

# LaTeX, beamer, tikz und Co.

## 13. Tikz: basics

Thomas Worsch

Fakultät für Informatik  
Karlsruher Institut für Technologie

Wintersemester 2017/2018

Einleitung

Grundlagen

Punkte

Einschub: pgf for

Pfadaktionen

Pfadoperationen

- move to

- line to

- cycle

- rectangle

- circle

- horizontal/vertical

- arc

- curve to

- to

# Überblick

Einleitung

Grundlagen

Punkte

Einschub: pgffor

Pfadaktionen

Pfadoperationen

## TikZ: Was ist das?

- ▶ `tikz` ist ein Paket, das auf `pgf` aufbaut
- ▶ `pgf` ist ein Paket
  - ▶ zur Erzeugung grafischer Darstellungen
  - ▶ unabhängig vom Ausgabeformat
  - ▶ Warnung: ich arbeite nur mit `pdflatex`
- ▶ Dokumentation *sehr* umfangreich: 1150 Seiten
- ▶ auf `pgf` aufbauend gibt es z. B. `pgfplots`

# Überblick

Einleitung

**Grundlagen**

Punkte

Einschub: `pgf for`

Pfadaktionen

Pfadoperationen

## Benutzung

in der Präambel

- ▶ `\usepackage{tikz}`
- ▶ sowie eventuell Zeilen der Form  
`\usetikzlibrary{<lib>}`

Beispiel für `\usetikzlibrary`:

- ▶ `arrows`
- ▶ `matrix`
- ▶ `backgrounds, shadings`
- ▶ `fit, calc`
- ▶ `intersections`
- ▶ `automata, petri, circuits.logic`
- ▶ `graphs`
- ▶ `decorations.text, ...`
- ▶ `shapes.multipart, ...`

## Bilder

- ▶ zwei Möglichkeiten, ein tikz-Bild zu erzeugen:

```
\begin{tikzpicture}  
  <Pfad 1>;  
  <Pfad 2>;  
  ⋮  
  <Pfad n>;  
\end{tikzpicture}
```

```
\tikz{  
  <Pfad 1>;  
  <Pfad 2>;  
  ⋮  
  <Pfad n>;  
}
```

## Pfade, Pfadaktionen, Pfadoperationen

- ▶ **Bild**: eine Folge von  $\langle \text{Pfad} \rangle$ en
- ▶ jeder  $\langle \text{Pfad} \rangle$  endet mit einem Semikolon ;
- ▶ **Pfad**:
  - ▶ spezifiziert im allgemeinen eine **Pfadaktion**
  - ▶ wird konstruiert durch eine Folge von **Pfadoperationen**
- ▶ Pfadoperationen beinhalten oft die Angabe von **Punkten** in einem (der vielen) **Koordinatensysteme**
  
- ▶ Beispielpfad:  
`\path[draw] (1cm,0cm) -- (2cm,1cm) ;`
  - ▶ Pfadaktion: `draw`
  - ▶ Pfadoperation 1: `(1cm,0cm)` „move-to“-Operation
  - ▶ Pfadoperation 2: `-- (2cm,1cm)` „line-to“-Operation



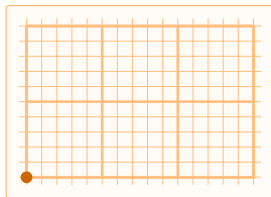
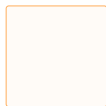
## fehlende Pfadaktion

ohne Pfadaktion wird nur der Platz für den Pfad bereit gestellt:

---

```
\begin{tikzpicture}  
  \path (1cm,0cm) -- (2cm,1cm) ;  
\end{tikzpicture}
```

---



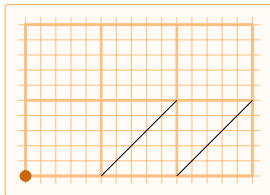
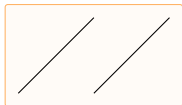
## Pfadaktion draw

`\path[draw]` oder kurz `\draw` zeichnet den Pfad

---

```
\begin{tikzpicture}
  \path[draw] (1cm,0cm) -- (2cm,1cm) ;
  \draw (2cm,0cm) -- (3cm,1cm) ;
\end{tikzpicture}
```

---

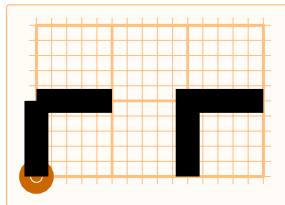


## Mehrere Pfadoperationen hintereinander

---

```
\begin{tikzpicture}[line width=9pt]
  \draw (0cm,0cm) -- (0cm,1cm) (0cm,1cm) -- (1cm,1cm);
  \draw[xshift=2cm] (0cm,0cm) -- (0cm,1cm) -- (1cm,1cm);
\end{tikzpicture}
```

---



# Überblick

Einleitung

Grundlagen

**Punkte**

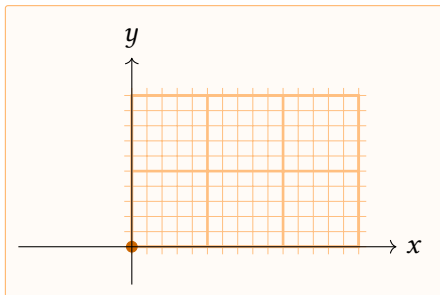
Einschub: `pgf for`

Pfadaktionen

Pfadoperationen

## Das *canvas coordinate system*

- ▶ zweidimensionale Fläche
  - ▶ positive  $x$ -Achse nach rechts
  - ▶ positive  $y$ -Achse nach oben

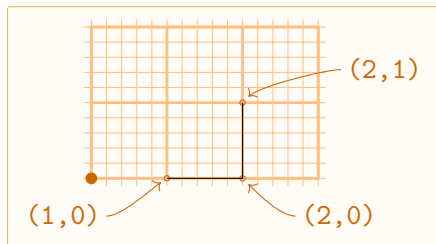


- ▶ Punkt spezifiziert durch ( $\langle x \text{ Distanz} \rangle$  ,  $\langle y \text{ Distanz} \rangle$ )
- ▶ alle „gängigen“  $\text{\LaTeX}$ -Einheiten (`cm`, `mm`, `pt`, `ex`, `em`, ...)

## Das xyz coordinate system

- ▶ Angabe von 2 oder 3 Zahlen *ohne* Längeneinheiten:  
 $(\langle x \rangle, \langle y \rangle)$       oder       $(\langle x \rangle, \langle y \rangle, \langle z \rangle)$
- ▶ fehlende z-Angabe wie 0 interpretiert

```
\tikz{  
  \draw (1,0) -- (2,0) -- (2,1) ;  
}
```

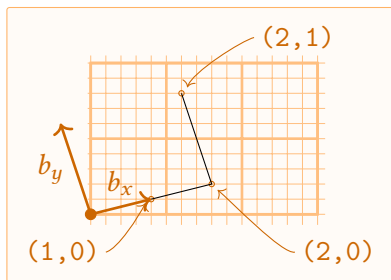
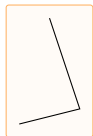


## Das *xyz coordinate system* (2a)

- ▶ zunächst kein Unterschied z. B. zwischen  $(2\text{cm}, 1\text{cm})$  und  $(2, 1)$
- ▶ Basisvektoren für  $x$  und  $y$ 
  - ▶ haben Länge 1 cm und
  - ▶ verlaufen (im Canvas-Koordinatensystem) nach rechts bzw. oben
- ▶ Das kann man aber ändern.

## Das xyz coordinate system (2b)

```
\tikz[x={(8mm,2mm)},y={(-4mm,12mm)}]{  
  \draw (1,0) -- (2,0) -- (2,1) ;  
}
```





## Das *polar coordinate system* (1)

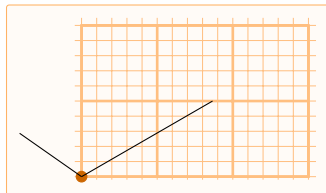
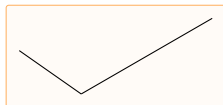
- ▶ Punkt spezifiziert in der Form (  $\langle Winkel \rangle$  :  $\langle Radius \rangle$  )
- ▶  $\langle Radius \rangle$ 
  - ▶ vorgeschrieben: Angabe *mit* Längeneinheit
  - ▶ wenn sie fehlt: ...
- ▶  $\langle Winkel \rangle$ 
  - ▶ bezogen auf die positive  $x$ -Achse im *canvas* Koordinatensystem
  - ▶ entgegen dem Uhrzeigersinn

## Das polar coordinate system: Beispiel

---

```
\tikz[x={(9mm,0mm)},y={(0mm,4mm)}]{  
  \draw (0,0) -- (30:2cm);  
  \draw (0,0) -- (-215:1cm);  
}
```

---



## Relative Koordinaten und der aktuelle Bezugspunkt (1)

- ▶ „**relative** Koordinatenangabe“:  
vor Koordinatenangabe stehen ein oder zwei Pluszeichen
- ▶ Koordinatenangaben beziehen sich
  - ▶ nicht auf den Ursprung des Koordinatensystems
  - ▶ sondern sind relativ zum **aktuellen Bezugspunkt**
    - ▶ im folgenden kurz CRRC  
(für *C*urrent *R*eference point for *R*elative *C*oordinates)
- ▶ zwei Varianten relativer Koordinaten
  - ▶ ++ ( *⟨Punkt⟩* )
  - ▶ + ( *⟨Punkt⟩* )

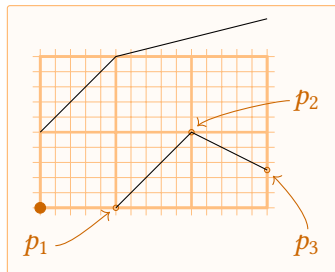
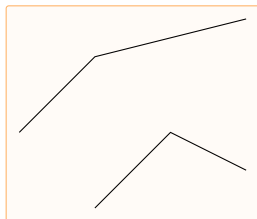
## Relative Koordinaten und der aktuelle Bezugspunkt (2)

Wo ist der CRRC bzw. wie wird ein Punkt zum CRRC? Drei Regeln:

1. Wird ein Punkt über *absolute* Koordinaten spezifiziert, wird er automatisch CRRC.
2. Wird ein Punkt über relative Koordinaten mit *zwei Pluszeichen ++* spezifiziert, wird er automatisch CRRC.
3. Wird ein Punkt über relative Koordinaten mit *einem Pluszeichen +* spezifiziert, wird er *nicht* CRRC, sondern das bleibt der Punkt, der es vorher auch schon war.

## Relative Koordinaten und der aktuelle Bezugspunkt (3)

```
\tikz{  
  \draw (0,1) -- ++(1,1) -- ++(2,0.5)  
        (1,0) -- +(1,1) -- ++(2,0.5) ;  
}
```



# Überblick

Einleitung

Grundlagen

Punkte

**Einschub: pgffor**

Pfadaktionen

Pfadoperationen

## Paket pgffor

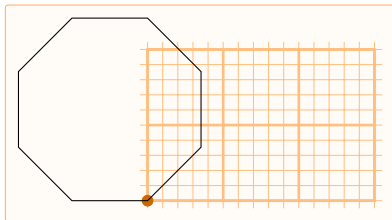
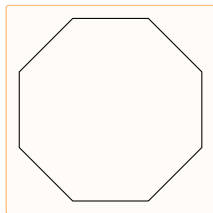
- ▶ wird von `tikz` automatisch mit geladen
- ▶ kann unabhängig von `tikz` verwendet werden:
  - ▶ `\usepackage{pgffor}`

## Einfache Schleifen bequem notiert

---

```
\tikz{\draw (0,0)
  -- ++(45:1) -- ++(90:1) -- ++(135:1) -- ++(180:1)
  -- ++(225:1) -- ++(270:1) -- ++(315:1) -- ++(360:1)
;}
```

---



---

(hier ist noch etwas ganz unschön; siehe [cycle](#) weiter hinten)

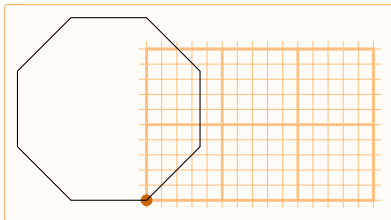
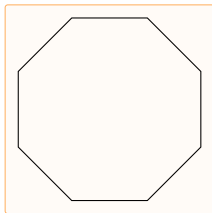


## Einfache Schleifen bequem notiert

---

```
\tikz{\draw (0,0)
  \foreach \d in {45,90,135,180,225,270,315,360}
    { -- ++(\d:1) }
;}
```

---

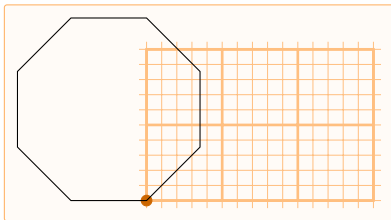
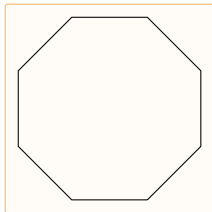


## Einfache Schleifen bequem notiert

---

```
\tikz{\draw (0,0)
  \foreach \d in {45,90,...,360} { -- ++(\d:1) }
;}
```

---

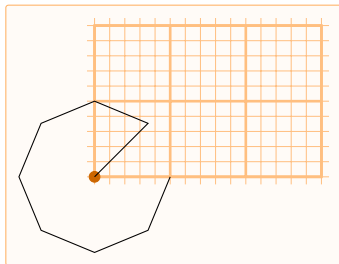
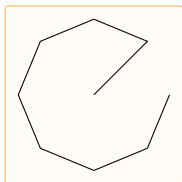


## Einfache Schleifen bequem notiert

---

```
\tikz{\draw (0,0)  
  \foreach \d in {45,90,...,360} { -- +(\d:1) }  
;}
```

---



# Überblick

Einleitung

Grundlagen

Punkte

Einschub: `pgf for`

**Pfadaktionen**

Pfadoperationen

## weitere Pfadaktionen

### Abkürzungen

- ▶ `\fill` für `\path[fill]`
- ▶ `\shade` für `\path[shade]`
- ▶ `\clip` für `\path[clip]`

### Kommandos für Kombinationen

- ▶ `\filldraw` für `\path[fill,draw]`
- ▶ `\shadedraw` für `\path[shade,draw]`

ansonsten nur als Optionen für Pfad:

- ▶ z. B. `\path[clip,draw,fill]`

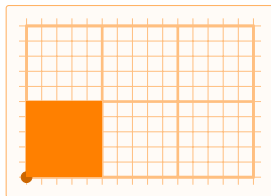
und weitere ...

## Pfadaktion fill

---

```
\tikz{  
  \fill[orange] (0,0) -- (1,0) -- (1,1) -- (0,1) -- cycle;  
}
```

---

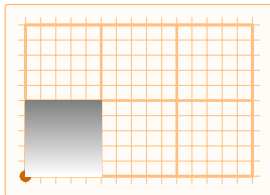


## Pfadaktion shade

---

```
\tikz{  
  \shade (0,0) -- (1,0) -- (1,1) -- (0,1) -- cycle;  
}
```

---



# Überblick

Einleitung

Grundlagen

Punkte

Einschub: pgf for

Pfadaktionen

**Pfadoperationen**

move to

line to

cycle



## Pfadoperation „move-to“

- ▶ Syntax z. B.:  $(\langle \text{Koordinaten} \rangle)$
- ▶ Setzt den CRRC durch **Angabe eines Punktes**.
- ▶ Spezifikation
  - ▶ in einem (der vielen) Koordinatensysteme
    - ▶  $(1\text{cm}, 2\text{cm})$
    - ▶  $(1, 2)$
    - ▶  $(45:10\text{mm})$
  - ▶ Bezug auf einen Knoten oder einen seiner Anker (siehe nächstes Kapitel)
    - ▶  $(\text{node1})$
    - ▶  $(\text{node2.south})$

## Pfadoperation -- (line-to)

- ▶ Syntax:     -- (*⟨Koordinaten⟩*)
- ▶ erzeugt eine gerade Linie
- ▶ vom aktuellen Punkt zum angegebenen Punkt

## Pfadoperation cycle

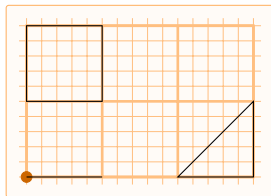
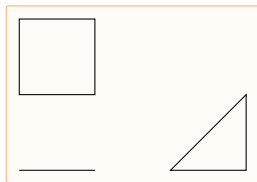
- ▶ Syntax: `-- cycle`
- ▶ erzeugt eine gerade Linie
- ▶ vom letzten Punkt zum Startpunkt des Teilpfads, der mit der letzten move-to Operation begann.
- ▶ Achtung: in verschiedenen Versionen von `tikz` unterschiedliches Verhalten
  - ▶ beim Mischen von relativen Koordinatenangaben und `cycle`
  - ▶ am besten vermeiden

## Beispiele für cycle

---

```
\begin{tikzpicture}
  \draw (0,0) -- (1,0)
        (2,0) -- (3,0) -- (3,1) -- cycle
        (0,1) -- (1,1) -- (1,2) -- +(-1,0) -- cycle;
\end{tikzpicture}
```

---

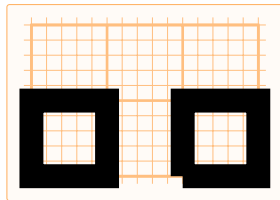


## Beispiele für cycle (2)

---

```
\tikz{ \draw[line width=9pt]
  (0,0) -- (1,0) -- (1,1) -- (0,1) -- cycle
  (2,0) -- (3,0) -- (3,1) -- (2,1) -- (2,0);
}
```

---

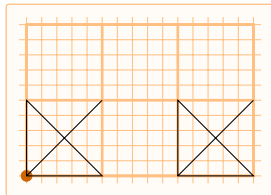
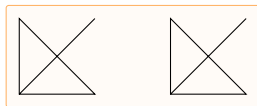


## Beispiele für cycle (3)

---

```
\tikz{ \draw  
  (0,0) -- (1,0) -- (0,1) -- cycle -- (1,1)  
  (2,0) -- (3,0) -- (2,1) -- cycle -- ++(1,1);  
}
```

---



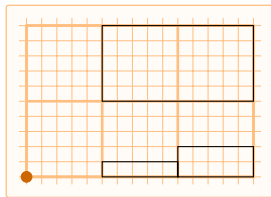
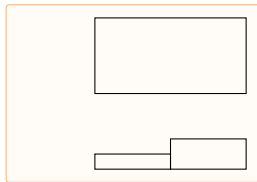
## Pfadoperation rectangle

- ▶ Syntax: `rectangle (<Punkt>)`
- ▶ erzeugt ein Rechteck
  - ▶ eine Ecke: der letzte Punkt
  - ▶ gegenüber liegende Ecke: `<Punkt>`

---

```
\begin{tikzpicture}  
  \draw (0,0) +(1,1) rectangle (3,2);  
  \draw (1,0.2) rectangle ++(1,-0.2) rectangle ++(1,0.4);  
\end{tikzpicture}
```

---



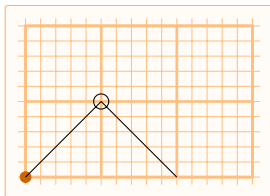
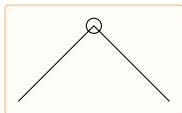
## Pfadoperation circle

- ▶ Syntax: `circle[radius= $\langle Radius \rangle$ ]`
- ▶ zeichnet einen Kreis um den letzten Punkt
- ▶ ändert den letzten Punkt nicht

---

```
\begin{tikzpicture}  
  \draw (0,0) -- (1,1) circle[radius=1mm] -- (2,0);  
\end{tikzpicture}
```

---





## Pfadoperationen | - und - |

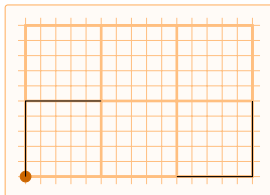
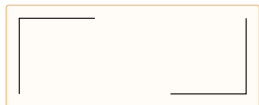
- ▶ Syntax: | - (  $\langle \text{Punkt} \rangle$  )
- ▶ verbindet den letzten Punkt  $p$  mit  $\langle \text{Punkt} \rangle$  durch zwei Geradenstücke:
  - ▶ ausgehend von  $p$  erst ein vertikales Stück
  - ▶ danach ein horizontales
- ▶ - | (  $\langle \text{Punkt} \rangle$  )
- ▶ verbindet den letzten Punkt  $p$  mit  $\langle \text{Punkt} \rangle$  durch zwei Geradenstücke:
  - ▶ ausgehend von  $p$  erst ein horizontales Stück
  - ▶ danach ein vertikales

## Pfadoperationen | - und - |: Beispiel

---

```
\begin{tikzpicture}  
  \draw (0,0) |- +(1,1) (2,0) -| +(1,1);  
\end{tikzpicture}
```

---



## Pfadoperation arc

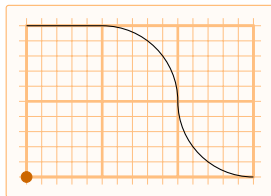
- ▶ zeichnet Teil eines Ellipsen- oder Kreisbogens
- ▶ `arc` [`⟨Optionen⟩`]
  - ▶ `radius=⟨r⟩` oder `x radius=⟨rx⟩, y radius=⟨ry⟩`
  - ▶ zwei der drei Optionen  
`start angle=⟨α⟩, end angle=⟨β⟩, delta angle=⟨δ⟩`  
Zusammenhang:  $\alpha + \delta = \beta$

## Pfadoperation arc: Beispiel

---

```
\begin{tikzpicture}
  \draw (0,2) -- (1,2)
        arc[radius=1, start angle= 90, end angle=0]
        arc[radius=1, start angle=180, delta angle=90] ;
\end{tikzpicture}
```

---



## Pfadoperation „curve-to“

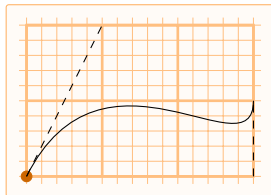
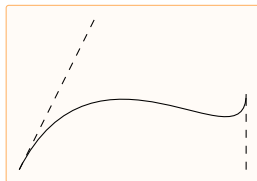
- ▶ Syntax: `.. controls <c> and <d> .. <Punkt>`
- ▶ erzeugt eine Linie, die kubischer Spline ist
- ▶ Syntax: `.. controls <c> .. <Punkt>`
  - ▶ Kurzform für `.. controls <c> and <c> .. <Punkt>`

## Pfadoperation „curve-to“: Beispiel

---

```
\begin{tikzpicture}
  \draw[dashed] (0,0) -- (1,2) (3,0) -- (3,1);
  \draw (0,0) .. controls (1,2) and (3,0) .. (3,1);
\end{tikzpicture}
```

---



---

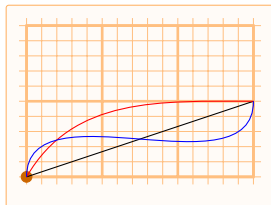
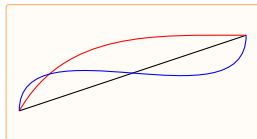
siehe auch `bezier.pdf`

## Pfadoperation to

- ▶ Syntax: `to`  $\langle$ *Punkt* $\rangle$
- ▶ default: zeichnet eine gerade Linie wie `--`  $\langle$ *Punkt* $\rangle$
- ▶ optional:
  - ▶ Winkel am Start- und Zielpunkt
  - ▶ Knoten als Label
  - ▶ Syntax: `to [  $\langle$ Optionen $\rangle$  ]  $\langle$ opt. Knoten $\rangle$   $\langle$ Punkt $\rangle$`

## Pfadoperation to: Beispiel

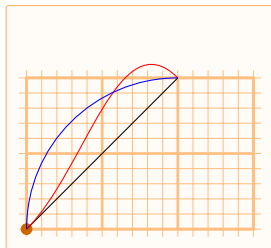
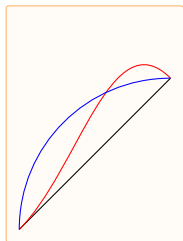
```
\begin{tikzpicture}  
  \draw      (0,0) to (3,1);  
  \draw[red] (0,0) to[out=60,in=180] (3,1);  
  \draw[blue] (0,0) to[out=90,in=270] (3,1);  
\end{tikzpicture}
```





## Pfadoperation to: Beispiel (2)

```
\begin{tikzpicture}  
  \draw      (0,0) to (2,2);  
  \draw[red] (0,0) to[out=45,in=135] (2,2);  
  \draw[blue] (0,0) to[out=45,in=135,relative] (2,2);  
\end{tikzpicture}
```



## Pfadoperation to: Beispiel (3)

---

```
\begin{tikzpicture}
  \draw      (0,0) to      (3,1);
  \draw[red] (0,0) to[relative,out=15,in=180-15] (3,1);
  \draw[blue] (0,0) to[bend right=15] (3,1);
\end{tikzpicture}
```

---

