Realizzare semplici animazioni in figure: come usare Ti*k*Z in Beamer

Claudio Fiandrino

Sommario

L'articolo illustra alcuni metodi per realizzare semplici animazioni per presentazioni in Beamer. In particolare, si focalizzerà l'attenzione su figure realizzate con Ti*k*Z sfruttando la compatibilità esistente fra la classe Beamer e l'omonimo pacchetto grafico.

Abstract

The aim of the article is to present methodologies to create simple animations for Beamer presentations. Specifically, Ti*k*Z-based pictures will be considered thanks to the interoperability between the Beamer class and the package itself.

1 Introduzione

La classe Beamer si pone oggi come prima scelta per la creazione di presentazioni con LaTeX (Tantau *et al.*, 2013). Leggendone la documentazione, si può trovare un gran numero di consigli per strutturare un'ottima presentazione scientifica. L'autore, Till Tantau, ha ideato la classe in vista della sua dissertazione di dottorato e ha raccolto nel corso degli anni una serie di linee guida riassunte nel capitolo *5 Guidelines for Creating Presentations* della documentazione. In merito alle animazioni, tali linee guida sono piuttosto categoriche e ne consigliano un utilizzo limitato e mirato, ad esempio la spiegazione progressiva dei vari passi di un algoritmo o di un sistema.

Quando le figure da realizzare sono semplici, conviene utilizzare direttamente LaTeX che offre numerosi pacchetti grafici. Uno di questi è Ti*k*Z, sviluppato dallo stesso autore della classe Beamer (Tantau, 2013). Ti*k*Z è l'interfaccia di alto livello di PGF (Portable Graphic Format); quest'ultimo viene utilizzato in Beamer per la creazione degli elementi grafici come sidebar e blocchi. Ti*k*Z è sviluppato in modo modulare attraverso le librerie: alcune sono fondamentali in quanto costituiscono il nucleo stesso del pacchetto, altre possono essere utilizzate dall'utente all'occorrenza. Nel secondo caso, infatti, le funzionalità introdotte non influscono sulla operatività del pacchetto. Le librerie ausiliarie devono essere caricate nel preambolo di un documento attraverso `\usetikzlibrary{`⟨*lista delle librerie*⟩`}` dove ⟨*lista delle librerie*⟩ è una lista separata da virgole.

Questo articolo presenta le diverse metodologie che permettono di creare semplici animazioni in figure sfruttando la compatibilità fra la classe Beamer e Ti*k*Z. Si illustreranno esempi molto semplici, con un massimo di due fotogrammi. Il lettore non percepisca questa scelta espositiva come limitazione intrinseca dei metodi presentati: utenti esperti possono anche realizzare animazioni relativamente complesse con gli strumenti che prenderemo in analisi.

È opportuno evidenziare che non si prenderanno in esame altri metodi per realizzare animazioni come quelli forniti dal pacchetto animate. Tale pacchetto crea animazioni basate su Javascript pertanto la soluzione non è supportata da tutti i visualizzatori di PDF. Le animazioni si definiscono attraverso il comando `\animategraphics` o l'ambiente animateinline e possono essere inserite in qualsiasi tipo di documento, non solo in presentazioni.

Il resto dell'articolo è così strutturato:

- la sezione 2 illustrerà brevemente come la classe Beamer gestisce le animazioni;
- la sezione 3 mostrerà in dettaglio gli strumenti standard di Ti*k*Z che "animano" le figure;
- la sezione 4 introdurrà il lettore a strumenti ausiliari, come lo stile `visible on` e il pacchetto aobs-tikz che estende gli strumenti forniti da Ti*k*Z (Fiandrino, 2014);
- la sezione 5 conclude il lavoro.

2 Animazioni in Beamer

Un'ottima introduzione che spiega come realizzare animazioni con Beamer è contenuta nell'articolo di Pignalberi (2010). In questa sezione, tuttavia, si concentra e focalizza l'attenzione sugli aspetti tecnici delle animazioni.

Beamer permette di creare animazioni con numerosi comandi: `\pause`, `\onslide`, `\only`, `\uncover`, `\visible`, `\invisible`, `\alt` e `\temporal`. Ogni comando svolge naturalmente una funzione diversa, sebbene la caratteristica che tutti condividono è quella di determinare in quali fotogrammi appare o non appare un certo contenuto. Ogni quadro (*frame* in inglese) è caratterizzato da uno o più fotogrammi, anche chiamati diapositive (*slide* in inglese). La progressiva esposizione di un certo numero di fotogrammi

in sequenza è un'animazione. In Beamer si può creare e controllare un'animazione agendo su numero (chiamato *overlay specification*) e ordine dei fotogrammi. In termini di codice, ciò si traduce dichiarando il numero del fotogramma come primo argomento nei comandi che *supportano le animazioni*; tale argomento viene delimitato dai caratteri `<` e `>` per evitare confusioni con i classici delimatatori `{`, `}`, `[` e `]`.

Si elencano di seguito alcune regole per definire *overlay specifications*. Al solito, si prendono in esame quelle rilevanti ai fini dell'articolo:

- se le parentesi uncinate contengono un solo numero, `<4>` ad esempio, l'azione del comando si realizza *soltanto* nel quarto fotogramma;
- se il numero è seguito da un "`-`", come `<3->`, l'azione del comando è valida dal terzo fotogramma in avanti;
- se il carattere "`-`" separa due numeri, `<1-3>` ad esempio, l'azione del comando si verifica nel primo, secondo e terzo fotogramma;
- se due o più numeri sono separati da una virgola, come `<1,3,4>`, l'azione del comando ha effetto nel primo, terzo e quarto fotogramma (è conveniente proteggere in un gruppo la dichiarazione, ossia `<{1,3,4}>`);
- è possibile dichiarare espressioni complesse partendo dai costrutti sin qui elencati: `<{1-3,5,7}>` attiva l'azione del comando dal primo al terzo fotogramma e successivamente nel quinto e settimo.

I comandi precedentemente elencati non sono gli unici che *supportano le animazioni*, ma sono i comandi "nativi" di Beamer. Queste *macro base* permettono di definire altre macro in grado di gestire l'ordine di apparizione nei fotogrammi. Ad esempio, celebri comandi LaTeX come `\item` e `\includegraphics` sono stati modificati in modo da poter controllare in quali fotogrammi appariranno le descrizioni degli elenchi o le figure. Ad esempio:

```
\includegraphics<2>{figura}
```

mostra la `figura` solo nel secondo fotogramma. Come è possibile tale magia? Il trucco è molto semplice. Occorre definire il nuovo comando tenendo conto di un argomento aggiuntivo che controlli i fotogrammi, in gergo "rendere il comando *overlay-aware*". In genere si procede così:

```
\newcommand<>{\nuovo}[1]{%
 \comando#2{\funzione{#1}}%
}
```

Attraverso `\newcommand<>` si impone che il `\nuovo` comando in fase di definizione possa controllare in quale fotogramma (argomento `#2`) svolgere la propria `\funzione`. `\comando` è una delle *macro base* riportate nell'elenco iniziale. Vediamo un esempio concreto. Con il seguente codice, si definisce un comando `\rosso` per fare comparire del testo con il colore rosso:

```
\newcommand<>{\rosso}[1]{%
 \only#2{\textcolor{red}{#1}}%
}
```

Successivamente, grazie a `\rosso<2>{ciao}` la parola *ciao* comparirà in rosso solo nel secondo fotogramma del quadro.

3 Il supporto nativo di TikZ

In Ti*k*Z il supporto per il controllo dei fotogrammi è garantito grazie alla seguente definizione:[1]

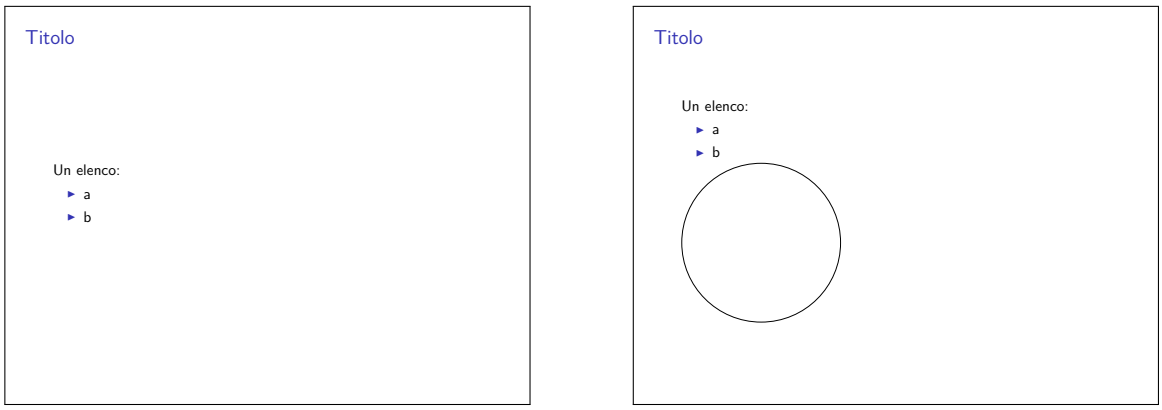
```
\def\tikz@path@overlay#1{%
 \let\tikz@signal@path=\tikz@signal@path%
 % for detection at begin of matrix cell
 \pgfutil@ifnextchar<{%
   \tikz@path@overlayed{#1}}{\path #1}%
}
\def\tikz@path@overlayed#1<#2>{%
 \path<#2> #1%
}
```

Agendo su `\path`, il comando base con cui si possono costruire segmenti e nodi, automaticamente anche `\draw`, `\fill`, `\filldraw`, `\shade`, `\shadedraw` e `\node` ereditano la caratteristica *overlay-aware*.

Prendiamo in esame un esempio pratico. Disegniamo con Ti*k*Z un cerchio visibile solo nel secondo fotogramma del quadro. Con il codice seguente si ottiene il risultato mostrato nelle figure 1a e 1b.

```
\documentclass{beamer}
\usepackage{lmodern}
\usepackage{tikz}
\setbeamertemplate{navigation symbols}{}

\begin{document}

\begin{frame}{Titolo}
Un elenco:
\begin{itemize}
\item a
\item b
\end{itemize}

\begin{tikzpicture}
\draw<2>[radius=2cm] (0,0)circle ;
\end{tikzpicture}
\end{frame}

\end{document}
```

1. Ringrazio Ilhan Polat per avere trovato la definizione nel file omnicomprensivo `tikz.code.tex`.



(a) Primo fotogramma

(b) Secondo fotogramma

Figura 1: Un esempio di animazione con Ti*k*Z

Abbiamo quindi capito che i comandi nativi di Ti*k*Z sono in grado di eseguire la propria funzione, in determinati fotogrammi. Ma Ti*k*Z offre molto di più. La tecnica illustrata nell'esempio precedente è molto semplice e permette soltanto di far comparire degli elementi in determinati fotogrammi. In questo secondo esempio, vediamo come cambiare le proprietà di un elemento sempre visibile. Ti*k*Z permette di definire le proprietà di un elemento attraverso stili (Pantieri e Gordini, 2014): si tratta quindi di far assumere allo stesso stile proprietà diverse in fotogrammi diversi. Tutto ciò è possibile grazie al comando `\only`. Nelle figure 2a e 2b il cerchio presenta, partendo dall'alto verso il basso, una sfumatura dal rosso al bianco e dal bianco al rosso rispettivamente nel primo e secondo fotogramma.

```
\documentclass{beamer}
\usepackage{lmodern}
\usepackage{tikz}
\setbeamertemplate{navigation symbols}{}

\begin{document}
\begin{frame}{Titolo}

\begin{tikzpicture}
\only<1>{%
\tikzset{colora/.style={%
 top color=red,
 bottom color=white}}
}
\only<2>{%
\tikzset{colora/.style={%
 top color=white,
 bottom color=red}}
}
\draw[radius=2cm,colora] (0,0) circle ;
\end{tikzpicture}
\end{frame}

\end{document}
```

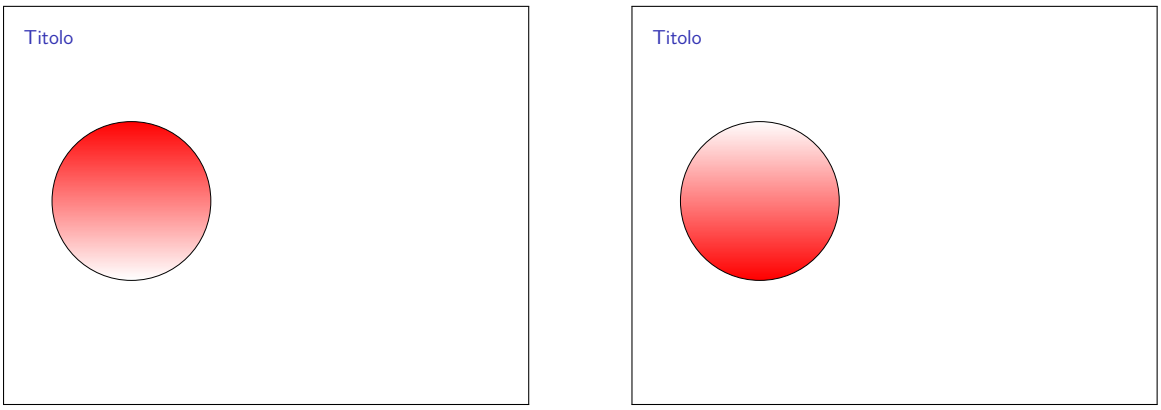
A questo punto, è chiaro come creare animazioni più complesse: combinando le due tecniche si possono far comparire gli elementi del disegno in determinati fotogrammi e alterarne le proprietà (naturalmente quando sono visibili).

4 Altri strumenti

I due metodi presentati nella sezione 3 rappresentano il supporto che intercorre automaticamente fra Ti*k*Z e Beamer. L'utente può contare su queste tecniche in ogni momento. Tuttavia, con l'utilizzo del primo metodo, può insorgere un problema chiamato *jumping effect*. I fotogrammi riportati nelle figure 1a e 1b, se visualizzati in sequenza, ben evidenziano l'inconveniente. Il cerchio, comparendo nel secondo fotogramma, *sposta* la posizione che il testo occupava nella prima diapositiva: l'effetto percepito dal pubblico è quello di un salto, da qui il termine jumping effect. Tale fenomeno è dovuto al meccanismo interno con cui Beamer elabora il contenuto del quadro: ogni fotogramma è considerato indipendente dagli altri. Beamer, non ha nozione che lo stesso elenco è ripetuto e che il cerchio compare soltanto nella seconda diapositiva. Quindi, nel primo fotogramma non si tiene conto dello spazio che il cerchio andrà ad occupare.

Per ovviare a questo inconveniente esistono due alternative. La prima consiste nell'indicare espressamente a Beamer di tenere conto del contenuto di ogni fotogramma. A tal proposito è necessario utilizzare gli ambienti overlayarea e overprint. Il secondo approccio consiste nel disegnare *sempre* il cerchio e renderlo visibile solo negli opportuni fotogrammi. Questo metodo è stato implementato sotto forma di *stile* Ti*k*Z perciò può essere utilizzato direttamente nella figura. In termini di codice, si traduce nell'utilizzare l'opzione `opacity` con valore `0` per mantenere invisibili determiniati elementi della figura. Ecco la definizione dello stile:[2]

2. L'autore ha sviluppato questo stile su TeX.SX: `http://tex.stackexchange.com/q/55806/13304`



(a) Primo fotogramma (b) Secondo fotogramma

FIGURA 2: Animazione con alterazione delle proprietà di un elemento

```
\tikzset{
 invisible/.style={opacity=0},
 visible on/.style={%
  alt={#1{}{invisible}}
 },
 alt/.code args={<#1>#2#3}{%
  \alt<#1>{%
    \pgfkeysalso{#2}
  }{%
    \pgfkeysalso{#3}}
 },
}
```

Cambiando il codice riportato nel precedente documento:

```
\draw<2>[radius=2cm] (0,0)circle ;
```

con:

```
\draw[radius=2cm,
 visible on=<2>] (0,0)circle ;
```

si ottiene il risultato mostrato in figura 3 senza il problema evidenziato.

In maniera del tutto analoga alla prima tecnica spiegata nella sezione 3, lo stile `visible on` permette soltanto di rendere visibili gli elementi della figura in alcuni fotogrammi. Recentemente, su CTAN è apparso un pacchetto che, mantenendo la stessa filosofia dello stile `visible on`, permette di alterare le proprietà di elementi nella figura in diversi fotogrammi (FIANDRINO, 2014). Il pacchetto, aobs-tikz, definisce la libreria `overlay-beamer-styles` con questo scopo. Le proprietà che possono venire alterate riguardano il colore del bordo, del riempimento e del testo, le sfumature e l'aspetto del segmento. Per creare un'animazione è necessario specificare:

- le proprietà di default dell'elemento;
- le modifiche alle proprietà di default;
- i fotogrammi in cui le proprietà modificate vengono utilizzate; automaticamente, negli altri fotogrammi si applicano le proprietà di default.

Nell'esempio seguente illustriamo l'utilizzo della libreria mettendone in luce uno dei suoi punti di forza. L'opzione `double` di Ti*k*Z permette di disegnare un segmento con linee parallele; dietro le quinte, l'opzione attiva una modalità

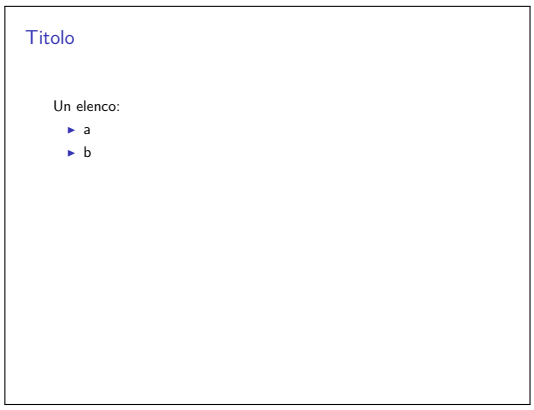
```
\tikz@addmode{\tikz@mode@doubletrue}%
```

che viene applicata globalmente al segmento. Una volta applicata, la modalità non può essere rimossa: in un'animazione, tuttavia, è necessario poterla disabilitare in determinati fotogrammi. aobs-tikz risolve questo problema introducendo l'opzione `double disabled`.

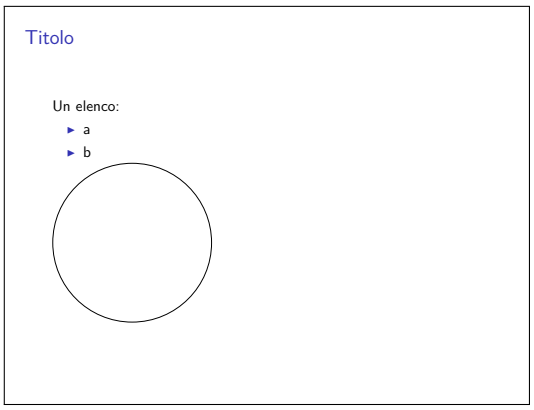
È importante notare che tutte le regole elencate nella sezione 2 per la dichiarazione delle *overlay specifications* sono valide sia per lo stile `visible on` sia per gli stili definiti da aobs-tikz. Con questi metodi, tuttavia, è *obbligatorio* proteggere in un gruppo tutte quelle definizioni che contengono delle virgole. Infatti, senza tale precauzione, il documento non compilerebbe in quanto la virgola è utilizzata dal parser di Ti*k*Z per separare opzioni diverse.

Le figure 4a e 4b illustrano un esempio in cui avviene uno scambio di informazione fra due nodi. Il nodo sorgente sarà contraddistinto dal colore azzurro mentre il nodo destinazione dal colore arancione e dal bordo con opzione `double` attiva. Grazie alla libreria, vediamo come realizzare una semplice animazione in cui nel primo fotogramma il nodo a sinistra è il mittente mentre nel secondo è il destinatario. Inoltre, rendiamo esplicito il senso della comunicazione indicando il verso con una freccia e denotando con un tratteggio la risposta al primo messaggio.

Il codice seguente rappresenta il documento completo dell'animazione in figura 4.



(a) Primo fotogramma



(b) Secondo fotogramma

FIGURA 3: Animazione priva del problema jumping effect

```
\documentclass{beamer}
\usepackage{lmodern}
\usepackage{tikz}
\usetikzlibrary{overlay-beamer-styles}
\setbeamertemplate{navigation symbols}{}

\begin{document}

\begin{frame}{Titolo}
\begin{tikzpicture}[
elemento/.style={%
 circle,
 minimum size=2cm,
 draw,
 background default shade={%
  top color=white,
  bottom color=cyan!80!blue
 },
 background shade={
  top color=white,
  bottom color=orange
 },
 background default aspect={
  double disabled
 },
 background aspect={
  double distance=2pt
 }
},
freccia/.style={
 background default aspect={
  -stealth, solid,
 },
 background aspect={
  stealth-, dashed,
 }
}
]
\node[elemento,
 shade on=<2>,
 aspect on=<2>
 ]
 (A) at (0,0) {};

\node[elemento,
 shade on=<1>,
 aspect on=<1>,
 ]
 (B) at (5,0) {};

\draw[ultra thick,
 freccia,
 aspect on=<2>,
 ] (A)--(B);
\end{tikzpicture}
\end{frame}

\end{document}
```

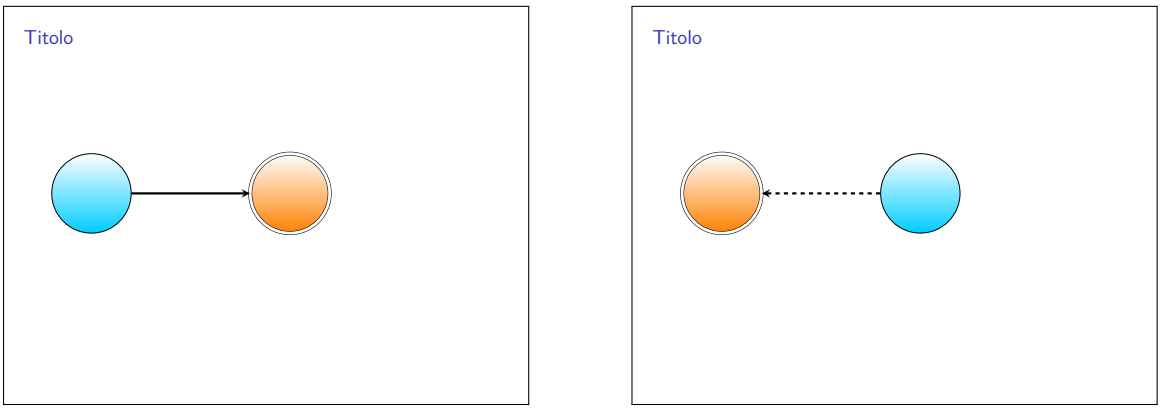
Analizziamolo passo a passo. Il codice principalmente consiste in due fasi:

1. la definizione degli stili `elemento` e `freccia`;
2. l'utilizzo degli stili nella figura.

Lo stile `elemento` è un cerchio con una dimensione prestabilita che viene colorato normalmente in azzurro (`background default shade`) e solo in alcuni fotogrammi in arancione (`background shade`). Inoltre, l'opzione `double` è normalmente disabilitata (`background default aspect`). Pertanto con:

```
\node[elemento,
shade on=<2>,
aspect on=<2>
]
(A) at (0,0) {};
```

stiamo definendo il nodo A come la sorgente poiché solo nel secondo fotogramma (`shade on=<2>` e `aspect on=<2>`) andrà ad assumere il colore arancione e il doppio tratto. Per il nodo B, vale il contrario: infatti le modifiche alle proprietà di default vengono attivate nel primo fotogramma (`shade on=<1>` e `aspect on=<1>`).



(a) Primo fotogramma (b) Secondo fotogramma

FIGURA 4: Animazione con la libreria `overlay-beamer-styles`

In conclusione, prendiamo in analisi lo stile `freccia`. Normalmente, il segmento non è tratteggiato e il verso della freccia punta verso la fine del segmento stesso. Il comportamento modificato, invece, inverte il verso della freccia e rende il segmento tratteggiato. Per questo motivo, la modifica è attivata nel secondo fotogramma con `aspect on=<2>`.

5 Conclusioni

L'articolo ha illustrato in modo semplice come utilizzare TikZ per realizzare animazioni con la classe Beamer sfruttando la compatibilità fra il pacchetto e la classe. Altre tecniche che permettono di realizzare animazioni con Beamer non sono state considerate. Sono stati presi in esame diversi metodi che hanno lo scopo di rendere visibili/invisibili alcuni elementi delle figure e di modificarne le proprietà.

Gli strumenti esistenti non permettono certo di realizzare animazioni mirabolanti. Tuttavia, sono più che sufficienti nel contesto di presentazioni scientifiche, nel corso delle quali le animazioni devono essere di supporto per chi espone e non uno strumento utilizzato per stupire il pubblico.

Riferimenti bibliografici

FIANDRINO, C. (2014). *aobs-TikZ*. URL `http://www.ctan.org/pkg/aobs-tikz`. Consultabile con: `texdoc aobs-tikz`

PANTIERI, L. e GORDINI, T. (2014). *L'arte di disegnare con LaTeX*. URL `http://www.lorenzopantieri.net/LaTeX_files/ArteTikZ.pdf`.

PIGNALBERI, G. (2010). «Presentazioni animate in LaTeX». *ArsTEXnica*, (10), pp. 33–40. URL `http://www.guit.sssup.it/arstexnica/`.

TANTAU, T. (2013). *The TikZ and PGF Packages*. URL `http://www.ctan.org/pkg/pgf`. Consultabile con: `texdoc tikz`

TANTAU, T., WRIGHT, J. e MILETIĆ, V. (2013). *The Beamer Class*. URL `http://www.ctan.org/pkg/beamer`. Consultabile con: `texdoc beamer`

▷ Claudio Fiandrino
`claudio dot fiandrino at gmail dot com`