6.849	Class 1	Feb. 8, 2017

6.849: Geometric Folding Algorithms

Prof. Erik Demaine
Colecturers: Martin Demaine, Dr. Jason Ku
TAs: Adam Hesterberg, Jayson Lynch
http://courses.csail.mit.edu/6.849/spring17/

Inverted lectures: (new format)

- online video lectures + notes + slides
  (from 2010; also bonus videos from 2012)
  - DEMO: slide sync & jump, playback speed, required feedback form
- class time for interactivity
  - answering questions
  - additional detail/material of interest
  - guest lectures
  - activities: folding, building, etc.
  - * solving problems: solved & unsolved!
    ~ math, programming, and design

+ optional open problem session (if interest)
- Coauthor software to coordinate in/outside class

DISCUSS

Requirements:

- sign up for account on Coauthor] on website
- short survey of your background]
- watch video lecture by NOON on Tuesday
  e.g. Monday night OR weekend
  - fill out form to confirm watching DEMO
  - post any questions/feedback to Coauthor
- attend classes (email me about exceptions)
  - generally not videoed
  - solve problems together!
- problem sets ≈ weekly
  - choose your own adventure DEMO PS1
  - often: solve 2 out of 3 problems
- project & presentation
  - build/design physical structure
  - implement algorithm/illustration/tool
  - pose open problem
  - research: try to solve an open problem
  - survey subfield (not in textbook)
  - Wikipedia (write/improve several articles)
- textbook: Demaine & O'Rourke, CUP 2007
  (working on discount)

Geometric folding algorithms:

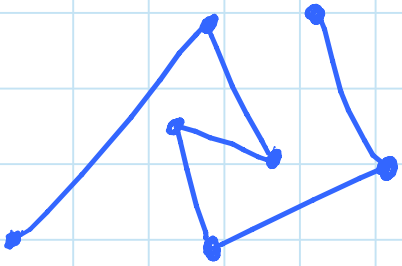
In general: Mathematics & algorithms behind (un)folding of geometric objects

Applications/connections to:

- robotics → arms. Transformers, programmable matter, ...
- graphics → morphing, animation, ...
- mechanics → steam engines, ...
- manufacturing → sheet-metal & tube bending, nanomanufacturing, optics, ...
- medical → stents, drug delivery, ...
- aerospace → telescope deployment, ...
- biology → protein folding & design, ...
- sculpture → origami, interactive sculpture, ...
- architecture → dynamic architecture, deployable/collapsible structures, ...

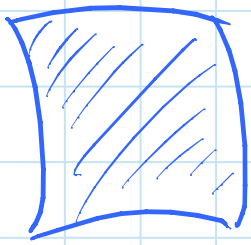
Geometric objects & rules for folding:

(I) linkage



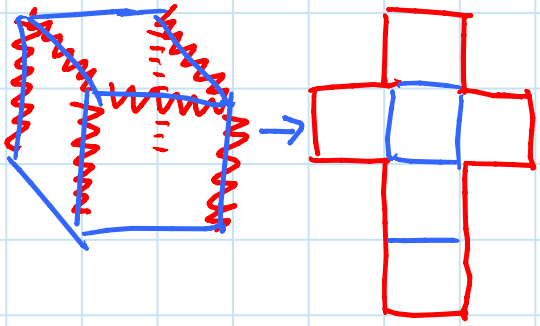
↳ rigid bars
↳ stay connected
↳ [don't cross]

(II) paper



↳ don't stretch
↳ don't tear
↳ don't cross

(III) polyhedron



↳ cut surface
↳ one piece
↳ no overlap

Problem types:

e.g. can this fold? how?
↑ FOLDABILITY/ANALYSIS

structure ⇄ foldings/properties

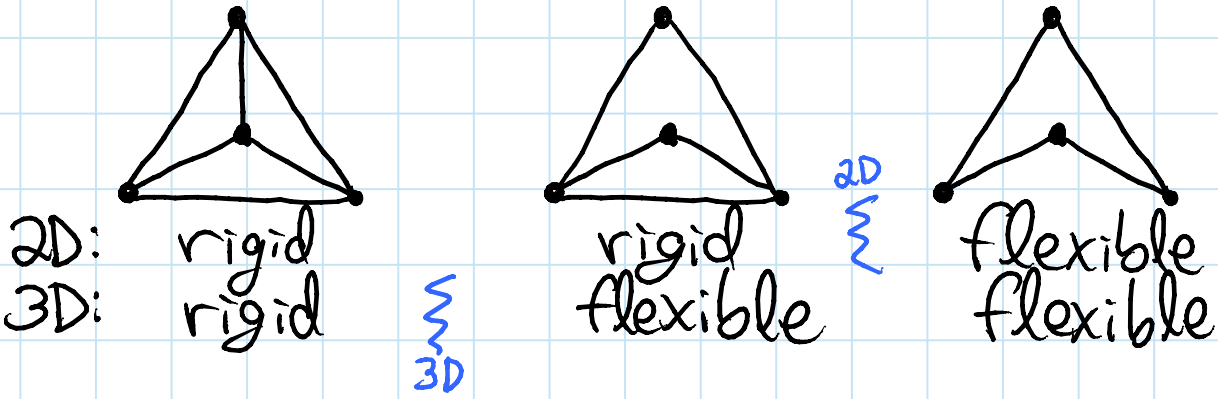
DESIGN/SYNTHESIS ↙
e.g. how to fold desired shape

Result types:

- UNIVERSALITY: Everything is foldable!
  (& here's an algorithm to do it)
- DECISION: Efficient algorithm to decide foldability
- HARDNESS: Computationally intractable to decide foldability

(I) LINKAGES:

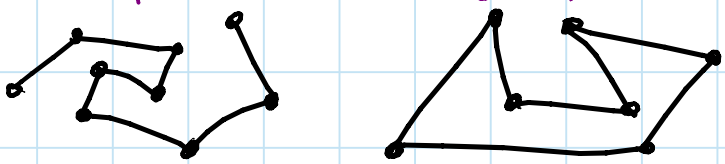
Rigidity: which linkages fold at all?



- efficient characterization in 2D
- OPEN: 3D

Universality: which linkages can fold into *all* possible configurations?

- 2D chains/polygons: UNIVERSAL
  [Connelly, Demaine, Rote 2000; Streinu 2000; Cantarella, Demaine, Iben, O'Brien 2004]



- 3D chains: CAN LOCK (related to protein folding)
  [Cantarella & Johnston 1998]

- $4D^+$ chains: UNIVERSAL
  [Cocan & O'Rourke 2001]

II PAPER:

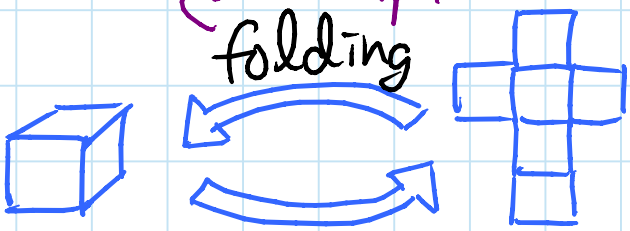
Foldability: which crease patterns fold flat?
- NP-hard [Bern & Hayes 1996] & [Akitaya, Cheung, Demaine, Horiyama, Hull, Ku, Tachi, Uehara 2015]

(⇒ likely no efficient algorithm)

Design: what shapes can be folded?
- universal: any 2D polygon, 3D polyhedron, 2-color pattern (inefficiently) [Demaine, Demaine, Mitchell 2001]
- Origamizer: practical [Tachi 2006; Demaine & Tachi]
- maze folding: fold extruded orthogonal graph with scale factor 3 (independent of maze) [Demaine, Demaine, Ku 2010]
- fold & cut: any set of line segments can be aligned by flat folding [Demaine, Demaine, Lubiw 1998; Bern, Demaine, Eppstein, Hayes 1998]

III Polyhedra:

[Metamorphosis of the Cube]



polyhedron	edge unfolding	general unfolding
convex	OPEN	YES
general	NO	OPEN

IV Hinged Dissections:

- any finite set of polygons of same area can be folded from one chain of polygons (without collision)
  [Abbott, Abel, Charlton, Demaine, Demaine, Kominers 2008]

Solved & open problems!

DEMO COAUTHOR