Revista digital
Matemática, Educación e Internet
(https://tecdigital.tec.ac.cr/revistamatematica/).
Vol 20, No 1. Setiembre – Febrero, 2020

Artículo de sección
ISSN 1659 -0643

Construcciones geométricas con el paquete tkz-euclide

Geometric constructions with tkz-euclide software

Reiman Y. Acuña Chacón
reiacuna@itcr.ac.cr
Escuela de Matemática
Instituto Tecnológico de Costa Rica - Costa Rica

Estibaliz O. Rojas Quesada
erojasq@uned.ac.cr
Encargada de Cátedra Matemáticas Básicas
Universidad Estatal a Distancia - Costa Rica

Recibido: 3 octubre 2018 Aceptado: 26 junio 2019

Resumen. El paquete tkz-euclide.sty es un conjunto de macros especializadas en LaTeX para realizar construcciones geométricas en dos dimensiones. Está construido sobre PGF (Portable Graphics Format) y su interfaz TikZ. En este documento se explica el uso básico de este paquete así como la realización de algunas construcciones complejas con el fin de que docentes universitarios y de secundaria puedan elaborar material de alta calidad para sus trabajos personales y académicos.

Palabras clave: tkz-euclide, construcciones geométricas, LaTeX

Abstract. The tkz-euclide.sty package is a set of specialized macros in LaTeX to make geometrical constructions in two dimensions. It is built on PGF (Portable Graphics Format) and its TikZ interface. This document explains the basic use of this package as well as the realization of some complex constructions with the purpose of university and secondary teachers for high quality material for their personal and academic works.

KeyWords: tkz-euclide, geometric constructions , LaTeX

1.1 Introducción

El 3 de junio del año 2011, el francés Alain Matthes publica la documentación sobre el paquete **tkz-euclide**, el cual permite crear dibujos relacionados con la geometría en dos dimensiones. Los argumentos del autor para generar este paquete han sido los siguientes:

1. Algunos usuarios no quieren aprender nada sobre **TikZ**[1], lo cual es respetable y conlleva a simplificar el código. La sintaxis no es más como **TikZ**, pero más como LaTeX. (Ver [4] para mayor información)

2. Los nombres de las macros tienen un significado más matemático y pertinente con la construcción.

3. La gran diferencia con **TikZ** es que es posible implementar cálculos mediante el uso del paquete **fp**[2].

4. Es posible modificar fácilmente los estilos para los objetos principales; que son los puntos, las líneas, círculos, arcos, etc.

5. Es un paquete ideado para los principiantes, sin que estos tengan que conocer el uso del paquete **TikZ**.

Con los argumentos, el autor expone que es posible mezclar los códigos de los paquetes **TikZ** y **tkz-euclide** y que ello puede facilitar las construcciones. No obstante, el principio inicial es facilitar una herramienta flexible y voluble para los principiantes que quieran adentrarse al mundo de las construcciones geométricas y les permita entender lo que están escribiendo.

1.2 Sintaxis

Con base en [1] y [2], para el uso de este paquete se necesita conocer algunos comandos básicos que permiten generar objetos primitivos, es decir, los puntos, rectas, círculos entre otros. En general, todos los comandos inician con `\tkz...` **y no se colocan punto y coma al final de la línea**, (como pasa en **TikZ**) solo se cambia de entrada. Para empezar, cualquier documento en el preámbulo debe contener las siguientes líneas
La primera línea carga el paquete y la segunda carga todas las herramientas que trae el paquete, como círculos, arcos, transportador, entre otros. Luego, se hace uso del ambiente **tikzpicture** para hacer las construcciones respectivas. Las siguientes subsecciones ejemplifican parte de este proceso.

Puntos

La creación de puntos parte de una idea geométrica: se definen y luego se representan. Las coordenadas que se usan son cartesianas como polares. La sintaxis implementada es la siguiente

- `\tkzDefPoint(a,b){A}`: Se define un punto con nombre A en las coordenadas cartesianas (a,b).
- `\tkzDefPoints{a1/b1/A1, a2/b2/A2, . . ., an/bn/An}` se definen una cantidad finita de puntos con nombres A_1, A_2, ..., A_n en las coordenadas cartesianas (a_1,b_1), (a_2,b_2), ..., (a_n,b_n) respectivamente.
- `\tkzDefShiftPoint[A](a,b){B}`: Se define el punto B con respecto al punto A con respecto a las coordenadas cartesianas (a,b)

[1]Este es un paquete que representa un conjunto de macros de alto nivel que utiliza PGF. Till Tantau es el diseñador de estos lenguajes, y también el desarrollador principal, los cuales están escritos en TeX.
[2]EL paquete **fp** (fixed point) es un paquete que realiza cálculos en LaTeX con redondeo o truncamiento, definidos por el usuario. En este caso, el paquete **tkz-eulide** consolida los cálculos con este paquete automáticamente.

- `\tkzDrawPoint(a,b){A}`: Se dibuja un punto con nombre A en las coordenadas cartesianas (a,b).
- `\tkzLabelPoints[condiciones](A)`: Se etiqueta un punto A con condiciones dadas. Estas condiciones pueden ser color o posición de la etiqueta.

Es importante indicar que en todos los casos donde se consideran las coordenadas cartesianas (a,b), estas se puede cambiar a las coordenadas polares $(\theta : r)$, donde θ hace referencia al ángulo en grados y r al radio en puntos (Aquí un centímetro equivale a 28.3465 puntos aproximadamente) .

Un ejemplo del uso de estos comandos lo vemos en el siguiente código, cuyo resultado se muestra en la figura 1.1

```
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDrawPoint(0,0){A}
\tkzLabelPoints[below](A)
\end{tikzpicture}
```

Figura 1.1: Representación de un punto

Note que el ambiente sigue siendo **TikZ** pero el código utilizado es diferente: primero **se define** el par ordenado $(0,0)$ como el punto A a través del comando `\tkzDefPoint(0,0){A}`; luego **se dibuja** el punto con el comando `\tkzDrawPoint(0,0){A}`; por último **se coloca** la etiqueta del punto. Se puede etiquetar sin definir, o dibujar sin definir. Eso va depender del tipo de construcción a realizar[3]. Si se desea representar más de un punto, se puede usar el siguiente código, cuyo resultado se muestra en la figura 1.2

```
\begin{tikzpicture}
\tkzDefPoints{0/0/O, 3/0/A, 1.5/1.5/B,  3.5/1.5/C}
\tkzDrawPoints(O,A,B)
\tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

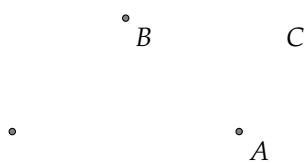


Figura 1.2: Representación de varios puntos

Note que aquí se definieron cuatro puntos, se marcaron los puntos O, A y B y se etiquetaron los puntos A, B y C. También se puede cambiar el color y el grosor de los puntos. Esto se verá en próximos

[3]Por ejemplo, puede suceder que solo se necesite hacer una construcción donde se muestre el resultado final. En tal caso, los puntos auxiliares para generar la construcción se mantienen ocultos a través de la definición, pero no se dibujan. Solo son "andamios" en dicho proceso

ejemplos, así como la ubicación y posición de las etiquetas.

Segmentos y Rectas

Definiendo los puntos, es posible tener segmentos o rectas. Para ello se usa el comando `\tkzDrawSegment` y `\tkzDrawLine` respectivamente. En tal caso, la sintaxis que se utiliza es la siguiente

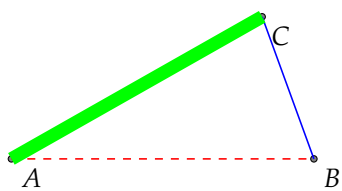
- `\tkzDrawSegment[opciones](A,B)`: Se dibuja el segmento entre los puntos A y B. Las opciones pueden ser color, el tipo de línea o el grosor, y cada opción se separa por una coma. Por ejemplo, `color=red` genera un segmento en color rojo. El comando `dashed` proporciona una línea segmentada. Por último,
  `line width= 5pt` define un segmento o recta con un grosor de 5 pt.

- `\tkzDrawLine[opciones](A,B)`: Se dibuja una línea que pasa por los puntos A y B. Las opciones pueden ser incluir flechas o extender la extensión de la línea, y cada opción se separa por una coma. Por ejemplo, `arrows= triangle 60 - latex'` agrega antes del guión, una flecha a la izquierda con forma de triángulo equilátero y, luego del guión, un estilo de flecha refinada. Esto es permitido siempre y cuando se cargue en libreria de **TikZ** la opción `arrows` (en el preámbulo del documento[4]). Por último, la opción `add=.3 and .4` extiende la longitud de la recta (hacia la izquierda) en un 30% de la medida de la longitud entre los puntos A y B; y de forma similar se extiende la longitud de la recta (hacia la derecha) en un 40% de la medida de la longitud entre los puntos A y B;

Cabe destacar la versatilidad del comando `\tkzLabelSegment[condiciones](A,B){ejemplo}`, el cual ubica la etiqueta *ejemplo* sobre el segmento que pasa por los puntos A y B. En las condiciones se puede indicar la posición, por ejemplo `above` (arriba) o `below` (abajo). Un ejemplo ilustrativo del posible uso de estos comandos es el siguiente:

```
\begin{tikzpicture}
% Puntos
\tkzDefPoint(0,0){A}
\tkzDefPoint(0:4){B}
\tkzDefShiftPoint[B](110:2){C}
\tkzDrawPoints[color=black](A,B,C)
\tkzLabelPoints(A,B,C)
% Segmentos
\tkzDrawSegment[color=red,dashed](A,B)
\tkzDrawSegment[color=blue](B,C)
\tkzDrawSegment[color=green, line width= 5pt](C,A)
\end{tikzpicture}
```

Con lo cual se obtiene la figura 1.3.



[4] `\usetikzlibrary{librerias}`

Figura 1.3: Definición de segmentos

En este ejemplo, los segmentos se dibujan a partir de las posiciones de los puntos brindados. En el caso que deseemos dibujar una recta tendremos el siguiente ejemplo

```
\begin{tikzpicture}
% Puntos
\tkzDefPoints{0/2/A, 8/3/B}
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
% Línea
\tkzDrawLine[arrows= triangle 60 - latex', add=.2 and .2 ](A,B)
\end{tikzpicture}
```

Cuyo resultado es la figura 1.4

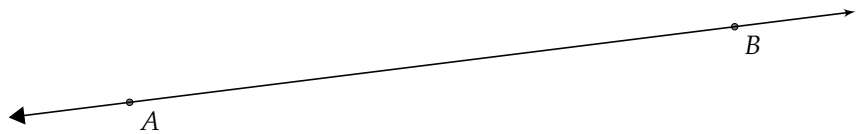


Figura 1.4: Definición de recta

Círculos, Arcos y Sectores

Para dibujar círculos, usamos el comando `\tkzDrawCircle`. En el caso de los arcos y sectores tenemos los comandos `\tkzDrawArc` y `\tkzDrawSector`. En tal caso, la sintaxis que se utiliza es la siguiente:

- `\tkzDrawCircle[R,opciones](A,p cm)`: Se dibuja un círculo centrado en el punto A con radio de p cm. Las opciones pueden ser color u opacidad. Cada condición se separa por una coma. En este caso la opción `R` indica que es un círculo caracterizado por un punto y un radio. El comando `fill=gray` indica que el círculo se rellena de color gris (para el borde ver la descripción siguiente). El comando `opacity=.1` indica que el color que se use tenga una opacidad del 10%.
- `\tkzDrawArc[opciones](A,E)(F){arco}`: Se dibuja un arco, cuyo centro es el punto A, el arco empieza en E y termina en F. En las opciones se le puede agregar color al borde con el comando `color=orange`, el cual pinta el arco del círculo en tono naranja para el caso que se use.
- `\tkzDrawSector[opciones](A,E)(F){arco}`: Se dibuja un sector, cuyo centro es el punto A, el sector empieza en E y termina en F. Se pueden usar las mismas opciones que las descritas para el comando `\tkzDrawArc[opciones](A,E)(F){arco}`.

Un ejemplo ilustrativo de su uso es el siguiente:

```
\begin{tikzpicture}
% Puntos
\tkzDefPoint(0,0){A}
\tkzDefPoint(0:3){B}
\tkzDefPoint(100:3){E}
\tkzDefPoint(150:3){F}
\tkzDrawPoints(A,B,E,F)
% Círculo
```

```
\tkzDrawCircle[R,fill=gray, opacity=.1](A,3cm)
% Arco
\tkzDrawArc[color=red,line width= 5pt](A,E)(F){arco}
% Sector
\tkzDrawSector[fill=yellow, opacity=0.5](A,E)(F)
% Etiqueta de Puntos
\tkzLabelPoints[above left](B,E,F)
\tkzLabelPoints[below left](A)
% Segmento
\tkzDrawSegment[color=black](A,B)
\tkzLabelSegment[above](A,B){radio}
\end{tikzpicture}
```

El cual queda representado en la figura 1.5

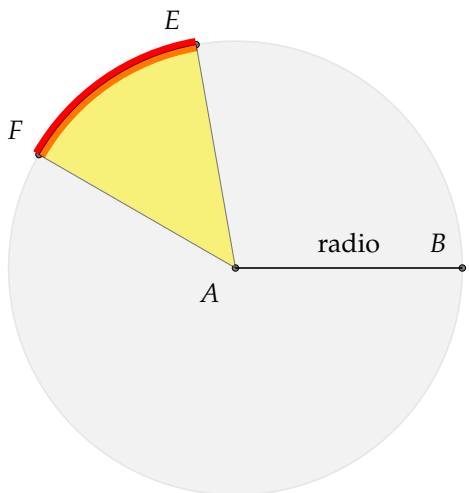


Figura 1.5: Círculo, arco y sector

Se debe notar que se usaron coordenadas polares para que fuera más fácil la posición de los puntos sobre la circunferencia. Por otro lado, note que, para la colocación de etiquetas se hizo uso de las opciones `above` (arriba) y `left` (izquierda) para la ubicación más precisa de las mismas. Se pueden hacer combinaciones de estas posiciones o usar los comandos `xshift` y `yshift` para hacer desplazamientos horizontales y verticales, respectivamente, de las etiquetas.

Observación: En caso de que lo que se desea es solo rellenar el sector, debe emplearse el comando `\tkzFillSector`. Puede cambiar la línea donde se dibujó el sector y usar este comando.

Ángulos

Los ángulos se representan con base en su definición matemática: el espacio definido entre dos rayos y un vértice. En tal caso, se utilizan dos comandos: `\tkzMarkAngle` y `\tkzLabelAngle`. El primero define el relleno del ángulo y el segundo la etiqueta que se quiere colocar sobre el mismo. Un ejemplo es el siguiente:

```
\begin{tikzpicture}[scale=1.25]
% Puntos
```

```
\tkzDefPoints{-1.08/2.77/A, -2.94/2.6/O -2.38/4.9/B}
\tkzDrawPoint[size=10,color=red,fill=white](O)
\tkzDrawPoint[shape=cross out,size=10,color=blue](A)
\tkzDrawPoint[shape=cross out,size=10,color=blue](B)
\tkzLabelPoints[below left](O)
\tkzLabelPoints[xshift=-10, yshift=-3](A,B)
% Rayos
\tkzDrawLines[arrows=-triangle 45, add=0 and .2](O,A O,B)
% Ángulo
\tkzMarkAngle[fill= yellow,size=0.8, mark=||](B,O,A)
\tkzLabelAngle[pos=1.2](B,O,A){$m$}
\end{tikzpicture}
```

El cual queda representado en la figura 1.6

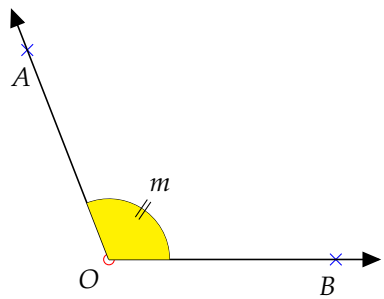


Figura 1.6: Representación de un ángulo

Observamos que

1. `\tkzMarkAngle[opciones](B,O,A)`: marca el ángulo que inicia en B, con vértice en O y termina en A. Entre las opciones se puede rellenar con algún color, cambiar el tamaño y hacerle una marca particular, como se aprecia en el ejemplo.
2. `\tkzLabelAngle[pos=número](B,O,A){$m$}`: permite colocar una etiqueta el ángulo que inicia en B, con vértice en O y termina en A. Se le coloca la letra "m" en una posición de acuerdo al valor de *número*.

En este caso, se debe notar que se ha escalado el tamaño del ángulo con el comando `scale=1.25` al inicio del ambiente. Se destaca también la manera natural en que se define el ángulo y en la cual se procede a rellenarlo.

De igual forma se pueden trazar diferentes ángulos desde un lado inicial modificando el comando `size`. Esto se puede visualizar en la figura 1.7, donde se han representado diferentes ángulos con lado inicial OB.

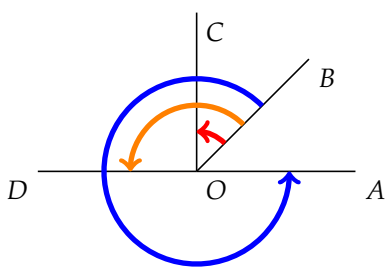


Figura 1.7: Trazado de diferentes ángulos

El código implementado ha sido el siguiente. (Se omite la explicación de la creación de los puntos y segmentos)

```
\begin{tikzpicture}[scale=1]
% Puntos
\tkzDefPoint(0,0){O}
\tkzDefPoint(0:3){A}
\tkzDefPoint(45:3){B}
\tkzDefPoint(90:3){C}
\tkzDefPoint(180:3){D}
% Segmentos
\tkzDrawSegments[color=black](O,A O,B O,C O,D)
\tkzLabelPoints(O,A,B,C)
\tkzLabelPoints[below left](D)
% Ángulos
\tkzMarkAngle[color=red,line width=2pt,size=0.75cm,arrows=->](B,O,C)
\tkzMarkAngle[color=orange,line width=2pt,size=1.25cm,arrows=->](B,O,D)
\tkzMarkAngle[color=blue,line width=2pt,size=1.75cm,
opacity=1,arrows=->](B,O,A)
\end{tikzpicture}
```

Por último, existe el comando `\tkzShowLine[bisector, opciones](B,A,C)` el cual muestra los trazos del ángulo bisector BAC de acuerdo con las opciones dadas. Por ejemplo, `size=4,gap=7,length=3` define el tamaño del ángulo bisector en en 4 pt y la lejanía de la intersección en 7 pt. La última opción indica que la longitud del trazo sea de 3 pt.

Trazado de Polígonos

El trazado de polígono es sumamente sencillo. El comando a utilizar es `\tkzDrawPolygon`. Para hacer uso del comando, solo se necesitan definir (ni siquiera trazar) tres puntos como mínimo. Luego se indican los puntos por los cuales se debe hacer el trazo. Un ejemplo se muestra en la figura 1.8

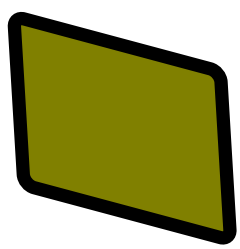


Figura 1.8: Trazado de un polígono

El código implementado ha sido el siguiente:

```
\begin{tikzpicture}[rotate=30,scale=1.5]
% Puntos
\tkzDefPoints{0/0/A, 2/-2/B, 3/0/C, 1/2/D}
% Polígono
\tkzDrawPolygon[fill=red!50!green,
```

```
line width=5pt,rounded corners](A,B,C,D)
% Se define el polígono con vértices en A, B, C y D.
% Se rellena con una mezcla de rojo a verde en un 50%
% El grosor de la línea es de 5pt
% Las esquinas han sido redondeadas
\end{tikzpicture}
```

Note que la figura se rotó 30 grados y se escaló 1.25 veces de su tamaño original. En general, tenemos que

1. `\tkzDrawPolygon[opciones](A1,A2,...,An)`: Permite definir un polígono con una cantidad finita de puntos con nombres $A_1, A_2, \ldots, A_n$ en coordenadadas cartesianas o polares previamente definidas. Entre las opciones podemos dar color al borde o al relleno (por ejemplo `fill=red!50!green`), cambiar el grosor al borde (por ejemplo `line width=5pt`) o generar esquinas redondeadas (por ejemplo `rounded corners`).

Otros comandos de importancia

Hasta este punto, se ha ejemplificado cómo se procede en la creación de objetos de mayor importancia que materializa el paquete. No obstante, es posible articular sobre las construcciones algunos comandos de carácter operacional.

Definición extendida de puntos

Se puede hacer uso de los comandos `\tkzDefPointBy` y `\tkzDefPointWith` para definir puntos por medio de transformaciones o propiedades vectoriales respectivamente.

Los comandos `\tkzInterLL`, `\tkzInterLC` y `\tkzInterCC` se pueden usar como generadores de puntos como resultado de que se intersequen dos rectas, un círculo y una recta; o dos círculos respectivamente.

En cualquiera de los casos anteriores se pueden obtener los puntos mediante el comando `\tkzGetPoint{M}` para obtener el punto "M" por ejemplo, o `\tkzGetFirstPoint{M}` y `\tkzGetFirstPoint{N}` si hay claridad de que se obtienen dos puntos "M" y "N". Un ejemplo alusivo es el siguiente

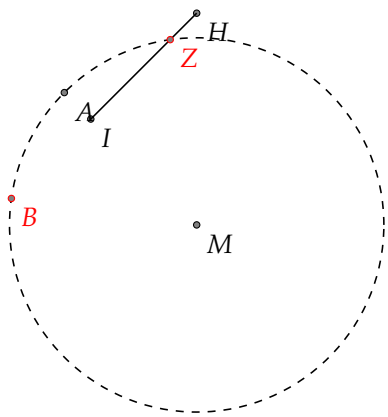


Figura 1.9: Definición extendida de puntos

Cuyo código es:

```
\begin{tikzpicture}[scale=1]
% Puntos
\tkzDefPoints{0/0/I, 2/-2/M, -0.5/0.5/A}
\tkzDrawPoints(I,M,A)
\tkzLabelPoints(I,M,A)
% Extensión
\tkzDefPointWith[orthogonal](I,M) \tkzGetPoint{H}
\tkzDrawPoints(H)
\tkzLabelPoints(H)
% Segmento
\tkzDrawSegment(I,H)
% Círculo
\tkzDrawCircle[dashed](M,A)
% Extensión 2
\tkzInterLC(I,H)(M,A) \tkzGetFirstPoint{B} \tkzGetSecondPoint{Z}
\tkzDrawPoints[color=red](B,Z)
\tkzLabelPoints[color=red](B,Z)
\end{tikzpicture}
```

De acuerdo con el ejemplo tenemos los siguientes comandos:

1. `\tkzDefPointWith[orthogonal](I,M) \tkzGetPoint{H}`: Se define un punto ortogonal al segmento que contiene a los puntos I e M. La ortogonal se traza en I. Se obtiene el punto H.
2. `\tkzInterLC(I,H)(M,A) \tkzGetFirstPoint{B} \tkzGetSecondPoint{Z}`: Se definen los puntos B y Z como resultado de la intersección de la recta que contiene a los puntos I e H con el círculo centrado en M de radio MA.

Otros comandos de interés, a los cuales se les pueden combinar `\tkzGetPoint` son:

1. **Punto Medio**: El comando `\tkzDefMidPoint(A,B)` define el punto medio entre A y B.
2. **Coordenadas Baricéntricas**: El comando

   `\tkzDefBarycentricPoint(A1=número1,A2=número2,A3=número3,...,AN=númeroN)`

   define las coordenadas baricéntricas con respecto a los puntos $A1$, $A2$, $A3$,...,AN. Intuitivamente, se desea encontrar la posición de un punto en una razón dada con respecto a los puntos considerados. Por ejemplo, si se desea encontrar un punto I tal que

$$\overrightarrow{AI} = \frac{3}{5}\overrightarrow{AB}$$

   debe cumplirse que

$$\begin{aligned}\overrightarrow{AI} = \frac{3}{5}\overrightarrow{AB} &\Leftrightarrow I - A = \frac{3}{5}(B - A)\\ &\Leftrightarrow I = A + \frac{3}{5}B - \frac{3}{5}A\\ &\Leftrightarrow I = A - \frac{3}{5}A + \frac{3}{5}B\\ &\Leftrightarrow I = \frac{2}{5}A + \frac{3}{5}B\end{aligned}$$

Con lo cual las coordenadas baricéntricas serán (2,3) y el comando asociado será :

```
\tkzDefBarycentricPoint(A=2,B=3) \tkzGetPoint{I}
```

que definiría al punto I a tres quintos del vector AB, como se muestra en la imagen 1.10

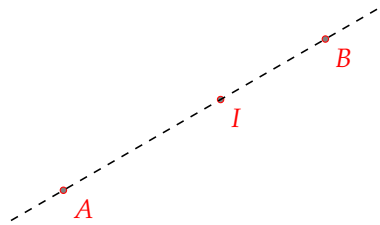


Figura 1.10: Coordenadas Baricéntricas con dos puntos

Para el caso de tres puntos, si todos se igualan a la unidad, representará el centro de gravedad o baricentro de un triángulo por ejemplo. Un estudio más exhaustivo lo puede encontrar en [3].

3. **Baricentro**: El comando `\tkzCentroid(A,B,C)` define el baricentro con los puntos A, B y C.

4. **Circuncentro**: El comando `\tkzCircumCenter(A,B,C)` define el circuncentro con los puntos A, B y C.

5. **Incentro**: El comando `tkzInCenter(A,B,C)` define el incentro con los puntos A, B y C.

6. **Ortocentro**: El comando `\tkzOrthoCenter(A,B,C)` define el ortocentro con los puntos A, B y C.

7. **Punto Arbitrario**: El comando `\tkzGetRandPointOn[opcion]{A}` genera un punto arbitrario de nombre "A" sobre la opción. La opción puede ser un rectángulo (`P and Q`), segmento (`P -- Q`), línea (`P -- Q`) o círculo (`center P radius 3cm`).

8. **Puntos como intersección entre rectas:** El comando `\tkzInterLL(A,B)(P,Q) \tkzGetPoint{X}` permite definir el punto X como la intersección entre las rectas AB y PQ. En caso de que las rectas no se intersequen se genera un error.

9. **Puntos como intersección de círculos:** El comando `\tkzInterCC(A,L)(C,D) \tkzGetFirstPoint{E} \tkzGetSecondPoint{F}` permite definir los puntos E y F como resultado de la intersección del círculo centrado en A con radio AL y del círculo centrado en C con radio CD. En caso de círculos tangentes solo se produce el punto E. En caso de que los círculos no se intersequen se genera un error.

10. **Altura y Punto:** El comando `\tkzDrawAltitude[opciones](A,B)(C) \tkzGetPoint{H}` permite dibujar la altura sobre el segmento AB pasando por el punto C. Se obtiene de paso el punto H. Entre las opciones se puede dar color, grosor, flechas y más. Por ejemplo las opciones

```
arrows=triangle 45-triangle 45,add=0.3 and 0.3
```

definiría una altura con flechas formadas por triángulos con ángulo agudo en la punta de 45 grados, extendiendo la altura en un 30% (a ambos lados) del segmento HC.

11. **Mediana y Punto:** El comando `\tkzDrawMedian[opciones](A,B)(C) \tkzGetPoint{M}` permite dibujar la mediana sobre el segmento AB pasando por el punto C. Se obtiene de paso el punto M. Entre las opciones se puede dar color, grosor, flechas y más.

12. **Bisectriz y Punto:** El comando `\tkzDrawBisector[opciones](A,C,B) \tkzGetPoint{D}` permite dibujar la recta bisectriz para el angulo ACB con vértice en C. Se obtiene de paso el punto D. Entre las opciones se puede dar color, grosor, flechas y más.

Un ejemplo ilustrativo donde se usan estos comandos es el siguiente:

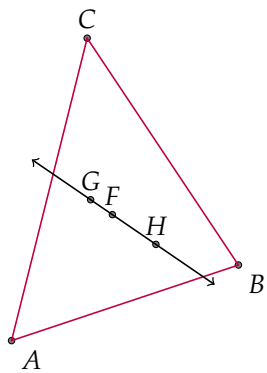


Figura 1.11: Recta de Euler

En la figura 1.11, se muestra la recta de Euler, la cual pasa por el **circuncentro**, **el incentro** y el **ortocentro**. El código implementado fue el siguiente:

```
\begin{tikzpicture}[scale=0.5]
\clip(-3,-2) rectangle (7,9);
\tkzDefPoint(0,0){A}
\tkzDefPoint(6,2){B}
\tkzDefPoint(2,8){C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above](C)
\tkzDrawPolygon[color=purple](A,B,C)
% Puntos Especiales
\tkzCircumCenter(A,B,C) \tkzGetPoint{G}
\tkzCentroid(A,B,C) \tkzGetPoint{F}
\tkzOrthoCenter(A,B,C) \tkzGetPoint{H}
\tkzDrawPoints(F,G,H)
\tkzLabelPoints[above](F,G,H)
% Recta de Euler
\tkzDrawLines[arrows=<->,add = .9 and .9](G,H)
\end{tikzpicture}
```

De acuerdo con el ejemplo, tenemos los siguientes comandos:

1. `\clip (x1,y1) rectangle (x2,y2);`: recorta y solo muestra la figura en el rectángulo de coordenadas cartesianas especificadas. En tal caso (x_1,y_1) representa la esquina inferior izquierda y (x_2,y_2) representa la esquina superior derecha.

2. `\tkzCircumCenter(A,B,C) \tkzGetPoint{G}`: Se define el punto G como el circuncentro para los puntos A,B y C.

3. `\tkzCentroid(A,B,C) \tkzGetPoint{F}`: Se define el punto F como el baricentro para los puntos A,B y C.

4. `\tkzOrthoCenter(A,B,C) \tkzGetPoint{H}`: Se define el punto H como el ortocentro para los puntos A,B y C.

Uso de compás

Puede ser interés del lector hacer hincapié en la marcas del compás cuando desea, por ejemplo, hacer una construcción con regla y compás paso a paso. Para ello, el comando asociado será:

```
\tkzCompass[opciones locales](A,B)
```

Donde `A` representa el punto inicial y `B` representa el punto final donde se trazará la marca. Es claro que la medida de AB representa el radio del círculo centrado en A con punto extremo en B, pero no se desea el trazo de todo el círculo. Para entender el funcionamiento del compás, se trazará la construcción de un triángulo equilátero paso a paso. Esto es:

1. Trazar un segmento con extremos A y B, como se muestra en figura 1.12

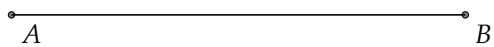


Figura 1.12: Base Triángulo Equilátero

```
\begin{tikzpicture}
\tkzDefPoints{-3/0/A, 3/0/B}
\tkzDrawPoints[color=black](A,B)
\tkzDrawSegment[](A,B)
\tkzLabelPoints[black](A,B)
\end{tikzpicture}
```

2. Trazar las marcas definidas por el círculo con centro en A y radio AB y, de igual forma, las marcas definidas por el círculo con centro en B y radio BA (figura 1.13)

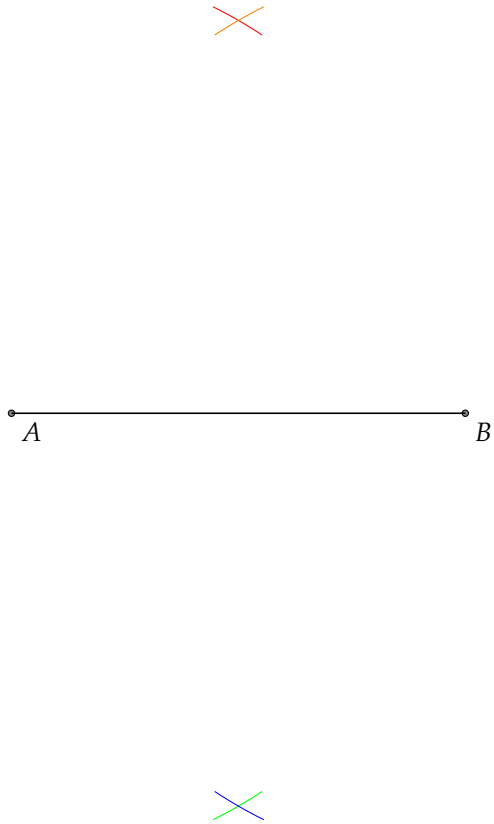


Figura 1.13: Primera Marca Triángulo Equilátero

```
\begin{tikzpicture}
\tkzDefPoints{-3/0/A, 3/0/B}
\tkzDrawPoints[color=black](A,B)
\tkzDrawSegment(A,B)
\tkzLabelPoints[black](A,B)
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{C}
\tkzGetSecondPoint{D}
\tkzCompass[color=red](A,C)
\tkzCompass[color=orange](B,C)
\tkzCompass[color=green](A,D)
\tkzCompass[color=blue](B,D)
\end{tikzpicture}
```

3. Definir y dibujar los puntos generados por las intersección de estas marcas (producto de los círculos definidos) (figura 1.14)

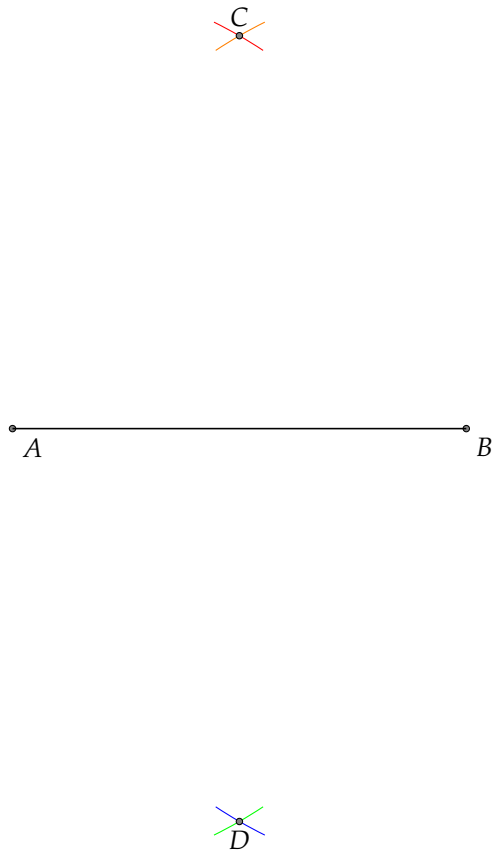


Figura 1.14: Primera Marca Triángulo Equilátero

```
\begin{tikzpicture}
\tkzDefPoints{-3/0/A, 3/0/B}
\tkzDrawPoints[color=black](A,B)
\tkzDrawSegment(A,B)
\tkzLabelPoints[black](A,B)
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{C}
\tkzGetSecondPoint{D}
\tkzCompass[color=red](A,C)
\tkzCompass[color=orange](B,C)
\tkzCompass[color=green](A,D)
\tkzCompass[color=blue](B,D)
\tkzDrawPoints[color=black](C,D)
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[below](D){$D$}
\end{tikzpicture}
```

4. Se trazan los segmentos AC y BC para formar el triángulo equilátero. (figura 1.15)

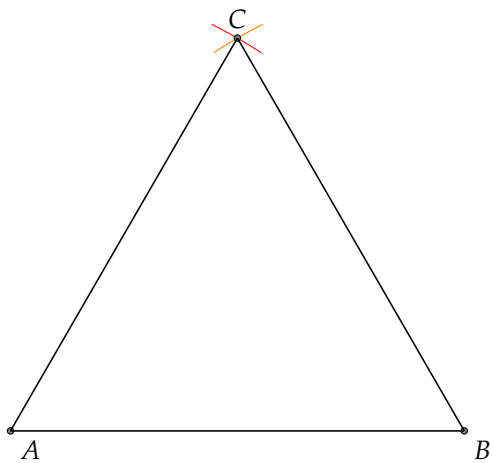


Figura 1.15: Triángulo Equilátero Listo

```
\begin{tikzpicture}
\tkzInit[xmax=5, ymax=6 , xmin=-5, ymin=-1 ] % se define una zona de trabajo
\tkzClip % se procede al recorte de la zona de trabajo
\tkzDefPoints{-3/0/A, 3/0/B}
\tkzDrawPoints[color=black](A,B)
\tkzDrawSegment(A,B)
\tkzLabelPoints[black](A,B)
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{C}
\tkzGetSecondPoint{D}
\tkzCompass[color=red](A,C)
\tkzCompass[color=orange](B,C)
\tkzCompass[color=green](A,D)
\tkzCompass[color=blue](B,D)
\tkzDrawPoints[color=black](C,D)
\tkzLabelPoint[above](C){C}
\tkzLabelPoint[below](D){D}
\tkzDrawSegment(A,C)
\tkzDrawSegment(B,C)
\end{tikzpicture}
```

Observe el uso de los comandos

```
\tkzInit[xmax=5, ymax=6 , xmin=-5, ymin=-1 ]
```

y

```
\tkzClip
```

El primero define la zona de trabajo y el segundo el aplica el recorte sobre dicha zona. Esto se ha hecho con el fin de eliminar el punto D de la zona visible de construcción. Un comando de importancia es `\tkzSetUpCompass[opciones]`, el cual define opciones principales para el suo del compás, como el color,

grosor y estilo de marca. Por ejemplo, las opciones `color=brown,line width=2 pt,style=dashed` indican que el compás se marque de color café, con un grosor de 2pt y un estilo de línea punteada. La siguiente figura muestra el uso de este comando

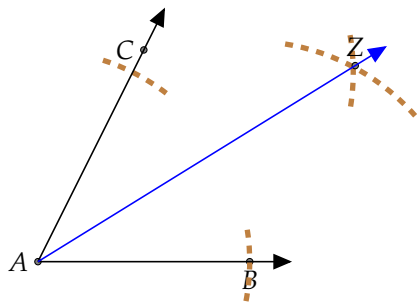


Figura 1.16: Bisección de un ángulo

```
\begin{tikzpicture}[scale=0.7]
\tkzSetUpCompass[color=brown,line width=2 pt,style=dashed]
\tkzDefPoints{0/0/A, 4/0/B, 2/4/C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints[left](A,C)
\tkzLabelPoint[below](B){$B$}
\tkzDefLine[bisector](B,A,C) \tkzGetPoint{Z}
\tkzShowLine[bisector,size=4,gap=7,length=3](B,A,C)
\tkzDrawLines[arrows=-triangle 45, add=0 and .2](A,C A,B)
\tkzDrawPoint(Z)
\tkzLabelPoint[above](Z){$Z$}
\tkzDrawLines[arrows=-triangle 45, add=0 and 0.1,color=blue](A,Z)
\end{tikzpicture}
```

1.3 Ejemplos de Construcciones Geométricas

Con base en la sección anterior, es posible generar diversas construcciones geométricas. Las mismas han sido obtenidas de la información brindada en [5], con lo cual algunas se generaran paso a paso; otras serán presentadas de manera terminada.

Localizar el punto medio de un segmento dado

1. Sea AB un segmento dado

```
\begin{tikzpicture}[scale=1]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{0/0/A, 4/0/B}
\tkzDrawPoints(A,B)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzDrawSegment(A,B)
\end{tikzpicture}
```

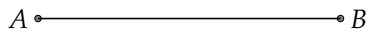


Figura 1.17: Segmento AB

2. Construir un círculo centrado en A con radio AB y otro centrado en B con radio BA. Sean C y D los puntos de intersección de estos círculos.

```
\begin{tikzpicture}[scale=1]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{0/0/A, 4/0/B}
\tkzDrawPoints(A,B)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzDrawSegment(A,B)
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{C}
\tkzGetSecondPoint{D}
\tkzCompass[](A,C)
\tkzCompass[](B,C)
\tkzCompass[](A,D)
\tkzCompass[](B,D)
\tkzDrawPoints[color=black](C,D)
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[below](D){$D$}
\end{tikzpicture}
```

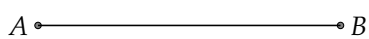


Figura 1.18: Segmento AB

3. Conectar CD. Sea M el punto de intersección de CD y AB. Entonces M es el punto medio del segmento AB.

```
\begin{tikzpicture}[scale=1]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{0/0/A, 4/0/B}
\tkzDrawPoints(A,B)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzDrawSegment(A,B)
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{C}
\tkzGetSecondPoint{D}
\tkzCompass[](A,C)
\tkzCompass[](B,C)
\tkzCompass[](A,D)
\tkzCompass[](B,D)
\tkzDrawPoints[color=black](C,D)
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[below](D){$D$}
\tkzDrawSegment(C,D)
\tkzInterLL(A,B)(C,D)
\tkzGetPoint{M}
\tkzDrawPoints[color=blue](M)
\tkzLabelPoint[color=blue, above right](M){$M$}
\end{tikzpicture}
```

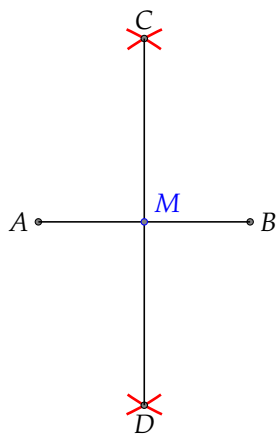


Figura 1.19: Punto Medio M

Localizar el círculo que pasa por tres puntos dados

1. Sean A, B y C tres puntos tales que la recta AB no pase por C.

```
\begin{tikzpicture}[scale=0.6]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{-3/-1/A, 2/0/B,  0/3/C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoint[left](A){$A$}
```

```
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\end{tikzpicture}
```

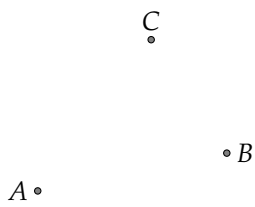


Figura 1.20: Los puntos A, B y C

2. Sabiendo como encontrar el punto medio de un segmento dado, (subsección anterior) se construyen las mediatrices PQ y RS de los segmentos AB y BC respectivamente.

```
\begin{tikzpicture}[scale=0.6]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{-3/-1/A, 2/0/B,  0/3/C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzDrawSegments(A,B B,C)
% Primera Mediatriz
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{P}
\tkzGetSecondPoint{Q}
\tkzCompass[](A,P)
\tkzCompass[](B,P)
\tkzCompass[](A,Q)
\tkzCompass[](B,Q)
\tkzDrawPoints[color=black](P,Q)
\tkzLabelPoint[above](P){$P$}
\tkzLabelPoint[below](Q){$Q$}
\tkzDrawSegment[dashed](P,Q)
\tkzInterLL(A,B)(P,Q)
\tkzGetPoint{X}
\tkzDrawPoints[color=blue](X)
%\tkzLabelPoint[color=blue, above right](X){$X$}
% Segunda Mediatriz
\tkzInterCC(C,B)(B,C)
\tkzGetFirstPoint{R}
\tkzGetSecondPoint{S}
\tkzCompass[](C,R)
\tkzCompass[](B,R)
\tkzCompass[](B,S)
\tkzCompass[](C,S)
\tkzDrawPoints[color=black](R,S)
\tkzLabelPoint[above right](R){$R$}
```

```
\tkzLabelPoint[below left](S){$S$}
\tkzDrawSegment[dashed](R,S)
\tkzInterLL(C,B)(R,S)
\tkzGetPoint{Y}
\tkzDrawPoints[color=blue](Y)
%\tkzLabelPoint[color=blue, above](Y){$Y$}
\end{tikzpicture}
```

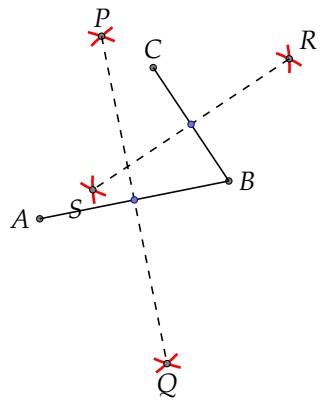


Figura 1.21: Construcción de las mediatrices

3. Sea O el punto de intersección de las rectas PQ y RS.

```
\begin{tikzpicture}[scale=0.6]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{-3/-1/A, 2/0/B,  0/3/C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzDrawSegments(A,B B,C)
% Primera Mediatriz
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{P}
\tkzGetSecondPoint{Q}
\tkzCompass[](A,P)
\tkzCompass[](B,P)
\tkzCompass[](A,Q)
\tkzCompass[](B,Q)
\tkzDrawPoints[color=black](P,Q)
\tkzLabelPoint[above](P){$P$}
\tkzLabelPoint[below](Q){$Q$}
\tkzDrawSegment[dashed](P,Q)
\tkzInterLL(A,B)(P,Q)
\tkzGetPoint{X}
\tkzDrawPoints[color=blue](X)
%\tkzLabelPoint[color=blue, above right](X){$X$}
% Segunda Mediatriz
```

```
\tkzInterCC(C,B)(B,C)
\tkzGetFirstPoint{R}
\tkzGetSecondPoint{S}
\tkzCompass[](C,R)
\tkzCompass[](B,R)
\tkzCompass[](B,S)
\tkzCompass[](C,S)
\tkzDrawPoints[color=black](R,S)
\tkzLabelPoint[above right](R){$R$}
\tkzLabelPoint[below left](S){$S$}
\tkzDrawSegment[dashed](R,S)
\tkzInterLL(C,B)(R,S)
\tkzGetPoint{Y}
\tkzDrawPoints[color=blue](Y)
%\tkzLabelPoint[color=blue, above](Y){$Y$}
% Centro
\tkzInterLL(R,S)(P,Q)
\tkzGetPoint{O}
\tkzDrawPoints[color=red](O)
\tkzLabelPoint[color=red, above left](O){$O$}
\end{tikzpicture}
```

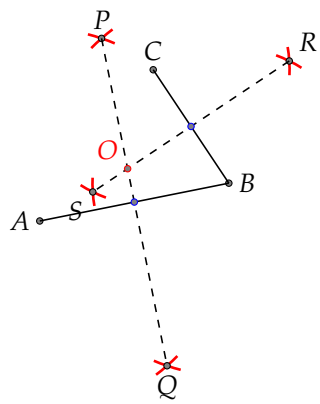


Figura 1.22: Definición del punto O

4. Describa el círculo con centro en O de radio OA. Este círculo pasa por A y también por B y C

```
\begin{tikzpicture}[scale=0.6]
\tkzSetUpCompass[color=red, line width=1 pt,style=solid]
\tkzDefPoints{-3/-1/A, 2/0/B,  0/3/C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzDrawSegments(A,B B,C)
% Primera Mediatriz
\tkzInterCC(A,B)(B,A)
\tkzGetFirstPoint{P}
```

```
\tkzGetSecondPoint{Q}
\tkzCompass[](A,P)
\tkzCompass[](B,P)
\tkzCompass[](A,Q)
\tkzCompass[](B,Q)
\tkzDrawPoints[color=black](P,Q)
\tkzLabelPoint[above](P){$P$}
\tkzLabelPoint[below](Q){$Q$}
\tkzDrawSegment[dashed](P,Q)
\tkzInterLL(A,B)(P,Q)
\tkzGetPoint{X}
\tkzDrawPoints[color=blue](X)
%\tkzLabelPoint[color=blue, above right](X){$X$}
% Segunda Mediatriz
\tkzInterCC(C,B)(B,C)
\tkzGetFirstPoint{R}
\tkzGetSecondPoint{S}
\tkzCompass[](C,R)
\tkzCompass[](B,R)
\tkzCompass[](B,S)
\tkzCompass[](C,S)
\tkzDrawPoints[color=black](R,S)
\tkzLabelPoint[above right](R){$R$}
\tkzLabelPoint[below left](S){$S$}
\tkzDrawSegment[dashed](R,S)
\tkzInterLL(C,B)(R,S)
\tkzGetPoint{Y}
\tkzDrawPoints[color=blue](Y)
%\tkzLabelPoint[color=blue, above](Y){$Y$}
% Centro
\tkzInterLL(R,S)(P,Q)
\tkzGetPoint{O}
\tkzDrawPoints[color=red](O)
\tkzLabelPoint[color=red, above left](O){$O$}
% Círculo
\tkzDrawCircle[thick](O,A)
\end{tikzpicture}
```

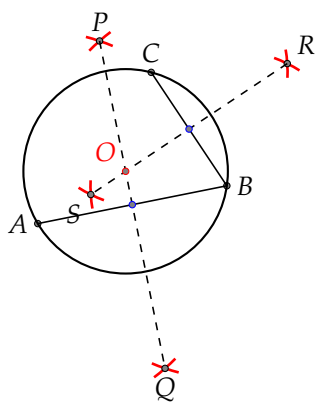
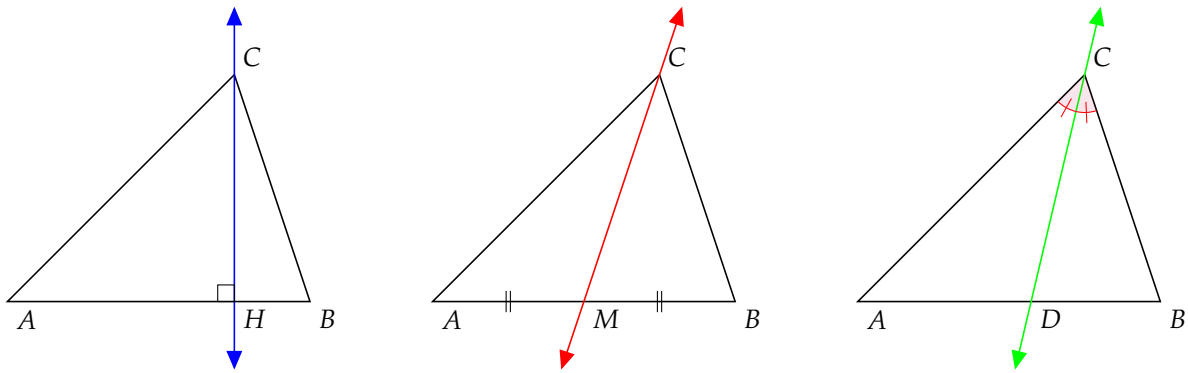


Figura 1.23: Círculo que pasa por A, B y C

Rectas Notables de un triángulo

La **altura** de un triángulo es la recta perpendicular a un lado (o a su prolongación) por el vértice opuesto. La **mediana** es el segmento que va desde un vértice hasta el punto medio del lado opuesto. La **bisectriz** de un ángulo es la semirrecta que divide al ángulo en dos partes iguales. También es el lugar geométrico de los puntos que equidistan de los lados del ángulo. Las tres rectas se presentan en la figura 1.24.



(a) *Altura* (b) *Mediana* (c) *Bisectriz*

Figura 1.24: Rectas Notables

El código utilizado ha sido el siguiente:

```
\begin{tabular}{ccc}
\begin{tikzpicture}
\clip (-0.6,-0.9) rectangle (4.6,3.9);
\tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B} \tkzDefPoint(3,3){C}
\tkzDrawPolygon(A,B,C)
\tkzSetUpLine[color=blue] \tkzDrawAltitude[arrows=triangle 45-triangle 45,add=0.3 and 0.3](A,B)(C) \
  tkzGetPoint{H}
\tkzMarkRightAngle(A,H,C) \tkzLabelPoints[below right](A,B,H) \tkzLabelPoints[above right](C)
\end{tikzpicture}
&
\begin{tikzpicture}
\clip (-0.6,-0.9) rectangle (4.6,3.9);
\tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B} \tkzDefPoint(3,3){C} \tkzDrawPolygon(A,B,C)
\tkzSetUpLine[color=red] \tkzDrawMedian[arrows=triangle 45-triangle 45,add=0.3 and 0.3](A,B)(C) \
  tkzGetPoint{M}
\tkzMarkSegments[mark=||](A,M M,B)
\tkzLabelPoints[below right](A,B,M) \tkzLabelPoints[above right](C)
\end{tikzpicture}
&
\begin{tikzpicture}
\clip (-0.6,-0.9) rectangle (4.6,3.9);
\tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B} \tkzDefPoint(3,3){C}
\tkzDrawPolygon(A,B,C) \tkzSetUpLine[color=green] \tkzDrawBisector[arrows=triangle 45-triangle 45,add
  =0.3 and 0.3](A,C,B)
```

```
\tkzGetPoint{D}
\tkzMarkAngles[size=0.5cm, color=red,fill=purple, fill opacity=0.1, mark=|](A,C,D D,C,B)
\tkzLabelPoints[below right](A,B,D) \tkzLabelPoints[above right](C)
\end{tikzpicture}
\\[2mm]
(a) \textit{Altura} & (b) \textit{Mediana} & (c) \textit{Bisectriz}
\end{tabular}
```

Dos rectas tangentes a un círculo dado desde un punto exterior

1. Sea K un círculo con centro O y sea P algún punto exterior a K.

```
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/O, 4/0/P, 1.5/0/L}
\tkzDrawPoints(O,P)
\tkzLabelPoint[left](O){$O$}
\tkzLabelPoint[above right](P){$P$}
\tkzDrawCircle(O,L)
\end{tikzpicture}
```

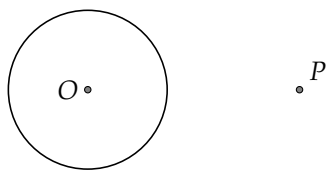


Figura 1.25: Círculo K y punto exterior P

2. Conéctese OP y localice su punto medio C.

```
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/O, 4/0/P, 1.5/0/L}
\tkzDrawPoints(O,P)
\tkzLabelPoint[left](O){$O$}
\tkzLabelPoint[above right](P){$P$}
\tkzDrawCircle(O,L)
\tkzDrawSegment(O,P)
\tkzDefMidPoint(O,P)
\tkzGetPoint{C}
\tkzDrawPoint(C)
\tkzLabelPoint[below](C){$C$}
\end{tikzpicture}
```

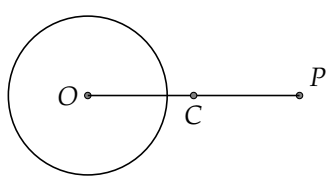


Figura 1.26: Definición del Punto medio C

3. Describa el círculo con centro en C y radio CO. Sean A y B sus intersecciones con K.

```
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/O, 4/0/P, 1.5/0/L}
\tkzDrawPoints(O,P)
\tkzLabelPoint[left](O){$O$}
\tkzLabelPoint[above right](P){$P$}
\tkzDrawCircle(O,L)
\tkzDrawSegment(O,P)
\tkzDefMidPoint(O,P)
\tkzGetPoint{C}
\tkzDrawPoint(C)
\tkzLabelPoint[below](C){$C$}
\tkzDrawCircle[dashed](C,O)
\tkzInterCC(O,L)(C,O)
\tkzGetPoints{A}{B}
\tkzDrawPoints[color=red](A,B)
\tkzLabelPoint[color=red, above](A){$A$}
\tkzLabelPoint[color=red, below](B){$B$}
\end{tikzpicture}
```

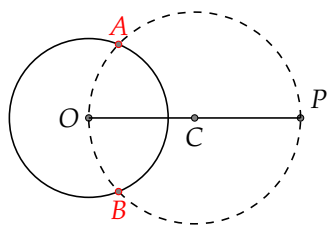


Figura 1.27: Rectas Tangentes a un círculo desde un punto exterior

4. Trace la rectas PA, PB; estas son las tangentes desde P a K.

```
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/O, 4/0/P, 1.5/0/L}
\tkzDrawPoints(O,P)
\tkzLabelPoint[left](O){$O$}
\tkzLabelPoint[above right](P){$P$}
\tkzDrawCircle(O,L)
\tkzDrawSegment(O,P)
\tkzDefMidPoint(O,P)
\tkzGetPoint{C}
\tkzDrawPoint(C)
\tkzLabelPoint[below](C){$C$}
\tkzDrawCircle[dashed](C,O)
\tkzInterCC(O,L)(C,O)
\tkzGetPoints{A}{B}
\tkzDrawPoints[color=red](A,B)
```

```
\tkzLabelPoint[color=red, above](A){$A$}
\tkzLabelPoint[color=red, below](B){$B$}
\tkzDrawLines[add = 1/3 and 1/3](A,P B,P)
\end{tikzpicture}
```

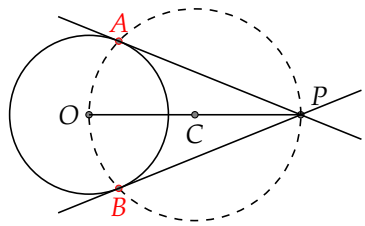


Figura 1.28: Rectas Tangentes a un círculo desde un punto exterior

Caracol con hoyuelo

Podemos hacer dibujos complejos como el de un caracol con hoyuelo, mostrado en la figura 1.29

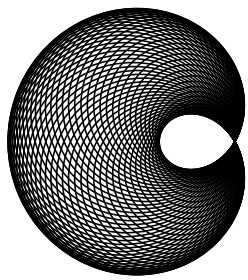


Figura 1.29: Caracol con hoyuelo

El código utilizado ha sido el siguiente:

```
\begin{tikzpicture}[scale=0.8]
\tkzDefPoint(0,0){O}
\tkzDefPoint(2,0){A}
\foreach \ang in {5,10,15,...,360}{%
\tkzDefPoint(\ang:1){M}
\tkzDrawCircle[](M,A)
}
\end{tikzpicture}
```

En este caso, la definición del punto M y el punto A definen al caracol como el lugar geométrico de los círculos centrados en M con radio A. En tal caso el punto M es una réplica sostenida de un ángulo que varía cada cinco grados en su abscisa y una ordenada constante de uno. Es importante aclarar que la toma de los valores del ángulo debe ser continuo, no obstante, el cómputo de cálculos que TEX realiza no lo permite.

1.4 Conclusión

El paquete **tkz-euclide** es una herramienta en sí misma diseñada para la creación de figuras geométricas sin un conocimiento completo del paquete **TikZ**, lo cual lo vuelve innecesario si el lector tiene un conocimiento avanzado de las librerías de dicho paquete. No obstante, el uso se encuentra dentro de una curva aprendizaje baja y se vuelve adecuado dentro de la experiencia de los y las estudiantes. Podemos resumir, con ello, las siguientes conclusiones:

1. Los cálculos internos dependen de TEX lo cual lo hace lento para muchas acciones o precisiones en un documento extenso.
2. El manejo de sintaxis diferentes es posible, pero puede conducir al error por la diferencias mismas que existen, por ejemplo, usar punto y coma al final o colocar un nombre diferente a los nodos.
3. El manejo simulado de reglas y compás se vuelve más apropiado y fácil que en **TikZ**.
4. El paquete ofrece un proceso de construcción y desarrollo muy similar al que se haría en papel, articulando y estimulando la lógica en los y las estudiantes.
5. El paquete no viene a sustituir el proceso de construcción como tal, sino que se enfoca en el diseño editorial de las construcciones

Bibliografía

[1] Matthes, A. (2010) *tkz-euclide.*. Descargado de: http://www.altermundus.fr/downloads/documents/Sangaku.pdf

[2] Matthes, A. (2011) *tkz-euclide.*. Descargado de: http://mirrors.ucr.ac.cr/CTAN/macros/latex/contrib/tkz/tkz-euclide/doc/tkz-euclide-screen.pdf

[3] Montesdeoca, A. (2017) *Geometría métrica y proyectiva en el plano con coordenadas baricéntricas. Algunos tópicos*. Descargado de: https://amontes.webs.ull.es/pdf/geoba.pdf

[4] Tantau, T. (2015) *The TikZ and PGF Packages*. Descargado de: http://mirrors.ucr.ac.cr/CTAN/graphics/pgf/base/doc/pgfmanual.pdf

[5] Varilly, J. (2014) *Elementos de Geometría Plana*. (Segunda Edición). San José: Editorial de la Universidad de Costa Rica.