TikZ 几何作图

2023 年 12 月 21 日

目录

第 1 章　基本作图命令

这里的命令都是通过 /tikz/insert path[1] 在当前路径上插入新的路径.

1.1　仿射组合 Affine Combination

调用方式

```
affine={A,B,k}
```

参数说明

A, B 两点坐标

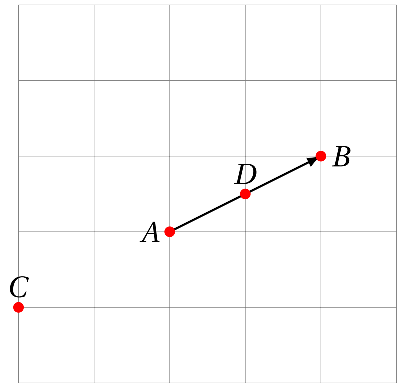
k 系数

返回点 A, B 的仿射组合: $A + k \cdot (B - A)$.

示例

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (5,5);
  \coordinate (A) at (2,2);
  \coordinate (B) at (4,3);
  \coordinate [affine={A,B,-1}] (C);
  \coordinate [affine={A,B,.5}] (D);
  \draw[thick, -latex] (A) -- (B);
  \foreach \p/\placement in {A/left,B/right,
```

```
  C/above,D/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.2 平移 Translate

调用方式

```
translate={A,B,C}
```

参数说明

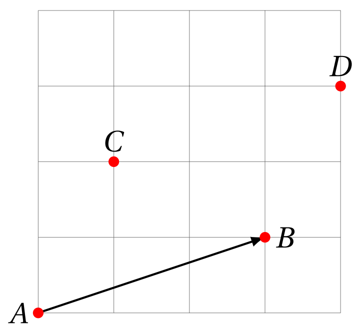
A,B,C 三点坐标

返回 C 按向量 AB 移动所得的坐标: $C+(B-A)$.

示例

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
```

```
  \coordinate (B) at (3,1);
  \coordinate (C) at (1,2);
  \coordinate [translate={A,B,C}] (D);
  \draw[thick, -latex] (A) -- (B);
  \foreach \p/\placement in
   ↪ {A/left,B/right,C/above,D/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.3 对称点 Reflect

调用方式

```
reflect={A,B,C}
```

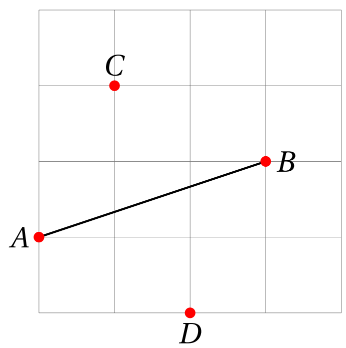
参数说明

A,B,C 三点坐标

返回 C 关于直线 AB 的对称点的坐标 (设 D 为 C 在 AB 的投影): $C+2(D-C)$.

示例

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,1);
  \coordinate (B) at (3,2);
  \coordinate (C) at (1,3);
  \coordinate [reflect={A,B,C}] (D);
  \draw[thick] (A) -- (B);
  \foreach \p/\placement in
   ↪  {A/left,B/right,C/above,D/below}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.4 反演 Inverse

调用方式

```
inverse={O,A,P}
```

参数说明

O 圆心

A 圆上一点

P 平面上任一点

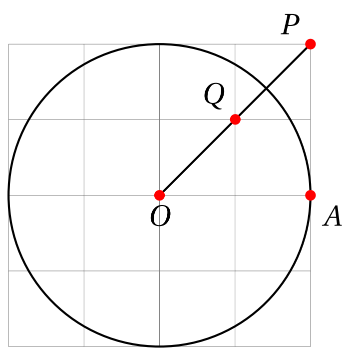
返回 P 关于圆 (O, A) 的反演点.

示例

```
\begin{tikzpicture}
  \draw[help lines] (-2,-2) grid (2,2);
  \coordinate (O) at (0,0);
  \coordinate (A) at +(0:2);  % 圆上一点，相对坐标
  \coordinate (P) at (2,2);
  \coordinate[inverse={O,A,P}] (Q);

  \draw[thick,circle={O,A}];
  \draw[thick] (O) -- (P);

  \foreach \p/\placement in {O/below,A/below right,
  P/above left,Q/above left}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.5 旋转 Revolve

调用方式

```
revolve={A,B}
```

参数说明

A,B 两点坐标

注 为了避免覆盖 tikz 的 rotate, 这里将旋转命令为 revolve.

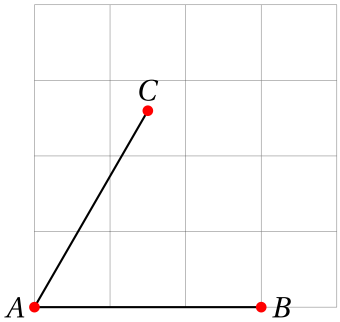
返回 B 绕 A 旋转的点.

还需要指定 revolve/angle (default: 0) 和 revolve/angle scale(default: 1) 两个选项, 可以通过下面的方式来指定 /revolve/angle:

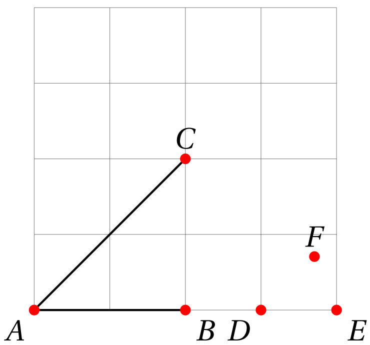
1. 直接指定角度: `revolve/angle=60`
2. 位置向量与 x 轴夹角: `revolve/angle={P1}`
3. 两位置向量的夹角: `revolve/angle={P1,P2}`
4. 由三点定义的角 (P_1 为顶点, P_2 为起点, P_3 为终点): `revolve/angle={P1,P2,P3}`
5. 两向量的夹角 (逆时针方向): `revolve/angle={P1,P2,P3,P4}`

示例

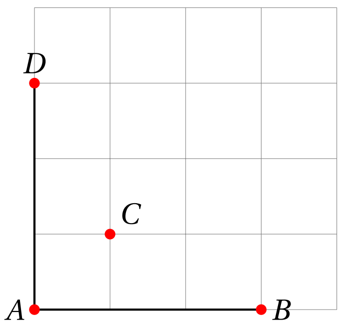
```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
  \coordinate (B) at (3,0);
  \coordinate [revolve/angle=60, revolve={A,B}] (C);
  \draw[thick] (A) -- (B) (A) -- (C);
  \foreach \p/\placement in {A/left,B/right,C/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```




```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
  \coordinate (B) at (2,0);
  \coordinate (C) at (2,2);
  \coordinate (D) at (3,0);
  \coordinate (E) at (4,0);
  \coordinate [revolve/angle={A,B,C},revolve={D,E}] (F);
  \draw[thick] (A) -- (B) (A) -- (C);
  \foreach \p/\placement in {
  A/below left,B/below right,C/above,
  D/below left,E/below right,F/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
  \coordinate (B) at (3,0);
  \coordinate (C) at (1,1);
  \coordinate [revolve/angle={C},
    revolve/scale=2,
    revolve={A,B}] (D);
  \draw[thick] (A) -- (B) (A) -- (D);
  \foreach \p/\placement in {A/left,B/right,
  C/above right,D/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```

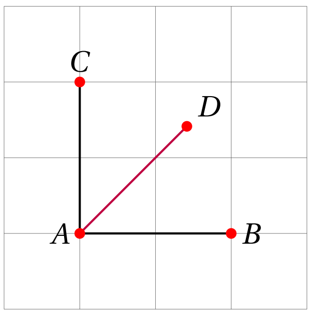


1.6 角平分线 Angle Bisector

可以由 resovle 来获取角平分线上的一点.

示例

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (1,1);
  \coordinate (B) at (3,1);
  \coordinate (C) at (1,3);
  \coordinate [revolve/angle={A,B,C},
    revolve/scale=1/2,
    revolve={A,B}] (D);
  \draw[thick] (A) -- (B) (A) -- (C);
  \draw[thick, purple] (A) -- (D);
  \foreach \p/\placement in {A/left,B/right,C/above,
  D/above right}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```

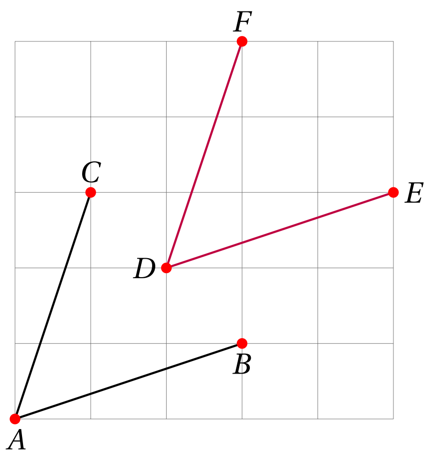


1.7 构造角 Angle

可以由 resovle 来构造一个角.

示例

```
\begin{tikzpicture}[scale=1]
  \draw[help lines] (0,0) grid (5,5);
  \coordinate (A) at (0,0);
  \coordinate (B) at (3,1);
  \coordinate (C) at (1,3);
  \coordinate (D) at (2,2);
  \coordinate (E) at (5,3);
  \coordinate [revolve/angle={A,B,C},
    revolve/scale=1,
    revolve={D,E}] (F);
  \draw[thick] (A) -- (B) (A) -- (C);
  \draw[thick, purple] (D) -- (E) (D) -- (F);
  \foreach \p/\placement in {A/below,B/below,C/above,
  D/left,E/right,F/above}{
    \fill[red] (\p) circle (2pt);
    \node[\placement] at (\p) {$\p$};
  }
\end{tikzpicture}
```



1.8 截取 Intercept

调用方式

```
intercept={A,B}
```

参数说明

A,B 两点坐标

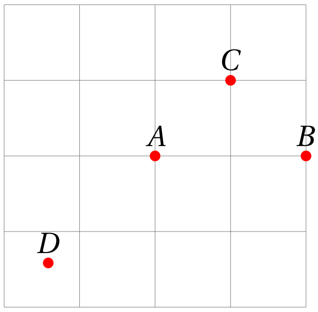
在直线 AB 截取指定长度线段, A 为新线段的起点, AB 是方向.

需要指定 intercept/length (default: 1cm) 和 intercept/scale(default: 1) 两个选项. 其中 intercept/length 有两种形式:

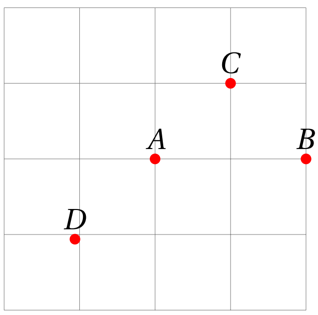
1. 直接指定长度: `intercept/length=2cm`
2. 指定线段长度: `intercept/length={P1,P2}`

示例

```
\begin{tikzpicture}
  \draw[help lines] (-2,-2) grid (2,2);
  \coordinate (A) at (0,0);
  \coordinate (B) at (2,0);
  \coordinate (C) at (1,1);
  \coordinate[intercept/length={A,B},
    intercept/scale=-1, intercept={A,C}] (D);
  \foreach \p/\placement in
   ↪  {A/above,B/above,C/above,D/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```




```
\begin{tikzpicture}
  \draw[help lines] (-2,-2) grid (2,2);
  \coordinate (A) at (0,0);
  \coordinate (B) at (2,0);
  \coordinate (C) at (1,1);
  \coordinate[intercept/length=1.5cm,
    intercept/scale=-1,
    intercept={A,C}] (D);
  \foreach \p/\placement in
   ↪ {A/above,B/above,C/above,D/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.9 直线与直线的交点 Line-Line Intersection

调用方式

```
intersect={A,B,C,D}
```

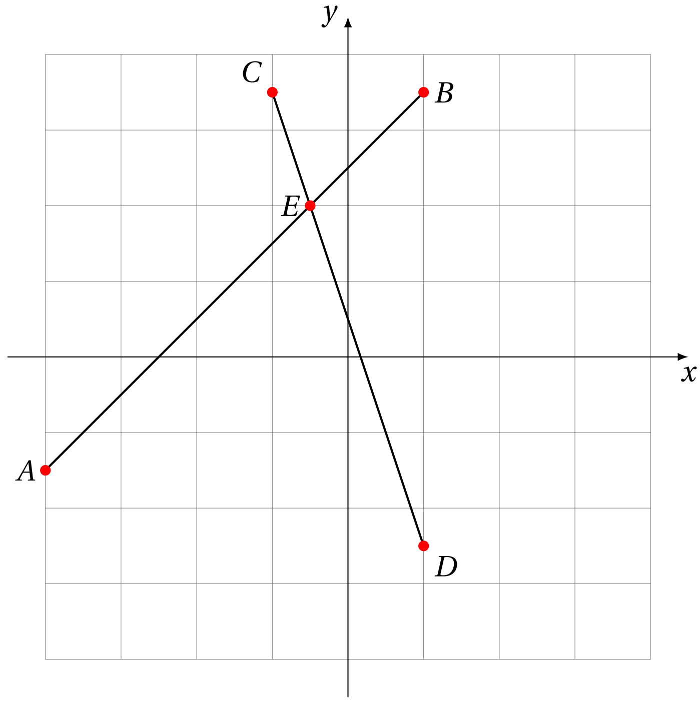
参数说明

A,B,C,D 四点坐标

返回 AB 与 CD 的交点 (可以是延长线相交点).

示例

```
\begin{tikzpicture}
  \draw[help lines] (-4,-4) grid[step=1] (4,4);
  \draw[-latex] (-4.5,0) -- (4.5,0) node[below] {$x$};
  \draw[-latex] (0,-4.5) -- (0,4.5) node[left] {$y$};
  \coordinate (A) at (-4,-1.5);
  \coordinate (B) at (1,3.5);
  \coordinate (C) at (-1,3.5);
  \coordinate (D) at (1,-2.5);
  \coordinate [intersect={A,B,C,D}] (E);
  \draw[thick] (A) -- (B) (C) -- (D);
  \foreach \p/\placement in {A/left,B/right,
  C/above left,D/below right,E/left}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.10 垂直平分线/中垂线 Perpendicular Bisector

调用方式

```
perpendicular bisector={A,B}
```

参数说明

A,B 两点坐标

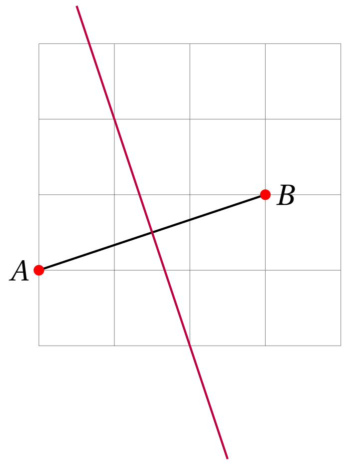
构造 AB 的中垂线, 默认起点为 $.5(A+B)+(B-A)\cdot\mathbf{i}$, 终点为 $.5(A+B)-(B-A)\cdot\mathbf{i}$. 可以对起始点进行调整, 见1.18.

示例

使用默认参数:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
```

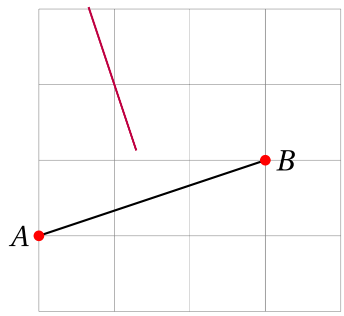
```
  \coordinate (A) at (0,1);
  \coordinate (B) at (3,2);
  \draw[thick] (A) -- (B);
  \draw[thick,purple,perpendicular bisector={A,B}];
  \foreach \p/\placement in {A/left,B/right}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



指定两端的长度:

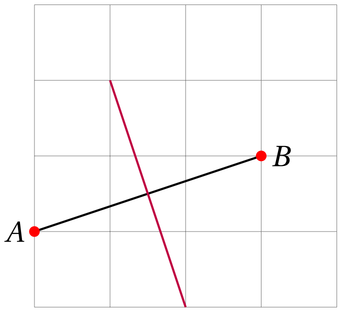
```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,1);
  \coordinate (B) at (3,2);
  \draw[thick] (A) -- (B);
  \draw[thick, purple,
    start modifier=.5cm, end modifier=2.5cm,
    perpendicular bisector={A,B}];
```

```
  \foreach \p/\placement in {A/left,B/right}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



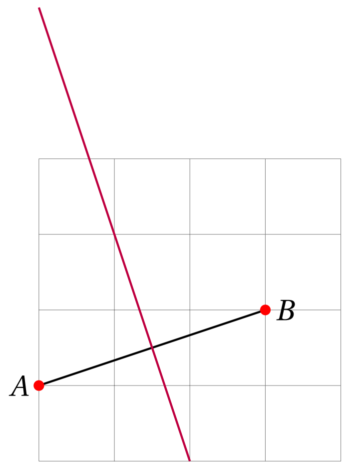
指定系数:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,1);
  \coordinate (B) at (3,2);
  \draw[thick] (A) -- (B);
  \draw[thick,purple,
    start modifier=.25,end modifier=.75,
    perpendicular bisector={A,B}];
  \foreach \p/\placement in {A/left,B/right}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



可以是负数, 这样就在相反方向:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,1);
  \coordinate (B) at (3,2);
  \draw[thick] (A) -- (B);
  \draw[thick,purple,
    start modifier=-.25,end modifier=0.75,
    perpendicular bisector={A,B}];
  \foreach \p/\placement in {A/left,B/right}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.11 垂线 Perpendicular Line

调用方式

```
perpendicular={A,B,C}
```

参数说明

A,B,C 三点坐标

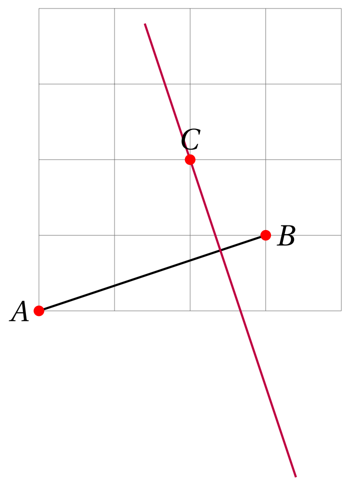
构造过 C 垂直于 AB 的直线 (设垂足为 D), 默认起点为 $D+(B-A)\cdot\mathbf{i}$, 终点为 $D-(B-A)\cdot\mathbf{i}$. 可以对起始点进行调整, 见1.18.

示例

过直线外一点的垂线:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
  \coordinate (B) at (3,1);
  \coordinate (C) at (2,2);
  \draw[thick] (A) -- (B);
```

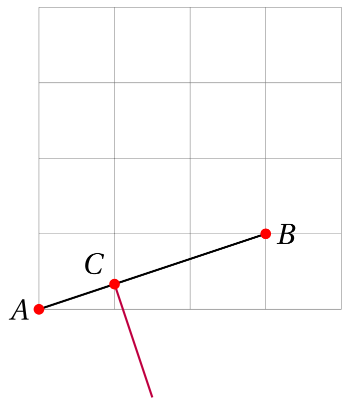
```
  \path[draw, thick, purple, perpendicular={A,B,C}];
  \foreach \p/\placement in {A/left,B/right,C/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



过直线上一点的垂线:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
  \coordinate (B) at (3,1);
  \coordinate (C) at ($(A)!1/3!(B)$);
  \draw[thick] (A) -- (B);
  \path[draw, thick, purple,
    start modifier=.5, end modifier=.75,
    perpendicular={A,B,C}];
  \foreach \p/\placement in {A/left,B/right,C/above
   ↪ left}{
```

```
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.12 平行线 Parallel Line

调用方式

```
parallel={A,B,C}
```

参数说明

过一点 C 作直线 AB 平行线, (如果 C 在 AB 上, 则重合).

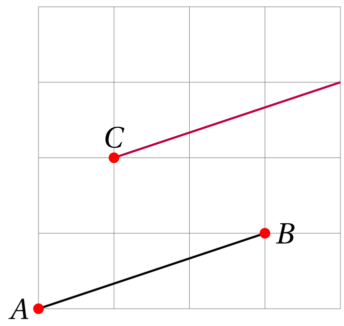
首先将点 C 按向量 AB 平移至 D. 可以对起始点进行调整, 见1.18.

示例

指定起始点距离 C 的位置, 方向是 CD, 负值代表相反方向:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
```

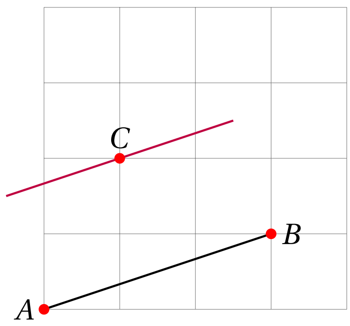
```
  \coordinate (B) at (3,1);
  \coordinate (C) at (1,2);
  \draw[thick] (A) -- (B);
  \path[draw, thick, purple, parallel={A,B,C}];
  \foreach \p/\placement in {A/left,B/right,C/above}{
    \fill[red] (\p) circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



指定系数:

```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (4,4);
  \coordinate (A) at (0,0);
  \coordinate (B) at (3,1);
  \coordinate (C) at (1,2);
  \draw[thick] (A) -- (B);
  \path[draw, thick, purple,
    start modifier=-.5, end modifier=.5,
    parallel={A,B,C}];
  \foreach \p/\placement in {A/left,B/right,C/above}{
    \fill[red] (\p) circle (2pt);
```

```
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



1.13 圆 Circle

调用方式

```
circle={O,A}
```

参数说明

O 圆心

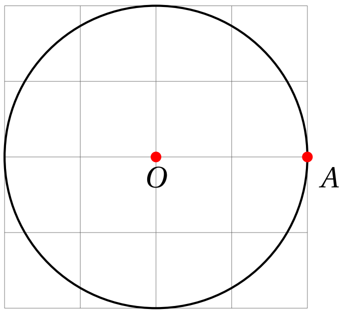
A 圆上一点

构造圆心为 O, 经过 A 的圆.

示例

```
\begin{tikzpicture}
  \draw[help lines] (-2,-2) grid (2,2);
  \coordinate (O) at (0,0);
  \coordinate (A) at +(0:2);  % 圆上一点，相对坐标
  \draw[thick,circle={O,A}];
```

```
  \foreach \p in {O,A}
    \fill[red] (\p) circle (2pt);
  \draw (O) node[below] {$O$};
  \draw (A) node[below right] {$A$};
\end{tikzpicture}
```



1.14 直线与圆的切点 Tangent Point

调用方式

```
tangent point={O,A,P}
```

参数说明

O ：圆心坐标

A ：为圆上任意一点

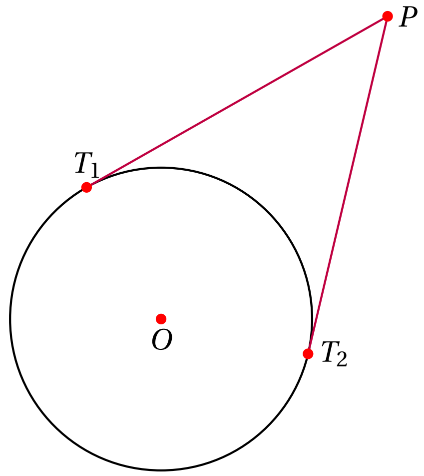
P ：圆外一点坐标

过圆 (O 为圆心, A 为圆上任意一点) 外一点 P 作切线, 求得一个切点 (在向量 OP 的左边), 另外一点可以通过对称 (`reflect={O,P,T}`) 求得.

示例

```
\begin{tikzpicture}
  \coordinate (O) at (0,0);
```

```
  \coordinate (A) at +(0:2);  % 圆上一点，相对坐标
  \coordinate (P) at (3,4);
  \coordinate[tangent point={O,A,P}] (T1);
  \coordinate[reflect={O,P,T1}] (T2);
  \draw[thick,circle={O,A}];
  \draw[thick,purple] (P) -- (T1) (P) -- (T2);
  \foreach \p in {O,P,T1,T2}
    \fill[red] (\p) circle (2pt);
  \draw (O) node[below] {$O$};
  \draw (P) node[right] {$P$};
  \draw (T1) node[above] {$T_1$};
  \draw (T2) node[right] {$T_2$};
\end{tikzpicture}
```



1.15 外位似中心 External Homothetic Center

调用方式

```
external center={O1,A1,O2,A2}
```

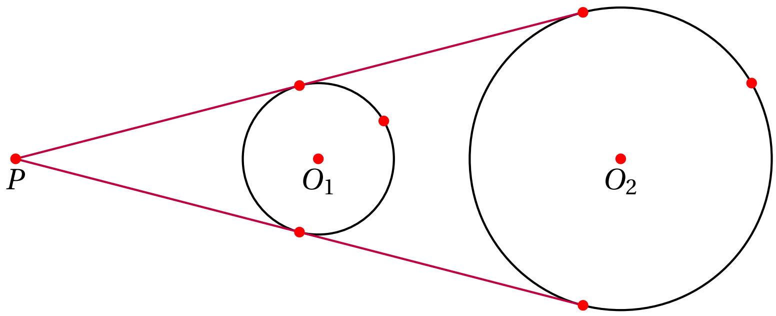
参数说明

求圆 1 (O_1 为圆心, A_1 为圆上任意一点) 和圆 2 (O_2 为圆心, A_2 为圆上任意一点) 的外位似中心 (external homothetic center)[2].

示例

作外公切线: 先求位似中心, 可以求得两圆的外公切线.

```
\begin{tikzpicture}
  \tikzmath {
    \a = 30;
    \b = \a;
    \r1 = 1;
    \r2 = 2;
  }
  \coordinate (O1) at (0,0);
  \coordinate (A1) at ($(O1)+(\a:\r1)$);
  \coordinate (O2) at (4,0);
  \coordinate (A2) at ($(O2)+(\b:\r2)$);
  \coordinate[external center={O1,A1,O2,A2}] (P);
  \coordinate[tangent point={O1,A1,P}] (B);
  \coordinate[tangent point={O2,A2,P}] (C);
  \coordinate[reflect={O1,O2,B}] (D);
  \coordinate[reflect={O1,O2,C}] (E);
  \draw[thick,circle={O1,A1}];
  \draw[thick,circle={O2,A2}];
  \draw[thick,purple] (P) -- (C) (P) -- (E);
  \foreach \p in {A1,A2,B,C,D,E,O1,O2,P}
    \fill[red] (\p) circle (2pt);
  \draw (O1) node[below] {$O_1$};
  \draw (O2) node[below] {$O_2$};
  \draw (P) node[below] {$P$};
\end{tikzpicture}
```



1.16 内位似中心 Internal Homothetic Center

调用方式

```
internal center={O1,A1,O2,A2}
```

参数说明

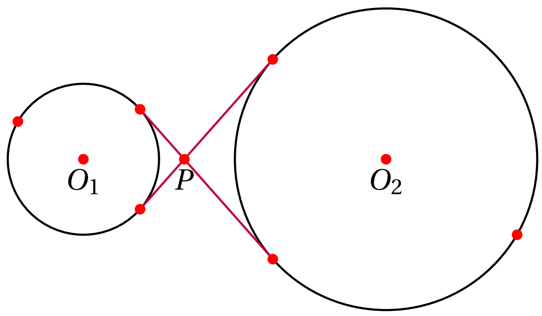
求圆 1 (O_1 为圆心, A_1 为圆上任意一点) 和圆 2 (O_2 为圆心, A_2 为圆上任意一点) 的内位似中心 (internal homothetic center) [2].

示例

作内公切线: 先求位似中心, 可以求得两圆的内公切线.

```
\begin{tikzpicture}
  \tikzmath {
    \a = 150;
    \b = \a - 180;
    \r1 = 1;
    \r2 = 2;
  }
  \coordinate (O1) at (0,0);
  \coordinate (A1) at ($(O1)+(\a:\r1)$);
  \coordinate (O2) at (4,0);
  \coordinate (A2) at ($(O2)+(\b:\r2)$);
```

```
  \coordinate[internal center={O1,A1,O2,A2}] (P);
  \coordinate[tangent point={O1,A1,P}] (B);
  \coordinate[tangent point={O2,A2,P}] (C);
  \coordinate[reflect={O1,O2,B}] (D);
  \coordinate[reflect={O1,O2,C}] (E);
  \draw[thick,circle={O1,A1}];
  \draw[thick,circle={O2,A2}];
  \draw[thick,purple] (P) -- (B) (P) -- (C) (P) -- (D)
   ↪ (P) -- (E);
  \foreach \p in {A1,A2,B,C,D,E,O1,O2,P}
    \fill[red] (\p) circle (2pt);
  \draw (O1) node[below] {$O_1$};
  \draw (O2) node[below] {$O_2$};
  \draw (P) node[below] {$P$};
\end{tikzpicture}
```



1.17 根轴 Radical Axis

调用方式

```
radical axis={O1,A1,O2,A2}
```

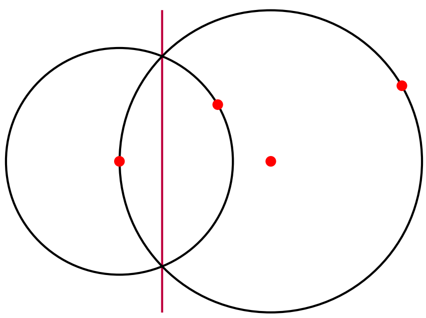
参数说明

构造两圆的根轴, 设与 O_1O_2 的交点为 P, 则默认起点为 $P+(O_2-O_1)\cdot\mathbf{i}$, 终点为 $P-(O_2-O_1)\cdot\mathbf{i}$. 可以对起始点进行调整, 见1.18.

示例

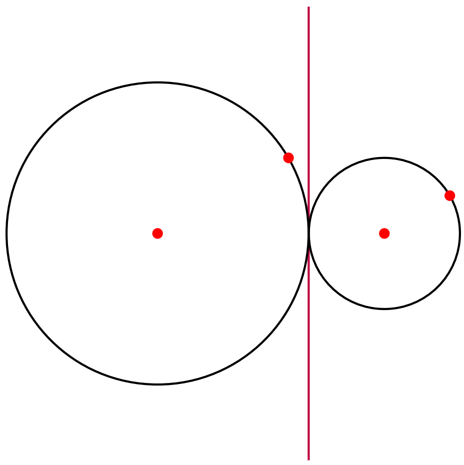
两圆相交:

```
\begin{tikzpicture}
  \tikzmath {
    \a = 30;
    \b = \a;
    \r1 = 1.5;
    \r2 = 2;
  }
  \coordinate (O1) at (0,0);
  \coordinate (A1) at ($(O1)+(\a:\r1)$);
  \coordinate (O2) at (2,0);
  \coordinate (A2) at ($(O2)+(\b:\r2)$);
  \draw[thick,purple,radical axis={O1,A1,O2,A2}];
  \draw[thick,circle={O1,A1}];
  \draw[thick,circle={O2,A2}];
  \foreach \p in {A1,A2,O1,O2}
    \fill[red] (\p) circle (2pt);
\end{tikzpicture}
```



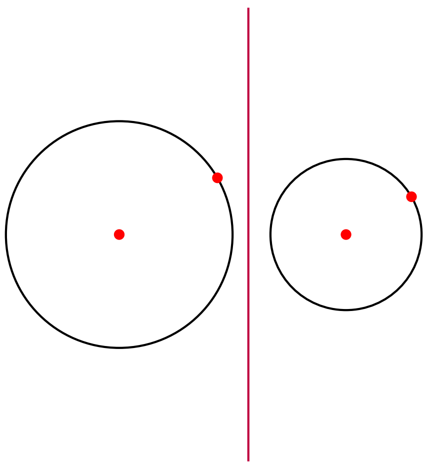
两圆外切:

```
\begin{tikzpicture}
  \tikzmath {
    \a = 30;
    \b = \a;
    \r1 = 2;
    \r2 = 1;
  }
  \coordinate (O1) at (0,0);
  \coordinate (A1) at ($(O1)+(\a:\r1)$);
  \coordinate (O2) at (3,0);
  \coordinate (A2) at ($(O2)+(\b:\r2)$);
  \draw[thick,purple,radical axis={O1,A1,O2,A2}];
  \draw[thick,circle={O1,A1}];
  \draw[thick,circle={O2,A2}];
  \foreach \p in {A1,A2,O1,O2}
    \fill[red] (\p) circle (2pt);
\end{tikzpicture}
```



两圆外离:

```
\begin{tikzpicture}
  \tikzmath {
    \a = 30;
    \b = \a;
    \r1 = 1.5;
    \r2 = 1;
  }
  \coordinate (O1) at (0,0);
  \coordinate (A1) at ($(O1)+(\a:\r1)$);
  \coordinate (O2) at (3,0);
  \coordinate (A2) at ($(O2)+(\b:\r2)$);
  \coordinate[radical axis={O1,A1,O2,A2}] (P);
  \draw[thick,purple,radical axis={O1,A1,O2,A2}];
  \draw[thick,circle={O1,A1}];
  \draw[thick,circle={O2,A2}];
  \foreach \p in {A1,A2,O1,O2}
    \fill[red] (\p) circle (2pt);
\end{tikzpicture}
```



1.18 Partway Modifiers and Distance Modifiers

`perpendicular bisector`,`perpendicular`,`parallel`,`radical axis` 等线段图形可以对起始点进行调整, 调整参数如下 [3]:

start modifier (default: 0), 长度或系数, 如: 1cm 或.75

end modifier (default: 1), 长度或系数, 如: 1cm 或.75

第 2 章　三角形的中心

2.1　重心 Centroid

调用方式

```
centroid={A,B,C}
```

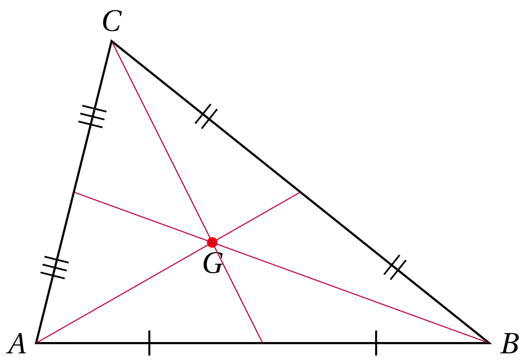
参数说明

A, B, C　三角形的顶点

示例

```
\begin{tikzpicture}
  \coordinate (A) at (-2,0);
  \coordinate (B) at (4,0);
  \coordinate (C) at (-1,4);
  \coordinate (D) at ($(B)!0.5!(C)$);
  \coordinate (E) at ($(C)!0.5!(A)$);
  \coordinate (F) at ($(A)!0.5!(B)$);
  \path[centroid={A,B,C}] coordinate (G);
  \fill (G) [red] circle (2pt);
  \draw (G) node[below] {$G$};
  \draw[thick] (A) -- (B) -- (C) -- cycle;
  \draw[purple] (A) -- (D) (B) -- (E) (C) -- (F);
```

```
  \draw (A) node[left] {$A$};
  \draw (B) node[right] {$B$};
  \draw (C) node[above] {$C$};
  \draw (A) -- (B) node[near start,sloped] {$|$}
   ↪ node[near end,sloped] {$|$};
  \draw (B) -- (C) node[near start,sloped] {$||$}
   ↪ node[near end,sloped] {$||$};
  \draw (C) -- (A) node[near start,sloped] {$|||$}
   ↪ node[near end,sloped] {$|||$};
\end{tikzpicture}
```



2.2 垂心 Orthocenter

调用方式

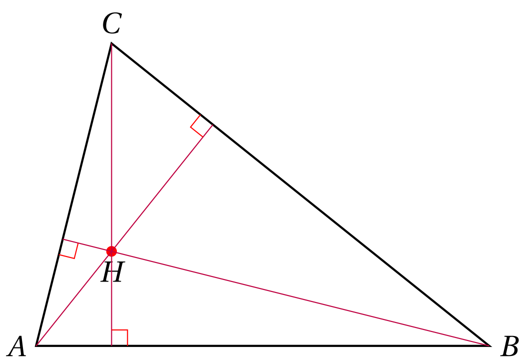
```
orthocenter={A,B,C}
```

参数说明

A, B, C 三角形的顶点

示例

```
\begin{tikzpicture}
  \coordinate (A) at (-2,0);
  \coordinate (B) at (4,0);
  \coordinate (C) at (-1,4);
  \path[orthocenter={A,B,C}] coordinate (H);
  \fill (H) [red] circle (2pt);
  \draw (H) node[below] {$H$};
  \draw[thick] (A) -- (B) -- (C) -- cycle;
  \coordinate (D) at ($(B)!(A)!(C)$);
  \coordinate (E) at ($(A)!(B)!(C)$);
  \coordinate (F) at ($(B)!(C)!(A)$);
  \draw[purple] (A) -- (D) (B) -- (E) (C) -- (F);
  \draw (A) node[left] {$A$};
  \draw (B) node[right] {$B$};
  \draw (C) node[above] {$C$};
  \pic [draw,red,angle radius=6pt] {right angle=H--D--C};
  \pic [draw,red,angle radius=6pt] {right angle=H--E--A};
  \pic [draw,red,angle radius=6pt] {right angle=H--F--B};
\end{tikzpicture}
```



2.3 外心 Circumcenter

调用方式

```
circumcenter={A,B,C}
```

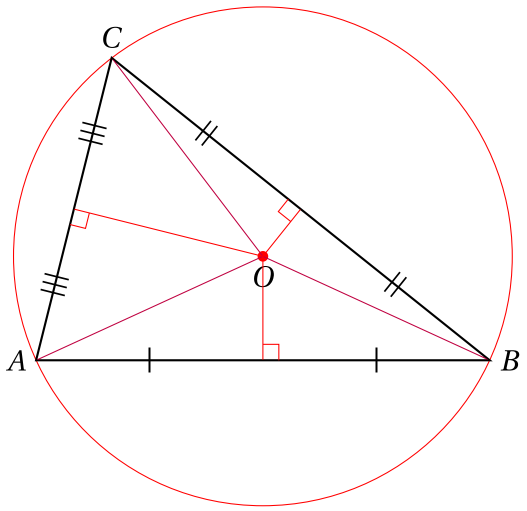
参数说明

A, B, C 三角形的顶点

示例

```
\begin{tikzpicture}
  \coordinate (A) at (-2,0);
  \coordinate (B) at (4,0);
  \coordinate (C) at (-1,4);
  \path[circumcenter={A,B,C}] coordinate (O);
  \fill (O) [red] circle (2pt);
  \draw (O) node[below] {$O$};
  \node[draw,red] at (O) [circle through=(A)]{};
  \draw[thick] (A) -- (B) -- (C) -- cycle;
  \draw[purple] (A) -- (O) (B) -- (O) (C) -- (O);
  \draw (A) node[left] {$A$};
  \draw (B) node[right] {$B$};
  \draw (C) node[above] {$C$};
  \coordinate (D) at ($(B)!(O)!(C)$);
  \coordinate (E) at ($(C)!(O)!(A)$);
  \coordinate (F) at ($(A)!(O)!(B)$);
  \draw[red] (O) -- (D) (O) -- (E) (O) -- (F);
  \draw (A) -- (B) node[near start,sloped] {$|$}
   ↪ node[near end,sloped] {$|$};
```

```
  \draw (B) -- (C) node[near start,sloped] {$||$}
    node[near end,sloped] {$||$};
  \draw (C) -- (A) node[near start,sloped] {$|||$}
    node[near end,sloped] {$|||$};
  \pic [draw,red,angle radius=6pt] {right angle=O--D--C};
  \pic [draw,red,angle radius=6pt] {right angle=O--E--A};
  \pic [draw,red,angle radius=6pt] {right angle=O--F--B};
\end{tikzpicture}
```



2.4 内心 Incenter

调用方式

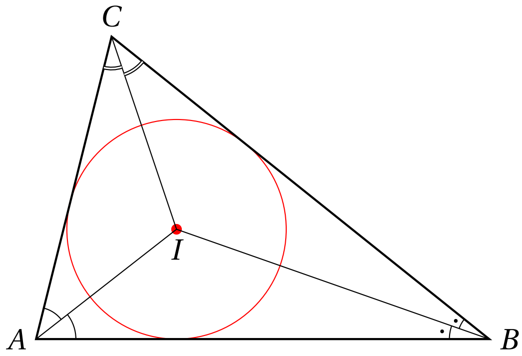
```
incenter={A,B,C}
```

参数说明

A, B, C 三角形的顶点

示例

```
\begin{tikzpicture}
  \coordinate (A) at (-2,0);
  \coordinate (B) at (4,0);
  \coordinate (C) at (-1,4);
  \path[incenter={A,B,C}] coordinate (I);
  \fill (I) [red] circle (2pt);
  \draw (I) node[below] {$I$};
  \node[draw,red] at (I) [circle through=($(B)!(I)!(C)$)]{};
  \draw[thick] (A) -- (B) -- (C) -- cycle;
  \draw (A) node[left] {$A$};
  \draw (B) node[right] {$B$};
  \draw (C) node[above] {$C$};
  \draw (A) -- (I) (B) -- (I) (C) -- (I);
  \pic [draw,angle radius=12pt] {angle=I--A--C};
  \pic [draw,angle radius=15pt] {angle=B--A--I};
  \pic [draw,double,angle radius=12pt] {angle=A--C--I};
  \pic [draw,double,angle radius=15pt] {angle=I--C--B};
  \pic [draw,pic text=.,angle radius=12pt,
    angle eccentricity=1.2] {angle=C--B--I};
  \pic [draw,pic text=.,angle radius=15pt,
    angle eccentricity=1.2] {angle=I--B--A};
\end{tikzpicture}
```



2.5 旁心 Excenter

调用方式

```
excenter={A,B,C}
```

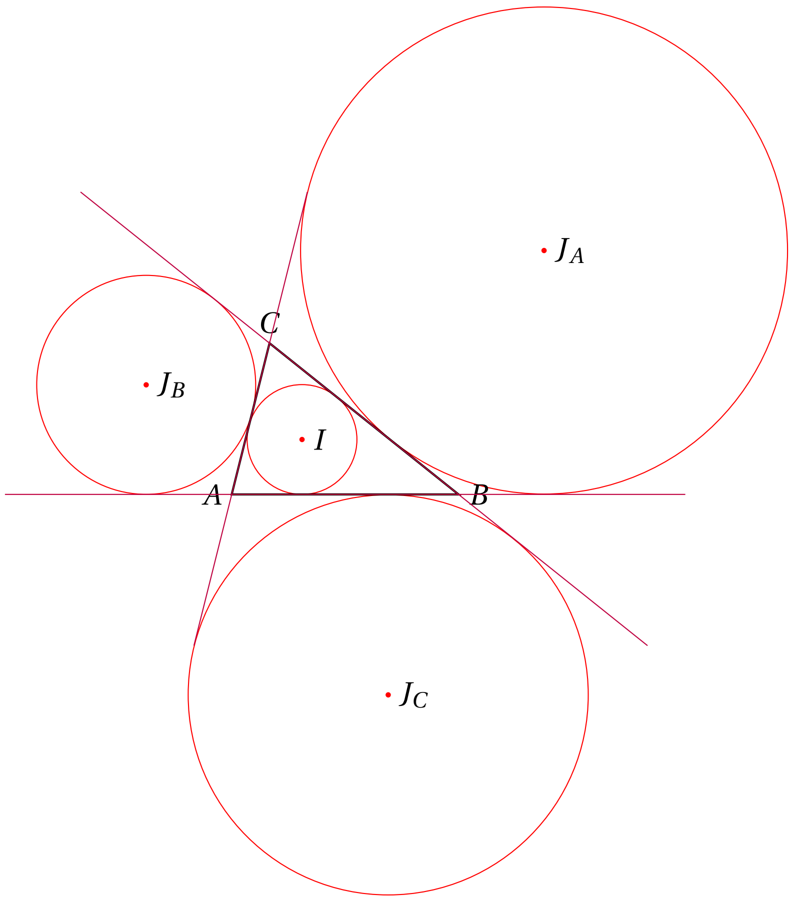
参数说明

A, B, C 三角形的顶点, 返回与 A 相对的旁心, 调换顶点顺序就可以得到 3 个旁心.

示例

```
\begin{tikzpicture}[scale=.5]
  \coordinate (A) at (-2,0);
  \coordinate (B) at (4,0);
  \coordinate (C) at (-1,4);
  \path[incenter={A,B,C}] coordinate (I);
  \path[excenter={A,B,C}] coordinate (JA);
  \path[excenter={B,A,C}] coordinate (JB);
  \path[excenter={C,A,B}] coordinate (JC);
  \foreach \point in {I,JA,JB,JC}
    \fill (\point) [red] circle (2pt);
```

```
    \node[draw,red] at (I) [circle
    ↪ through=($(B)!(I)!(C)$)]{};
    \node[draw,red] at (JA) [circle
    ↪ through=($(B)!(JA)!(C)$)]{};
    \node[draw,red] at (JB) [circle
    ↪ through=($(B)!(JB)!(C)$)]{};
    \node[draw,red] at (JC) [circle
    ↪ through=($(B)!(JC)!(C)$)]{};
    \draw[thick] (A) -- (B) -- (C) -- cycle;
    \draw (A) node[left] {$A$};
    \draw (B) node[right] {$B$};
    \draw (C) node[above] {$C$};
    \draw[purple] ($(A)!-1!(B)$) -- ($(A)!2!(B)$);
    \draw[purple] ($(B)!-1!(C)$) -- ($(B)!2!(C)$);
    \draw[purple] ($(C)!-1!(A)$) -- ($(C)!2!(A)$);
    \draw (I) node[right] {$I$};
    \draw (JA) node[right] {$J_A$};
    \draw (JB) node[right] {$J_B$};
    \draw (JC) node[right] {$J_C$};
\end{tikzpicture}
```



2.6 九点圆圆心 Nine-Point Center

调用方式

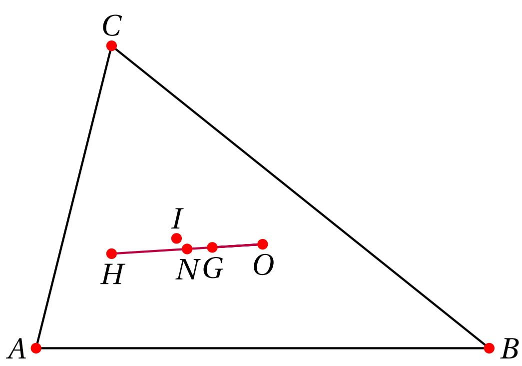
```
nine-point center={A,B,C}
```

参数说明

A, B, C 三角形的顶点

示例

```latex
\begin{tikzpicture}
  \coordinate (A) at (-2,0);
  \coordinate (B) at (4,0);
  \coordinate (C) at (-1,4);
  \path[orthocenter={A,B,C}] coordinate (H);
  \path[circumcenter={A,B,C}] coordinate (O);
  \path[centroid={A,B,C}] coordinate (G);
  \path[incenter={A,B,C}] coordinate (I);
  \path[nine-point center={A,B,C}] coordinate (N);
  \draw[thick] (A) -- (B) -- (C) -- cycle;
  \draw[thick,purple] (H) -- (O) -- (G);
  \foreach \p/\placement in {A/left,B/right,C/above,
  H/below,O/below,G/below,I/above,N/below}{
    \fill (\p) [red] circle (2pt);
    \draw (\p) node[\placement] {$\p$};
  }
\end{tikzpicture}
```



附录 A　两直线的交点

求解两直线交点的方程 [4]:

$$\begin{vmatrix} x & y & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} = 0$$

$$\begin{vmatrix} x & y & 1 \\ x_3 & y_3 & 1 \\ x_4 & y_4 & 1 \end{vmatrix} = 0$$

注意, 两个方程的系数都是行列式, 解得:

$$x = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & \begin{vmatrix} x_3 & 1 \\ x_4 & 1 \end{vmatrix} \end{vmatrix}}{\begin{vmatrix} \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix} & \begin{vmatrix} y_1 & 1 \\ y_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & 1 \\ x_4 & 1 \end{vmatrix} & \begin{vmatrix} y_3 & 1 \\ y_4 & 1 \end{vmatrix} \end{vmatrix}} = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & x_1 - x_2 \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & x_3 - x_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & y_1 - y_2 \\ x_3 - x_4 & y_3 - y_4 \end{vmatrix}}$$

$$y = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & \begin{vmatrix} y_1 & 1 \\ y_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & \begin{vmatrix} y_3 & 1 \\ y_4 & 1 \end{vmatrix} \end{vmatrix}}{\begin{vmatrix} \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix} & \begin{vmatrix} y_1 & 1 \\ y_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & 1 \\ x_4 & 1 \end{vmatrix} & \begin{vmatrix} y_3 & 1 \\ y_4 & 1 \end{vmatrix} \end{vmatrix}} = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & y_1 - y_2 \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & y_3 - y_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & y_1 - y_2 \\ x_3 - x_4 & y_3 - y_4 \end{vmatrix}}$$

进一步化简得到[1]:

$$x = \frac{(x_1 y_2 - y_1 x_2)(x_3 - x_4) - (x_1 - x_2)(x_3 y_4 - y_3 x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}$$
$$y = \frac{(x_1 y_2 - y_1 x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 y_4 - y_3 x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}$$

上述方法给出的交点坐标公式在 TikZ 环境中的计算稳定性不够好, 经常出现 `Dimension too large` 错误, 究其原因是分母可能有时会比较小. 下面给出一个计算更稳定的公式.

我们可以给出两条直线的参数方程:

直线 L_1 的方程:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + s \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix}$$

直线 L_2 的方程:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} + t \begin{bmatrix} x_4 - x_3 \\ y_4 - y_3 \end{bmatrix}$$

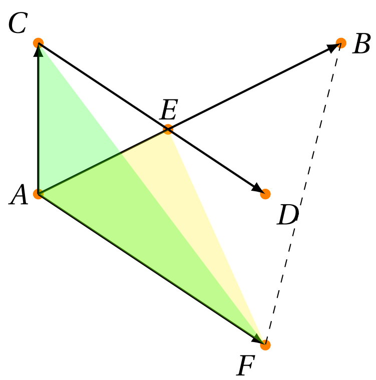
[1] https://en.wikipedia.org/wiki/Lineline_intersection

可以解出 s, t:

$$s = \frac{\begin{vmatrix} x_1 - x_3 & x_3 - x_4 \\ y_1 - y_3 & y_3 - y_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & x_3 - x_4 \\ y_1 - y_2 & y_3 - y_4 \end{vmatrix}}$$

$$t = \frac{\begin{vmatrix} x_1 - x_3 & x_1 - x_2 \\ y_1 - y_3 & y_1 - y_2 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & x_3 - x_4 \\ y_1 - y_2 & y_3 - y_4 \end{vmatrix}}$$

我们也可从几何的角度来分析:



$$\overrightarrow{AE} = s\overrightarrow{AB}$$

$$\begin{aligned} s &= \frac{S_{\triangle AEF}}{S_{\triangle ABF}} \\ &= \frac{S_{\triangle ACF}}{S_{\triangle ABF}} \\ &= \frac{\overrightarrow{AF} \times \overrightarrow{AC}}{\overrightarrow{AF} \times \overrightarrow{AB}} \\ &= \frac{\overrightarrow{CD} \times \overrightarrow{AC}}{\overrightarrow{CD} \times \overrightarrow{AB}} \end{aligned}$$

为了保证数值计算的稳定性, 可以对下面的方程进行列主元消元法求解:

$$x_1 + s(x_2 - x_1) = x3 + t(x_4 - x3)$$
$$y_1 + s(y_2 - y_1) = y3 + t(y_4 - y3)$$

附录 B　源代码

```
\ProvidesFile{tikzlibraryeuclidea.code.tex}[2023/12/21
↪ v1.2.0 A tikz library for plane geometry]

\usetikzlibrary{math,calc}

% https://tex.stackexchange.com/questions/455991/
↪ pgfmath-function-for-strings-and-numbers
% Solving the error:
% Package PGF Math: Could not parse input 'A' as a
↪ floating
% point number, sorry. The unreadable part was near
'A'..
\pgfkeys{
  /pgf/fpu/handlers/invalid number/.code = {%
    \pgfmathfloatparsenumber{3Y0.0e0]}%
  }
}

\makeatletter

% 注意：计算过程是保留坐标单位（pt）的，所以存在乘除法单位
↪ 的问题，首先数值始终携带单位，
```

```
% 在 calc 运算时有的需要转换为标量；将坐标转换为 pt 值，数值可能超出限值，出现
% Dimension too large 错误，在计算长度时及时进行缩小
% https://tex.stackexchange.com/questions/475556/tikz-why-is-dimension-too-large
% 具体方法是修改默认的 1cm，如：[scale=1.0,x=0.5cm,y=0.5cm]
% 注意此处的变量不要和 tikzpicture 环境重名，否则被替换掉
% triangle centers:
% https://mathworld.wolfram.com/BarycentricCoordinates.html
\tikzmath{
  % 采用列主元消元法求直线 P1P2 与直线 P3P4 的交点 P 位置参数 s: s = P1P/P1P2
  function intersectll(\x1,\y1,\x2,\y2,\x3,\y3,\x4,\y4)
{
    \a1 = \x2-\x1; \b1 = \x3-\x4; \c1 = \x3-\x1;
    \a2 = \y2-\y1; \b2 = \y3-\y4; \c2 = \y3-\y1;
    \dmax = max(max(abs(\a1),abs(\a2)), max(abs(\b1),abs(\b2)));
    \a1 = \a1/\dmax; \b1 = \b1/\dmax; \c1=\c1/\dmax;
    \a2 = \a2/\dmax; \b2 = \b2/\dmax; \c2=\c2/\dmax;
    if abs(\a1) < abs(\a2) then {
      \temp = \a1; \a1 = \a2; \a2 = \temp;
      \temp = \b1; \b1 = \b2; \b2 = \temp;
      \temp = \c1; \c1 = \c2; \c2 = \temp;
    };
    \b1 = \b1/\a1; \c1 = \c1/\a1; \a1 = 1.0;
    \b2 = \b2-\a2*\b1; \c2 = \c2-\a2*\c1; \a2 = 0.0;
    \n2 = \c2/\b2; \n1 = \c1-\b1*\n2;
```

```
    return \n1;
  };
}

\tikzset{
  % specifying start and end with modifiers(see tikz
   ↪  manual 13.5)
  % commands supporting partway modifiers:
  % radical axis, perpendicular bisector, perpendicular,
   ↪  parallel
  start modifier/.initial = 0,
  start modifier/.default = 0,
  end modifier/.initial = 1,
  end modifier/.default = 1,
  % ========== Coordinates Transformations ==========
  % affine={A,B,k}: returns affine combination of two
   ↪  points
  % with affine ratio, i.e. A + k * ( B - A )
  affine/.style args = {#1,#2,#3}{
    insert path = {
      ($(#1)!{#3}!(#2)$)
    }
  },
  % translate={A,B,C}: returns translation of C by
  % the vector AB, i.e. C + ( B - A )
  translate/.style args = {#1,#2,#3}{
    insert path = {
      ($(#3)+(#2)-(#1)$)
    }
  },
```

```
% reflect={A,B,C}: reflects point C across line AB.
reflect/.style args = {#1,#2,#3}{
  insert path = {
    let
      \p{ft} = ($(#1)!(#3)!(#2)$),% perpendicular foot
    in ($(#3)!2!(\p{ft})$)
  }
},
% inverse={O,A,P}: returns inverse point P with respect to
% a reference circle(O,A).
inverse/.style args = {#1,#2,#3}{
  insert path = {
    let
      \p{OA} = ($(#2)-(#1)$),
      \p{OP} = ($(#3)-(#1)$),
      \n{r} = {veclen(\p{OA})},
      \n{d} = {veclen(\p{OP})},
      \n1 = {scalar((\n{r}/\n{d}))},
    in ($(#1)!\n1*\n1!(#3)$)
  }
},
revolve/scale/.initial = 1,% angle scale
revolve/@angle/.initial = 90,
revolve/@argn/.initial = 1,% arguments count
% set revolve/@angle with certain degrees or angle of a vector
revolve/@set angle 1/.code args = {#1}{
  \pgfmathanglebetweenpoints
    {\pgfpoint{0cm}{0cm}}
```

```
    {\pgfpointanchor{#1}{center}}
  \pgfkeysalso{/tikz/revolve/@angle = \pgfmathresult}
  \typeout{=========================}
  \typeout{/tikz/revolve/@angle:\pgfkeysvalueof{/tikz/␣
   ↪  revolve/@angle}}
  \typeout{=========================}
},
% set revolve/@angle with angle between two position
 ↪  vectors
revolve/@set angle 2/.code args = {#1,#2}{
  \pgfmathanglebetweenpoints
    {\pgfpointanchor{#1}{center}}
    {\pgfpointanchor{#2}{center}}
  \pgfkeysalso{/tikz/revolve/@angle = \pgfmathresult}
  \typeout{=========================}
  \typeout{/tikz/revolve/@angle:\pgfkeysvalueof{/tikz/␣
   ↪  revolve/@angle}}
  \typeout{=========================}
},
% set revolve/@angle with angle {A,B,C}, angle between
 ↪  two sides
% (A is apex, B is the start point, C is the end point)
 ↪
revolve/@set angle 3/.code args = {#1,#2,#3}{
  \pgfmathanglebetweenlines
    {\pgfpointanchor{#1}{center}}
    {\pgfpointanchor{#2}{center}}
    {\pgfpointanchor{#1}{center}}
    {\pgfpointanchor{#3}{center}}
  \pgfkeysalso{/tikz/revolve/@angle = \pgfmathresult}
```

```
    \typeout{=========================}
    \typeout{/tikz/revolve/@angle:\pgfkeysvalueof{/tikz/␣
    ↪  revolve/@angle}}
    \typeout{=========================}
  },
  % set revolve/@angle with angle between two
  ↪  vectors(ccw, AB and CD)
  revolve/@set angle 4/.code args = {#1,#2,#3,#4}{
    \pgfmathanglebetweenlines
      {\pgfpointanchor{#1}{center}}
      {\pgfpointanchor{#2}{center}}
      {\pgfpointanchor{#3}{center}}
      {\pgfpointanchor{#4}{center}}
    \pgfkeysalso{/tikz/revolve/@angle = \pgfmathresult}
    \typeout{=========================}
    \typeout{/tikz/revolve/@angle:\pgfkeysvalueof{/tikz/␣
    ↪  revolve/@angle}}
    \typeout{=========================}
  },
  revolve/angle/.code = {%
    \pgfmathfloatparsenumber{#1}
    \pgfmathfloattomacro{\pgfmathresult}{\F}{\M}{\E}
    \ifnum \F < 3%number
      \pgfmathparse{#1}
    \else
      \euclidea@ParseArguments#1\euclidea@stop
      \euclidea@ComputeAngle#1\euclidea@stop
    \fi
    \pgfkeysalso{/tikz/revolve/@angle = \pgfmathresult}
  },
```

```
% revolve={A,B}: rotates point B by the angle around
 ↪  point A.
revolve/.style args = {#1,#2}{
  insert path = {
    let
      \n1 = {\pgfkeysvalueof{/tikz/revolve/@angle}},
      \n2 = {\pgfkeysvalueof{/tikz/revolve/scale}}
    in ($(#1)!1!\n1*\n2:(#2)$)
  }
},
% cut a line segment of a certain length on a straight
 ↪  line
intercept/@length/.initial = 1cm,
intercept/scale/.initial = 1,% length scale
intercept/length/.code = {% set length by distance of
 ↪  segment
  \pgfutil@in@{,}{#1}
  \ifpgfutil@in@%compute segment length
    \euclidea@ComputeLength#1\euclidea@stop
    \pgfkeysalso{/tikz/intercept/@length =
     ↪  \pgfmathresult}
  \else
    \pgfkeysalso{/tikz/intercept/@length = #1}
  \fi
  \typeout{=========================}
  \typeout{/tikz/intercept/@length:\pgfkeysvalueof{/␣
   ↪  tikz/intercept/@length}}
  \typeout{=========================}
},
% intercept={A,B}: intercepts a line segment(starting
```

```
% from point A) of a certain length on line AB.
intercept/.style args = {#1,#2}{
  insert path = {
    let
      \n1 = {\pgfkeysvalueof{/tikz/intercept/@length}},
      \n2 = {\pgfkeysvalueof{/tikz/intercept/scale}},
      \p{AB} = ($(#2)-(#1)$),
      \n{d} = {veclen(\p{AB})},
      \n3 = {scalar(\n1*\n2/\n{d})}
    in ($(#1)!\n3!(#2)$)
  }
},
% intersect={A,B,C,D}: returns the intersection
 ↪  coordinate
% of line AB and line CD.
% https://en.wikipedia.org/wiki/Line%E2%80%
 ↪  93line_intersection
intersect/.style args = {#1,#2,#3,#4}{
  insert path = {
    let
      \p1 = (#1), \p2 = (#2), \p3 = (#3), \p4 = (#4),
      \n1 = {intersectll(\x1,\y1,\x2,\y2,\x3,\y3,\x4,\
       ↪  y4)},
    in ($(\p1)!\n1!(\p2)$)
  }
},
% ========== Triangle Centers ==========
% calculated from barycentric coordinates
% incenter = {A,B,C}
incenter/.style args = {#1,#2,#3}{
```

```
    insert path = {
      let
        \p1 = (#1), \p2 = (#2), \p3 = (#3),
        \p{AB} = ($(#2)-(#1)$),
        \p{BC} = ($(#3)-(#2)$),
        \p{CA} = ($(#1)-(#3)$),
        \n{a} = {veclen(\x{BC}, \y{BC})},
        \n{b} = {veclen(\x{CA}, \y{CA})},
        \n{c} = {veclen(\x{AB}, \y{AB})},
        \n{s} = {\n{a}+\n{b}+\n{c}},
        \n1 = {\n{a}/\n{s}},
        \n2 = {\n{b}/\n{s}},
        \n3 = {\n{c}/\n{s}},
      in ({\n1*\x1+\n2*\x2+\n3*\x3,\n1*\y1+\n2*\y2+\n3*\
       ↪  y3})
    }
  },
  % excenter = {A,B,C}, returns excenter opposite to the
   ↪  vertex A
  excenter/.style args = {#1,#2,#3}{
    insert path = {
      let
        \p1 = (#1), \p2 = (#2), \p3 = (#3),
        \p{AB} = ($(#2)-(#1)$),
        \p{BC} = ($(#3)-(#2)$),
        \p{CA} = ($(#1)-(#3)$),
        \n{a} = {veclen(\x{BC}, \y{BC})},
        \n{b} = {veclen(\x{CA}, \y{CA})},
        \n{c} = {veclen(\x{AB}, \y{AB})},
        \n{s} = {-\n{a}+\n{b}+\n{c}},
```

```
      \n1 = {\n{a}/\n{s}},
      \n2 = {\n{b}/\n{s}},
      \n3 = {\n{c}/\n{s}},
    in ({-\n1*\x1+\n2*\x2+\n3*\x3,-\n1*\y1+\n2*\y2+\
     ↪  n3*\y3})
  }
},
% circumcenter = {A,B,C}
circumcenter/.style args = {#1,#2,#3}{
  insert path = {
    let
      \p1 = (#1), \p2 = (#2), \p3 = (#3),
      \p{AB} = ($(#2)-(#1)$),
      \p{BC} = ($(#3)-(#2)$),
      \p{CA} = ($(#1)-(#3)$),
      \n{a} = {veclen(\x{BC}, \y{BC})},
      \n{b} = {veclen(\x{CA}, \y{CA})},
      \n{c} = {veclen(\x{AB}, \y{AB})},
      \n{m} = {max(max(\n{a},\n{b}),\n{c})},
      \n{a} = {\n{a}/\n{m}},
      \n{a} = {\n{a}*\n{a}},
      \n{b} = {\n{b}/\n{m}},
      \n{b} = {\n{b}*\n{b}},
      \n{c} = {\n{c}/\n{m}},
      \n{c} = {\n{c}*\n{c}},
      \n1 = {\n{a}*(\n{b}+\n{c}-\n{a})},
      \n2 = {\n{b}*(\n{c}+\n{a}-\n{b})},
      \n3 = {\n{c}*(\n{a}+\n{b}-\n{c})},
      \n{s} = {\n1+\n2+\n3},
      \n1 = {\n1/\n{s}},
```

```
      \n2 = {\n2/\n{s}},
      \n3 = {\n3/\n{s}},
    in ({\n1*\x1+\n2*\x2+\n3*\x3,\n1*\y1+\n2*\y2+\n3*\
    ↪  y3})
  }
},
% orthocenter = {A,B,C}
orthocenter/.style args = {#1,#2,#3}{
  insert path = {
    let
      \p1 = (#1), \p2 = (#2), \p3 = (#3),
      \p{AB} = ($(#2)-(#1)$),
      \p{BC} = ($(#3)-(#2)$),
      \p{CA} = ($(#1)-(#3)$),
      \n{a} = {veclen(\x{BC}, \y{BC})},
      \n{b} = {veclen(\x{CA}, \y{CA})},
      \n{c} = {veclen(\x{AB}, \y{AB})},
      \n{m} = {max(max(\n{a},\n{b}),\n{c})},
      \n{a} = {\n{a}/\n{m}},
      \n{a} = {\n{a}*\n{a}},
      \n{b} = {\n{b}/\n{m}},
      \n{b} = {\n{b}*\n{b}},
      \n{c} = {\n{c}/\n{m}},
      \n{c} = {\n{c}*\n{c}},
      \n{a2} = {\n{b}+\n{c}-\n{a}},
      \n{b2} = {\n{c}+\n{a}-\n{b}},
      \n{c2} = {\n{a}+\n{b}-\n{c}},
      \n1 = {\n{c2}*\n{b2}},
      \n2 = {\n{a2}*\n{c2}},
      \n3 = {\n{b2}*\n{a2}},
```

```
        \n{s} = {\n1+\n2+\n3},
        \n1 = {\n1/\n{s}},
        \n2 = {\n2/\n{s}},
        \n3 = {\n3/\n{s}},
      in ({\n1*\x1+\n2*\x2+\n3*\x3,\n1*\y1+\n2*\y2+\n3*\
       ↪ y3})
    }
  },
  % centroid = {A,B,C}
  centroid/.style args = {#1,#2,#3}{
    insert path = {
      let
        \p1 = (#1), \p2 = (#2), \p3 = (#3),
      in ({(\x1+\x2+\x3)/3},{(\y1+\y2+\y3)/3})
    }
  },
  % nine-pint center = {A,B,C}
  nine-point center/.style args = {#1,#2,#3}{
    insert path = {
      let
        \p1 = (#1), \p2 = (#2), \p3 = (#3),
        \p{AB} = ($(#2)-(#1)$),
        \p{BC} = ($(#3)-(#2)$),
        \p{CA} = ($(#1)-(#3)$),
        \n{a} = {veclen(\x{BC}, \y{BC})},
        \n{b} = {veclen(\x{CA}, \y{CA})},
        \n{c} = {veclen(\x{AB}, \y{AB})},
        \n{m} = {max(max(\n{a},\n{b}),\n{c})},
        \n{a} = {\n{a}/\n{m}},
        \n{a} = {\n{a}*\n{a}},
```

```
      \n{b} = {\n{b}/\n{m}},
      \n{b} = {\n{b}*\n{b}},
      \n{c} = {\n{c}/\n{m}},
      \n{c} = {\n{c}*\n{c}},
      \n1 = {\n{a}*(\n{b}+\n{c})-(\n{b}-\n{c})*(\n{b}-
       ↪ \n{c})},
      \n2 = {\n{b}*(\n{c}+\n{a})-(\n{c}-\n{a})*(\n{c}-
       ↪ \n{a})},
      \n3 = {\n{c}*(\n{a}+\n{b})-(\n{a}-\n{b})*(\n{a}-
       ↪ \n{b})},
      \n{s} = {\n1+\n2+\n3},
      \n1 = {\n1/\n{s}},
      \n2 = {\n2/\n{s}},
      \n3 = {\n3/\n{s}},
    in ({\n1*\x1+\n2*\x2+\n3*\x3,\n1*\y1+\n2*\y2+\n3*\
     ↪ y3})
  }
},
% ========== Circle Operations ==========
% circle = {O,A}, creates circle with the center (O)
 ↪ through A
circle/.style args = {#1,#2}{
  insert path = {
    let
      \p{OA} = ($(#2)-(#1)$),
    in (#1) circle ({veclen(\p{OA})})
  }
},
% tagent point = {O,A,P}
```

```
% O,A: center of circle and an abitary point on the
  ↪ circle
% P: a point outside the circle
tangent point/.style args = {#1,#2,#3}{
  insert path = {
    let
      \p{OA} = ($(#2)-(#1)$), % 半径
      \p{OP} = ($(#3)-(#1)$),
      \n1 = {veclen(\p{OA})},
      \n2 = {veclen(\p{OP})},
      \n3 = {scalar(\n1/\n2)}
    in ($(#1)!\n3!{acos(\n1/\n2)}:(#3)$)
  }
},
% external homothetic center
% O1,A1: center of circle 1 and an abitary point on the
  ↪ circle
% O2,A2: center of circle 2 and an abitary point on the
  ↪ circle
external center/.style args = {#1,#2,#3,#4}{
  insert path = {
    let
      \p{O1A1} = ($(#2)-(#1)$),% 半径 O1A1
      \p{O2A2} = ($(#4)-(#3)$),% 半径 O2A2
      \n{r1} = {veclen(\p{O1A1})},
      \n{r2} = {veclen(\p{O2A2})},
      \n1 = {scalar(\n{r1}/(\n{r1}-\n{r2}))}
    in ($(#1)!\n1!(#3)$)
  }
},
```

```
% internal homothetic center
% O1,A1: center of circle 1 and an abitary point on the
  ↪ circle
% O2,A2: center of circle 2 and an abitary point on the
  ↪ circle
internal center/.style args = {#1,#2,#3,#4}{
  insert path = {
    let
      \p{O1A1} = ($(#2)-(#1)$),% 半径 O1A1
      \p{O2A2} = ($(#4)-(#3)$),% 半径 O2A2
      \n{r1} = {veclen(\p{O1A1})},
      \n{r2} = {veclen(\p{O2A2})},
      \n1 = {scalar(\n{r1}/(\n{r1}+\n{r2}))}
    in ($(#1)!\n1!(#3)$)
  }
},
% creates the radical axis of two non-concentric
  ↪ circles
% O1,A1: center of circle 1 and an abitary point on the
  ↪ circle
% O2,A2: center of circle 2 and an abitary point on the
  ↪ circle
radical axis/.style args = {#1,#2,#3,#4}{
  insert path = {
    let
      \n{s} = {\pgfkeysvalueof{/tikz/start modifier}},
      \n{e} = {\pgfkeysvalueof{/tikz/end modifier}},
      \p{O1A1} = ($(#2)-(#1)$),% 半径 O1A1
      \p{O2A2} = ($(#4)-(#3)$),% 半径 O2A2
      \p{O1O2} = ($(#3)-(#1)$),
```

```
      \n{r1} = {veclen(\p{O1A1})},
      \n{r2} = {veclen(\p{O2A2})},
      \n{d} = {veclen(\p{O1O2})},
      \n1 = {scalar(\n{r1}/\n{d})},
      \n2 = {scalar(\n{r2}/\n{d})},
      \n3 = {.5*(1+\n1*\n1-\n2*\n2)},
      \p{ft} = ($(#1)!\n3!(#3)$),% perpendicular foot
      \p{s0} = ($(\p{ft})+(-\y{O1O2},\x{O1O2})$),
      \p{e0} = ($(\p{ft})+(\y{O1O2},-\x{O1O2})$),
      \p{s} = ($(\p{s0})!\n{s}!(\p{e0})$),% start
      \p{e} = ($(\p{s0})!\n{e}!(\p{e0})$)% end
    in (\p{s}) -- (\p{e})
  }
},
% ========== Path Definitions ==========
% perpendicular bisector of the line segment (#1 -- #2)
perpendicular bisector/.style args = {#1,#2}{
  insert path = {
    let
      \n{s} = {\pgfkeysvalueof{/tikz/start modifier}},
      \n{e} = {\pgfkeysvalueof{/tikz/end modifier}},
      \p{AB} = ($(#2)-(#1)$),
      \p{m} = ($(#1)!0.5!(#2)$),% midpoint
      \p{s0} = ($(\p{m})+(-\y{AB},\x{AB})$),% rotate ccw, default start
      \p{e0} = ($(\p{m})+(\y{AB},-\x{AB})$),% rotate cw, default end
      \p{s} = ($(\p{s0})!\n{s}!(\p{e0})$),% start
      \p{e} = ($(\p{s0})!\n{e}!(\p{e0})$)% end
    in (\p{s}) -- (\p{e})
```

```
  }
},
% perpendicular line of the line (#1 -- #2) through #3
% specifying start and end with modifiers(see tikz
↪ manual 13.5)
perpendicular/.style args = {#1,#2,#3}{
  insert path = {
    let
      \n{s} = {\pgfkeysvalueof{/tikz/start modifier}},
      \n{e} = {\pgfkeysvalueof{/tikz/end modifier}},
      \p{AB} = ($(#2)-(#1)$),
      \p{ft} = ($(#1)!(#3)!(#2)$),% perpedicular foot
      \p{s0} = ($(\p{ft})+(-\y{AB},\x{AB})$),
      \p{e0} = ($(\p{ft})+(\y{AB},-\x{AB})$),
      \p{s} = ($(\p{s0})!\n{s}!(\p{e0})$),% start
      \p{e} = ($(\p{s0})!\n{e}!(\p{e0})$)% end
    in (\p{s}) -- (\p{e})
  }
},
% parallel line of the line (#1 -- #2) through #3
% specifying start and end with modifiers(see tikz
↪ manual 13.5)
parallel/.style args = {#1,#2,#3}{
  insert path = {
    let
      \n{s} = {\pgfkeysvalueof{/tikz/start modifier}},
      \n{e} = {\pgfkeysvalueof{/tikz/end modifier}},
      \p{s0} = (#3),
      \p{e0} = ($(#3)+(#2)-(#1)$),
      \p{s} = ($(\p{s0})!\n{s}!(\p{e0})$),% start
```

```
        \p{e} = ($(\p{s0})!\n{e}!(\p{e0})$)% end
      in (\p{s}) -- (\p{e})
    }
  },
}

% Utilities for implementation of 'revolve'
% parse comma separated arguments recursively
% store arguments number in /tikz/revolve/@argn
\def\euclidea@ParseArguments#1\euclidea@stop{%
  \pgfutil@in@{,}{#1}
  \ifpgfutil@in@%comma separated arguments
    \euclidea@ParseSeparatedArguments#1\euclidea@stop
  \fi
}

\def\euclidea@ParseSeparatedArguments#1,#2\␣
↪  euclidea@stop{%
  \pgfmathparse{int(add(\pgfkeysvalueof{/tikz/revolve/␣
  ↪  @argn},1))}
  \pgfkeysalso{/tikz/revolve/@argn = \pgfmathresult}

  \pgfutil@in@{,}{#2}
  \ifpgfutil@in@%comma separated arguments
    \euclidea@ParseSeparatedArguments#2\euclidea@stop
  \fi
}

% compute angle
\def\euclidea@ComputeAngle#1\euclidea@stop{
```

```
  \ifnum \pgfkeysvalueof{/tikz/revolve/@argn} = 1
    \tikzset{revolve/@set angle 1 = {#1}}
  \else
    \ifnum \pgfkeysvalueof{/tikz/revolve/@argn} = 2
      \tikzset{revolve/@set angle 2 = {#1}}
    \else
      \ifnum \pgfkeysvalueof{/tikz/revolve/@argn} = 3
        \tikzset{revolve/@set angle 3 = {#1}}
      \else
        \ifnum \pgfkeysvalueof{/tikz/revolve/@argn} = 4
          \tikzset{revolve/@set angle 4 = {#1}}
        \else
          \pgferror{"Incorrect number of arguments!"}
        \fi
      \fi
    \fi
  \fi
}

% Utilities for implementation of 'intercept'
\def\euclidea@ComputeLength#1,#2\euclidea@stop{
  \newdimen\euclidea@ax
  \newdimen\euclidea@ay
  \newdimen\euclidea@bx
  \newdimen\euclidea@by
  \pgfextractx{\euclidea@ax}{\pgfpointanchor{#1}{␣
center}}
  \pgfextracty{\euclidea@ay}{\pgfpointanchor{#1}{␣
center}}
```

```
  \pgfextractx{\euclidea@bx}{\pgfpointanchor{#2}{␣
center}}
  \pgfextracty{\euclidea@by}{\pgfpointanchor{#2}{␣
center}}
  % 以下 showthe 指令 overleaf.com 编译通过，而在
   ↪  macOS+texlive 2021 报错
  % \showthe\euclidea@ax
  % \showthe\euclidea@ay
  % \showthe\euclidea@bx
  % \showthe\euclidea@by
  \pgfmathveclen{\euclidea@ax-\euclidea@bx}{\␣
   ↪  euclidea@ay-\euclidea@by}
}

\makeatother
```

参考文献

[1] Syntax for path specifications. https://tikz.dev/tikz-paths.

[2] Homothetic center. https://en.wikipedia.org/wiki/Homothetic_center.

[3] Coordinate calculations. https://tikz.dev/tikz-coordinates#sec-13.5.

[4] Line-line intersection. https://mathworld.wolfram.com/Line-LineIntersection.html.