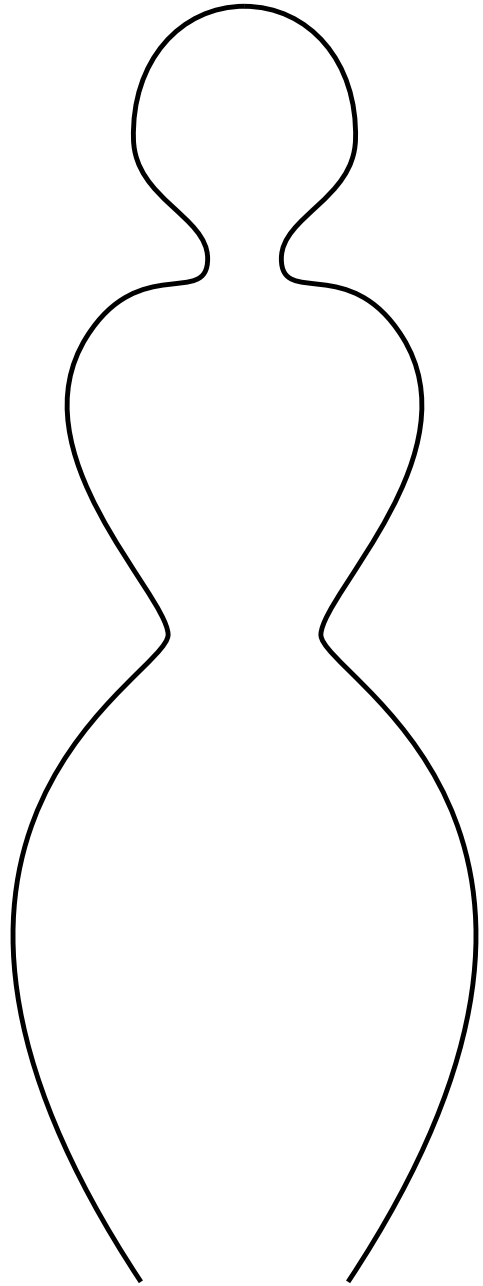
Ebenenbasiertes Kantenrouting mit Splines

Tibor Toepffer

27.08.2014

Schöne Kurven

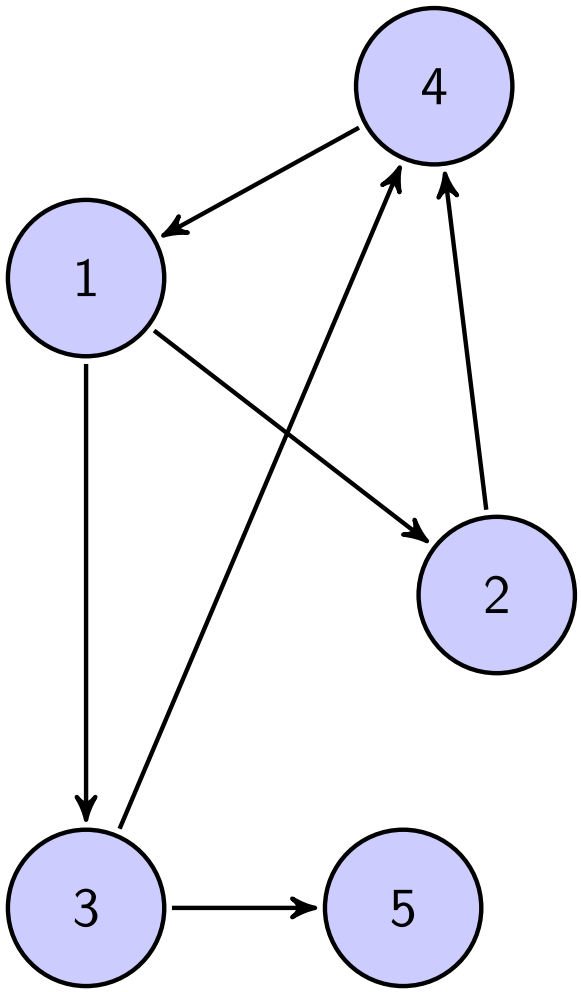


KLay Layered

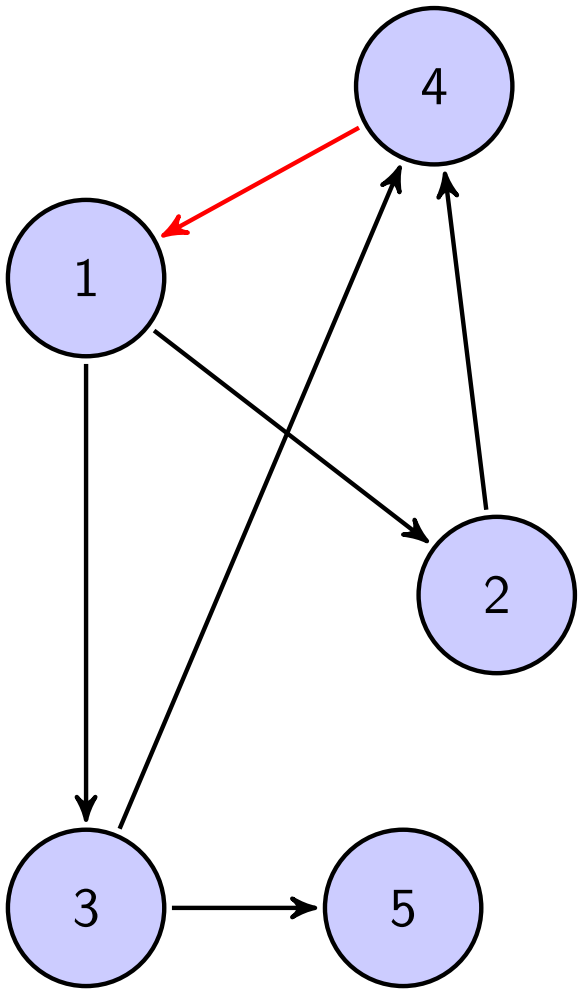
KLay Layered

Ebenbasiertes Layout von Graphen nach Kozo Sugiyama.

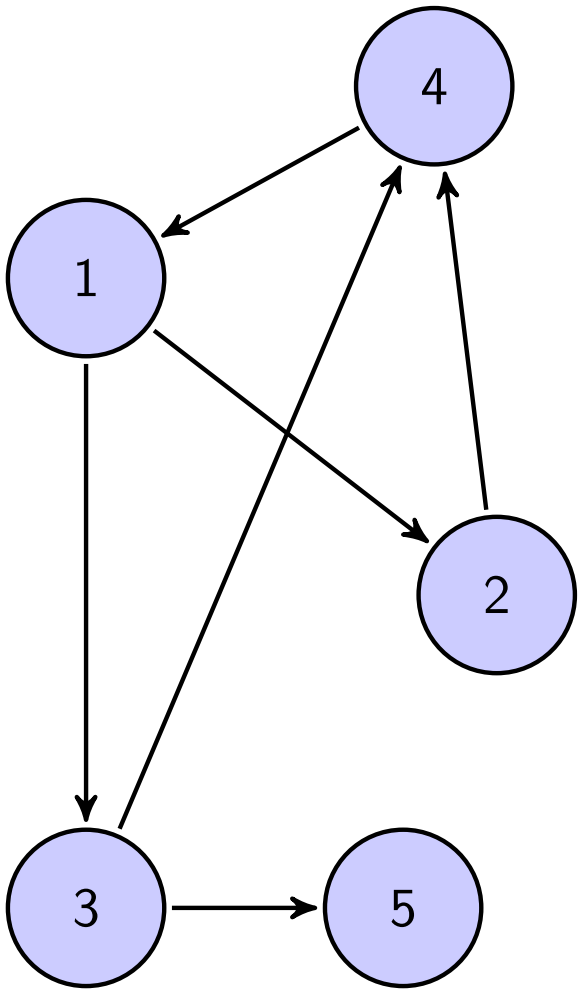
Phase 1: Zyklen auflösen

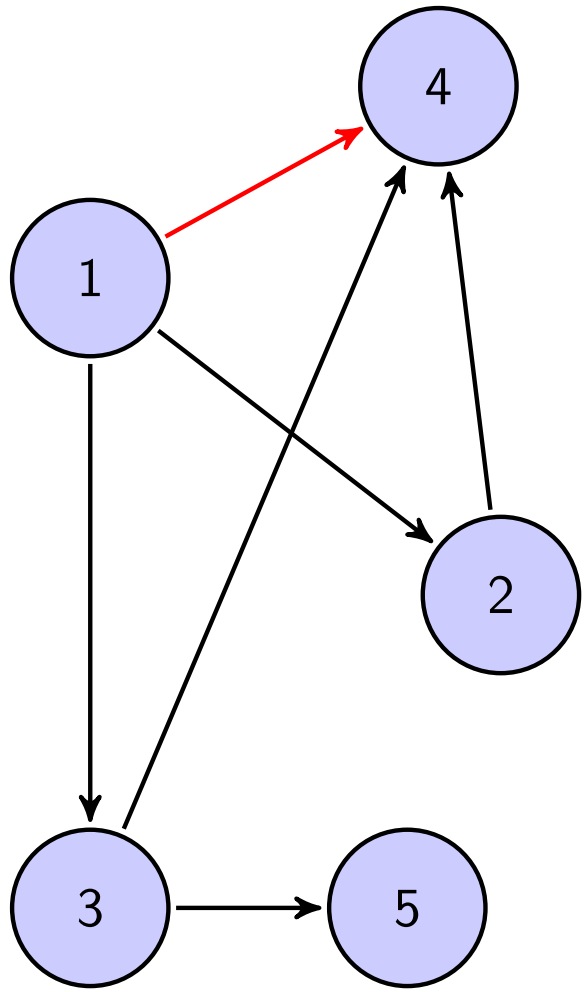


Phase 1: Zyklen auflösen

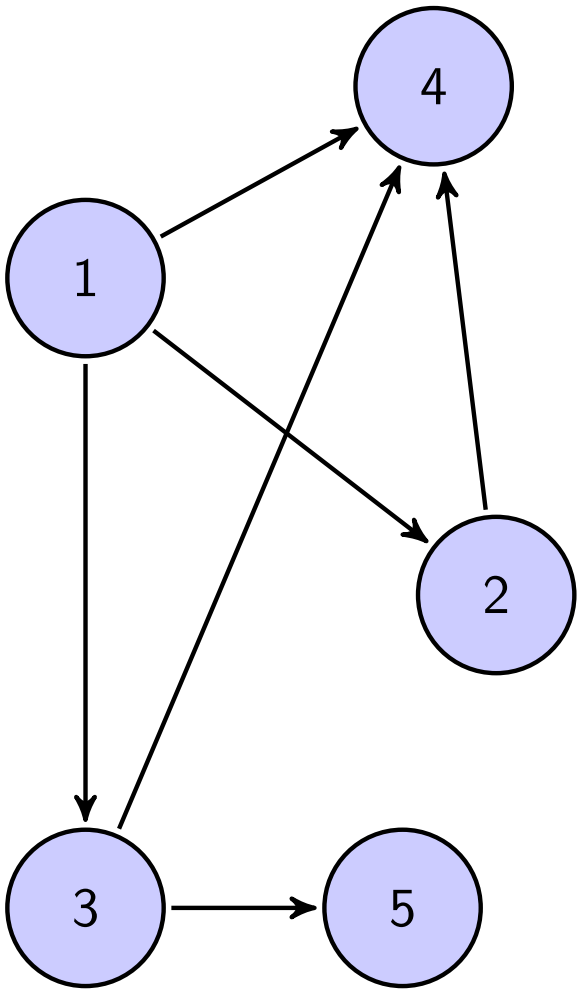


Phase 1: Zyklen auflösen

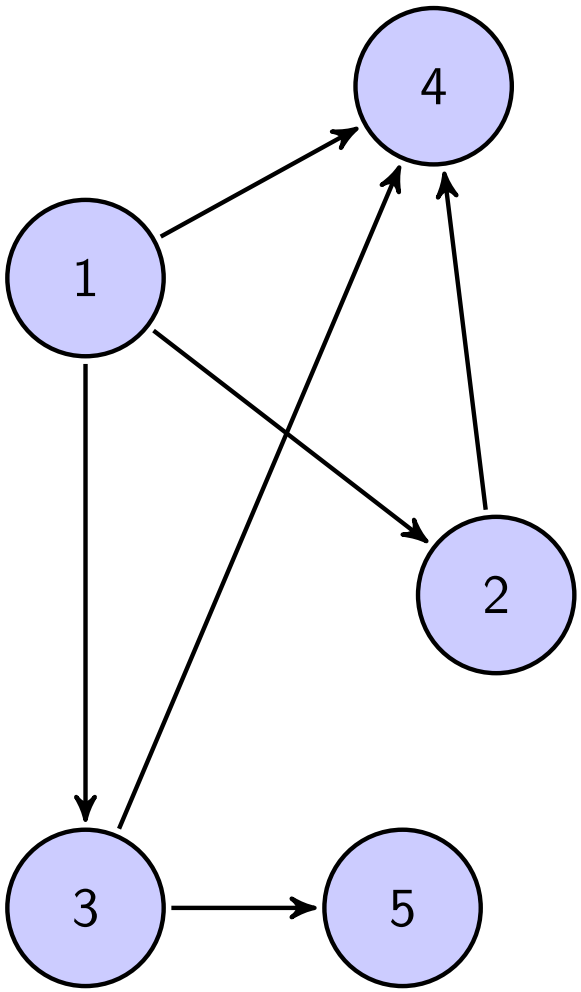


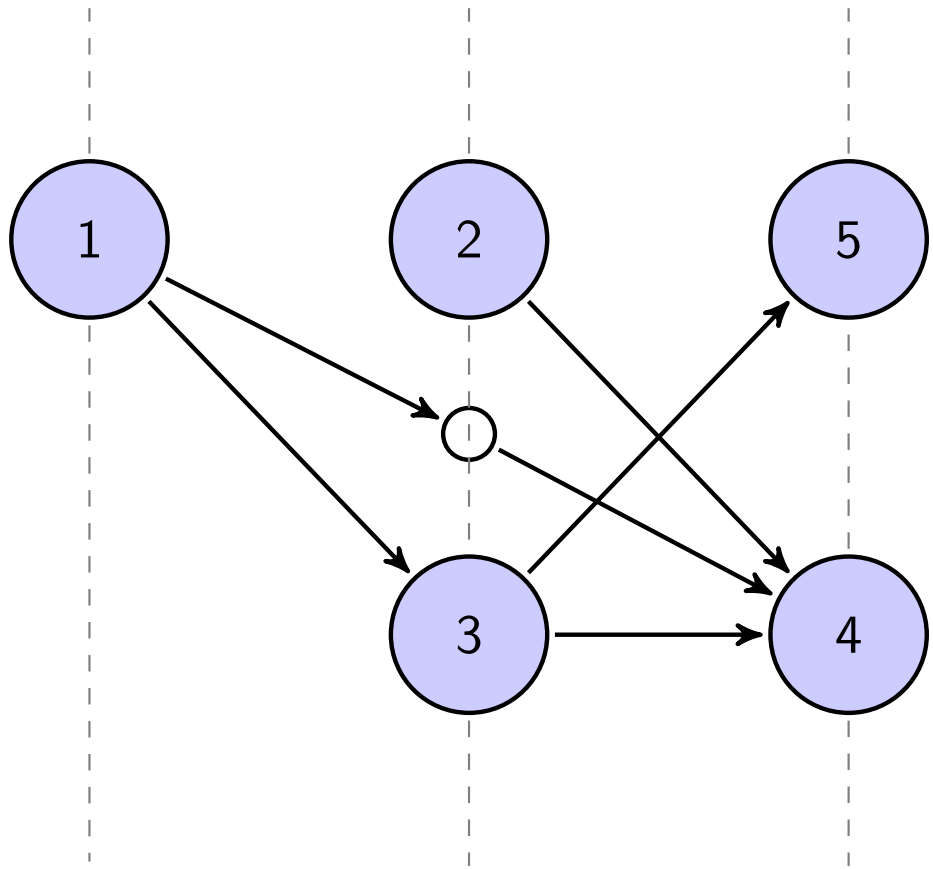


Phase 2: Knoten auf Ebenen verteilen

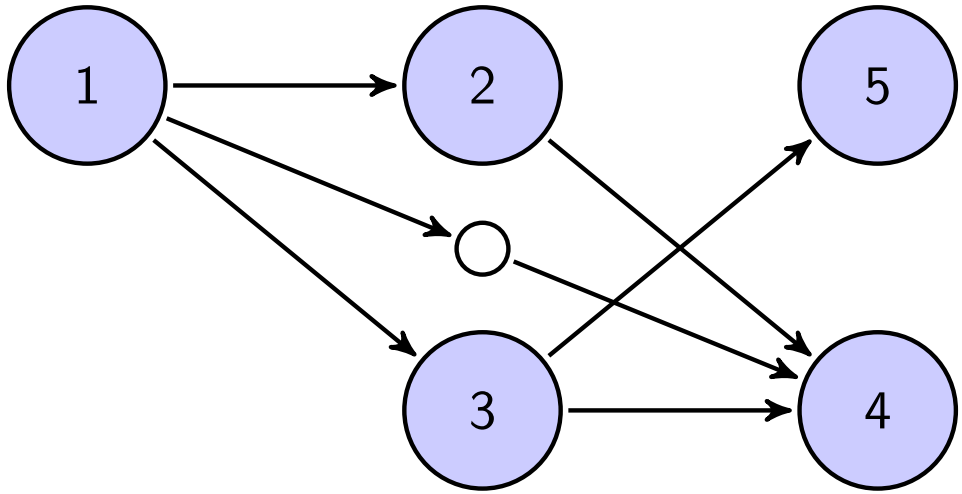


Phase 2: Knoten auf Ebenen verteilen

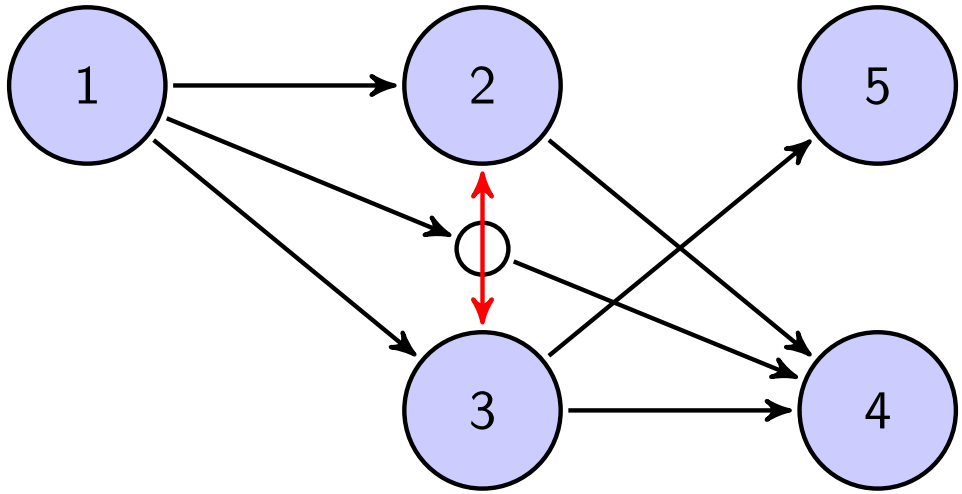




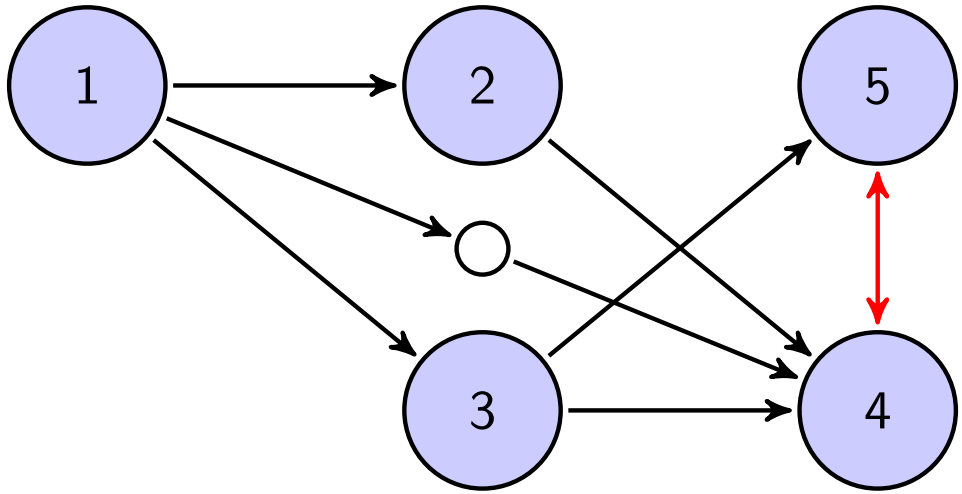
Phase 3: Kantenkreuzungen minimieren



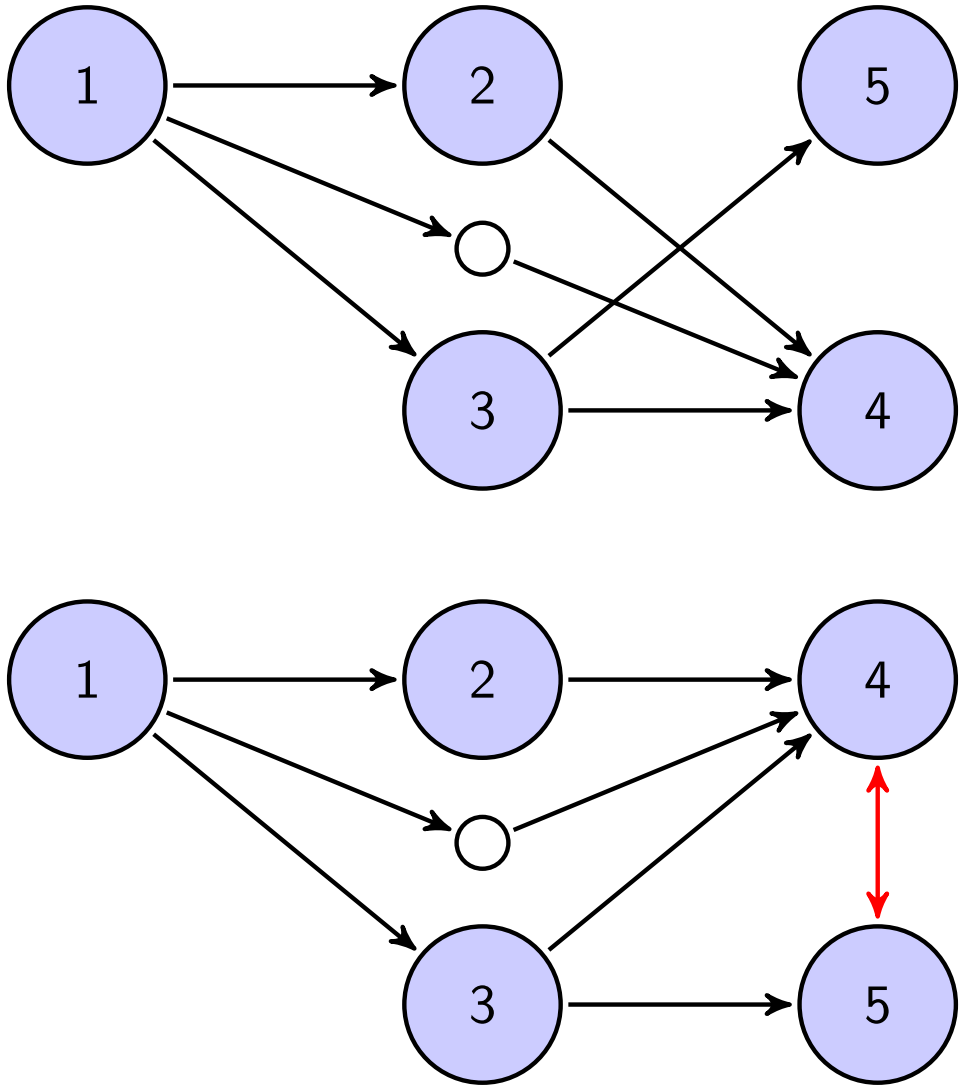
Phase 3: Kantenkreuzungen minimieren



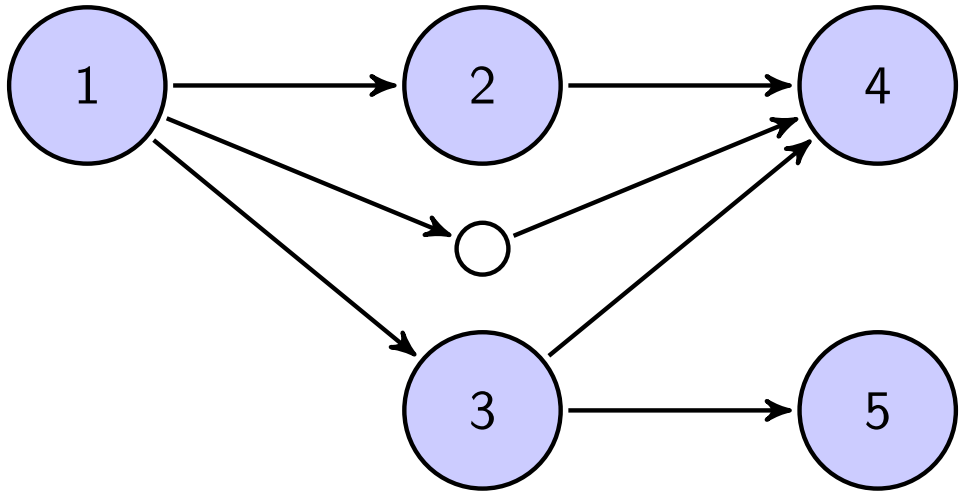
Phase 3: Kantenkreuzungen minimieren



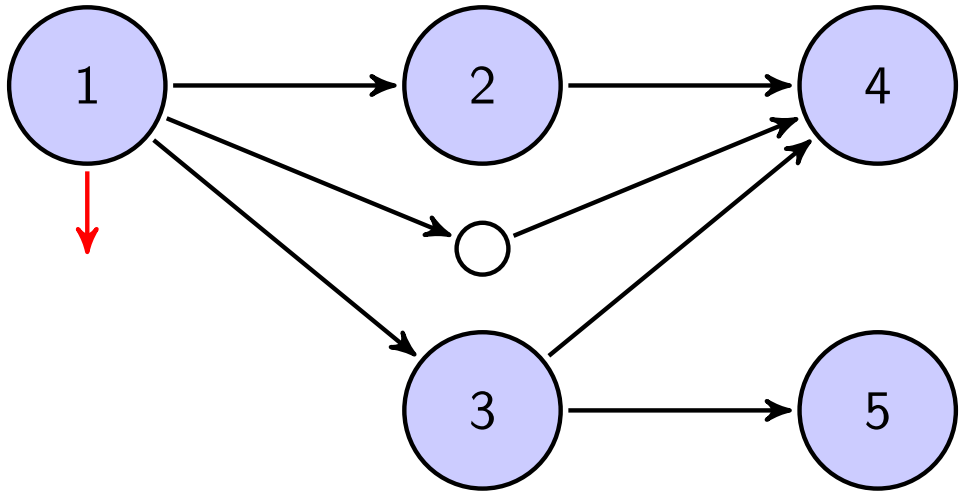
Phase 3: Kantenkreuzungen minimieren



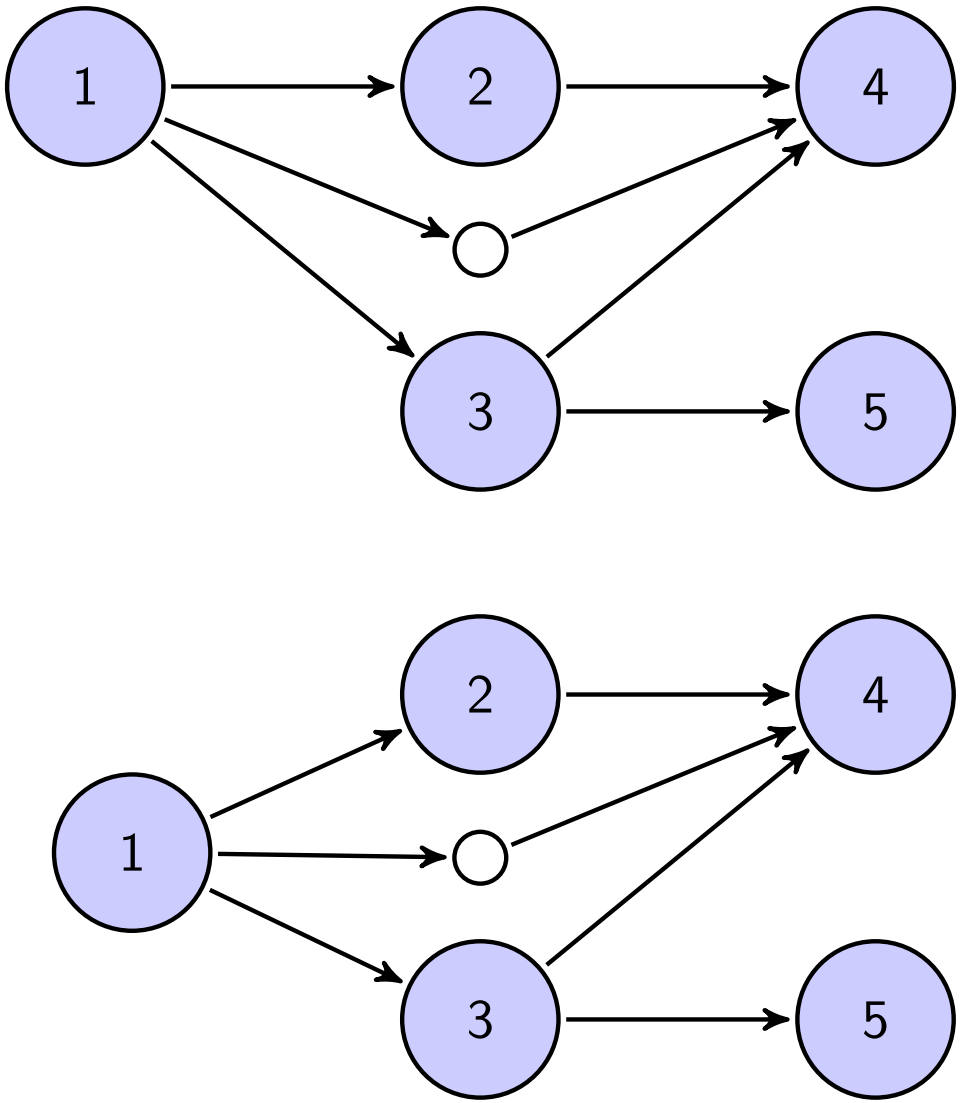
Phase 4: Vertikale Knotenposition festlegen



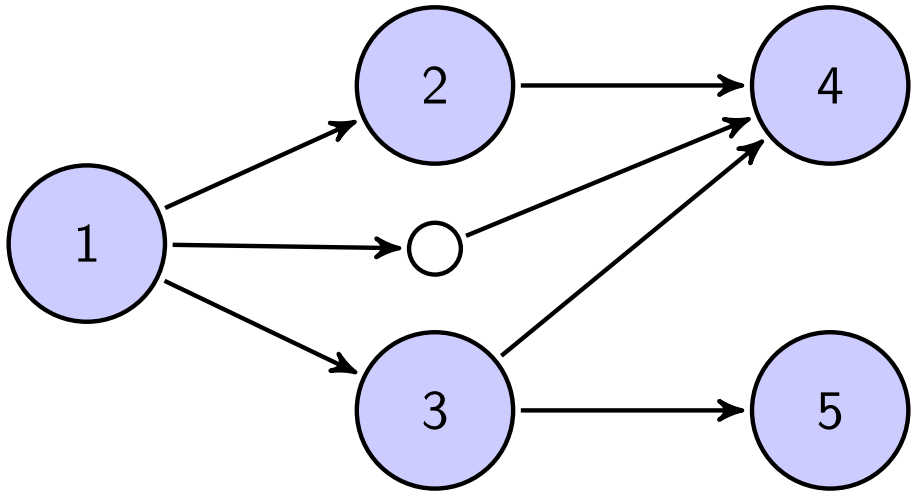
Phase 4: Vertikale Knotenposition festlegen



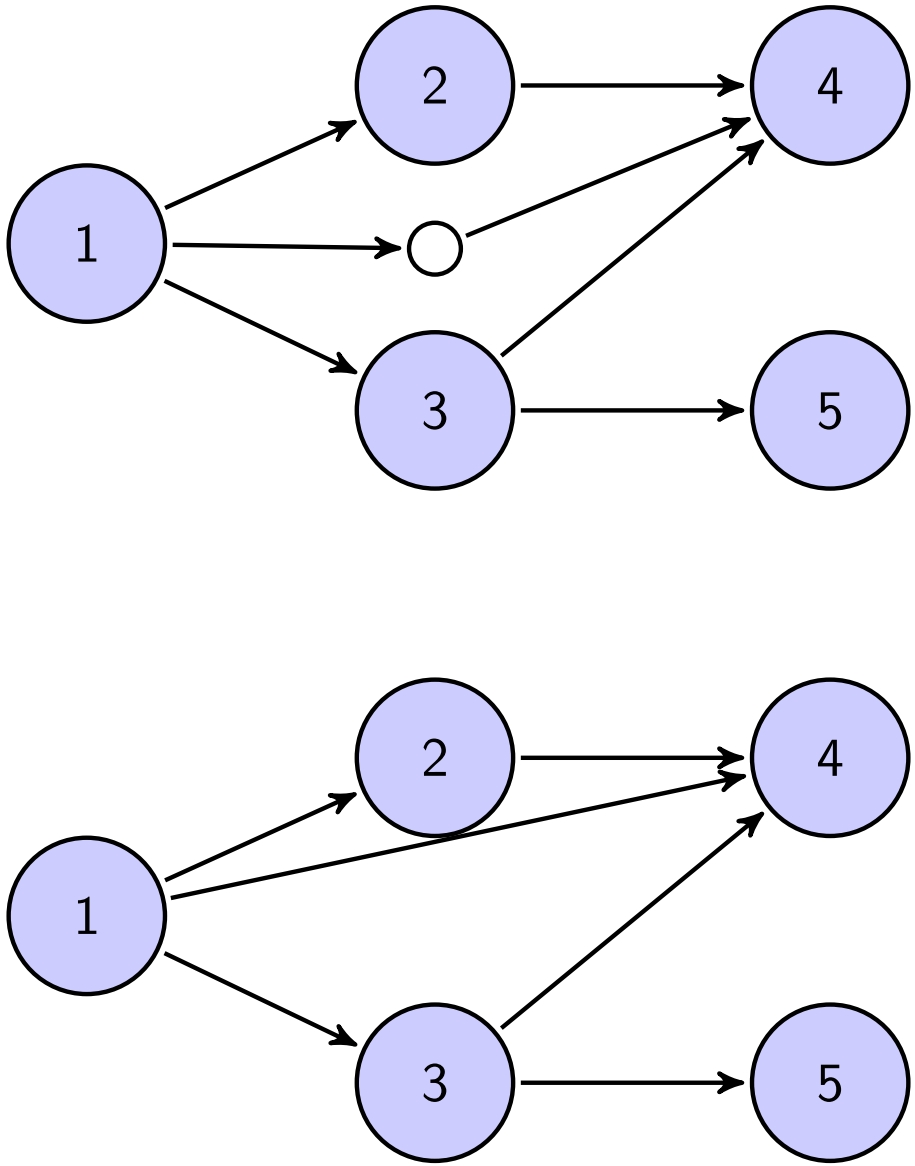
Phase 4: Vertikale Knotenposition festlegen



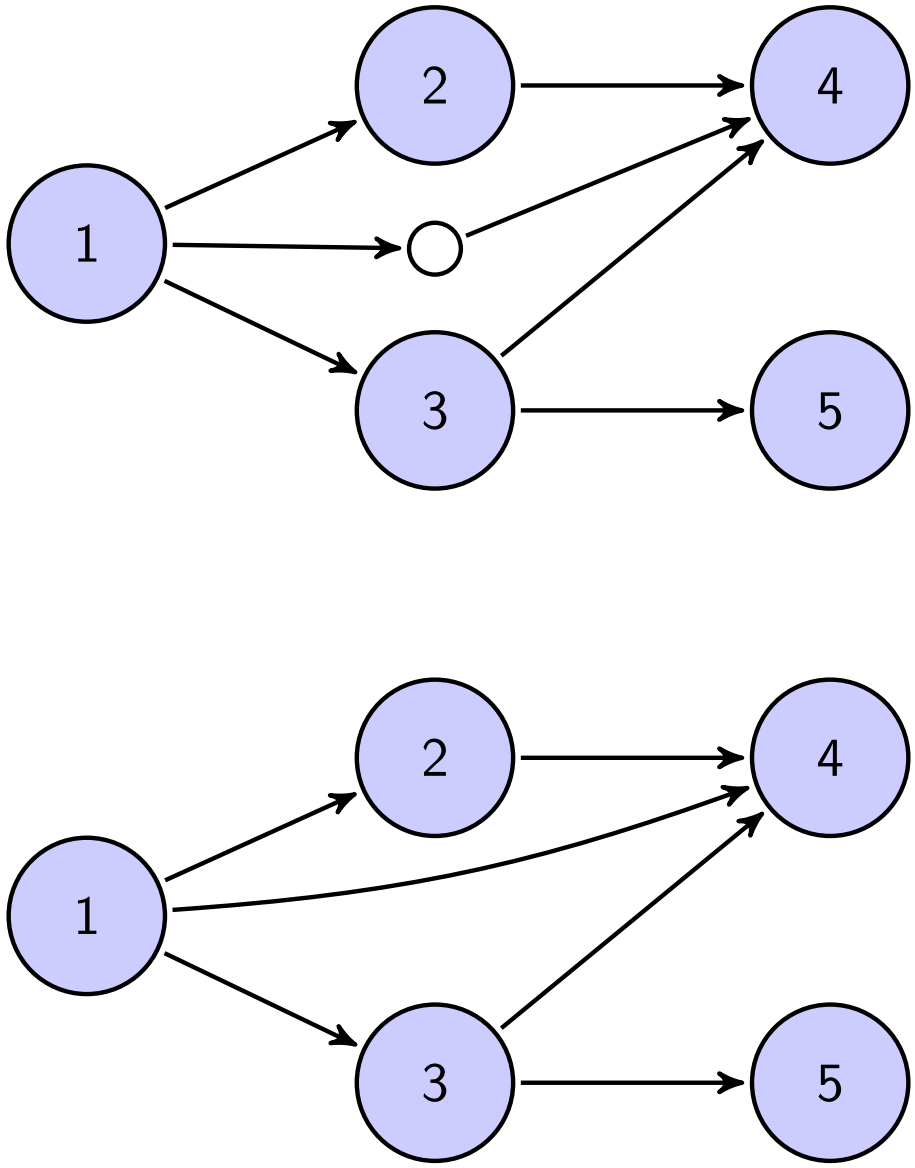
Phase 5: Kanten routen



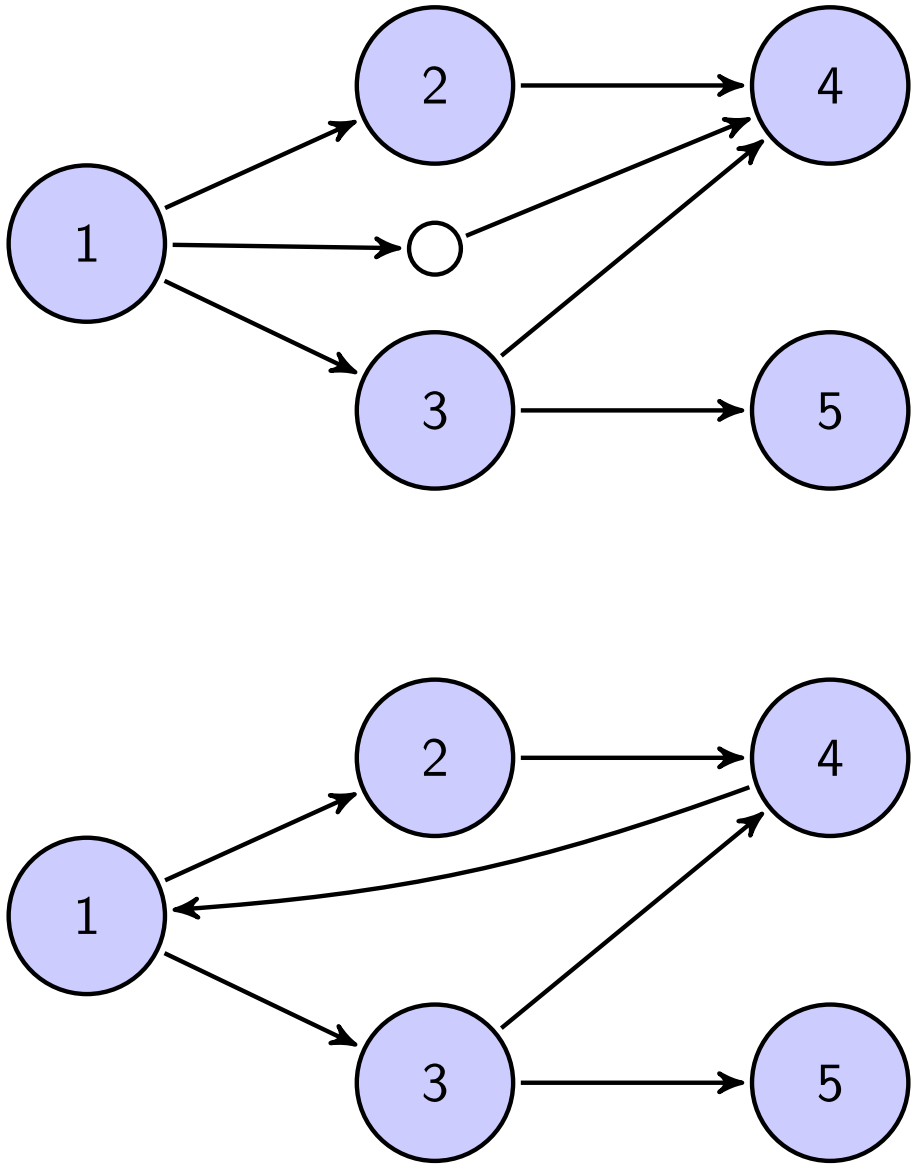
Phase 5: Kanten routen



Phase 5: Kanten routen



Phase 5: Kanten routen

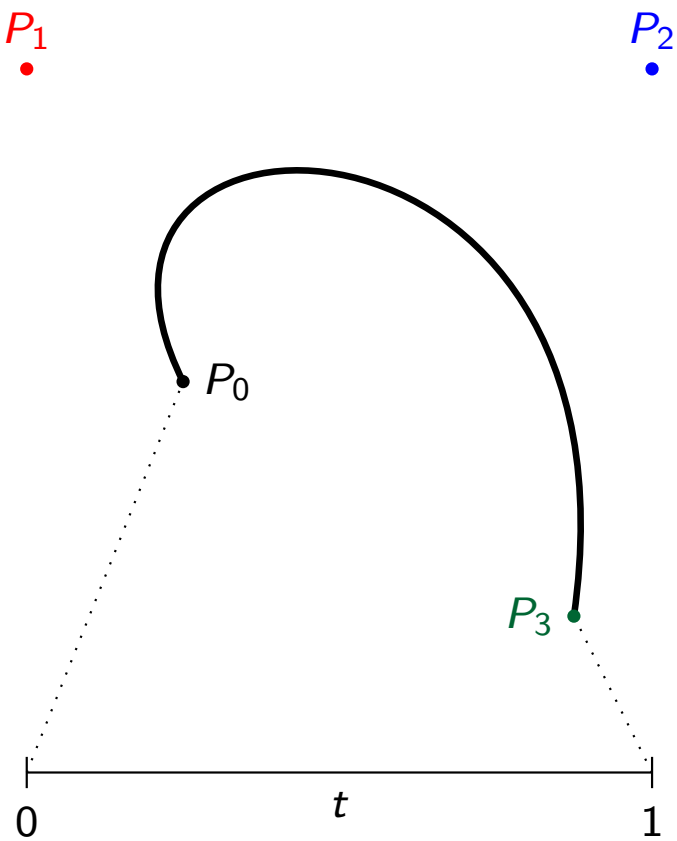


Erkenntnis vom letzen Vortrag?

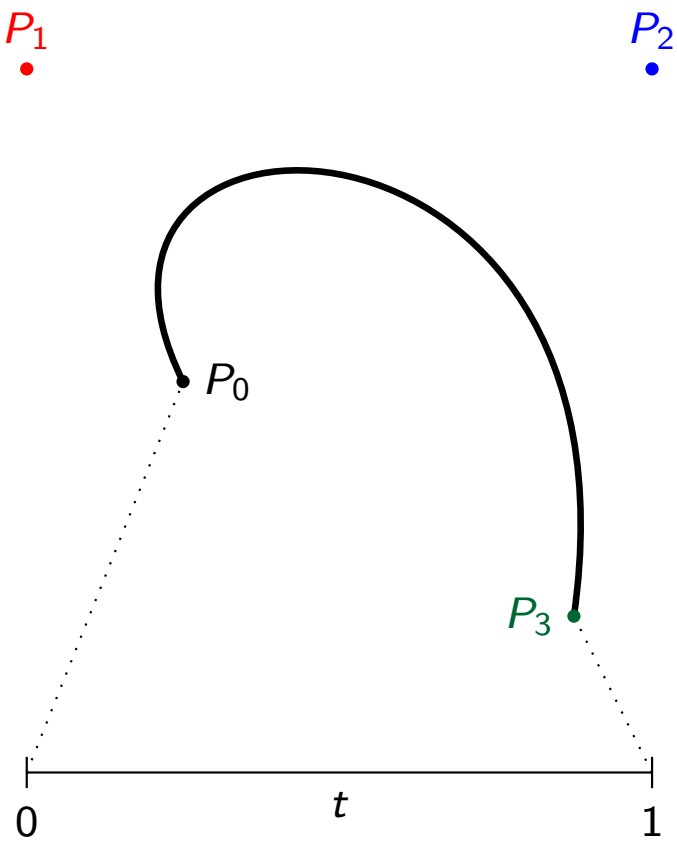
Erkenntnis vom letzen Vortrag?

Es gibt viele Arten von Splines!

Bezier-Kurven

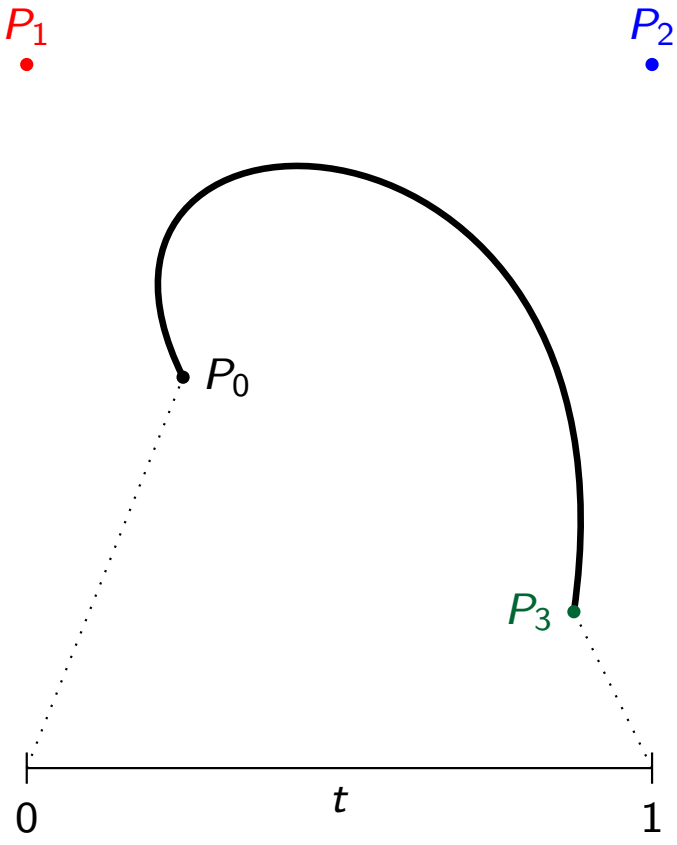


Bezier-Kurven

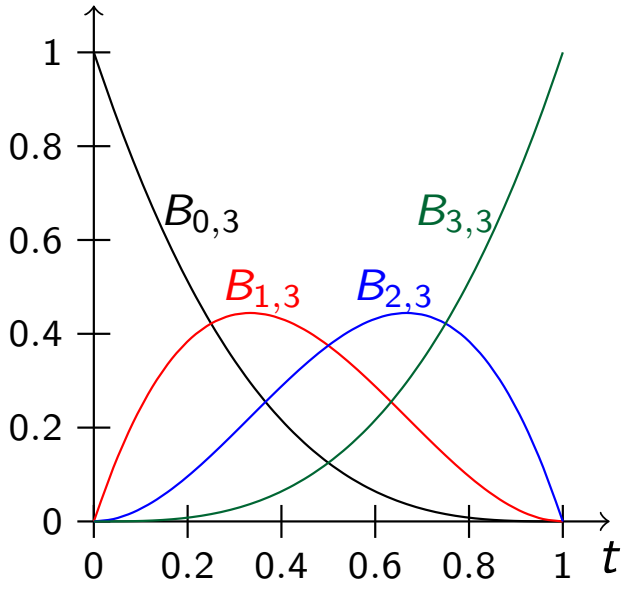


$$Bezier(t) = \sum_{i=0}^{k} B_{i,k}(t)\mathbf{P_i}$$

Bezier-Kurven

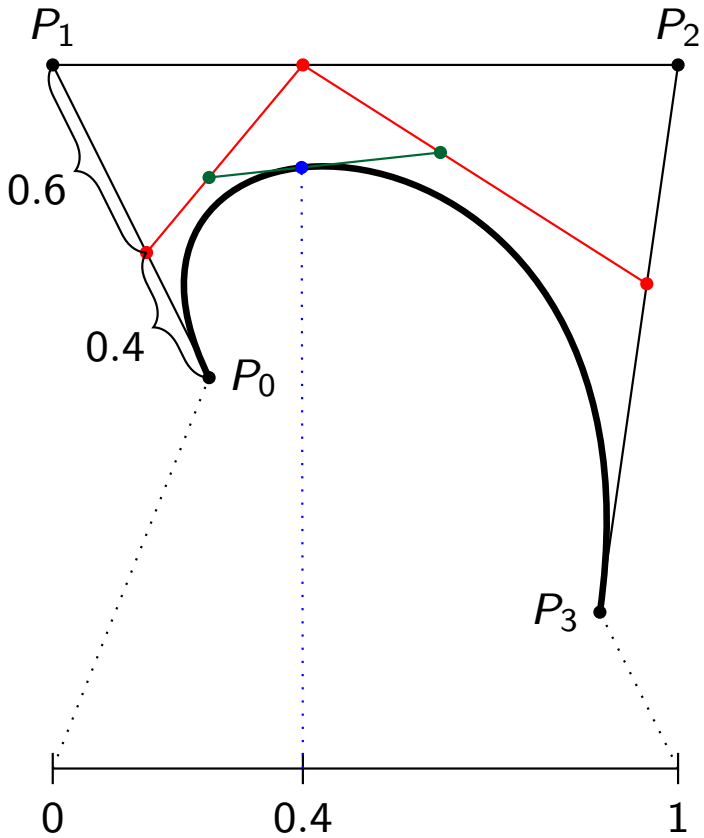


$$Bezier(t) = \sum_{i=0}^{k} B_{i,k}(t)\mathbf{P_i}$$



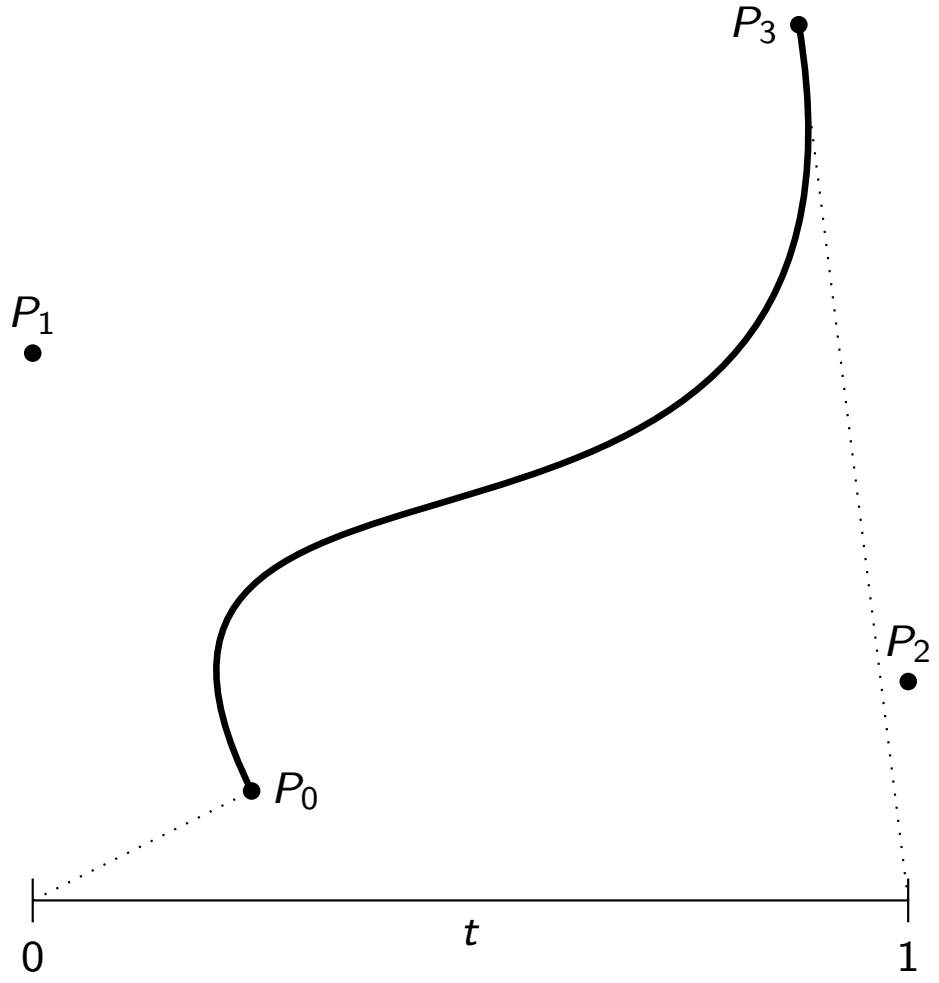
Bernstein Polynome

DeCastelau Algorithmus

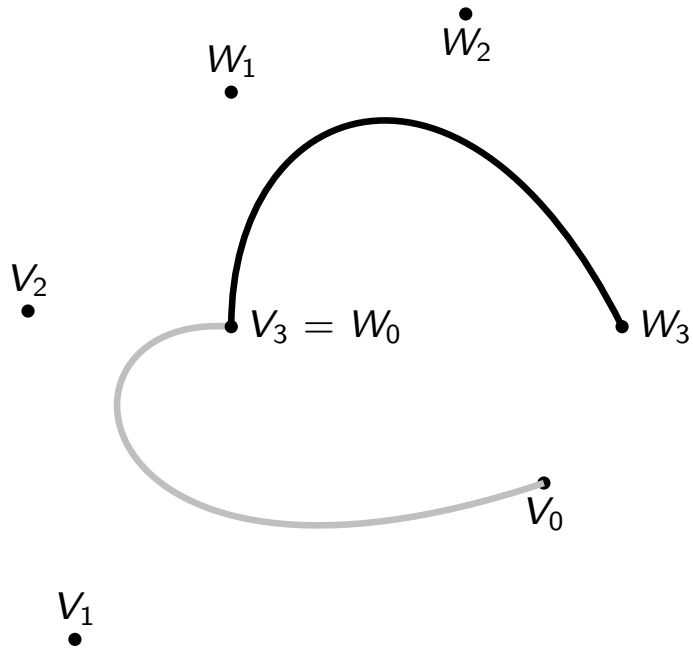


Beispiel für $t = 0.4$

Warum Splines?

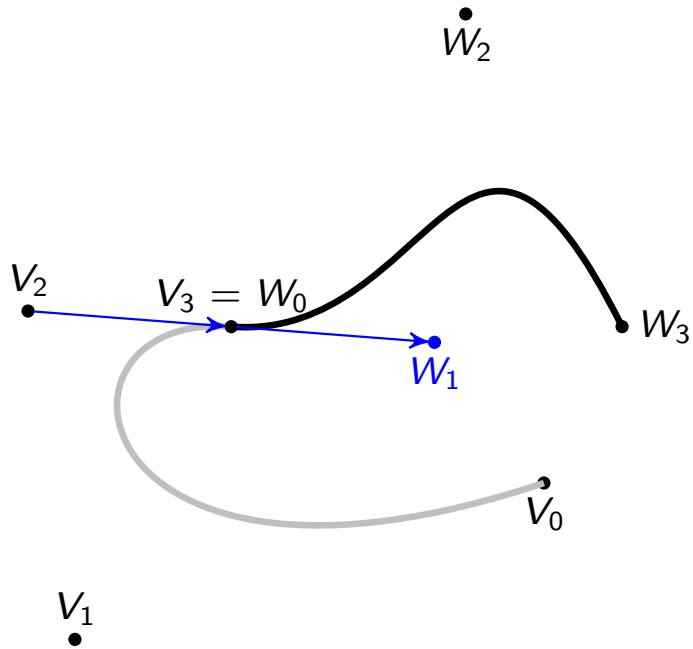


Bezier-Kurven $\Rightarrow$ Bezier-Splines



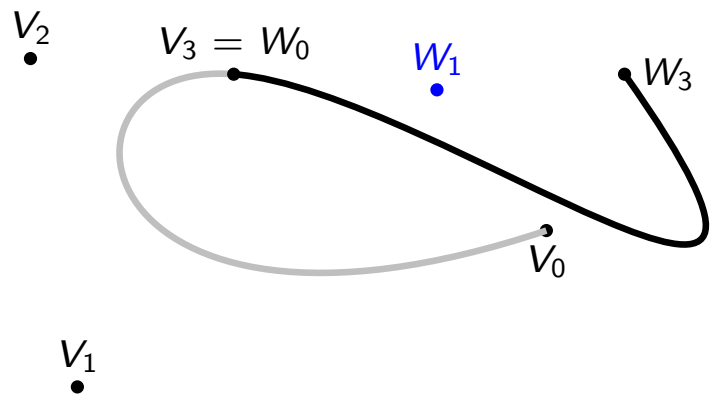
Stetig ersten Grades (C^1)

Bezier-Kurven $\Rightarrow$ Bezier-Splines



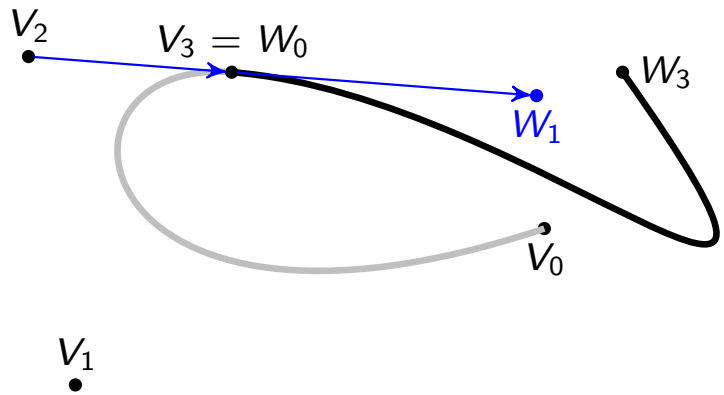
Stetig zweiten Grades (C^2)

Bezier-Kurven $\Rightarrow$ Bezier-Splines



Stetig dritten Grades (C^3)

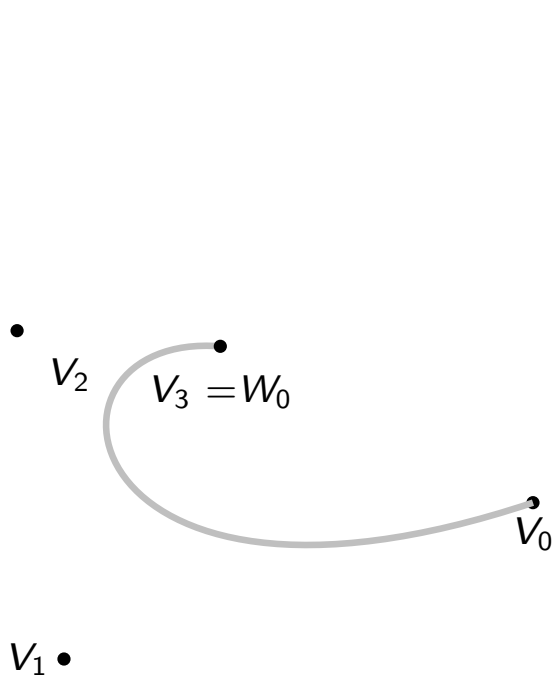
Bezier-Kurven $\Rightarrow$ Bezier-Splines



Geometrisch Stetige (G^2)

W_2

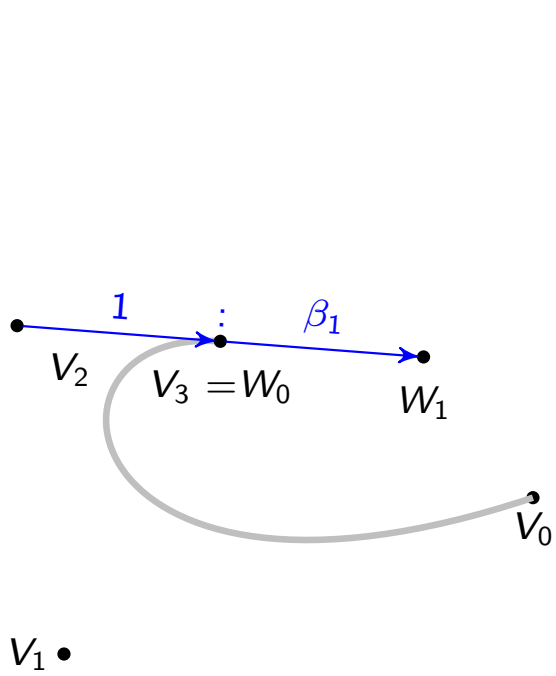
β-Bezier-Kurven: Farin-Böhm Konstruktion



Freie Variablen:

- $V_0,\ V_1,\ V_2,\ V_3,\ W_3$

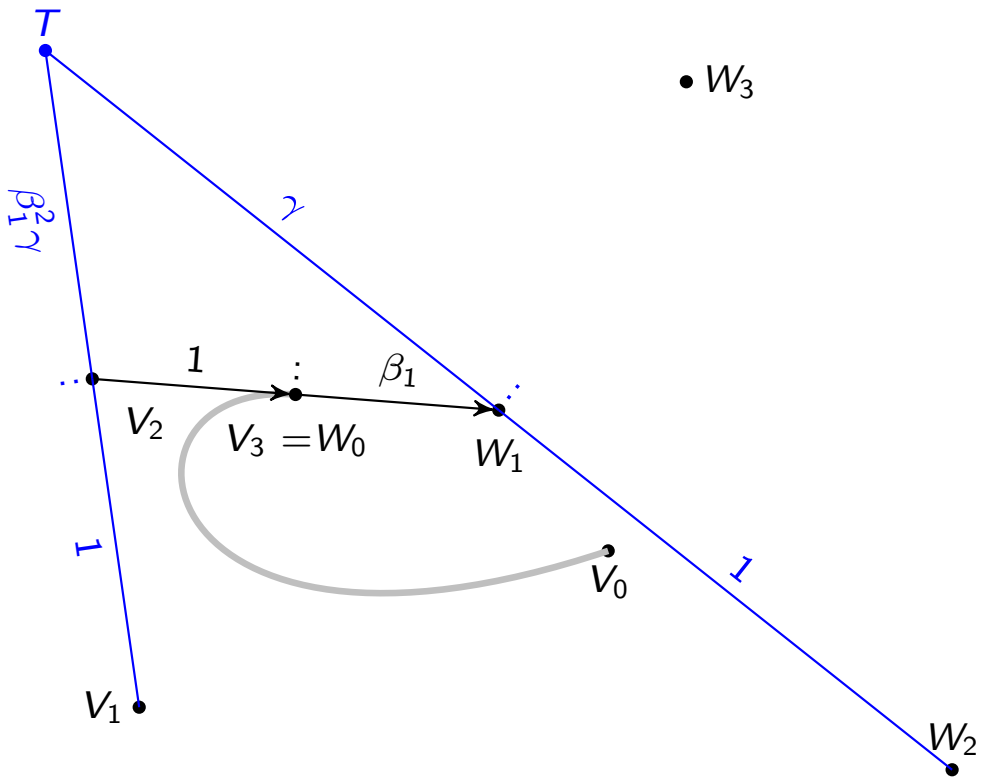
β-Bezier-Kurven: Farin-Böhm Konstruktion



Freie Variablen:

- $V_0,\ V_1,\ V_2,\ V_3,\ W_3$
- $\beta_1 \in \mathbb{R}^{>0}$

β-Bezier-Kurven: Farin-Böhm Konstruktion

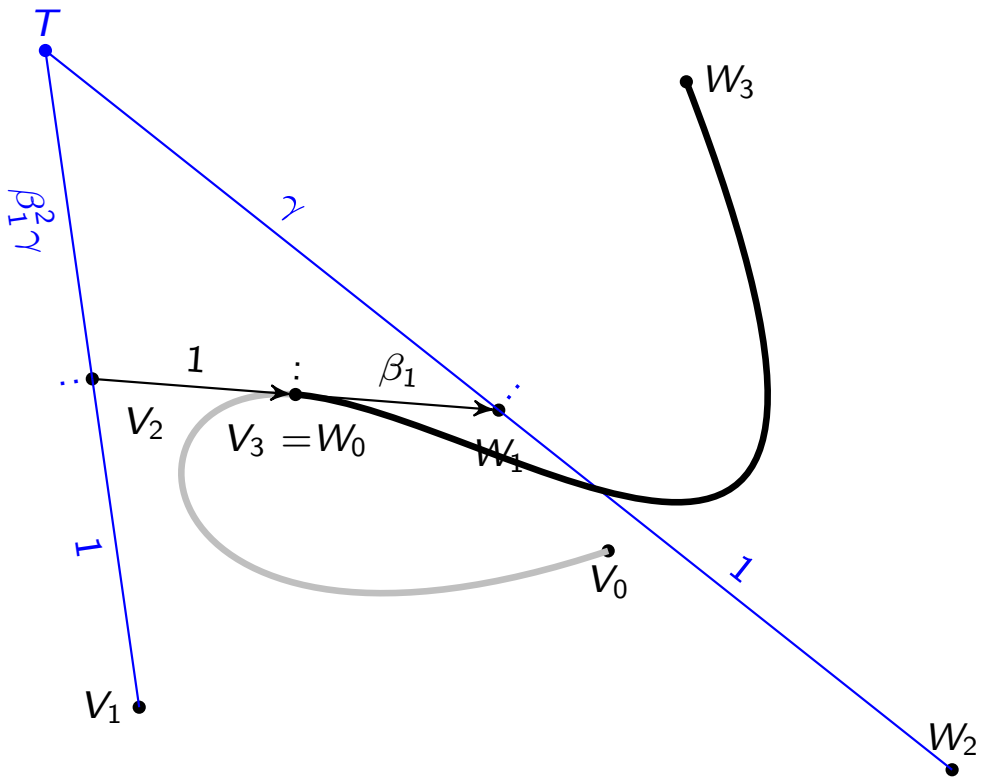


Freie Variablen:

- $V_0,\ V_1,\ V_2,\ V_3,\ W_3$
- $\beta_1 \in \mathbb{R}^{>0}$
- $\beta_2 \in \mathbb{R}$

$$\gamma = \frac{2(1+\beta_1)}{\beta_2 + 2\beta_1(1+\beta_1)}$$

β-Bezier-Kurven: Farin-Böhm Konstruktion

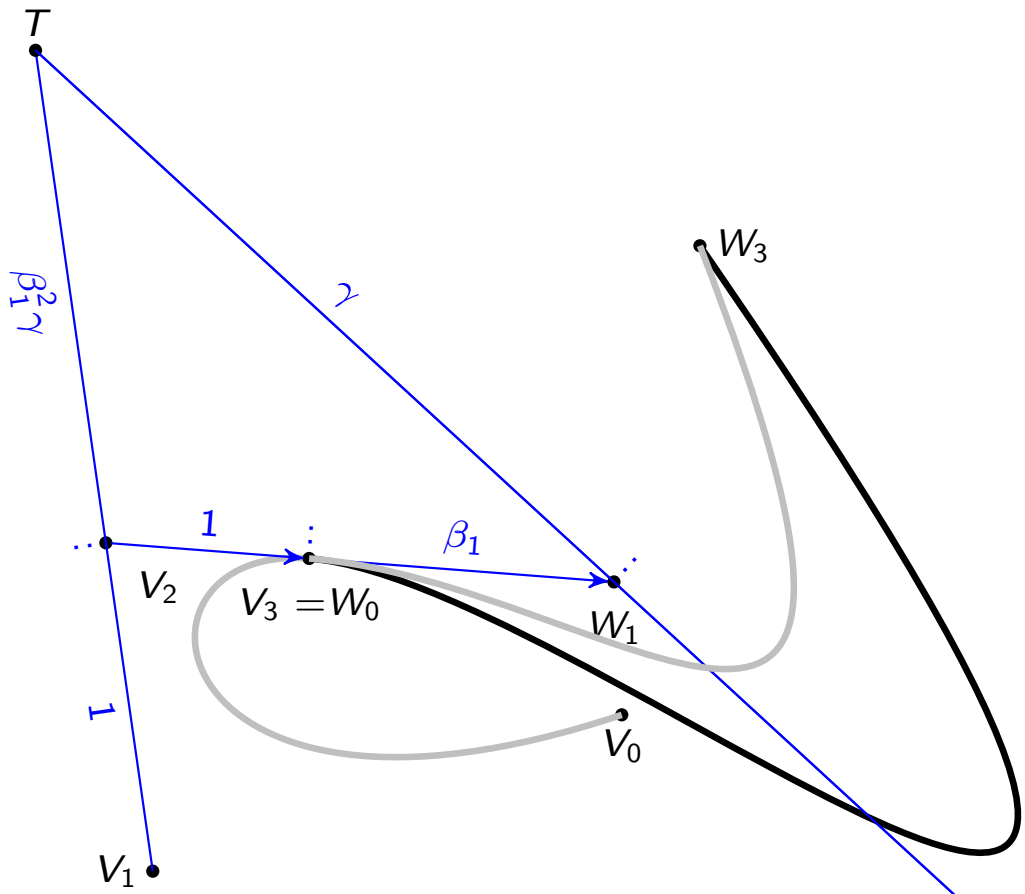


Freie Variablen:

- $V_0,\ V_1,\ V_2,\ V_3,\ W_3$
- $\beta_1 \in \mathbb{R}^{>0}$
- $\beta_2 \in \mathbb{R}$

$$\gamma = \frac{2(1+\beta_1)}{\beta_2 + 2\beta_1(1+\beta_1)}$$

β-Bezier-Kurven: Farin-Böhm Konstruktion



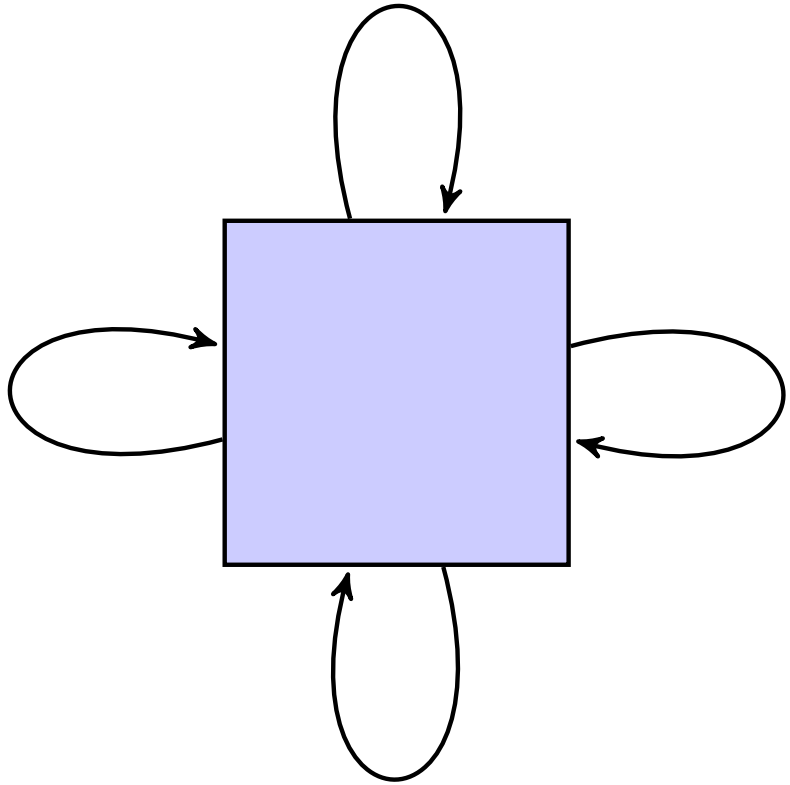
Freie Variablen:

- V_0, V_1, V_2, V_3, W_3
- $\beta_1 \in \mathbb{R}^{>0}$
- $\beta_2 \in \mathbb{R}$

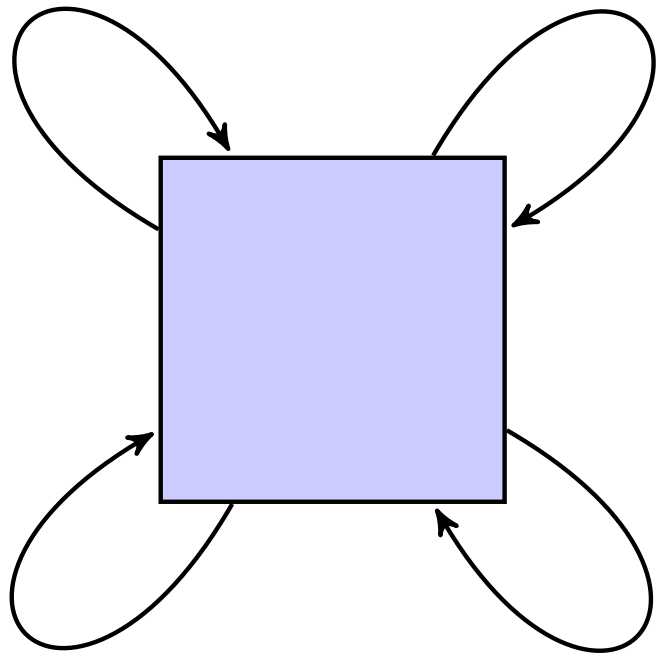
$$\gamma = \frac{2(1+\beta_1)}{\beta_2 + 2\beta_1(1+\beta_1)}$$

Self-Loops

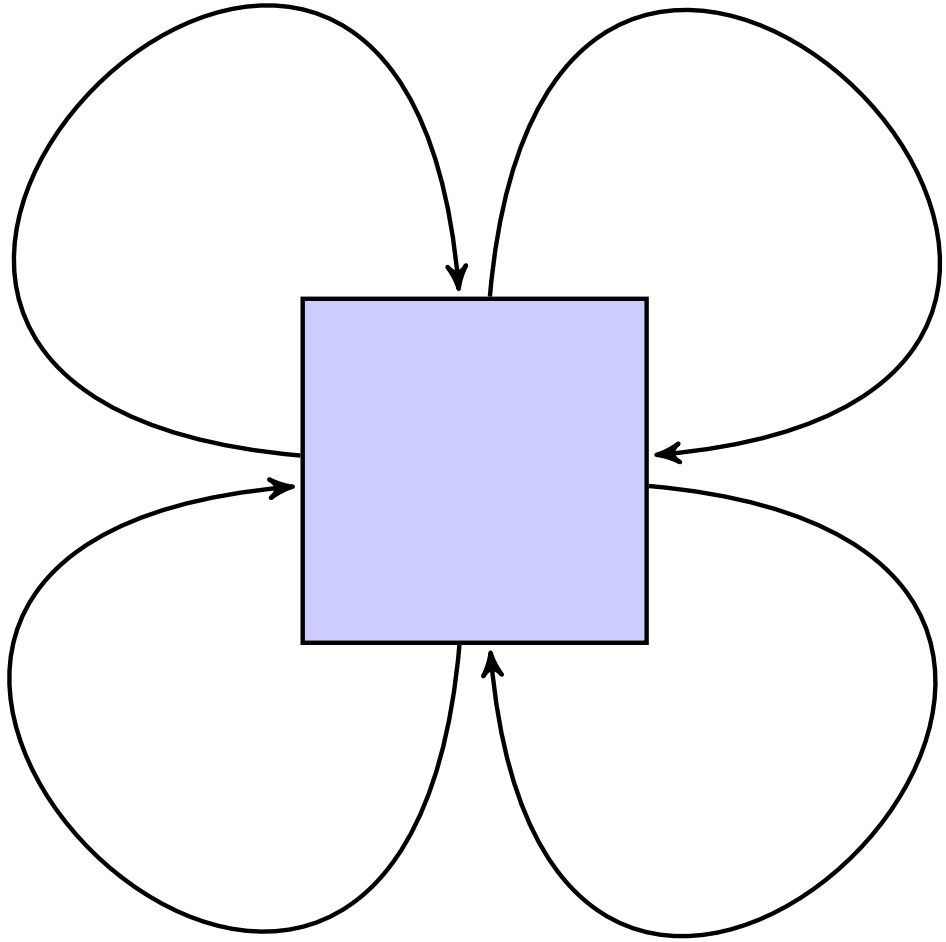
Self-Loops



Self-Loops

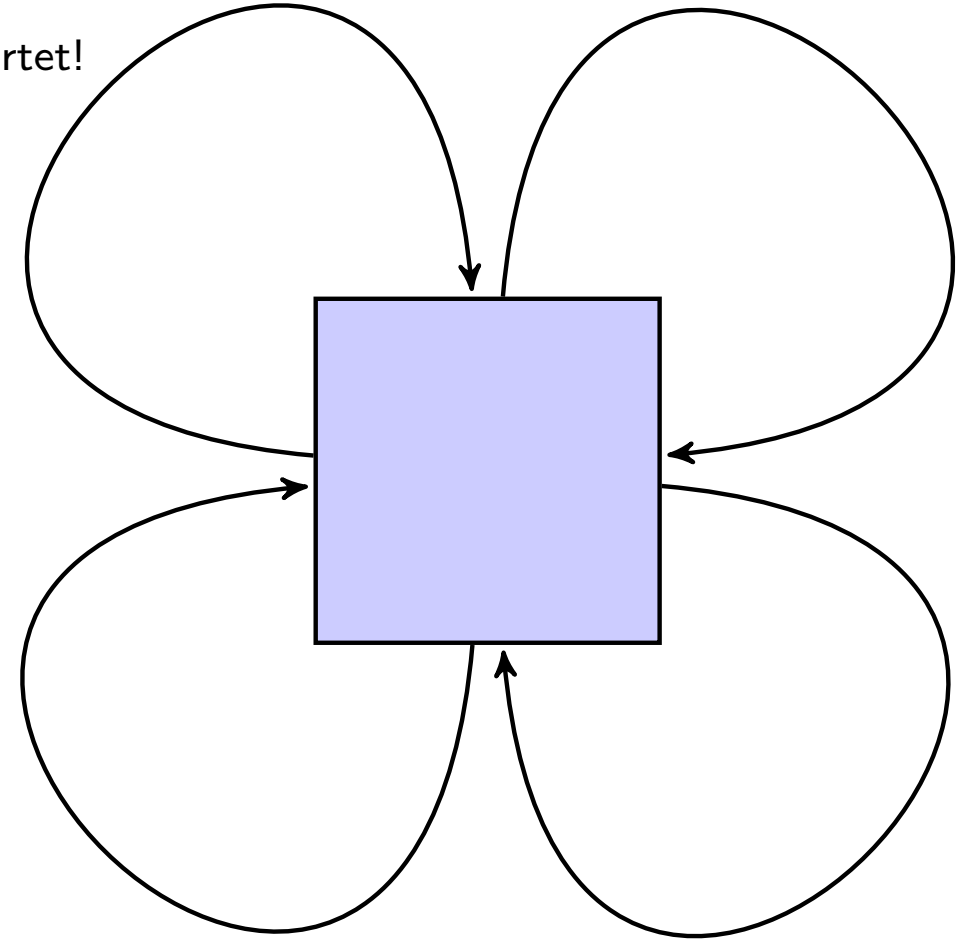


Self-Loops



Self-Loops

Besser als erwartet!



Jetzt neu: NURBS!

NURBS

Non-Uniform Rational B-Splines

$$Bezier(t) = \sum_{i=1}^{k} B_{i,k}(t)\mathbf{P_i}$$

NURBS

Non-Uniform Rational B-Splines

$$Bezier(t) = \sum_{i=1}^{k} B_{i,k}(t)\mathbf{P_i}$$

$$NURBS(t) = \frac{\sum_{i=1}^{k} N_{i,k}(t) w_i \mathbf{P_i}}{\sum_{i=1}^{k} N_{i,k}(t) w_i}$$

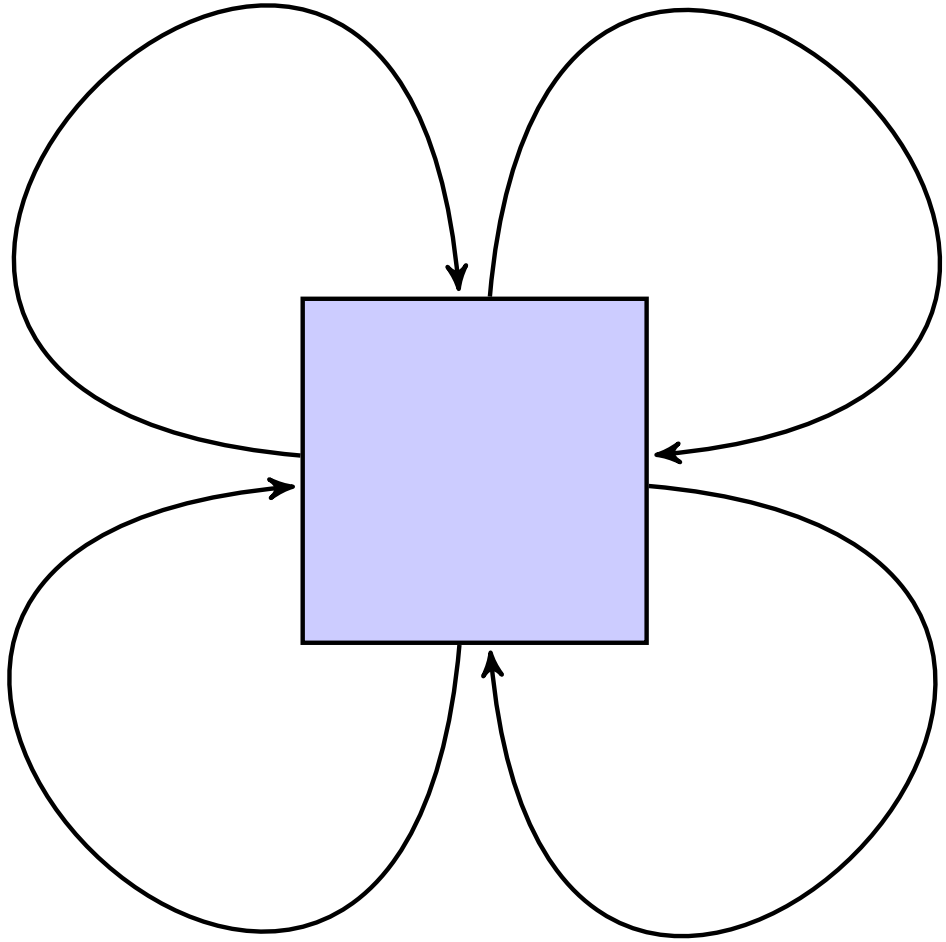
NURBS

Non-Uniform ~~Rational~~ B-Splines

$$Bezier(t) = \sum_{i=1}^{k} B_{i,k}(t)\mathbf{P_i}$$

$$NUBS(t) = \sum_{i=1}^{k} N_{i,k}(t)\mathbf{P_i}$$

Self-Loops mit NUBS



Fazit

- Splines gut geeignet

Fazit

- Splines gut geeignet
- Lokale Kontrolle wichtig

Fazit

- Splines gut geeignet
- Lokale Kontrolle wichtig
- NURBS die Technologie der Wahl

Fazit

- Splines gut geeignet
- Lokale Kontrolle wichtig
- NURBS die Technologie der Wahl
- In linearer Zeit in Bezier umrechenbar

Fazit

- Splines gut geeignet
- Lokale Kontrolle wichtig
- ~~NURBS~~ die Technologie der Wahl
- In linearer Zeit in Bezier umrechenbar
- Man muss loslassen können