Programmierung von CAx-Systemen

Übung 3: Geometrie – Kurven und Transformationen

David Straub

Lernziele

Nach dieser Übung können Sie:

- Geometrietypen von Kanten und Flächen über `geom_type` ablesen und interpretieren
- Punkte und Tangenten auf parametrischen Kurven mit `position_at` und `tangent_at` berechnen
- Transformationen (Verschieben, Drehen, Spiegeln) anwenden und verketten
- Konstruktionsebenen aus vorhandenen Flächen ableiten
- Versatzkurven erzeugen und deren Geometrietyp interpretieren

Startcode

```
import build123d as bd
from build123d import Axis, Plane, Location, Vector
```

Aufgabe 1: Geometrie im B-Rep ablesen

1.1 – Zylinder analysieren

Erstellen Sie einen Zylinder (`radius=15`, `height=30`) und geben Sie aus:

1. Für jede **Fläche**: `geom_type` und Flächeninhalt (`area`)
2. Für jede **Kante**: `geom_type` und Länge (`length`)

Beantworten Sie: - Wie viele Kanten hat der Zylinder? Welche Geometrietypen kommen vor? - Welche Kante hat `geom_type = LINE`? Was stellt sie geometrisch dar?

Hinweise: `.faces()`, `.edges()`, `.geom_type`, `.area`, `.length`

1.2 – Vergleich mit anderen Körpern

Erstellen Sie einen **Quader** (`Box(30, 20, 10)`) und eine **Kugel** (`Sphere(radius=10)`).

1. Welche `geom_type`-Werte kommen bei Flächen und Kanten vor?
2. Warum hat die Kugel nur **eine** Fläche? Überprüfen Sie es.
3. Vergleichen Sie mit dem Zylinder: Was haben alle drei gemein, was unterscheidet sie?

Hinweise: `set(f.geom_type for f in ...)`, `len(...)`

1.3 – Kegel

Erstellen Sie einen Kegel:

```
kegel = bd.Cone(bottom_radius=20, top_radius=0, height=30)
```

1. Welchen `geom_type` hat die Mantelfläche? Was erwarten Sie – und warum?
2. Wie viele Kanten hat der Kegel? Welche Typen?
3. Vergleichen Sie mit dem Zylinder: Was fehlt beim Kegel und warum?

Hinweise: `.geom_type`, `.faces()`, `.edges()`

Aufgabe 2: Parametrische Kurven erkunden

2.1 – Punkte auf einer Kreiskante

Nehmen Sie den Zylinder aus Aufgabe 1.1. Selektieren Sie eine Kreiskante und berechnen Sie `position_at` für $t \in \{0{,}0;\ 0{,}25;\ 0{,}5;\ 0{,}75\}$.

1. Auf welcher Höhe liegen die Punkte? Auf dem Deckel oder dem Boden?
2. Berechnen Sie die Abstände zwischen aufeinanderfolgenden Punkten. Sind sie gleich?
3. Erklärt das Ergebnis, warum der Parameter t **nicht** dem Bogenmaß entspricht?

Hinweise: `.filter_by(bd.GeomType.CIRCLE)[0]`, `.position_at(t)`, `(p2 - p1).length`

2.2 – Tangente und Normalenrichtung

Berechnen Sie für dieselbe Kreiskante `tangent_at(t)` bei $t \in \{0{,}0;\ 0{,}25;\ 0{,}5;\ 0{,}75\}$.

1. In welche Richtung zeigt die Tangente bei $t = 0{,}0$? Macht das geometrisch Sinn?
2. Berechnen Sie den Radiusvektor (Richtung vom Kreismittelpunkt zum Punkt) und das Skalarprodukt mit der Tangente. Was ergibt sich – und warum?

Hinweise: `.tangent_at(t)`, `Vector(0, 0, z)`, `.dot()`, `.normalized()`

2.3 – Ellipse: gleichmäßiger Parameter, ungleichmäßige Punkte?

Erstellen Sie eine Ellipse und berechnen Sie 8 gleichmäßig verteilte Parameterpunkte:

```
ellipse = bd.Edge.make_ellipse(x_radius=30, y_radius=15)
punkte = [ellipse.position_at(i / 8) for i in range(8)]
```

1. Berechnen Sie die Abstände zwischen aufeinanderfolgenden Punkten.
2. Sind die Abstände gleichmäßig? Was sagt das über die Parametrisierung aus?
3. Wo auf der Ellipse liegen die „dichteren" Punkte – bei großer oder kleiner Krümmung?

Hinweise: `(p2 - p1).length`, Folie „Parametrische Kurven"

Aufgabe 3: Transformationen

3.1 – Reihenfolge von Drehen und Verschieben

Erstellen Sie einen Zylinder (`radius=10`, `height=20`) und führen Sie die gleichen Operationen in **unterschiedlicher Reihenfolge** durch:

- Variante A: 45° um die Z-Achse drehen, dann um $(30, 0, 0)$ verschieben
- Variante B: um $(30, 0, 0)$ verschieben, dann 45° um die Z-Achse drehen

1. Wo liegt der Mittelpunkt (`center()`) jeweils?
2. Erklären Sie anhand der Formel $\mathbf{TR} \neq \mathbf{RT}$ den Unterschied.

Hinweise: `.rotate(Axis.Z, winkel)`, `.move(Location((dx, dy, dz)))`, `.center()`

3.2 – Konstruktionsebene aus Fläche

Erstellen Sie einen Quader (`Box(40, 30, 10)`) und leiten Sie eine Konstruktionsebene von der **Oberseite** ab. Platzieren Sie darauf einen Zylinder (`radius=8`, `height=15`).

1. Wo liegt der Ursprung der abgeleiteten Ebene?
2. Verbinden Sie Quader und Zapfen. Wie viele Flächen hat der Gesamtkörper?
3. **Diskussion:** Was wäre der Nachteil, wenn Sie statt `Plane(oberseite)` direkt `Plane.XY.offset(5)` verwendet hätten?

Hinweise: `.faces().sort_by(Axis.Z).last`, `Plane(face)`, `ebene * bd.Cylinder(...)`, `+` oder `fuse`

3.3 – Spiegelung

Bauen Sie eine asymmetrische Form aus zwei Quadern (sie sollen nicht spiegelsymmetrisch sein). Spiegeln Sie die Form an `Plane.YZ` und verbinden Sie Original und Spiegelbild.

1. Überprüfen Sie, dass Original und Spiegelbild zusammen eine symmetrische Form ergeben.
2. Welche Flächen „verschwinden” beim Verbinden?

Hinweise: `.mirror(Plane.YZ)`, `bd.Pos(x, y) * ...`, `+`

Aufgabe 4: Versatzkurven

4.1 – Versatz eines Kreises

Erstellen Sie einen freistehenden Kreis und berechnen Sie den Versatz:

```
kreis = bd.Edge.make_circle(radius=20)
versatz = kreis.offset_2d(5)
```

1. Welchen `geom_type` haben die resultierenden Kanten? Was bedeutet das geometrisch?
2. Warum liefert `offset_2d` einen `Wire` (mehrere Kanten) statt einer einzelnen Kante?
3. Wie groß ist der Radius der Versatz-Kanten? Berechnen Sie ihn aus der Länge.

Hinweise: `.edges()`, `.geom_type`, `.length`, $r = l/(2\pi)$ (voller Kreis) oder $r = l/\pi$ (Halbkreis)

4.2 – Versatz einer Ellipse

Erstellen Sie eine Ellipse (`x_radius=30`, `y_radius=15`) und versetzen Sie sie um 5.

1. Welchen `geom_type` haben die resultierenden Kanten? Unterschied zu 4.1?
2. Warum ergibt der Versatz keine analytische Ellipse mehr? (Folie „Versatz einer Ellipse")
3. Vergrößern Sie den Versatz auf 20. Was passiert und warum?

Hinweise: `.offset_2d(d)`, `.edges()`, `.geom_type`

Zusatzaufgabe: Lochkreis analysieren

Ringscheibe mit Bohrungen

Erstellen Sie eine Ringscheibe mit 6 Bohrungen auf einem Lochkreis:

```
import math

scheibe = bd.Cylinder(radius=40, height=8) - bd.Cylinder(radius=12, height=8)

for winkel in range(0, 360, 60):
    x = 28 * math.cos(math.radians(winkel))
    y = 28 * math.sin(math.radians(winkel))
    scheibe = scheibe - bd.Pos(x, y) * bd.Cylinder(radius=4, height=8)
```

Geometrieanalyse

1. Wie viele Kanten hat die Scheibe? Welche Geometrietypen kommen vor und wie oft?
2. Selektieren Sie alle Kreiskanten und bestimmen Sie die verschiedenen **Radien**. Welche Radien erwarten Sie – und passen sie zu den Konstruktionsmaßen?
3. Verrunden Sie alle **oberen** Kreiskanten (Bohrungseintritt oben) mit `radius=0.5`.

Hinweise: `collections.Counter`, `.filter_by(bd.GeomType.CIRCLE)`, `.radius`, `.filter_by_position(Axis.Z, minimum=..., maximum=...)`, `bd.fillet(..., radius=...)`