tikz 制图指南

在 xelatex 指南之上

主要是文档中 *inline* 的制图，大型复杂制图还是考虑外部绘图软件。

万泽[1] | 德山书生[2]

版本：0.01

[1]作者：湖南常德人氏
[2]编者：德山书生，湖南常德人氏。邮箱：a358003542@gmail.com。

前言

TikZ is not an interactive drawing program.

1.Graphics with TikZ Andrew Mertz and William Slough

2.A very minimal introduction to TikZ Jacques Crémer

3.the tikz 官方文档，这个用 texdoc 命令调不出官方文档，用 google 搜索“tikz pdf”吧

目 录

前言 **i**

目录 **ii**

0.1 准备工作 1

1 tikz 基础 **2**

1.1 第一个例子 2

1.1.1 画网格 2

1.1.2 画直线 3

1.1.3 画圆 4

1.1.4 画椭圆 5

1.1.5 点的定义 5

1.1.6 放大图形 6

1.1.7 点的相对偏移 7

1.1.8 画长方形 8

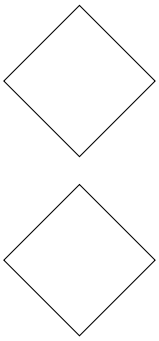
1.1.9 画函数 9

准备工作

tikz 宏包的加载是必须的，因为我这里 myconfig.sty 里面的已经加载了 chemfig 宏包，而 chemfig 宏包又加载了 tikz 宏包，所以不需要修改什么了。

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

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



有两种使用方法，一种命令式的，一种环境式的。命令式用 tikz 命令包围起来，命令式是 inline 模式的。环境式用 tikzpicture 命令包围起来。

这里推荐 KTikZ 软件，在 ubuntu 软件中心里面就有，用这个软件可以边写 tikz 代码边看画的图形，还可以导出图形，早期推荐先使用这个软件练练手熟悉下 tikz 制图代码。值得一提的是这个软件不支持中文注释，会生成乱码。

tikz 基础

第一个例子

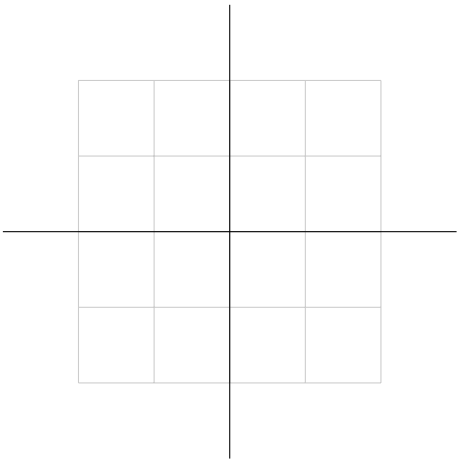
画网格

```
\begin{tikzpicture}
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\end{tikzpicture}
```


step 是网格之间的间距，color 是网格的颜色。第一个坐标点是网格的左底点，第二个坐标点是网格的右顶点。我们可以看到 tikzpicture 下每一条命令最后都要跟一个分号; 。

画直线

```
\begin{tikzpicture}
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\draw (-3,0) -- (3,0);
\draw (0,-3) -- (0,3);
\end{tikzpicture}
```



画直线就是两个坐标点相连，中间 -- 符号表示直线的意思。之前网格是 grid 表示网格的意思。

直线带上箭头

draw 命令可以跟上可选项 **->**，这样直线的右端就有一个箭头了。此外还有：**-»**，**->|**，**-to**，**-latex**，**-stealth**。

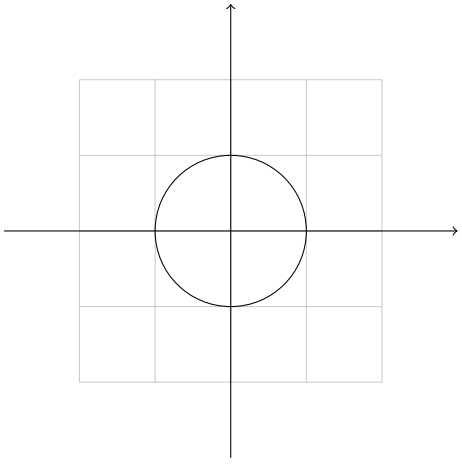
他们的效果从上到下依次演示如下：

类似的还有左端比如 **<-**，或者两端比如 **latex-latex**，这里就不多说了。

画圆

接著上面的图案画一个圆，加入了以下代码：

```
\draw (0,0) circle (1);
```

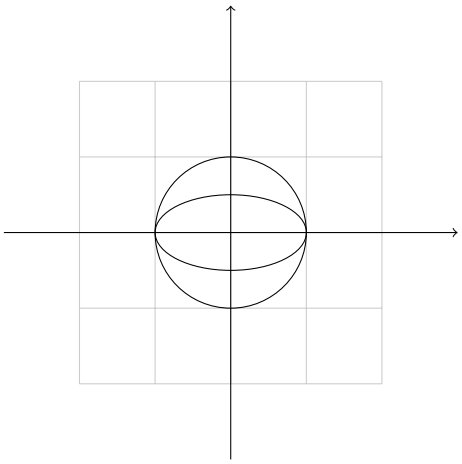


其中第一个点是圆中心，circle 表示画圆，第二个参数是半径大小。

画椭圆

接著画一个椭圆：

```
\draw (0,0) ellipse (1 and 0.5);
```

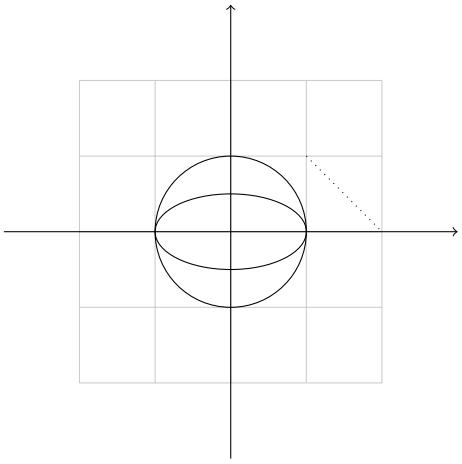


这里第一个点是椭圆的中心点，ellipse 表示画椭圆，后面参数两个值第一个是 a 也就是椭圆的半长，第二个是 b 也就是椭圆的半高。

点的定义

tikz 中定义一个点方便之后使用：

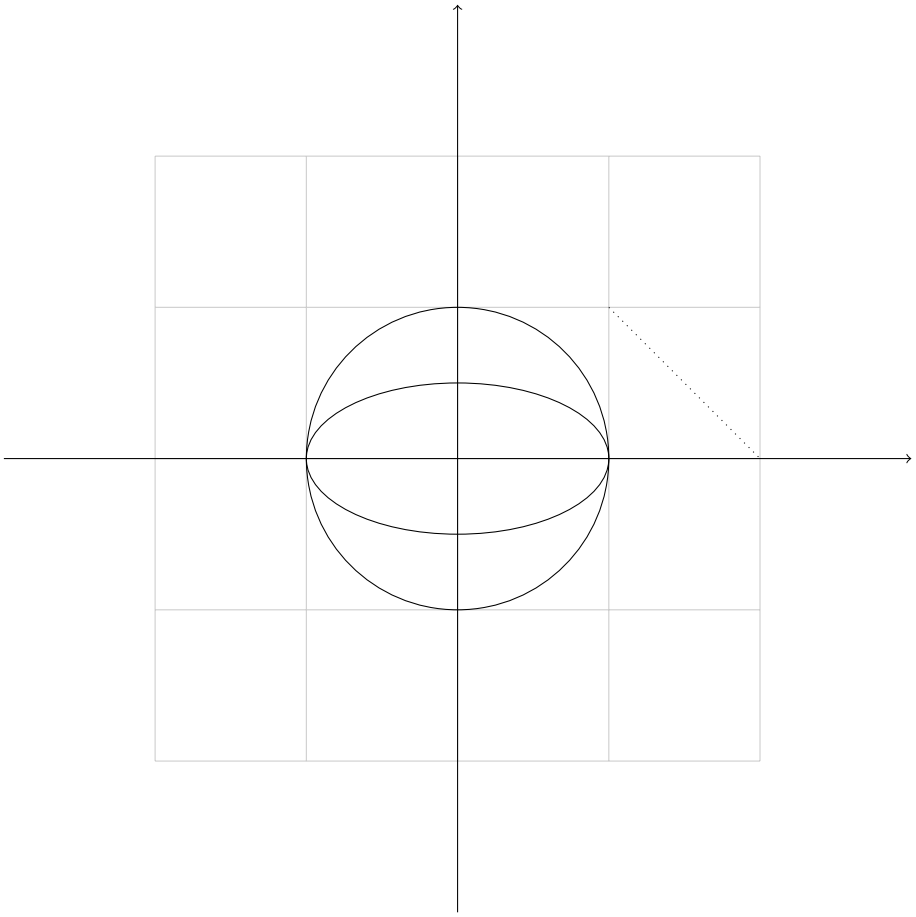
```
\path (1,1) coordinate (point001);
\path (2,0) coordinate (point002);
\draw[dotted] (point001) -- (point002)  ;
```



这里代码的第 6，7 行定义了两个点，名字叫做 point001 和 point002。然后用这两个点作为参数画了一个直线，这个直线有可选项 **dotted**，点线样式。

放大图形

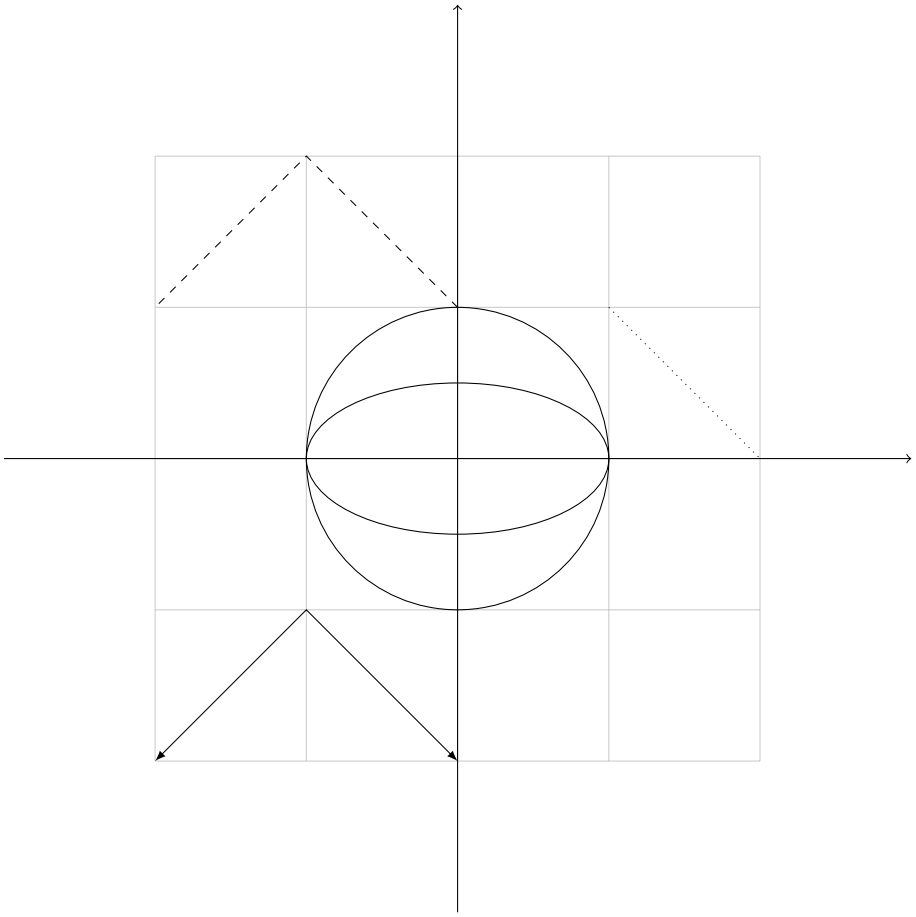
在 tikzpicture 环境后面跟上可选项 [scale=2]，即将图形放大两倍。注意控制别越界了。



点的相对偏移

现在加上这样两行代码：

```
\draw[<->] (0,-2) -- ++(-1,1) -- ++(-1,-1);
\draw[dashed] (0,1) -- +(-1,1) -- +(-2,0);
```

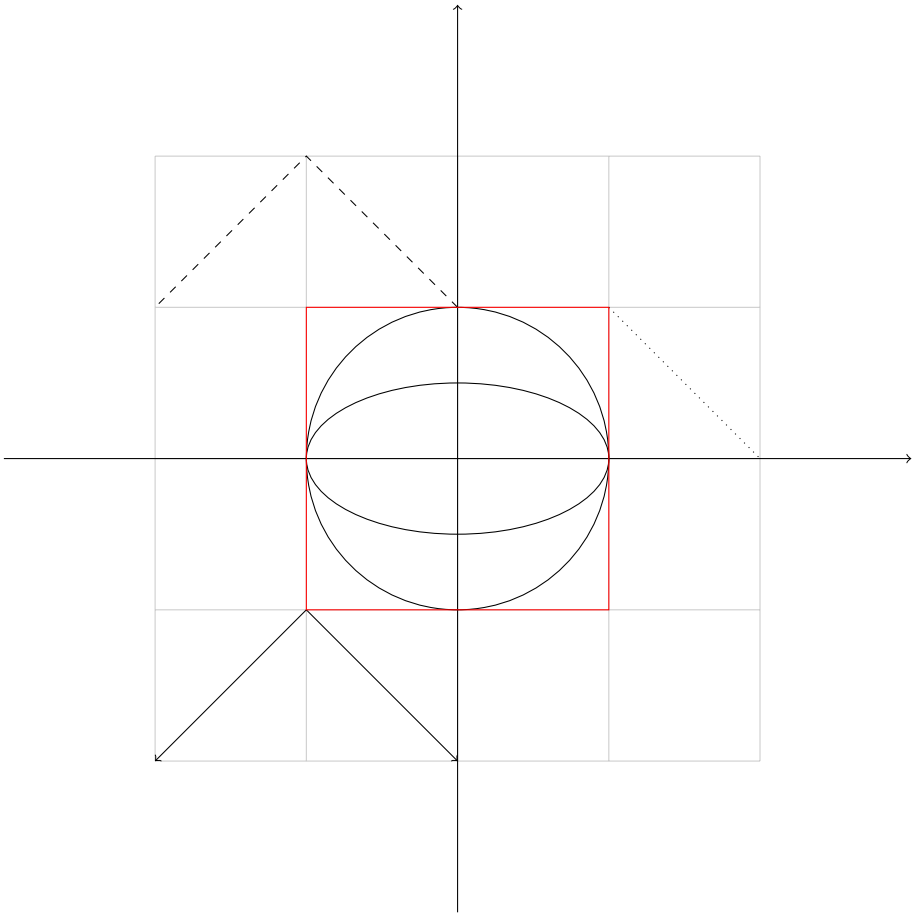


tikz 中有一个重要的概念，当前点，然后点可以通过当前点根据相对偏移来确定一个新的点。上面代码第 9 行的 ++ 符号和第 10 行的 + 符号都根据当前点然后进行了 Δx 和 Δy 的相对偏移从而确定了一个新的点。这两个符号的区别在于是不是更新当前点数据。++ 符号更新当前点，而 + 符号不更新。

画长方形

现在加上这一行代码来画一个长方形：

```
\draw[color=red] (-1,-1) rectangle (1,1);
```



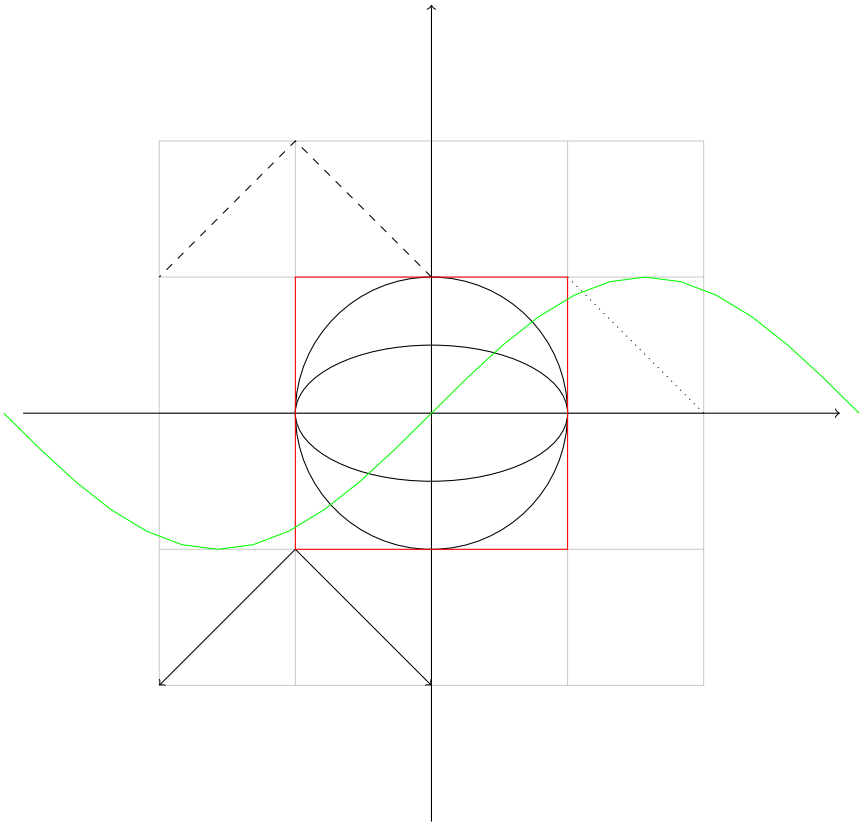
这里使用了可选项 **color=red** 来控制线条的颜色，然后画长方形的第一个点是左底点，rectangle 表示画长方形，第二个点表示右顶点。

画函数

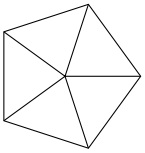
画函数的功能是通过外部程序 gnuplot 来实现了，所以需要打开`--shell-escape`，或者`--enable-write18`

这里最后加了一个语句：

```
\draw[domain=-pi:pi,color=green] plot function{sin(x)};
```



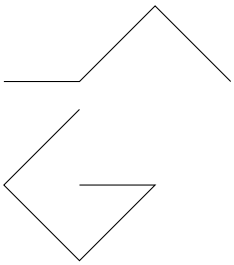
这里可选项 **domain=-pi:pi** 控制画的函数的 x 范围，可以直接用 pi 表示 π，然后接下来 plot function 表示画一个函数，接下来的花括号里面放着 gnuplot 的各种命令，这里就是简单的 $sin(x)$



he concept of the current point plays an important role when multiple actions are involved. For example, suppose two line segments are drawn joining points P and Q along with Q and R

A relative point may be defined by providing offsets in each of the horizontal and vertical directions.

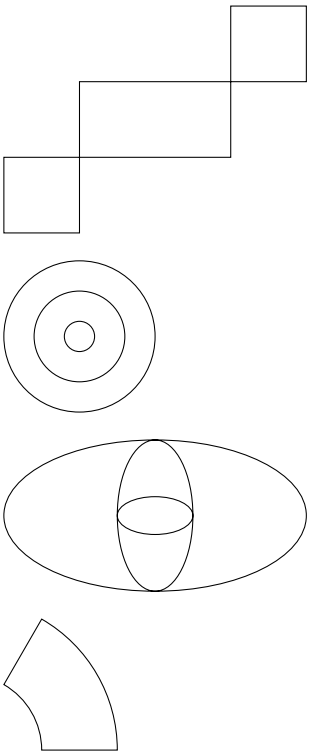
There are two forms of relative points —one which updates the current point and one which does not. The ++ prefix updates the current point while the + prefix does not.

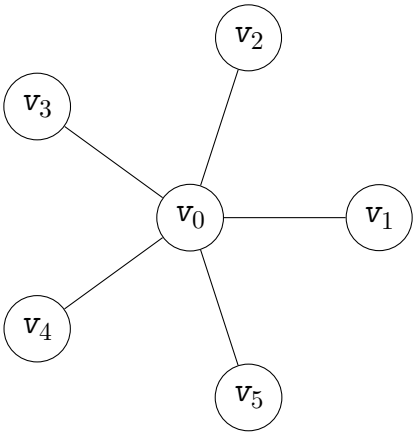


every offset which appears is performed relative to the initial point, (0, 0) .

– Grids and rectangles

– Circles and ellipses – Arcs – Bézier curves




v_2
v_3
v_0
v_1
v_4
v_5