, the submatrix $d_{1,2}$ is full rank for all values of ϵ, which can be shown by the linear independence of its rows:

$$k \begin{bmatrix} 2+\epsilon & -(1+\epsilon) \end{bmatrix} \neq \begin{bmatrix} 1+\epsilon & -\epsilon \end{bmatrix} \tag{18}$$

for any real number k. So all 2×2 submatrices of both closure design matrices D are generically full rank and thus prescribing any two points in a closure linkage determines the full closure linkage with the exception of a small set of singular cases identified above. This completes the proof.



Supplementary Figure 2: **Linkage strip design.** Note that in the j-th linkage, the node $\mathbf{x}_{j,0} = \mathbf{x}_{j-1,2}$ depends on the $(j-1)$-th linkage. Therefore, the new nodes $\mathbf{x}_{j,1}$ and $\mathbf{x}_{j,2}$ are dependent of $\mathbf{x}_{j-1,2}$ as well as the node $\mathbf{x}_{j,3}$, the angle ϕ_j and the offset ϵ_j.

1.2 Linkage strips

We now apply this insight to calculate the design matrices for a strip of n linkages $\mathbf{x}_{j,k}$, where $j \in \{0, \ldots, n-1\}$ is the linkage index, $k \in \{0, 1, 2, 3\}$ is the point in linkage index and $\mathbf{x}_{j,2} = \mathbf{x}_{j+1,0}$ denotes a shared node (see Supplementary Figure 2). First, Let $E(j)$ be a $2 \times 2n$ block matrix comprised of the identity block in the j^{th} preceded by position $j-1$ zero blocks and followed by $n-j$ zero blocks:

$$E(j) = \begin{bmatrix} 0 & \cdots & 0 & I & 0 & \cdots & 0 \end{bmatrix}. \tag{19}$$

Then the base case is

$$D_{0,0} = E(0), \tag{20}$$
$$D_{j,3} = E(j+1), \tag{21}$$

and the recursive case is

$$D_{j,1} = G_j^{0,0} D_{j-1,2} + G_j^{0,1} D_{j,3}, \tag{22}$$
$$D_{j,2} = G_j^{1,0} D_{j-1,2} + G_j^{1,1} D_{j,3}, \tag{23}$$

where $G_j^{0,0}, G_j^{0,1}, G_j^{1,0}, G_j^{1,1}$ are the components of the following "generator" matrix

$$\begin{bmatrix} G_j^{0,0} & G_j^{0,1} \\ G_j^{1,0} & G_j^{1,1} \end{bmatrix} = \begin{bmatrix} I - (1+\epsilon_j)R(-\phi_j) & (1+\epsilon_j)R(-\phi_j) \\ -(1+\epsilon_j)R(-\phi_j) & I + (1+\epsilon_j)R(-\phi_j) \end{bmatrix}, \tag{24}$$

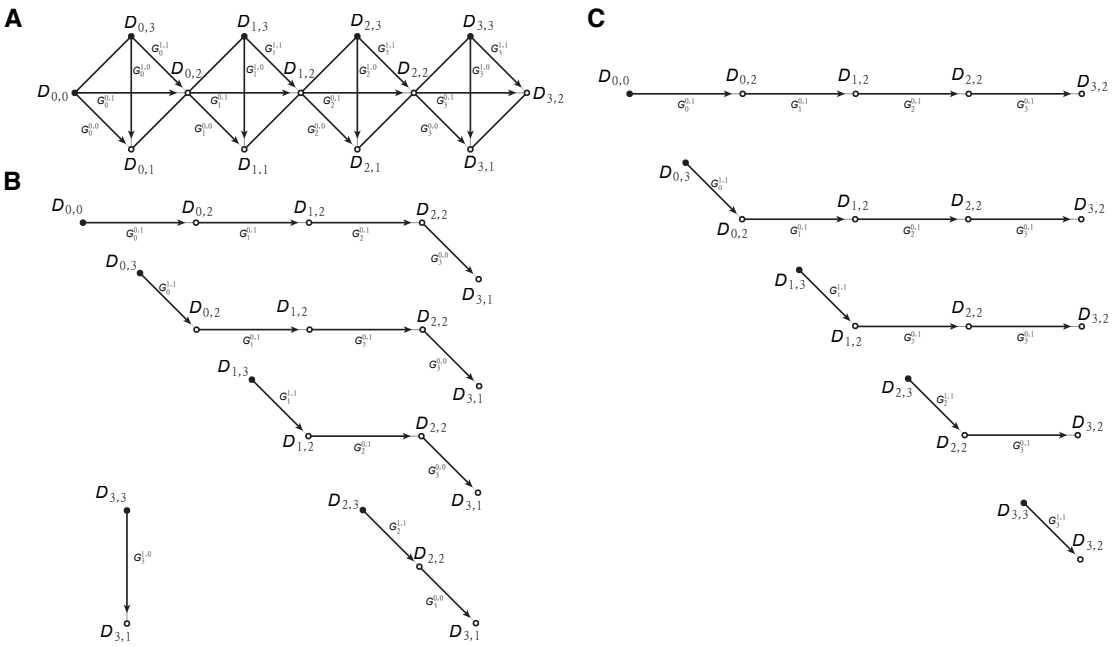
which takes two points in a linkage and produces the other two points in the linkage.

These rules define the full $(3n+1) \times (2n)$ linkage strip design matrix. The dynamic program defined above can be solved to give closed-form expressions for the dependent block rows:

$$D_{j,1} = G_j^{0,0} \Big(\prod_{y=j-1}^{0} G_y^{0,1} \Big) D_{0,0} + \sum_{z=0}^{j-1} G_j^{0,0} \Big(\prod_{y=j-1}^{z+1} G_y^{0,1} \Big) G_z^{0,0} D_{z,3} + G_j^{1,0} D_{j,3}, \tag{25}$$

$$D_{j,2} = \Big(\prod_{y=j}^{0} G_y^{0,1} \Big) D_{0,0} + \sum_{z=0}^{j} \Big(\prod_{y=j}^{z+1} G_y^{0,1} \Big) G_z^{1,0} D_{z,3}. \tag{26}$$

All other block rows correspond to seed input points and are therefore independent/base-case identity rows. It is helpful to look more closely at the terms that comprise Equation (26) to develop some intuition for the



Supplementary Figure 3: **Linkage strip example.** **(A)** Example strip of four linkages. Block design matrix rows $D_{j,k}$ corresponding to the k^{th} node of the j^{th} linkage $\mathbf{x}_{j,k}$ are labeled along with the generator components $G_j^{\{0,1\},\{0,1\}}$ which form directed edges in the paths from seed nodes to determined nodes. **(B)** The five generator product paths originating at seed nodes and terminating at $D_{3,1}$. **(C)** The five generator product paths originating at seed nodes and terminating at $D_{3,2}$.

relationships between seed positions and dependent positions. Note that the left expression is the only term involving left seed rows $D_{0,0}$ and the right expression produces a term for each of the top seed rows $D_{z,3}$ that $D_{j,2}$ depends on. Each of these terms can be viewed as a path through the structure that originates at a seed and terminates at the dependent node, collecting components of G as left products along the way. So a dependent node block row can be viewed as the sum of all the paths originating at the boundary seed nodes and terminating at that dependent node.

This heuristic will eventually form the foundation for our analysis of two-dimensional linkage arrays. Because the current example is a strip of linkages, though, there is only one path from each seed node to each determined node. This will change significantly when we look at linkage arrays, so it is instructive to look in detail at the calculation of linkage strips through this lens now. Consider a strip comprised of four linkages as shown in Supplementary Figure 3A. As an example, we are interested in the design matrix block rows $D_{3,1}$ and $D_{3,2}$ corresponding to the structure node $\mathbf{x}_{3,1}$ and $\mathbf{x}_{3,2}$, respectively. Applying Equation (26), we have

$$\begin{aligned} D_{3,1} = \; & G_3^{0,0}G_2^{0,1}G_1^{0,1}G_0^{0,1}D_{0,0} + \\ & G_3^{0,0}G_2^{0,1}G_1^{0,1}G_0^{1,1}D_{0,3} + \\ & G_3^{0,0}G_2^{0,1}G_1^{1,1}D_{1,3} + \\ & G_3^{0,0}G_2^{1,1}D_{2,3} + \\ & G_3^{1,0}D_{3,3}, \end{aligned} \tag{27}$$

which shows the five generator product terms that correspond to the five paths connecting seed nodes to $\mathbf{x}_{3,1}$

(Supplementary Figure 3B). Similarly, we have

$$\begin{aligned} D_{3,2} = \; & G_3^{0,1} G_2^{0,1} G_1^{0,1} G_0^{0,1} D_{0,0} + \\ & G_3^{0,1} G_2^{0,1} G_1^{0,1} G_0^{1,1} D_{0,3} + \\ & G_3^{0,1} G_2^{0,1} G_1^{1,1} D_{1,3} + \\ & G_3^{0,1} G_2^{1,1} D_{2,3} + \\ & G_3^{1,1} D_{3,3}, \end{aligned} \tag{28}$$

which shows the five generator product terms that correspond to the five paths connecting seed nodes to $\mathbf{x}_{3,2}$ (Supplementary Figure 3C). It is easy to see how this process generalizes to calculate any $D_{j,\{1,2\}}$ in a strip design matrix.

Now we can put all of these pieces together to calculate the full linkage strip design matrix S:

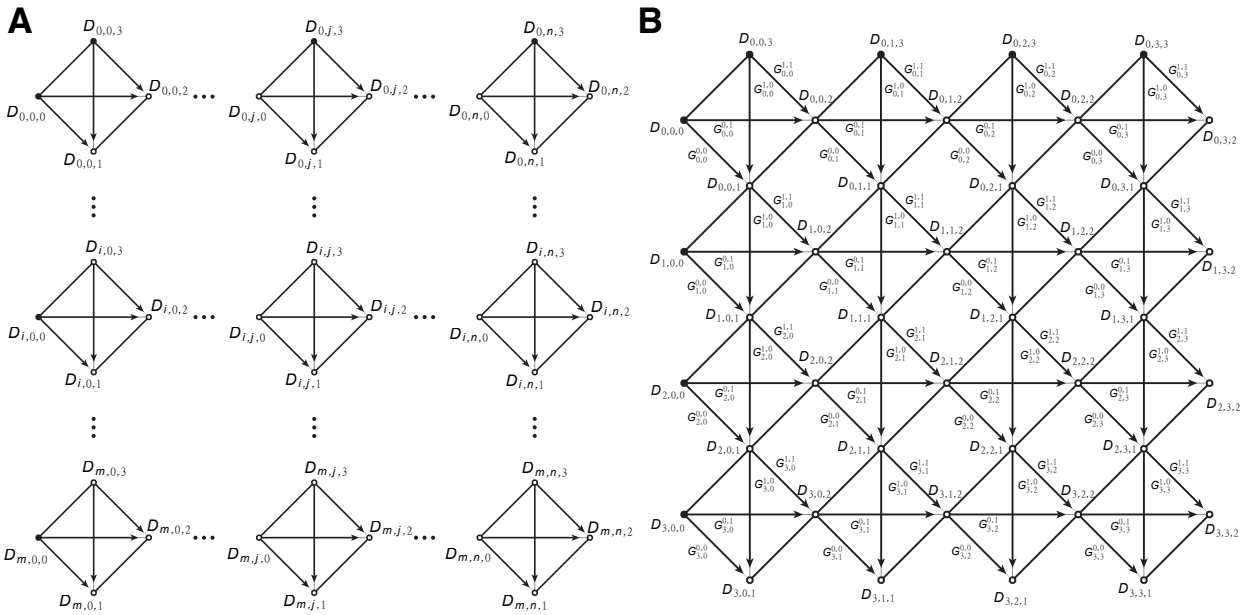
$$S = \begin{bmatrix} D_{0,0} \\ D_{0,1} \\ D_{0,2} \\ D_{0,3} \\ D_{1,1} \\ D_{1,2} \\ D_{1,3} \\ D_{2,1} \\ D_{2,2} \\ D_{2,3} \\ \vdots \\ D_{n-1,1} \\ D_{n-1,2} \\ D_{n-1,3} \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & 0 & \\ G_0^{0,0} & G_0^{0,1} & 0 & 0 & \\ G_0^{1,0} & G_0^{1,1} & 0 & 0 & \\ 0 & I & 0 & 0 & \\ G_1^{0,0} G_0^{0,0} & G_1^{0,0} G_0^{0,1} & G_1^{0,1} & 0 & \cdots \\ G_1^{1,0} G_0^{1,0} & G_1^{1,0} G_0^{1,1} & G_1^{1,1} & 0 & \\ 0 & 0 & I & 0 & \\ G_2^{0,0} G_1^{1,0} G_0^{1,0} & G_2^{0,0} G_1^{1,0} G_0^{1,1} & G_2^{0,0} G_1^{1,1} & G_2^{0,1} & \\ G_2^{1,0} G_1^{1,0} G_0^{1,0} & G_2^{1,0} G_1^{1,0} G_0^{1,1} & G_2^{1,0} G_1^{1,1} & G_2^{1,1} & \\ 0 & 0 & 0 & I & \\ & \vdots & & & \ddots \end{bmatrix}. \tag{29}$$

The entire linkage strip is then given by

$$S \begin{bmatrix} \mathbf{x}_{0,0} \\ \mathbf{x}_{0,3} \\ \mathbf{x}_{1,3} \\ \mathbf{x}_{2,3} \\ \vdots \\ \mathbf{x}_{n-1,3} \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & 0 & \\ G_0^{0,0} & G_0^{0,1} & 0 & 0 & \\ G_0^{1,0} & G_0^{1,1} & 0 & 0 & \\ 0 & I & 0 & 0 & \\ G_1^{0,0} G_0^{0,0} & G_1^{0,0} G_0^{0,1} & G_1^{0,1} & 0 & \cdots \\ G_1^{1,0} G_0^{1,0} & G_1^{1,0} G_0^{1,1} & G_1^{1,1} & 0 & \\ 0 & 0 & I & 0 & \\ G_2^{0,0} G_1^{1,0} G_0^{1,0} & G_2^{0,0} G_1^{1,0} G_0^{1,1} & G_2^{0,0} G_1^{1,1} & G_2^{0,1} & \\ G_2^{1,0} G_1^{1,0} G_0^{1,0} & G_2^{1,0} G_1^{1,0} G_0^{1,1} & G_2^{1,0} G_1^{1,1} & G_2^{1,1} & \\ 0 & 0 & 0 & I & \\ & \vdots & & & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{x}_{0,0} \\ \mathbf{x}_{0,3} \\ \mathbf{x}_{1,3} \\ \mathbf{x}_{2,3} \\ \vdots \\ \mathbf{x}_{n-1,3} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{0,0} \\ \mathbf{x}_{0,1} \\ \mathbf{x}_{0,2} \\ \mathbf{x}_{0,3} \\ \mathbf{x}_{1,1} \\ \mathbf{x}_{1,2} \\ \mathbf{x}_{1,3} \\ \mathbf{x}_{2,1} \\ \mathbf{x}_{2,2} \\ \mathbf{x}_{2,3} \\ \vdots \\ \mathbf{x}_{n-1,1} \\ \mathbf{x}_{n-1,2} \\ \mathbf{x}_{n-1,3} \end{bmatrix}. \tag{30}$$

1.3 Linkage arrays

After getting the linkage strip design matrix, we proceed to consider linkage arrays (Supplementary Figure 4A). Similar to the case of linkage strips, in an array of four-bar linkages, there will be multiple paths connecting



Supplementary Figure 4: **The relationship between nodes in a linkage array. (A)**, An $(m+1)\times(n+1)$ linkage array. **(B)**, A 4×4 linkage array with all paths connecting seed nodes to each point in the array. Here, $D_{i,j,0}, D_{i,j,1}, D_{i,j,2}, D_{i,j,3}$ are the block design matrix rows, and $\{G^{0,0}_{i,j}, G^{0,1}_{i,j}, G^{1,0}_{i,j}, G^{1,1}_{i,j}\}$ are the generator components.

seed nodes to each point in the array (see Supplementary Figure 4B). Although the relationship between the nodes becomes more complicated in two-dimensional linkage arrays, the general idea, which hopefully the reader now has a more concrete intuition for, remains the same: **generator matrix product paths originating at seed nodes (i.e. seed design matrix block rows) sum to determine other nodes (i.e. other design matrix block rows) in the linkage structure.**

Ultimately, we obtain the full linkage array design matrix M as described in the main text:

$$M\begin{bmatrix}\mathbf{x}^0_{0,0}\\ \mathbf{x}^0_{0,3}\\ \mathbf{x}^0_{1,3}\\ \vdots\\ \mathbf{x}^0_{n-1,3}\\ \mathbf{x}^1_{0,0}\\ \mathbf{x}^2_{0,0}\\ \vdots\\ \mathbf{x}^{m-1}_{0,0}\end{bmatrix} = \begin{bmatrix}D^0_0\\ D^0_1\\ \vdots\\ D^0_{n-1}\\ D^1_0\\ D^1_1\\ \vdots\\ D^{m-1}_{n-2}\\ D^{m-1}_{n-1}\end{bmatrix}\begin{bmatrix}\mathbf{x}^0_{0,0}\\ \mathbf{x}^0_{0,3}\\ \mathbf{x}^0_{1,3}\\ \vdots\\ \mathbf{x}^0_{n-1,3}\\ \mathbf{x}^1_{0,0}\\ \mathbf{x}^2_{0,0}\\ \vdots\\ \mathbf{x}^{m-1}_{0,0}\end{bmatrix} = \begin{bmatrix}I & 0 & 0 & 0 & \\ G^{0,0}_{0,0} & G^{0,1}_{0,0} & 0 & 0 & \\ G^{1,0}_{0,0} & G^{1,1}_{0,0} & 0 & 0 & \\ 0 & I & 0 & 0 & \\ G^{0,0}_{0,1}G^{0,0}_{0,0} & G^{0,0}_{0,1}G^{0,1}_{0,0} & G^{0,1}_{0,1} & 0 & \cdots\\ G^{1,0}_{0,1}G^{1,0}_{0,0} & G^{1,0}_{0,1}G^{1,1}_{0,0} & G^{1,1}_{0,1} & 0 & \\ 0 & 0 & I & 0 & \\ G^{0,0}_{0,2}G^{1,0}_{0,1}G^{1,0}_{0,0} & G^{0,0}_{0,2}G^{1,0}_{0,1}G^{1,1}_{0,0} & G^{0,0}_{0,2}G^{1,1}_{0,1} & G^{0,1}_{0,2} & \\ G^{1,0}_{0,2}G^{1,0}_{0,1}G^{1,0}_{0,0} & G^{1,0}_{0,2}G^{1,0}_{0,1}G^{1,1}_{0,0} & G^{1,0}_{0,2}G^{1,1}_{0,1} & G^{1,1}_{0,2} & \\ 0 & 0 & 0 & I & \\ & \vdots & & & \ddots\end{bmatrix}\begin{bmatrix}\mathbf{x}^0_{0,0}\\ \mathbf{x}^0_{0,3}\\ \mathbf{x}^0_{1,3}\\ \vdots\\ \mathbf{x}^0_{n-1,3}\\ \mathbf{x}^1_{0,0}\\ \mathbf{x}^2_{0,0}\\ \vdots\\ \mathbf{x}^{m-1}_{0,0}\end{bmatrix}, \tag{31}$$

where $\{\mathbf{x}^i_{j,k}\}$ are the coordinates of the nodes in the linkage array (with $i\in\{0,1,\ldots,m-1\}, j\in\{0,1,\ldots,n-$

$1\}, k \in \{0,1,2,3\})$, D_j^i are the linkage design matrices, and $\{G_{i,j}^{0,0}, G_{i,j}^{0,1}, G_{i,j}^{1,0}, G_{i,j}^{1,1}\}$ form the generator matrix

$$\begin{bmatrix} G_{i,j}^{0,0} & G_{i,j}^{0,1} \\ G_{i,j}^{1,0} & G_{i,j}^{1,1} \end{bmatrix} = \begin{bmatrix} I - Q_j^i & Q_j^i \\ -Q_j^i & I + Q_j^i \end{bmatrix} = \begin{bmatrix} I - (1+\epsilon_j^i)R(-\phi_j^i) & (1+\epsilon_j^i)R(-\phi_j^i) \\ -(1+\epsilon_j^i)R(-\phi_j^i) & I + (1+\epsilon_j^i)R(-\phi_j^i) \end{bmatrix}, \tag{32}$$

with $Q_j^i = (1+\epsilon_j^i)R(-\phi_j^i)$ being the scaled rotation matrix for the linkage (i,j).

Specifically, for the first linkage $(i,j) = (0,0)$, we have

$$D_0^0 = \begin{bmatrix} D_{0,0,0} \\ D_{0,0,1} \\ D_{0,0,2} \\ D_{0,0,3} \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & \cdots \\ G_{0,0}^{0,0} & G_{0,0}^{0,1} & 0 & \cdots \\ G_{0,0}^{1,0} & G_{0,0}^{1,1} & 0 & \cdots \\ 0 & I & 0 & \cdots \end{bmatrix}. \tag{33}$$

For the remaining linkages in the first strip, i.e. $(i,j) = (0,j)$, $j = 1, 2, \ldots, n-1$, we have

$$D_j^0 = \begin{bmatrix} D_{0,j,1} \\ D_{0,j,2} \\ D_{0,j,3} \end{bmatrix} \tag{34}$$

where

$$D_{0,j,1} = G_{0,j}^{0,0}\Big(\prod_{y=j-1}^{0} G_{0,y}^{0,1}\Big) D_{0,0,0} + \sum_{z=0}^{j-1} G_{0,j}^{0,0}\Big(\prod_{y=j-1}^{z+1} G_{0,y}^{0,1}\Big) G_{0,z}^{0,0} D_{0,z,3} + G_{0,j}^{1,0} D_{0,j,3}, \tag{35}$$

$$D_{0,j,2} = \Big(\prod_{y=j}^{0} G_{0,y}^{0,1}\Big) D_{0,0,0} + \sum_{z=0}^{j} \Big(\prod_{y=j}^{z+1} G_{0,y}^{0,1}\Big) G_{0,z}^{1,0} D_{0,z,3}, \tag{36}$$

$$D_{0,j,3} = \begin{bmatrix} 0 & \cdots & 0 & I & 0 & \cdots & 0 \end{bmatrix}, \tag{37}$$

with the identity block I located at j-th block column. Here, note that we do not need the block row $D_{0,j,0}$ in D_j^0 as it is identical to the block row $D_{0,j-1,2}$, which is already encoded in the previous linkage design matrix D_{j-1}^0.

Starting from the second strip, i.e. $i \geq 1$, we have

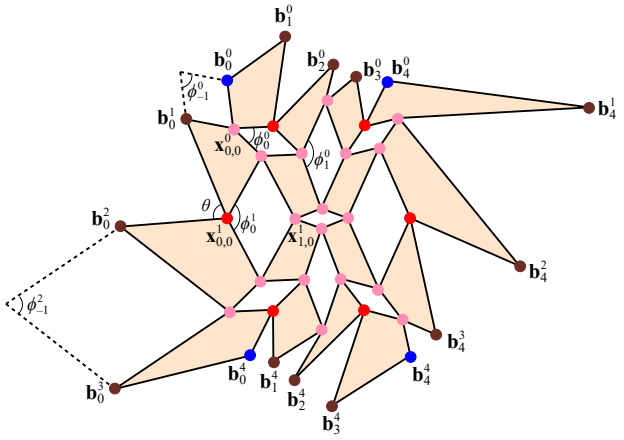
$$D_0^i = \begin{bmatrix} D_{i,0,0} \\ D_{i,0,1} \\ D_{i,0,2} \end{bmatrix} \quad \text{and} \quad D_j^i = \begin{bmatrix} D_{i,j,1} \\ D_{i,j,2} \end{bmatrix},\ j = 1, \ldots, n-1, \tag{38}$$

where $D_{i,0,0}$ is of the form $\begin{bmatrix} 0 & \cdots & 0 & I & 0 & \cdots & 0 \end{bmatrix}$, and $D_{i,0,1}, D_{i,0,2}, D_{i,j,1}, D_{i,j,2}$ are determined using dynamic programming. Again, here we do not need the block row $D_{i,0,3}$ in D_0^i as it is identical to $D_{i-1,0,1}$ in the linkage design matrix D_0^{i-1}. Similarly, we do not need the block rows $D_{i,j,0}$ and $D_{i,j,3}$ in D_j^i as they are identical to $D_{i,j-1,2}$ in D_{j-1}^i and $D_{i-1,j,1}$ in D_j^{i-1} respectively.

Altogether, M consists of $4 + 3(n-1) + (m-1)(3 + 2(n-1)) = 2mn + m + n$ block rows, where each block row is a $2 \times 2(m+n)$ matrix. Therefore, M is a $(4mn + 2m + 2n) \times (2m + 2n)$ matrix.

1.4 Determining the boundary node positions in the final kirigami pattern

As described in the main text, after setting up the design matrix, fixing the position of $2m + 2n$ nodes in the linkage array appropriately will uniquely determine the entire $m \times n$ linkage array. However, in the corresponding $(m+1) \times (n+1)$ kirigami pattern, the boundary nodes that are not included in the linkage



Supplementary Figure 5: **Determining the boundary node positions for the kirigami pattern.** We first fix the position of the red linkage boundary nodes and use the design matrix equation to uniquely determine all the remaining nodes in the linkage array (pink). Then, we can further prescribe four boundary node positions for the kirigami pattern (blue) and a set of boundary offsets, which uniquely determine all the remaining boundary nodes (brown) in the kirigami pattern. Specifically, we can use the prescribed corner position $\mathbf{b}_0^0$, the information $\mathbf{x}_{0,0}^0$ and ϕ_{-1}^0 together with a chosen boundary offset ϵ_b to form a ghost four-bar linkage containing $\mathbf{b}_0^0$, $\mathbf{b}_0^1$, $\mathbf{x}_{0,0}^0$ (dotted lines), thereby uniquely determining $\mathbf{b}_0^1$. We can then use $\mathbf{b}_0^1$ together with the obtained linkage array and another chosen boundary offset to uniquely determine $\mathbf{b}_0^2$, and then continue the above process to determine the next left boundary node $\mathbf{b}_0^3$ and so on. After that, we can start with another prescribed corner $\mathbf{b}_0^4$ to determine the bottom boundary nodes $\mathbf{b}_1^4, \mathbf{b}_2^4, \mathbf{b}_3^4$ one by one. Similarly, the right boundary nodes and the top boundary nodes can be uniquely determined using the prescribed corners $\mathbf{b}_4^4$ and $\mathbf{b}_0^4$ together with the chosen boundary offsets.

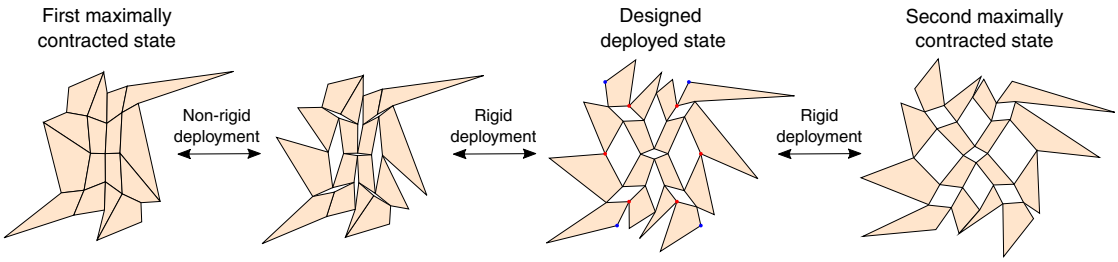
array are still not fixed. To determine the coordinates of those nodes, it suffices to further prescribe four corner positions and a set of boundary offsets. To see this, consider the $(m+1)\times(n+1) = 4\times 4$ kirigami pattern in Supplementary Figure 5. Once the red linkage boundary nodes are prescribed, the 3×3 linkage array (consisting of the red and pink nodes) are uniquely determined by the matrix equation. Suppose the positions of the four corners $\mathbf{b}_0^0, \mathbf{b}_4^0, \mathbf{b}_0^4, \mathbf{b}_4^4$ (the blue nodes) are further prescribed.

Now, we use the prescribed position $\mathbf{b}_0^0$ to determine the position of its neighboring boundary node $\mathbf{b}_0^1$. Recall that in the deployment angle field constraints introduced in the main text, ϕ_{-1}^0 is a ghost point for the angle constraint $\phi_{-1}^0 + 2\phi_0^0 + \phi_1^0 = 2\pi$ at $(i,j) = (0,0)$ which has already been prescribed when setting the design matrix earlier. One can therefore consider a ghost four-bar linkage involving $\mathbf{b}_0^0, \mathbf{b}_0^1, \mathbf{x}_{0,0}^0$ with ϕ_{-1}^0 being the deployment angle of it (see dotted lines). It is then natural to introduce a *boundary offset* ϵ_b analogous to the linkage offset ϵ_j^i, which relates the two edge lengths $\|\mathbf{b}_0^0 - \mathbf{x}_{0,0}^0\|$ and $\|\mathbf{b}_0^1 - \mathbf{x}_{0,0}^0\|$ by

$$\|\mathbf{b}_0^0 - \mathbf{x}_{0,0}^0\| = (1+\epsilon_b)\|\mathbf{b}_0^1 - \mathbf{x}_{0,0}^0\|. \tag{39}$$

In particular, setting $\epsilon_b = 0$ means that we would like the two edges to be equal in length. Altogether, using the prescribed corner position $\mathbf{b}_0^0$, the prescribed boundary offset ϵ_b, as well as the information $\mathbf{x}_{0,0}^0$ and ϕ_{-1}^0 obtained from the linkage array, we can uniquely determine $\mathbf{b}_0^1$.

Similarly, we can introduce another boundary offset parameter for the ratio between $\|\mathbf{b}_0^1 - \mathbf{x}_{0,0}^0\|$ and $\|\mathbf{b}_0^2 - \mathbf{x}_{0,0}^0\|$ and use that to determine $\mathbf{b}_0^2$ uniquely. This time, note that the two tile angles at $\mathbf{x}_{0,0}^1$ and the two tile angles at $\mathbf{x}_{1,0}^0$ should add up to 2π. As the two tile angles at $\mathbf{x}_{1,0}^0$ are uniquely determined in the obtained linkage array, the sum of the two tile angles at $\mathbf{x}_{0,0}^1$ is known. Also, the deployment angle ϕ_0^1 is given by the linkage array. Therefore, the angle $\theta = \angle(\mathbf{b}_0^1, \mathbf{x}_{0,0}^0, \mathbf{b}_0^2)$ is uniquely determined, and hence one



Supplementary Figure 6: **An example of contractible but not fully rigid-deployable patterns obtained by our framework.** In the construction of the kirigami pattern, all slits are assumed to be parallelograms and the contractible angle constraint is satisfied as in main text Figure 2B. If we have $\epsilon = 0$ for all linkages, then the resulting pattern will only be rigid-deployable within a certain range in the deployment process.

can uniquely determine $\mathbf{b}_0^2$.

The above process can be continued to determine all left boundary nodes $\mathbf{b}_0^1, \mathbf{b}_0^2, \ldots, \mathbf{b}_0^m$ using the prescribed corner position $\mathbf{b}_0^0$ and the prescribed boundary offsets. Similarly, one can use the prescribed corner positions $\mathbf{b}_0^{m+1}, \mathbf{b}_{n+1}^{m+1}, \mathbf{b}_{n+1}^0$ to uniquely determine all bottom boundary nodes $\mathbf{b}_1^{m+1}, \mathbf{b}_2^{m+1}, \ldots, \mathbf{b}_n^{m+1}$, right boundary nodes $\mathbf{b}_{n+1}^m, \mathbf{b}_{n+1}^{m-1}, \ldots, \mathbf{b}_{n+1}^1$, and top boundary nodes $\mathbf{b}_n^0, \mathbf{b}_{n-1}^0, \ldots, \mathbf{b}_1^0$, respectively. Altogether, prescribing four corner positions as well as the boundary offsets uniquely determines the final kirigami pattern. In our experiments, we simply set $\epsilon_b = 0$ for all boundary nodes to avoid the occurrence of zig-zag boundary shapes in the contracted state.

1.5 Rigid-deployability

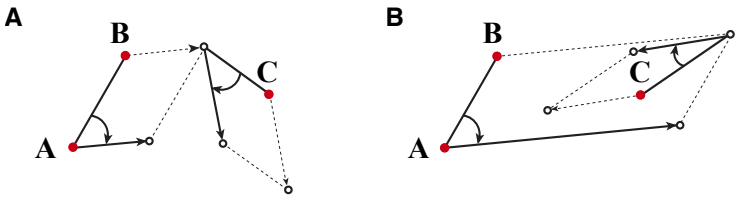
As described in the main text, the parallelogram four-bar linkage setup does not necessarily yield a fully rigid-deployable pattern. Recall that in our approach, there are degrees of freedom in the deployment angle field so that we can specify the state of deployment of the resulting pattern. In the designed deployed pattern, every slit (i, j) forms a parallelogram with angles in the form $(\phi_j^i, \pi - \phi_j^i, \phi_j^i, \pi - \phi_j^i)$ without degeneracy if we have $0 < \phi_j^i < \pi$. However, we may only be able to rigidly morph the pattern from the designed deployed state within a certain range in the deployment process. In other words, the above $(\theta, \pi - \theta, \theta, \pi - \theta)$ angle condition may only hold within a certain range in the deployment process.

For instance, consider the contractible pattern in main text Figure 2B. As shown in the deployment process in Supplementary Figure 6, the designed deployed pattern is only rigid-deployable within a certain range in the deployment process. To further contract the pattern from the designed deployed state to a closed and compact state, we will have to snap some of the parallelogram slits into closed V-shaped slits, which involves a non-rigid process.

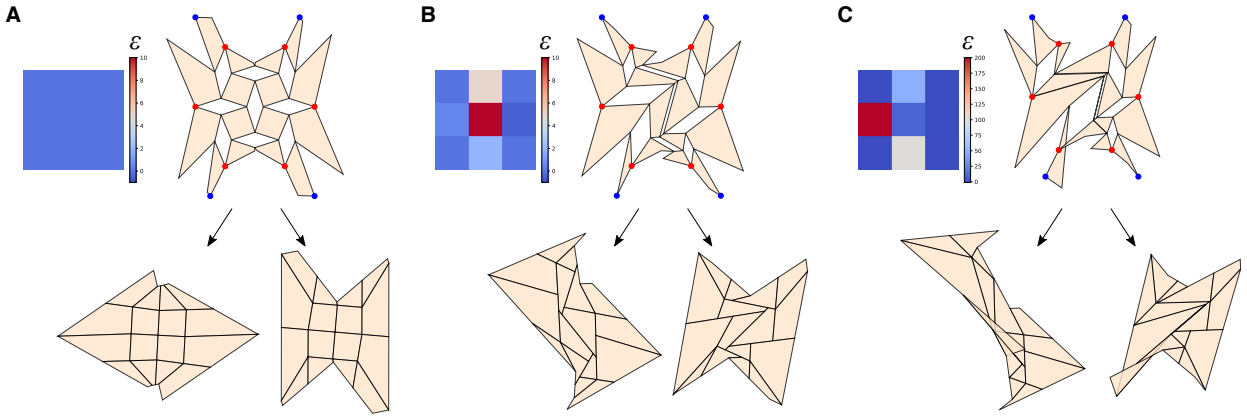
1.6 Self-intersection

As mentioned in the main text, the offsets $\{\epsilon_j^i\}$ of all linkages can be chosen independently as long as they do not lead to degeneracies or self-intersections. In Section 1.1, we have shown that one should avoid setting $\epsilon = -1$ or $\phi > \pi$ for each linkage. Here, we perform a more detailed analysis to characterize the conditions for self-intersection and how to prevent this.

Consider two adjacent four-bar linkages with fixed seed node positions A, B, C as shown in Supplementary Figure 7. Suppose the deployment angle of the first linkage is also fixed. As shown in Supplementary Figure 7A, one can achieve a pattern without any self-intersection if the offset parameter ϵ for the first



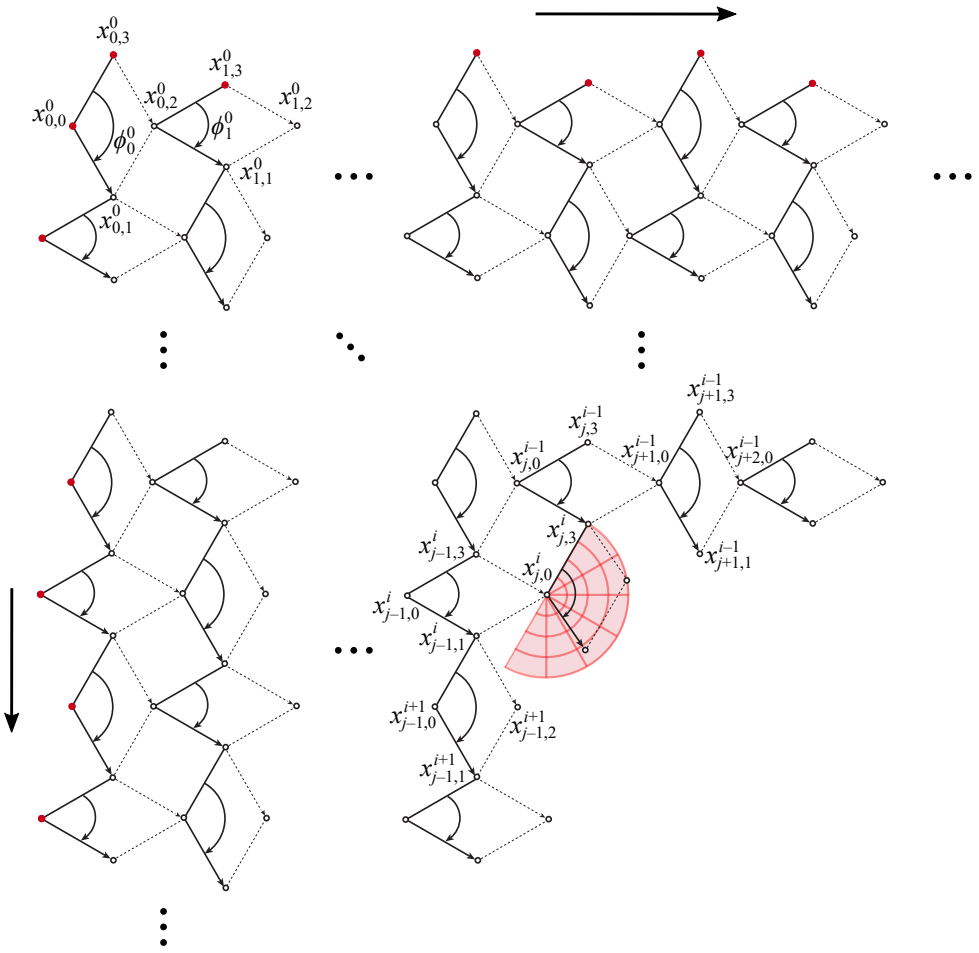
Supplementary Figure 7: **An illustration of potential self-intersections.** Here, A, B, C are the fixed seed node positions. It can be observed that by changing the offset parameter (and hence the side length ratio) for the first linkage, one can get **(A)** a pattern without self-intersection or **(B)** a pattern with self-intersection.



Supplementary Figure 8: **Three kirigami patterns with different offset parameters. (A)** The compact reconfigurable kirigami pattern in main text Figure 2B, with the offset parameters $\epsilon = 0$ everywhere. **(B)** Another compact reconfigurable kirigami pattern created using a different set of offset parameters ϵ while keeping the boundary node constraints (highlighted in red), corner constraints (highlighted in blue) and deployment angle field unchanged. Note that there is no self-intersection in the configurations at different states. **(C)** A pattern created using more extreme values of ϵ. Self-intersections can be observed at different deployment states.

linkage is sufficiently small. By contrast, if a large ϵ is used so that the first four-bar linkage contains point C, then self-intersection will occur as shown in Supplementary Figure 7B, no matter what deployment angle and offset parameters are used for the second linkage. This example suggests that the presence of self-intersections is closely related to the choice of the offset parameters. More generally, for an $m \times n$ linkage array, we can consider every pair of adjacent linkages and repeat the above analysis. In this case, note that the corresponding points A, B, C are determined by the previous linkages via the dynamic programming formulation. Therefore, the occurrence of self-intersections will be related to the ϵ parameters locally as well as the parameters for the previous linkages, which makes it difficult to give a universal bound of the parameters for preventing self-intersections. As an illustration, in Supplementary Figure 8 we consider changing the offset parameters of the compact reconfigurable kirigami pattern in main text Figure 2B. It can be observed that for some large values of ϵ, there can indeed be self-intersections in the resulting pattern at different deployment states. Nevertheless, we can establish a sufficient condition for getting kirigami patterns without any self-intersection as follows:

Theorem 1. *For any $m \times n$ linkage array, denote the nodes of the (i, j)-th linkage (where $0 \le i \le m-1$ and $0 \le j \le n-1$) as $\mathbf{x}^i_{j,0}, \mathbf{x}^i_{j,1}, \mathbf{x}^i_{j,2}, \mathbf{x}^i_{j,3}$ as shown in Supplementary Figure 9. Note that $\mathbf{x}^i_{j,2} = \mathbf{x}^i_{j+1,0}$ for all $j < n-1$ and $\mathbf{x}^i_{j,1} = \mathbf{x}^{i+1}_{j,3}$ all for $i < m-1$. Suppose $\epsilon^i_j > -1$ and the deployment angles satisfy $\phi^i_j \in [0, \pi]$*



Supplementary Figure 9: **An illustration of the bound on the offset parameters for ensuring no self-intersections.** Each parallelogram four-bar linkage consists of four vertices $\mathbf{x}^i_{j,0}, \mathbf{x}^i_{j,1}, \mathbf{x}^i_{j,2}, \mathbf{x}^i_{j,3}$, with $\mathbf{x}^i_{j,2} = \mathbf{x}^i_{j+1,0}$ for all $j < n-1$ and $\mathbf{x}^i_{j,1} = \mathbf{x}^{i+1}_{j,3}$ all for $i < m-1$. The nodes highlighted in red are the seed nodes.

for all (i, j). If the condition

$$\max\{1 + \epsilon^i_j, \sqrt{(1 + \epsilon^i_j + \cos\phi^i_j)^2 + \sin^2\phi^i_j}\} \leq \frac{d^i_j}{\|\mathbf{x}^i_{j,0} - \mathbf{x}^i_{j,3}\|} \tag{40}$$

is further satisfied for all (i, j), where d^i_j is the minimum distance between $\mathbf{x}^i_{j,0}$ and all edges in all linkages $(\tilde{i}, :)$ and $(:, \tilde{j})$ with $\tilde{i} < i$ and $\tilde{j} < j$ (excluding the edges that contain $\mathbf{x}^i_{j,0}$), then there is no self-intersection in the entire resulting pattern.

Proof. Consider the (i, j)-th linkage in the $m \times n$ linkage array. We first observe that:

- The two vertices $\mathbf{x}^i_{j,0}, \mathbf{x}^i_{j,3}$ in the (i, j)-th linkage are determined by the linkages $\{(p, q) : 0 \leq p \leq i-2 \text{ and } 0 \leq q \leq j-2\}$ (i.e. in the top left corner of the (i, j)-th linkage), $\{(i, q) : 0 \leq q \leq j-1\}$ (i.e. in the same column and above the (i, j)-th linkage) and $\{(p, j) : 0 \leq p \leq i-1\}$ (i.e. in the same row and on the left of the (i, j)-th linkage).

- All linkages $\{(p,q) : 0 \le p \le i-2 \text{ and } q \ge j\}$ (i.e. in the top right corner of the (i,j)-th linkage) are determined by the linkages $\{(p,q) : 0 \le p \le i-2 \text{ and } 0 \le q \le j-1\}$.
- All linkages $\{(p,q) : p \ge i \text{ and } 0 \le q \le j-2\}$ (i.e. in the bottom left corner of the (i,j)-th linkage) are determined by the linkages $\{(p,q) : 0 \le p \le i-1 \text{ and } 0 \le q \le j-2\}$.
- All linkages $\{(p,q) : p \ge i-1 \text{ and } q \ge j\}$ and $\{(p,q) : p \ge i \text{ and } q \ge j-1\}$ (i.e. on the right of the (i,j)-th linkage or below it) are dependent of the (i,j)-th linkage.

This suggests that to determine a suitable value of ϵ_j^i, we only need to consider the linkages $(\tilde{i},:)$ and $(:,\tilde{j})$ with $\tilde{i} < i$ and $\tilde{j} < j$, all of which are not affected by the choice of the parameters in the (i,j)-th linkage.

Now, we want to check whether the (i,j)-th linkage will intersect any other such linkages $(\tilde{i},:)$ and $(:,\tilde{j})$. By the convexity of the linkage, it suffices to consider whether any of the two new vertices $x_{j,1}^i, x_{j,2}^i$ lie inside some other linkages. Note that we have

$$\|x_{j,0}^i - x_{j,1}^i\| = \|x_{j,0}^i - x_{j,3}^i\|(1+\epsilon_j^i), \tag{41}$$

and by the law of cosines we have

$$\begin{aligned}
\|x_{j,0}^i - x_{j,2}^i\| &= \sqrt{\|x_{j,0}^i - x_{j,3}^i\|^2 + \|x_{j,0}^i - x_{j,1}^i\|^2 - 2\|x_{j,0}^i - x_{j,3}^i\|\|x_{j,0}^i - x_{j,1}^i\|\cos(\pi - \phi_j^i)} \\
&= \|x_{j,0}^i - x_{j,3}^i\|\sqrt{1 + (1+\epsilon_j^i)^2 + 2(1+\epsilon_j^i)\cos\phi_j^i} \\
&= \|x_{j,0}^i - x_{j,3}^i\|\sqrt{(1+\epsilon_j^i + \cos\phi_j^i)^2 + 1 - \cos^2\phi_j^i} \\
&= \|x_{j,0}^i - x_{j,3}^i\|\sqrt{(1+\epsilon_j^i + \cos\phi_j^i)^2 + \sin^2\phi_j^i}.
\end{aligned} \tag{42}$$

If the given condition on ϵ_j^i in Equation (40) is satisfied, we have

$$\|x_{j,0}^i - x_{j,1}^i\| = \|x_{j,0}^i - x_{j,3}^i\|(1+\epsilon_j^i) \le d_j^i \tag{43}$$

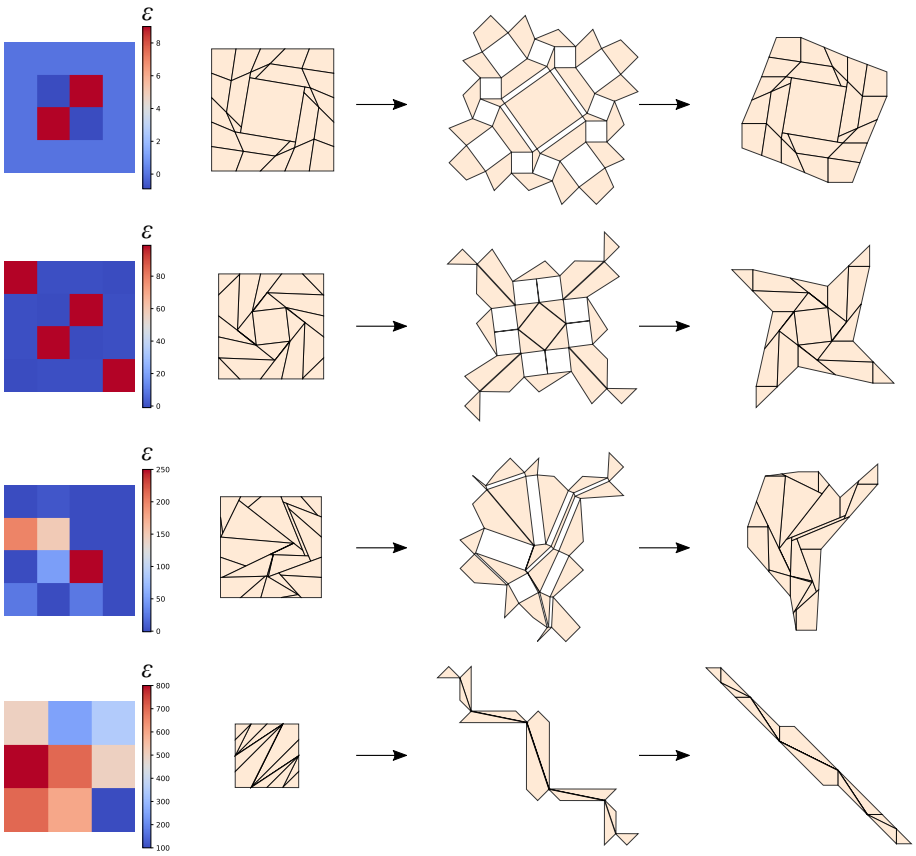
and

$$\|x_{j,0}^i - x_{j,2}^i\| = \|x_{j,0}^i - x_{j,3}^i\|\sqrt{(1+\epsilon_j^i + \cos\phi_j^i)^2 + \sin^2\phi_j^i} \le d_j^i. \tag{44}$$

Therefore, we can draw a semicircle centered at $\mathbf{x}_{j,0}^i$ with radius d_j^i as shown in Supplementary Figure 9, and the two new vertices $\mathbf{x}_{j,1}$ and $\mathbf{x}_{j,2}$ will always lie inside the semicircle. Hence, the (i,j)-th linkage will not lie inside any other linkages $(\tilde{i},:)$ or $(:,\tilde{j})$, where $\tilde{i} < i$ and $\tilde{j} < j$. Consequently, if the condition is satisfied for all (i,j), there is no self-intersection in the entire pattern. This completes the proof.

The above theorem indicates that a non-self-intersecting kirigami pattern can be obtained by choosing all ϵ_j^i to be sufficiently small. However, we remark that the above condition is sufficient but not necessary. For instance, there can be non-self-intersecting patterns generated using very large values of ϵ (see Supplementary Figure 10).

From a more practical perspective, note that $\epsilon = 99$ gives a $100:1$ side length ratio, while $\epsilon = 0.01$ gives a $1:100$ side length ratio. Such extreme values of ϵ are unlikely to be necessary in practical applications, and so it is reasonable for us to consider the offset parameters within a smaller range. Supplementary Figure 11 shows a gallery of rigid-deployable, compact reconfigurable heart structures obtained using our method with different offset parameters, including constant offsets and varying offsets in either $[-0.9, 0]$, $[-0.9, 1]$ or $[-0.9, 10]$. From the examples, it can be observed that one can already create a wide range of non-self-intersecting patterns with very different deployed and reconfigured shapes by considering only a small range of ϵ.



Supplementary Figure 10: **Rigid-deployable, compact reconfigurable square patterns obtained with different offset parameters.** For each pattern, the values of the offsets $\{\epsilon_j^i\}$ and three deployment snapshots at $\phi = 0$ (first contracted state), $\phi = \pi/2$ (fully deployed state), and $\phi = \pi$ (second contracted state) are shown. It can be observed that there is no self-intersection in all patterns.

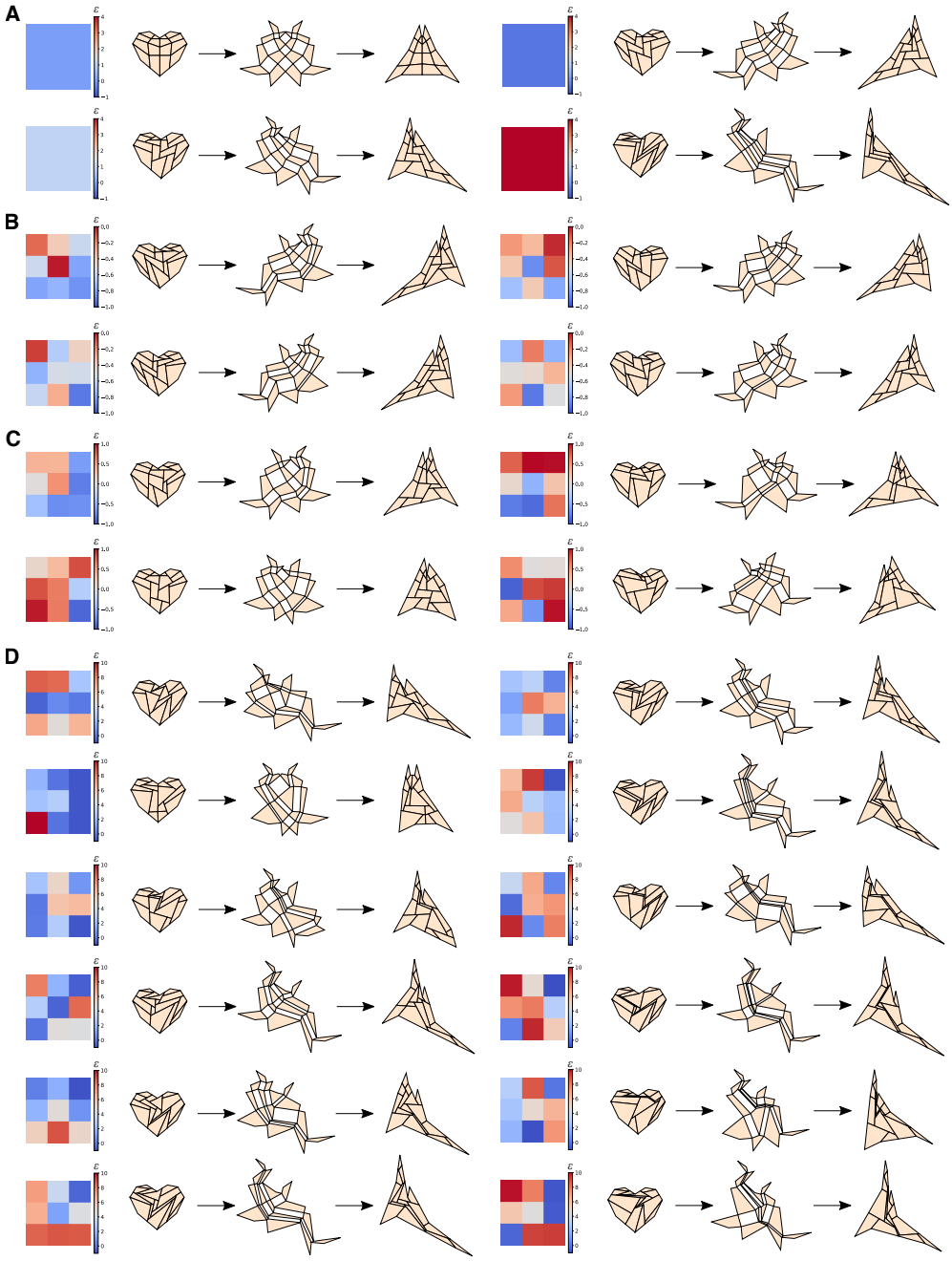
2 Nonlinear optimization

As shown in the main text, we can produce rigid-deployable, compact reconfigurable kirigami patterns that morph from a square to a prescribed reconfigured shape via nonlinear optimization.

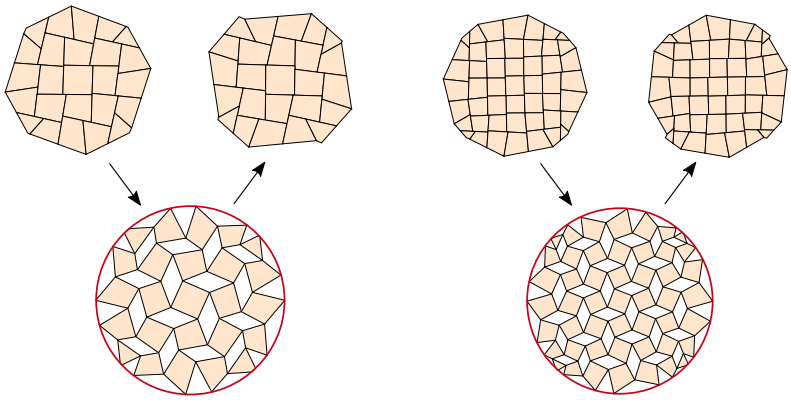
To obtain the square-to-circle pattern in main text Figure 3B, note that the deployment angle field $\{\phi_j^i\}$ of the second contracted state is already determined by one single angle $\phi = \pi$. Therefore, we can search for the optimal set of offset parameters $\mathcal{E} = \{\epsilon_j^i\}$ that yields a circular second contracted shape. More specifically, we optimize the circularity of the boundary points in the second contracted state by solving the following problem:

$$\min_{\mathcal{E}} \sum_{\mathbf{p}_i \in \mathcal{B}} \left(\|\mathbf{p}_i - \mathbf{c}\| - \text{mean}_j \|\mathbf{p}_j - \mathbf{c}\| \right)^2, \tag{45}$$

where $\mathcal{B}$ is the set of dual boundary nodes in the second contracted configuration and $\mathbf{c}$ is the center of the second contracted configuration. Alternatively, from the isoperimetric inequality we have $L^2 \geq 4\pi A$, where L is the length of any closed curve in $\mathbb{R}^2$ and A is the area of the region enclosed by it, and the equality holds if and only if the region is a circle. We can then optimize the circularity by minimizing $\left(\frac{L^2}{4\pi A} - 1\right)^2$ of the second contracted configuration.



Supplementary Figure 11: **A gallery of rigid-deployable, compact reconfigurable heart structures obtained using our linear inverse design method.** For each pattern, the values of the offsets $\{\epsilon_j^i\}$ and three deployment snapshots at $\phi = 0$ (first contracted state), $\phi = \pi/2$ (fully deployed state), and $\phi = \pi$ (second contracted state) are shown. **(A)** Examples with constant offsets $\epsilon = 0, -0.5, 1, 4$. **(B)** Examples with varying offsets in $[-0.9, 0]$. **(C)** Examples with varying offsets in $[-0.9, 1]$. **(D)** Examples with varying offsets in $[-0.9, 10]$.



Supplementary Figure 12: **Rigid-deployable, compact reconfigurable kirigami patterns that match a circle at a prescribed deployed state $\phi = \pi/4$.** The deployment paths of two patterns with different resolution are shown.

More generally, given a set of discrete points D representing the target shape, we can solve the following optimization problem to minimize the shape difference between the second contracted state and the target shape:

$$\min_{\mathcal{E}} \sum_{\mathbf{p}_i \in \mathcal{B}} \|\mathbf{p}_i - \mathrm{proj}_D \mathbf{p}_i\|^2, \tag{46}$$

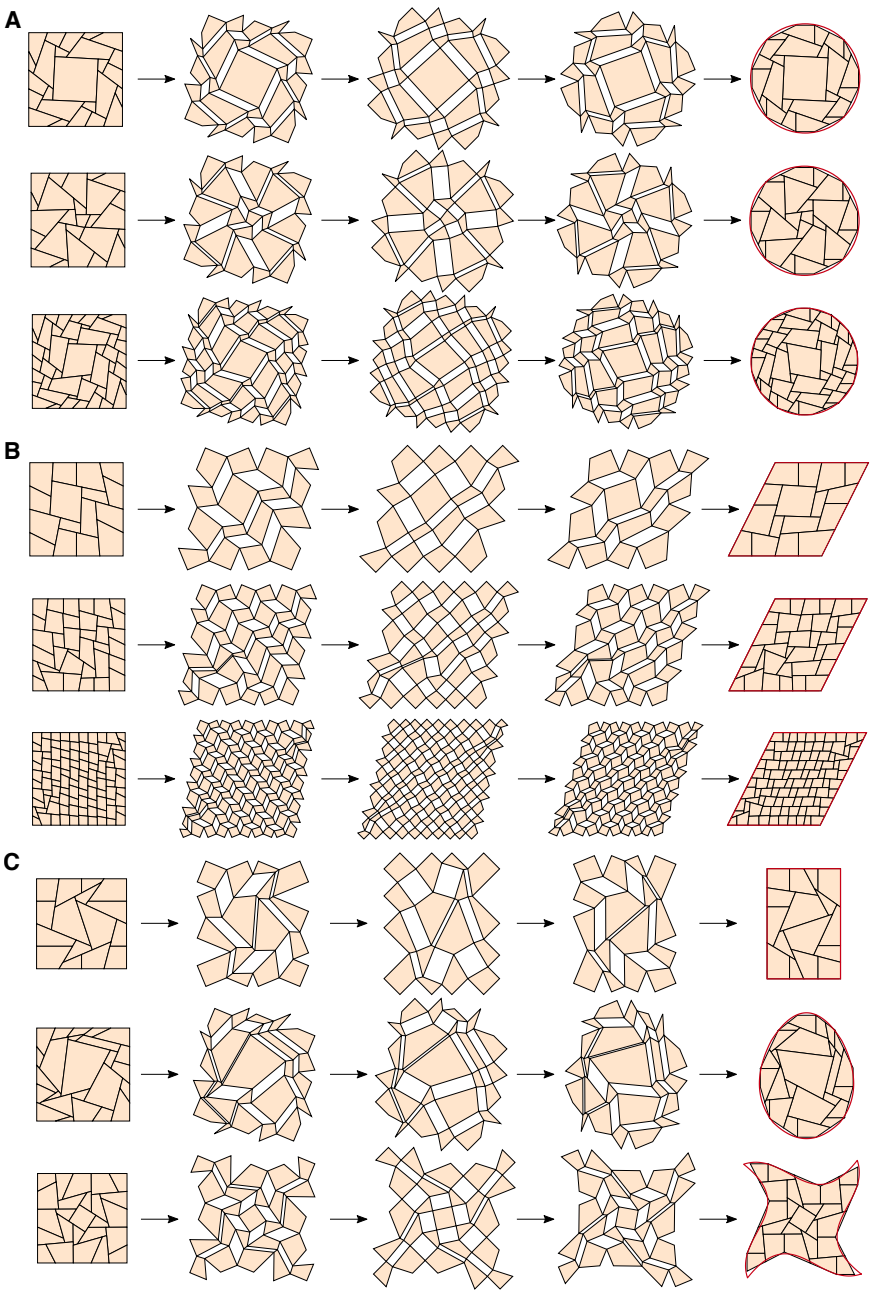
where $\mathrm{proj}_D \mathbf{p}_i$ is the projection of $\mathbf{p}_i$ onto the target shape D. This yields the patterns in Section 3 that morph from a square to a target second contracted shape.

3 Results

In main text Figure 3A, we showed a rigid-deployable, compact reconfigurable kirigami pattern that matches a circle at the prescribed deployed state $\phi = \pi/2$. In Supplementary Figure 12, we show two other patterns obtained by our additive design framework that match a circle at another prescribed deployed state $\phi = \pi/4$.

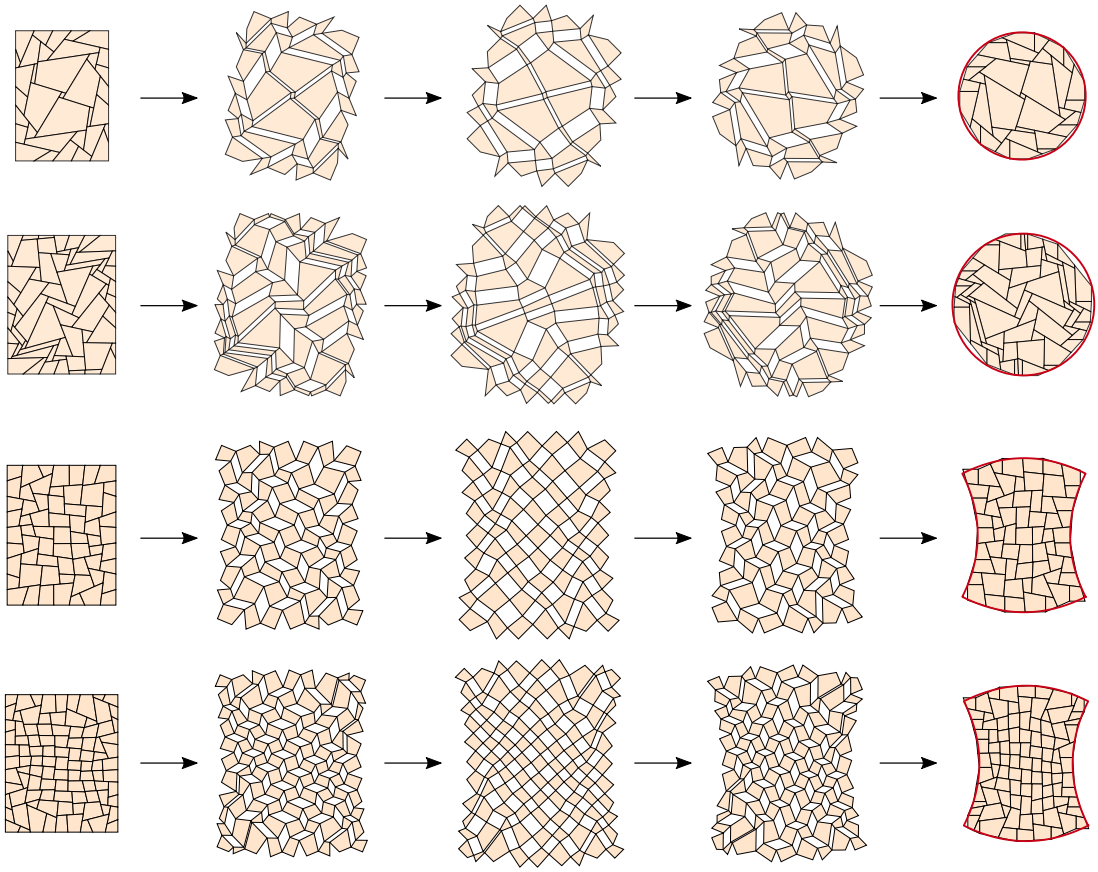
Supplementary Figure 13 shows more results of rigid-deployable, compact reconfigurable square kirigami patterns approximating a prescribed reconfigured shape obtained using our nonlinear optimization framework. Note that by using different pattern sizes and different optimization solvers (`scipy.optimize.least_squares` and `scipy.optimize.minimize` etc.), one can obtain various results that achieve the same shape change. For instance, Supplementary Figure 13A shows several patterns achieving a square-to-circle transformation. Here we remark that the angles at half of the boundary points in the second contracted state should always add up to π for rigid-deployable quad kirigami patterns, and hence the second contracted configuration cannot match a circle perfectly at those points. Nevertheless, by increasing the pattern resolution, the approximation becomes more accurate. By contrast, in case the target shape consists of zero curvature curves only (such as a parallelogram), it is possible to produce kirigami patterns that morph from a square to the target shape (Supplementary Figure 13B). Overall, as shown in Supplementary Figure 13C, our method is capable of producing kirigami patterns that morph from a square to different target shape with different curvature properties accurately.

Besides starting with a perfect square, we can also start with a rectangle and search for kirigami patterns achieving a prescribed shape change. Supplementary Figure 14 shows examples that morph from a rectangle to either a circle or a mixed curvature shape in the second contracted state. We remark that for the mixed curvature shape, the approximation near the four corners of the target shape is not perfect due to the

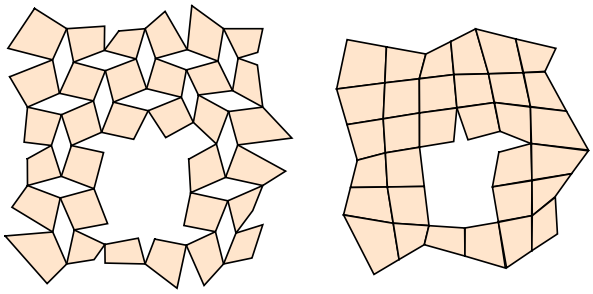


Supplementary Figure 13: **Rigid-deployable, compact reconfigurable kirigami patterns obtained using our framework.** For each example, the first contracted shape is enforced to be a perfect square and a target shape for the second contracted state is prescribed as the optimization objective (highlighted in red). **(A)** Various square-to-circle patterns. **(B)** Square-to-parallelogram patterns with different resolutions. **(C)** Kirigami patterns that morph from a square to a rectangle (zero curvature), an egg (positive curvature), and a star (mixed curvature) respectively.

rectangular shape constraint for the first contracted state.

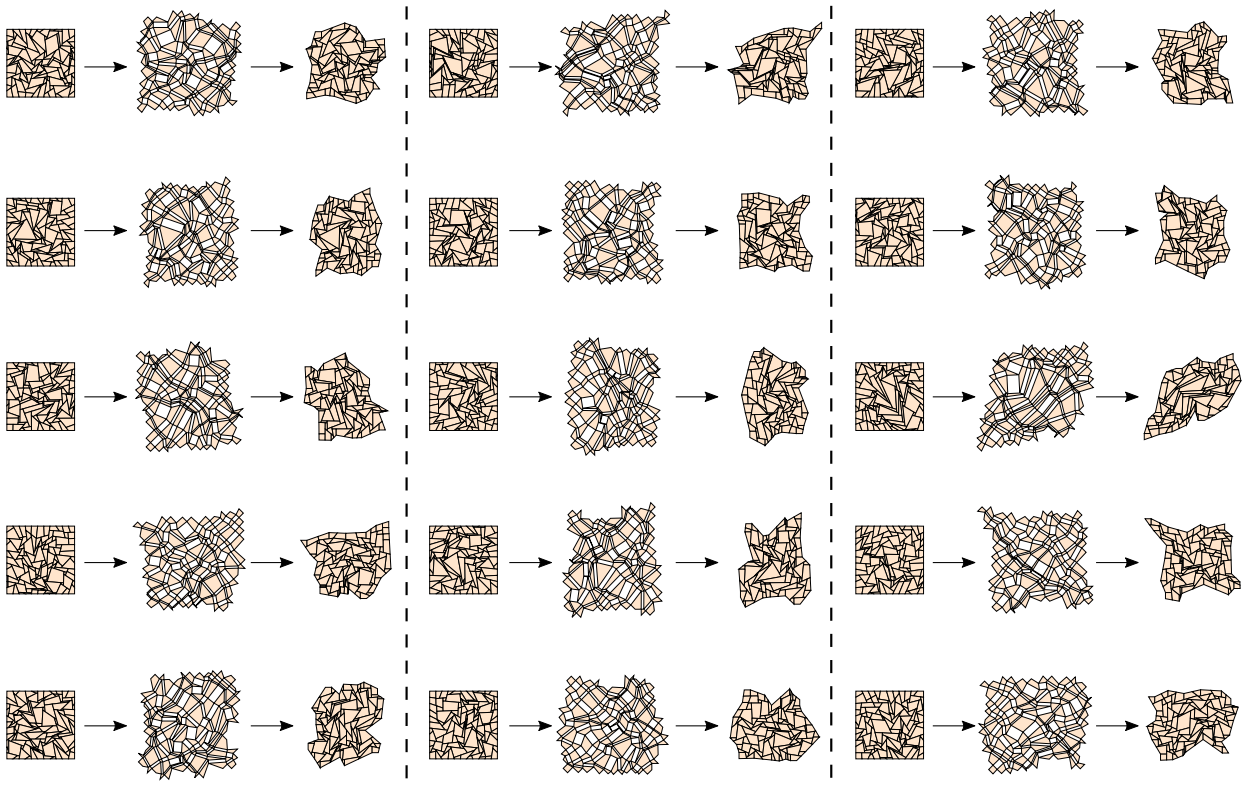


Supplementary Figure 14: **Rigid-deployable, compact reconfigurable kirigami patterns that morph from a rectangle to a prescribed target shape.**

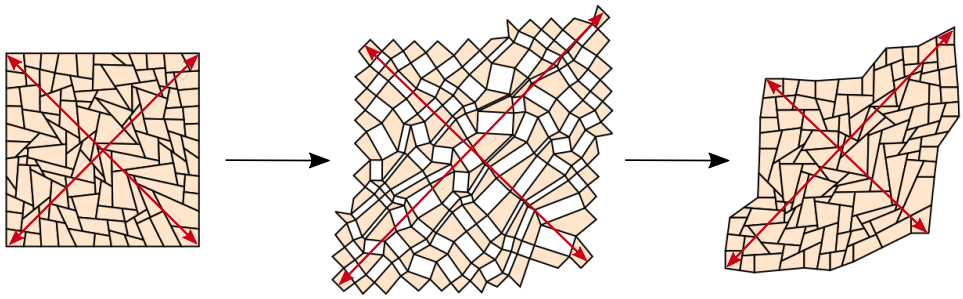


Supplementary Figure 15: **Designing kirigami patterns with partial linkage array.** (Left) A deployed kirigami pattern produced using our proposed approach, with no constraints enforced in the missing elements. (Right) The contracted state of the pattern.

While we have been considering full $m \times n$ linkage arrays so far, it is possible to apply the proposed approach for the case where the number of elements for the strips is not equal. Specifically, we can simply treat the missing linkages as virtual linkages without any constraints enforced in their length and angle parameters, and then construct the design matrix and create a kirigami pattern as usual (see Supplementary Figure 15 for an example).



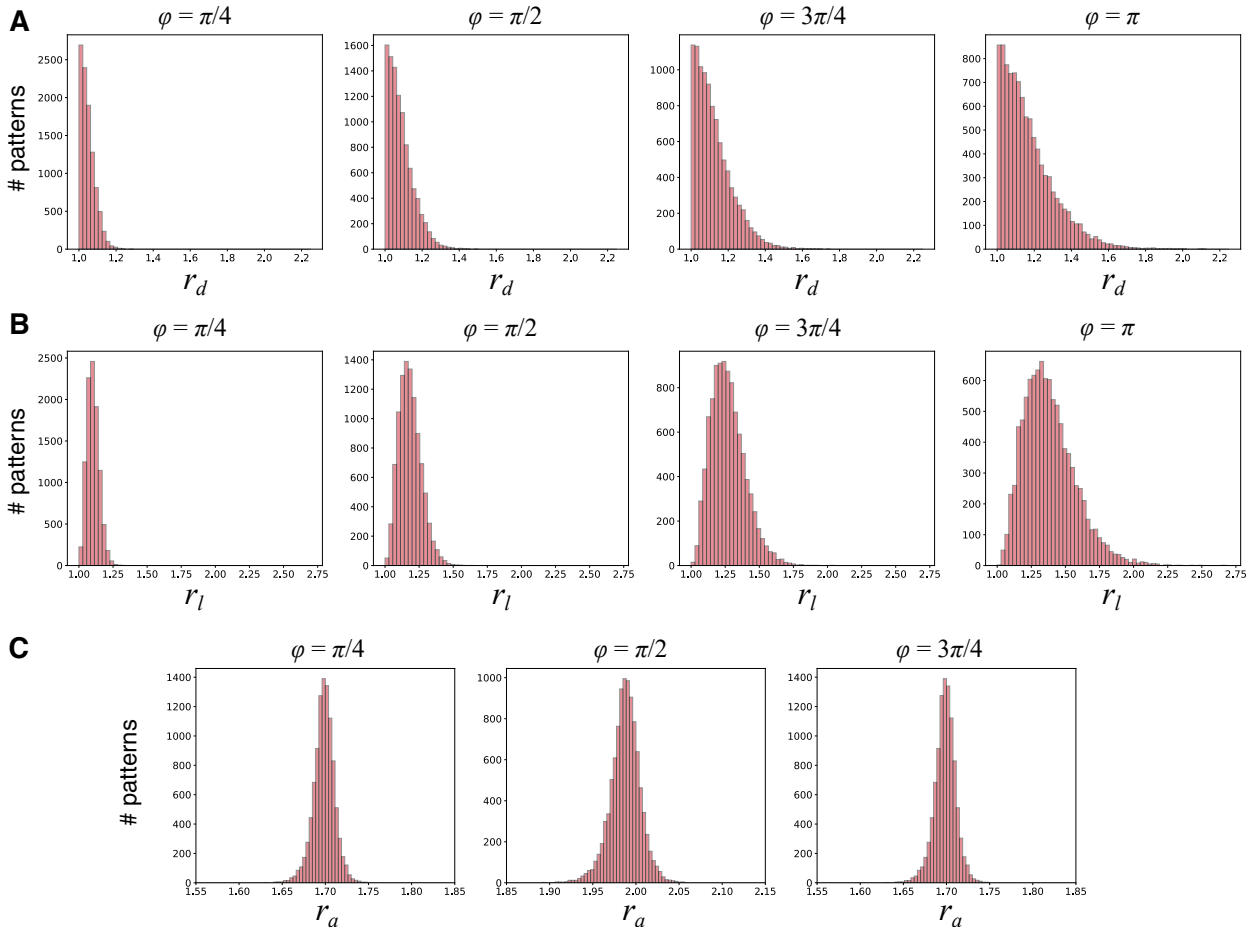
Supplementary Figure 16: **15 random square kirigami structures generated using our linear inverse design method.** For each pattern, three deployment snapshots at $\phi = 0$ (first contracted state), $\phi = \pi/2$ (fully deployed state), and $\phi = \pi$ (second contracted state) are shown.



Supplementary Figure 17: **Assessing the diagonal ratio of different configurations.** We consider the two diagonals of each configuration (highlighted in red) and calculate the ratio r_d of their length.

4 Analysis of random kirigami patterns

As described in the main text, 10000 random kirigami patterns were generated using our proposed method. Supplementary Figure 16 shows 15 examples of the random kirigami patterns generated, from which we can already see a large geometric variation. To quantify the geometric variation, several quantities are considered.



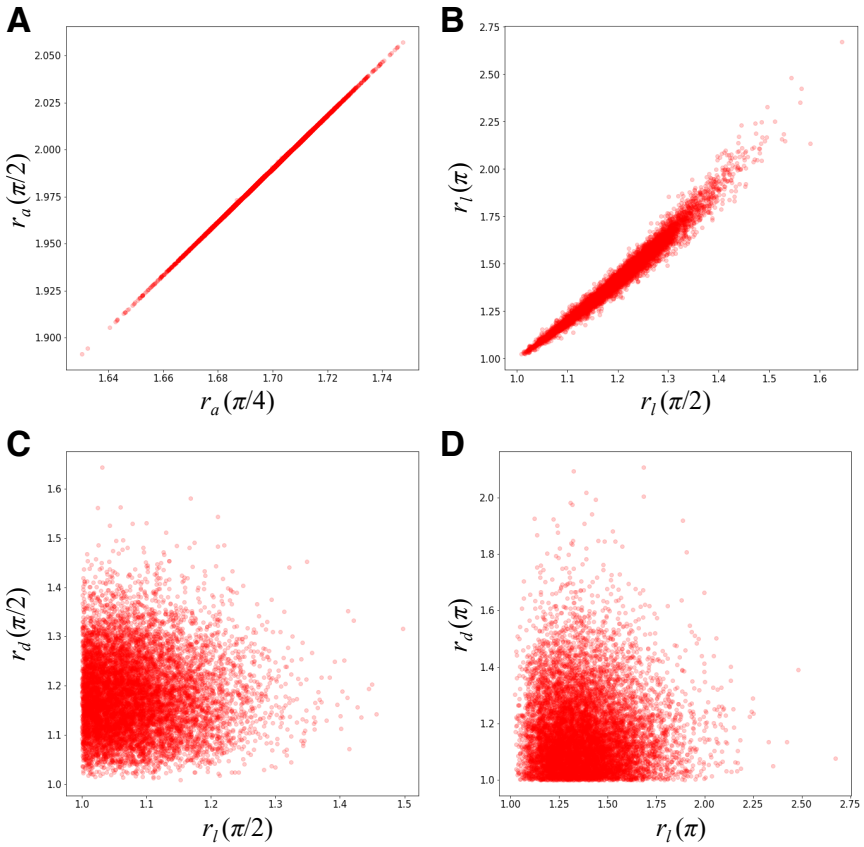
Supplementary Figure 18: **Statistics of the 10000 random kirigami patterns. (A)** Histograms of the diagonal ratio r_d for different deployment angle ϕ. **(B)** Histograms of the side length ratio r_l for different ϕ. **(C)** Histograms of the area ratio r_a for different ϕ.

4.1 Diagonal ratio

To assess the shape change of a random kirigami pattern at different deployment states, we consider the two diagonals of the pattern and compute the diagonal ratio $r_d = p/q$, where p is the length of the longer diagonal and q is the length of the shorter diagonal (see Supplementary Figure 17). For the random experiments shown in main text Figure 3C, we start with a square pattern and hence the diagonal ratio of the first contracted state is always 1. Throughout the deployment, the diagonal ratio r_d changes, where a larger r_d indicates a larger shear in the overall shape. Supplementary Figure 18A shows the histograms of r_d, from which it can be observed that r_d increases as ϕ increases. The variation of r_d also becomes larger as the structures morph from the square to the second contracted shape.

4.2 Side length ratio

Another way to assess the shape change of the random kirigami patterns is to consider the length ratio of the four corners of the patterns. In particular, note that the quadrilateral formed by the four corners of a random kirigami pattern is a perfect square at the initial contracted state, i.e. the lengths of the four sides are identical. As the structure deploys, the four side lengths may become different. One can therefore consider



Supplementary Figure 19: **Relationship between different quantities.** **(A)** A scatter plot of the area ratio $r_a(\pi/4)$ versus the area ratio $r_a(\pi/2)$. **(B)** A scatter plot of the side length ratio $r_l(\pi/2)$ versus the side length ratio $r_l(\pi)$. **(C)** A scatter plot of the side length ratio $r_l(\pi/2)$ versus the diagonal ratio $r_d(\pi/2)$. **(D)** A scatter plot of the side length ratio $r_l(\pi)$ versus the diagonal ratio $r_d(\pi)$.

the side length ratio $r_l = \max\{a,b,c,d\}/\min\{a,b,c,d\}$, where a,b,c,d are the four side lengths. A larger r_l indicates that the pattern deviates more from a rhombus. Supplementary Figure 18B shows the histograms of r_l, from which it can again be observed that the value and variation of r_l increase as ϕ increases.

4.3 Area ratio

We can also consider the overall area of a kirigami pattern, possibly including the negative spaces. Denote the overall area ratio by r_a (note that the initial area is 1). For rigid-deployable, compact reconfigurable quad kirigami structures, it is easy to see that the two contracted states corresponding to $\phi = 0$ and $\phi = \pi$, where ϕ is the only DOF in the deployment angle field, always have the same total area as they are formed by the same set of quads. In our simulations, the random structures always form a unit square and hence we have $r_a(0) = r_a(\pi) = 1$.

To consider the area of the structures at a certain deployed state, note that the four-bar linkage negative spaces are parallelograms. Hence, the area of each negative space can be expressed as

$$\|\mathbf{x}^i_{j,0} - \mathbf{x}^i_{j,1}\|\|\mathbf{x}^i_{j,0} - \mathbf{x}^i_{j,3}\| \sin\phi^i_j = \|\mathbf{x}^i_{j,0} - \mathbf{x}^i_{j,3}\|^2(1+\epsilon^i_j)\sin\phi^i_j \le \|\mathbf{x}^i_{j,0} - \mathbf{x}^i_{j,3}\|^2(1+\epsilon^i_j). \tag{47}$$

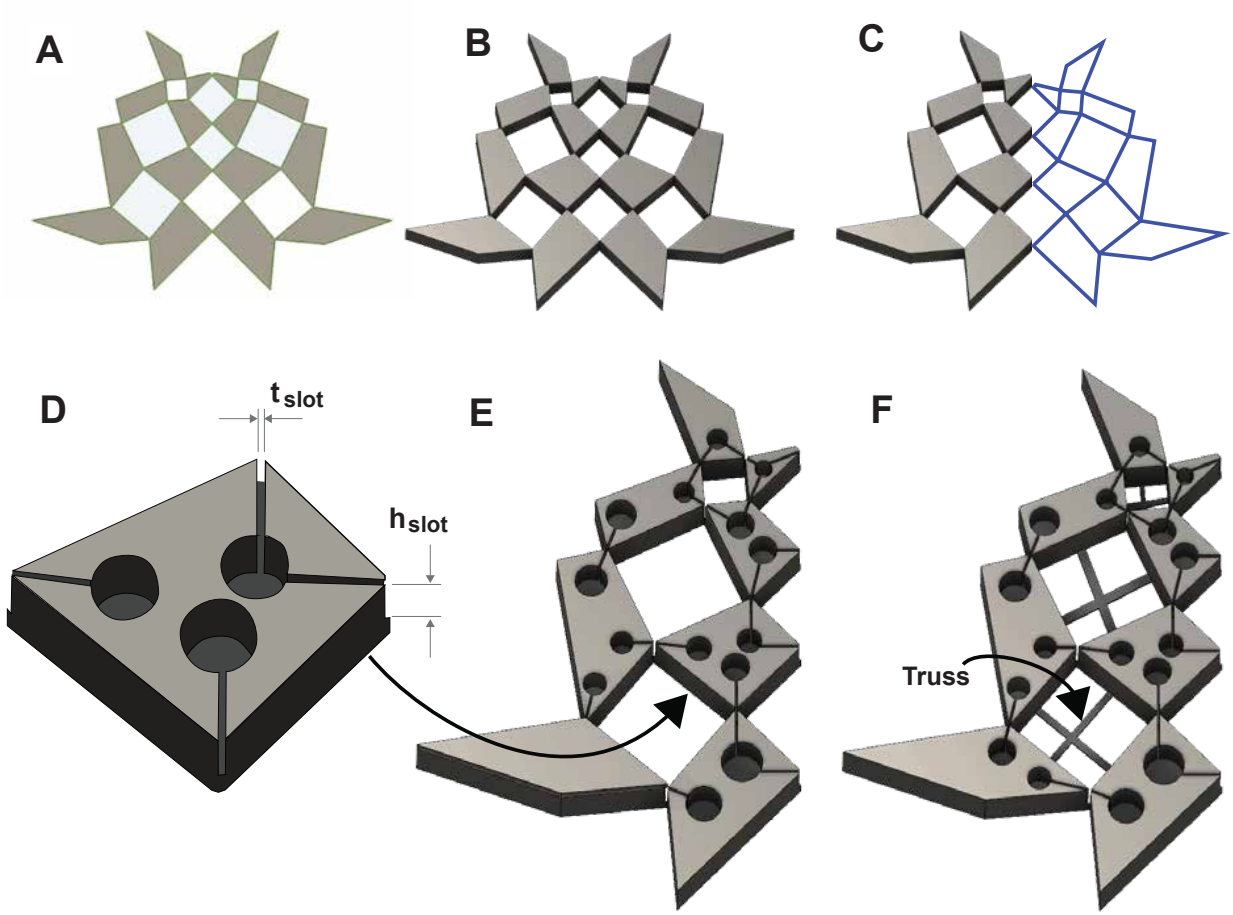
Consequently, r_a always attains its maximum at $\phi = \pi/2$ (recall that $\phi^i_j = \phi$ or $\pi - \phi$ for all i,j). In other

words, the maximum area is attained when all negative spaces are rectangles. Supplementary Figure 18C shows the histograms of r_a at $\phi = \pi/4, \pi/2, 3\pi/4$. Note that at different deployment states, the variation in r_a does not change significantly.

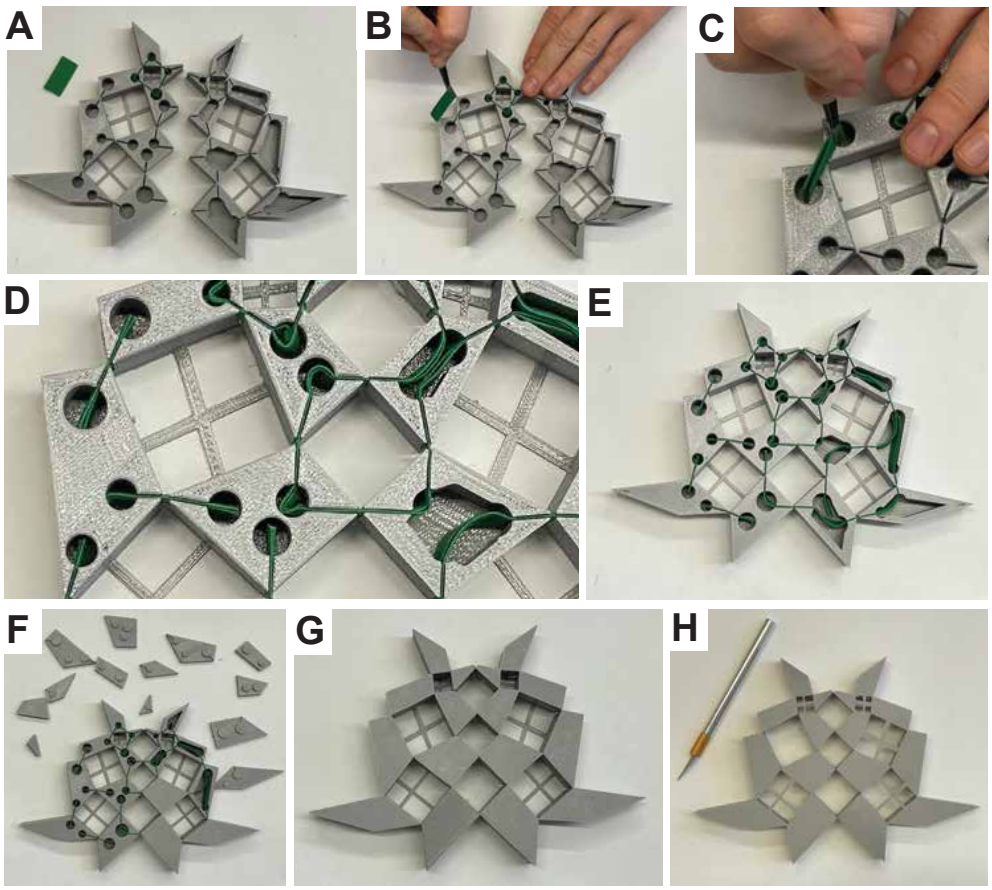
4.4 Relationship between the quantities

We first consider the relationship between the same quantity at different stages of the deployment. As shown in main text Figure 3C, the diagonal ratio r_d increases linearly as the deployment angle ϕ increases. From the scatter plot in Supplementary Figure 19A, we can clearly see that the area ratio r_a increases linearly in ϕ. We can also see a linear increasing trend in the side length ratio r_l from the scatter plot in Supplementary Figure 19B.

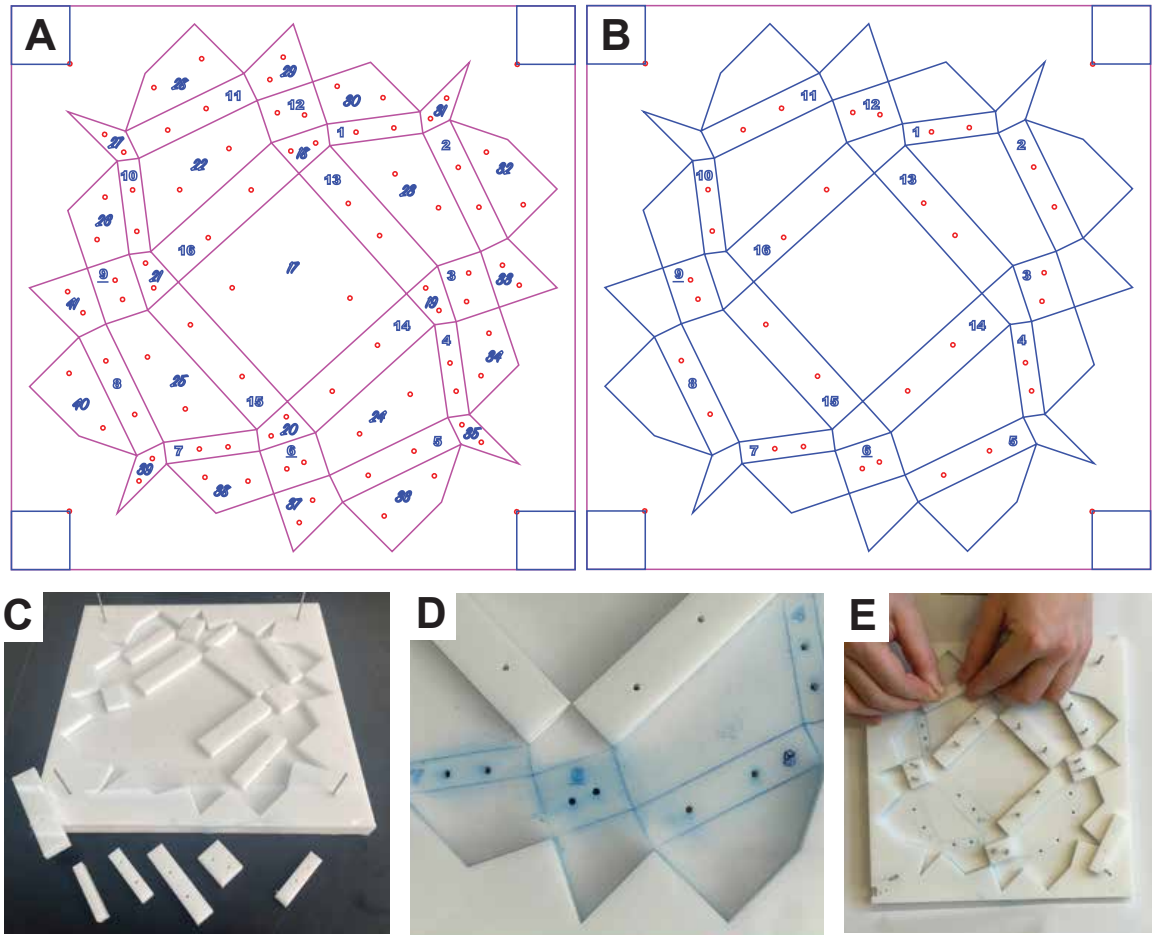
Next, we consider the relationship between different quantities. As shown in main text Figure 3C, the diagonal ratio r_d and the area ratio r_a at the fully deployed state $\phi = \pi/2$ are positively correlated. On the contrary, as shown the scatter plot in Supplementary Figure 19C–D, there is no clear relationship between the side length ratio r_l and the diagonal ratio r_d at either the fully deployed state $\phi = \pi/2$ or the second contracted state $\phi = \pi$.



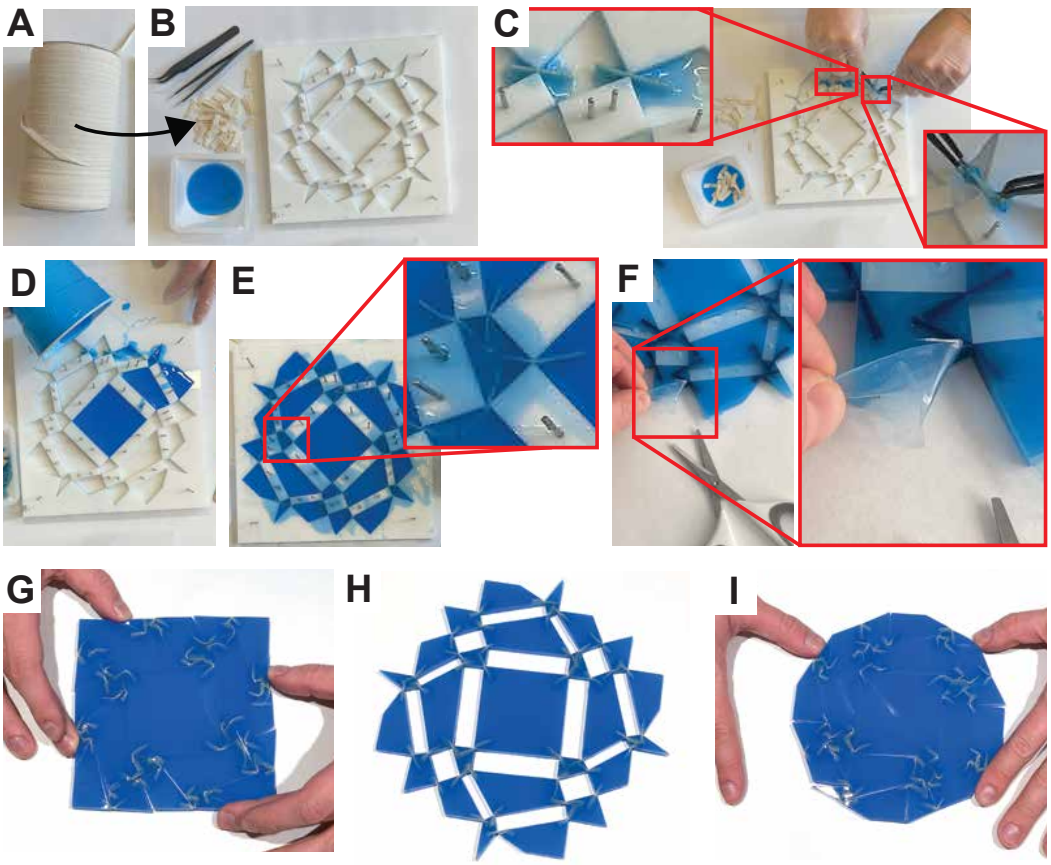
Supplementary Figure 20: **Digital design preparation for physical model fabrication via 3D printing.** **(A)** A vector image of the rigid-deployable heart kirigami pattern in main text Figure 2C imported into 3D modeling software. **(B)** An extrusion out of plane of the kirigami pattern. **(C)** An example of dividing kirigami pattern into symmetric subunits to reduce the overall processing time. **(D)** Slots are added to the individual kirigami tiles to later insert hinges. Circular pockets on the interior end of the slot allow clearance for tweezers to aid in hinge installation. **(F)** Temporary trusses are added at the negative spaces to fix the alignment of the model during assembly.



Supplementary Figure 21: **Assembly and hinge installation of a 3D printed physical model. (A)** A single textile hinge prior to installation and two mirrored halves of A 3D printed model of the partially deployed configuration of the rigid-deployable heart kirigami pattern in main text Figure 2C. Two variants of clearance pockets are shown between the two halves. **(B)** Installation of a single hinge across two kirigami tiles. **(C)** A closeup view of hing installation aided by tweezers. **(D)** A closeup view of single hinges installed across two kirigami tiles as well as examples of textile wound through the interior pockets of multiple adjacent tiles. **(E)** Heart kirigami pattern with all hinges installed. **(F)** Partially installed caps to cover hinge slots and clearance pockets. **(G)** Heart kirigami assembly with all caps installed. **(H)** Printed kirigami heart structure with one of the pairs of temporary trusses removed.



Supplementary Figure 22: **Mold preparation process for a deployed configuration of the rigid-deployable square-to-circle kirigami pattern in main text Figure 4.** **(A)** A vector image use to create the top layer of a laser cut mold of the square-to-circle kirigami pattern. **(B)** A vector image use to create the bottom layer of a laser cut mold of the square-to-circle kirigami pattern. **(C)** Picture of a partially assembled laser cut acrylic mold made using the vector images in panels A and B. The mold is an inversion of the final geometry that will be molded. **(D)** Numbers and lines etched in the mold parts help guide the location of parts and are easier to see when colored with a silicone compatible dye. **(E)** The mold is held together with press-fit pins and can be fully assembled and disassembled by hand without the aid of tools.



Supplementary Figure 23: **Physical model fabrication via rubber molding with embedded fabric hinges in a laser cut mold based on a deployed configuration of the rigid-deployable square-to-circle kirigami pattern in main text Figure 3B. (A)** Spool of cotton twill tape used to create hinges. **(B)** The twill tape is cut into pieces to create the hinges between adjacent kirigami tiles. **(C)** The hinges are dipped in rubber and inserted in the gap between mold part, extending into the adjacent cavities. Two pairs of tweezers are used to pull the hinges into place. **(D)** Silicone rubber is poured into the mold. **(E)** The embedded hinges are visible when molded into translucent rubber. **(F)** Rubber that overflowed the mold creates a web between the kirigami tiles and must be trimmed away for the hinges to rotate. **(G)** The soft kirigami sheet in the square configuration with embedded hinges visible in translucent rubber. **(H)** The soft kirigami sheet of the square-to-circle pattern in mid deployment state. **(I)** The soft kirigami sheet of the square-to-circle pattern in the circle configuration.