

Analyse wandelbarer, starrer Faltstrukturen mit Anwendungsbeispielen

Karl-Heinz Brakhage

Kurzanleitung zu den Programmen

Enthaltene Programme / Demos:

foldBagSI.exe	Selbstdurchdringung bei der einfachen Tragetasche
karton.exe	einfacher Faltkarton, unterschiedliche Höhen
simpVert.exe	geometrische Illustration des Belcastro-Hull-Theorems
foldBagExt.exe	flach faltbare Tragetasche, unterschiedliche Höhen
marsDesign.exe	einfaches Beispiel zum MARS-Design
marsDesignNF.exe	Beispiel zum modifizierten MARS-Design – nicht flach faltbar
foldinUW.exe	Faltung / Verdrehung Archimedischer Körper
interaction.mp4	Video von Henri Buffart (Herbst 2015, damals Trako, RWTH)

Vorab:

Die Programme benötigen die DLL freeglut.dll. Das ist eine Variante von GLUT - The OpenGL Utility Toolkit.

Nach Aufruf der Programme muss man zur Aktivierung einmal mit der Maus im Fenster die linke Maustaste drücken.

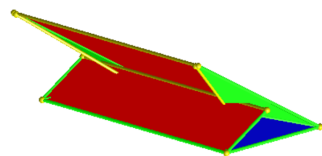
Bewegt man mit gedrücktem linken Button die Maus im Fenster, so ändert sich die Blickrichtung.

In einem Textfenster werden diverse Informationen ausgegeben. Diese kann man ignorieren. Wer möchte kann aber die Änderungen der Werte des Startfaltwinkels ρ und bei marsDesignNF die Änderungen des Winkels Δ beobachten.

Zu fast allen Programmen gibt es auch Erläuterungen im Vortragsskript.

Allgemeine Befehle bei allen Programmen:

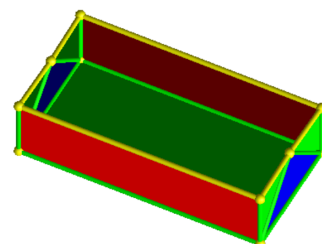
v	show vertices an / aus
e	show edges an / aus
f	show faces an / aus
n	show normals an / aus
+/-	größer / kleiner
a	Automatisches drehen der Blickrichtung (außer bei foldinUW)



foldBagSI:

Ziel: Zeige die Selbstdurchdringung bei der einfachen Tragetasche

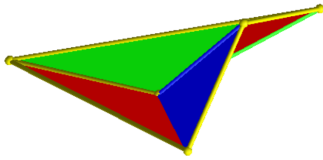
r	rotate in (decrease ρ)
R	rotate out (increase ρ)



karton:

Ziel: Demonstration der Faltmöglichkeiten an einem einfachen Beispiel

r	rotate in (decrease ρ)
R	rotate out (increase ρ)
1	Standardkarton
2	hoher Karton – nicht zusammenfaltbar
3	kleiner Karton – nicht zusammenfaltbar
4	Modifikation kleiner Karton – zusammenfaltbar



simpVert:

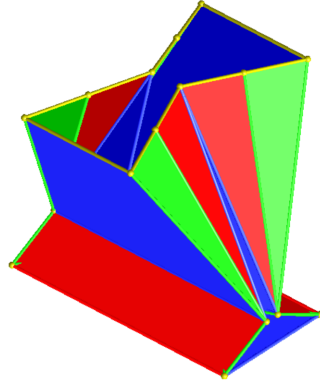
Ziel: Zeige die geometrische Idee des Belcastro-Hull-Theorems. Dreht immer abwechselnd um die z- und x-Achse weiter.

r	rotate to next status
---	-----------------------

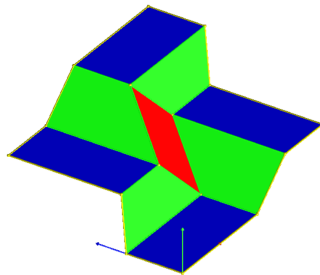
foldBagExt:

Ziel: Demonstration der Faltmöglichkeiten an einem erweiterten Modell der Falttasche

Zuerst werden die oberen äußeren Seitenflächen ausgeklappt (in einem Schritt). Durch weiteres Nachaußenziehen der oberen äußeren Seitenflächen wird der obere Bereich maximal entfaltet (bis $\rho \approx 86^\circ$ bei 1, $\rho \approx 87^\circ$ bei 2 und $\rho \approx 80^\circ$ bei 3). Danach muss die Tasche von den anderen Seiten unten eingedrückt werden. Bei $\rho \approx 53^\circ$ kommt es bei Tasche 1 zur Berührung im oberen Bereich. Bei Tasche 2 liegt ab ca. 69.5° bis 33° eine Selbstdurchdringung vor. Tasche 3 hat keine Selbstberührung.



r	rotate in (decrease ρ)
R	rotate out (increase ρ)
1	maximale Höhe für zusammenfaltbar (ohne Selbstdurchdringungen)
2	zu hoch \rightarrow Selbstdurchdringungen
3	niedriger als <i>maximale Höhe</i> – zusammenfaltbar



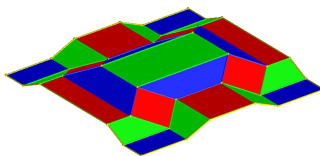
marsDesign:

Ziel: Zeige eine einfache Anwendung, die auf dem MARS-design basiert. Das Gebilde ist also von flach nach flach faltbar.

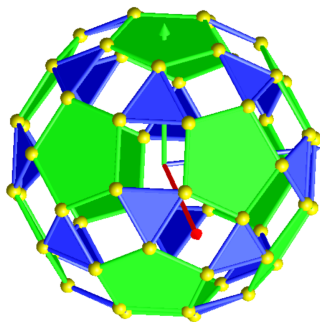
r	rotate in (decrease ρ)
R	rotate out (increase ρ)

marsDesignNF:

Ziel: Zeige eine einfache Faltstruktur die faltbar, aber nicht von flach nach flach faltbar ist. Sie beruht auf einer Modifikation des MARS-designs. Es ist flach zusammenfaltbar – etwa für den Transport –, aber nicht in eine ebene Ausgangslage faltbar. Es kann in verschiedenen Positionen fixiert werden und so z.B. als leicht erhöhte Auftrittsbühne dienen. Dieses Design ist in Zusammenarbeit mit Henri Buffart (damals Trako, RWTH) entstanden.



r	rotate in (decrease ρ)
R	rotate out (increase ρ)
d	decrease Δ
D	increase Δ



foldinUW:

Ziel: Zeigt (in Urform) die Leistungsfähigkeit vom Algorithmus.

→	next status
T	Tetraeder
t	Kuboktaeder - Tetraeder schon ab abgestumpft und abgeschrägt
	H,h, O,o, D,d, I,i – die übrigen Körper (Hexaeder bis Ikosaeder)
F	Abschrägungsflächen (Vierecke) an / aus (mit Diagonale)

Wenn man die Platonischen Körper mit großen Buchstaben aufruft, so ist der Übergang zum nächsten Status die gleichzeitige Abstumpfung der Vertices und Abschrägung der Kanten. Mit dem entsprechenden kleinen Buchstaben landet man direkt in dem Status. In dieser Konstellation ist der Körper nicht (starr) faltbar / verdrehbar.

Beim nächsten Status sind die Abschrägungsflächen entfernt. Jeder der 5 so entstehenden Körper hat jetzt genau 6 Freiheitsgrade. Beispiel Kuboktaeder: 12 Vertices $\rightarrow 12 \cdot 3 - 6 = 30$ Freiheitsgrade (DOF). (-6 wegen der beliebigen Lage im Raum) Die 8 Dreiecke liefern, da sie sich nur punktuell berühren, wegen ihrer insgesamt 24 Kanten auch 24 Bedingungen. Es bleiben also 6 DOF.

Die *Faltung* ist also nicht eindeutig. Der Algorithmus funktioniert dennoch, weil wir die Faltung (die Winkelkurve) durch die Wahl der kleinsten 2-Norm Lösung eindeutig machen. Je nach erster Störung zum Start in die Kurve können aber *merkwürdige* Körper entstehen. Durch geschickte Wahl von 6 Diagonalen, die man gleichmäßig zusammenzieht, wird die *Faltung* eindeutig.

Beim Dodekaeder und Ikosaeder ergeben sich so 180 Unbekannte und 174 Gleichungen, ggf. plus 6 für die Diagonalen. Wenn man die Gleichmäßigkeit haben möchte und mit einbaut, kann man das System auf eine Gleichung für 2 Unbekannte reduzieren. Die Bestimmungsstücke dafür werden in den nächsten beiden Zuständen zunächst ein und dann wieder ausgeblendet. Danach wird durch jedes Drücken der Taste → soweit gedreht bzw. zurückgedreht, bis der nächste Archimedische Körper erreicht ist.