UCL
TU WIEN TECHNISCHE UNIVERSITÄT WIEN Vienna | Austria

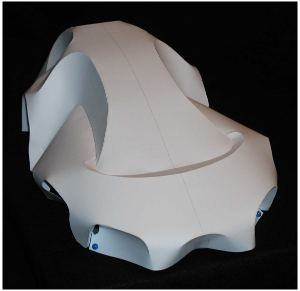
# String Actuated Curved Folded Surfaces

Martin Kilian Aron Monszpart Niloy J. Mitra

# Curved Folding

[Albers, 1927]

[Huffman, 1970s]

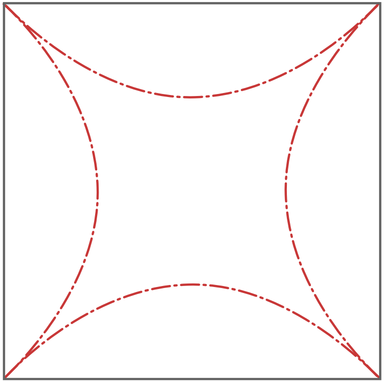


[Epps, 2007]

Models made of paper and PVC.

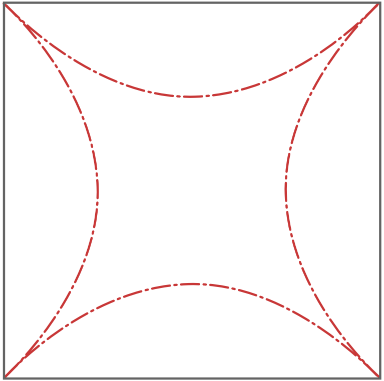
# Developable surfaces with curved creases

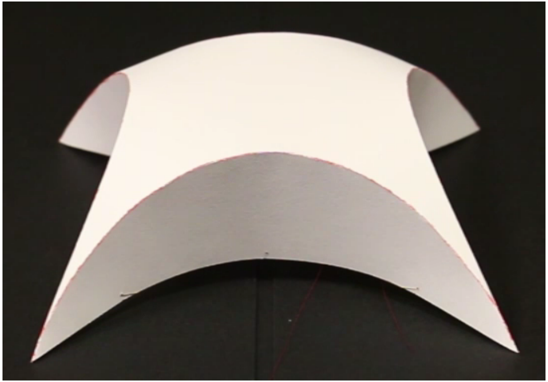
## A simple example



# Developable surfaces with curved creases

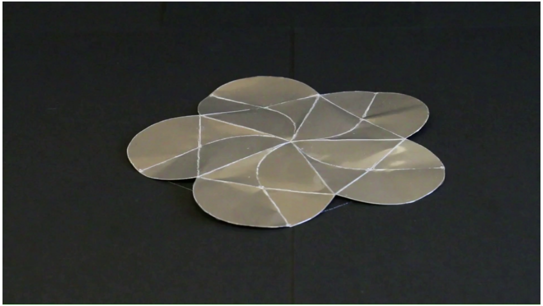
## A simple example



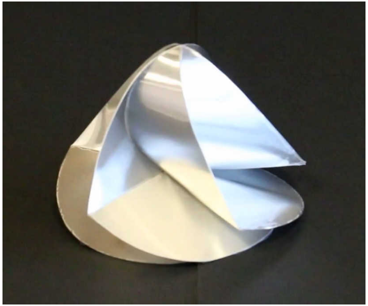


# Developable surfaces with curved creases

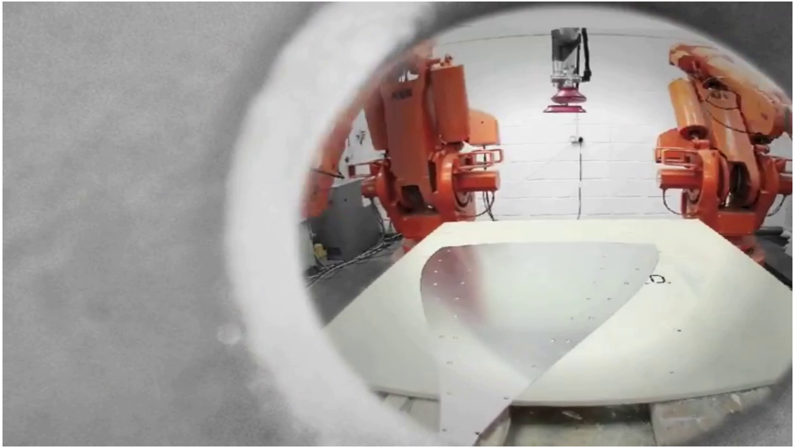
A **static surface** that results from folding a **given** crease pattern.
**Folding dynamics** have not been considered so far.



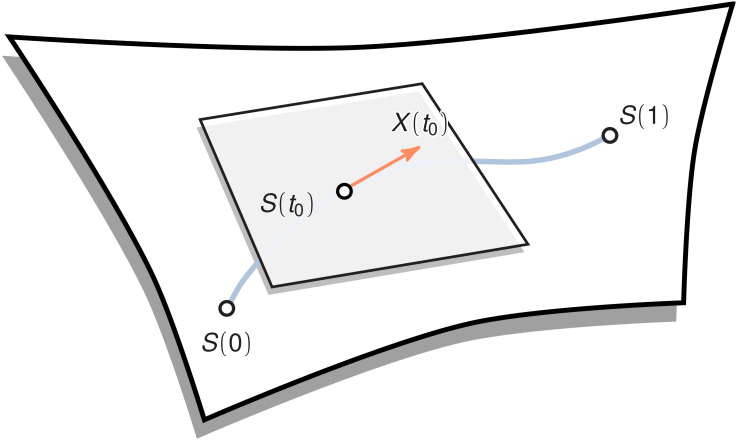
→



# Developable surfaces with curved creases



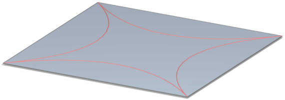
# High Level Idea



# String actuated folding

Given a **crease pattern** we are looking for a **network of strings** that drives the folding when being pulled. Starting from a **folding sequence**

$U$



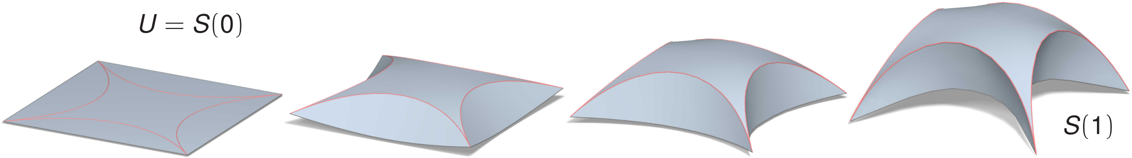
we compute a set of **actuation points** and a **network of strings** in order to reproduce the given deformation $S$.

# String actuated folding

Given a **crease pattern** we are looking for a **network of strings** that drives the folding when being pulled. Starting from a **folding sequence**

$$S : [0, 1] \times U \to \mathbb{R}^3$$
$$(t, \mathbf{u}) \mapsto S(t, \mathbf{u})$$

$U = S(0)$



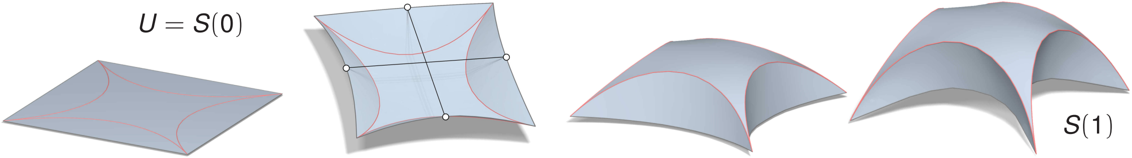
$S(1)$

we compute a set of **actuation points** and a **network of strings** in order to reproduce the given deformation $S$.

# String actuated folding

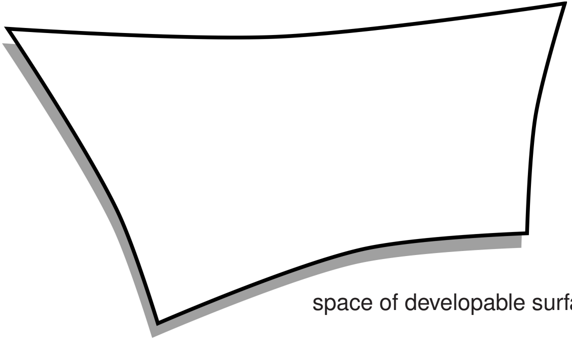
Given a **crease pattern** we are looking for a **network of strings** that drives the folding when being pulled. Starting from a **folding sequence**

$$
\begin{aligned}
S : [0, 1] \times U &\rightarrow \mathbb{R}^3 \\
(t, \mathbf{u}) &\mapsto S(t, \mathbf{u})
\end{aligned}
$$



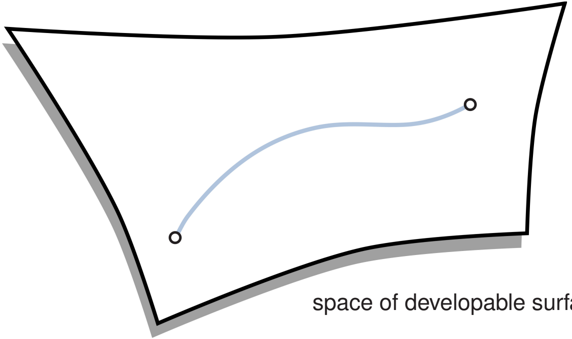

we compute a set of **actuation points** and a **network of strings** in order to reproduce the given deformation $S$.

# Analyzing a folding motion



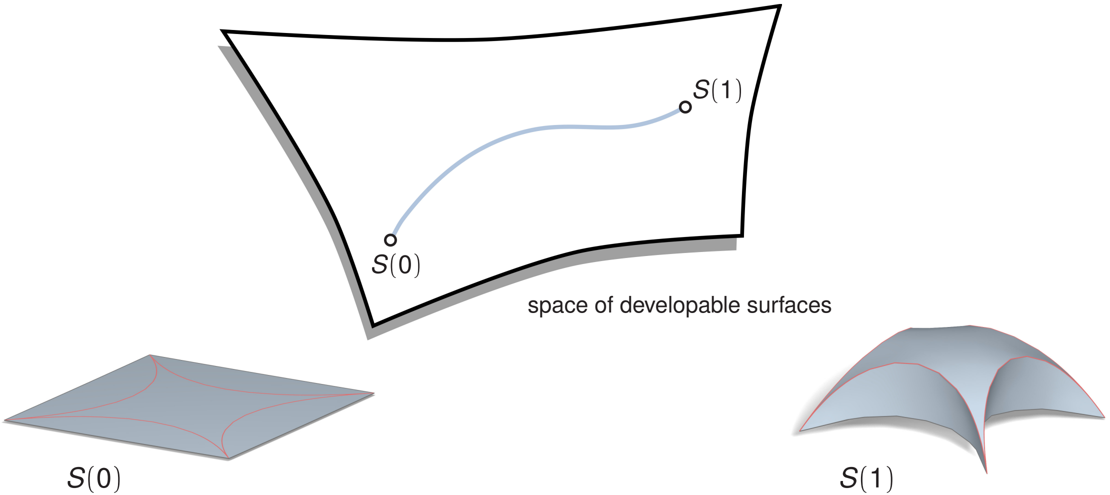
space of developable surfaces

# Analyzing a folding motion

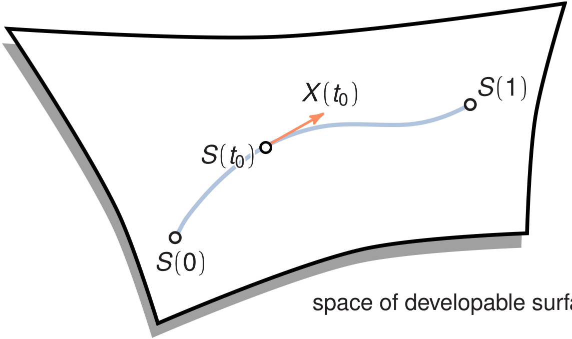


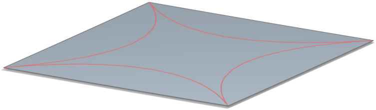

A folding motion *S* defines a **curve** in the space of developable surfaces.

# Analyzing a folding motion

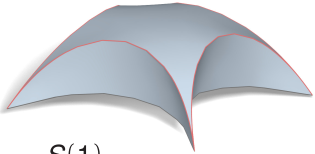


# Analyzing a folding motion



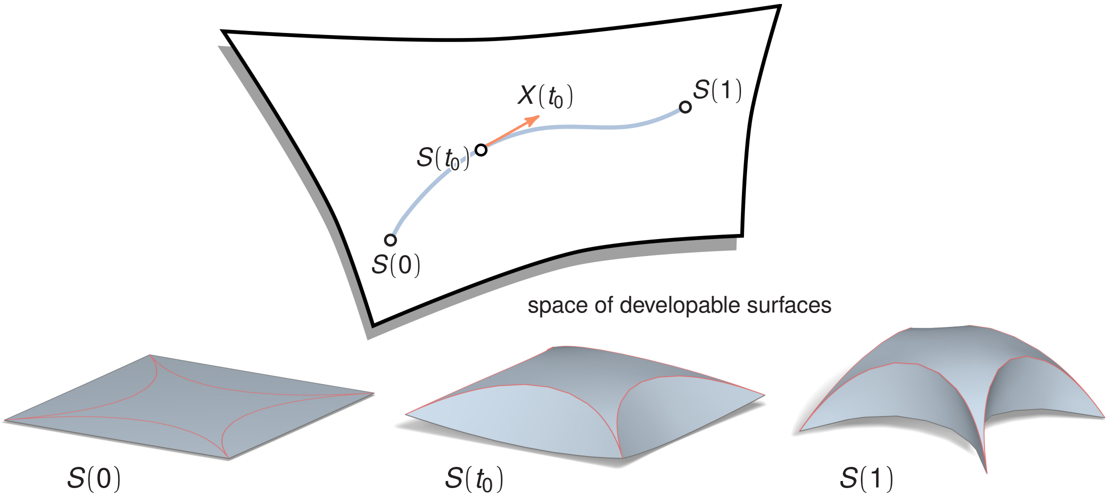



$S(0)$

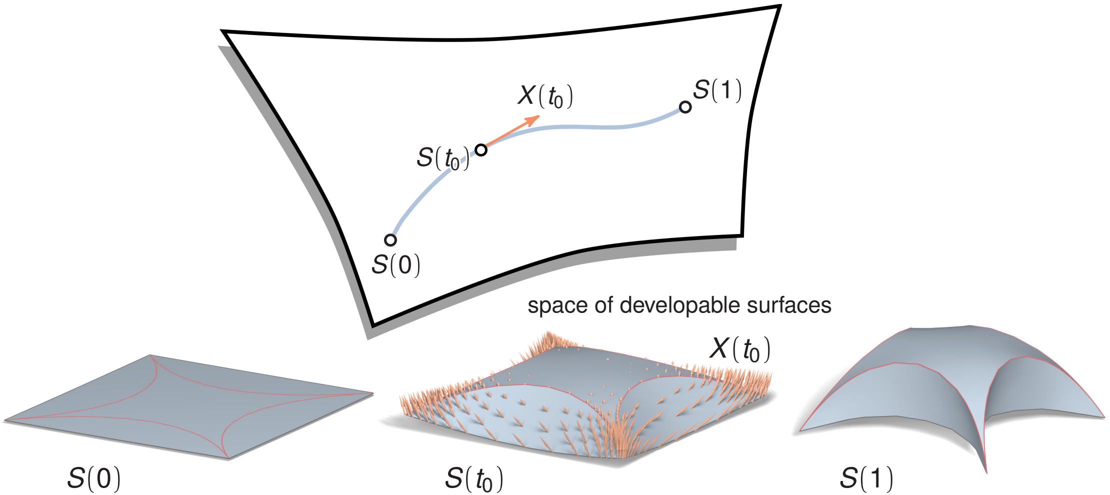


$S(1)$

# Analyzing a folding motion



# Analyzing a folding motion



# Observation

Any **actuation mechanism** that reproduces the target motion $S$ also has to reproduce the deformation fields $X(t)$, $t \in [0, 1]$.

# Observation

Any **actuation mechanism** that reproduces the target motion $S$ also has to reproduce the deformation fields $X(t)$, $t \in [0, 1]$.

$$S \xrightarrow{\text{sampling}} S(t_1) \qquad S(t_2) \qquad S(t_3) \qquad \ldots \qquad S(t_n)$$

# Observation

Any **actuation mechanism** that reproduces the target motion $S$ also has to reproduce the deformation fields $X(t)$, $t \in [0, 1]$.

$$
\begin{array}{cccccccc}
S & \xrightarrow{\text{sampling}} & S(t_1) & S(t_2) & S(t_3) & \dots & S(t_n) & \\
 & & \downarrow & \downarrow & \downarrow & & \downarrow & \\
 & & X(t_1) & X(t_2) & X(t_3) & \dots & X(t_n) & \text{linearization}
\end{array}
$$

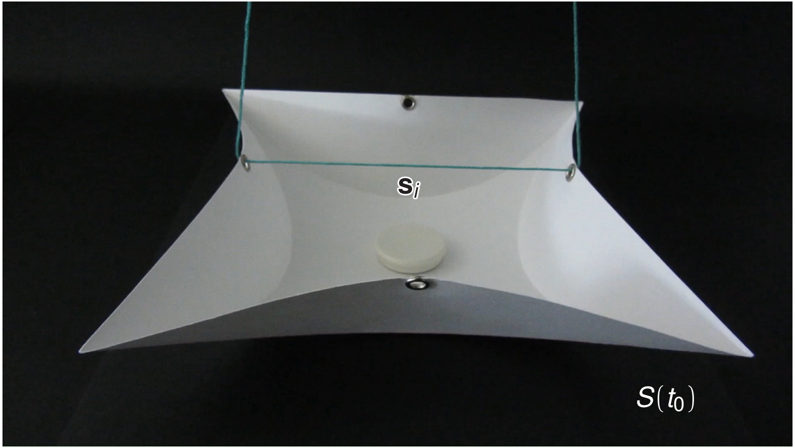
# Observation

Any **actuation mechanism** that reproduces the target motion $S$ also has to reproduce the deformation fields $X(t)$, $t \in [0, 1]$.

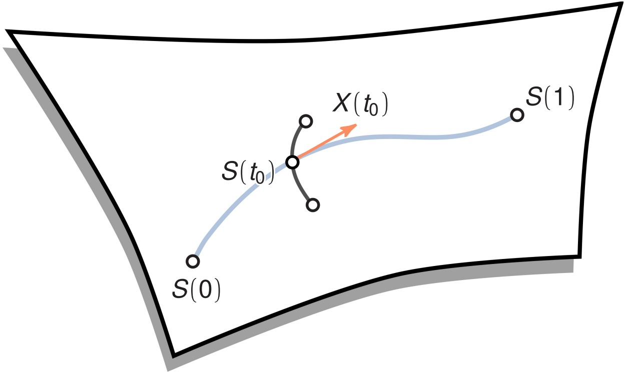
$$
\begin{tikzcd}
S \arrow[r, "\text{sampling}"] & S(t_1) \arrow[d] & S(t_2) \arrow[d] & S(t_3) \arrow[d] & \cdots & S(t_n) \arrow[d] & \\
& X(t_1) \arrow[drr] & X(t_2) \arrow[dr] & X(t_3) \arrow[d] & \cdots & X(t_n) \arrow[dll] & \text{linearization} \\
& & & \mathcal{S}^* & & & \text{solution}
\end{tikzcd}
$$

# Analyzing a folding motion

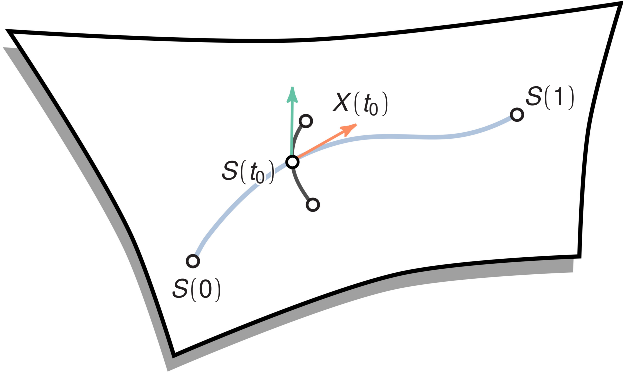
Attach an **arbitrary string $\mathbf{s}_i$** to the **rest shape** $S(t_0)$:



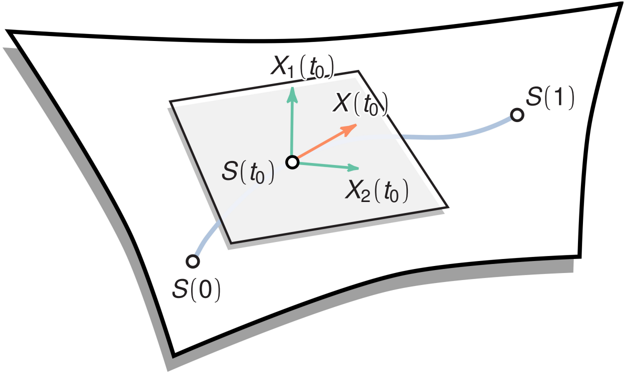
# Analyzing a folding motion



# Analyzing a folding motion

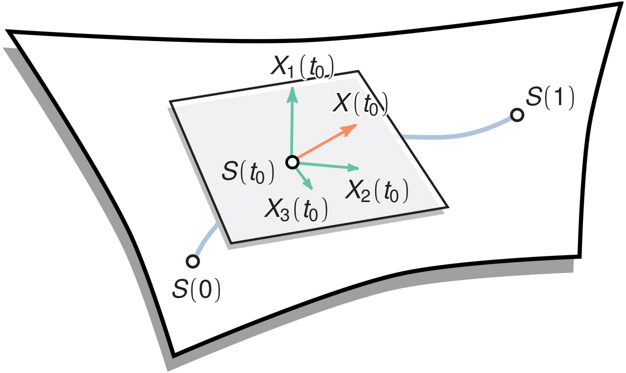


# Analyzing a folding motion



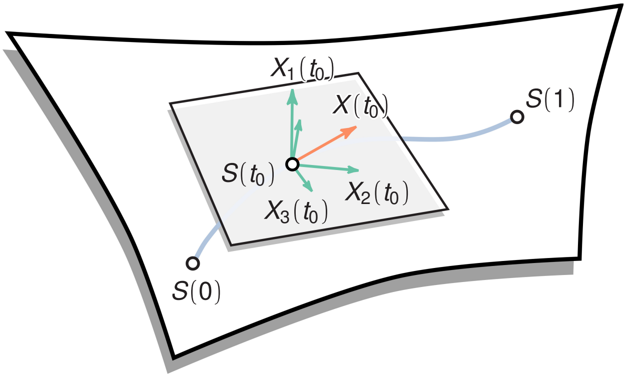

Each $X_i(t_0)$ is the **deformation field** induced by string $\mathbf{s}_i$, connecting exactly two surface points.

# Analyzing a folding motion



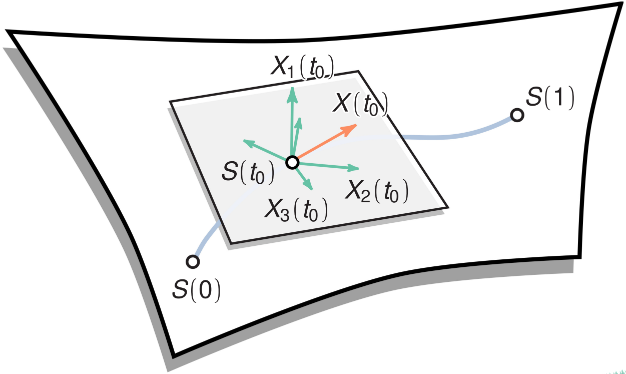

Each $X_i(t_0)$ is the **deformation field** induced by string $\mathbf{s}_i$, connecting exactly two surface points.

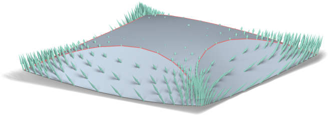
# Analyzing a folding motion




Each $X_i(t_0)$ is the **deformation field** induced by string $\mathbf{s}_i$,
connecting exactly two surface points.

# Analyzing a folding motion



We call the $X_i(t_0)$ **actuation modes.**

# Local Problem

At time $t_0$ find strings $\mathbf{s}_i$ that reproduce $S|_{[t_0-\varepsilon, t_0+\varepsilon]}$.
**Local action** of $\mathbf{s}_i$ is described by $X_i(t_0)$.

Because of linearization:
**joint deformation** field of $\{\mathbf{s}_i\}$ is **superposition** of the $X_i(t_0)$.

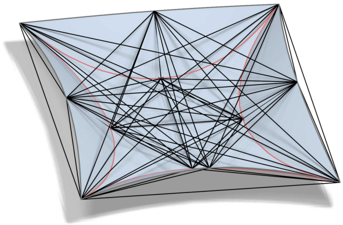
# Local Solution

Finding strings $\mathbf{s}_i$ whose joint deformation reproduces $S|_{[t_0-\varepsilon, t_0+\varepsilon]}$ is equivalent to finding actuation modes $X_i(t_0)$ that reproduce $X(t_0)$:

$$\sum \lambda_i X_i(t_0) \approx X(t_0).$$

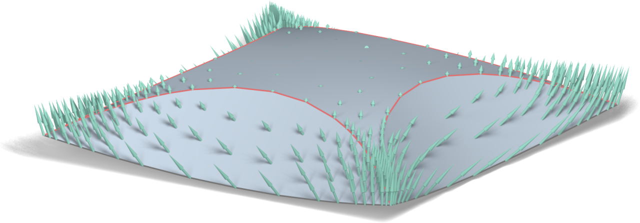
# Actuation mode selection

- Sample crease curves to get a set $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$ of **candidate strings**.



# Actuation mode selection

- Sample crease curves to get a set $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$ of **candidate strings**. At $S(t_0)$ compute the corresponding modes $X_i(t_0)$.

$\dots$  $\dots$

- Find representation of $X(t_0)$ in terms of the modes $X_i(t_0)$,

$$X(t_0) \approx \sum_{i=1}^{m} \lambda_i(t_0) X_i(t_0).$$

# Actuation mode selection

Compute a **sparse solution** according to

$$\lambda^* = \arg\min_{\lambda} \left[ \omega \|\lambda\|_0 + \|X - \sum_{i=1}^{m} \lambda_i X_i\|^2 \right] \tag{1}$$

# Actuation mode selection

Compute a **sparse solution** according to

$$\lambda^* = \arg\min_{\lambda} \left[ \omega \|\lambda\|_0 + \|X - \sum_{i=1}^{m} \lambda_i X_i\|^2 \right] \tag{1}$$

$$\|\lambda\|_0 = \sum_{i=1}^{m} (\lambda_i \neq 0)\ ?\ 1 : 0 \qquad \text{(sparsity)}$$

$$\|X - \sum_{i=1}^{m} \lambda_i X_i\|^2 \qquad \text{(fitting)}$$

# Actuation mode selection

Introducing auxiliary variables $\xi_i$, one can show that

$$\|\lambda\|_0 = \min_{\xi} \sum_{i=1}^{m} (1 - \xi_i)$$

subject to $\lambda_i \xi_i = 0$ and $0 \leqslant \xi_i \leqslant 1$.

# Actuation mode selection

Introducing auxiliary variables $\xi_i$, one can show that

$$\|\lambda\|_0 = \min_{\xi} \sum_{i=1}^{m} (1 - \xi_i)$$

subject to $\lambda_i \xi_i = 0$ and $0 \leqslant \xi_i \leqslant 1$.

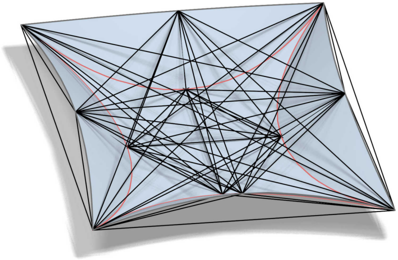
The variables $\xi_i$ act as **indicator** variables:

$$\left[\lambda_i = 0 \iff \xi_i = 1\right] \quad \text{and} \quad \left[\lambda_i \neq 0 \iff \xi_i = 0\right]$$

[Feng et al.: ***Comp. Formulations of $\ell_0$-norm Optimization Problems***, 2015]

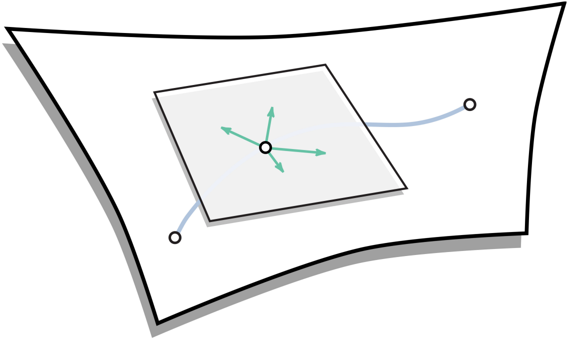
# Towards a global solution

1 initialize $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$



# Towards a global solution

1. initialize $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$
2. compute $X_i(t_0)$, $i = 1, \ldots, m$



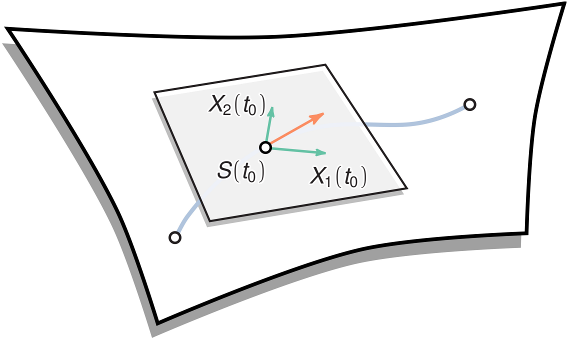
# Towards a global solution

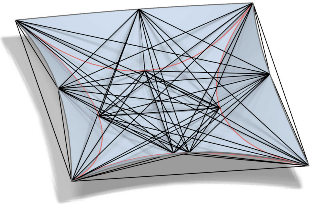
1. initialize $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$
2. compute $X_i(t_0)$, $i = 1, \ldots, m$
3. solve (1)

[Wächter, Biegler: ***On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming***, 2006]

# Towards a global solution

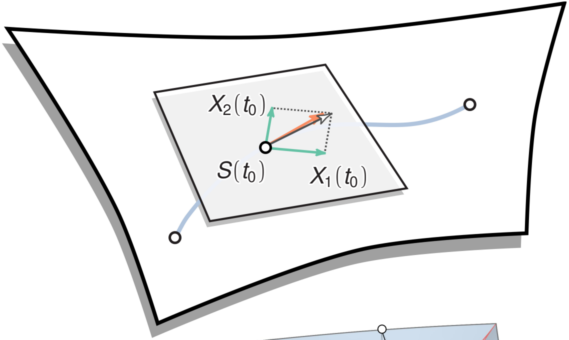
1. initialize $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$
2. compute $X_i(t_0)$, $i = 1, \ldots, m$
3. solve (1)
4. pick the $\mathbf{s}_i$ with $\xi_i = 0$

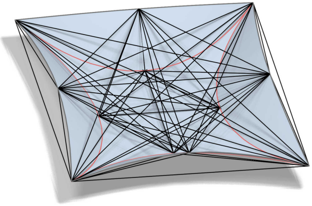


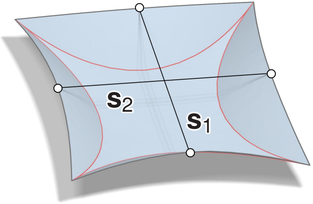
# Towards a global solution

1. initialize $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^{m}$
2. compute $X_i(t_0),\ i = 1, \ldots, m$
3. solve (1)
4. pick the $\mathbf{s}_i$ with $\xi_i = 0$

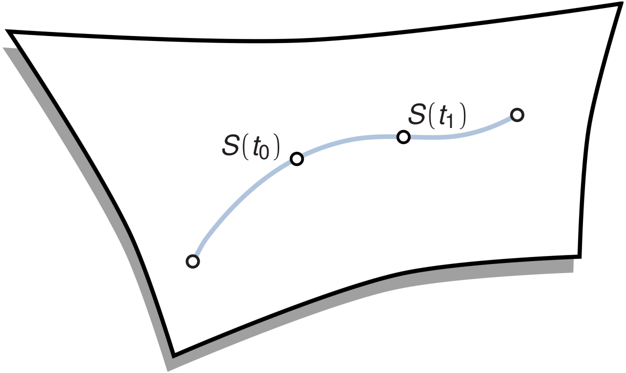


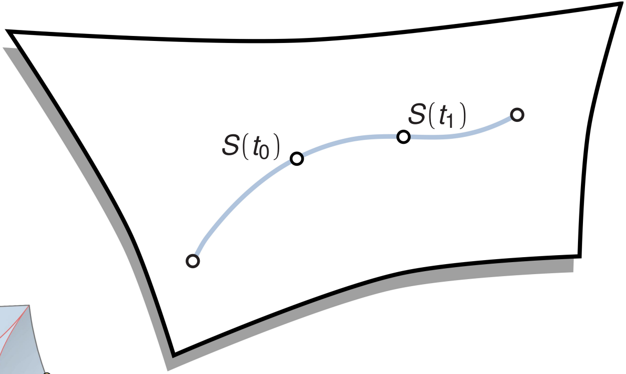
$\rightarrow$

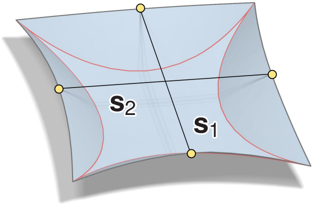


# Towards a global solution



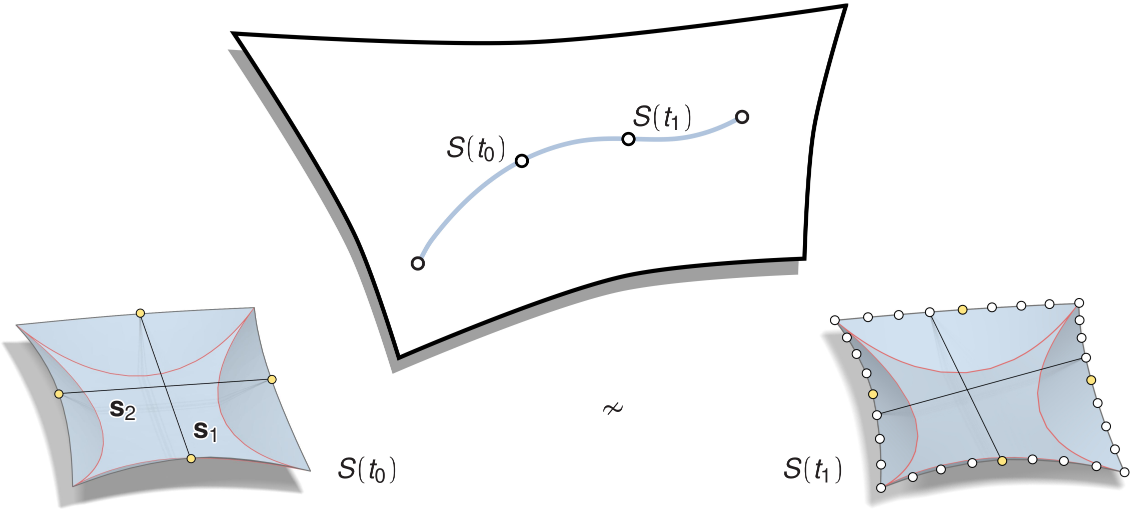
# Towards a global solution




$S(t_0)$

# Towards a global solution



# Global optimization

Only one $\xi_i$ per $\mathbf{s}_i$ independent of the number of poses $S(t_j)$:

$$\min_{\xi,\lambda}\left[\omega\sum_{i=1}^{m}(1-\xi_i)+\sum_{j=1}^{n}\|X_j-\sum_{i=1}^{m}\lambda_{ij}X_{ij}\|^2\right] \tag{2}$$

subject to

$$0\leqslant\xi_i\leqslant1,\quad \lambda_{ij}\xi_i=0,\quad i=1,\ldots,m,\ j=1,\ldots,n.$$

# Global optimization

Only one $\xi_i$ per $\mathbf{s}_i$ independent of the number of poses $S(t_j)$:

$$\min_{\xi,\lambda} \left[ \omega \sum_{i=1}^{m} (1 - \xi_i) + \sum_{j=1}^{n} \| X_j - \sum_{i=1}^{m} \lambda_{ij} X_{ij} \|^2 \right] \tag{2}$$

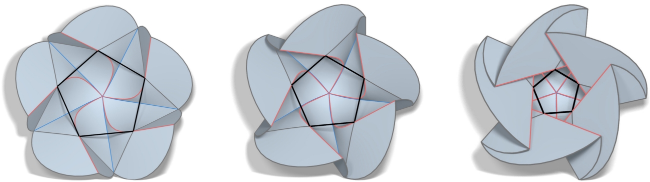
subject to

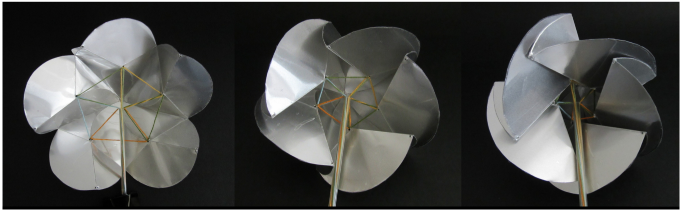
$$0 \leqslant \xi_i \leqslant 1, \quad \lambda_{ij} \xi_i = 0, \quad i = 1, \ldots, m, \; j = 1, \ldots, n.$$

*Have to pay* **unit cost** 1 *to use* $\mathbf{s}_i$.

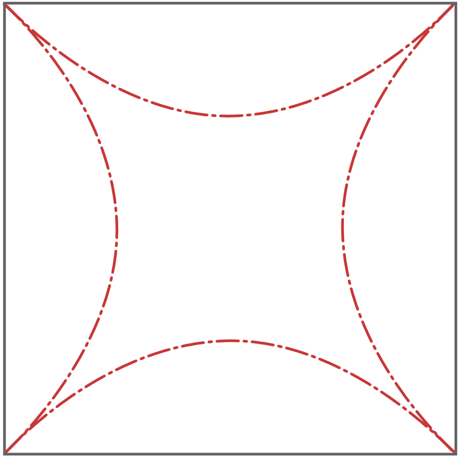
**Re-using** *it at other time steps* **is free!**

# Results

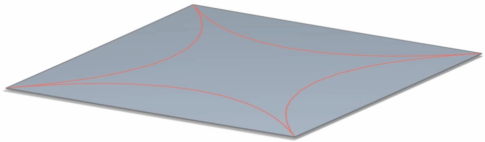




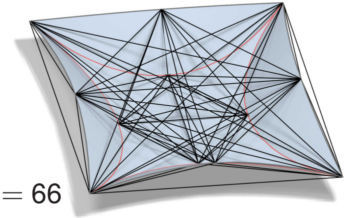
# Quad



# Quad

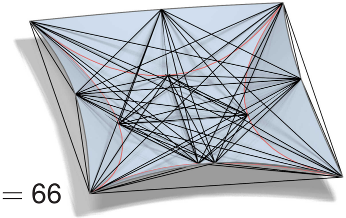


# Quad

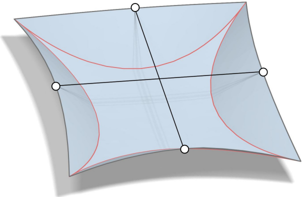


$|\mathbb{S}| = 66$

# Quad

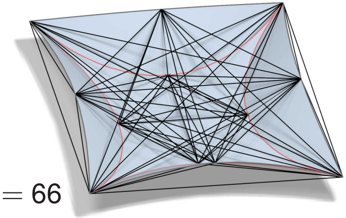


$|\mathbb{S}| = 66$

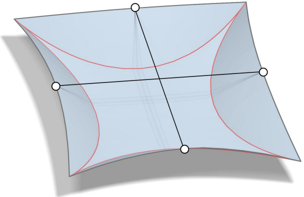


#1

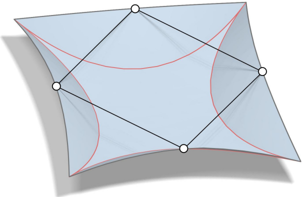
# Quad



$|\mathbb{S}| = 66$

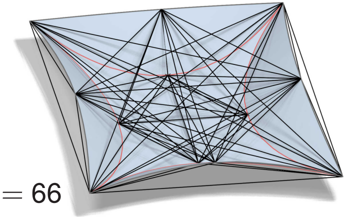


#1

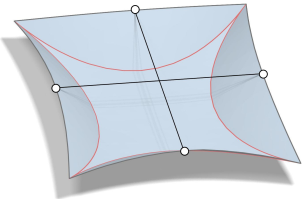


#2

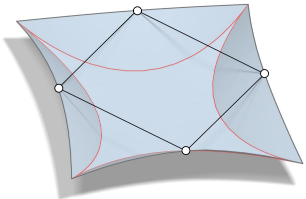
# Quad



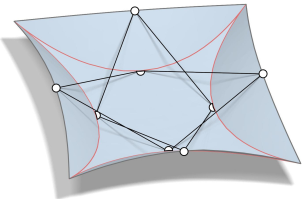
$|\mathbb{S}| = 66$



#1

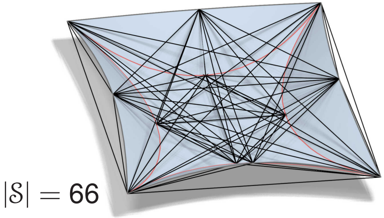


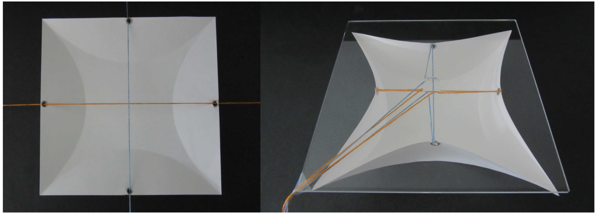
#2

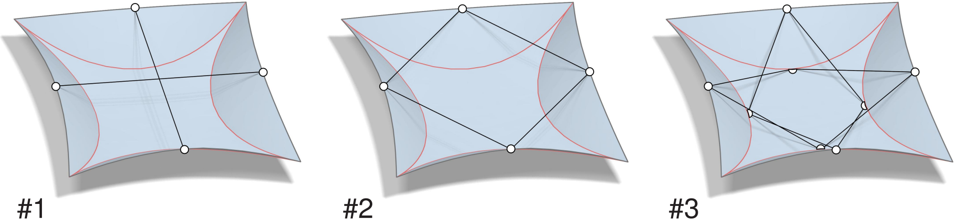


#3

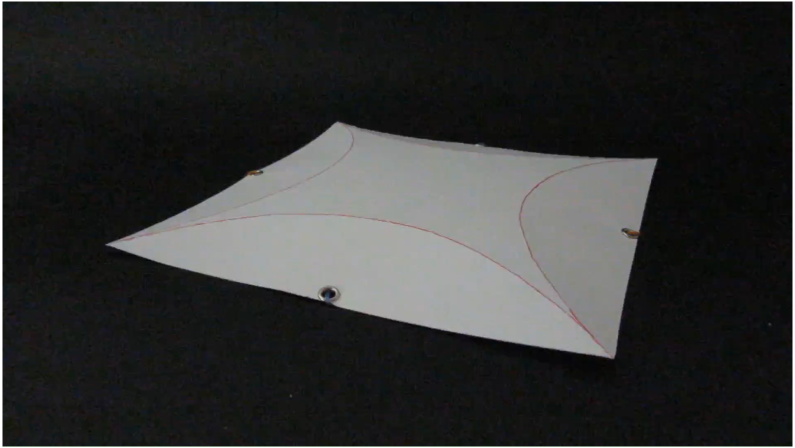
# Quad



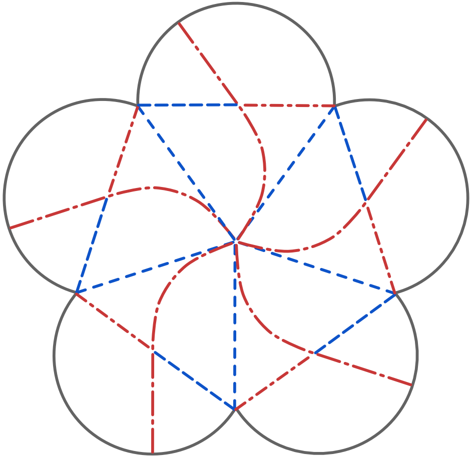





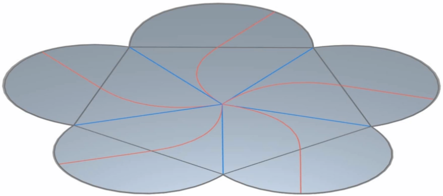
# Quad



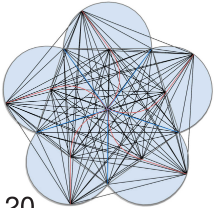
# Apricot (Design by J. Mitani)



# Apricot

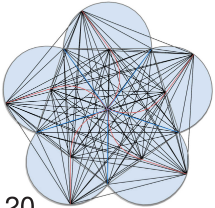


# Apricot

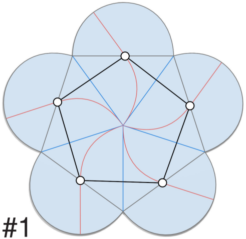


$|\mathcal{S}| = 120$

# Apricot

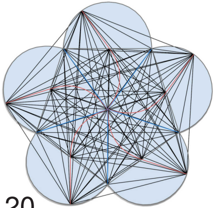


$|\mathbb{S}| = 120$

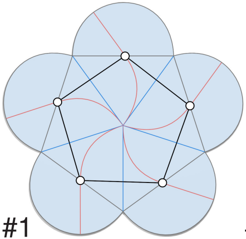


#1

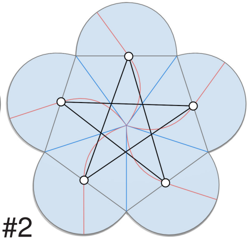
# Apricot



$|\mathcal{S}| = 120$

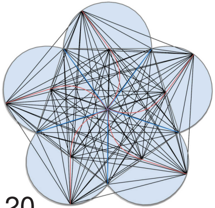


#1

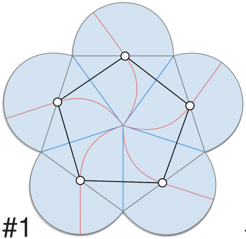


#2

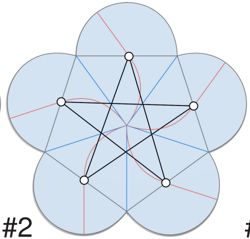
# Apricot



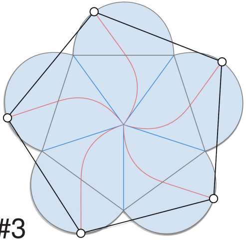
$|\mathcal{S}| = 120$



#1

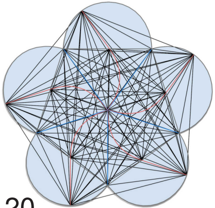


#2

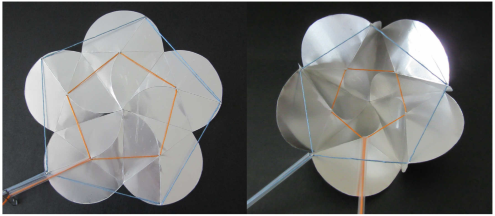


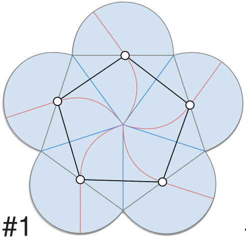
#3

# Apricot

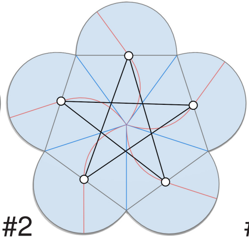


$|\mathbb{S}| = 120$

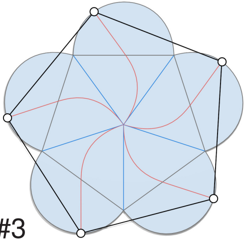




#1

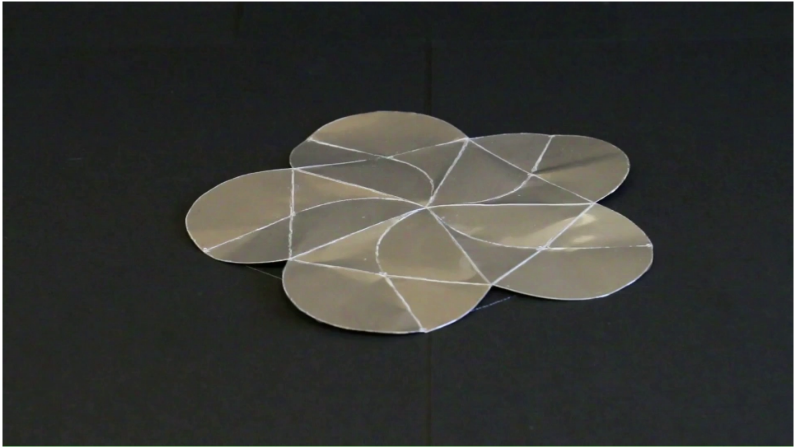


#2



#3

# Apricot



# Conclusion

## Limitations

- Not every shape is foldable using strings
- Material weight and gravity not considered
- No exterior actuation points

# Conclusion

## Limitations

- Not every shape is foldable using strings
- Material weight and gravity not considered
- No exterior actuation points

## Future work

- Motion design
- Staged folding (keyframes)

# Conclusion

## Limitations

- Not every shape is foldable using strings
- Material weight and gravity not considered
- No exterior actuation points

## Future work

- Motion design
- Staged folding (keyframes)

Combined pattern, shape, and motion optimization

# Thank you!

FWF

Discretization in Geometry and Dynamics
SFB Transregio 109

erc

Adobe

http://geometry.cs.ucl.ac.uk/projects/2017/string-actuated/