tkz-euclide

Alain Matthes

tkz-euclide passes in version 5 with the possibility of carrying out part of the calculations using lua. See the "news" and "lua" sections for more information.

tkz-euclide is a set of convenient macros for drawing in a plane (fundamental two-dimensional object) with a Cartesian coordinate system. It handles the most classic situations in Euclidean Geometry. tkz-euclide is built on top of PGF and its associated front-end Ti\LaTeX{} and is a (La)\TeX{}-friendly drawing package. The aim is to provide a high-level user interface to build graphics relatively simply. The idea is to allow you to follow step by step a construction that would be done by hand as naturally as possible.

English is not my native language so there might be some errors.

Firstly, I would like to thank Till Tantau for the beautiful \LaTeX{} package, namely Ti\LaTeX{}.

Acknowledgements : I received much valuable advice, remarks, corrections and examples from Jean-Côme Charpentier, Josselin Noirel, Manuel Pégourié-Gonnard, Franck Pastor, David Arnold, Ulrike Fischer, Stefan Kottwitz, Christian Tellechea, Nicolas Kisselhoff, David Arnold, Wolfgang Böchel, John Kitzmiller, Dimitri Kapetas, Gaétan Marris, Mark Wibrow, Yves Combe for his work on a protractor, Paul Gaborit, Laurent Van Deik for all his corrections, remarks and questions and Muzimuzhi Z for the code about the option "dim". A big thank you to Chetan Shirore and Dr. Ajit Kumar because their work on complex numbers in their package luamaths helped me a lot.

I would also like to thank Eric Weisstein, creator of MathWorld: MathWorld.

You can find some examples on my site: altermundus.fr. under construction!

Please report typos or any other comments to this documentation to: Alain Matthes.

This file can be redistributed and/or modified under the terms of the \LaTeX{} Project Public License Distributed from CTAN archives.
# Contents

## I. General survey: a brief but comprehensive review

0.1. With 5.0 version ................................................. 15  
0.2. With 4.2 version ................................................ 15  
0.3. Changes with previous versions ............................... 17  

1. Working with lua: option lua ..................................... 19  

2. Installation .................................................................. 19  

3. Presentation and Overview  
   3.1. Why \texttt{tkz-euclide}? ..................................... 20  
   3.2. \texttt{TikZ} vs \texttt{tkz-euclide}  
      3.2.1. Book I, proposition I _Euclid's Elements_ ......... 20  
      3.2.2. Complete code with \texttt{tkz-euclide} ............ 21  
      3.2.3. Book I, Proposition II _Euclid's Elements_ ....... 22  
   3.3. \texttt{tkz-euclide 4} vs \texttt{tkz-euclide 3} ............... 24  
   3.4. \texttt{tkz-euclide 5} vs \texttt{tkz-euclide 4} ............... 24  
   3.5. How to use the \texttt{tkz-euclide} package? .......... 24  
      3.5.1. Let's look at a classic example .................. 24  
      3.5.2. Part I: golden triangle .............................. 26  
      3.5.3. Part II: two others methods with golden and euclid triangle 27  
      3.5.4. Complete but minimal example .................. 28  

4. The Elements of \texttt{tkz} code  
   4.1. Objects and language ...................................... 30  
   4.2. Notations and conventions ................................. 31  
   4.3. \texttt{Set}, \texttt{Calculate}, \texttt{Draw}, \texttt{Mark}, \texttt{Label} .................. 32  

5. About this documentation and the examples .................. 33  

## II. Setting

6. First step: fixed points .......................................... 35  

7. Definition of a point: \texttt{\textbackslash tkzDefPoint} or \texttt{\textbackslash tkzDefPoints}  
   7.1. Defining a named point \texttt{\textbackslash tkzDefPoint} .... 36  
      7.1.1. Cartesian coordinates ............................... 36  
      7.1.2. Calculations with \texttt{xfp} ......................... 37  
      7.1.3. Polar coordinates .................................... 37  
      7.1.4. Relative points ....................................... 37  
   7.2. Point relative to another: \texttt{\textbackslash tkzDefShiftPoint} ....... 37  
      7.2.1. Isosceles triangle .................................. 38  
      7.2.2. Equilateral triangle ............................... 38  
      7.2.3. Parallelogram ....................................... 38  
   7.3. Definition of multiple points: \texttt{\textbackslash tkzDefPoints} .......... 39  
   7.4. Create a triangle .......................................... 39  
   7.5. Create a square ........................................... 39

\texttt{tkz-euclide}  
 AlterMundus
III. Calculating

8. Auxiliary tools

8.1. Constants  statutory

8.2. New point by calculation  statutory

9. Special points

9.1. Middle of a segment \tkzDefMidPoint 

9.1.1. Use of \tkzDefMidPoint  statutory

9.2. Golden ratio \tkzDefGoldenRatio 

9.2.1. Use the golden ratio to divide a line segment  statutory

9.2.2. Golden arbelos  statutory

9.3. Barycentric coordinates with \tkzDefBarycentricPoint 

9.3.1. with two points  statutory

9.3.2. with three points  statutory

9.4. Internal and external Similitude Center

9.4.1. Internal and external with node  statutory

9.4.2. D’Alembert Theorem  statutory

9.4.3. Example with node  statutory

9.5. Harmonic division with \tkzDefHarmonic 

9.5.1. options ext and int  statutory

9.5.2. Bisector and harmonic division  statutory

9.5.3. option both  statutory

9.6. Equidistant points with \tkzDefEquiPoints 

9.6.1. Using \tkzDefEquiPoints with options  statutory

9.7. Middle of an arc  statutory

10. Point on line or circle

10.1. Point on a line with \tkzDefPointOnLine 

10.1.1. Use of option pos  statutory

10.2. Point on a circle with \tkzDefPointOnCircle 

10.2.1. Altshiller’s Theorem  statutory

10.2.2. Use of \tkzDefPointOnCircle  statutory

11. Special points relating to a triangle

11.1. Triangle center: \tkzDefTriangleCenter 

11.1.1. Option ortho or orthic  statutory

11.1.2. Option centroid  statutory

11.1.3. Option circum  statutory

11.1.4. Option in  statutory

11.1.5. Option ex  statutory

11.1.6. Option euler  statutory

11.1.7. Option symmedian  statutory

11.1.8. Option spieker  statutory

11.1.9. Option spieker  statutory

11.1.10. Option nagel  statutory

11.1.11. Option mittenpunkt  statutory

11.1.12. Relation between gergonne, centroid and mittenpunkt  statutory

12. Definition of points by transformation

12.1. \tkzDefPointBy 

12.1.1. translation  statutory

12.1.2. reflection (orthogonal symmetry)  statutory

12.1.3. homothety and projection  statutory
# Contents

12.1.4. **projection** ................................................................. 61
12.1.5. **symmetry** .......................................................... 61
12.1.6. **rotation** .............................................................. 62
12.1.7. **rotation in radian** ................................................... 62
12.1.8. **rotation with nodes** ................................................ 62
12.1.9. **inversion** ............................................................. 62
12.1.10. **Inversion of lines ex 1** ......................................... 64
12.1.11. **inversion of lines ex 2** ......................................... 64
12.1.12. **inversion of lines ex 3** ......................................... 64
12.1.13. **inversion of circle and homothety** .......................... 65
12.1.14. **inversion** of **Triangle with respect to the Incircle** .... 65
12.1.15. **inversion** of **orthogonal circle with inversion circle** ... 65
12.1.16. **inversion negative** ............................................... 66
12.2. **Transformation** of **multiple points**: \texttt{\textbackslash tkzDefPointsBy} ________________________ 67
12.2.1. **translation** of **multiple points** .............................. 67
12.2.2. **symmetry** of **multiple points**: an **oval** ................. 68

13. **Defining points using a vector** ........................................ 68
13.1. \texttt{\textbackslash tkzDefPointWith} ...................................... 68
   13.1.1. **Option** \texttt{colinear at}, **simple example** .......... 69
   13.1.2. **Option** \texttt{colinear at}, **complex example** ......... 69
   13.1.3. **Option** \texttt{colinear at} ...................................... 70
   13.1.4. **Option** \texttt{colinear at} ...................................... 70
   13.1.5. **Option** \texttt{orthogonal} ....................................... 70
   13.1.6. **Option** \texttt{orthogonal} ....................................... 71
   13.1.7. **Option** \texttt{orthogonal} **more complicated example** .. 71
   13.1.8. **Options** \texttt{colinear} and \texttt{orthogonal} ............. 72
   13.1.9. **Option** \texttt{orthogonal normed} ............................ 72
   13.1.10. **Option** \texttt{orthogonal normed} and \texttt{K=2} ........... 72
   13.1.11. **Option** \texttt{linear} ........................................... 72
   13.1.12. **Option** \texttt{linear normed} ................................ 73
13.2. \texttt{\textbackslash tkzGetVectxy} ........................................ 73
   13.2.1. **Coordinate transfer with** \texttt{\textbackslash tkzGetVectxy} ...... 73

14. **Straight lines** ................................................................... 73
14.1. **Definition** of **straight lines** ...................................... 73
   14.1.1. With \texttt{mediator} ................................................ 74
   14.1.2. An **envelope** with **option** \texttt{mediator} .......... 74
   14.1.3. A **parabola** with **option** \texttt{mediator} .............. 74
   14.1.4. With **options** \texttt{bisector} and \texttt{normed} .......... 75
   14.1.5. With **option** \texttt{parallel=through} ....................... 75
   14.1.6. With **option** \texttt{orthogonal} and \texttt{parallel} ........ 76
   14.1.7. With **option** \texttt{altitude} ...................................... 76
   14.1.8. With **option** \texttt{euler} ......................................... 76
   14.1.9. **Tangent** passing through a **point** on the circle \texttt{tangent at} ........................................ 77
   14.1.10. **Choice** of **contact point** with tangents passing through an **external point** \texttt{tangent from} ......................... 77
   14.1.11. **Example** of **tangents** passing through an **external point** ........................................ 78
   14.1.12. **Example** of Andrew Mertz .................................... 78
   14.1.13. Drawing a **tangent** option \texttt{tangent from} .......... 79
15. Triangles
  15.1. Definition of triangles \texttt{\textls{tkzDefTriangle}} ........................................ 79
      15.1.1. Option \texttt{equilateral} .................................................. 80
      15.1.2. Option \texttt{two angles} .................................................... 81
      15.1.3. Option \texttt{school} .......................................................... 81
      15.1.4. Option \texttt{pythagore} ...................................................... 81
      15.1.5. Option \texttt{pythagore} and \texttt{swap} ................................. 81
      15.1.6. Option \texttt{golden} .......................................................... 82
      15.1.7. Option \texttt{euclid} ........................................................... 82
      15.1.8. Option \texttt{isosceles right} ............................................... 83
      15.1.9. Option \texttt{gold} .............................................................. 83
  15.2. Specific triangles with \texttt{\textls{tkzDefSpcTriangle}} .................................. 84
      15.2.1. How to name the vertices ..................................................... 84
  15.3. Option \texttt{medial} or \texttt{centroid} ........................................... 84
      15.3.1. Option \texttt{in} or \texttt{incentral} ........................................ 85
      15.3.2. Option \texttt{ex} or \texttt{excentral} ....................................... 85
      15.3.3. Option \texttt{intouch} or \texttt{contact} ................................... 86
      15.3.4. Option \texttt{extouch} .......................................................... 86
      15.3.5. Option \texttt{orthic} ............................................................. 87
      15.3.6. Option \texttt{feuerbach} ....................................................... 88
      15.3.7. Option \texttt{tangential} ...................................................... 88
      15.3.8. Option \texttt{euler} .............................................................. 89
      15.3.9. Option \texttt{euler} and Option \texttt{orthic} ................................ 90
      15.3.10. Option \texttt{symmedial} ..................................................... 91
  15.4. Permutation of two points of a triangle ............................................. 91
      15.4.1. Modification of the \texttt{school} triangle ................................ 92

16. Definition of polygons ................................................................. 92
  16.1. Defining the points of a square .................................................. 92
      16.1.1. Using \texttt{\textls{tkzDefSquare}} with two points ......................... 92
      16.1.2. Use of \texttt{\textls{tkzDefSquare}} to obtain an isosceles right-angled triangle 93
      16.1.3. Pythagorean Theorem and \texttt{\textls{tkzDefSquare}} ....................... 93
  16.2. Defining the points of a rectangle .............................................. 93
      16.2.1. Example of a rectangle definition ......................................... 93
  16.3. Definition of parallelogram ...................................................... 94
      16.3.1. Example of a parallelogram definition ..................................... 94
  16.4. The golden rectangle ............................................................... 94
      16.4.1. Golden Rectangles .............................................................. 94
      16.4.2. Construction of the golden rectangle ..................................... 95
  16.5. Regular polygon ........................................................................ 95
      16.5.1. Option \texttt{center} ............................................................. 95
      16.5.2. Option \texttt{side} ............................................................... 96

17. Circles ......................................................................................... 97
  17.1. Characteristics of a circle: \texttt{\textls{tkzDefCircle}} ......................... 97
      17.1.1. Example with option \texttt{R} .................................................. 98
      17.1.2. Example with option \texttt{diameter} ....................................... 98
      17.1.3. Circles inscribed and circumscribed for a given triangle .............. 98
      17.1.4. Example with option \texttt{ex} .................................................. 98
      17.1.5. Euler's circle for a given triangle with option \texttt{euler} .......... 99
      17.1.6. Apollonius circles for a given segment option \texttt{apollonius} ...... 100
      17.1.7. Circles exinscribed to a given triangle option \texttt{ex} ............... 100
      17.1.8. Spieker circle with option \texttt{spieker} .................................. 100
17.2. Projection of excenters ........................................... 101
  17.2.1. Excircles ......................................................... 102
  17.2.2. Orthogonal from .............................................. 103
  17.2.3. Orthogonal through ......................................... 103
17.3. Definition of circle by transformation; $\texttt{\textbackslash tkzDefCircleBy}$ ................................................. 104
  17.3.1. Translation ..................................................... 105
  17.3.2. Reflection (orthogonal symmetry) .............................. 105
  17.3.3. Homothety ....................................................... 105
  17.3.4. Symmetry ........................................................ 106
  17.3.5. Rotation ........................................................ 106
  17.3.6. Inversion ........................................................ 106
18. Intersections ....................................................... 117
  18.1. Intersection of two straight lines $\texttt{\textbackslash tkzInterLL}$ ......................................................... 107
    18.1.1. Example of intersection between two straight lines ................................................. 107
  18.2. Intersection of a straight line and a circle $\texttt{\textbackslash tkzInterLC}$ ......................................................... 107
    18.2.1. test line-circle intersection .................................. 108
    18.2.2. Line-circle intersection ....................................... 108
    18.2.3. Line passing through the center option $\texttt{\textbackslash common}$ ......................................................... 108
    18.2.4. Line-circle intersection with option $\texttt{\textbackslash common}$ ......................................................... 109
    18.2.5. Line-circle intersection order of points ....................... 109
    18.2.6. Example with $\texttt{\textbackslash foreach}$ ......................... 110
    18.2.7. Line-circle intersection with option $\texttt{\textbackslash near}$ ......................................................... 110
    18.2.8. More complex example of a line-circle intersection ............. 111
    18.2.9. Circle defined by a center and a measure, and special cases ......................................................... 111
    18.2.10. Calculation of radius ......................................... 112
    18.2.11. Option “with nodes” .......................................... 113
  18.3. Intersection of two circles $\texttt{\textbackslash tkzInterCC}$ ......................................................... 113
    18.3.1. test circle-circle intersection .................................. 114
    18.3.2. circle-circle intersection with $\texttt{\textbackslash common}$ point ......................................................... 114
    18.3.3. circle-circle intersection order of points ....................... 114
    18.3.4. Construction of an equilateral triangle ......................................................... 115
    18.3.5. Segment trisection .............................................. 115
    18.3.6. With the option “with nodes” .................................. 116
    18.3.7. Mix of intersections ............................................ 116
    18.3.8. Altshiller-Court’s theorem ...................................... 116
19. Angles ................................................................. 117
  19.1. Definition and usage with $\texttt{\textbackslash tkz-euclide}$ ......................................................... 117
  19.2. Recovering an angle $\texttt{\textbackslash tkzGetAngle}$ ......................................................... 118
  19.3. Angle formed by three points ...................................... 118
    19.3.1. Verification of angle measurement ................................ 119
    19.3.2. Determination of the three angles of a triangle .................. 119
    19.3.3. Angle between two circles ...................................... 119
  19.4. Angle formed by a straight line with the horizontal axis $\texttt{\textbackslash tkzFindSlopeAngle}$ ......................................................... 120
    19.4.1. How to use $\texttt{\textbackslash tkzFindSlopeAngle}$ ......................................................... 120
    19.4.2. Use of $\texttt{\textbackslash tkzFindSlopeAngle}$ and $\texttt{\textbackslash tkzGetAngle}$ ......................................................... 120
    19.4.3. Another use of $\texttt{\textbackslash tkzFindSlopeAngle}$ ......................................................... 121
20. Random point definition ........................................... 121
  20.1. Obtaining random points ........................................... 121
    20.1.1. Random point in a rectangle .................................... 122
    20.1.2. Random point on a segment or a line ............................ 122
## IV. Drawing and Filling

### 21. Drawing

- **21.1. Draw a point or some points**
  - Drawing points `\tkzDrawPoint` ........................................ 124
  - Example of point drawings ........................................... 124
  - Example .............................................................. 125

### 22. Drawing the lines

- **22.1. Draw a straight line**
  - Examples with `add` .................................................. 126
  - Example with `\tkzDrawLines` ...................................... 126

### 23. Drawing a segment

- **23.1. Draw a segment `\tkzDrawSegment`** ............................. 126
  - Example with point references ..................................... 127
  - Example of extending an segment with option `add` .......... 127
  - Adding dimensions with option `dim` new code from Muzimuzhi Z ........................................ 127
  - Adding dimensions with option `dim part I` ................. 128
  - Adding dimensions with option `dim part II` ............... 129
- **23.2. Drawing segments `\tkzDrawSegments`** ............................. 129
  - Place an arrow on segment ......................................... 129
- **23.3. Drawing line segment of a triangle** ............................ 130
  - How to draw `Altitude` .............................................. 130
- **23.4. Drawing a polygon** ............................................ 130
  - `\tkzDrawPolygon` .................................................. 130
  - Option `two angles` ............................................... 131
  - Style of line ...................................................... 131
- **23.5. Drawing a polygonal chain** .................................... 131
  - Polygonal chain .................................................. 132
  - The idea is to inscribe two squares in a semi-circle .......... 132
  - Polygonal chain: index notation .................................. 132

### 24. Draw a circle with `\tkzDrawCircle`

- **24.1. Draw one circle** ............................................... 132
  - Circles and styles, draw a circle and color the disc .......... 133
- **24.2. Drawing circles** .............................................. 133
  - Circles defined by a triangle .................................... 134
  - Concentric circles ................................................ 134
  - Exinscribed circles ............................................... 135
  - Cardioid ............................................................ 135
- **24.3. Drawing semicircle** ............................................ 136
  - Use of `\tkzDrawSemiCircle` .................................... 136
- **24.4. Drawing semicircles** .......................................... 136
  - Use of `\tkzDrawSemiCircles` : Golden arbelos ............... 137

### 25. Drawing arcs

- **25.1. Macro: `\tkzDrawArc`** ........................................ 137
  - Option `towards` .................................................. 137
  - Option `towards` .................................................. 138
  - Option `rotate` .................................................. 138
28. Clipping different objects

28.1. Clipping a polygon
   28.1.1. \texttt{tkzClipPolygon}
   28.1.2. \texttt{tkzClipPolygon[out]}
   28.1.3. Example: use of "Clip" for Sangaku in a square

28.2. Clipping a disc
   28.2.1. Simple clip
   28.2.2. Clip out

28.3. Intersection of disks

28.4. Clipping a sector
   28.4.1. \texttt{tkzClipSector}
   28.4.2. \texttt{tkzClipSector[R with nodes]}
   28.4.3. Example 1
   28.4.4. Example 2

28.5. Options from \LaTeX{}: trim left or right
   28.5.1. Example 1
   28.5.2. Example 2

28.6. Intersection of circles
V. Marking 158
28.9. Mark a segment \texttt{\tkzMarkSegment} .................................................. 159
28.9.1. Several marks .................................................. 159
28.9.2. Use of \texttt{mark} .................................................. 159
28.10. Marking segments \texttt{\tkzMarkSegments} ........................................ 159
28.10.1. Marks for an isosceles triangle ........................................ 159
28.11. Another marking .................................................. 160
28.12. Mark an arc \texttt{\tkzMarkArc} ................................................ 160
28.12.1. Several marks .................................................. 160
28.13. Mark an angle mark : \texttt{\tkzMarkAngle} ....................................... 160
28.13.1. Example with \texttt{mark = x} and with \texttt{mark =} ............. 161
28.14. Problem to mark a small angle: \texttt{Option veclen} .................. 161
28.15. Marking a right angle: \texttt{\tkzMarkRightAngle} ....................... 162
28.15.1. Example of marking a right angle ....................................... 162
28.15.2. Example of marking a right angle, german style .................... 163
28.15.3. Mix of styles .................................................. 163
28.15.4. Full example .................................................. 164
28.16. \texttt{\tkzMarkRightAngles} .......................................... 164
28.17. Angles Library .................................................. 164
28.17.1. Angle with Ti\LaTeX{} ........................................ 165

VI. Labelling 166
29. Labelling 167
29.1. Label for a point .................................................. 167
29.1.1. Example with \texttt{\tkzLabelPoint} ................................ 167
29.1.2. Label and reference ........................................... 167
29.2. Add labels to points \texttt{\tkzLabelPoints} ................................ 167
29.2.1. Example with \texttt{\tkzLabelPoints} .................................. 168
29.3. Automatic position of labels \texttt{\tkzAutoLabelPoints} ............. 168
29.3.1. Label for points with \texttt{\tkzAutoLabelPoints} ................... 168
30. Label for a segment .................................................. 168
30.0.1. First example .................................................. 169
30.0.2. Example : blackboard ........................................... 169
30.0.3. Labels and option : \texttt{swap} ..................................... 169
30.0.4. Labels for an isosceles triangle .................................. 170
31. Add labels on a straight line \texttt{\tkzLabelLine} 170
31.0.1. Example with \texttt{\tkzLabelLine} .................................. 170
31.1. Label at an angle : \texttt{\tkzLabelAngle} ................................ 170
31.1.1. Example author js bibra stackexchange ......................... 171
31.1.2. With \texttt{pos} .................................................. 171
31.1.3. \texttt{pos} and \texttt{\tkzLabelAngles} ................................. 172
31.2. Giving a label to a circle ........................................... 172
31.2.1. Example ....................................................... 173

\texttt{tkz-euclide} AlterMundus
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>32.</td>
<td>Label for an arc</td>
<td>173</td>
</tr>
<tr>
<td>32.0.1.</td>
<td>Label on arc</td>
<td>173</td>
</tr>
<tr>
<td>VII.</td>
<td>Complements</td>
<td>174</td>
</tr>
<tr>
<td>33.</td>
<td>Using the compass</td>
<td>175</td>
</tr>
<tr>
<td>33.1.</td>
<td>Main macro \texttt{\textbackslash{}tkzCompass}</td>
<td>175</td>
</tr>
<tr>
<td>33.1.1.</td>
<td>Option \texttt{\textbackslash{}length}</td>
<td>175</td>
</tr>
<tr>
<td>33.1.2.</td>
<td>Option \texttt{\textbackslash{}delta}</td>
<td>175</td>
</tr>
<tr>
<td>33.2.</td>
<td>Multiple constructions \texttt{\textbackslash{}tkzCompass}</td>
<td>175</td>
</tr>
<tr>
<td>33.2.1.</td>
<td>Use \texttt{\textbackslash{}tkzCompass}</td>
<td>176</td>
</tr>
<tr>
<td>34.</td>
<td>The Show</td>
<td>176</td>
</tr>
<tr>
<td>34.1.</td>
<td>Show the constructions of some lines \texttt{\textbackslash{}tkzShowLine}</td>
<td>176</td>
</tr>
<tr>
<td>34.1.1.</td>
<td>Example of \texttt{\textbackslash{}tkzShowLine} and \texttt{\textbackslash{}parallel}</td>
<td>176</td>
</tr>
<tr>
<td>34.1.2.</td>
<td>Example of \texttt{\textbackslash{}tkzShowLine} and \texttt{\textbackslash{}perpendicular}</td>
<td>177</td>
</tr>
<tr>
<td>34.1.3.</td>
<td>Example of \texttt{\textbackslash{}tkzShowLine} and \texttt{\textbackslash{}bisector}</td>
<td>177</td>
</tr>
<tr>
<td>34.1.4.</td>
<td>Example of \texttt{\textbackslash{}tkzShowLine} and \texttt{\textbackslash{}mediator}</td>
<td>177</td>
</tr>
<tr>
<td>34.2.</td>
<td>Constructions of certain transformations \texttt{\textbackslash{}tkzShowTransformation}</td>
<td>177</td>
</tr>
<tr>
<td>34.2.1.</td>
<td>Example of the use of \texttt{\textbackslash{}tkzShowTransformation}</td>
<td>178</td>
</tr>
<tr>
<td>34.2.2.</td>
<td>Another example of the use of \texttt{\textbackslash{}tkzShowTransformation}</td>
<td>178</td>
</tr>
<tr>
<td>35.</td>
<td>Protractor</td>
<td>179</td>
</tr>
<tr>
<td>35.1.</td>
<td>The macro \texttt{\textbackslash{}tkzProtractor}</td>
<td>179</td>
</tr>
<tr>
<td>35.1.1.</td>
<td>The circular protractor</td>
<td>179</td>
</tr>
<tr>
<td>35.1.2.</td>
<td>The circular protractor, transparent and returned</td>
<td>180</td>
</tr>
<tr>
<td>36.</td>
<td>Miscellaneous tools and mathematical tools</td>
<td>180</td>
</tr>
<tr>
<td>36.1.</td>
<td>Duplicate a segment</td>
<td>180</td>
</tr>
<tr>
<td>36.1.1.</td>
<td>Use of \texttt{\textbackslash{}tkzDuplicateSegment}</td>
<td>180</td>
</tr>
<tr>
<td>36.1.2.</td>
<td>Proportion of gold with \texttt{\textbackslash{}tkzDuplicateSegment}</td>
<td>181</td>
</tr>
<tr>
<td>36.1.3.</td>
<td>Golden triangle or sublime triangle</td>
<td>181</td>
</tr>
<tr>
<td>36.2.</td>
<td>Segment length \texttt{\textbackslash{}tkzCalcLength}</td>
<td>181</td>
</tr>
<tr>
<td>36.2.1.</td>
<td>Compass square construction</td>
<td>182</td>
</tr>
<tr>
<td>36.2.2.</td>
<td>Example</td>
<td>182</td>
</tr>
<tr>
<td>36.3.</td>
<td>Transformation from pt to cm or cm to pt</td>
<td>182</td>
</tr>
<tr>
<td>36.4.</td>
<td>Change of unit</td>
<td>183</td>
</tr>
<tr>
<td>36.5.</td>
<td>Get point coordinates</td>
<td>183</td>
</tr>
<tr>
<td>36.5.1.</td>
<td>Coordinate transfer with \texttt{\textbackslash{}tkzGetPointCoord}</td>
<td>183</td>
</tr>
<tr>
<td>36.5.2.</td>
<td>Sum of vectors with \texttt{\textbackslash{}tkzGetPointCoord}</td>
<td>183</td>
</tr>
<tr>
<td>36.6.</td>
<td>Swap labels of points</td>
<td>184</td>
</tr>
<tr>
<td>36.6.1.</td>
<td>Use of \texttt{\textbackslash{}tkzSwapPoints}</td>
<td>184</td>
</tr>
<tr>
<td>36.7.</td>
<td>Dot Product</td>
<td>184</td>
</tr>
<tr>
<td>36.7.1.</td>
<td>Simple example</td>
<td>185</td>
</tr>
<tr>
<td>36.7.2.</td>
<td>Cocyclic points</td>
<td>185</td>
</tr>
<tr>
<td>36.8.</td>
<td>Power of a point with respect to a circle</td>
<td>186</td>
</tr>
<tr>
<td>36.8.1.</td>
<td>Power from the radical axis</td>
<td>186</td>
</tr>
<tr>
<td>36.9.</td>
<td>Radical axis</td>
<td>186</td>
</tr>
<tr>
<td>36.9.1.</td>
<td>Two circles disjonted</td>
<td>187</td>
</tr>
<tr>
<td>36.10.</td>
<td>Two intersecting circles</td>
<td>187</td>
</tr>
<tr>
<td>36.11.</td>
<td>Two externally tangent circles</td>
<td>187</td>
</tr>
<tr>
<td>36.12.</td>
<td>Two circles tangent internally</td>
<td>188</td>
</tr>
<tr>
<td>36.12.1.</td>
<td>Three circles</td>
<td>188</td>
</tr>
</tbody>
</table>

\texttt{tkz-euclide} AlterMundus
VIII. Working with style

37. Predefined styles

38. Points style

38.1. Use of \tkzSetUpPoint

38.1.1. Global style or local style

38.1.2. Local style

38.1.3. Style and scope

38.1.4. Simple example with \tkzSetUpPoint inside a group

38.1.5. Use of \tkzSetUpPoint inside a group

39. Lines style

39.1. Use of \tkzSetUpLine

39.1.1. Change line width

39.1.2. Change style of line

39.1.3. Example 3: extend lines

40. Arc style

40.1. The macro \tkzSetUpArc

40.1.1. Use of \tkzSetUpArc

41. Compass style, configuration macro \tkzSetUpCompass

41.1. The macro \tkzSetUpCompass

41.1.1. Use of \tkzSetUpCompass

41.1.2. Use of \tkzSetUpCompass with \tkzShowLine

42. Label style

42.1. The macro \tkzSetUpLabel

42.1.1. Use of \tkzSetUpLabel

43. Own style

43.1. The macro \tkzSetUpStyle

43.1.1. Use of \tkzSetUpStyle

44. How to use arrows

44.1. Arrows at endpoints on segment, ray or line

44.1.1. Scaling an arrow head

44.1.2. Using vector style

44.2. Arrows on middle point of a line segment

44.2.1. In a parallelogram

44.2.2. A line parallel to another one

44.2.3. Arrow on a circle

44.3. Arrows on all segments of a polygon

44.3.1. Arrow on each segment with \tkz arrows

44.3.2. Using \tkz arrows with a circle
IX. Examples 202

45. Different authors 203
45.1. Code from Andrew Swan 203
45.2. Example: Dimitris Kapeta 203
45.3. Example: John Kitzmiller 204
45.4. Example 1: from Indonesia 205
45.5. Example 2: from Indonesia 206
45.6. Illustration of the Morley theorem by Nicolas François 208
45.7. Gou gu theorem / Pythagorean Theorem by Zhao Shuang 209
45.8. Reuleaux-Triangle 210

46. Some interesting examples 211
46.1. Square root of the integers 211
46.2. About right triangle 211
46.3. Archimedes 212
46.3.1. Square and rectangle of same area; Golden section 214
46.3.2. Steiner Line and Simson Line 215
46.4. Lune of Hippocrates 216
46.5. Lunes of Hasan Ibn al-Haytham 216
46.6. About clipping circles 218
46.7. Similar isosceles triangles 219
46.8. Revised version of "Tangente" 220
46.9. "Le Monde" version 221
46.10. Triangle altitudes 222
46.11. Altitudes - other construction 223
46.12. Three circles in an Equilateral Triangle 224
46.13. Law of sines 225
46.14. Flower of Life 226
46.15. Pentagon in a circle 228
46.16. Pentagon in a square 229
46.17. Hexagon Inscribed 230
46.17.1. Hexagon Inscribed version 1 230
46.17.2. Hexagon Inscribed version 2 230
46.18. Power of a point with respect to a circle 231
46.19. Radical axis of two non-concentric circles 232
46.20. External homothetic center 233
46.21. Tangent lines to two circles 234
46.22. Tangent lines to two circles with radical axis 235
46.23. Middle of a segment with a compass 237
46.24. Definition of a circle _Apollonius_ 238
46.25. Application of Inversion: Pappus chain 239
46.26. Book of lemmas proposition 1 Archimedes 240
46.27. Book of lemmas proposition 6 Archimedes 240
46.28. "The" Circle of APOLLONIUS 243

X. FAQ 246

47. FAQ 247
47.1. Most common errors 247
Part I.

General survey: a brief but comprehensive review
News and compatibility

1.1. With 5.0 version

- Finally, I added the "lua" option for the package \texttt{tkz-euclide}. This allows to do the calculations for the main functions using lua; (see 1). The syntax is unchanged. Nothing changes for the user.

- The "xfp" option has become "veclen" see 28.14;

1.2. With 4.2 version

Some changes have been made to make the syntax more homogeneous and especially to distinguish the definition and search for coordinates from the rest, i.e. drawing, marking and labelling. Now the definition macros are isolated, it will be easier to introduce a phase of coordinate calculations using \texttt{Lua}.

Here are some of the changes.

- I recently discovered a problem when using the "scale" option. When plotting certain figures with certain tools, extensive use of \texttt{pgfmathreciprocal} involves small computational errors but can add up and render the figures unfit. Here is how to proceed to avoid these problems:

1. On my side I introduced a patch proposed by Muzimuzhi that modifies \texttt{pgfmathreciprocal};

2. Another idea proposed by Muzimuzhi is to pass as an option for the \texttt{tikzpicture} environment this \texttt{/pgf/fpu/install only={reciprocal}} after loading of course the \texttt{fpu} library;

3. I have in the methods chosen to define my macros tried to avoid as much as possible the use of \texttt{pgfmathreciprocal};

4. There is still a foolproof method which consists in avoiding the use of \texttt{scale = \ldots}. It's quite easy if, like me, you only work with fixed points fixed at the beginning of your code. The size of your figure depends only on these fixed points so you just have to adapt the coordinates of these.

- Now \texttt{tkzDefCircle} gives two points as results: the center of the circle and a point of the circle. When a point of the circle is known, it is enough to use \texttt{tkzGetPoint} or \texttt{tkzGetFirstPoint} to get the center, otherwise \texttt{tkzGetPoints} will give you the center and a point of the circle. You can always get the length of the radius with \texttt{tkzGetLength}. I wanted to favor working with nodes and banish the appearance of numbers in the code.

- In order to isolate the definitions, I deleted or modified certain macros which are: \texttt{tkzDrawLine}, \texttt{tkzDrawTriangle}, \texttt{tkzDrawCircle}, \texttt{tkzDrawSemiCircle} and \texttt{tkzDrawRectangle};

Thus \texttt{tkzDrawSquare(A,B)} becomes \texttt{tkzDefSquare(A,B)\tkzGetPoints{C}{D}} then \texttt{tkzDrawPolygon(A,B,C,D)};

If you want to draw a circle, you can't do so \texttt{tkzDrawCircle[R](A,1)}. First you have to define the point through which the circle passes, so you have to do \texttt{tkzDefCircle[R](A,1) \tkzGetPoint{a}} and finally \texttt{tkzDrawCircle(A,a)}. Another possibility is to define a point on the circle \texttt{tkzDefShiftPoint[A](1,0){a}};

- The following macros \texttt{tkzDefCircleBy[orthogonal through]} and \texttt{tkzDefCircleBy[orthogonal from]} become \texttt{tkzDefCircle[orthogonal through]} and \texttt{tkzDefCircle[orthogonal from]};

- \texttt{tkzDefLine[euler](A,B,C)} is a macro that allows you to obtain the line of \texttt{Euler} when possible. \texttt{tkzDefLine[altitude]} is possible again, as well as \texttt{tkzDefLine[tangent at=A](0) and tkzDefLine[tangent from=P](0,A)} which did not works;

- \texttt{tkzDefTangent} is replaced by \texttt{tkzDefLine[tangent from = \ldots] or\tkzDefLine[tangent at = \ldots]};
– I added the macro `\tkzPicAngle[tikz options](A,B,C)` for those who prefer to use TikZ;

– The macro `\tkzMarkAngle` has been corrected;

– The macro linked to the `apollonius` option of the `\tkzDefCircle` command has been rewritten;

– (4.23) The macro `\tkzDrawSemiCircle` has been corrected;

– The order of the arguments of the macro `\tkzDefPointOnCircle` has changed: now it is center, angle and point or radius. I have added two options for working with radians which are `through in rad` and `R in rad`.

– I added the option `reverse` to the arcs paths. This allows to reverse the path and to reverse if necessary the arrows that would be present.

– I have unified the styles for the labels. There is now only `label style` left which is valid for points, segments, lines, circles and angles. I have deleted `label segment style`, `label line style` and `label angle style`.

– I added the macro `\tkzFillAngles` to use several angles.

– Correction option `return` with `\tkzProtractor`

As a reminder, the following changes have been made previously:

– `\tkzDrawMedian`, `\tkzDrawBisector`, `\tkzDrawAltitude`, `\tkzDrawMedians`, `\tkzDrawBisectors` et `\tkzDrawAltitudes` do not exist anymore. The creation and drawing separation is not respected so it is preferable to first create the coordinates of these points with `\tkzDefSpcTriangle[median]` and then to choose the ones you are going to draw with `\tkzDrawSegments` or `\tkzDrawLines`;

– `\tkzDrawTriangle` has been deleted. `\tkzDrawTriangle[equilateral]` was handy but it is better to get the third point with `\tkzDefTriangle[equilateral]` and then draw with `\tkzDrawPolygon`; idem for `\tkzDrawSquare` and `\tkzDrawGoldRectangle`;

– The circle inversion was badly defined so I rewrote the macro. The input arguments are always the center and a point of the circle, the output arguments are the center of the image circle and a point of the image circle or two points of the image line if the antecedent circle passes through the pole of the inversion. If the circle passes the inversion center, the image is a straight line, the validity of the procedure depends on the choice of the point on the antecedent circle;

– Correct allocation for gold sublime and euclide triangles;

– I added the option “next to” for the intersections LC and CC;

– Correction option isosceles right;

– (4.22 and 4.23) Correction of the macro `\tkzMarkAngle`;

– `\tkzDefMidArc(O,A,B)` gives the middle of the arc center O from A to B;

– Good news: Some useful tools have been added. They are present on an experimental basis and will undoubtedly need to be improved;
– The options "orthogonal from and through" depend now of \texttt{tkzDefCircleBy}

1. \texttt{tkzDotProduct(A,B,C)} computes the scalar product in an orthogonal reference system of the vectors \( \vec{A}, \vec{B}, \vec{C} \).
   \[ \texttt{tkzDotProduct(A,B,C)=aa'+bb'} \text{ if } \text{vec}(\vec{AB})=(a,b) \text{ and vec}(\vec{AC})=(a',b') \]

2. \texttt{tkzPowerCircle(A)(B,C)} power of point A with respect to the circle of center B passing through C;

3. \texttt{tkzDefRadicalAxis(A,B)(C,D)} Radical axis of two circles of center A and C;

4. (4.23) The macro \texttt{tkzDefRadicalAxis} has been completed

5. Some tests: \texttt{tkzIsOrtho(A,B,C) and tkzIsLinear(A,B,C)} The first indicates whether the lines \((A, B)\) and \((A, C)\) are orthogonal. The second indicates whether the points A, B and C are aligned;
   \[ \texttt{tkzIsLinear(A,B,C) if A,B,C are aligned then \texttt{tkzLineartrue you can use \texttt{iftkzLinear (idem for \texttt{tkzIsOrtho)}}} \]

6. A style for vectors has been added that you can of course modify
   \[ \texttt{tikzset{vector style/.style={>=Latex,->}};} \]

7. Now it's possible to add an arrow on a line or a circle with the option \texttt{tkz arrow}.

\section*{3. Changes with previous versions}

– I remind you that an important novelty is the recent replacement of the \texttt{fp} package by \texttt{xfp}. This is to improve the calculations a little bit more and to make it easier to use;

– First of all, you don't have to deal with TikZ the size of the bounding box. Early versions of \texttt{tkz-euclide} did not control the size of the bounding box. The bounding box is now controlled in each macro (hopefully) to avoid the use of \texttt{tkzInit} followed by \texttt{tkzClip};

– With \texttt{tkz-euclide} loads all objects, so there's no need to place \texttt{usetkzobj{all}};

– Added macros for the bounding box: \texttt{tkzSaveBB} \texttt{tkzClipBB} and so on;

– Logically most macros accept TikZ options. So I removed the "duplicate" options when possible thus the "label options" option is removed;

– The unit is now the cm;

– \texttt{tkzCalcLength} \texttt{tkzGetLength} gives result in cm;

– \texttt{tkzMarkArc} and \texttt{tkzLabelArc} are new macros;

– Now \texttt{tkzClipCircle} and \texttt{tkzClipPolygon} have an option \texttt{out}. To use this option you must have a Bounding Box that contains the object on which the Clip action will be performed. This can be done by using an object that encompasses the figure or by using the macro \texttt{tkzInit};

– The options \texttt{end} and \texttt{start} which allowed to give a label to a straight line are removed. You now have to use the macro \texttt{tkzLabelLine};

– Introduction of the libraries \texttt{quotes} and \texttt{angles}; it allows to give a label to a point, even if I am not in favour of this practice;
– The notion of vector disappears, to draw a vector just pass "->" as an option to `\tkzDrawSegment`;

– `\tkzDefIntSimilitudeCenter` and `\tkzDefExtSimilitudeCenter` do not exist anymore, now you need to use `\tkzDefSimilitudeCenter[int]` or `\tkzDefSimilitudeCenter[ext]`;

– `\tkzDefRandPointOn` is replaced by `\tkzGetRandPointOn`;

– An option of the macro `\tkzDefTriangle` has changed, in the previous version the option was "euclide" with an "e". Now it's "euclid";

– Random points are now in `tkz-euclide` and the macro `\tkzGetRandPointOn` is replaced by `\tkzDefRandPointOn`. For homogeneity reasons, the points must be retrieved with `\tkzGetPoint`;

– New macros have been added: `\tkzDrawSemiCircles, \tkzDrawPolygons, \tkzDrawTriangles`;

– Option "isosceles right" is a new option of the macro `\tkzDefTriangle`;

– Appearance of the macro `\usetkztool` which allows to load new "tools";

– The styles can be modified with the help of the following macros: `\tkzSetUpPoint, \tkzSetUpLine, \tkzSetUpArc, \tkzSetUpCompass, \tkzSetUpLabel` and `\tkzSetUpStyle`. The last one allows you to create a new style.
1. Working with lua : option lua

You can now use the "lua" option with `tkz-euclide` version 5. You just have to write in your preamble `\usepackage[\texttt{lua}]{tkz-euclide}`. Évidemment vous devrez compiler avec LuaLaTeX. Nothing changes for the syntax.

Without the option you can use `tkz-euclide` with the proposed code of version 4.25. This version is not yet finalized although the documentation you are currently reading has been compiled with this option.

Some information about the method used and the results obtained. Concerning the method, I considered two possibilities. The first one was simply to replace everywhere I could the calculations made by "xfp" or sometimes by "lua". This is how I went from "fp" to "xfp" and now to "lua". The second and more ambitious possibility would have to be associated to each point a complex number and to make the calculations on the complexes with "lua". Unfortunately for that I have to use libraries for which I don't know the license.

Otherwise the results are good. This documentation with "Lualatex" and "xfp" compiles in 47s while with "lua" it takes only 30s for 236 pages.

Another document of 61 pages is compiled 16s with "pdflatex" and "xfp" and 13s with "Lualatex" and "xfp".

This documentation compiles with `\usepackage{tkz-base}` and `\usepackage[\texttt{lua}]{tkz-euclide}` but I didn't test all the interactions thoroughly.

2. Installation

`tkz-euclide` is on the server of the CTAN\(^1\). If you want to test a beta version, just put the following files in a `texmf` folder that your system can find. You will have to check several points:

- The `tkz-euclide` folder must be located on a path recognized by \texttt{latex}.

- The `tkz-euclide` uses \texttt{xfp}.

- You need to have \texttt{PGF} installed on your computer. \texttt{tkz-euclide} use several libraries of \texttt{TiKZ}
  - angles,
  - arrows,
  - arrows.meta,
  - calc,
  - decorations,
  - decorations.markings,
  - decorations.pathreplacing,
  - decorations.shapes,
  - decorations.text,
  - decorations.pathmorphing,
  - intersections,
  - math,
  - plotmarks,
  - positioning,
  - quotes,
  - shapes.misc,
  - through

- This documentation and all examples were obtained with \texttt{lualatex} but \texttt{pdflatex} or \texttt{xelatex} should be suitable.

\(^1\) `tkz-euclide` is part of \texttt{TeXLive} and \texttt{tlngr} allows you to install them. This package is also part of \texttt{MiKTeX} under Windows.
3. Presentation and Overview

3.1. Why tkz-euclide?

My initial goal was to provide other mathematics teachers and myself with a tool to quickly create Euclidean geometry figures without investing too much effort in learning a new programming language. Of course, **tkz-euclide** is for math teachers who use **LaTeX** and makes it possible to easily create correct drawings by means of **LaTeX**.

It appeared that the simplest method was to reproduce the one used to obtain construction by hand. To describe a construction, you must, of course, define the objects but also the actions that you perform. It seemed to me that syntax close to the language of mathematicians and their students would be more easily understandable; moreover, it also seemed to me that this syntax should be close to that of **LaTeX**. The objects, of course, are points, segments, lines, triangles, polygons and circles. As for actions, I considered five to be sufficient, namely: define, create, draw, mark and label. The syntax is perhaps too verbose but it is, I believe, easily accessible. As a result, the students like teachers were able to easily access this tool.

3.2. Ti\textit{k}Z vs tkz-euclide

I love programming with Ti\textit{k}Z, and without Ti\textit{k}Z I would never have had the idea to create **tkz-euclide** but never forget that behind it there is Ti\textit{k}Z and that it is always possible to insert code from Ti\textit{k}Z. **tkz-euclide** doesn’t prevent you from using Ti\textit{k}Z. That said, I don’t think mixing syntax is a good thing.

There is no need to compare Ti\textit{k}Z and **tkz-euclide**. The latter is not addressed to the same audience as Ti\textit{k}Z. The first one allows you to do a lot of things, the second one only does geometry drawings. The first one can do everything the second one does, but the second one will more easily do what you want.

The main purpose is to define points to create geometrical figures. **tkz-euclide** allows you to draw the essential objects of Euclidean geometry from these points but it may be insufficient for some actions like coloring surfaces. In this case you will have to use Ti\textit{k}Z which is always possible.

Here are some comparisons between Ti\textit{k}Z and **tkz-euclide**. For this I will use the geometry examples from the PGFManual. The two most important Euclidean tools used by early Greeks to construct different geometrical shapes and angles were a compass and a straightedge. My idea is to allow you to follow step by step a construction that would be done by hand (with compass and straightedge) as naturally as possible.

3.2.1. Book I, proposition I _Euclid's Elements_

To construct an equilateral triangle on a given finite straight line.

Explanation:

The fourth tutorial of the **PgfManual** is about geometric constructions. T. Tantau proposes to get the drawing with its beautiful tool Ti\textit{k}Z. Here I propose the same construction with **tkz-elements**. The color of the Ti\textit{k}Z code is orange and that of **tkz-elements** is red.
\usepackage{tikz}
\usetikzlibrary{calc,intersections,through,backgrounds}
\usepackage{tkz-euclide}

How to get the line AB? To get this line, we use two fixed points.

\coordinate [label=left:$A$] (A) at (0,0);
\coordinate [label=right:$B$] (B) at (1.25,0.25);
\draw (A) -- (B);
\tkzDefPoint(0,0){A}
\tkzDefPoint(1.25,0.25){B}
\tkzDrawSegment(A,B)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}

We want to draw a circle around the points A and B whose radius is given by the length of the line AB.

\draw let \p1 = ($ (B) - (A) $),
\n2 = {veclen(\x1,\y1)} in
(A) circle (\n2)
(B) circle (\n2);
\tkzDrawCircles(A,B B,A)

The intersection of the circles $\mathcal{D}$ and $\mathcal{E}$

draw [name path=A--B] (A) -- (B);
node (D) [name path=D,draw,circle through=(B),label=left:$D$] at (A) {};
node (E) [name path=E,draw,circle through=(A),label=right:$E$] at (B) {};
path [name intersections={of=D and E, by={[label=above:$C$]C, [label=below:$C'$]C'}}];
draw [name path=C--C',red] (C) -- (C');
path [name intersections={of=A--B and C--C',by=F}];
ode [fill=red,inner sep=1pt,label=-45:$F$] at (F) {};
\tkzInterCC(A,B)(B,A) \tkzGetPoints{C}{X}

How to draw points:

\foreach \point in {A,B,C}
\fill [black,opacity=.5] (\point) circle (2pt);
\tkzDrawPoints[fill=gray,opacity=.5](A,B,C)

3.2.2. Complete code with tkz-euclide

We need to define colors
\colorlet{input}{red!80!black}
\colorlet{output}{red!70!black}
\colorlet{triangle}{orange!40}

\tkz-euclide AlterMundus
3. Presentation and Overview

3.2.3. Book I, Proposition II _Euclid's Elements_

To place a straight line equal to a given straight line with one end at a given point.

Explanation
In the first part, we need to find the midpoint of the straight line AB. With TikZ we can use the calc library

\coordinate [label=left:$A$] (A) at (0,0);
\coordinate [label=right:$B$] (B) at (1.25,0.25);
\draw (A) -- (B);
\node [fill=red,inner sep=1pt,label=below:$X$] (X) at ($ (A)!.5!(B) $) {};

With \texttt{tkz-euclide} we have a macro \texttt{tkzDefMidPoint}, we get the point X with \texttt{tkzGetPoint} but we don't need this point to get the next step.

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(1.25+rand(),0.25+rand()){B}
\tkzInterCC(A,B)(B,A) \tkzGetPoints{C}{X}
\tkzFillPolygon[triangle,opacity=.5](A,B,C)
\tkzDrawSegment[input](A,B)
\tkzDrawSegments[red](A,C B,C)
\tkzDrawCircles[help lines](A,B B,A)
\tkzDrawPoints[fill=gray,opacity=.5](A,B,C)
\tkzLabelPoints(A,B)
\tkzLabelCircle[below=12pt](A,B)(180){\mathcal{D}}
\tkzLabelCircle[above=12pt](B,A)(180){\mathcal{E}}
\tkzLabelPoint[above,red](C){$C$}
\end{tikzpicture}
Then we need to construct a triangle equilateral. It's easy with \texttt{tkz-euclide}. With TikZ you need some effort because you need to use the midpoint X to get the point D with trigonometry calculation.

\begin{verbatim}
\node [fill=red,inner sep=1pt,label=below:$X$] (X) at ($ (A)!.5!(B) $) {}; \
\node [fill=red,inner sep=1pt,label=above:$D$] (D) at ($ (X) ! {sin(60)*2} ! 90:(B) $) {}; \
\draw (A) -- (D) -- (B); \\
\tkzDefTriangle[equilateral](A,B) \tkzGetPoint{D}
\end{verbatim}

We can draw the triangle at the end of the picture with

\begin{verbatim}
\tkzDrawPolygon{A,B,C}
\end{verbatim}

We know how to draw the circle $\mathcal{H}$ around B through C and how to place the points E and F

\begin{verbatim}
\node (H) [label=135:$H$,draw,circle through=(C)] at (B) {}; \
\draw (D) -- ($ (D) ! 3.5 ! (B) $) coordinate [label=below:$F$] (F); \
\draw (D) -- ($ (D) ! 2.5 ! (A) $) coordinate [label=below:$E$] (E); \\
\tkzDrawCircle(B,C) \
\tkzDrawLines[add=0 and 2](D,A D,B)
\end{verbatim}

We can place the points E and F at the end of the picture. We don't need them now.

Intersecting a Line and a Circle : here we search the intersection of the circle around B through C and the line DB. The infinite straight line DB intercepts the circle but with TikZ we need to extend the lines DB and that can be done using partway calculations. We get the point F and BF or DF intercepts the circle

\begin{verbatim}
\node (H) [label=135:$H$,draw,circle through=(C)] at (B) {}; \
\path let \p1 = ($ (B) - (C) $) in \
\coordinate [label=left:$G$] (G) at ($ (B) ! veclen(\x1,\y1) ! (F) $); \
\fill[red,opacity=.5] (G) circle (2pt);
\end{verbatim}

Like the intersection of two circles, it's easy to find the intersection of a line and a circle with \texttt{tkz-euclide}. We don't need F

\begin{verbatim}
\tkzInterLC(B,D)(B,C)\tkzGetFirstPoint{G}
\end{verbatim}

There are no more difficulties. Here the final code with some simplifications. We draw the circle $\mathcal{H}$ with center D and passing through G. It intersects the line AD at point L. AL = BC.

\begin{verbatim}
\tkzDrawCircle(D,G) \
\tkzInterLC(D,A)(D,G)\tkzGetSecondPoint{L}
\end{verbatim}
3. Presentation and Overview

3.3. \texttt{tkz-euclide 4 vs tkz-euclide 3}

Now I am no longer a Mathematics teacher, and I only spend a few hours studying geometry. I wanted to avoid multiple complications by trying to make \texttt{tkz-euclide} independent of \texttt{tkz-base}. Thus was born \texttt{tkz-euclide 4}. The latter is a simplified version of its predecessor. The macros of \texttt{tkz-euclide 3} have been retained. The unit is now cm. If you need some macros from \texttt{tkz-base}, you may need to use the \texttt{tkzInit}.

3.4. \texttt{tkz-euclide 5 vs tkz-euclide 4}

Rien ne change pour l’utilisateur. La compilation doit être effectuée avec le moteur LuaLaTeX et les résultats sont plus précis et obtenus plus rapidement. Il suffit de charger \texttt{tkz-euclide 5} comme ceci: \texttt{\usepackage[lua]{tkz-euclide}}.

3.5. How to use the \texttt{tkz-euclide} package ?

3.5.1. Let’s look at a classic example

In order to show the right way, we will see how to build an equilateral triangle. Several possibilities are open to us, we are going to follow the steps of Euclid.

- First of all, you have to use a document class. The best choice to test your code is to create a single figure with the class \texttt{standalone}.

\begin{verbatim}
\documentclass{standalone}
\end{verbatim}

- Then load the \texttt{tkz-euclide} package:

\begin{verbatim}
\usepackage{tkz-euclide} or \usepackage[lua]{tkz-euclide}
\end{verbatim}

You don’t need to load TikZ because the \texttt{tkz-euclide} package works on top of TikZ and loads it.

- Start the document and open a TikZ picture environment:

\begin{verbatim}
\begin{document}
\begin{tikzpicture}
\end{tikzpicture}
\end{document}
\end{verbatim}
– Now we define two fixed points:

\begin{verbatim}
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,2){B}
\end{verbatim}

– Two points define two circles, let’s use these circles: circle with center A through B and circle with center B through A. These two circles have two points in common.

\begin{verbatim}
\tkzInterCC(A,B)(B,A)
\end{verbatim}

We can get the points of intersection with

\begin{verbatim}
\tkzGetPoints{C}{D}
\end{verbatim}

– All the necessary points are obtained, we can move on to the final steps including the plots.

\begin{verbatim}
\tkzDrawCircles[gray,dashed](A,B B,A)
\tkzDrawPolygon(A,B,C)% The triangle
\end{verbatim}

– Draw all points A, B, C and D:

\begin{verbatim}
\tkzDrawPoints(A,...,D)
\end{verbatim}

– The final step, we print labels to the points and use options for positioning:

\begin{verbatim}
\tkzLabelSegments[swap](A,B){$c$}
\tkzLabelPoints(A,B,D)
\tkzLabelPoints[above](C)
\end{verbatim}

– We finally close both environments

\begin{verbatim}
\end{tikzpicture}
\end{document}

– The complete code

\begin{verbatim}
\begin{tikzpicture}[scale=.5]
% fixed points
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,2){B}
% calculus
\tkzInterCC(A,B)(B,A)
\tkzGetPoints{C}{D}
% drawings
\tkzDrawCircles(A,B B,A)
\tkzDrawPolygon(A,B,C)% The triangle
\tkzDrawPoints(A,...,D)
% marking
\tkzMarkSegments{mark=||}{A,B B,C C,A}
% labelling
\tkzLabelSegments[swap]{A,B}{$c$}
\tkzLabelPoints(A,B,D)
\tkzLabelPoints[above]{C}
\end{tikzpicture}
\end{verbatim}
3.5.2. Part I: golden triangle

Let's analyze the figure

1. CBD and DBE are isosceles triangles;
2. BC = BE and (BD) is a bisector of the angle CBE;
3. From this we deduce that the CBD and DBE angles are equal and have the same measure $\alpha$
   
   $\overline{BAC} + \overline{ABC} + \overline{BCA} = 180^\circ$ in the triangle BAC
   
   then
   
   $3\alpha + \overline{BCA} = 180^\circ$ in the triangle CBD
   
   or
   
   $\overline{BCA} = 90^\circ - \alpha/2$
4. Finally
   
   the triangle CBD is a "golden" triangle.

How construct a golden triangle or an angle of 36°?

1. We place the fixed points C and D. \texttt{\tkzDefPoint(0,0){C}} and \texttt{\tkzDefPoint(4,0){D}};
2. We construct a square CDef and we construct the midpoint m of [Cf];
   
   We can do all of this with a compass and a rule;
3. Then we trace an arc with center m through e. This arc cross the line (Cf) at n;
4. Now the two arcs with center C and D and radius Cn define the point B.
After building the golden triangle $BCD$, we build the point $A$ by noticing that $BD = DA$. Then we get the point $E$ and finally the point $F$. This is done with already intersections of defined objects (line and circle).

### 3.5.3. Part II: two others methods with golden and euclid triangle

\texttt{tkz-euclide} knows how to define a "golden" or "euclide" triangle. We can define $BCD$ and $BCA$ like gold triangles.

\begin{tikzpicture}
\tkzDefPoint(0,0){C}
\tkzDefPoint(4,0){D}
\tkzDefTriangle[golden](C,D)
\tkzGetPoint{B}
\tkzDefTriangle[golden](B,C)
\tkzGetPoint{A}
\tkzInterLC(B,A)(B,D) \tkzGetSecondPoint{E}
\tkzDrawPolygon(B,...,D)
\tkzDrawPoints(C,D,B)
\tkzMarkRightAngle(B,F,C)
\tkzMarkAngles(C,B,D E,A,D){$\alpha$}
\tkzLabelPoints[below](A,C,D,E)
\tkzLabelPoints[above right](B,F)
\end{tikzpicture}

Here is a final method that uses rotations:
\begin{tikzpicture}
\tkzDefPoint(0,0){C}
\tkzDefPoint(2,6){B}
% We get D and E with a rotation
\tkzDefPointBy[rotation= center B angle 36](C) \tkzGetPoint{D}
\tkzDefPointBy[rotation= center B angle 72](C) \tkzGetPoint{E}
% To get A we use an intersection of lines
\tkzInterLL(B,E)(C,D) \tkzGetPoint{A}
\tkzInterLL(C,E)(B,D) \tkzGetPoint{H}
% drawing
\tkzDrawArc[delta=10](B,C)(E)
\tkzDrawPolygon(C,B,D)
\tkzDrawSegments(D,A B,A C,E)
% angles
\tkzMarkAngles(C,B,D E,A,D) \%this is to draw the arcs
\tkzLabelAngles[pos=1.5](C,B,D E,A,D){\$\alpha\$}
\tkzMarkRightAngle(B,H,C)
\tkzDrawPoints(A,...,E)
% Label only now
\tkzLabelPoints[below left](C,A)
\tkzLabelPoints[below right](D)
\tkzLabelPoints[above](B,E)
\end{tikzpicture}

### 3.5.4. Complete but minimal example

A unit of length being chosen, the example shows how to obtain a segment of length $\sqrt{a}$ from a segment of length $a$, using a ruler and a compass.

IB = a, AI = 1
3. Presentation and Overview

\begin{tikzpicture}[scale=1,ra/.style={fill=gray!20}]
% fixed points
\tkzDefPoint(0,0){A}
\tkzDefPoint(1,0){I}
% calculation
\tkzDefPointBy[homothety=center A ratio 10 ](I) \tkzGetPoint{B}
\tkzDefMidPoint(A,B) \tkzGetPoint{M}
\tkzDefPointWith[orthogonal](I,M) \tkzGetPoint{H}
\tkzInterLC(I,H)(M,B) \tkzGetFirstPoint{C}
\tkzDrawSegment[style=orange](I,C)
\tkzDrawArc(M,B)(A)
\tkzDrawSegment[dim={$1$,-16pt,}](A,I)
\tkzDrawSegment[dim={$a/2$,-10pt,}](I,M)
\tkzDrawSegment[dim={$a/2$,-16pt,}](M,B)
\tkzMarkRightAngle[ra](A,I,C)
\tkzDrawPoints(I,A,B,C,M)
\tkzLabelPoint[left](A){$A(0,0)$}
\tkzLabelPoints[above right](I,M)
\tkzLabelPoint[above left](C)
\tkzLabelSegment[right=4pt](I,C){$\sqrt{a^2}=a \ (a>0)$}
\end{tikzpicture}

Comments

– The Preamble

Let us first look at the preamble. If you need it, you have to load xcolor before \texttt{tkz-euclide}, that is, before TikZ. TikZ may cause problems with the active characters, but... provides a library in its latest version that's supposed to solve these problems \texttt{babel}.

\documentclass{standalone} % or another class
\usepackage{xcolor} % before tikz or \texttt{tkz-euclide} if necessary
\usepackage{tkz-euclide} % no need to load TikZ
\usetikzlibrary{babel} % if there are problems with the active characters

The following code consists of several parts:

– Definition of fixed points: the first part includes the definitions of the points necessary for the construction, these are the fixed points. The macros \texttt{\tkzInit} and \texttt{\tkzClip} in most cases are not necessary.

\tkzDefPoint(0,0){A}
\tkzDefPoint(1,0){I}

– The second part is dedicated to the creation of new points from the fixed points; a B point is placed at 10 cm from A. The middle of [AB] is defined by M and then the orthogonal line to the (AB) line is searched for at the I point. Then we look for the intersection of this line with the semi-circle of center M passing through A.

\tkzDefPointBy[homothety=center A ratio 10 ](I) \tkzGetPoint{B}
\tkzDefMidPoint(A,B) \tkzGetPoint{M}
\tkzDefPointWith[orthogonal](I,M) \tkzGetPoint{H}
\tkzInterLC(I,H)(M,B) \tkzGetFirstPoint{C}
4. The Elements of tkz code

To work with my package, you need to have notions of \LaTeX as well as \TiKZ. In this paragraph, we start looking at the "rules" and "symbols" used to create a figure with tkz-euclide.

4.1. Objects and language

The primitive objects are points. You can refer to a point at any time using the name given when defining it. (it is possible to assign a different name later on).

To get new points you will use macros. \tkz-euclide macros have a name beginning with tkz. There are four main categories starting with: \tkzDefPoint, \tkzDrawPoint, \tkzMarkRightAngle and \tkzLabelPoint. The used points are passed as parameters between parentheses while the created points are between braces.

The code of the figures is placed in an environment \texttt{tikzpicture}.

Contrary to \TiKZ, you should not end a macro with ";". We thus lose the important notion which is the \emph{path}.

However, it is possible to place some code between the macros tkz-euclide. Among the first category, \tkzDefPoint allows you to define fixed points. It will be studied in detail later. Here we will see in detail the macro \texttt{tkzDefTriangle}.

This macro makes it possible to associate to a pair of points a third point in order to define a certain triangle \texttt{tkzDefTriangle(A,B)}. The obtained point is referenced \texttt{tkzPointResult} and it is possible to choose another reference with \texttt{tkzGetPoint{C}} for example.

\texttt{tkzDefTriangle[euclid](A,B) \tkzGetPoint{C}}

Parentheses are used to pass arguments. In \texttt{(A,B)} A and B are the points with which a third will be defined. However, in \texttt{C} we use braces to retrieve the new point.

In order to choose a certain type of triangle among the following choices: equilateral, isosceles right, half, pythagoras, school, golden or sublime, euclid, gold, cheops... and two angles you just have to choose between hooks, for example:
4. The Elements of tkz code

\begin{tikzpicture}[scale=.5]
\tkzDefPoints{0/0/A,8/0/B}
\foreach \tr in {golden, equilateral}
{\tkzDefTriangle[\tr](A,B) \tkzGetPoint{C}
\tkzDrawPoint(C)
\tkzLabelPoint[right](C){\tr}
\tkzDrawSegments(A,C,C,B)}
\tkzDrawPoints(A,B)
\tkzDrawSegments(A,B)
\tkzLabelPoints(A,B)
\end{tikzpicture}

4.2. Notations and conventions

I deliberately chose to use the geometric French and personal conventions to describe the geometric objects represented. The objects defined and represented by \texttt{tkz-euclide} are points, lines and circles located in a plane. They are the primary objects of Euclidean geometry from which we will construct figures. According to \textbf{Euclid}, these figures will only illustrate pure ideas produced by our brain. Thus a point has no dimension and therefore no real existence. In the same way the line has no width and therefore no existence in the real world. The objects that we are going to consider are only representations of ideal mathematical objects. \texttt{tkz-euclide} will follow the steps of the ancient Greeks to obtain geometrical constructions using the ruler and the compass.

Here are the notations that will be used:

- The points are represented geometrically either by a small disc or by the intersection of two lines (two straight lines, a straight line and a circle or two circles). In this case, the point is represented by a cross.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/2/B}
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
\end{tikzpicture}

or else

\begin{tikzpicture}
\tkzSetUpPoint[shape=cross, color=red]
\tkzDefPoints{0/0/A,4/2/B}
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
\end{tikzpicture}

The existence of a point being established, we can give it a label which will be a capital letter (with some exceptions) of the Latin alphabet such as A, B or C. For example:

- O is a center for a circle, a rotation, etc.;
- M defined a midpoint;
- H defined the foot of an altitude;
- P' is the image of P by a transformation;
It is important to note that the reference name of a point in the code may be different from the label to designate it in the text. So we can define a point A and give it as label \( P \). In particular the style will be different, point A will be labeled \( A \).

\[
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDrawPoints(A)
\tkzLabelPoint(A){$P$}
\end{tikzpicture}
\]

Exceptions: some points such as the middle of the sides of a triangle share a characteristic, so it is normal that their names also share a common character. We will designate these points by \( M_a, M_b, \) and \( M_c \) or \( M_A, M_B, \) and \( M_C \).

In the code, these points will be referred to as: \( M_A, M_B \) and \( M_C \).

Another exception relates to intermediate construction points which will not be labelled. They will often be designated by a lowercase letter in the code.

- The line segments are designated by two points representing their ends in square brackets: \([AB]\).
- The straight lines are in Euclidean geometry defined by two points so A and B define the straight line \((AB)\). We can also designate this straight line using the Greek alphabet and name it \((\delta)\) or \((\Delta)\). It is also possible to designate the straight line with lowercase letters such as \( d \) and \( d' \).
- The semi-straight line is designated as follows \([AB)\).
- Relation between the straight lines. Two perpendicular \((AB)\) and \((CD)\) lines will be written \((AB) \perp (CD)\) and if they are parallel we will write \((AB) \parallel (CD)\).
- The lengths of the sides of triangle ABC are \( AB, AC \) and \( BC \). The numbers are also designated by a lowercase letter so we will write: \( AB = c, AC = b \) and \( BC = a \). The letter \( a \) is also used to represent an angle, and \( r \) is frequently used to represent a radius, \( d \) a diameter, \( l \) a length, \( d \) a distance.
- Polygons are designated afterwards by their vertices so \( ABC \) is a triangle, \( EFGH \) a quadrilateral.
- Angles are generally measured in degrees (ex \( 60^\circ \)) and in an equilateral \( ABC \) triangle we will write \( \angle ABC = 60^\circ \).
- The arcs are designated by their extremities. For example if A and B are two points of the same circle then \( AB \).
- Circles are noted either \( \mathcal{C} \) if there is no possible confusion or \( \mathcal{C} (O ; a) \) for a circle with center \( O \) and passing through the point \( A \) or \( \mathcal{C} (O ; 1) \) for a circle with center \( O \) and radius \( 1 \) cm.
- Name of the particular lines of a triangle: I used the terms bisector, bisector out, mediator (sometimes called perpendicular bisectors), altitude, median and symmedian.
- \((x_1, y_1)\) coordinates of the point \( A_1 \), \((x_A, y_A)\) coordinates of the point \( A \).

4.3. Set, Calculate, Draw, Mark, Label

The title could have been: Separation of Calculus and Drawings

When a document is prepared using the \LaTeX{} system, the source code of the document can be divided into two parts: the document body and the preamble. Under this methodology, publications can be structured, styled and typeset with minimal effort. I propose a similar methodology for creating figures with \texttt{tkz-euclide}.

The first part defines the fixed points, the second part allows the creation of new points. \texttt{Set} and \texttt{Calculate} are the two main parts. All that is left to do is to draw (or fill), mark and label. It is possible that \texttt{tkz-euclide} is insufficient for some of these latter actions but you can use TikZ.
One last remark that I think is important, it is best to avoid introducing coordinates within a code as much as possible. I think that the coordinates should appear at the beginning of the code with the fixed points. Then the use of references is recommended. Most macros have the option \texttt{nodes} or \texttt{with nodes}.

I also think it's best to define the styles of the different objects from the beginning.

\section*{5. About this documentation and the examples}

It is obtained by compiling with "lualatex". I use a class \texttt{doc.cls} based on \texttt{scrartcl}.

Below the list of styles used in the document. To understand how to use the styles see the section 37

\begin{verbatim}
\tkzSetUpColors[background=white,text=black]
\tkzSetUpCompass[color=orange, line width=.2pt,delta=10]
\tkzSetUpArc[color=gray,line width=.2pt]
\tkzSetUpPoint[size=2,color=teal]
\tkzSetUpLine[line width=.2pt,color=teal]
\tkzSetUpStyle[color=orange,line width=.2pt]{new}
\tikzset{every picture/.style={line width=.2pt}}
\tikzset{label angle style/.append style={color=teal,font=\footnotesize}}
\tikzset{label style/.append style={below,color=teal,font=\scriptsize}}
\end{verbatim}

Some examples use predefined styles like

\begin{verbatim}
\tikzset{new/.style={color=orange,line width=.2pt}}
\end{verbatim}
Part II.

Setting
6. First step: fixed points

The first step in a geometric construction is to define the fixed points from which the figure will be constructed. The general idea is to avoid manipulating coordinates and to prefer to use the references of the points fixed in the first step or obtained using the tools provided by the package. Even if it's possible, I think it's a bad idea to work directly with coordinates. Preferable is to use named points.

*tkz-euclide* uses macros and vocabulary specific to geometric construction. It is of course possible to use the tools of TikZ but it seems more logical to me not to mix the different syntaxes.

A point in *tkz-euclide* is a particular "node" for TikZ. In the next section we will see how to define points using coordinates. The style of the points (color and shape) will not be discussed. You will find some indications in some examples; for more information you can read the following section 37.

7. Definition of a point : \texttt{\tkzDefPoint} or \texttt{\tkzDefPoints}

Points can be specified in any of the following ways:

- Cartesian coordinates;
- Polar coordinates;
- Named points;
- Relative points.

A point is defined if it has a name linked to a unique pair of decimal numbers. Let (x,y) or (a:d) i.e. (x abscissa, y ordinate) or (a angle: d distance). This is possible because the plan has been provided with an orthonormed Cartesian coordinate system. The working axes are (ortho)normed with unity equal to 1 cm.

The Cartesian coordinate (a,b) refers to the point a centimeters in the x-direction and b centimeters in the y-direction.

A point in polar coordinates requires an angle $\alpha$, in degrees, and a distance d from the origin with a dimensional unit by default it’s the cm.

The \texttt{\tkzDefPoint} macro is used to define a point by assigning coordinates to it. This macro is based on \texttt{\coordinate}, a macro of TiKZ. It can use TiKZ-specific options such as \texttt{shift}. If calculations are required then the \texttt{xfp} package is chosen. We can use Cartesian or polar coordinates.

Cartesian coordinates

\begin{tikzpicture}[scale=1]
\tkzInit[xmax=5,ymax=5]
\tkzDrawX[>=latex]
\tkzDrawY[>=latex]
\tkzDefPoints{0/0/O,1/0/I,0/1/J}
\tkzDefPoint(3,4){A}
\tkzDrawPoints(O,A)
\tkzLabelPoint[above](A){$A_1 (x_1,y_1)$}
\tkzShowPointCoord[xlabel=$x_1$,
ylabel=$y_1$](A)
\tkzLabelPoints(O,I)
\tkzLabelPoints[above](J)
\end{tikzpicture}

Polar coordinates

\begin{tikzpicture}[scale=1]
\tkzInit[xmax=5,ymax=5]
\tkzDrawX[>=latex]
\tkzDrawY[>=latex]
\tkzDefPoints{0/0/O,1/0/I,0/1/J}
\tkzDefPoint(40:4){P}
\tkzDrawSegment[dim={$d$,16pt,above=6pt}](O,P)
\tkzDrawPoints(O,P)
\tkzMarkAngle[mark=none,->](I,O,P)
\tkzFillAngle[opacity=.5](I,O,P)
\tkzLabelAngle[pos=1.25](I,O,P){$\alpha$}
\tkzLabelPoint[right](P){$P (\alpha : d )$}
\tkzDrawPoints[shape=cross](I,J)
\tkzLabelPoints(O,I)
\tkzLabelPoints[above](J)
\end{tikzpicture}
7. Definition of a point: \texttt{tkzDefPoint} or \texttt{tkzDefPoints}

\begin{itemize}
  \item \texttt{tkzDefPoint[⟨local options⟩]((x,y)){⟨ref⟩}} or \texttt{(⟨α: d⟩){⟨ref⟩}}
\end{itemize}

The obligatory arguments of this macro are two dimensions expressed with decimals, in the first case they are two measures of length, in the second case they are a measure of length and the measure of an angle in degrees. Do not confuse the reference with the name of a point. The reference is used by calculations, but frequently, the name is identical to the reference.

\begin{tabular}{lll}
\hline
arguments & default & definition \\
\hline
(x,y) & no default & x and y are two dimensions, by default in cm. \\
(α:d) & no default & α is an angle in degrees, d is a dimension \\
{⟨ref⟩} & no default & Reference assigned to the point: A, T_a , P1 or P \\
\hline
\end{tabular}

7.1. Defining a named point \texttt{tkzDefPoint}

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=5] % limits the size of the axes
\tkzDrawX[>=latex]
\tkzDrawY[>=latex]
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
\tkzDefPoint(0,3){C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}

7.1.1. Cartesian coordinates

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=5] % limits the size of the axes
\tkzDrawX[>=latex]
\tkzDrawY[>=latex]
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
\tkzDefPoint(0,3){C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}
7. Definition of a point: \texttt{tkzDefPoint} or \texttt{tkzDefPoints}

7.1.2. Calculations with xfp

\begin{tikzpicture}[scale=1]
\tkzInit[xmax=4,ymax=4]
\tkzDrawX\tkzDrawY
\tkzDefPoint(-1+2,sqrt(4)){O}
\tkzDefPoint({3*ln(exp(1))},{exp(1)}){A}
\tkzDefPoint({4*sin(pi/6)},{4*cos(pi/6)}){B}
\tkzDrawPoints(O,B,A)
\end{tikzpicture}

7.1.3. Polar coordinates

\begin{tikzpicture}
\foreach \an [count=i] in {0,60,...,300}
{ \tkzDefPoint(\an:3){A_\i}}
\tkzDrawPolygon(A_1,A_...,A_6)
\tkzDrawPoints(A_1,A_...,A_6)
\end{tikzpicture}

7.1.4. Relative points

First, we can use the \texttt{scope} environment from TikZ. In the following example, we have a way to define an equilateral triangle.

\begin{tikzpicture}[scale=1]
\begin{scope}[rotate=30]
\tkzDefPoint(2,3){A}
\begin{scope}[shift=(A)]
\tkzDefPoint(90:5){B}
\tkzDefPoint(30:5){C}
\end{scope}
\end{scope}
\tkzDrawPolygon(A,B,C)
\tkzLabelPoints[above](B,C)
\tkzLabelPoints[below](A)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}

7.2. Point relative to another: \texttt{tkzDefShiftPoint}
7. Definition of a point: \texttt{\tkzDefPoint} or \texttt{\tkzDefPoints}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x,y)</td>
<td>no default</td>
<td>x and y are two dimensions, by default in cm.</td>
</tr>
<tr>
<td>((\alpha):d)</td>
<td>no default</td>
<td>(\alpha) is an angle in degrees, d is a dimension</td>
</tr>
<tr>
<td>{ref}</td>
<td>no default</td>
<td>Reference assigned to the point: A, T_a, P1 or P1</td>
</tr>
</tbody>
</table>

\begin{center}
\begin{tabular}{|c|c|}
\hline
options & default \tabularnewline \hline
[pt] & no default \\tkzDefShiftPoint[A](0:4){B} \\
\hline
\end{tabular}
\end{center}

7.2.1. Isosceles triangle

This macro allows you to place one point relative to another. This is equivalent to a translation. Here is how to construct an isosceles triangle with main vertex A and angle at vertex of 30°.

\begin{tikzpicture}
\usetikzlibrary{calc}
\begin{scope}[rotate=-30]
\tkzDefPoint(2,3){A}
\tkzDefShiftPoint[A](0:4){B}
\tkzDefShiftPoint[A](30:4){C}
\tkzDrawSegments(A,B B,C C,A)
\tkzMarkSegments[mark=]{A,B A,C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints[right](B,C)
\tkzLabelPoints[above left](A)
\end{scope}
\end{tikzpicture}

7.2.2. Equilateral triangle

Let’s see how to get an equilateral triangle (there is much simpler)

\begin{tikzpicture}[scale=1]
\tkzDefPoint(2,3){A}
\tkzDefShiftPoint[A](30:3){B}
\tkzDefShiftPoint[A](-30:3){C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints[right](B,C)
\tkzLabelPoints[above left](A)
\tkzMarkSegments[mark=]{A,B A,C B,C}
\end{tikzpicture}

7.2.3. Parallelogram

There’s a simpler way

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(30:3){B}
\tkzDefShiftPointCoord[B](10:2){C}
\tkzDefShiftPointCoord[A](10:2){D}
\tkzDrawPolygon(A,...,D)
\tkzDrawPoints(A,...,D)
\end{tikzpicture}
7. Definition of a point: \texttt{\textbackslash{}tkzDefPoint or \textbackslash{}tkzDefPoints}

7.3. Definition of multiple points: \texttt{\textbackslash{}tkzDefPoints}

\begin{verbatim}
\texttt{\textbackslash{}tkzDefPoints\{\textbackslash{}local\textbackslash{}options\}\{x_1/y_1/n_1,x_2/y_2/r_2,\ldots\}]
\end{verbatim}

\textit{x}_i and \textit{y}_i are the coordinates of a referenced point \textit{r}_i

\begin{tabular}{ll}
arguments & default example \\
\texttt{x_i/y_i/r_i} & \texttt{\textbackslash{}tkzDefPoints\{0/0,2/2/A\}} \\
\texttt{options} & default definition \\
\texttt{shift} & no default Adds (x,y) or (a:d) to all coordinates
\end{tabular}

7.4. Create a triangle

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,4/0/B,4/3/C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}

7.5. Create a square

Note here the syntax for drawing the polygon.

\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,2/0/B,2/2/C,0/2/D}
\tkzDrawPolygon(A,...,D)
\tkzDrawPoints(A,...,D)
\end{tikzpicture}

\texttt{tkz-euclide}\hspace{1cm}\texttt{AlterMundus}
Part III.

Calculating
Now that the fixed points are defined, we can with their references using macros from the package or macros that you will create get new points. The calculations may not be apparent but they are usually done by the package. You may need to use some mathematical constants, here is the list of constants defined by the package. You may need to use some mathematical constants, here is the list of constants defined by the package.

8. Auxiliary tools

8.1. Constants

\texttt{tkz-euclide} knows some constants, here is the list:

\begin{tabular}{l}
\texttt{\def\tkzPhi{1.618034}}
\texttt{\def\tkzInvPhi{0.618034}}
\texttt{\def\tkzSqrtPhi{1.27202}}
\texttt{\def\tkzSqrTwo{1.414213}}
\texttt{\def\tkzSqrThree{1.7320508}}
\texttt{\def\tkzSqrFive{2.2360679}}
\texttt{\def\tkzSqrTwobyTwo{0.7071065}}
\texttt{\def\tkzPi{3.1415926}}
\texttt{\def\tkzEuler{2.71828182}}
\end{tabular}

8.2. New point by calculation

When a macro of \texttt{tkz\_nameofpack} creates a new point, it is stored internally with the reference \texttt{tkzPointResult}. You can assign your own reference to it. This is done with the macro \texttt{tkzGetPoint}. A new reference is created, your choice of reference must be placed between braces.

\begin{tabular}{l}
\texttt{\tkzGetPoint{⟨ref⟩}}
\end{tabular}

If the result is in \texttt{tkzPointResult}, you can access it with \texttt{tkzGetPoint}.

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ref</td>
<td>no default</td>
<td>\tkzGetPoint{M} see the next example</td>
</tr>
</tbody>
</table>

Sometimes you need to get two points. It’s possible with

\begin{tabular}{l}
\texttt{\tkzGetPoints{⟨ref1⟩}{⟨ref2⟩}}
\end{tabular}

The result is in \texttt{tkzPointFirstResult} and \texttt{tkzPointSecondResult}.

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨ref1,ref2⟩</td>
<td>no default</td>
<td>\tkzGetPoints{M,N} It’s the case with \tkzInterCC</td>
</tr>
</tbody>
</table>

If you need only the first or the second point you can also use:

\begin{tabular}{l}
\texttt{\tkzGetFirstPoint{⟨ref⟩}}
\end{tabular}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ref1</td>
<td>no default</td>
<td>\tkzGetFirstPoint{M}</td>
</tr>
</tbody>
</table>

\texttt{tkz-euclide}
9. Special points

Here are some special points.

9.1. Middle of a segment \( \texttt{tkzDefMidPoint} \)

It is a question of determining the middle of a segment.

\[
\begin{tikzpicture}
[\text{scale=1}]
\texttt{tkzDefPoint}(2,3){A}
\texttt{tkzDefPoint}(6,2){B}
\texttt{tkzDefMidPoint(A,B)}
\texttt{tkzGetPoint(M)}
\texttt{tkzDrawSegment(A,B)}
\texttt{tkzDrawPoints(A,B,M)}
\texttt{tkzLabelPoints[below](A,B,M)}
\end{tikzpicture}
\]

9.2. Golden ratio \( \texttt{tkzDefGoldenRatio} \)

From Wikipedia: In mathematics, two quantities are in the golden ratio if their ratio is the same as the ratio of their sum to the larger of the two quantities. Expressed algebraically, for quantities \( a, b \) such as \( a > b > 0; a + b \) is to \( a \) as \( a \) is to \( b \).
9. Special points

\[
a + b = \frac{a}{b} = \phi = \frac{1 + \sqrt{5}}{2}
\]

One of the two solutions to the equation \(x^2 - x - 1 = 0\) is the golden ratio \(\phi\), \(\phi = \frac{1 + \sqrt{5}}{2}\).

9.2.1. Use the golden ratio to divide a line segment

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/C}
\tkzDefMidPoint(A,C) \tkzGetPoint{I}
\tkzDefPointWith[linear,K=\tkzInvPhi](A,C) \tkzGetPoint{B}
\tkzDrawSegments(A,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B,C)
\end{tikzpicture}

9.2.2. Golden arbelos

\begin{tikzpicture}[scale=.6]
\tkzDefPoints{0/0/A,10/0/B}
\tkzDefGoldenRatio(A,B) \tkzGetPoint{C}
\tkzDefMidPoint(A,B) \tkzGetPoint{O_1}
\tkzDefMidPoint(A,C) \tkzGetPoint{O_2}
\tkzDefMidPoint(C,B) \tkzGetPoint{O_3}
\tkzDrawSemiCircles[fill=purple!15](O_1,B)
\tkzDrawSemiCircles[fill=teal!15](O_2,C O_3,B)
\end{tikzpicture}

It is also possible to use the following macro.

9.3. Barycentric coordinates with \texttt{tkzDefBarycentricPoint}

\(p_1, p_2, \ldots, p_n\) being \(n\) points, they define \(n\) vectors \(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_n\) with the origin of the referential as the common endpoint. \(a_1, a_2, \ldots, a_n\) are \(n\) numbers, the vector obtained by:

\[
\frac{a_1 \vec{v}_1 + a_2 \vec{v}_2 + \cdots + a_n \vec{v}_n}{a_1 + a_2 + \cdots + a_n}
\]

defines a single point.

\begin{tikzpicture}
\tkzDefBarycentricPoint{(p1=a_1, p2=a_2, \ldots)}
\end{tikzpicture}

\textbf{Arguments:} \((p1=a_1, p2=a_2, \ldots)\) \hspace{2cm} \textbf{Default:} \(\) \hspace{2cm} \textbf{Definition:} Each point has a assigned weight

\textit{You need at least two points. Result in \texttt{tkzPointResult}.}
9.3.1. with two points

In the following example, we obtain the barycenter of points A and B with coefficients 1 and 2, in other words:

\[
\overrightarrow{AI} = \frac{2}{3} \overrightarrow{AB}
\]

\begin{tikzpicture}
  \tkzDefPoint(2,3){A}
  \tkzDefShiftPointCoord[2,3](30:4){B}
  \tkzDefBarycentricPoint(A=1,B=2)
  \tkzGetPoint{G}
  \tkzDrawLine(A,B)
  \tkzDrawPoints(A,B,G)
  \tkzLabelPoints(A,B,G)
\end{tikzpicture}

9.3.2. with three points

This time M is simply the center of gravity of the triangle.

For reasons of simplification and homogeneity, there is also \texttt{\tkzCentroid}.

\begin{tikzpicture}[scale=.8]
  \tkzDefPoints{2/1/A,5/3/B,0/6/C}
  \tkzDefBarycentricPoint(A=1,B=1,C=1)
  \tkzGetPoint{G}
  \tkzDefMidPoint(A,B) \tkzGetPoint{C'}
  \tkzDefMidPoint(A,C) \tkzGetPoint{B'}
  \tkzDefMidPoint(C,B) \tkzGetPoint{A'}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawLines[add=0 and 1,new](A,G B,G C,G)
  \tkzDrawPoints[new](A',B',C',G)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoint[above right](G){$G$}
  \tkzAutoLabelPoints[center=G](A,B,C)
  \tkzLabelPoints[above right](A')
  \tkzLabelPoints[below](B',C')
\end{tikzpicture}

9.4. Internal and external Similitude Center

The centers of the two homotheties in which two circles correspond are called external and internal centers of similitude. You can use \texttt{\tkzDefIntSimilitudeCenter} and \texttt{\tkzDefExtSimilitudeCenter} but the next macro is better.

\begin{verbatim}
\tkzDefSimilitudeCenter([options])((O,A))(O',B))
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>((pt1,pt2))((pt3,pt4)) (O,A)(O',B)\</td>
<td>(r = OA,r' = O'B)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ext</td>
<td>ext</td>
<td>external center</td>
</tr>
<tr>
<td>int</td>
<td>ext</td>
<td>internal center</td>
</tr>
</tbody>
</table>
9.4.1. Internal and external with node

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/O,4/-5/A,3/0/B,5/-5/C}
\tkzDefSimilitudeCenter[int](O,B)(A,C) \tkzGetPoint{I}
\tkzDefSimilitudeCenter[ext](O,B)(A,C) \tkzGetPoint{J}
\tkzDefLine[tangent from = I](O,B) \tkzGetPoints{D}{E}
\tkzDefLine[tangent from = I](A,C) \tkzGetPoints{D'}{E'}
\tkzDefLine[tangent from = J](O,B) \tkzGetPoints{F}{G}
\tkzDefLine[tangent from = J](A,C)
\tkzGetPoints{F'}{G'}
\tkzDrawCircles(O,B A,C)
\tkzDrawSegments[add = .5 and .5,new](D,D' E,E')
\tkzDrawSegments[add= 0 and 0.25,new](J,F J,G)
\tkzDrawPoints(O,A,I,J,D,E,F,G,D',E',F',G')
\end{tikzpicture}

9.4.2. D'Alembert Theorem

\begin{tikzpicture}[scale=.6,rotate=90]
\tkzDefPoints{0/0/A,3/0/a,7/-1/B,5.5/-1/b}
\tkzDefPoints{5/-4/C,4.25/-4/c}
\tkzDrawCircles(A,a B,b C,c)
\tkzDefExtSimilitudeCenter(A,a)(B,b) \tkzGetPoint{I}
\tkzDefExtSimilitudeCenter(A,a)(C,c) \tkzGetPoint{J}
\tkzDefExtSimilitudeCenter(C,c)(B,b) \tkzGetPoint{K}
\tkzDefIntSimilitudeCenter(A,a)(B,b) \tkzGetPoint{I'}
\tkzDefIntSimilitudeCenter(A,a)(C,c) \tkzGetPoint{J'}
\tkzDefIntSimilitudeCenter(C,c)(B,b) \tkzGetPoint{K'}
\tkzDrawPoints(A,B,C,I,J,K,I',J',K')
\tkzDrawSegments[new](I,J' I',J I',K)
\end{tikzpicture}

You can use $\tkzDefBarycentricPoint$ to find a homothetic center
$\tkzDefBarycentricPoint(O=r,A=R) \tkzGetPoint{I}$
$\tkzDefBarycentricPoint(O=-r,A=R) \tkzGetPoint{J}$
9.4.3. Example with node

\begin{tikzpicture}[rotate=60,scale=.5]
\tkzDefPoints{0/0/A,5/0/C}
\tkzDefGoldenRatio(A,C) \tkzGetPoint{B}
\tkzDefSimitudeCenter(A,B)(C,B) \tkzGetPoint{J}
\tkzDefTangent[from = J](A,B) \tkzGetPoints{F}{G}
\tkzDefTangent[from = J](C,B) \tkzGetPoints{F'}{G'}
\tkzDrawCircles(A,B C,B)
\tkzDrawSegments[add= 0 and 0.25,cyan](J,F J,G)
\tkzDrawPoints(A,J,F,G,F',G')
\end{tikzpicture}
9.5. Harmonic division with \texttt{\tkzDefHarmonic}

\begin{verbatim}
\texttt{\tkzDefHarmonic[\texttt{options}](\texttt{pt1,pt2,pt3}) or \texttt{(pt1,pt2,k)}}
\end{verbatim}

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>both</td>
<td>both</td>
<td>((A,B,2)) we look for (C) and (D) such that ((A,B;C,D) = -1) and (CA=2CB)</td>
</tr>
<tr>
<td>ext</td>
<td>both</td>
<td>((A,B,C)) we look for (D) such that ((A,B;C,D) = -1)</td>
</tr>
<tr>
<td>int</td>
<td>both</td>
<td>((A,B,D)) we look for (C) such that ((A,B;C,D) = -1)</td>
</tr>
</tbody>
</table>

9.5.1. options ext and int

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,4/0/C}
\tkzDefHarmonic[ext](A,B,C) \tkzGetPoint{J}
\tkzDefHarmonic[int](A,B,J) \tkzGetPoint{I}
\tkzDrawPoints(A,B,I,J)
\tkzDrawLine[add=.5 and 1](A,B)
\end{tikzpicture}

9.5.2. Bisector and harmonic division

\begin{tikzpicture}[scale=1.25]
\tkzDefPoints{0/0/A,4/0/C,5/3/X}
\tkzDefLine[bisector](A,X,C) \tkzGetPoint{x}
\tkzInterLL(X,x)(A,C) \tkzGetPoint{B}
\tkzDefHarmonic[ext](A,C,B) \tkzGetPoint{D}
\tkzDrawPolygon(A,X,C)
\tkzDrawSegments(X,B C,D D,X)
\tkzDrawPoints(A,B,C,D,X)
\tkzMarkAngles[mark=s|](A,X,B B,X,C)
\tkzMarkRightAngle[size=.4,
fill=gray!20,
opacity=.3](B,X,D)
\tkzLabelPoints(A,B,C,D)
\tkzLabelPoints[above right](X)
\end{tikzpicture}
9. Special points

9.5.3. option both

both is the default option

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B}
\tkzDefHarmonic(A,B,{1/2})\tkzGetPoints{I}{J}
\tkzDrawPoints(A,B,I,J)
\tkzDrawLine[add=1 and .5](A,B)
\tkzLabelPoints(A,B,I,J)
\end{tikzpicture}

9.6. Equidistant points with \tkzDefEquiPoints

\begin{center}
\begin{tabular}{|c|c|c|}
\hline
arguments & default & definition \\
\hline
(pt1,pt2) & no default & unordered list of two items \\
\hline
options & default & definition \\
\hline
dist & 2 (cm) & half the distance between the two points \\
from=pt & no default & reference point \\
show & false & if true displays compass traces \\
/compass/delta & \emptyset & compass trace size \\
\hline
\end{tabular}
\end{center}

This macro makes it possible to obtain two points on a straight line equidistant from a given point.

9.6.1. Using \tkzDefEquiPoints with options

\begin{tikzpicture}
\tkzSetUpCompass[color=purple,line width=1pt]
\tkzDefPoints{0/1/A,5/2/B,3/4/C}
\tkzDefEquiPoints[from=C,dist=1,show,
/compass/delta=20](A,B)
\tkzGetPoints{E}{H}
\tkzDrawLines[color=blue](C,E C,H A,B)
\tkzDrawPoints[color=blue](A,B,C)
\tkzDrawPoints[color=red](E,H)
\tkzLabelPoints(E,H)
\tkzLabelPoints[color=blue](A,B,C)
\end{tikzpicture}

9.7. Middle of an arc

\begin{tikzpicture}
\tkzDefMidArc(pt1,pt2,pt3)
\tkzDrawPoints(A,B,C)
\tkzDrawLines(A,E A,F B,C)
\tkzLabelPoints(A,B,C)
\end{tikzpicture}

\begin{center}
\begin{tabular}{|c|c|c|}
\hline
arguments & default & definition \\
\hline
pt1,pt2,pt3 & no default & pt1 is the center, pt2pt3 the arc \\
\hline
\end{tabular}
\end{center}
9. Special points
10. Point on line or circle

10.1. Point on a line with \tkzDefPointOnLine

\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,10/0/B}
\tkzDefGoldenRatio(A,B) \tkzGetPoint{C}
\tkzDefMidPoint(A,B) \tkzGetPoint{O_1}
\tkzDefMidPoint(A,C) \tkzGetPoint{O_2}
\tkzDefMidPoint(C,B) \tkzGetPoint{O_3}
\tkzDefMidArc(O_3,B,C) \tkzGetPoint{P}
\tkzDefMidArc(O_2,C,A) \tkzGetPoint{Q}
\tkzDefMidArc(O_1,B,A) \tkzGetPoint{L}
\tkzDefPointBy[rotation=center C angle 90](B) \tkzGetPoint{c}
\tkzInterCC[common=B](P,B)(O_1,B) \tkzGetPoint{P_1}
\tkzInterCC[common=C](P,C)(O_2,C) \tkzGetPoint{P_2}
\tkzInterCC[common=C](Q,C)(O_3,C) \tkzGetPoint{P_3}
\tkzInterLC[near](c,C)(O_1,A) \tkzGetPoint{D}
\tkzInterLL(A,P_1)(C,D) \tkzGetPoint{P_1'}
\tkzDefPointBy[inversion = center A through D](P_2) \tkzGetPoint{P_2'}
\tkzDefCircle[circum](P_3,P_2,P_1) \tkzGetPoint{O_4}
\tkzInterLL(B,Q)(A,P) \tkzGetPoint{S}
\tkzDefMidPoint(P_2',P_1') \tkzGetPoint{o}
\tkzDefPointBy[inversion = center A through D](S) \tkzGetPoint{S'}
\tkzDrawArc[cyan,\delta=0](Q,A)(P_1)
\tkzDrawArc[cyan,\delta=0](P,P_1)(B)
\tkzDrawSemiCircles[teal](O_1,B O_2,C O_3,B)
\tkzDrawCircles[new](o,P_4,P_1)
\tkzDrawSegments(A,B)
\tkzDrawSegments[cyan](A,P_1 A,S' A,P_2')
\tkzDrawSegments[purple](B,L C,P_2' B,Q B,L S',P_1')
\tkzDrawLines[add=0 and .8](B,P_2')
\tkzDrawLines[add=0 and .4](C,D)
\tkzDrawPoints(A,B,C,P,Q,P_3,P_2,P_1,P_1',D,P_2',L,S,S')
\tkzLabelPoints(A,B,C,P,Q)
\tkzLabelPoints[above](P,Q,P_1)
\tkzLabelPoints[above right](P_2,P_2',D,S')
\tkzLabelPoints[above left](L,S)
\tkzLabelPoints[below left](P_1')
\end{tikzpicture}
10.1.1. Use of option pos

```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/0/B}
  \tkzDefPointOnLine[pos=1.2](A,B) \tkzGetPoint{P}
  \tkzDefPointOnLine[pos=-0.2](A,B) \tkzGetPoint{R}
  \tkzDefPointOnLine[pos=0.5](A,B) \tkzGetPoint{S}
  \tkzDrawLine[new](A,B)
  \tkzDrawPoints(A,B,P)
  \tkzLabelPoints(A,B)
  \tkzLabelPoint[above](P){pos=$1.2$}
  \tkzLabelPoint[above](R){pos=$-.2$}
  \tkzLabelPoint[above](S){pos=$.5$}
  \tkzDrawPoints(A,B,P,R,S)
  \tkzLabelPoints(A,B)
\end{tikzpicture}
```

10.2. Point on a circle with \tkzDefPointOnCircle

The order of the arguments has changed: now it is center, angle and point or radius. I have added two options for working with radians which are through in rad and R in rad.

```
\tkzDefPointOnCircle[⟨local options⟩]
```

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>examples definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>through</td>
<td>through = center K1 angle 30 point B]</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>R = center K1 angle 30 radius \rAp</td>
<td></td>
</tr>
<tr>
<td>through in rad</td>
<td>through in rad= center K1 angle pi/4 point B]</td>
<td></td>
</tr>
<tr>
<td>R in rad</td>
<td>R in rad = center K1 angle pi/6 radius \rAp</td>
<td></td>
</tr>
</tbody>
</table>

*The new order for arguments are: center, angle and point or radius.*

10.2.1. Altshiller’s Theorem

The two lines joining the points of intersection of two orthogonal circles to a point on one of the circles met the other circle in two diametricaly opposite points. Altshiller p 176

```
\begin{tikzpicture}[scale=.4]
  \tkzDefPoints{0/0/P,5/0/Q,3/2/I}
  \tkzDefCircle[orthogonal from=P](Q,I)
  \tkzGetFirstPoint{E}
  \tkzDrawCircles(P,E Q,E)
  \tkzInterCC[common=E](P,E)(Q,E) \tkzGetFirstPoint{F}
  \tkzDefPointOnCircle[through = center P angle 80 point E]
  \tkzGetPoint{A}
  \tkzInterLC[common=E](A,E)(Q,E) \tkzGetFirstPoint{C}
  \tkzInterLL(A,F)(C,Q) \tkzGetPoint{D}
  \tkzDrawLines[add=0 and .75](P,Q)
  \tkzDrawLines[add=0 and 2](A,E)
  \tkzDrawSegments(P,E E,F F,C A,F C,D)
  \tkzDrawPoints(P,Q,E,F,A,C,D)
  \tkzLabelPoints(P,Q,F,C,D)
  \tkzLabelPoints[above]{E,A}
\end{tikzpicture}
```
10.2.2. Use of \tkzDefPointOnCircle

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B,0.8/3/C}
\tkzDefPointOnCircle[R = center B angle 90 radius 1](I)
\tkzDefCircle[ circum](A,B,C)
\tkzDefPointOnCircle[through = center G angle 30 point g](J)
\tkzDefCircle[R](B,1)(b)
\tkzDrawCircle[teal](B,b)
\tkzDrawCircle(G,J)
\tkzDrawPoints(A,B,C,G,I,J)
\tkzAutoLabelPoints[center=G](A,B,C,J)
\tkzLabelPoints[below](G,I)
\end{tikzpicture}
11. Special points relating to a triangle

11.1. Triangle center: \texttt{\textbackslash tkzDefTriangleCenter}

\begin{verbatim}
\tkzDefTriangleCenter[(local options)]((A,B,C))
\end{verbatim}

This macro allows you to define the center of a triangle. Be careful, the arguments are lists of three points. This macro is used in conjunction with \texttt{\textbackslash tkzGetPoint} to get the center you are looking for.

You can use \texttt{\textbackslash tkzPointResult} if it is not necessary to keep the results.

\begin{tabular}{|l|l|l|}
\hline
arguments & default & example \\
\hline
(pt1,pt2,pt3) & no default & \texttt{\textbackslash tkzDefTriangleCenter[ortho]}(B,C,A) \\
\hline
\end{tabular}

\begin{tabular}{|l|l|}
\hline
options & definition \\
\hline
ortho & \texttt{circum} intersection of the altitudes \\
orthic & \texttt{circum} \ldots \\
centroid & \texttt{circum} \texttt{circum} \texttt{center} \texttt{circumscribed} \\
median & \texttt{circum} \texttt{center} \texttt{of} \texttt{the} \texttt{medians} \\
circum & \texttt{circum} \texttt{center} \texttt{of} \texttt{the} \texttt{circumscribed} \texttt{circle} \\
in & \texttt{circum} \texttt{center} \texttt{of} \texttt{the} \texttt{circle} \texttt{inscribed} \texttt{in} \texttt{a} \texttt{triangle} \\
ine & \texttt{circum} \texttt{center} \texttt{of} \texttt{the} \texttt{bisectors} \\
euler & \texttt{circum} \texttt{center} \texttt{of} \texttt{Euler's} \texttt{circle} \\
gergonne & \texttt{circum} \texttt{defined} \texttt{with} \texttt{the} \texttt{Contact} \texttt{triangle} \\
symmedian & \texttt{circum} \texttt{Lemoine's} \texttt{point} \texttt{or} \texttt{symmedian} \texttt{center} \texttt{or} \texttt{Grebe's} \texttt{point} \\
lemoine & \texttt{circum} \texttt{...} \\
grebe & \texttt{circum} \texttt{...} \\
spieker & \texttt{circum} \texttt{Spiker} \texttt{circle} \texttt{center} \\
nagel & \texttt{circum} \texttt{Nagel} \texttt{Center} \\
mittenpunkt & \texttt{circum} \texttt{Or} \texttt{middlepoint} \\
feuerbach & \texttt{circum} \texttt{Feuerbach} \texttt{Point} \\
\hline
\end{tabular}

11.1.1. Option ortho or orthic

The intersection $H$ of the three altitudes of a triangle is called the orthocenter.

\begin{verbatim}
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,1){B}
\tkzDefPoint(1,4){C}
\tkzDefTriangleCenter[ortho](B,C,A)
\tkzGetPoint{H}
\tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
\tkzDrawPolygon(A,B,C)
\tkzDrawSegments[thick](A,Ha B,Hb C,Hc A)
\tkzDrawPoints(A,B,C,H)
\tkzLabelPoints{below}(A,B)
\tkzLabelPoints{above}(C)
\tkzMarkRightAngles(A,Ha B B,Hb C C,Hc A)
\end{tikzpicture}
\end{verbatim}
11. Special points relating to a triangle

11.1.2. Option centroid

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,5/0/B,1/4/C}
\tkzDefTriangleCenter[centroid](A,B,C)
\tkzGetPoint(G)
\tkzDrawPolygon(A,B,C)
\tkzDrawLines[add = 0 and 2/3,new](A,G B,G C,G)
\tkzDrawPoints(A,B,C,G)
\tkzLabelPoint(G){$G$}
\end{tikzpicture}

11.1.3. Option circum

\begin{tikzpicture}
\tkzDefPoints{0/1/A,3/2/B,1/4/C}
\tkzDefTriangleCenter[circum](A,B,C)
\tkzGetPoint(O)
\tkzDrawPolygon(A,B,C)
\tkzDrawCircle(O,A)
\tkzDrawPoints(A,B,C,O)
\tkzLabelPoint(O){$O$}
\end{tikzpicture}

11.1.4. Option in

In geometry, the incircle or inscribed circle of a triangle is the largest circle contained in the triangle; it touches (is tangent to) the three sides. The center of the incircle is a triangle center called the triangle’s incenter. The center of the incircle, called the incenter, can be found as the intersection of the three internal angle bisectors. The center of an excircle is the intersection of the internal bisector of one angle (at vertex A, for example) and the external bisectors of the other two. The center of this excircle is called the excenter relative to the vertex A, or the excenter of A. Because the internal bisector of an angle is perpendicular to its external bisector, it follows that the center of the incircle together with the three excircle centers form an orthocentric system. (Article on Wikipedia)

We get the center of the inscribed circle of the triangle. The result is of course in \(\text{tkzPointResult} \). We can retrieve it with \(\text{tkzGetPoint} \).

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefTriangleCenter[in](A,B,C)
\tkzGetPoint(I)
\tkzDrawLines(A,B B,C C,A)
\tkzDefCircle[in](A,B,C) \tkzGetPoints{I}{i}
\tkzDrawCircle(I,i)
\tkzDrawPoint[red](I)
\tkzDrawPoints(A,B,C)
\tkzLabelPoint(I){$I$}
\end{tikzpicture}

11.1.5. Option ex

An excircle or escribed circle of the triangle is a circle lying outside the triangle, tangent to one of its sides and tangent to the extensions of the other two. Every triangle has three distinct excircles, each tangent to one of the
11. Special points relating to a triangle

11.1.6. Option euler

This macro allows to obtain the center of the circle of the nine points or euler's circle or Feuerbach's circle. The nine-point circle, also called Euler's circle or the Seuerbach circle, is the circle that passes through the perpendicular feet $H_A$, $H_B$, and $H_C$ dropped from the vertices of any reference triangle $ABC$ on the sides opposite them. Euler showed in 1765 that it also passes through the midpoints $M_A$, $M_B$, $M_C$ of the sides of $ABC$. By Feuerbach's theorem, the nine-point circle also passes through the midpoints $E_A$, $E_B$, and $E_C$ of the segments that join the vertices and the orthocenter $H$. These points are commonly referred to as the Euler points.

(https://mathworld.wolfram.com/Nine-PointCircle.html)

11.1.7. Option symmedian

The point of concurrence $K$ of the symmedians, sometimes also called the Lemoine point (in England and France) or the Grebe point (in Germany).

11. Special points relating to a triangle

11.1.8. Option spieker

The Spieker center is the center $Sp$ of the Spieker circle, i.e., the incenter of the medial triangle of a reference triangle.


11.1.9. Option gergonne

The Gergonne Point is the point of concurrency which results from connecting the vertices of a triangle to the opposite points of tangency of the triangle's incircle. (Joseph Gergonne French mathematician)
11. Special points relating to a triangle

11.1.10. Option nagel

Let $T_a$ be the point at which the excircle with center $J_a$ meets the side $BC$ of a triangle $ABC$, and define $T_b$ and $T_c$ similarly. Then the lines $A T_a$, $B T_b$, and $C T_c$ concur in the Nagel point $N_a$.


11.1.11. Option mittenpunkt

The mittenpunkt (also called the middlespoint) of a triangle $ABC$ is the symmedian point of the excentral triangle, i.e., the point of concurrence $M$ of the lines from the excenters through the corresponding triangle side midpoints.

11.1.12. Relation between gergonne, centroid and mittenpunkt

The Gergonne point $G_e$, triangle centroid $G$, and mittenpunkt $M$ are collinear, with $G_eG/GM=2$. 

12. Definition of points by transformation

These transformations are:
- translation;
- homothety;
12. Definition of points by transformation

- orthogonal reflection or symmetry;
- central symmetry;
- orthogonal projection;
- rotation (degrees or radians);
- inversion with respect to a circle.

12.1. \texttt{tkzDefPointBy}

The choice of transformations is made through the options. There are two macros, one for the transformation of a single point \texttt{tkzDefPointBy} and the other for the transformation of a list of points \texttt{tkzDefPointsBy}. By default the image of $A$ is $A'$. For example, we’ll write:

\begin{verbatim}
\tkzDefPointBy[translation= from A to A'](B)
\end{verbatim}

The result is in \texttt{tkzPointResult}

\begin{verbatim}
\tkzDefPointBy[(local options)](pt)
\end{verbatim}

The argument is a simple existing point and its image is stored in \texttt{tkzPointResult}. If you want to keep this point then the macro \texttt{tkzGetPoint{M}} allows you to assign the name $M$ to the point.

\begin{tabular}{ll}
\hline
\textbf{arguments} & \textbf{definition} & \textbf{examples} \\
\hline
pt & existing point name ($A$) & \begin{verbatim}
\tkzDefPointBy[translation=from A to B](E)
\end{verbatim} \\
\hline
options & examples & \begin{verbatim}
\tkzDefPointBy[translation=from A to B](E)
\end{verbatim} \\
\hline
translation & = from #1 to #2 & \begin{verbatim}
\tkzDefPointBy[translation=from A to B](E)
\end{verbatim} \\
\hline
homothety & = center #1 ratio #2 & \begin{verbatim}
\tkzDefPointBy[homothety=center A ratio .5](E)
\end{verbatim} \\
\hline
reflection & = over #1--#2 & \begin{verbatim}
\tkzDefPointBy[reflection=over A--B](E)
\end{verbatim} \\
\hline
symmetry & = center #1 & \begin{verbatim}
\tkzDefPointBy[symmetry=center A](E)
\end{verbatim} \\
\hline
projection & = onto #1--#2 & \begin{verbatim}
\tkzDefPointBy[projection=onto A--B](E)
\end{verbatim} \\
\hline
rotation & = center #1 angle #2 & \begin{verbatim}
\tkzDefPointBy[rotation=center 0 angle 30](E)
\end{verbatim} \\
\hline
rotation in rad & = center #1 angle #2 & \begin{verbatim}
\tkzDefPointBy[rotation in rad=center 0 angle pi/3](E)
\end{verbatim} \\
\hline
rotation with nodes & = center #1 from #2 to #3 & \begin{verbatim}
\tkzDefPointBy[rotation with nodes=center 0 from A to B](E)
\end{verbatim} \\
\hline
inversion & = center #1 through #2 & \begin{verbatim}
\tkzDefPointBy[inversion=center 0 through A](E)
\end{verbatim} \\
\hline
inversion negative & = center #1 through #2 & \begin{verbatim}
\tkzDefPointBy[inversion negative=center 0 through A](E)
\end{verbatim} \\
\hline
\end{tabular}

\begin{verbatim}
The image is only defined and not drawn.
\end{verbatim}

12.1.1. translation

\begin{verbatim}
\begin{tikzpicture}[>=latex]
\tkzDefPoints{0/0/A,3/1/B,3/0/C}
\tkzDefPointBy[translation= from B to A](C)
\tkzGetPoint{D}
\tkzDrawPoints[teal](A,B,C,D)
\tkzLabelPoints[teal](A,B,C,D)
\tkzDrawSegments[orange,->](A,B D,C)
\end{tikzpicture}
\end{verbatim}

\texttt{tkz-euclide AlterMundus}
12.1.2. reflection (orthogonal symmetry)

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{-2/-2/A,-1/-1/C,-4/2/D,-4/0/O}
\tkzDrawCircle(O,A)
\tkzDefPointBy[reflection = over C--D](A)
\tkzGetPoint{A'}
\tkzDefPointBy[reflection = over C--D](O)
\tkzGetPoint{O'}
\tkzDrawCircle(O',A')
\tkzDrawLine[add= .5 and .5](C,D)
\tkzDrawPoints(C,D,O,O')
\end{tikzpicture}

12.1.3. homothety and projection

\begin{tikzpicture}
\tkzDefPoints{0/1/A,5/3/B,3/4/C}
\tkzDefLine[bisector](B,A,C) \tkzGetPoint{a}
\tkzDrawLine[add=0 and 0,color=magenta!50](A,a)
\tkzDefPointBy[homothety=center A ratio .5](a)
\tkzGetPoint{a'}
\tkzDefPointBy[projection = onto A--B](a')
\tkzGetPoint{k'}
\tkzDefPointBy[projection = onto A--B](a)
\tkzGetPoint{k}
\tkzDefLines[add= & .3](A,k A,C)
\tkzDrawSegments[blue](a',k' a,k)
\tkzDrawPoints(a,a',k,k')
\tkzDrawCircles(a',k',a,k)
\tkzLabelPoints(a,a',k,A)
\end{tikzpicture}
12. Definition of points by transformation

12.1.4. projection

\begin{tikzpicture}[scale=1.5]
\tkzDefPoints{A/A/0,B/B/0}
\tkzDefTriangle[pythagore](B,A) \tkzGetPoint{C}
\tkzDefLine[bisector](B,C,A) \tkzGetPoint{c}
\tkzInterLL(C,c)(A,B) \tkzGetPoint{D}
\tkzDefPointBy[projection=onto B--C](D)
\tkzGetPoint{G}
\tkzInterLC(C,D)(D,A) \tkzGetPoints{E}{F}
\tkzDrawPolygon(A,B,C)
\tkzDrawSegment(C,D)
\tkzDrawCircle(D,A)
\tkzDrawSegment[new](D,G)
\tkzMarkRightAngle[fill=orange!10,opacity=.4](D,G,B)
\tkzDrawPoints(A,C,F) \tkzLabelPoints(A,C,F)
\tkzDrawPoints(B,D,E,G)
\tkzLabelPoints[above right](B,D,E)
\tkzLabelPoints[above](G)
\end{tikzpicture}

12.1.5. symmetry

\begin{tikzpicture}[scale=1]
\tkzDefPoints{A/A/2,B/B/2,O/O/0}
\tkzDefPointsBy[symmetry=center O](B,A){}
\tkzDrawLine(A,A')
\tkzDrawLine(B,B')
\tkzMarkAngle[mark=s,arc=lll,size=1.5,mkcolor=red](A,0,B)
\tkzLabelAngle[pos=2,circle,draw,fill=blue!10,font=\scriptsize](A,0,B){$60^\circ$}
\tkzDrawPoints(A,B,0,A',B')
\tkzLabelPoints[below](A,0,A')
\end{tikzpicture}
12. Definition of points by transformation

12.1.6. rotation

\begin{tikzpicture}[scale=0.5]
\tkzDefPoints{0/0/A,5/0/B}
\tkzDrawSegment(A,B)
\tkzDefPointBy[rotation=center A angle 60](B)
\tkzGetPoint{C}
\tkzDefPointBy[symmetry=center C](A)
\tkzGetPoint{D}
\tkzDrawSegment(A,tkzPointResult)
\tkzDrawLine(B,D)
\tkzDrawArc(A,B)(C) \tkzDrawArc(B,C)(A)
\tkzDrawArc(C,D)(D)
\tkzMarkRightAngle(D,B,A)
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above](C)
\tkzLabelPoints[right](D)
\end{tikzpicture}

12.1.7. rotation in radian

\begin{tikzpicture}
\tkzDefPoint\"A" left(1,5){A}
\tkzDefPoint\"B" right(4,3){B}
\tkzDefPointBy[rotation in rad= center A angle pi/3](B)
\tkzGetPoint{C}
\tkzDrawSegment(A,B)
\tkzDrawPoints(A,B,C)
\tkzCompass(A,C)
\tkzCompass(B,C)
\tkzLabelPoints(C)
\end{tikzpicture}

12.1.8. rotation with nodes

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\tkzDefPoint(0:2){A}
\tkzDefPoint(40:2){B}
\tkzDefPoint(20:4){C}
\tkzDrawLine(O,A)
\tkzDefPointBy[rotation with nodes %
\quad=center O from A to B](C)
\tkzGetPoint{D}
\tkzDrawPoints(A,B,C,D)
\tkzDrawCircle(O,A)
\tkzLabelPoints(A,C,D)
\tkzLabelPoints[above](B)
\end{tikzpicture}

12.1.9. inversion

Inversion is the process of transforming points to a corresponding set of points known as their inverse points. Two points \( P \) and \( P' \) are said to be inverses with respect to an inversion circle having inversion center \( O \) and inversion radius \( k \) if \( P' \) is the perpendicular foot of the altitude of \( OQP \), where \( Q \) is a point on the circle such that
OQ is perpendicular to PQ.
The quantity $k^2$ is known as the circle power (Coxeter 1969, p. 81). (https://mathworld.wolfram.com/Inversion.html)

Some propositions:
- The inverse of a circle (not through the center of inversion) is a circle.
- The inverse of a circle through the center of inversion is a line.
- The inverse of a line (not through the center of inversion) is a circle through the center of inversion.
- A circle orthogonal to the circle of inversion is its own inverse.
- A line through the center of inversion is its own inverse.
- Angles are preserved in inversion.

Explanation:
Directly (Center O power $= k^2 = OA^2 = OP \times OP'$)

```latex
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{4/0/A,6/0/P,0/0/O}
  \tkzDefLine[orthogonal=through P](O,P)
  \tkzGetPoint{L}
  \tkzDefLine[tangent from = P](O,A) \tkzGetPoints{R}{Q}
  \tkzDefPointBy[projection=onto O--A](Q) \tkzGetPoint{P'}
  \tkzDrawSegments(O,P O,A)
  \tkzDrawSegments(P,L)
  \tkzLabelPoints[left,font={\scriptsize}](O,P')
  \tkzLabelPoints[above right,font={\scriptsize}](P,Q)
  \tkzDrawPoints(O,P)
  \tkzDrawPoints[new](Q,P')
  \tkzMarkRightAngles(A,P',Q,P,Q,O)
  \tkzLabelCircle[above=.5cm,font={\scriptsize}](O,A){inversion circle}
  \tkzLabelPoint[left,font={\scriptsize}](O){inversion center}
  \tkzLabelPoint[left,font={\scriptsize}](L){polar}
\end{tikzpicture}
```
12.1.10. Inversion of lines ex 1

\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/O,3/0/I,4/3/P,6/-3/Q}
  \tkzDrawCircle(O,I)
  \tkzDefPointBy[projection= onto P--Q](O) \tkzGetPoint{A}
  \tkzDefPointBy[inversion = center O through I](A)
  \tkzGetPoint{A'}
  \tkzDefPointBy[inversion = center O through I](P)
  \tkzGetPoint{P'}
  \tkzDefCircle[diameter](O,A')\tkzGetPoint{o}
  \tkzDrawCircle[new](o,A')
  \tkzDrawLines[add=.25 and .25,red](P,Q)
  \tkzDrawLines[add=.25 and .25](O,A)
  \tkzDrawSegments(O,P)
  \tkzDrawPoints(A,P,O) \tkzDrawPoints[new](A',P')
\end{tikzpicture}

12.1.11. inversion of lines ex 2

\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/O,3/0/I,3/2/P,3/-2/Q}
  \tkzDrawCircle(O,I)
  \tkzDefPointBy[projection= onto P--Q](O) \tkzGetPoint{A}
  \tkzDefPointBy[inversion = center O through I](A)
  \tkzGetPoint{A'}
  \tkzDefPointBy[inversion = center O through I](P)
  \tkzGetPoint{P'}
  \tkzDefCircle[diameter](O,A')\tkzGetPoint{o}
  \tkzDrawCircle[new](o,A')
  \tkzDrawLines[add=.25 and .25,red](P,Q)
  \tkzDrawLines[add=.25 and .25](O,A)
  \tkzDrawSegments(O,P)
  \tkzDrawPoints(A,P,O) \tkzDrawPoints[new](A',P')
\end{tikzpicture}

12.1.12. inversion of lines ex 3

\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/O,3/0/I,2/1/P,2/-2/Q}
  \tkzDrawCircle(O,I)
  \tkzDefPointBy[projection= onto P--Q](O) \tkzGetPoint{A}
  \tkzDefPointBy[inversion = center O through I](A)
  \tkzGetPoint{A'}
  \tkzDefPointBy[inversion = center O through I](P)
  \tkzGetPoint{P'}
  \tkzDefCircle[diameter](O,A')\tkzGetPoint{i}
  \tkzDrawCircle[new](I,A')
  \tkzDrawLines[add=.25 and .75,red](P,Q)
  \tkzDrawLines[add=.25 and .25](O,A')
  \tkzDrawSegments(O,P')
  \tkzDrawPoints(A,P,O) \tkzDrawPoints[new](A',P')
\end{tikzpicture}
12. Definition of points by transformation

12.1.13. inversion of circle and homothety

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/O,3/2/A,2/1/P}
\tkzDefLine[tangent from = O](A,P) \tkzGetPoints{T}{X}
\tkzDefPointsBy[homothety = center O ratio 1.25](A,P,T){}
\tkzInterCC(A,P)(A',P') \tkzGetPoints{C}{D}
\tkzCalcLength(A,P) \tkzGetLength{rAP}
\tkzDefPointOnCircle[R= center A angle 190 radius \rAP](M)
\tkzDefPointBy[inversion = center O through C](M){}
\tkzDrawCircles[new](A,P A',P')
\tkzDrawCircle(0,C)
\tkzDrawLines[add=0 and .5](0,T' 0,A' 0,M' 0,P')
\tkzLabelPoints(O,T,T',M,M')
\tkzLabelPoints[below](P,P')
\end{tikzpicture}

12.1.14. inversion of Triangle with respect to the Incircle

\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,5/1/B,3/6/C}
\tkzDefTriangleCenter[in](A,B,C) \tkzGetPoint{O}
\tkzDefPointBy[projection= onto A--C](O) \tkzGetPoint{a}
\tkzDefPointBy[projection= onto B--C](O) \tkzGetPoint{b}
\tkzDefPointBy[projection= onto A--B](O) \tkzGetPoint{c}
\tkzDefPointsBy[inversion = center O through b](a,b,c){Ia,Ib,Ic}
\tkzDefMidPoint(O,Ia) \tkzGetPoint{Ja}
\tkzDefMidPoint(O,Ib) \tkzGetPoint{Jb}
\tkzDefMidPoint(O,Ic) \tkzGetPoint{Jc}
\tkzInterCC(Ja,O)(Jb,O) \tkzGetPoints{x}
\tkzInterCC(Ja,O)(Jc,O) \tkzGetPoints{y}
\tkzInterCC(Jb,O)(Jc,O) \tkzGetPoints{z}
\tkzDrawPolygon(A,B,C)
\tkzDrawCircle(O,b)
\tkzDrawCircles[dashed,gray](Ja,x Jb,y Jc,z)
\tkzLabelPoint[below](A){$A$} \tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[below](B){$B$}
\end{tikzpicture}

12.1.15. inversion: orthogonal circle with inversion circle

The inversion circle itself, circles orthogonal to it, and lines through the inversion center are invariant under inversion. If the circle meets the reference circle, these invariant points of intersection are also on the inverse circle. See I and J in the next figure.
12. Definition of points by transformation

For a more complex example see Pappus 46.25

12.1.16. inversion negative

It's an inversion followed by a symmetry of center O

\begin{tikzpicture}[scale=1.5]
\tkzDefPoints{1/0/A,0/0/O}
\tkzDefPoint(-1.5,-1.5){z1}
\tkzDefPoint(0.35,-2){z2}
\tkzDefPointBy[inversion negative = center O through A](z1)
\tkzGetPoint{Z1}
\tkzDefPointBy[inversion negative = center O through A](z2)
\tkzGetPoint{Z2}
\tkzDrawCircle(O,A)
\tkzDrawPoints[color=black, fill=red,size=4](Z1,Z2)
\tkzDrawSegments(z1,Z1 z2,Z2)
\tkzDrawPoints[color=black, fill=red,size=4](O,z1,z2)
\tkzLabelPoints[color=black, font={\scriptsize}](O,A,z1,z2,Z1,Z2)
\end{tikzpicture}
12. Definition of points by transformation

12.2. Transformation of multiple points; \tkzDefPointsBy

Variant of the previous macro for defining multiple images. You must give the names of the images as arguments, or indicate that the names of the images are formed from the names of the antecedents, leaving the argument empty.

\tkzDefPointsBy[translation= from A to A'](B,C){}
The images are B' and C'.

\tkzDefPointsBy[translation= from A to A'](B,C){D,E}
The images are D and E.

\tkzDefPointsBy[translation= from A to A'](B){}
The image is B'.

\tkzDefPointsBy[(local options)]((list of points))((list of points))

<table>
<thead>
<tr>
<th>arguments</th>
<th>examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>(list of points)(list of pts)</td>
<td>(A,B){E,F} E,F images of A, B</td>
</tr>
</tbody>
</table>

If the list of images is empty then the name of the image is the name of the antecedent to which "'" is added.

<table>
<thead>
<tr>
<th>options</th>
<th>examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation = from #1 to #2</td>
<td><a href="E">translation=from A to B</a>{}</td>
</tr>
<tr>
<td>homothety = center #1 ratio #2</td>
<td><a href="E">homothety=center A ratio .5</a>{}</td>
</tr>
<tr>
<td>reflection = over #1--#2</td>
<td><a href="E">reflection=over A--B</a>{}</td>
</tr>
<tr>
<td>symmetry = center #1</td>
<td><a href="E">symmetry=center A</a>{}</td>
</tr>
<tr>
<td>projection = onto #1--#2</td>
<td><a href="E">projection=onto A--B</a>{}</td>
</tr>
<tr>
<td>rotation = center #1 angle #2</td>
<td><a href="E">rotation=center angle 30</a>{}</td>
</tr>
<tr>
<td>rotation in rad = center #1 angle #2</td>
<td>for instance angle pi/3</td>
</tr>
<tr>
<td>rotation with nodes = center #1 from #2 to #3</td>
<td><a href="E">center 0 from A to B</a>{}</td>
</tr>
<tr>
<td>inversion = center #1 through #2</td>
<td><a href="E">inversion = center 0 through A</a>{}</td>
</tr>
<tr>
<td>inversion negative = center #1 through #2</td>
<td>...</td>
</tr>
</tbody>
</table>

The points are only defined and not drawn.

12.2.1. translation of multiple points

\begin{tikzpicture}[>=latex]
\tkzDefPoints{0/0/A,3/0/B,3/1/A',1/2/C}
\tkzDefPointsBy[translation= from A to A'](B,C){}
\tkzDrawPolygon(A,B,C)
\tkzDrawPolygon(new)(A',B',C')
\tkzDrawPoints(A,B,C)
\tkzDrawPoints(new)(A',B',C')
\tkzLabelPoints(A,B,A',B')
\tkzLabelPoints(above)(C,C')
\tkzDrawSegments[color = gray,->, style=dashed](A,A' B,B' C,C')
\end{tikzpicture}
12.2.2. symmetry of multiple points: an oval

\begin{tikzpicture}[scale=0.4]
\tkzDefPoint(-4,0){I}
\tkzDefPoint(4,0){J}
\tkzDefPoint(0,0){O}
\tkzInterCC(J,O)(O,J) \tkzGetPoints{L}{H}
\tkzInterCC(I,O)(O,I) \tkzGetPoints{K}{G}
\tkzInterLL(I,K)(J,H) \tkzGetPoint{M}
\tkzInterLL(I,G)(J,L) \tkzGetPoint{N}
\tkzDefPointsBy[ symmetry=center J](L,H){D,E}
\tkzDefPointsBy[ symmetry=center I](G,K){C,F}
\begin{scope}[line style/.style = {very thin,teal}]
\tkzDrawLines[add=1.5 and 1.5](I,K I,G J,H J,L)
\tkzDrawLines[add=.5 and .5](I,J)
\tkzDrawCircles(O,I I,O J,O)
\tkzDrawArc[delta=0,orange](N,D)(C)
\tkzDrawArc[delta=0,orange](M,F)(E)
\tkzDrawArc[delta=0,orange](J,E)(D)
\tkzDrawArc[delta=0,orange](I,C)(F)
\end{scope}
\end{tikzpicture}

13. Defining points using a vector

13.1. \texttt{tkzDefPointWith}

There are several possibilities to create points that meet certain vector conditions. This can be done with \texttt{tkzDefPointWith}. The general principle is as follows, two points are passed as arguments, i.e. a vector. The different options allow to obtain a new point forming with the first point (with some exceptions) a collinear vector or a vector orthogonal to the first vector. Then the length is either proportional to that of the first one, or proportional to the unit. Since this point is only used temporarily, it does not have to be named immediately. The result is in \texttt{tkzPointResult}. The macro \texttt{tkzGetPoint} allows you to retrieve the point and name it differently. There are options to define the distance between the given point and the obtained point. In the general case this distance is the distance between the 2 points given as arguments if the option is of the "normed" type then the distance between the given point and the obtained point is 1 cm. Then the \texttt{K} option allows to obtain multiples.

\texttt{\tkzDefPointWith((pt1,pt2))}

It is in fact the definition of a point meeting vectorial conditions.

<table>
<thead>
<tr>
<th>arguments</th>
<th>definition</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)</td>
<td>point couple</td>
<td>the result is a point in \texttt{tkzPointResult}</td>
</tr>
</tbody>
</table>

In what follows, it is assumed that the point is recovered by \texttt{tkzGetPoint(C)}

<table>
<thead>
<tr>
<th>options</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>orthogonal</td>
<td><a href="A,B">orthogonal</a></td>
<td>\overrightarrow{AC} = \overrightarrow{AB} and \overrightarrow{AC} \perp \overrightarrow{AB}</td>
</tr>
<tr>
<td>orthogonal normed</td>
<td><a href="A,B">orthogonal normed</a></td>
<td>\overrightarrow{AC} = 1 and \overrightarrow{AC} \perp \overrightarrow{AB}</td>
</tr>
<tr>
<td>linear</td>
<td><a href="A,B">linear</a></td>
<td>\overrightarrow{AC} = K \times \overrightarrow{AB}</td>
</tr>
<tr>
<td>linear normed</td>
<td><a href="A,B">linear normed</a></td>
<td>\overrightarrow{AC} = K and \overrightarrow{AC} = k \times \overrightarrow{AB}</td>
</tr>
<tr>
<td>colinear= at #1</td>
<td><a href="A,B">colinear= at C</a></td>
<td>\overrightarrow{CD} = \overrightarrow{AB}</td>
</tr>
<tr>
<td>colinear normed= at #1</td>
<td><a href="A,B">colinear normed= at C</a></td>
<td>\overrightarrow{CD} = \overrightarrow{AB}</td>
</tr>
<tr>
<td>K</td>
<td><a href="A,B">linear</a>,K=2</td>
<td>\overrightarrow{AC} = 2 \times \overrightarrow{AB}</td>
</tr>
</tbody>
</table>
13. Defining points using a vector

13.1.1. Option colinear at, simple example

\( \overrightarrow{AB} = \overrightarrow{CD} \)

\begin{tikzpicture}
[\text{scale=1.2},
 vect/.style={->,shorten >=1pt,>=latex'}]
\tkzDefPoint(2,3){A} \tkzDefPoint(4,2){B}
\tkzDefPoint(0,1){C}
\tkzDefPointWith[colinear=at C](A,B)
\tkzGetPoint{D}
\tkzDrawPoints{new}(A,B,C,D)
\tkzLabelPoints{above right=3pt}(A,B,C,D)
\tkzDrawSegments[vect](A,B C,D)
\end{tikzpicture}

13.1.2. Option colinear at, complex example
13. Defining points using a vector

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/B,3.6/0/C,1.5/4/A}
\tkzDefSpcTriangle[ortho](A,B,C){Ha,Hb,Hc}
\tkzDefTriangleCenter[ortho](A,B,C) \tkzGetPoint{H}
\tkzDefSquare(A,C) \tkzGetPoints{R}{S}
\tkzDefSquare(B,A) \tkzGetPoints{M}{N}
\tkzDefSquare(C,B) \tkzGetPoints{P}{Q}
\tkzDefPointWith[colinear= at M](A,S) \tkzGetPoint{A'}
\tkzDefPointWith[colinear= at P](B,N) \tkzGetPoint{B'}
\tkzDefPointWith[colinear= at Q](C,R) \tkzGetPoint{C'}
\tkzDefPointBy[projection=onto P--Q](Ha) \tkzGetPoint{Pa}
\tkzDrawPolygon[teal,thick](A,C,R,S) \tkzDrawPolygon[teal,thick](A,B,N,M)
\tkzDrawPolygon[teal,thick](C,B,P,Q)
\tkzDrawSegments[ultra thin,teal,dashed](A,Ha B,Hb C,Hc)
\tkzDrawPoints[teal,size=2](A,B,C,Ha,Hb,Hc,Pa)
\tkzDrawSegments[ultra thin,teal,dashed](B,S' A,S' A',B',Q P,C' M,S Ha,Pa)
\tkzDrawArc(A,S)(S')
\end{tikzpicture}

13.1.3. Option colinear at

How to use \( K \)

\begin{tikzpicture}
\tkzDefPoints{0/0/A,5/0/B,1/2/C}
\tkzDefPointWith[colinear=at C](A,B) \tkzGetPoint{G}
\tkzDefPointWith[colinear=at C, K=0.5](A,B) \tkzGetPoint{H}
\tkzLabelPoints(A,B,C,G,H)
\tkzDrawPoints[A,B,C,G,H]
\tkzDrawSegments[vector](A,B C,D)
\end{tikzpicture}

13.1.4. Option colinear at

With \( K = \sqrt{2}/2 \)

\begin{tikzpicture}
\tkzDefPoints{1/1/A,4/2/B,2/2/C}
\tkzDefPointWith[colinear=at C, K=sqrt(2)/2](A,B) \tkzGetPoint{D}
\tkzDrawPoints[A,B,C,G,H]
\tkzDrawSegments[vector](A,B C,D)
\end{tikzpicture}

13.1.5. Option orthogonal

\( AB=AC \) since \( K = 1 \).
13. Defining points using a vector

13.1.6. Option orthogonal

With $K = -1$ $\text{OK} = \text{OI}$ since $|K| = 1$ then $\text{OI} = \text{OJ} = \text{OK}$. 

13.1.7. Option orthogonal more complicated example
13. Defining points using a vector

13.1.8. Options colinear and orthogonal

\begin{tikzpicture}[scale=1.2, vect/.style={->,shorten >=1pt,>=latex'}]
\tkzDefPoints{2/1/A,6/2/B}
\tkzDefPointWith[orthogonal,K=.5](A,B)
\tkzGetPoint{C}
\tkzDefPointWith[colinear=at C,K=.5](A,B)
\tkzGetPoint{D}
\tkzMarkRightAngle[fill=gray!20](B,A,C)
\tkzDrawSegments[vect](A,B A,C C,D)
\tkzDrawPoints(A,...,D)
\end{tikzpicture}

13.1.9. Option orthogonal normed

K = 1 AC = 1.

\begin{tikzpicture}[scale=1.2, vect/.style={->,shorten >=1pt,>=latex'}]
\tkzDefPoints{2/3/A,4/2/B}
\tkzDefPointWith[orthogonal normed](A,B)
\tkzGetPoint{C}
\tkzDrawPoints[color=red](A,B,C)
\tkzDrawSegments[vect](A,B A,C)
\tkzMarkRightAngle[fill=gray!20](B,A,C)
\end{tikzpicture}

13.1.10. Option orthogonal normed and K=2

K = 2 therefore AC = 2.

\begin{tikzpicture}[scale=1.2, vect/.style={->,shorten >=1pt,>=latex'}]
\tkzDefPoints{2/3/A,5/1/B}
\tkzDefPointWith[orthogonal normed,K=2](A,B)
\tkzGetPoint{C}
\tkzDrawPoints[color=red](A,B,C)
\tkzDefCircle[R](A,2) \tkzGetPoint{a}
\tkzDrawCircle(A,a)
\tkzDrawSegments[vect](A,B A,C)
\tkzMarkRightAngle[fill=gray!20](B,A,C)
\tkzLabelPoints[above=3pt](A,B,C)
\end{tikzpicture}

13.1.11. Option linear

Here K = 0.5. This amounts to applying a homothety or a multiplication of a vector by a real. Here is the middle of [AB].

\begin{tikzpicture}[scale=1.2]
\tkzDefPoints{1/3/A,4/2/B}
\tkzDefPointWith[linear,K=0.5](A,B)
\tkzGetPoint{C}
\tkzDrawPoints[color=red](A,B,C)
\tkzDrawSegment(A,B)
\tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
13.1.12. Option linear normed

In the following example $AC = 1$ and $C$ belongs to $(AB)$.

\begin{tikzpicture}[scale=1.2]
\tkzDefPoints{1/3/A,4/2/B}
\tkzDefPointWith[linear normed](A,B)
\tkzGetPoint{C}
\tkzDrawPoints[color=red](A,B,C)
\tkzDrawSegment(A,B)
\tkzLabelSegment(A,C){$1$}
\tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}

13.2. \texttt{tkzGetVectxy}

Retrieving the coordinates of a vector.

\begin{align*}
\texttt{tkzGetVectxy}(A,B)\{\text{(text)}\}
\end{align*}

Allows to obtain the coordinates of a vector.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨point⟩{name of macro} \texttt{tkzGetVectxy}(A,B){V}</td>
<td>Vx,Vy: coordinates of $\overrightarrow{AB}$</td>
<td></td>
</tr>
</tbody>
</table>

13.2.1. Coordinate transfer with \texttt{tkzGetVectxy}

\begin{tikzpicture}
\tkzDefPoints{0/0/O,1/1/A,4/2/B}
\tkzGetVectxy(A,B){v}
\tkzDefPoint(\vx,\vy){V}
\tkzDrawSegment[->,color=red](O,V)
\tkzDrawSegment[->,color=blue](A,B)
\tkzDrawPoints(A,B,O)
\tkzLabelPoints(A,B,O,V)
\end{tikzpicture}

14. Straight lines

It is of course essential to draw straight lines, but before this can be done, it is necessary to be able to define certain particular lines such as mediators, bisectors, parallels or even perpendiculars. The principle is to determine two points on the straight line.

14.1. Definition of straight lines

\texttt{tkzDefLine[⟨local options⟩]((pt1,pt2)) or ((pt1,pt2,pt3))}

The argument is a list of two or three points. Depending on the case, the macro defines one or two points necessary to obtain the line sought. Either the macro \texttt{tkzGetPoint} or the macro \texttt{tkzGetPoints} must be used. I used the term “mediator” to designate the perpendicular bisector line at the middle of a line segment.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨(pt1,pt2)⟩</td>
<td><a href="(A,B)">mediator</a></td>
<td>mediator of the segment $[A,B]$</td>
</tr>
<tr>
<td>⟨(pt1,pt2,pt3)⟩</td>
<td><a href="(A,B,C)">bisector</a></td>
<td>bisector of $ABC$</td>
</tr>
<tr>
<td>⟨(pt1,pt2,pt3)⟩</td>
<td><a href="(A,B,C)">altitude</a></td>
<td>altitude from $B$</td>
</tr>
<tr>
<td>⟨pt1⟩</td>
<td><a href="(0)">tangent at=A</a></td>
<td>tangent at $A$ on the circle center $O$</td>
</tr>
<tr>
<td>⟨(pt1,pt2)⟩</td>
<td><a href="(0,B)">tangent from=A</a></td>
<td>circle center $O$ through $B$</td>
</tr>
</tbody>
</table>
### 14. Straight lines

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mediator</td>
<td></td>
<td>perpendicular bisector of a line segment</td>
</tr>
<tr>
<td>perpendicular=through...</td>
<td>mediator</td>
<td>perpendicular to a straight line passing through a point</td>
</tr>
<tr>
<td>orthogonal=through...</td>
<td>mediator</td>
<td>see above</td>
</tr>
<tr>
<td>parallel=through...</td>
<td>mediator</td>
<td>parallel to a straight line passing through a point</td>
</tr>
<tr>
<td>bisector</td>
<td>mediator</td>
<td>bisector of an angle defined by three points</td>
</tr>
<tr>
<td>bisector out</td>
<td>mediator</td>
<td>exterior angle bisector</td>
</tr>
<tr>
<td>symmedian</td>
<td>mediator</td>
<td>symmedian from a vertex</td>
</tr>
<tr>
<td>altitude</td>
<td>mediator</td>
<td>altitude from a vertex</td>
</tr>
<tr>
<td>euler</td>
<td>mediator</td>
<td>euler line of a triangle</td>
</tr>
<tr>
<td>tangent at</td>
<td>mediator</td>
<td>tangent at a point of a circle</td>
</tr>
<tr>
<td>tangent from</td>
<td>mediator</td>
<td>tangent from an exterior point</td>
</tr>
<tr>
<td>K</td>
<td>1</td>
<td>coefficient for the perpendicular line</td>
</tr>
<tr>
<td>normed</td>
<td>false</td>
<td>normalizes the created segment</td>
</tr>
</tbody>
</table>

#### 14.1.1. With mediator

\begin{tikzpicture}[rotate=25]
\tkzDefPoints{-2/0/A,1/2/B}
\tkzDefLine[mediator](A,B) \tkzGetPoints{C}{D}
\tkzDefPointWith[linear,K=.75](C,D) \tkzGetPoint{E}
\tkzDefMidPoint(A,B) \tkzGetPoint{I}
\tkzFillPolygon[color=teal!20](A,C,B,D)
\tkzDrawSegments(A,B C,D)
\tkzMarkRightAngle(B,I,C)
\tkzDrawSegments(D,B D,A)
\tkzDrawSegments(C,B C,A)
\end{tikzpicture}

#### 14.1.2. An envelope with option mediator

Based on a figure from O. Reboux with pst-eucl by D Rodriguez.

\begin{tikzpicture}[scale=.6]
\tkzInit[xmin=-6,ymin=-4,xmax=6,ymax=6]
\tkzClip
\tkzSetUpLine[thin,color=magenta]
\tkzDefPoint(0,0){O}
\tkzDefPoint(132:4){A}
\tkzDefPoint(5,0){B}
\foreach \ang in {5,10,...,360}{%\tkzDefPoint(\ang:5){M}
\tkzDrawSegments(A,M)
\tkzDrawSegments(M,B)
\tkzDrawSegments(D,B D,A)
\tkzDrawSegments(C,B C,A)
\end{tikzpicture}

#### 14.1.3. A parabola with option mediator

Based on a figure from O. Reboux with pst-eucl by D Rodriguez. It is not necessary to name the two points that define the mediator.
\begin{tikzpicture}[scale=.6]
\tkzInit[xmin=-6,ymin=-4,xmax=6,ymax=6]
\tkzClip
\tkzSetUpLine[thin,color=teal]
\tkzDefPoint(0,0){O}
\tkzDefPoint(132:5){A}
\tkzDefPoint(4,0){B}
\foreach \ang in {5,10,...,360}{
  \tkzDefPoint(\ang:4){M}
  \tkzDefLine[mediator](A,M)
  \tkzGetPoints{x}{y}
  \tkzDrawLine[add= 3 and 3](x,y)}
\end{tikzpicture}

14.1.4. With options bisector and normed

\begin{tikzpicture}[rotate=25,scale=.75]
\tkzDefPoints{0/0/C, 2/-3/A, 4/0/B}
\tkzDefLine[bisector,normed](B,A,C) \tkzGetPoint{a}
\tkzDrawLines[add= 0 and .5](A,B A,C)
\tkzShowLine[bisector,gap=4,size=2,color=red](B,A,C)
\tkzDrawLines[new,dashed,add= 0 and 3](A,a)
\end{tikzpicture}

14.1.5. With option parallel=through

Archimedes' Book of Lemmas proposition 1

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/O_1,0/1/O_2,0/3/A}
\tkzDefPoint(15:3){F}
\tkzInterLC(F,O_1)(O_1,A) \tkzGetSecondPoint{E}
\tkzDefLine[parallel=through O_2](E,F) \tkzGetPoint{x}
\tkzInterLC(x,O_2)(O_2,A) \tkzGetPoints{D}{C}
\tkzDrawCircles(O_1,A O_2,A)
\tkzDrawSegments[new](O_1,A E,F C,D)
\tkzDrawSegments[purple](A,E A,F)
\tkzDrawPoints(A,0_1,0_2,E,F,C,D)
\tkzLabelPoints(A,0_1,0_2,E,F,C,D)
\end{tikzpicture}
14.1.6. With option orthogonal and parallel

\begin{tikzpicture}
\tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-0.7/1/C}
\tkzDrawLine(A,B)
\tkzLabelLine[pos=1.25, below left](A,B){$(d_1)$}
\tkzDrawPoints(A,B,C)
\tkzDefLine[orthogonal=through C](B,A) \tkzGetPoint{c}
\tkzLabelLine[pos=1.25, left](C,c){$(\delta)$}
\tkzInterLL(A,B)(C,c) \tkzGetPoint{I}
\tkzMarkRightAngle(C,I,B)
\tkzDefLine[parallel=through C](A,B) \tkzGetPoint{c'}
\tkzDrawLine(C,c')
\tkzLabelLine[pos=1.25, below left](C,c'){$(d_2)$}
\tkzMarkRightAngle(I,C,c')
\end{tikzpicture}

14.1.7. With option altitude

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefLine[altitude](A,B,C) \tkzGetPoint{b}
\tkzDefLine[altitude](B,C,A) \tkzGetPoint{c}
\tkzDefLine[altitude](B,A,C) \tkzGetPoint{a}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints[blue](a,b,c)
\tkzDrawSegments[blue](A,a B,b C,c)
\tkzLabelPoints(A,B,c)
\tkzLabelPoints[above](C,a)
\tkzLabelPoints[above left](b)
\end{tikzpicture}

14.1.8. With option euler

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefLine[euler](A,B,C) \tkzGetPoints{h}{e}
\tkzDefTriangleCenter[circum](A,B,C) \tkzGetPoint{o}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints[red](A,B,C,h,e,o)
\tkzDrawLine[add= 2 and 2](h,e)
\tkzLabelPoints(A,B,C,h,e,o)
\tkzLabelPoints[above](C)
\end{tikzpicture}
14.1.9. Tangent passing through a point on the circle tangent at

\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(6,6){E}
  \tkzDefRandPointOn[circle=center O radius 3]
  \tkzGetPoint{A}
  \tkzDrawSegment(O,A)
  \tkzDrawCircle(O,A)
  \tkzDefLine[tangent at=A](O)
  \tkzGetPoint{h}
  \tkzDrawLine[add = 4 and 3](A,h)
  \tkzMarkRightAngle[fill=teal!30](O,A,h)
\end{tikzpicture}

14.1.10. Choice of contact point with tangents passing through an external point option tangent from

The tangent is not drawn. With option \texttt{at}, a point of the tangent is given by \texttt{tkzPointResult}. With option \texttt{from} you get two points of the circle with \texttt{tkzFirstPointResult} and \texttt{tkzSecondPointResult}. You can choose between these two points by comparing the angles formed with the outer point, the contact point and the center. The two possible angles have different directions. Angle counterclockwise refers to \texttt{tkzFirstPointResult}.

\begin{tikzpicture}[scale=1,rotate=-30]
  \tkzDefPoints{0/0/Q,0/2/A,6/-1/O}
  \tkzDefLine[tangent from = O](Q,A) \tkzGetPoints{R}{S}
  \tkzInterLC[near](O,Q)(Q,A) \tkzGetPoints{M}{N}
  \tkzDrawCircle(Q,M)
  \tkzDrawSegments[new,add = 0 and .2](O,R O,S)
  \tkzDrawSegments[gray](N,O R,Q S,Q)
  \tkzDrawPoints(O,Q,R,S,M,N)
  \tkzMarkAngle[gray,-stealth,size=1](O,R,Q)
  \tkzFindAngle(O,R,Q) \tkzGetAngle{an}
  \tkzLabelAngle(O,R,Q){$\pgfmathprintnumber{\an}^\circ$}
  \tkzMarkAngle[gray,-stealth,size=1](O,S,Q)
  \tkzFindAngle(O,S,Q) \tkzGetAngle{an}
  \tkzLabelAngle(O,S,Q){$\pgfmathprintnumber{\an}^\circ$}
  \tkzLabelPoints(Q,O,M,N,R)
  \tkzLabelPoints[above,text=red](S)
\end{tikzpicture}
14. Straight lines

14.1.11. Example of tangents passing through an external point

\begin{tikzpicture}[scale=.8]
\tkzDefPoints{0/0/c,1/0/d,3/0/a0}
def\tkzRadius{1}
\tkzDrawCircle(c,d)
\foreach \an in {0,10,...,350}{
  \tkzDefPointBy[rotation=center c angle \an](a0)
  \tkzGetPoint{a}
  \tkzDefLine[tangent from = a](c,d)
  \tkzGetPoints{e}{f}
  \tkzDrawLines(a,f a,e)
  \tkzDrawSegments(c,e c,f)}
\end{tikzpicture}

14.1.12. Example of Andrew Mertz

\begin{tikzpicture}[scale=.6]
\tkzDefPoint(100:8){A}\tkzDefPoint(50:8){B}
\tkzDefPoint(0,0){C} \tkzDefPoint(0,-4){R}
\tkzDrawCircle(C,R)
\tkzDefLine[tangent from = A](C,R) \tkzGetPoints{D}{E}
\tkzDefLine[tangent from = B](C,R) \tkzGetPoints{F}{G}
\tkzDrawSector[fill=teal!20,opacity=0.5](A,E)(D)
\tkzFillSector[color=teal,opacity=0.5](B,G)(F)
\end{tikzpicture}

http://www.texample.net/tikz/examples/
14.1.13. Drawing a tangent option \textit{tangent from}\begin{tikzpicture}[scale=.6]
tkzDefPoint(0,0){B}
tkzDefPoint(0,8){A}
tkzDefSquare(A,B)
tkzGetPoints{C}{D}
tkzDrawPolygon(A,B,C,D)
tkzClipPolygon(A,B,C,D)
tkzDefPoint(4,8){F}
tkzDefPoint(4,0){E}
tkzDefPoint(4,4){Q}
tkzFillPolygon[color = green](A,B,C,D)
tkzDrawCircle[fill = orange](B,A)
tkzDrawCircle[fill = purple](E,B)
tkzDefLine[tangent from = B](F,A)
tkzInterLL(F,tkzSecondPointResult)(C,D)
tkzInterLL(A,tkzPointResult)(F,E)
tkzDrawCircle[fill = yellow](tkzPointResult,Q)
tkzDefPointBy[projection= onto B--A](tkzPointResult)
tkzDrawCircle[fill = blue!50!black](tkzPointResult,A)
\end{tikzpicture}

15. Triangles

15.1. Definition of triangles \texttt{\textbackslash tkzDefTriangle}

The following macros will allow you to define or construct a triangle from \textit{at least} two points.
At the moment, it is possible to define the following triangles:

- two \textit{angles} determines a triangle with two angles;
- equilateral determines an equilateral triangle;
- isosceles right determines an isosceles right triangle;
- half determines a right-angled triangle such that the ratio of the measurements of the two adjacent sides to the right angle is equal to 2;
- pythagore determines a right-angled triangle whose side measurements are proportional to 3, 4 and 5;
- school determines a right-angled triangle whose angles are 30, 60 and 90 degrees;
- golden determines a right-angled triangle such that the ratio of the measurements on the two adjacent sides to the right angle is equal to $\Phi = 1.618034$, I chose "golden triangle" as the denomination because it comes from the golden rectangle and I kept the denomination "gold triangle" or "Euclid's triangle" for the isosceles triangle whose angles at the base are 72 degrees;
- euclid or gold for the gold triangle; in the previous version the option was "euclide" with an "e".
- cheops determines a third point such that the triangle is isosceles with side measurements proportional to 2, $\Phi$ and $\Phi$. 

\texttt{tkz-euclide}

\texttt{AlterMundus}
The points are ordered because the triangle is constructed following the direct direction of the trigonometric circle. This macro is either used in partnership with \texttt{tkzGetPoint} or by using \texttt{tkzPointResult} if it is not necessary to keep the name.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>two angles= #1 and #2</td>
<td>no defaut</td>
<td>triangle knowing two angles</td>
</tr>
<tr>
<td>equilateral</td>
<td>equilateral</td>
<td>equilateral triangle</td>
</tr>
<tr>
<td>half</td>
<td>equilateral</td>
<td>B rectangle $AB = 2BC$ $AC$ hypotenuse</td>
</tr>
<tr>
<td>isosceles right</td>
<td>equilateral</td>
<td>isosceles right triangle</td>
</tr>
<tr>
<td>pythagore</td>
<td>equilateral</td>
<td>proportional to the pythagorean triangle 3-4-5</td>
</tr>
<tr>
<td>pythagoras</td>
<td>equilateral</td>
<td>same as above</td>
</tr>
<tr>
<td>egyptian</td>
<td>equilateral</td>
<td>same as above</td>
</tr>
<tr>
<td>school</td>
<td>equilateral</td>
<td>angles of 30, 60 and 90 degrees</td>
</tr>
<tr>
<td>gold</td>
<td>equilateral</td>
<td>angles of 72, 72 and 36 degrees, A is the apex</td>
</tr>
<tr>
<td>euclid</td>
<td>equilateral</td>
<td>angles of 72, 72 and 36 degrees, C is the apex</td>
</tr>
<tr>
<td>golden</td>
<td>equilateral</td>
<td>angles of 72, 72 and 36 degrees, C is the apex</td>
</tr>
<tr>
<td>sublime</td>
<td>equilateral</td>
<td>angles of 72, 72 and 36 degrees, C is the apex</td>
</tr>
<tr>
<td>cheops</td>
<td>equilateral</td>
<td>$AC=BC$, $AC$ and $BC$ are proportional to 2 and $\Phi$.</td>
</tr>
<tr>
<td>swap</td>
<td>false</td>
<td>gives the symmetric point with respect to $AB$.</td>
</tr>
</tbody>
</table>

\texttt{tkzGetPoint} allows you to store the point otherwise \texttt{tkzPointResult} allows for immediate use.

15.1.1. Option \texttt{equilateral}

\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefTriangle[equilateral](A,B)
  \tkzGetPoint{C}
  \tkzDrawPolygons(A,B,C)
  \tkzDefTriangle[equilateral](B,A)
  \tkzGetPoint{D}
  \tkzDrawPolygon(B,A,D)
  \tkzMarkSegments[mark=s|](A,B B,C A,C A,D B,D)
\end{tikzpicture}
15.1.2. **Option two angles**

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(5,0){B} \tkzDefTriangle[two angles = 50 and 70](A,B) \tkzGetPoint{C} \tkzDrawSegment(A,B) \tkzDrawPoints(A,B) \tkzLabelPoints(A,B) \tkzDrawSegments[new](A,C B,C) \tkzDrawPoints[new](C) \tkzLabelPoints[above,new](C) \tkzLabelAngle[pos=1.4](B,A,C){$50^\circ$} \tkzLabelAngle[pos=0.8](C,B,A){$70^\circ$}
\end{tikzpicture}

15.1.3. **Option school**

The angles are 30, 60 and 90 degrees.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B} \tkzDefTriangle[school](A,B) \tkzGetPoint{C} \tkzMarkRightAngles(C,B,A) \tkzLabelAngle[pos=0.8](B,A,C){$30^\circ$} \tkzLabelAngle[pos=0.8](C,B,A){$90^\circ$} \tkzLabelAngle[pos=0.8](A,C,B){$60^\circ$} \tkzDrawSegments(A,B) \tkzDrawSegments[new](A,C B,C) \tkzLabelPoints(A,B) \tkzLabelPoints[above:new](C) \tkzLabelPoints[above:new](C)
\end{tikzpicture}

15.1.4. **Option pythagore**

This triangle has sides whose lengths are proportional to 3, 4 and 5.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B} \tkzDefTriangle[pythagore](A,B) \tkzGetPoint{C} \tkzDrawSegments(A,B) \tkzDrawSegments[new](A,C B,C) \tkzMarkRightAngles(A,B,C) \tkzDrawPoints[new](C) \tkzDrawPoints[above:new](A,B) \tkzDrawPoints[above:new](C)
\end{tikzpicture}

15.1.5. **Option pythagore and swap**

This triangle has sides whose lengths are proportional to 3, 4 and 5.
15. Triangles

15.1.6. Option golden

\begin{tikzpicture}[scale=.8]
% Code for golden triangle
\end{tikzpicture}

15.1.7. Option euclid

Euclid and golden are identical but the segment AB is a base in one and a side in the other.

\begin{tikzpicture}[scale=.75]
% Code for euclid triangle
\end{tikzpicture}
15.1.8. Option isosceles right

\begin{tikzpicture}
    \tkzDefPoint(0,0){A}
    \tkzDefPoint(4,0){B}
    \tkzDefTriangle[isosceles right](A,B)
    \tkzGetPoint{C}
    \tkzDrawPolygons(A,B,C)
    \tkzDrawPoints(A,B,C)
    \tkzMarkRightAngles(A,C,B)
    \tkzLabelPoints(A,B)
    \tkzLabelPoints[above](C)
\end{tikzpicture}

15.1.9. Option gold

\begin{tikzpicture}
    \tkzDefPoints{0/0/A,4/0/B}
    \tkzDefTriangle[gold](A,B)
    \tkzGetPoint{C}
    \tkzDrawPolygon(A,B,C)
    \tkzDrawPoints(A,B,C)
    \tkzLabelPoints[above](A,B)
    \tkzLabelPoints[below](C)
    \tkzMarkRightAngle(A,B,C)
    \tkzText(0,-2){$\dfrac{AC}{AB} = \varphi$}
\end{tikzpicture}
15.2. Specific triangles with \texttt{tkzDefSpcTriangle}

The centers of some triangles have been defined in the "points" section, here it is a question of determining the three vertices of specific triangles.

\texttt{\textbackslash tkzDefSpcTriangle[\{local options\}](p1,p2,p3){(r1,r2,r3)}}

The order of the points is important! \(p1p2p3\) defines a triangle then the result is a triangle whose vertices have as reference a combination with \texttt{name} and \(r1,r2,\ r3\). If \texttt{name} is empty then the references are \(r1,r2\) and \(r3\).

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>orthic</td>
<td>centroid</td>
<td>determined by endpoints of the altitudes ...</td>
</tr>
<tr>
<td>centroid or medial</td>
<td>centroid</td>
<td>intersection of the triangle's three triangle medians</td>
</tr>
<tr>
<td>in or incentral</td>
<td>centroid</td>
<td>determined with the angle bisectors</td>
</tr>
<tr>
<td>ex or excentral</td>
<td>centroid</td>
<td>determined with the excenters</td>
</tr>
<tr>
<td>extouch</td>
<td>centroid</td>
<td>formed by the points of tangency with the excircles</td>
</tr>
<tr>
<td>intouch or contact</td>
<td>centroid</td>
<td>formed by the points of tangency of the incircle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>each of the vertices</td>
</tr>
<tr>
<td>euler</td>
<td>centroid</td>
<td>formed by Euler points on the nine-point circle</td>
</tr>
<tr>
<td>symmedial</td>
<td>centroid</td>
<td>intersection points of the symmedians</td>
</tr>
<tr>
<td>tangential</td>
<td>centroid</td>
<td>formed by the lines tangent to the circumcircle</td>
</tr>
<tr>
<td>feuerbach</td>
<td>centroid</td>
<td>formed by the points of tangency of the nine-point ...</td>
</tr>
<tr>
<td></td>
<td></td>
<td>circle with the excircles</td>
</tr>
<tr>
<td>name</td>
<td>empty</td>
<td>used to name the vertices</td>
</tr>
</tbody>
</table>

15.2.1. How to name the vertices

With \texttt{\textbackslash tkzDefSpcTriangle[medial,name=M](A,B,C){_A,_B,_C}} you get three vertices named \(M_A,\ M_B\) and \(M_C\).

With \texttt{\textbackslash tkzDefSpcTriangle[medial](A,B,C){a,b,c}} you get three vertices named and labeled \(a,\ b\) and \(c\).

Possible \texttt{\textbackslash tkzDefSpcTriangle[medial,name=M_](A,B,C){A,B,C}} you get three vertices named \(M_A,\ M_B\) and \(M_C\).

15.3. Option \texttt{medial} or \texttt{centroid}

The geometric centroid of the polygon vertices of a triangle is the point \(G\) (sometimes also denoted \(M\)) which is also the intersection of the triangle's three triangle medians. The point is therefore sometimes called the median point. The centroid is always in the interior of the triangle.


In the following example, we obtain the Euler circle which passes through the previously defined points.
15.3.1. **Option in or incentral**

The incentral triangle is the triangle whose vertices are determined by the intersections of the reference triangle's angle bisectors with the respective opposite sides.

*Weisstein, Eric W. "Incentral triangle" From MathWorld–A Wolfram Web Resource.*

15.3.2. **Option ex or excentral**

The excentral triangle of a triangle $ABC$ is the triangle $I_aI_bI_c$ with vertices corresponding to the excenters of $ABC$. 
15.3.3. Option intouch or contact

The contact triangle of a triangle $ABC$, also called the intouch triangle, is the triangle formed by the points of tangency of the incircle of $ABC$ with $ABC$.

Weisstein, Eric W. "Contact triangle" From MathWorld–A Wolfram Web Resource.

We obtain the intersections of the bisectors with the sides.

15.3.4. Option extouch

The extouch triangle $T_aT_bT_c$ is the triangle formed by the points of tangency of a triangle $ABC$ with its excircles $J_a$, $J_b$, and $J_c$. The points $T_a$, $T_b$, and $T_c$ can also be constructed as the points which bisect the perimeter of $A_1A_2A_3$ starting at $A$, $B$, and $C$.


We obtain the points of contact of the exinscribed circles as well as the triangle formed by the centers of the exinscribed circles.
15. Triangles

15.3.5. Option orthic

Given a triangle ABC, the triangle $H_A H_B H_C$ whose vertices are endpoints of the altitudes from each of the vertices of ABC is called the orthic triangle, or sometimes the altitude triangle. The three lines $AH_A$, $BH_B$, and $CH_C$ are concurrent at the orthocenter $H$ of ABC.
15.3.6. **Option feuerbach**

The Feuerbach triangle is the triangle formed by the three points of tangency of the nine-point circle with the excircles.


The points of tangency define the Feuerbach triangle.

\begin{tikzpicture}[scale=1]
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,0){B}
\tkzDefPoint(0.5,2.5){C}
\tkzDefCircle[euler](A,B,C) \tkzGetPoint{N}
\tkzDefSpcTriangle[feuerbach, name=F](A,B,C){_a,_b,_c}
\tkzDefSpcTriangle[excentral, name=J](A,B,C){_a,_b,_c}
\tkzDefSpcTriangle[extouch, name=T](A,B,C){_a,_b,_c}
\tkzLabelPoints[below left](J_a,J_b,J_c)
\tkzClipBB \tkzShowBB
\tkzDrawCircle[purple](N,F_a)
\tkzDrawPolygon(A,B,C)
\tkzDrawPolygon[new](F_a,F_b,F_c)
\tkzDrawCircles[gray](J_a,F_a J_b,F_b J_c,F_c)
\tkzDrawPoints[blue](J_a,J_b,J_c, F_a,F_b,F_c,A,B,C)
\tkzLabelPoints(A,B,F_c)
\tkzLabelPoints[above](C)
\tkzLabelPoints[right](F_a)
\tkzLabelPoints[left](F_b)
\end{tikzpicture}

15.3.7. **Option tangential**

The tangential triangle is the triangle $T_a T_b T_c$ formed by the lines tangent to the circumcircle of a given triangle $ABC$ at its vertices. It is therefore antipedal triangle of $ABC$ with respect to the circumcenter $O$.


\begin{tikzpicture}[scale=.5,rotate=80]
\tkzDefPoints{0/0/A,6/0/B,1.8/4/C}
\tkzDefSpcTriangle[tangential, name=T](A,B,C){_a,_b,_c}
\tkzDrawPolygon(A,B,C)
\tkzDrawPolygon[new](T_a,T_b,T_c)
\tkzDrawPoints(A,B,C)
\tkzDrawPoints[new](T_a,T_b,T_c)
\tkzDrawCircle[circum](A,B,C)
\tkzGetPoint{O}
\tkzDrawCircle(O,A)
\tkzLabelPoints(A)
\tkzLabelPoints[above](B)
\tkzLabelPoints[left](C)
\tkzLabelPoints[right](F_a)
\tkzLabelPoints[left](F_b)
\end{tikzpicture}
15.3.8. Option euler

The Euler triangle of a triangle ABC is the triangle $E_AE_BE_C$ whose vertices are the midpoints of the segments joining the orthocenter $H$ with the respective vertices. The vertices of the triangle are known as the Euler points, and lie on the nine-point circle.

15.3.9. Option euler and Option orthic
15.3.10. Option symmedial

The symmedial triangle $K_A K_B K_C$ is the triangle whose vertices are the intersection points of the symmedians with the reference triangle $ABC$.

15.4. Permutation of two points of a triangle

Example:

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Example</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(pt1, pt2, pt3)$</td>
<td>$\text{tkzPermute}(A, B, C)$</td>
<td>$A, B, A, C$ are unchanged, $B, C$ exchange their position</td>
</tr>
</tbody>
</table>

The triangle is unchanged.
15.4.1. Modification of the school triangle

This triangle is constructed from the segment [AB] on [A, x).

If we want the segment [AC] to be on [A, x), we just have to swap B and C.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B,6/0/x}
\tkzDefTriangle[school](A,B)
\tkzGetPoint{C}
\tkzPermuate(A,B,C)
\tkzDrawSegments(A,B C,x)
\tkzDrawSegments(A,C B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,C,x)
\tkzLabelPoints[above](B)
\tkzMarkRightAngles(C,B,A)
\end{tikzpicture}

Remark: Only the first point is unchanged. The order of the last two parameters is not important.

16. Definition of polygons

16.1. Defining the points of a square

We have seen the definitions of some triangles. Let us look at the definitions of some quadrilaterals and regular polygons.

\begin{verbatim}
\tkzDefSquare((pt1,pt2))
\end{verbatim}

The square is defined in the forward direction. From two points, two more points are obtained such that the four taken in order form a square. The square is defined in the forward direction.

The results are in \texttt{tkzFirstPointResult} and \texttt{tkzSecondPointResult}.

We can rename them with \texttt{tkzGetPoints}.

<table>
<thead>
<tr>
<th>Arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(⟨pt1,pt2⟩)</td>
<td>\texttt{tkzDefSquare((A,B))}</td>
<td>The square is defined in the direct direction.</td>
</tr>
</tbody>
</table>

16.1.1. Using \texttt{tkzDefSquare} with two points

Note the inversion of the first two points and the result.

\begin{verbatim}
\begin{tikzpicture}[scale=.5]
\tkzDefPoint(0,0){A} \tkzDefPoint(3,0){B}
\tkzDefSquare(A,B)
\tkzDrawPolygon(A,B,tkzFirstPointResult,\% tkzSecondPointResult)
\tkzDefSquare(B,A)
\tkzDrawPolygon(B,A,tkzFirstPointResult,\% tkzSecondPointResult)
\end{tikzpicture}
\end{verbatim}

We may only need one point to draw an isosceles right-angled triangle so we use \texttt{tkzGetFirstPoint} or \texttt{tkzGetSecondPoint}.
16.1.2. Use of \texttt{tkzDefSquare} to obtain an isosceles right-angled triangle

\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefSquare(A,B) \tkzGetFirstPoint{C}
  \tkzDrawSegment(A,B)
  \tkzDrawSegments[new](A,C B,C)
  \tkzMarkRightAngles(A,B,C)
  \tkzDrawPoints(A,B) \tkzDrawPoint[new](C)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[new,above](C)
\end{tikzpicture}

16.1.3. Pythagorean Theorem and \texttt{tkzDefSquare}

\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){C}
  \tkzDefPoint(4,0){A}
  \tkzDefPoint(0,3){B}
  \tkzDefSquare(B,A) \tkzGetPoints{E}{F}
  \tkzDefSquare(A,C) \tkzGetPoints{G}{H}
  \tkzDefSquare(C,B) \tkzGetPoints{I}{J}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon(A,C,G,H)
  \tkzDrawPolygon(C,B,I,J)
  \tkzDrawPolygon(B,A,E,F)
  \tkzLabelSegment(A,C){$a$}
  \tkzLabelSegment[right](C,B){$b$}
  \tkzLabelSegment[swap](A,B){$c$}
\end{tikzpicture}

16.2. Defining the points of a rectangle

\texttt{\tkzDefRectangle((pt1,pt2))}

The rectangle is defined in the forward direction. From two points, two more points are obtained such that the four taken in order form a rectangle. The two points passed in arguments are the ends of a diagonal of the rectangle. The sides are parallel to the axes.

The results are in \texttt{tkzFirstPointResult} and \texttt{tkzSecondPointResult}.

We can rename them with \texttt{tkzGetPoints}.

<table>
<thead>
<tr>
<th>Arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\texttt{pt1,pt2})</td>
<td>\texttt{\tkzDefRectangle((A,B))}</td>
<td>The rectangle is defined in the direct direction.</td>
</tr>
</tbody>
</table>

16.2.1. Example of a rectangle definition

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/2/C}
  \tkzDefRectangle(A,C) \tkzGetPoints{B,D}
  \tkzDrawPolygon[fill=teal!15](A,...,D)
\end{tikzpicture}
16.3. Definition of parallelogram

Defining the points of a parallelogram. It is a matter of completing three points in order to obtain a parallelogram.

\texttt{\tkzDefParallelogram((pt1,pt2,pt3))}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2,pt3)</td>
<td>no default</td>
<td>Three points are necessary</td>
</tr>
</tbody>
</table>

From three points, another point is obtained such that the four taken in order form a parallelogram. The result is in \texttt{tkzPointResult}.

We can rename it with the name \texttt{tkzGetPoint}...

16.3.1. Example of a parallelogram definition

```
\begin{tikzpicture}[scale=1]
  \tkzDefPoints{0/0/A,3/0/B,4/2/C}
  \tkzDefParallelogram(A,B,C)
  \tkzGetPoint{D}
  \tkzDrawPolygon(A,B,C,D)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above right](C,D)
  \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

16.4. The golden rectangle

\texttt{\tkzDefGoldenRectangle((point,point))}

The macro determines a rectangle whose size ratio is the number $\Phi$.

The created points are in \texttt{tkzFirstPointResult} and \texttt{tkzSecondPointResult}.

They can be obtained with the macro \texttt{tkzGetPoints}. The following macro is used to draw the rectangle.

\begin{verbatim}
\begin{tikzpicture}[scale=.6]
  \tkzDefPoint(0,0){A} \tkzDefPoint(8,0){B}
  \tkzDefGoldRectangle(A,B) \tkzGetPoints{C}{D}
  \tkzDefGoldRectangle(B,C) \tkzGetPoints{E}{F}
  \tkzDefGoldRectangle(C,E) \tkzGetPoints{G}{H}
  \tkzDrawPolygon(A,B,C,D)
  \tkzDrawSegments(E,F G,H)
\end{tikzpicture}
\end{verbatim}

16.4.1. Golden Rectangles

\begin{verbatim}
\begin{tikzpicture}[scale=.6]
  \tkzDefPoint(0,0){A} \tkzDefPoint(8,0){B}
  \tkzDefGoldRectangle(A,B) \tkzGetPoints{C}{D}
  \tkzDefGoldRectangle(B,C) \tkzGetPoints{E}{F}
  \tkzDefGoldRectangle(C,E) \tkzGetPoints{G}{H}
  \tkzDrawPolygon(A,B,C,D)
  \tkzDrawSegments(E,F G,H)
\end{tikzpicture}
\end{verbatim}
16.4.2. Construction of the golden rectangle

Without the previous macro here is how to get the golden rectangle.

```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(8,0){B}
  \tkzDefMidPoint(A,B)
  \tkzGetPoint{I}
  \tkzDefSquare(A,B)\tkzGetPoints{C}{D}
  \tkzInterLC(A,B)(I,C)\tkzGetPoints{G}{E}
  \tkzDefPointWith[colinear= at C](E,B)
  \tkzGetPoint{F}
  \tkzDefPointBy[projection=onto D--C ](E)
  \tkzGetPoint{H}
  \tkzDrawArc[style=dashed](I,E)(D)
  \tkzDrawPolygon(A,B,C,D)
  \tkzDrawPoints(C,D,E,F,H)
  \tkzLabelPoints(A,B,C,D,E,F,H)
  \tkzLabelPoints[above](C,D,F,H)
  \tkzDrawSegments[style=dashed,color=gray](E,F C,F B,E F,H H,C E,H)
\end{tikzpicture}
```

16.5. Regular polygon

```
\tkzDefRegPolygon[(local_options)]((pt1,pt2))
```

From the number of sides, depending on the options, this macro determines a regular polygon according to its center or one side.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)</td>
<td>(O,A)</td>
<td>with option &quot;center&quot;, O is the center of the polygon.</td>
</tr>
<tr>
<td>(pt1,pt2)</td>
<td>(A,B)</td>
<td>with option &quot;side&quot;, [AB] is a side.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>P</td>
<td>The vertices are named P1,P2,...</td>
</tr>
<tr>
<td>sides</td>
<td>5</td>
<td>number of sides.</td>
</tr>
<tr>
<td>center</td>
<td>center</td>
<td>The first point is the center.</td>
</tr>
<tr>
<td>side</td>
<td>center</td>
<td>The two points are vertices.</td>
</tr>
<tr>
<td>Options</td>
<td>TikZ</td>
<td>...</td>
</tr>
</tbody>
</table>

16.5.1. Option center

```
\begin{tikzpicture}
  \tkzDefPoints{0/0/P0,0/0/Q0,2/0/P1}
  \tkzDefMidPoint(P0,P1) \tkzGetPoint{Q1}
  \tkzDefRegPolygon[center,sides=7](P0,P1)
  \tkzDefMidPoint(P1,P2) \tkzGetPoint{Q1}
  \tkzDefRegPolygon[center,sides=7,name=Q](P0,Q1)
  \tkzFillPolygon[teal!20](Q0,Q1,P2,Q2)
  \tkzDrawPolygon(P1,P...,P7)
  \foreach \j in {1,...,7} \%
    \tkzDrawSegment[black](P0,Q\j)}
\end{tikzpicture}
```
16.5.2. Option \texttt{side}

\begin{tikzpicture}[scale=1]
\tkzDefPoints{-4/0/A, -1/0/B}
\tkzDefRegPolygon[side,sides=5,name=P](A,B)
\tkzDrawPolygon[thick](P1,P\ldots,P5)
\end{tikzpicture}
17. Circles

Among the following macros, one will allow you to draw a circle, which is not a real feat. To do this, you will need to know the center of the circle and either the radius of the circle or a point on the circumference. It seemed to me that the most frequent use was to draw a circle with a given center passing through a given point. This will be the default method, otherwise you will have to use the R option. There are a large number of special circles, for example the circle circumscribed by a triangle.

- I have created a first macro \tkzDefCircle which allows, according to a particular circle, to retrieve its center and the measurement of the radius in cm. This recovery is done with the macros \tkzGetPoint and \tkzGetLength;
- then a macro \tkzDrawCircle;
- then a macro that allows you to color in a disc, but without drawing the circle \tkzFillCircle;
- sometimes, it is necessary for a drawing to be contained in a disk, this is the role assigned to \tkzClipCircle;
- it finally remains to be able to give a label to designate a circle and if several possibilities are offered, we will see here \tkzLabelCircle.

17.1. Characteristics of a circle: \tkzDefCircle

This macro allows you to retrieve the characteristics (center and radius) of certain circles.

\[
\text{\texttt{\tkzDefCircle[(local options)]((A,B)) or ((A,B,C))}}
\]

\[O \rightarrow \]

Attention the arguments are lists of two or three points. This macro is either used in partnership with \tkzGetPoints to obtain the center and a point on the circle, or by using \tkzFirstPointResult and \tkzSecondPointResult if it is not necessary to keep the results. You can also use \tkzGetLength to get the radius.

| arguments example explanation |
|-----------------------------|------------------|
| ((pt1,pt2)) or ((pt1,pt2,pt3)) ((A,B)) [AB] is radius A is the center |

<table>
<thead>
<tr>
<th>options default definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>R circum circle characterized by a center and a radius</td>
</tr>
<tr>
<td>diameter circum circle characterized by two points defining a diameter</td>
</tr>
<tr>
<td>circum circum circle circumscribed of a triangle</td>
</tr>
<tr>
<td>in circum incircle a triangle</td>
</tr>
<tr>
<td>ex circum excircle of a triangle</td>
</tr>
<tr>
<td>euler or nine circum Euler's Circle</td>
</tr>
<tr>
<td>spieker circum Spieker Circle</td>
</tr>
<tr>
<td>apollonius circum circle of Apollonius</td>
</tr>
<tr>
<td>orthogonal from circum {orthogonal from = A }(0,M)</td>
</tr>
<tr>
<td>orthogonal through circum {orthogonal through = A and B}(0,M)</td>
</tr>
<tr>
<td>K 1 coefficient used for a circle of Apollonius</td>
</tr>
</tbody>
</table>

\[In the following examples, I draw the circles with a macro not yet presented. You may only need the center and a point on the circle.\]
17.1.1. Example with option \texttt{R}

We obtain with the macro \texttt{tkzGetPoint} a point of the circle which is the East pole.

\begin{tikzpicture}[scale=1]
\tkzDefPoint(3,3){C}
\tkzDefPoint(5,5){A}
\tkzCalcLength(A,C) \tkzGetLength{rAC}
\tkzDefCircle[R](C,rAC) \tkzGetPoint{B}
\tkzDrawCircle(C,B)
\tkzDrawSegment(C,A)
\tkzLabelSegment[above left](C,A){$2\sqrt{2}$}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,C,B)
\end{tikzpicture}

17.1.2. Example with option \texttt{diameter}

It is simpler here to search directly for the middle of $[AB]$. The result is the center and if necessary

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\tkzDefPoint(2,2){B}
\tkzDefCircle[diameter](O,B) \tkzGetPoint{A}
\tkzDrawCircle(A,B)
\tkzDrawPoints(O,A,B)
\tkzDrawSegment(O,B)
\tkzLabelPoints(O,A,B)
\tkzLabelSegment[above left](O,A){$\sqrt{2}$}
\tkzLabelSegment[above left](A,B){$\sqrt{2}$}
\tkzMarkSegments[mark=s||](O,A A,B)
\end{tikzpicture}

17.1.3. Circles inscribed and circumscribed for a given triangle

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(2,2){A} \tkzDefPoint(5,-2){B}
\tkzDefPoint(1,-2){C}
\tkzDefCircle[in](A,B,C)
\tkzGetPoints{I}{x}
\tkzDefCircle[circum](A,B,C)
\tkzGetPoint{K}
\tkzDrawCircles[new](I,x K,A)
\tkzDrawPoints[below](B,C)
\tkzLabelPoints[below](B,C)
\tkzLabelPoints[above left](A,I,K)
\tkzDrawPolygon(A,B,C)
\tkzLabelPoints(A,B,C,I,K)
\end{tikzpicture}

17.1.4. Example with option \texttt{ex}

We want to define an excircle of a triangle relatively to point $C$
17.1.5. Euler's circle for a given triangle with option `euler`

We verify that this circle passes through the middle of each side.
17.1.6. Apollonius circles for a given segment option apollonius

\begin{tikzpicture}[scale=0.75]
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
\tkzDefCircle[apollonius,K=2](A,B)
\tkzGetPoints{K1}{x}
\tkzDrawCircle[color = teal!50!black, fill=teal!20,opacity=.4](K1,x)
\tkzDefCircle[apollonius,K=3](A,B)
\tkzGetPoints{K2}{y}
\tkzDrawCircle[color=orange!50, fill=orange!20,opacity=.4](K2,y)
\tkzLabelPoints[below](A,B,K1,K2)
\tkzDrawPoints(A,B,K1,K2)
\tkzDrawLine[add=.2 and 1](A,B)
\end{tikzpicture}

17.1.7. Circles exinscribed to a given triangle option ex

You can also get the center and the projection of it on one side of the triangle.
with \tkzGetFirstPoint{Jb} and \tkzGetSecondPoint{Tb}.

\begin{tikzpicture}[scale=.6]
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,0){B}
\tkzDefPoint(1,2.5){C}
\tkzDefCircle[ex](A,B,C) \tkzGetPoints{I}{i}
\tkzDefCircle[ex](C,A,B) \tkzGetPoints{J}{j}
\tkzDefCircle[ex](B,C,A) \tkzGetPoints{K}{k}
\tkzDefCircle[in](B,C,A) \tkzGetPoints{O}{o}
\tkzDrawCircles[new](J,j I,i K,k 0,o)
\tkzDrawLines[add=1.5 and 1.5](A,B A,C B,C)
\tkzDrawPolygon[purple](I,J,K)
\tkzDrawSegments[new](A,K B,J C,I)
\tkzDrawPoints(A,B,C)
\tkzDrawPoints[new](I,J,K)
\tkzLabelPoints(A,B,C,I,J,K)
\end{tikzpicture}

17.1.8. Spieker circle with option spieker

The incircle of the medial triangle $M_a M_b M_c$ is the Spieker:

\begin{tikzpicture}[scale=0.6]
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,0){B}
\tkzDefPoint(1,2.5){C}
\tkzDefCircle[spieker](A,B,C) \tkzGetPoints{O}{o}
\tkzDrawCircle[spieker](A,B,C)
\tkzDrawSegments[add=1.5 and 1.5](A,K B,J C,I)
\tkzDrawPoints(A,B,C)
\tkzDrawPoints[new](I,J,K)
\tkzLabelPoints(A,B,C,I,J,K)
\end{tikzpicture}
17. Circles

17.2. Projection of excenters

\begin{tikzpicture}[scale=1]
\tkzDefPoints{ 0/0/A,4/0/B,0.8/4/C}
\tkzDefSpcTriangle[medial](A,B,C){M_a,M_b,M_c}
\tkzDefTriangleCenter[spieker](A,B,C)
\tkzGetPoint{S_p}
\tkzDrawPolygon(A,B,C)
\tkzDrawPolygon[cyan](M_a,M_b,M_c)
\tkzDrawPoints(B,C,A)
\tkzDefCircle[spieker](A,B,C)
\tkzDrawPoints[new](M_a,M_b,M_c,S_p)
\tkzDrawCircle[new](tkzFirstPointResult,tkzSecondPointResult)
\tkzLabelPoints[right](M_a)
\tkzLabelPoints[left](M_b)
\tkzLabelPoints[below](A,B,M_c,S_p)
\tkzLabelPoints[above](C)
\end{tikzpicture}

\begin{tikzpicture}[scale=1]
\tkzDefProjExcenter[⟨local options⟩](⟨A,B,C⟩)(⟨a,b,c⟩){⟨X,Y,Z⟩}
\end{tikzpicture}

Each excenter has three projections on the sides of the triangle ABC. We can do this with one macro \( \text{\texttt{tkzDefProjExcenter}}[\text{\texttt{name}}=\text{\texttt{J}}](\text{\texttt{A,B,C}})(\text{\texttt{a,b,c}})\{\text{\texttt{X,Y,Z}}}\).

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>no defaut</td>
<td>used to name the vertices</td>
</tr>
<tr>
<td>arguments</td>
<td>default</td>
<td>definition</td>
</tr>
<tr>
<td>(pt1=(\alpha_1),pt2=(\alpha_2),...)</td>
<td>no defaut</td>
<td>Each point has a assigned weight</td>
</tr>
</tbody>
</table>
17.2.1. Excircles
\begin{tikzpicture} \[scale=.6\] \tikzset{line style/.append style={line width=.2pt}} \tikzset{label style/.append style={color=teal,font=\footnotesize}} \tkzDefPoints{0/0/A,5/0/B,0.8/4/C} \tkzDefSpcTriangle[excentral,name=J](A,B,C){a,b,c} \tkzDefSpcTriangle[intouch,name=I](A,B,C){a,b,c} \tkzDefProjExcenter[name=J](A,B,C){a,b,c}{X,Y,Z} \tkzDefCircle[in](A,B,C) \tkzGetPoint{I} \tkzGetSecondPoint{T} \tkzDrawCircles[red](Ja,Xa Jb,Yb Jc,Zc) \tkzDrawCircle(I,T) \tkzDrawPolygon[dashed, color=blue](Ja,Jb,Jc) \tkzDrawLines[add=1.5 and 1.5](A,C A,B B,C) \tkzDrawSegments(Ja,Xa Ja,Ya Ja,Jb,Ja,Jb,Jb,Jb,Jb,Jc,Jc,Jc,Jc) \tkzMarkRightAngles[size=.2,fill=gray!15](Ja,Za,B Ja,Xa,B Ja,Ya,C Jb,Yb,C Jc,Yc,A Jc,Zc,B Jc,Xc,C Jc,Ia,Ib,Ic,A) \tkzDrawSegments[blue](Jc,C Ja,A Jb,B) \tkzDrawPoints(A,B,C,Xa,Xb,Xc,Ja,Jb,Jc,Ja,Ia,Ib,Ic,Ya,Yb,Yc,Za,Zb,Zc) \tkzLabelPoints[right](O,A,B,C) \end{tikzpicture}

\subsection*{17.2.2. Orthogonal from}
Orthogonal circle of given center. \texttt{\tkzGetPoints{z1}{z2}} gives two points of the circle.

\begin{tikzpicture} \[scale=.75\] \tkzDefPoints{0/0/O,1/0/A} \tkzDefPoints{1.5/1.25/B,-2/-3/C} \tkzDefCircle[orthogonal from=B](O,A) \tkzGetPoints{z1}{z2} \tkzDefCircle[orthogonal from=C](O,A) \tkzGetPoints{t1}{t2} \tkzDrawCircle(O,A) \tkzDrawCircles[new](B,z1 C,t1) \tkzDrawPoints(t1,t2,C) \tkzDrawPoints(z1,z2,0,A,B) \tkzLabelPoints[right](O,A,B,C) \end{tikzpicture}

\subsection*{17.2.3. Orthogonal through}
Orthogonal circle passing through two given points.
17. Circles

17.3. Definition of circle by transformation; \tkzDefCircleBy

These transformations are:

- translation;
- homothety;
- orthogonal reflection or symmetry;
- central symmetry;
- orthogonal projection;
- rotation (degrees);
- inversion.

The choice of transformations is made through the options. The macro is \tkzDefCircleBy and the other for the transformation of a list of points \tkzDefCirclesBy. For example, we'll write:

\tkzDefCircleBy[translation= from A to A']{(O,M)}

O is the center and M is a point on the circle. The image is a circle. The new center is \tkzFirstPointResult and \tkzSecondPointResult is a point on the new circle. You can get the results with the macro \tkzGetPoints.

\begin{tikzpicture}[scale=1]
\tkzDefPoint(0,0){O}
\tkzDefPoint(1,0){A}
\tkzDrawCircle(O,A)
\tkzDefPoint(-1.5,-1.5){z1}
\tkzDefPoint(1.5,-1.25){z2}
\tkzDefCircle[orthogonal through=z1 and z2](O,A)
\tkzGetPoint{c}
\tkzDrawCircle[new](tkzPointResult,z1)
\tkzDrawPoints[new](O,A,z1,z2,c)
\tkzLabelPoints[right](O,A,z1,z2,c)
\end{tikzpicture}

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Definition</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>pt1,pt2</td>
<td>existing points (O,M)</td>
<td></td>
</tr>
<tr>
<td>options</td>
<td></td>
<td></td>
</tr>
<tr>
<td>translation= from #1 to #2</td>
<td><a href="O,M">translation=from A to B</a></td>
<td></td>
</tr>
<tr>
<td>homothety = center #1 ratio #2</td>
<td><a href="O,M">homothety=center A ratio .5</a></td>
<td></td>
</tr>
<tr>
<td>reflection = over #1--#2</td>
<td><a href="O,M">reflection=over A--B</a></td>
<td></td>
</tr>
<tr>
<td>symmetry = center #1</td>
<td><a href="O,M">symmetry=center A</a></td>
<td></td>
</tr>
<tr>
<td>projection = onto #1--#2</td>
<td><a href="O,M">projection=onto A--B</a></td>
<td></td>
</tr>
<tr>
<td>rotation = center #1 angle #2</td>
<td><a href="O,M">rotation=center 0 angle 30</a></td>
<td></td>
</tr>
<tr>
<td>inversion = center #1 through #2</td>
<td><a href="O,M">inversion =center 0 through A</a></td>
<td></td>
</tr>
</tbody>
</table>

The image is only defined and not drawn.
17.3.1. Translation

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(3,1){B} \tkzDefPoint(3,2){C} \tkzDefPoint(4,3){D} \tkzDefCircleBy[translation= from B to A](C,D) \tkzGetPoints{C'}{D'} \tkzDrawPoints[A,B,C,D,C',D'] \tkzDrawSegments[orange,->](A,B) \tkzDrawCircles(C,D C',D') \tkzLabelPoints[color=teal](A,B,C,C') \tkzLabelPoints[color=teal,above](D,D') \end{tikzpicture}

17.3.2. Reflection (orthogonal symmetry)

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(3,1){B} \tkzDefPoint(3,2){C} \tkzDefPoint(4,3){D} \tkzDefCircleBy[reflection = over A--B](C,D) \tkzGetPoints{C'}{D'} \tkzDrawPoints[A,B,C,D,C',D'] \tkzDrawLine[add =0 and 1](A,B) \tkzDrawCircles(C,D C',D') \tkzLabelPoints[color=teal](A,B,C,C') \tkzLabelPoints[color=teal,right](D,D') \end{tikzpicture}

17.3.3. Homothety

\begin{tikzpicture}[scale=1.2]
\tkzDefPoint(0,0){A} \tkzDefPoint(3,1){B} \tkzDefPoint(3,2){C} \tkzDefPoint(4,3){D} \tkzDefCircleBy[homothety=center A ratio .5](C,D) \tkzGetPoints{C'}{D'} \tkzDrawPoints[A,C,D,C',D'] \tkzDrawCircles(C,D C',D') \tkzLabelPoints[color=teal](A,C,C') \tkzLabelPoints[color=teal,right](D,D') \end{tikzpicture}
17.3.4. Symmetry

\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A} \tkzDefPoint(3,1){B}
  \tkzDefPoint(3,2){C} \tkzDefPoint(4,3){D}
  \tkzDefCircleBy[center B](C,D)
  \tkzGetPoints{C'}{D'}
  \tkzDrawPoints[teal](B,C,D,C',D')
  \tkzDrawLines[orange](C,C' D,D')
  \tkzDrawCircles(C,D C',D')
  \tkzLabelPoints[color=teal](A,C,C')
  \tkzLabelPoints[color=teal,above](D)
  \tkzLabelPoints[color=teal,below](D')
\end{tikzpicture}

17.3.5. Rotation

\begin{tikzpicture}[scale=0.5]
  \tkzDefPoint(3,-1){B} \tkzDefPoint(3,2){C} \tkzDefPoint(4,3){D}
  \tkzDefCircleBy[center B angle 90](C,D)
  \tkzGetPoints{C'}{D'}
  \tkzDrawPoints[teal](B,C,D,C',D')
  \tkzDrawCircles(C,D C',D')
\end{tikzpicture}

17.3.6. Inversion

\begin{tikzpicture}[scale=1.5]
  \tkzSetUpPoint[size=3,color=red,fill=red!20]
  \tkzSetUpStyle[color=purple,ultra thin]{st1}
  \tkzSetUpStyle[color=cyan,ultra thin]{st2}
  \tkzDefPoint(2,0){A} \tkzDefPoint(3,0){B}
  \tkzDefPoint(3,2){C} \tkzDefPoint(4,2){D}
  \tkzDefCircleBy[center B through A](C,D)
  \tkzGetPoints{C'}{D'}
  \tkzDrawPoints(A,B,C,D,C',D')
  \tkzDrawCircles(B,A)
  \tkzDrawCircles(st1)(C,D)
  \tkzDrawCircles(st2)(C',D')
\end{tikzpicture}

18. Intersections

It is possible to determine the coordinates of the points of intersection between two straight lines, a straight line and a circle, and two circles.

The associated commands have no optional arguments and the user must determine the existence of the intersection points himself.
18. Intersections

18.1. Intersection of two straight lines $\texttt{tkzInterLL}$

$\texttt{\textbackslash tkzInterLL(⟨A, B⟩, ⟨C, D⟩)}$

 Defines the intersection point $\texttt{tkzPointResult}$ of the two lines $\texttt{(AB)}$ and $\texttt{(CD)}$. The known points are given in pairs (two per line) in brackets, and the resulting point can be retrieved with the macro $\texttt{tkzDefPoint}$.

18.1.1. Example of intersection between two straight lines

\begin{tikzpicture}[rotate=-45,scale=.75]
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,5){B}
  \tkzDefPoint(3,6){C}
  \tkzDefPoint(5,2){D}
  \tkzDrawLines(A,B C,D)
  \tkzInterLL(A,B)(C,D)
  \tkzGetPoint{I}
  \tkzDrawPoints[\color=blue](A,B,C,D)
  \tkzDrawPoint[\color=red](I)
\end{tikzpicture}

18.2. Intersection of a straight line and a circle $\texttt{tkzInterLC}$

As before, the line is defined by a couple of points. The circle is also defined by a couple:

\begin{itemize}
  \item (O,C) which is a pair of points, the first is the center and the second is any point on the circle.
  \item (O,r) The $r$ measure is the radius measure.
\end{itemize}

$\texttt{\textbackslash tkzInterLC(⟨\texttt{options}⟩,⟨A, B⟩,⟨O, C⟩ or ⟨O, r⟩ or ⟨O, C, D⟩)}$

So the arguments are two couples.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>(O,C) determines the circle</td>
</tr>
<tr>
<td>R</td>
<td>N</td>
<td>(O, 1) unit 1 cm</td>
</tr>
</tbody>
</table>

with nodes $\texttt{N}$ (O,C,D) CD is a radius

common=pt pt is common point; $\texttt{tkzFirstPoint}$ gives the other point

near $\texttt{tkzFirstPoint}$ is the closest point to the first point of the line

The macro defines the intersection points I and J of the line (AB) and the center circle O with radius r if they exist; otherwise, an error will be reported in the .log file. with nodes avoids you to calculate the radius which is the length of [CD]. If common and near are not used then $\texttt{tkzFirstPoint}$ is the smallest angle (angle with $\texttt{tkzSecondPoint}$ and the center of the circle).

\begin{tikzpicture}
  \tkzDefPoint(1,0){O}
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(3,2){B}
  \tkzCircle(O,R)
  \tkzInterLC(⟨O,A⟩)(⟨O′,B⟩)
\end{tikzpicture}

So the arguments are two couples which define a line and a circle with a center and a point on the circle. If there is a non empty intersection between these the line and the circle then the test $\texttt{\iftkzFlagLC}$ gives true.
18.2.1. test line-circle intersection

\begin{tikzpicture}[scale=1]
\tkzDefPoints{3 /4 /I, 3 /2 /P, 8 /2 /La, 8 /3 /Lb}
\tkzDrawCircle(I,P)
\foreach \i in {1,...,3}{\% 
\coordinate (Lb) at (8,\i);
\tkzDrawLine(La,Lb)
\tkzTestInterLC(La,Lb)(I,P)
\iftkzFlagLC\tkzInterLC(La,Lb)(I,P)\tkzGetPoints{a}{b}\tkzDrawPoints(a,b)\fi
}
\end{tikzpicture}

18.2.2. Line-circle intersection

In the following example, the drawing of the circle uses two points and the intersection of the straight line and the circle uses two pairs of points. We will compare the angles $\hat{D}, E, O$ and $\hat{E}, D, O$. These angles are in opposite directions. \texttt{tkzFirstPoint} is assigned to the point that forms the angle with the smallest measure (counterclockwise direction). The counterclockwise angle $\hat{D}, E, O$ has a measure equal to $360^\circ$ minus the measure of $\hat{O}, E, D$.

\begin{tikzpicture}[scale=.75]
\tkzInit[xmax=5,ymax=4]
\tkzDefPoint(1,1){O}
\tkzDefPoint(-2,4){La}
\tkzDefPoint(5,0){Lb}
\tkzDefPoint(3,3){C}
\tkzInterLC(La,Lb)(O,C) \tkzGetPoints{D}{E}
\tkzMarkAngle[->,size=1.5](E,D,O)
\tkzDrawPolygons[new](O,D,E)
\tkzMarkAngle[->,size=1.5](D,E,O)
\tkzDrawCircle(O,C)
\tkzDrawPoints[teal](O,La,Lb,C)
\tkzDrawPoints[red](D,E)
\tkzDrawLine(La,Lb)
\tkzLabelPoints[above right](O,La,Lb,C,D,E)
\end{tikzpicture}

18.2.3. Line passing through the center option common

This case is special. You cannot compare the angles. In this case, the option near must be used. \texttt{tkzFirstPoint} is assigned to the point closest to the first point given for the line. Here we want A to be closest to Lb.

\begin{tikzpicture}[scale=1]
\tkzDefPoints{3 /4 /I, 3 /2 /P, 8 /2 /La, 8 /3 /Lb}
\tkzDrawCircle(I,P)
\foreach \i in {1,...,3}{\% 
\coordinate (Lb) at (8,\i);
\tkzDrawLine(La,Lb)
\tkzTestInterLC(La,Lb)(I,P)
\iftkzFlagLC\tkzInterLC(La,Lb)(I,P)\tkzGetPoints{a}{b}\tkzDrawPoints(a,b)\fi
}
\end{tikzpicture}
18. Intersections

18.2.4. Line-circle intersection with option common

A special case that we often meet, a point of the line is on the circle and we are looking for the other common point.

18.2.5. Line-circle intersection order of points

The idea is to compare the angles formed with the first defining point of the line, a resultant point and the center of the circle. The first point is the one that corresponds to the smallest angle. As you can see $\angle BCO < \angle BEO$. To tell the truth, $\angle BEO$ is counterclockwise.
18.2.6. Example with \foreach

\begin{tikzpicture}[scale=3,rotate=180]
\tkzDefPoint(0,1){J}
\tkzDefPoint(0,0){O}
\foreach \i in {0,-5,-10,...,-90}{
  \tkzDefPoint({2.5*cos(\i*pi/180)},{1+2.5*sin(\i*pi/180)}){P}
  \tkzInterLC[R](P,J)(O,1)\tkzGetPoints{N}{M}
  \tkzDrawSegment[color=orange](J,N)
  \tkzDrawPoints[red](N)
}\foreach \i in {-90,-95,...,-175,-180}{
  \tkzDefPoint({2.5*cos(\i*pi/180)},{1+2.5*sin(\i*pi/180)}){P}
  \tkzInterLC[R](P,J)(O,1)\tkzGetPoints{N}{M}
  \tkzDrawSegment[color=orange](J,M)
  \tkzDrawPoints[red](M)
}\end{tikzpicture}

18.2.7. Line-circle intersection with option near

D is the point closest to b.
18. More complex example of a line-circle intersection

Figure from [http://gogeometry.com/problem/p190_tangent_circle](http://gogeometry.com/problem/p190_tangent_circle)

18.2.9. Circle defined by a center and a measure, and special cases

Let's look at some special cases like straight lines tangent to the circle.
18.2.10. Calculation of radius

With \texttt{pgfmath} and \texttt{\pgfmathsetmacro}

The radius measurement may be the result of a calculation that is not done within the intersection macro, but before. A length can be calculated in several ways. It is possible of course, to use the module \texttt{pgfmath} and the macro \texttt{\pgfmathsetmacro}. In some cases, the results obtained are not precise enough, so the following calculation \(0.0002 \div 0.0001\) gives 1.98 with pgfmath while xfp will give 2.

With \texttt{xfp} and \texttt{\fpeval}:

\begin{Verbatim}
\begin{tikzpicture}
\tkzDefPoint(2,2){A}
\tkzDefPoint(5,4){B}
\tkzDefPoint(4,4){O}
\pgfmathsetmacro\tkzLen{\fpeval{0.0002/0.0001}}% or \edef\tkzLen{\fpeval{0.0002/0.0001}}
\tkzInterLC[R](A,B){\tkzLen}
\tkzGetPoints{I}{J}
\tkzDrawCircle(O,I)
\tkzDrawPoints[color=blue](A,B)
\tkzDrawPoints[color=red](I,J)
\tkzDrawLine(I,J)
\end{tikzpicture}
\end{Verbatim}
18. Intersections

18.2.11. Option "with nodes"

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,4/0/B,1/1/D,2/0/E}
\tkzDefTriangle[equilateral](A,B)
\tkzGetPoint{C}
\tkzInterLC[with nodes](D,E)(C,A,B)
\tkzGetPoints{F}{G}
\tkzDrawCircle(C,A)
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,...,G)
\tkzDrawLine(F,G)
\end{tikzpicture}

18.3. Intersection of two circles \texttt{\tkzInterCC}

The most frequent case is that of two circles defined by their center and a point, but as before the option \texttt{R} allows to use the radius measurements.

\begin{verbatim}
\tkzInterCC[⟨options⟩](⟨O, A⟩)(⟨O', A'⟩) or ⟨(O, r)(⟨O', r'⟩) or ⟨(O,A,B) (⟨O',C,D⟩)
\end{verbatim}

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>OA and O'A' are radii, O and O' are the centers.</td>
</tr>
<tr>
<td>R</td>
<td>N</td>
<td>r and r' are dimensions and measure the radii.</td>
</tr>
<tr>
<td>with nodes</td>
<td>N</td>
<td>in (A,A,C)(C,B,F) AC and BF give the radii.</td>
</tr>
<tr>
<td>common=pt</td>
<td></td>
<td>pt is common point; \texttt{tkzFirstPoint} gives the other point.</td>
</tr>
</tbody>
</table>

This macro defines the intersection point(s) I and J of the two center circles $O$ and $O'$. If the two circles do not have a common point then the macro ends with an error that is not handled. If the centers are $O$ and $O'$ and the intersections are $A$ and $B$ then the angles $O,A,O'$ and $O,B,O'$ are in opposite directions. \texttt{tkzFirstPoint} is assigned to the point that forms the "clockwise" angle.

\begin{verbatim}
\tkzTestInterCC(⟨O, A⟩)(⟨O', B⟩)
\end{verbatim}

So the arguments are two couples which define two circles with a center and a point on the circle. If there is a non empty intersection between these two circles then the test \texttt{\iftkzFlagCC} gives true.
18.3.1. test circle-circle intersection

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0 /0 /A, 2 /0 /B, 4 /0 /I, 1 /0 /P}
\tkzDrawCircle(A,B)
\foreach \i in {1,...,3}{
\coordinate (P) at (\i,0);
\tkzDrawCircle\new(I,P)
\tkzTestInterCC(A,B)(I,P)
\iftkzFlagCC\tkzInterCC(A,B)(I,P) \tkzGetPoints{a}{b}\fi}
\end{tikzpicture}

18.3.2. circle-circle intersection with common point.

\begin{tikzpicture}[scale=.5]
\tkzDefPoints{0/0/O,5/-1/A,2/2/B}
\tkzDrawPoints(O,A,B)
\tkzDrawCircles(O,B A,B)
\tkzInterCC\new[common=B](O,B)(A,B)\tkzGetFirstPoint{C}
\tkzDrawPoint(C)
\tkzLabelPoints[above](O,A,B,C)
\end{tikzpicture}

18.3.3. circle-circle intersection order of points.

The idea is to compare the angles formed with the first center, a resultant point and the center of the second circle. The first point is the one that corresponds to the smallest angle.

As you can see $\angle ODB < \angle OBE$

\begin{tikzpicture}[scale=.5]
\pgfkeys{/pgf/number format/.cd,fixed relative,precision=4}
\tkzDefPoints{0/0/O,5/-1/A,2/2/B,2/-1/C}
\tkzDrawPoints(O,A,B)
\tkzDrawCircles(O,A B,C)
\tkzInterCC(O,A)(B,C)\tkzGetPoints{D}{E}
\tkzDrawPoints(C,D,E)
\tkzLabelPoints(O,A,B,C)
\tkzLabelPoints[above](D,E)
\tkzDrawSegments[cyan](D,O D,B)
\tkzMarkAngle[red,->,size=1.5](O,D,B) \tkzGetAngle{an}
\tkzFindAngle(O,D,B) \tkzGetAngle{an}
\tkzLabelAngle(O,D,B){$\pgfmathprintnumber{\an}$}
\tkzDrawSegments[cyan](E,O E,B)
\tkzMarkAngle[red,->,size=1.5](O,E,B) \tkzGetAngle{an}
\tkzFindAngle(O,E,B) \tkzGetAngle{an}
\tkzLabelAngle(O,E,B){$\pgfmathprintnumber{\an}$}
\end{tikzpicture}
18.3.4. Construction of an equilateral triangle.

A, C, B is a clockwise angle

\begin{tikzpicture}[trim left=-1cm,scale=.5]
\tkzDefPoint(1,1){A}
\tkzDefPoint(5,1){B}
\tkzInterCC(A,B)(B,A)\tkzGetPoints{C}{D}
\tkzDrawPoint[color=black](C)
\tkzDrawCircles(A,B B,A)
\tkzCompass[color=red](A,C)
\tkzCompass[color=red](B,C)
\tkzDrawPolygon(A,B,C)
\tkzMarkSegments[mark=s\mid](A,C B,C)
\tkzLabelPoints[](A,B)
\tkzLabelPoint[above](C){$C$}
\end{tikzpicture}

18.3.5. Segment trisection

The idea here is to divide a segment with a ruler and a compass into three segments of equal length.

\begin{tikzpicture}[scale=.6]
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,2){B}
\tkzInterCC(A,B)(B,A) \tkzGetSecondPoint{D}
\tkzInterCC(D,B)(B,A) \tkzGetPoints{A}{C}
\tkzInterCC(D,B)(A,B) \tkzGetPoints{E}{B}
\tkzInterLC[common=D](C,D)(E,D) \tkzGetFirstPoint{F}
\tkzInterLL(A,F)(B,C) \tkzGetPoint{O}
\tkzInterLL(O,D)(A,B) \tkzGetPoint{H}
\tkzInterLL(O,E)(A,B) \tkzGetPoint{G}
\tkzDrawCircles(D,E A,B B,A E,A)
\tkzDrawSegments[](O,F O,B O,D O,E)
\tkzDrawPoints(A,...,H)
\tkzDrawSegments(A,B B,D A,D A,E,E,F C,F B,C)
\tkzMarkSegments[mark=s\mid](A,G G,H H,B)
\end{tikzpicture}
18.3.6. With the option "with nodes"

\begin{tikzpicture}[scale=.5]
\tkzDefPoints{0/0/A,0/5/B,5/0/C}
\tkzDefPoint(54:5){F}
\tkzInterCC[with nodes](A,A,C)(C,B,F)
\tkzGetPoints{a}{e}
\tkzInterCC(A,C)(a,e) \tkzGetFirstPoint{b}
\tkzInterCC(A,C)(b,a) \tkzGetFirstPoint{c}
\tkzInterCC(A,C)(c,b) \tkzGetFirstPoint{d}
\tkzDrawCircle[new](A,C)
\tkzDrawPoints(a,b,c,d,e)
\tkzDrawPolygon(a,b,c,d,e)
\foreach \vertex/\num in {a/36,b/108,c/180,d/252,e/324}{%
\tkzDrawPoint(\vertex)
\tkzLabelPoint[below right](\vertex){\num}
\tkzDrawSegment(A,\vertex)}
\end{tikzpicture}

18.3.7. Mix of intersections

\begin{tikzpicture}[scale = .75]
\tkzDefPoint(2,2){A}
\tkzDefPoint(0,0){B}
\tkzDefPoint(-2,2){C}
\tkzDefPoint(0,4){D}
\tkzDefPoint(4,2){E}
\tkzCircumCenter(A,B,C)\tkzGetPoint{O}
\tkzInterCC[R](O,2)(D,2) \tkzGetPoints{M1}{M2}
\tkzInterCC(O,A)(D,O) \tkzGetPoints{1}{2}
\tkzInterLC(A,E)(B,M1) \tkzGetSecondPoint{M3}
\tkzInterLC(O,C)(M3,D) \tkzGetSecondPoint{L}
\tkzDrawSegments(C,L)
\tkzDrawPoints(A,B,C,D,E,M1,M2,M3,O,L)
\tkzDrawSegments(0,E)
\tkzDrawSegments[new](C,A,D,B)
\tkzDrawPoint(0)
\tkzDrawCircles[new](M3,D B,M2 D,0)
\tkzDrawCircle(0,A)
\tkzLabelPoints[below right](A,B,C,D,E,M1,M2,M3,O,L)
\end{tikzpicture}

18.3.8. Altshiller-Court's theorem

The two lines joining the points of intersection of two orthogonal circles to a point on one of the circles met the other circle in two diametrally opposite points. Altshiller p 176
19. Angles

19.1. Definition and usage with tkz-euclide

In Euclidean geometry, an angle is the figure formed by two rays, called the sides of the angle, sharing a common endpoint, called the vertex of the angle. A ray with \texttt{tkz-euclide} is defined by two points also each angle is defined with three points like $\angle AOB$. The vertex $O$ is the second point. Their order is important because it is assumed that the angle is specified in the direct order (counterclockwise). In trigonometry and mathematics in general, plane angles are conventionally measured counterclockwise, starting with $0^\circ$ pointing directly to the right (or east), and $90^\circ$ pointing straight up (or north). Let us agree that an angle measured counterclockwise is positive.
Angles are involved in several macros like \texttt{\tkzDefPoint}, \texttt{\tkzDefPointBy[rotation = ...]}, \texttt{\tkzDrawArc} and the next one \texttt{\tkzGetAngle}. With the exception of the last one, all these macros accept negative angles.

\[
\begin{array}{c}
\text{Rotation 80° from (O, A) to (O, B)} \\
\text{\texttt{\tkzDefPointBy[rotation=center O angle 80]}} \\
\text{\texttt{\tkzFindAngle(A,O,B) gives 80}} \\
\end{array}
\begin{array}{c}
\text{Rotation \textendash 80° from (O, A) to (O, B)} \\
\text{\texttt{\tkzDefPointBy[rotation=center O angle -80]}} \\
\text{\texttt{\tkzFindAngle(A,O,B) gives 280}} \\
\end{array}
\]

As we can see, the \textendash 80° rotation defines a clockwise angle but the macro \texttt{\tkzFindAngle} recovers a counterclockwise angle.

19.2. Recovering an angle \texttt{\tkzGetAngle}

\texttt{\tkzGetAngle(name of macro)}

Assigns the value in degree of an angle to a macro. The value is positive and between 0° and 360°. This macro retrieves \texttt{\tkzAngleResult} and stores the result in a new macro.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>name of macro</td>
<td>\tkzGetAngle{ang}</td>
<td>\texttt{\ang} contains the value of the angle.</td>
</tr>
</tbody>
</table>

This is an auxiliary macro that allows you to retrieve the result of the following macro \texttt{\tkzFindAngle}.

19.3. Angle formed by three points

\texttt{\tkzFindAngle(pt1,pt2,pt3)}

The result is stored in a macro \texttt{\tkzAngleResult}.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2,pt3)</td>
<td>\tkzFindAngle(A,B,C)</td>
<td>\texttt{\tkzAngleResult} gives the angle (\overrightarrow{BA}, \overrightarrow{BC})</td>
</tr>
</tbody>
</table>

\texttt{tkz-euclide}  
\texttt{AlterMundus}
The measure is always positive and between $0^\circ$ and $360^\circ$. With the usual conventions, a counterclockwise angle smaller than a straight angle has always a measure between $0^\circ$ and $180^\circ$, while a clockwise angle smaller than a straight angle will have a measurement greater than $180^\circ$. \texttt{\tkzGetAngle} can retrieve the angle.

19.3.1. Verification of angle measurement

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(-1,1){A}
\tkzDefPoint(5,2){B}
\tkzDefEquilateral(A,B)
\tkzGetPoint{C}
\tkzDrawPolygon(A,B,C)
\tkzFindAngle(B,A,C) \tkzGetAngle{angleBAC}
edef\angleBAC{\fpeval{round(\angleBAC)}}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B)
\tkzLabelPoint[right](C){$C$}
\tkzLabelAngle(B,A,C){\angleBAC$^\circ$}
\tkzMarkAngle[size=1.5](B,A,C)
\end{tikzpicture}

19.3.2. Determination of the three angles of a triangle

\begin{tikzpicture}
\tikzset{label angle style/.append style={pos=1.4}}
\tkzDefPoints{0/0/a,5/3/b,3/6/c}
\tkzDrawPolygon(a,b,c)
\tkzFindAngle(c,b,a) \tkzGetAngle{angleCBA}
\pgfmathparse{round(1+\angleCBA)}
\let\angleCBA\pgfmathresult
\tkzFindAngle(a,c,b) \tkzGetAngle{angleACB}
\pgfmathparse{round(\angleACB)}
\let\angleACB\pgfmathresult
\tkzFindAngle(b,a,c) \tkzGetAngle{angleBAC}
\pgfmathparse{round(\angleBAC)}
\let\angleBAC\pgfmathresult
\tkzMarkAngle(c,b,a)
\tkzLabelAngle(c,b,a){\tiny $\angleCBA^\circ$}
\tkzMarkAngle(a,c,b)
\tkzLabelAngle(a,c,b){\tiny $\angleACB^\circ$}
\tkzMarkAngle(b,a,c)
\tkzLabelAngle(b,a,c){\tiny $\angleBAC^\circ$}
\end{tikzpicture}

19.3.3. Angle between two circles

We are looking for the angle formed by the tangents at a point of intersection.
19.4. Angle formed by a straight line with the horizontal axis \tkzFindSlopeAngle

Much more interesting than the last one. The result is between -180 degrees and +180 degrees.

\begin{equation}
\tkzFindSlopeAngle((A,B))
\end{equation}

Determines the slope of the straight line (AB). The result is stored in a macro \tkzAngleResult.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)</td>
<td>\tkzFindSlopeAngle(A,B)</td>
<td></td>
</tr>
</tbody>
</table>

\tkzGetAngle can retrieve the result. If retrieval is not necessary, you can use \tkzAngleResult.

19.4.1. How to use \tkzFindSlopeAngle

The point here is that (AB) is the bisector of \( \overline{CAD} \), such that the AD slope is zero. We recover the slope of (AB) and then rotate twice.

\begin{equation}
\begin{tikzpicture}
\tkzDefPoint(1,5){A} \tkzDefPoint(5,2){B}
\tkzFindSlopeAngle(A,B)\tkzGetAngle{tkzang}
\tkzDefPointBy[rotation= center A angle \tkzang ](B)
\tkzGetPoint(C)
\tkzDefPointBy[rotation= center A angle -\tkzang ](B)
\tkzGetPoint(D)
\tkzDrawSegment(A,B)
\tkzDrawSegments[new](A,C A,D)
\tkzDrawPoints(A,B,C,D)
\tkzCompass[length=1](A,C)
\tkzCompass[delta=10,brown](B,C)
\tkzLabelPoints(B,C,D)
\tkzLabelPoints[above left](A)
\end{tikzpicture}
\end{equation}

19.4.2. Use of \tkzFindSlopeAngle and \tkzGetAngle

Here is another version of the construction of a mediator
20. Random point definition

At the moment there are four possibilities:

1. point in a rectangle;
2. on a segment;
3. on a straight line;
4. on a circle.

20.1. Obtaining random points

This is the new version that replaces \tkzGetRandPointOn.
The result is a point with a random position that can be named with the macro `\tkzGetPoint`. It is possible to use `\tkzPointResult` if it is not necessary to retain the results.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>rectangle=pt1 and pt2</td>
<td></td>
<td>[rectangle=A and B]</td>
</tr>
<tr>
<td>segment=pt1--pt2</td>
<td></td>
<td>[segment=A--B]</td>
</tr>
<tr>
<td>line=pt1--pt2</td>
<td></td>
<td>[line=A--B]</td>
</tr>
<tr>
<td>circle =center pt1 radius dim</td>
<td></td>
<td>[circle = center A radius 2]</td>
</tr>
<tr>
<td>circle through=center pt1 through pt2</td>
<td></td>
<td>[circle through= center A through B]</td>
</tr>
<tr>
<td>disk through=center pt1 through pt2</td>
<td></td>
<td>[disk through= center A through B]</td>
</tr>
</tbody>
</table>

### 20.1.1. Random point in a rectangle

```latex
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/3/C}
  \tkzDefRandPointOn[rectangle = A and C]
  \tkzGetPoint{E}
  \tkzDefRectangle(A,C) \tkzGetPoints{B}{D}
  \tkzDrawPolygon[red](A,...,D)
  \tkzDrawPoints(A,...,E)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above](C,D,E)
\end{tikzpicture}
```

### 20.1.2. Random point on a segment or a line

```latex
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/2/C}
  \tkzDefRandPointOn[segment = A--C]
  \tkzGetPoint{B}
  \tkzDrawLine(A,C)
  \tkzDrawPoints(A,C) \tkzDrawPoint[red](B)
  \tkzLabelPoints(A,C) \tkzLabelPoints[red](B)
\end{tikzpicture}
```

### 20.1.3. Random point on a circle or a disk

```latex
\begin{tikzpicture}
  \tkzDefPoints{3/2/A,1/1/B}
  \tkzCalcLength(A,B) \tkzGetLength{rAB}
  \tkzDefRandPointOn[circle = center A radius \rAB]
  \tkzGetPoint{C}
  \tkzDefRandPointOn[circle through= center A through B]
  \tkzGetPoint{D}
  \tkzDefRandPointOn[disk through= center A through B]
  \tkzGetPoint{E}
  \tkzDrawCircle(A,B)
  \tkzDrawPoints(A,B)
  \tkzDrawPoints[red](C,D,E)
  \tkzLabelPoints[red,right](C,D,E)
\end{tikzpicture}
```
Part IV.

Drawing and Filling
21. Drawing

\texttt{tkz-euclide} can draw 5 types of objects: point, line or line segment, circle, arc and sector.

21.1. Draw a point or some points

There are two possibilities: \texttt{tkzDrawPoint} for a single point or \texttt{tkzDrawPoints} for one or more points.

21.1.1. Drawing points \texttt{tkzDrawPoint}

\begin{verbatim}
\tkzDrawPoint[[\langle local options \rangle]](\langle name \rangle)
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name of point</td>
<td>no</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Only one point name is accepted</td>
</tr>
</tbody>
</table>

The argument is required. The disc takes the color of the circle, but lighter. It is possible to change everything. The point is a node and therefore it is invariant if the drawing is modified by scaling.

\begin{verbatim}
<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TikZ options</td>
<td>all</td>
<td>all TikZ options are valid.</td>
</tr>
<tr>
<td>shape</td>
<td>circle</td>
<td>Possible \texttt{cross} or \texttt{cross out}</td>
</tr>
<tr>
<td>size</td>
<td>6\times \pgflinewidth</td>
<td></td>
</tr>
<tr>
<td>color</td>
<td>black</td>
<td>the default color can be changed</td>
</tr>
</tbody>
</table>
\end{verbatim}

We can create other forms such as \texttt{cross}

By default, point style is defined like this:

\begin{verbatim}
\tikzset{point style/.style = {%
  draw = black,
  inner sep = 0pt,
  shape = circle,
  minimum size = 3 pt,
  fill = black
}}
\end{verbatim}

21.1.2. Example of point drawings

Note that scale does not affect the shape of the dots. Which is normal. Most of the time, we are satisfied with a single point shape that we can define from the beginning, either with a macro or by modifying a configuration file.

\begin{verbatim}
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(1,3){A}
  \tkzDefPoint(4,1){B}
  \tkzDefPoint(0,0){O}
  \tkzDrawPoint[\textcolor{red}]{A}
  \tkzDrawPoint[\textcolor{blue!20},\textcolor{blue}]{B}
  \tkzDrawPoint[\textcolor{teal}\texttt{cross},\textcolor{teal}\texttt{size}=8pt]{O}
\end{tikzpicture}
\end{verbatim}

It is possible to draw several points at once but this macro is a little slower than the previous one. Moreover, we have to make do with the same options for all the points.
22. Drawing the lines

The following macros are simply used to draw, name lines.

22.1. Draw a straight line

To draw a normal straight line, just give a couple of points. You can use the add option to extend the line (This option is due to Mark Wibrow, see the code below).

The style of a line is by default:

\begin{verbatim}
\tikzset{line style/.style = {%
  line width = .6pt, 
  color = black, 
  style = solid, 
  add = {.2} and {.2} 
}}
\end{verbatim}

with

\begin{verbatim}
\tikzset{%
  add/.style args={#1 and #2}{
    to path=\%
    ($!-#1!(\tikztostart)!#2!(\tikztotarget)!-#1!(\tikztotarget)!#2!(\tikztostart)$)%
  \tikztonodes})
\end{verbatim}

You can modify this style with \texttt{\textbackslash \texttt{tkzSetUpLine}} see 39.1
23. Drawing a segment

\begin{tikzpicture}
\tkzInit[xmin=-2,xmax=3,ymin=-2.25,ymax=2.25]
\tkzClip[space=.25]
\tkzDefPoint(0,0){A} \tkzDefPoint(2,0.5){B}
\tkzDefPoint(0,-1){C}\tkzDefPoint(2,-0.5){D}
\tkzDefPoint(0,1){E} \tkzDefPoint(2,1.5){F}
\tkzDefPoint(0,-2){G} \tkzDefPoint(2,-1.5){H}
\tkzDrawLine(A,B) \tkzDrawLine[add = 0 and .5](C,D)
\tkzDrawLine[add = 1 and 0](E,F)
\tkzDrawLine[add = 0 and 0](G,H)
\tkzDrawPoints(A,B,C,D,E,F,G,H)
\tkzLabelPoints(A,B,C,D,E,F,G,H)
\end{tikzpicture}

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(2,0){B}
\tkzDefPoint(1,2){C}
\tkzDefPoint(3,2){D}
\tkzDrawLines(A,B C,D A,C B,D)
\tkzLabelPoints(A,B,C,D)
\end{tikzpicture}

23. Drawing a segment

There is, of course, a macro to simply draw a segment.

23.1. Draw a segment \tkzDrawSegment

\begin{tikzpicture}
\tkzInit[xmin=-2,xmax=3,ymin=-2.25,ymax=2.25]
\tkzClip[space=.25]
\tkzDefPoint(0,0){A} \tkzDefPoint(2,0.5){B}
\tkzDefPoint(0,-1){C}\tkzDefPoint(2,-0.5){D}
\tkzDefPoint(0,1){E} \tkzDefPoint(2,1.5){F}
\tkzDefPoint(0,-2){G} \tkzDefPoint(2,-1.5){H}
\tkzDrawLine(A,B) \tkzDrawLine[add = 0 and .5](C,D)
\tkzDrawLine[add = 1 and 0](E,F)
\tkzDrawLine[add = 0 and 0](G,H)
\tkzDrawPoints(A,B,C,D,E,F,G,H)
\tkzLabelPoints(A,B,C,D,E,F,G,H)
\end{tikzpicture}
23. Drawing a segment

The arguments are a list of two points. The styles of TikZ are available for the drawings.

**Argument example definition**

\[
\text{\texttt{\textbackslash tkzDrawSegment\[\langle local\ options\ \rangle\]\((\langle pt1, pt2 \rangle)\)}}
\]

The arguments are a list of two points. The styles of TikZ are available for the drawings.

**Table: argument example definition**

<table>
<thead>
<tr>
<th>Options</th>
<th>Example</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TikZ options</td>
<td>all TikZ options are valid.</td>
<td></td>
</tr>
<tr>
<td>dim</td>
<td>no default</td>
<td>\text{dim} = {\text{label}, \text{dim}, \text{option}, \ldots}</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>allows you to add dimensions to a figure.</td>
</tr>
</tbody>
</table>

This is of course equivalent to \texttt{\textbackslash draw \(A\)--(B)}; You can also use the option \texttt{add}.

23.1.1. Example with point references

```latex
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,1){B}
  \tkzDrawSegment[color=red,thin](A,B)
  \tkzDrawPoints(A,B)
  \tkzLabelPoints(A,B)
\end{tikzpicture}
```

23.1.2. Example of extending a segment with option \texttt{add}

```latex
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
  \tkzDefTriangleCenter[euler](A,B,C)
  \tkzGetPoint{E}
  \tkzDefCircle[euler](A,B,C)\tkzGetPoints{E}{e}
  \tkzDrawCircle[red](E,e)
  \tkzDrawLines[add=.5 and .5](A,B A,C B,C)
  \tkzDrawPoints(A,B,C,E)
  \tkzLabelPoints(A,B,C,E)
\end{tikzpicture}
```

23.1.3. Adding dimensions with option \texttt{dim} new code from Muzimuzhi Z

This code comes from an answer to this question on tex.stackexchange.com (change-color-and-style-of-dimension-lines-in-tkz-euclide). The code of \texttt{dim} is based on options of TikZ, you must add the units. You can use now two styles: \texttt{dim style} and \texttt{dim fence style}. You have several ways to use them. I'll let you look at the examples to see what you can do with these styles.
23. Drawing a segment

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/3/A, 1/-3/B}
\tkzDrawPoints(A,B)
\tkzDrawSegment[dim={\pgfmathprintnumber\ABl,6pt,text=red}](A,B)
\tkzDrawSegment[dim={\pgfmathprintnumber\BCl,6pt}](B,C)
\tkzDrawSegment[dim={\pgfmathprintnumber\ACl,-6pt}](A,C)
\end{tikzpicture}

23.1.4. Adding dimensions with option \texttt{dim partI}

\begin{tikzpicture}[scale=2]
\pgfkeys{/pgf/number format/.cd,fixed,precision=2}
\tkzDefPoint(0,0){A}
\tkzDefPoint(3.07,0){B}
\tkzInterCC[R](A,2.37)(B,1.82)
\tkzGetPoints{C}{C'}
\tkzDefCircle[in](A,B,C) \tkzGetPoints{G}{g}
\tkzDrawCircle(G,g)
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzCalcLength(A,B)\tkzGetLength{ABl}
\tkzCalcLength(B,C)\tkzGetLength{BCl}
\tkzCalcLength(A,C)\tkzGetLength{ACl}
\begin{scope}[dim style/.style={dashed,sloped,teal}]
\tkzDrawSegment[dim={\pgfmathprintnumber\BCl,6pt,text=red}](C,B)
\tkzDrawSegment[dim={\pgfmathprintnumber\ACl,6pt}](A,C)
\tkzDrawSegment[dim={\pgfmathprintnumber\ABl,-6pt}](A,B)
\end{scope}
\tkzLabelPoints(A,B) \tkzLabelPoints[above](C)
\end{tikzpicture}
23.1.5. Adding dimensions with option dim part II

\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/O,-2/0/A,2/0/B,-2/4/C,2/4/D,2/-4/E,-2/-4/F}
  \tkzDrawPolygon(C,...,F)
  \tkzDrawSegments(A,B)
  \tkzDrawPoints(A,...,F,O)
  \tkzLabelPoints[below left](A,...,F,O)
  \tkzDrawSegment[dim={ $\sqrt{5}$,2cm,}](C,E)
  \tkzDrawSegment[dim={ $\frac{\sqrt{5}}{2}$,1cm,}](O,E)
  \tkzDrawSegment[dim={ $2$,2cm,left=8pt}](F,C)
  \tkzDrawSegment[dim={ $1$,1cm,left=8pt}](F,A)
\end{tikzpicture}

23.2. Drawing segments \texttt{\tkzDrawSegments}

If the options are the same we can plot several segments with the same macro.

\begin{Verbatim}
\texttt{\tkzDrawSegments[(local options)]((pt1,pt2 pt3,pt4 ...))}
\end{Verbatim}

The arguments are a two-point couple list. The styles of \texttt{TikZ} are available for the plots.

\begin{tikzpicture}
  \tkzInit[xmin=-1,xmax=3,ymin=-1,ymax=2]
  \tkzClip[space=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,1){B}
  \tkzDefPoint(3,0){C}
  \tkzDrawSegments(A,B B,C)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,C)
  \tkzLabelPoints[above](B)
\end{tikzpicture}

23.2.1. Place an arrow on segment

\begin{tikzpicture}
  \tkzSetUpStyle[postaction=decorate, decoration={markings, mark=at position .5 with {\arrow[thick]}}]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,-4){B}
  \tkzDrawSegments(A,B)
  \tkzDrawPoints(A,B)
\end{tikzpicture}
23. Drawing a segment

23.3. Drawing line segment of a triangle

23.3.1. How to draw Altitude

\begin{tikzpicture}[rotate=-90]
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
\tkzDrawLine[dashed,color=magenta](C,Hc)
\tkzDrawSegment[green!60!black](A,C)
\tkzDrawSegment[green!60!black](C,B)
\tkzDrawSegment[green!60!black](B,A)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[left](Hc){$Hc$}
\tkzLabelSegment[auto](B,A){$c$}
\tkzLabelSegment[auto,swap](B,C){$a$}
\tkzLabelSegment[auto,swap](C,A){$b$}
\tkzMarkAngle[size=1,color=cyan,mark=\|](C,B,A)
\tkzMarkAngle[size=1,color=cyan,mark=\|](A,C,Hc)
\tkzMarkAngle[size=0.75,color=orange,mark=\|=](Hc,C,B)
\tkzMarkAngle[size=0.75,color=orange,mark=\|=](B,A,C)
\tkzMarkRightAngle(A,C,B)
\tkzMarkRightAngle(B,Hc,C)
\end{tikzpicture}

23.4. Drawing a polygon

\texttt{\tkzDrawPolygon[(local options)]((points list))}

Just give a list of points and the macro plots the polygon using the TikZ options present. You can replace \((A,B,C,D,E)\) by \((A,...,E)\) and \((P_1,P_2,P_3,P_4,P_5)\) by \((P_1,...,P_5)\).

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>((p1,p2,p3,...))</td>
<td>\tkzDrawPolygon<a href="A,B,C">gray,dashed</a></td>
<td>Drawing a triangle</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options TikZ</td>
<td>...</td>
<td>\tkzDrawPolygon<a href="A,B,C">red,line width=2pt</a></td>
</tr>
</tbody>
</table>

23.4.1. \texttt{\tkzDrawPolygon}

\begin{tikzpicture} [rotate=18,scale=1]
\tkzDefPoints{0/0/A,2.25/0.2/B,2.5/2.75/C,-0.75/2/D}
\tkzDrawPolygon(A,B,C,D)
\tkzDrawSegments[style=dashed](A,C B,D)
\end{tikzpicture}
23.4.2. Option two angles

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(6,0){B}
\tkzDefTriangle[two angles = 50 and 70](A,B) \tkzGetPoint{C}
\tkzDrawPolygon(A,B,C)
\tkzLabelAngle[pos=1.4](B,A,C){$50^\circ$}
\tkzLabelAngle[pos=0.8](C,B,A){$70^\circ$}
\end{tikzpicture}

23.4.3. Style of line

\begin{tikzpicture}[scale=.6]
\tkzSetUpLine[line width=5mm,color=teal]
\tkzDefPoint(0,0){O}
\foreach \i in {0,...,5}{
  \tkzDefPoint({30+60*\i}:4){p\i}
}\tkzDefMidPoint(p1,p3) \tkzGetPoint{m1}
\tkzDefMidPoint(p3,p5) \tkzGetPoint{m3}
\tkzDefMidPoint(p5,p1) \tkzGetPoint{m5}
\tkzDrawPolygon[line join=round](p1,p3,p5)
\tkzDrawPolygon[teal!80, line join=round](p0,p2,p4)
\tkzDrawSegments(m1,p3 m3,p5 m5,p1)
\tkzDefCircle[R](O,4.8)\tkzGetPoint{o}
\tkzDrawCircle[teal](O,o)
\end{tikzpicture}

23.5. Drawing a polygonal chain

\begin{tikzpicture}
\tkzDrawPolySeg[(local options)]((points list))
\end{tikzpicture}

Just give a list of points and the macro plots the polygonal chain using the \texttt{TikZ} options present.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2,pt3,...)</td>
<td>\tkzDrawPolySeg<a href="A,B,C">gray, dashed</a></td>
<td>Drawing a triangle</td>
</tr>
<tr>
<td>options</td>
<td>default</td>
<td>example</td>
</tr>
<tr>
<td>Options TikZ</td>
<td>...</td>
<td>\tkzDrawPolySeg<a href="A,B,C">red, line width=2pt</a></td>
</tr>
</tbody>
</table>

\texttt{tkz-euclide} AlterMundus
23.5.1. Polygonal chain

\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,3/4/C,2/2/D}
\tkzDrawPolySeg(A,...,D)
\tkzDrawPoints(A,...,D)
\end{tikzpicture}

23.5.2. The idea is to inscribe two squares in a semi-circle.

A Sangaku look! It is a question of proving that one can inscribe in a half-disc, two squares, and to determine the length of their respective sides according to the radius.

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,8/0/B,4/0/I}
\tkzDefSquare(A,B) \tkzGetPoints{C}{D}
\tkzInterLC(I,C)(I,B) \tkzGetPoints{E'}{E}
\tkzInterLC(I,D)(I,B) \tkzGetPoints{F'}{F}
\tkzDefPointsBy[projection=onto A--B](E,F){H,G}
\tkzDefPointsBy[symmetry = center H](I){J}
\tkzDefSquare(H,J) \tkzGetPoints{K}{L}
\tkzDrawSector(I,B)(A)
\tkzDrawPolySeg(H,E,F,G)
\tkzDrawPolySeg(J,K,L)
\tkzDrawPoints(E,G,H,F,J,K,L)
\end{tikzpicture}

23.5.3. Polygonal chain: index notation

\begin{tikzpicture}
\foreach \pt in {1,2,...,8} {%
\tkzDefPoint(\pt*20:3){P_\pt}}
\tkzDrawPolySeg(P_1,P_...,P_8)
\tkzDrawPoints(P_1,P_...,P_8)
\end{tikzpicture}

24. Draw a circle with \tkzDrawCircle

24.1. Draw one circle

\tkzDrawCircle[(local options)]((A,B))

Attention you need only two points to define a radius. An additional option R is available to give a measure directly.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)</td>
<td>(A,B)</td>
<td>A center through B</td>
</tr>
</tbody>
</table>

Of course, you have to add all the styles of TikZ for the tracings...
24.1.1. Circles and styles, draw a circle and color the disc

We'll see that it's possible to colour in a disc while tracing the circle.

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\tkzDefPoint(3,0){A}
% circle with center O and passing through A
\tkzDrawCircle(O,A)
% diameter circle $[OA]$ 
\tkzDefCircle[diameter](O,A) \tkzGetPoint{I}
\tkzDrawCircle[new,fill=orange!10,opacity=.5](I,A)
% circle with center O and radius = exp(1) cm
edef\rayon{0.25*exp(1)}
\tkzDefCircle[R](O,\rayon) \tkzGetPoint{o}
\tkzDrawCircle[color=orange](O,o)
\end{tikzpicture}

24.2. Drawing circles

\tkzDrawCircles[⟨local options⟩](⟨A,B C,D ...⟩)

Attention, the arguments are lists of two points. The circles that can be drawn are the same as in the previous macro. An additional option R is available to give a measure directly.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨pt1,pt2 pt3,pt4 ...⟩</td>
<td>⟨A,B C,D⟩</td>
<td>List of two points</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>through</td>
<td>through</td>
<td>circle with two points defining a radius</td>
</tr>
</tbody>
</table>

You do not need to use the default option through. Of course, you have to add all the styles of Ti\kZ for the tracings...
24.2.1. Circles defined by a triangle.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,2/0/B,3/2/C}
\tkzDrawPolygon(A,B,C)
\tkzDrawCircles(A,B B,C C,A)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B,C)
\end{tikzpicture}

24.2.2. Concentric circles.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,1/0/a,2/0/b,3/0/c}
\tkzDrawCircles(A,a A,b A,c)
\tkzDrawPoint(A)
\tkzLabelPoints(A)
\end{tikzpicture}
24.2.3. Exinscribed circles.

\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,4/0/B,1/2.5/C}
\tkzDrawPolygon(A,B,C)
\tkzDefCircle[ex](B,C,A)
\tkzGetPoint{J_c} \tkzGetSecondPoint{T_c}
\tkzDrawCircle(J_c,T_c)
\tkzDrawLines[add=0 and 1](C,A C,B)
\tkzDrawSegment(J_c,T_c)
\tkzMarkRightAngle(J_c,T_c,B)
\tkzDrawPoints(A,B,C,J_c,T_c)
\end{tikzpicture}

24.2.4. Cardioid

Based on an idea by O. Reboux made with pst-eucl (Pstricks module) by D. Rodriguez. Its name comes from the Greek *kardia* (heart), in reference to its shape, and was given to it by Johan Castillon (Wikipedia).

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

\[\texttt{\textbackslash{}tkzDrawSemiCircle[\{local\ options\}](O,A)}\]

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>((pt1,pt2)) ((O,A))</td>
<td>(OA=) radius</td>
<td>(O) center (A) extremity of the semicircle</td>
</tr>
</tbody>
</table>

24.3.1. Use of \texttt{\textbackslash{}tkzDrawSemiCircle}

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(6,0){B}
\tkzDefMidPoint(A,B) \tkzGetPoint{O}
\tkzDrawSemiCircle[blue](O,B)
\tkzDrawSemiCircle[red](O,A)
\tkzDrawPoints(O,A,B)
\tkzLabelPoints[below right](O,A,B)
\end{tikzpicture}

24.4. Drawing semicircles

\[\texttt{\textbackslash{}tkzDrawSemiCircles[\{local\ options\}](A,B,C,D \ldots)}\]

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>((pt1,pt2\ pt3,pt4\ \ldots)) ((A,B,C,D))</td>
<td></td>
<td>List of two points</td>
</tr>
</tbody>
</table>
25. Drawing arcs

25.1. Macro: \texttt{tkzDrawArc}

\begin{verbatim}
\tkzDrawArc[local options](O,...)(...)
\end{verbatim}

This macro traces the arc of center \texttt{O}. Depending on the options, the arguments differ. It is a question of determining a starting point and an end point. Either the starting point is given, which is the simplest, or the radius of the arc is given. In the latter case, it is necessary to have two angles. Either the angles can be given directly, or nodes associated with the center can be given to determine them. The angles are in degrees.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>towards</td>
<td>towards</td>
<td>O is the center and the arc from A to (OB)</td>
</tr>
<tr>
<td>rotate</td>
<td>towards</td>
<td>the arc starts from A and the angle determines its length</td>
</tr>
<tr>
<td>R</td>
<td>towards</td>
<td>We give the radius and two angles</td>
</tr>
<tr>
<td>R with nodes</td>
<td>towards</td>
<td>We give the radius and two points</td>
</tr>
<tr>
<td>angles</td>
<td>towards</td>
<td>We give the radius and two points</td>
</tr>
<tr>
<td>delta</td>
<td>0</td>
<td>angle added on each side</td>
</tr>
<tr>
<td>reverse</td>
<td>false</td>
<td>inversion of the arc's path, interesting to inverse arrow</td>
</tr>
</tbody>
</table>

Of course, you have to add all the styles of \texttt{TikZ} for the tracings...

<table>
<thead>
<tr>
<th>options</th>
<th>arguments</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>towards</td>
<td>(pt,pt)(pt)</td>
<td>\tkzDrawArc<a href="O,A">delta=10</a>(B)</td>
</tr>
<tr>
<td>rotate</td>
<td>(pt,pt)(an)</td>
<td>\tkzDrawArc<a href="O,A">rotate, color=red</a>(90)</td>
</tr>
<tr>
<td>R</td>
<td>(pt,r)(an,an)</td>
<td>\tkzDrawArc<a href="O,2">R</a>(30,90)</td>
</tr>
<tr>
<td>R with nodes</td>
<td>(pt,r)(an,an)</td>
<td>\tkzDrawArc<a href="O,2">R with nodes</a>(A,B)</td>
</tr>
<tr>
<td>angles</td>
<td>(pt,pt)(an,an)</td>
<td>\tkzDrawArc<a href="O,A">angles</a>(0,90)</td>
</tr>
</tbody>
</table>

Here are a few examples:

25.1.1. Option \texttt{towards}

It's useless to put \texttt{towards}. In this first example the arc starts from A and goes to B. The arc going from B to A is different. The salient is obtained by going in the direct direction of the trigonometric circle.
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPointBy[rotation= center O angle 90](A)
  \tkzGetPoint{B}
  \tkzDrawArc[color=orange, <->](O,A)(B)
  \tkzDrawArc(O,B)(A)
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}

25.1.2. Option towards

In this one, the arc starts from A but stops on the right (OB).

\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzDrawArc[color=blue, ->](O,A)(B)
  \tkzDrawArc[color=gray](O,B)(A)
  \tkzDrawArc(O,B)(A)
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}

25.1.3. Option rotate

\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-2){A}
  \tkzDefPoint(60:2){B}
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawArc[rotate, color=red](O,A)(180)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}

25.1.4. Option R

\begin{tikzpicture}[scale=0.75]
  \tkzDefPoints{0/0/O}
  \tkzSetUpCompass[<->]
  \tkzDrawArc[R, color=teal, double](O,3)(270,360)
  \tkzDrawArc[R, color=orange, double](O,2)(0,270)
  \tkzDrawPoint(O)
  \tkzLabelPoint[below](O){$O$}
\end{tikzpicture}
25.1.5. Option \texttt{R} with nodes

\begin{tikzpicture}[scale=0.75]
\tkzDefPoint(0,0){O}
\tkzDefPoint(2,-1){A}
\tkzDefPoint(1,1){B}
\tkzCalcLength(B,A)\tkzGetLength{radius}
\tkzDrawArc[R with nodes](B,\radius)(A,O)
\end{tikzpicture}

25.1.6. Option \texttt{delta}

This option allows a bit like \texttt{tkzCompass} to place an arc and overflow on either side. \texttt{delta} is a measure in degrees.

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,0){B}
\tkzDefPointBy[rotation= center A angle 60](B)
\tkzGetPoint{C}
\begin{scope}
% style only local
\tkzDefPointBy[symmetry= center C](A)
\tkzGetPoint{D}
\tkzDrawSegments(A,B A,D)
\tkzDrawLine(B,D)
\tkzSetUpCompass[color=orange]
\tkzDrawArc[orange,delta=10](A,B)(C)
\tkzDrawArc[orange,delta=10](B,C)(A)
\tkzDrawArc[orange,delta=10](C,D)(D)
\end{scope}
\tkzDrawPoints(A,B,C,D)
\tkzLabelPoints[below right](A,B,C,D)
\tkzMarkRightAngle(D,B,A)
\end{tikzpicture}

25.1.7. Option \texttt{angles}: example 1

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,0){B}
\tkzDefPoint(2.5,0){O}
\tkzDefPointBy[rotation=center O angle 60](B)
\tkzGetPoint{D}
\tkzDefPointBy[symmetry=center D](O)
\tkzGetPoint{E}
\begin{scope}
\tkzDrawArc[angles](O,B)(0,180)
\tkzDrawArc[angles,](B,O)(100,180)
\tkzCompass[delta=20](D,E)
\tkzDrawLines(A,B O,E B,E)
\tkzDrawPoints(A,B,O,D,E)
\end{scope}
\tkzLabelPoints[below right](A,B,O,D,E)
\tkzMarkRightAngle(O,B,E)
\end{tikzpicture}
26. Drawing a sector or sectors

25.1.8. Option angles: example 2

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\tkzDefPoint(5,0){I}
\tkzDefPoint(0,5){J}
\tkzInterCC(O,I)(I,O)\tkzGetPoints{B}{C}
\tkzInterCC(O,I)(J,O)\tkzGetPoints{D}{A}
\tkzInterCC(I,O)(J,O)\tkzGetPoints{L}{K}
\tkzDrawArc[angles](O,I)(0,90)
\tkzDrawArc[angles,color=gray,style=dashed](I,O)(90,180)
\tkzDrawArc[angles,color=gray,style=dashed](J,O)(-90,0)
\tkzDrawPoints(A,B,K)
\foreach \point in {I,A,B,J,K}{\tkzDrawSegment(O,\point)}
\end{tikzpicture}

25.1.9. Option reverse: inversion of the arrow

\begin{tikzpicture}
\tkzDefPoints{0/0/O,3/0/U}
\tkzDefPoint(10:1){A}
\tkzDefPoint(90:1){B}
\tkzLabelPoints(A,B)
\tkzDrawArc[reverse,tkz arrow={Stealth}](O,A)(B)
\tkzDrawPoints(A,B,O)
\end{tikzpicture}

26. Drawing a sector or sectors

26.1. \texttt{\tkzDrawSector}

Attention the arguments vary according to the options.

\begin{tabular}{|l|l|l|}
\hline
options & default & definition \\
\hline
\texttt{towards} & \texttt{towards} & O is the center and the arc from A to (OB) \\
\texttt{rotate} & \texttt{towards} & the arc starts from A and the angle determines its length \\
\texttt{R} & \texttt{towards} & We give the radius and two angles \\
\texttt{R with nodes} & \texttt{towards} & We give the radius and two points \\
\hline
\end{tabular}

You have to add, of course, all the styles of TikZ for tracings...

\begin{tabular}{|l|l|l|}
\hline
options & arguments & example \\
\hline
\texttt{towards} & \texttt{(pt,pt)} & \texttt{\tkzDrawSector(0,A)(B)} \\
\texttt{rotate} & \texttt{(pt,pt)} & \texttt{\tkzDrawSector[rotate, color=red](0,A)(90)} \\
\texttt{R} & \texttt{(pt,pt)} & \texttt{\tkzDrawSector[R, color=teal](0,2)(30,90)} \\
\texttt{R with nodes} & \texttt{(pt,pt)} & \texttt{\tkzDrawSector[R with nodes](0,2)(A,B)} \\
\hline
\end{tabular}

Here are a few examples:
26. Drawing a sector or sectors

26.1.1. \tkzDrawSector and towards

There's no need to put \texttt{towards}. You can use \texttt{fill} as an option.

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\tkzDefPoint(-30:1){A}
\tkzDefPointBy[rotation = center O angle -60](A)
\tkzDrawSector[teal](O,A)(tkzPointResult)
\begin{scope}[shift={(-60:1)}]
\tkzDefPoint(0,0){O}
\tkzDefPoint(-30:1){A}
\tkzDefPointBy[rotation = center O angle -60](A)
\tkzDrawSector[red](O,tkzPointResult)(A)
\end{scope}
\end{tikzpicture}

26.1.2. \tkzDrawSector and rotate

\begin{tikzpicture}[scale=2]
\tkzDefPoints{0/0/O,2/2/A,2/1/B}
\tkzDrawSector[rotate,orange](O,A)(20)
\tkzDrawSector[rotate,teal](O,B)(-20)
\end{tikzpicture}

26.1.3. \tkzDrawSector and R

\begin{tikzpicture}[scale=1.25]
\tkzDefPoint(0,0){O}
\tkzDefPoint(2,-1){A}
\tkzDrawSector[R](O,1)(30,90)
\tkzDrawSector[R](O,1)(90,180)
\tkzDrawSector[R](O,1)(180,270)
\tkzDrawSector[R](O,1)(270,360)
\end{tikzpicture}

26.1.4. \tkzDrawSector and R with nodes

In this example I use the option \texttt{fill} but \texttt{tkzFillSector} is possible.
26. Drawing a sector or sectors

\begin{tikzpicture}[scale=1.25]
\tkzDefPoint(0,0){O}
\tkzDefPoint(4,-2){A}
\tkzDefPoint(4,1){B}
\tkzDefPoint(3,3){C}
\tkzDrawSector[R with nodes, fill=teal!20](O,1)(B,C)
\tkzDrawSector[R with nodes, fill=orange!20](O,1.25)(A,B)
\tkzDrawSegments(O,A O,B O,C)
\tkzDrawPoints(O,A,B,C)
\tkzLabelPoints(A,B,C)
\tkzLabelPoints[left](O)
\end{tikzpicture}

26.1.5. \texttt{\tkzDrawSector} and R with nodes

\begin{tikzpicture} [scale=.4]
\tkzDefPoints{-1/-2/A,1/3/B}
\tkzDefRegPolygon[side,sides=6](A,B)
\tkzGetPoint{O}
\tkzDrawPolygon[fill=black!10, draw=blue](P1,P...,P6)
\tkzLabelRegPolygon[sep=1.05](O){A,...,F}
\tkzDrawCircle[dashed](O,A)
\tkzLabelSegment[above,sloped, midway](A,B){\(A B = 16 m\)}
\foreach \i [count=\xi from 1] in {2,...,6,1}
{\tkzDefMidPoint(P\xi,P\i)
path (O) to [pos=1.1] node \(\xi\) (tkzPointResult) ;}
\tkzDefRandPointOn[segment = P3--P5]
\tkzGetPoint{S}
\tkzDrawSegments[thick,dashed,red](A,S S,B)
\tkzDrawPoints(P1,P...,P6,S)
\tkzLabelPoint[left,above](S){\$S\$}
\tkzDrawSector[R with nodes,fill=red!20](S,2)(A,B)
\tkzLabelAngle[pos=1.5](A,S,B)\(\alpha\)
\end{tikzpicture}

26.2. Coloring a disc

This was possible with the macro \texttt{\tkzDrawCircle}, but disk tracing was mandatory, this is no longer the case.

\begin{tabular}{ll}
\texttt{\tkzFillCircle} & (local options)(\langle A,B \rangle) \\
\end{tabular}

\begin{tabular}{lr}
\texttt{options} & \texttt{default definition} \\
\hline
\texttt{radius} & \texttt{radius two points define a radius} \\
\texttt{R} & \texttt{radius a point and the measurement of a radius} \\
\end{tabular}

You don't need to put \texttt{radius} because that's the default option. Of course, you have to add all the styles of Ti\textsc{k}Z for the plots.
26. Drawing a sector or sectors

26.2.1. Yin and Yang

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(0,0){O}
\tkzDefPoint(-4,0){A}
\tkzDefPoint(4,0){B}
\tkzDefPoint(-2,0){I}
\tkzDefPoint(2,0){J}
\tkzDrawSector[fill=teal](O,A)(B)
\tkzFillCircle[fill=white](J,B)
\tkzFillCircle[fill=teal](I,A)
\tkzDrawCircle(O,A)
\end{tikzpicture}

26.2.2. From a sangaku

\begin{tikzpicture}
\tkzDefPoint(0,0){B} \tkzDefPoint(6,0){C}
\tkzDefSquare(B,C) \tkzGetPoints{D}{A}
\tkzClipPolygon(B,C,D,A)
\tkzDefMidPoint(A,D) \tkzGetPoint{F}
\tkzDefMidPoint(B,C) \tkzGetPoint{E}
\tkzDefMidPoint(B,D) \tkzGetPoint{Q}
\tkzDefLine[tangent from = B](F,A) \tkzGetPoints{H}{G}
\tkzInterLL(F,G)(C,D) \tkzGetPoint{J}
\tkzInterLL(A,J)(F,E) \tkzGetPoint{K}
\tkzDefPointBy[projection=onto B--A](K)
\tkzGetPoint{M}
\tkzDrawPolygon(A,B,C,D)
\tkzFillCircle[red!20](E,B)
\tkzFillCircle[blue!20](M,A)
\tkzFillCircle[green!20](K,Q)
\tkzDrawCircles(B,A M,A E,B K,Q)
\end{tikzpicture}
26. Drawing a sector or sectors

26.2.3. Clipping and filling part I

\begin{tikzpicture}
\tkzDefSquare(A,B)\tkzGetPoints{C}{D}
\tkzDefPointWith[colinear normed=at X,K=1](O,X)
\tkzGetPoint{F}
\begin{scope}
\tkzFillCircle[fill=teal!20](O,F)
\tkzFillPolygon[white](A,...,D)
\tkzClipPolygon(A,...,D)
\foreach \c/\t in {S/C,R/B,U/A,T/D}
{\tkzFillCircle[teal!20](\c,\t)}
\end{scope}
\foreach \c/\t in {X/C,Y/B,Z/A,W/D}
{\tkzFillCircle[white](\c,\t)}
\foreach \c/\t in {S/C,R/B,U/A,T/D}
{\tkzFillCircle[teal!20](\c,\t)}
\end{tikzpicture}

26.2.4. Clipping and filling part II

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,8/0/B,8/8/C,0/8/D}
\tkzDefMidPoint(A,B) \tkzGetPoint{F}
\tkzDefMidPoint(B,C) \tkzGetPoint{E}
\tkzDefMidPoint(D,B) \tkzGetPoint{I}
\tkzDefMidPoint(I,B) \tkzGetPoint{a}
\tkzInterLC(B,I)(B,C) \tkzGetSecondPoint{K}
\tkzDefMidPoint(I,K) \tkzGetPoint{b}
\begin{scope}
\tkzFillSector[fill=blue!10](B,C)(A)
\tkzDefMidPoint(A,B) \tkzGetPoint{x}
\tkzDrawSemiCircle[fill=white](x,B)
\tkzDefMidPoint(B,C) \tkzGetPoint{y}
\tkzDrawSemiCircle[fill=white](y,C)
\tkzClipCircle(E,B)
\tkzClipCircle(F,B)
\tkzClipCircle[fill=blue!10](B,A)
\end{scope}
\tkzDrawSemiCircle[thick](F,B)
\tkzDrawSemiCircle[thick](E,C)
\tkzDrawArc[thick](B,C)(A)
\tkzDrawSegments[thick](A,B,B,C)
\tkzDrawPoints(A,B,C,E,F)
\tkzLabelPoints[centered](a,b)
\tkzLabelPoints(A,B,C,E,F)
\end{tikzpicture}
26.2.5. Clipping and filling part III

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(1,0){B} \\
\tkzDefPoint(2,0){C} \tkzDefPoint(-3,0){a} \\
\tkzDefPoint(3,0){b} \tkzDefPoint(0,3){c} \\
\tkzDefPoint(0,-3){d} \\
\begin{scope} \\
\tkzClipPolygon(a,b,c,d) \\
\tkzFillCircle[teal!20](A,C) \\
\end{scope} \\
\tkzFillCircle[white](A,B) \\
\tkzDrawCircle[color=red](A,C) \\
\tkzDrawCircle[color=red](A,B) \\
\end{tikzpicture}

26.3. Coloring a polygon

\tkzFillPolygon[⟨local options⟩]{⟨points list⟩}

You can color by drawing the polygon, but in this case you color the inside of the polygon without drawing it.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨pt1,pt2,…⟩</td>
<td>⟨A,B,…⟩</td>
<td></td>
</tr>
</tbody>
</table>

26.3.1. \tkzFillPolygon

\begin{tikzpicture}[scale=.5] \\
\tkzDefPoint(0,0){C} \tkzDefPoint(4,0){A} \\
\tkzDefPoint(0,3){B} \\
\tkzDefSquare(B,A) \tkzGetPoints{E}{F} \\
\tkzDefSquare(A,C) \tkzGetPoints{G}{H} \\
\tkzDefSquare(C,B) \tkzGetPoints{I}{J} \\
\tkzFillPolygon[color = orange!30 ]{(A,C,G,H)} \\
\tkzFillPolygon[color = teal!40 ]{(C,B,I,J)} \\
\tkzFillPolygon[color = purple!20](B,A,E,F) \\
\tkzDrawPolygon[line width = 1pt](A,B,C) \\
\tkzDrawPolygon[line width = 1pt](A,C,G,H) \\
\tkzDrawPolygon[line width = 1pt](C,B,I,J) \\
\tkzDrawPolygon[line width = 1pt](B,A,E,F) \\
\tkzLabelSegment[above](C,A){$a$} \\
\tkzLabelSegment[right](B,C){$b$} \\
\tkzLabelSegment[below left](B,A){$c$} \\
\end{tikzpicture}

26.4. \tkzFillSector

⚠️ Attention the arguments vary according to the options.
26. Drawing a sector or sectors

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>towards</td>
<td>towards</td>
<td>O is the center and the arc from A to (OB)</td>
</tr>
<tr>
<td>rotate</td>
<td>towards</td>
<td>the arc starts from A and the angle determines its length</td>
</tr>
<tr>
<td>R</td>
<td>towards</td>
<td>We give the radius and two angles</td>
</tr>
<tr>
<td>R with nodes</td>
<td>towards</td>
<td>We give the radius and two points</td>
</tr>
</tbody>
</table>

Of course, you have to add all the styles of TiKZ for the tracings...

<table>
<thead>
<tr>
<th>options</th>
<th>arguments</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>towards</td>
<td>(pt,pt)(pt)</td>
<td>\tkzFillSector(O,A)(B)</td>
</tr>
<tr>
<td>rotate</td>
<td>(pt,pt)(an)</td>
<td>\tkzFillSector<a href="O,A">rotate,color=red</a>(90)</td>
</tr>
<tr>
<td>R</td>
<td>(pt,r)(an,an)</td>
<td>\tkzFillSector<a href="O,2">R,color=blue</a>(30,90)</td>
</tr>
<tr>
<td>R with nodes</td>
<td>(pt,r)(pt,pt)</td>
<td>\tkzFillSector<a href="O,2">R with nodes</a>(A,B)</td>
</tr>
</tbody>
</table>

26.4.1. \tkzFillSector and towards

It is useless to put towards and you will notice that the contours are not drawn, only the surface is colored.

```latex
\begin{tikzpicture}[scale=.6]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzFillSector[fill=purple!20](O,A)(tkzPointResult)
  \begin{scope}[shift={(-60:1)}]
    \tkzDefPoint(0,0){O}
    \tkzDefPoint(-30:3){A}
    \tkzGetPoint{A'}
    \tkzFillSector[color=teal!40](O,A')(A)
  \end{scope}
\end{tikzpicture}
```

26.4.2. \tkzFillSector and rotate

```latex
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O} \tkzDefPoint(2,2){A}
  \tkzFillSector[rotate,color=purple!20](O,A)(30)
  \tkzFillSector[rotate,color=teal!40](O,A)(-30)
\end{tikzpicture}
```

26.5. Colour an angle: \tkzFillAngle

The simplest operation
26. Drawing a sector or sectors

O is the vertex of the angle. OA and OB are the sides. Attention the angle is determined by the order of the points.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>1</td>
<td>this option determines the radius of the coloured angular sector.</td>
</tr>
</tbody>
</table>

Of course, you have to add all the styles of Ti\kZ, like the use of fill and shade...

26.5.1. Example with size

\begin{tikzpicture}
\tkzInit
\tkzDefPoints{0/0/O,2.5/0/A,1.5/2/B}
\tkzFillAngle[size=2, fill=gray!10](A,O,B)
\tkzDrawLines(O,A O,B)
\tkzDrawPoints(O,A,B)
\end{tikzpicture}

26.5.2. Changing the order of items

\begin{tikzpicture}
\tkzInit
\tkzDefPoints{0/0/O,2.5/0/A,1.5/2/B}
\tkzFillAngle[size=2, fill=gray!10](B,O,A)
\tkzDrawLines(O,A O,B)
\tkzDrawPoints(O,A,B)
\end{tikzpicture}

\begin{tikzpicture}
\tkzInit
\tkzDefPoints{0/0/O,5/0/A,3/4/B}
% Don't forget {} to get, () to use
\tkzFillAngle[size=4, left color=white, right color=red!50](A,O,B)
\tkzDrawLines(O,A O,B)
\tkzDrawPoints(O,A,B)
\end{tikzpicture}

\txkFillAngles[(local options)]{(A,O,B),(A',O',B')}etc.

With common options, there is a macro for multiple angles.
27. Controlling Bounding Box

From the \textit{PgfManual} \textquotedbl{}When you add the clip option, the current path is used for clipping subsequent drawings. Clipping never enlarges the clipping area. Thus, when you clip against a certain path and then clip again against another path, you clip against the intersection of both. The only way to enlarge the clipping path is to end the pgfscope in which the clipping was done. At the end of a pgfscope the clipping path that was in force at the beginning of the scope is reinstalled.\textquotedbl{} First of all, you don't have to deal with TiKZ the size of the bounding box. Early versions of \textit{tkz-euclide} did not control the size of the bounding box, now with \textit{tkz-euclide} 4 the size of the bounding box is limited.

The initial bounding box after using the macro \texttt{\tkzInit} is defined by the rectangle based on the points \((0, 0)\) and \((10, 10)\). The \texttt{\tkzInit} macro allows this initial bounding box to be modified using the arguments \((\text{xmin}, \text{xmax}, \text{ymin}, \text{ymax})\). Of course any external trace modifies the bounding box. TiKZ maintains that bounding box. It is possible to influence this behavior either directly with commands or options in TiKZ such as a command like \texttt{useasboundingbox} or the option \texttt{use as bounding box}. A possible consequence is to reserve a box for a figure but the figure may overflow the box and spread over the main text. The following command \texttt{\pgfresetboundingbox} clears a bounding box and establishes a new one.

27.1. Utility of \texttt{\tkzInit}

However, it is sometimes necessary to control the size of what will be displayed. To do this, you need to have prepared the bounding box you are going to work in, this is the role of the macro \texttt{\tkzInit}. For some drawings, it is interesting to fix the extreme values \((\text{xmin}, \text{xmax}, \text{ymin}, \text{ymax})\) and to "clip" the definition rectangle in order to control the size of the figure as well as possible.

The two macros that are useful for controlling the bounding box:

\begin{itemize}
  \item \texttt{\tkzInit}
  \item \texttt{\tkzClip}
\end{itemize}

To this, I added macros directly linked to the bounding box. You can now view it, backup it, restore it (see the section Bounding Box).
27.2. \texttt{tkzInit}

\begin{verbatim}
\begin{tikzpicture}
\tkzInit[xmax=4, ymax=3]
\tkzDefPoints{-1/-1/A,5/2/B}
\tkzDrawX \tkzDrawY
\tkzGrid
\tkzClip
\tkzDrawSegment(A,B)
\end{tikzpicture}
\end{verbatim}

The role of \texttt{tkzInit} is to define an orthogonal coordinates system and a rectangular part of the plane in which you will place your drawings using Cartesian coordinates. This macro allows you to define your working environment as with a calculator. With \texttt{tkz-euclide}, \texttt{xstep} and \texttt{ystep} are always 1. Logically it is no longer useful to use \texttt{tkzInit}, except for an action like “Clipping Out”.

27.3. \texttt{tkzClip}

\begin{verbatim}
\begin{tikzpicture}
\tkzClip[space=1]
\tkzDrawSegment(A,B)
\end{tikzpicture}
\end{verbatim}

The role of \texttt{tkzClip} is to “clip” the initial rectangle so that only the paths contained in this rectangle are drawn.

It is possible to add a bit of space

\begin{verbatim}
\tkzClip[space=1]
\end{verbatim}

27.4. \texttt{tkzClip} and the option \texttt{space}

This option allows you to add some space around the “clipped” rectangle.
27. Controlling Bounding Box

The dimensions of the "clipped" rectangle are \( \text{xmin}-1, \text{ymin}-1, \text{xmax}+1 \) and \( \text{ymax}+1 \).

27.5. \texttt{tkzShowBB}

The simplest macro.

\begin{verbatim}
\texttt{tkzShowBB[(local options)]}
\end{verbatim}

This macro displays the bounding box. A rectangular frame surrounds the bounding box. This macro accepts Ti\textit{k}Z options.

27.5.1. Example with \texttt{tkzShowBB}

\begin{verbatim}
\begin{tikzpicture}[scale=.5]
\tkzInit[ymax=5,xmax=8]
\tkzGrid
\tkzDefPoint(3,0){A}
\begin{scope}
\tkzClipBB
\tkzDefCircle[R](A,5) \tkzGetPoint{a}
\tkzDrawCircle(A,a)
\tkzShowBB[line width = 4pt,fill=teal!10,opacity=.4]
\end{scope}
\tkzDefCircle[R](A,4) \tkzGetPoint{b}
\tkzDrawCircle[red](A,b)
\end{tikzpicture}
\end{verbatim}

27.6. \texttt{tkzClipBB}

\begin{verbatim}
\texttt{tkzClipBB}
\end{verbatim}

The idea is to limit future constructions to the current bounding box.
27.6.1. Example with \tkzClipBB and the bisectors

\begin{tikzpicture}
\tkzInit[xmin=-3,xmax=6, ymin=-1,ymax=6]
\tkzDefPoint(0,0){O}\tkzDefPoint(3,1){I}
\tkzDefPoint(1,4){J}
\tkzDefLine[bisector](I,O,J) \tkzGetPoint{i}
\tkzDefLine[bisector out](I,O,J) \tkzGetPoint{j}
\tkzDrawPoints(O,I,J,i,j)
\tkzClipBB
\tkzDrawLines[add = 1 and 2,color=orange](O,I O,J)
\tkzDrawLines[add = 1 and 2](O,i O,j)
\tkzShowBB
\end{tikzpicture}
28. Clipping different objects

28.1. Clipping a polygon

\[ \texttt{\textbackslash{tkzClipPolygon}[(local options)](points list)} \]

This macro makes it possible to contain the different plots in the designated polygon.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>\langle pt1,pt2,pt3,...⟩ \rangle \langle A,B,C⟩ \rangle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>options</td>
<td>default</td>
<td>definition</td>
</tr>
<tr>
<td>out</td>
<td></td>
<td>allows to clip the outside of the object</td>
</tr>
</tbody>
</table>

28.1.1. \texttt{\textbackslash{tkzClipPolygon}}

\begin{verbatim}
\begin{tikzpicture}[scale=1.25]
tkzDefPoint(0,0){A}
tkzDefPoint(4,0){B}
tkzDefPoint(1,3){C}
tkzDrawPolygon(A,B,C)
tkzDefPoint(0,2){D}
tkzDefPoint(2,0){E}
tkzDrawPoints(D,E)
tkzLabelPoints(D,E)
tkzClipPolygon(A,B,C)
tkzDrawLine[new](D,E)
\end{tikzpicture}
\end{verbatim}

28.1.2. \texttt{\textbackslash{tkzClipPolygon}[out]}

\begin{verbatim}
\begin{tikzpicture}[scale=1]
tkzDefPoint(0,0){P1}
tkzDefPoint(4,0){P2}
tkzDefPoint(4,4){P3}
tkzDefPoint(0,4){P4}
tkzDefPoint(1,1){Q1}
tkzDefPoint(3,1){Q2}
tkzDefPoint(3,3){Q3}
tkzDefPoint(1,3){Q4}
tkzDrawPolygon(P1,P2,P3,P4)
\begin{scope}
tkzClipPolygon[out](Q1,Q2,Q3,Q4)
tkzFillPolygon[teal!20](P1,P2,P3,P4)
\end{scope}
tkzDrawPolygon(Q1,Q2,Q3,Q4)
\end{tikzpicture}
\end{verbatim}
28. Clipping different objects

28.1.3. Example: use of "Clip" for Sangaku in a square

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(0,0){A} \tkzDefPoint(8,0){B}
\tkzDefSquare(A,B) \tkzGetPoints{C}{D}
\tkzDefPoint(4,8){F}
\tkzDefTriangle[equilateral](C,D)
\tkzGetPoint{I}
\tkzDefPointBy[projection=onto B--C](I)
\tkzGetPoint{J}
\tkzInterLL(D,B)(I,J) \tkzGetPoint{K}
\tkzDefPointBy[symmetry=center K](B)
\tkzGetPoint{M}
\tkzClipPolygon(B,C,D,A)
\tkzFillPolygon[color = orange](A,B,C,D)
\tkzFillCircle[color = yellow](M,I)
\tkzFillCircle[color = blue!50!black](F,D)
\end{tikzpicture}

28.2. Clipping a disc

\begin{tabular}{|c|c|c|}
\hline
arguments & example & explanation \\
\hline
(A,B) & (A,B) & AB radius \\
\hline
options & default & definition \\
\hline
out & & allows to clip the outside of the object \\
\hline
\end{tabular}

It is not necessary to put radius because that is the default option.

28.2.1. Simple clip

\begin{tikzpicture}[scale=.5]
\tkzDefPoint(0,0){A} \tkzDefPoint(2,2){O}
\tkzDefPoint(4,4){B} \tkzDefPoint(5,5){C}
\tkzDrawPoints(O,A,B,C)
\tkzLabelPoints(O,A,B,C)
\tkzDrawCircle(O,A)
\tkzClipCircle(O,A)
\tkzDrawLine(A,C)
\tkzDrawCircle[fill=teal!10,opacity=.5](C,O)
\end{tikzpicture}
28. Clipping different objects

28.3. Clip out

\begin{tikzpicture}
\tkzInit[xmin=-3,ymin=-2,xmax=4,ymax=3]
\tkzDefPoint(0,0){O}
\tkzDefPoint(-4,-2){A}
\tkzDefPoint(3,1){B}
\tkzDefCircle[R](O,2) \tkzGetPoint{o}
\tkzDrawPoints(A,B)
% to have a good bounding box
\begin{scope}
\tkzClipCircle[out](O,o)
\tkzDrawLines(A,B)
\end{scope}
\end{tikzpicture}

28.4. Intersection of disks

\begin{tikzpicture}
\tkzDefPoints{0/0/O,4/0/A,0/4/B}
\tkzDrawPolygon[fill=teal](O,A,B)
\tkzClipPolygon(O,A,B)
\tkzClipCircle(A,O)
\tkzClipCircle(B,O)
\tkzFillPolygon[white](O,A,B)
\end{tikzpicture}

see a more complex example about clipping here: 46.6

28.5. Clipping a sector

Attention the arguments vary according to the options.

\begin{tabular}{|c|c|l|}
\hline
\textbf{options} & \textbf{default} & \textbf{definition} \\
\hline
towards & towards & O is the center and the sector starts from A to (OB) \\
rotate & towards & The sector starts from A and the angle determines its amplitude. \\
R & towards & We give the radius and two angles \\
\hline
\end{tabular}

You have to add, of course, all the styles of TikZ for tracings...

\begin{tabular}{|c|c|l|}
\hline
\textbf{options} & \textbf{arguments} & \textbf{example} \\
\hline
towards & ((pt,pt)) & \tkzClipSector(0,A)(B) \\
rotate & ((pt,pt)) & \tkzClipSector[rotate](0,A)(90) \\
R & ((pt,r)) & \tkzClipSector[R](0,2)(30,90) \\
\hline
\end{tabular}
28. Clipping different objects

28.5.1. Example 1

\begin{tikzpicture}[scale=0.5]
\tkzDefPoint(0,0){a}
\tkzDefPoint(12,0){b}
\tkzDefPoint(4,10){c}
\tkzInterCC[R](a,6)(b,8)
\tkzGetFirstPoint{AB1} \tkzGetSecondPoint{AB2}
\tkzInterCC[R](a,6)(c,6)
\tkzGetFirstPoint{AC1} \tkzGetSecondPoint{AC2}
\tkzInterCC[R](b,8)(c,6)
\tkzGetFirstPoint{BC1} \tkzGetSecondPoint{BC2}
\tkzDrawArc(a,AB2)(AB1)
\tkzDrawArc(b,AB1)(AB2)
\tkzDrawArc(a,AC2)(AC1)
\tkzDrawArc(c,AC1)(AC2)
\tkzDrawArc(b,BC2)(BC1)
\tkzDrawArc(c,BC1)(BC2)
\begin{scope}
\tkzClipSector(b,BC2)(BC1)
\tkzFillSector[teal!40!white](c,BC1)(BC2)
\end{scope}
\begin{scope}
\tkzClipSector(a,AB2)(AB1)
\tkzFillSector[teal!40!white](b,AB1)(AB2)
\end{scope}
\begin{scope}
\tkzClipSector(a,AC2)(AC1)
\tkzFillSector[teal!40!white](c,AC1)(AC2)
\end{scope}
\end{tikzpicture}

28.5.2. Example 2

\begin{tikzpicture}[scale=1.5]
\tkzDefPoint(0,0){O}
\tkzDefPoint(2,-1){A}
\tkzDefPoint(1,1){B}
\tkzDrawSector[new,dashed](O,A)(B)
\tkzDrawSector[new](O,B)(A)
\begin{scope}
\tkzClipSector(O,B)(A)
\tkzDefSquare(O,B) \tkzGetPoints{B'}{O'}
\tkzDrawPolygon[color=teal,fill=teal!20](O,B,B',O')
\end{scope}
\tkzDrawPoints(A,B,O)
\end{tikzpicture}

28.6. Options from TikZ: trim left or right

See the \texttt{pgfmanual}

28.7. TikZ Controls \texttt{\pgfinterruptboundingbox} and \texttt{endpgfinterruptboundingbox}

This command temporarily interrupts the calculation of the box and configures a new box. See the \texttt{pgfmanual}
28. Clipping different objects

28.7.1. Example about controlling the bounding box

\begin{tikzpicture}
\tkzDefPoint(0,5){A}\tkzDefPoint(5,4){B}
\tkzDefPoint(0,0){C}\tkzDefPoint(5,1){D}
\tkzDrawSegments(A,B C,D A,C)
\pgfinterruptboundingbox
 \tkzInterLL(A,B)(C,D)\tkzGetPoint{I}
\endpgfinterruptboundingbox
\tkzClipBB
\tkzDrawCircle(I,B)
\end{tikzpicture}

28.8. Reverse clip: \texttt{tkzreverseclip}

In order to use this option, a bounding box must be defined.

\begin{verbatim}
\tikzset{tkzreverseclip/.style={insert path={
  (current bounding box.south west) -- (current bounding box.north west) -- (current bounding box.north east) -- (current bounding box.south east) -- cycle}}}
\end{verbatim}

28.8.1. Example with \texttt{tkzClipPolygon[out]}

\texttt{tkzClipPolygon[out], tkzClipCircle[out]} use this option.
\begin{tikzpicture}[scale=1]
\tkzInit[xmin=-5,xmax=5,ymin=-4,ymax=6]
\tkzClip
\tkzDefPoints{-5/0/P1,5/0/P2}
\foreach \i [count=\j from 3] in {2,...,7}{% 
  \tkzDefShiftPoint[P\i]((45*(\i-1)):1){P\j}}
\tkzClipPolygon[out](P1,P...,P8)
\tkzCalcLength(P1,P5)\tkzGetLength{r}
\begin{scope}[blend group=screen]
  \foreach \i in {1,...,8}{% 
    \tkzDefCircle[R\i](P\i,r) \tkzGetPoint{x}
    \tkzFillCircle[color=teal](P\i,x)}
  \end{scope}
\end{tikzpicture}
Part V.

Marking
28.9. Mark a segment \texttt{tkzMarkSegment}

\begin{verbatim}
\texttt{tkzMarkSegment[(local options)](pt1,pt2)}
\end{verbatim}

The macro allows you to place a mark on a segment.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pos</td>
<td>.5</td>
<td>position of the mark</td>
</tr>
<tr>
<td>color</td>
<td>black</td>
<td>color of the mark</td>
</tr>
<tr>
<td>mark</td>
<td>none</td>
<td>choice of the mark</td>
</tr>
<tr>
<td>size</td>
<td>4pt</td>
<td>size of the mark</td>
</tr>
</tbody>
</table>

Possible marks are those provided by TikZ, but other marks have been created based on an idea by Yves Combe.

28.9.1. Several marks

\begin{tikzpicture}
\tkzDefPoint(2,1){A}
\tkzDefPoint(6,4){B}
\tkzDrawSegment(A,B)
\tkzMarkSegment[color=brown,size=2pt,pos=0.4, mark=z](A,B)
\tkzMarkSegment[color=blue,pos=0.2, mark=oo](A,B)
\tkzMarkSegment[pos=0.8,mark=s,color=red](A,B)
\end{tikzpicture}

28.9.2. Use of mark

\begin{tikzpicture}
\tkzDefPoint(2,1){A}
\tkzDefPoint(6,4){B}
\tkzDrawSegment(A,B)
\tkzMarkSegment[color=gray,pos=0.2,mark=s|](A,B)
\tkzMarkSegment[color=gray,pos=0.4,mark=s||](A,B)
\tkzMarkSegment[color=gray,pos=0.6,mark=|||](A,B)
\tkzMarkSegment[color=gray,pos=0.8,mark=||||](A,B)
\end{tikzpicture}

28.10. Marking segments \texttt{tkzMarkSegments}

\begin{verbatim}
\texttt{tkzMarkSegments[(local options)](pt1,pt2 pt3,pt4 ...)}
\end{verbatim}

Arguments are a list of pairs of points separated by spaces. The styles of TikZ are available for plots.

28.10.1. Marks for an isosceles triangle

\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/O,2/2/A,4/0/B,6/2/C}
\tkzDrawSegments(O,A,A,B)
\tkzDrawPoints(O,A,B)
\tkzDrawLine(O,B)
\tkzMarkSegments[mark=||,size=6pt](O,A,A,B)
\end{tikzpicture}
28.11. Another marking

\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A}\tkzDefPoint(3,2){B}
  \tkzDefPoint(4,0){C}\tkzDefPoint(2.5,1){P}
  \tkzDrawPolygon(A,B,C)
  \tkzDefEquilateral(A,P) \tkzGetPoint{P'}
  \tkzDefPointsBy[rotation=center A angle 60](P,B){P',C'}
  \tkzDrawPolygon(A,P,P')
  \tkzDrawPolySeg(P',C',A,P,B)
  \tkzDrawSegment(C,P)
  \tkzDefPointsBy[rotation=center A angle 60](P,B){P',C'}
  \tkzDrawPolygon(A,P,P')
  \tkzDrawPolySeg(P',C',A,P,B)
  \tkzDrawPoints(A,B,C,C',P,P')
  \tkzMarkSegments[mark=s|,size=6pt,color=blue](A,P,P,P',A)
  \tkzMarkSegments[mark=||,color=orange](B,P,P',C')
  \tkzLabelPoints(A,C) \tkzLabelPoints[below](P)
  \tkzLabelPoints[above right](P',C',B)
\end{tikzpicture}

28.12. Mark an arc \tkzMarkArc

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\pgfmathsetmacro\r{2}
\tkzDefPoint(30:\r){A}
\tkzDefPoint(85:\r){B}
\tkzDrawCircle(O,A)
\tkzMarkArc[color=red,mark=||](O,A,B)
\tkzDrawPoints(B,A,O)
\end{tikzpicture}

The macro allows you to place a mark on an arc. pt1 is the center, pt2 and pt3 are the endpoints of the arc.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pos</td>
<td>.5</td>
<td>position of the mark</td>
</tr>
<tr>
<td>color</td>
<td>black</td>
<td>color of the mark</td>
</tr>
<tr>
<td>mark</td>
<td>none</td>
<td>choice of the mark</td>
</tr>
<tr>
<td>size</td>
<td>4pt</td>
<td>size of the mark</td>
</tr>
</tbody>
</table>

Possible marks are those provided by TikZ, but other marks have been created based on an idea by Yves Combe.

\[\ldots, ||, |||, z, s, x, o, oo\]

28.12.1. Several marks

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\tkzDefPoint(30:\r){A}
\tkzDefPoint(85:\r){B}
\tkzDrawCircle(O,A)
\tkzMarkArc[color=red,mark=||](O,A,B)
\tkzDrawPoints(B,A,O)
\end{tikzpicture}

28.13. Mark an angle mark : \tkzMarkAngle

More delicate operation because there are many options. The symbols used for marking in addition to those of TikZ are defined in the file tkz-lib-marks.tex and designated by the following characters:
O is the vertex. Attention the arguments vary according to the options. Several markings are possible. You can simply draw an arc or add a mark on this arc. The style of the arc is chosen with the option \texttt{arc}, the radius of the arc is given by \texttt{mksize}, the arc can, of course, be colored.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>arc</td>
<td>1</td>
<td>choice of 1, ll and lll (single, double or triple).</td>
</tr>
<tr>
<td>size</td>
<td>1 (cm)</td>
<td>arc radius.</td>
</tr>
<tr>
<td>mark</td>
<td>none</td>
<td>choice of mark.</td>
</tr>
<tr>
<td>mksize</td>
<td>4pt</td>
<td>symbol size (mark).</td>
</tr>
<tr>
<td>mkcolor</td>
<td>black</td>
<td>symbol color (mark).</td>
</tr>
<tr>
<td>mkpos</td>
<td>0.5</td>
<td>position of the symbol on the arc.</td>
</tr>
</tbody>
</table>

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/O,5/0/A,3/4/B}
\tkzMarkAngle[size = 4,mark = x, arc=ll,mkcolor = red,mkpos=.33](A,O,B)
\tkzMarkAngle[size = 2,mark = ||, arc=ll,mkcolor = blue,mkpos=.66](A,O,B)
\tkzDrawLines(O,A O,B)
\tkzDrawPoints(O,A,B)
\end{tikzpicture}

With common options, there is a macro for multiple angles.

28.14. Problem to mark a small angle: Option veclen

The problem comes from the "decorate" action and from the value used in size in \texttt{tkzMarkAngle}. The solution is to enclose the macro \texttt{tkzMarkAngle}. In the next example without the "scope" the result is: \texttt{LaTeX Error: Dimension too large}.
28.15. Marking a right angle: \texttt{tkzMarkRightAngle}

\begin{verbatim}
\tkzMarkRightAngle[(\texttt{local options})](\texttt{\langle A, O, B \rangle})
\end{verbatim}

The \texttt{german} option allows you to change the style of the drawing. The option \texttt{size} allows to change the size of the drawing.

\begin{tabular}{llp{5cm}}
\hline
options & default & definition \\
\hline
\texttt{german} & \texttt{normal} & \texttt{german arc with inner point}. \\
\texttt{size} & \texttt{0.2} & \texttt{side size}. \\
\hline
\end{tabular}

28.15.1. Example of marking a right angle

\begin{verbatim}
\begin{tikzpicture}
\tkzDefPoints{\texttt{0/0/A,3/1/B,0.9/-1.2/P}}
\tkzDefPointBy[projection = onto \texttt{B--A}](\texttt{P}) \tkzGetPoint{H}
\tkzMarkRightAngle[fill=blue!20,size=.5,draw](\texttt{A,H,P})
\tkzMarkRightAngle[fill=red!20,size=.8](\texttt{B,H,P})
\tkzDrawPoints[\texttt{}](\texttt{\langle A,B,P,H \rangle})
\end{tikzpicture}
\end{verbatim}
28.15.2. Example of marking a right angle, german style

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/1/B,0.9/-1.2/P}
\tkzDefPointBy[projection = onto B--A](P) \tkzGetPoint{H}
\tkzDrawLines[add=.5 and .5](P,H)
\tkzMarkRightAngle[german,size=.5,draw](A,H,P)
\tkzDrawPoints[](A,B,P,H)
\tkzDrawLines[add=.5 and .5](A,B)
\tkzMarkRightAngle[german,size=.8](P,H,B)
\end{tikzpicture}

28.15.3. Mix of styles

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,1){B}
\tkzDefPoint(2,5){C}
\tkzDefPointBy[projection=onto B--A](C)
\tkzGetPoint{H}
\tkzDrawLine(A,B)
\tkzDrawLine[add = .5 and .2,color=red](C,H)
\tkzMarkRightAngle[.size=1,color=red](C,H,A)
\tkzMarkRightAngle[german,size=.8,color=blue](B,H,C)
\tkzFillAngle[opacity=.2,fill=blue!20,size=.8](B,H,C)
\tkzLabelPoints(A,B,C,H)
\tkzDrawPoints(A,B,C,H)
\end{tikzpicture}
28.15.4. Full example

\begin{tikzpicture}[rotate=-90]
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDrawSegment[green!60!black](A,C)
\tkzDrawSegment[green!60!black](C,B)
\tkzDrawSegment[green!60!black](B,A)
\tkzDefSpcTriangle[orthic](A,B,C){N,O,P}
\tkzDrawLine[dashed,color=magenta](C,P)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[left](P){$P$}
\tkzLabelSegment[auto](B,A){$c$}
\tkzLabelSegment[auto,swap](B,C){$a$}
\tkzLabelSegment[auto,swap](C,A){$b$}
\tkzMarkAngle[size=1,color=cyan,mark=\{\}](C,B,A)
\tkzMarkAngle[size=1,color=cyan,mark=\{\}](A,C,P)
\tkzMarkAngle[size=0.75,color=orange, mark=\{\}](P,C,B)
\tkzMarkAngle[size=0.75,color=orange, mark=\{\}](B,A,C)
\tkzMarkRightAngle[german](A,C,B)
\tkzMarkRightAngle[german](B,P,C)
\end{tikzpicture}

28.16. \texttt{tkzMarkRightAngles}

\texttt{tkzMarkRightAngles[(\textbf{local options})](\langle A',O',B'\rangle)\textetc.}

With common options, there is a macro for multiple angles.

28.17. Angles Library

If you prefer to use Ti\textsc{k}Z library \texttt{angles}, you can mark angles with the macro \texttt{tkzPicAngle} and \texttt{tkzPicRightAngle}.

\texttt{tkzPicAngle[(\texttt{tikz options})](\langle A,O,B\rangle)}

<table>
<thead>
<tr>
<th>options</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tikz option</td>
<td>see below drawing of the angle $\hat{AOB}$</td>
<td></td>
</tr>
</tbody>
</table>

\texttt{tkzPicRightAngle[(\texttt{tikz options})](\langle A,O,B\rangle)}

<table>
<thead>
<tr>
<th>options</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tikz option</td>
<td>see below drawing of the right angle $\hat{AOB}$</td>
<td></td>
</tr>
</tbody>
</table>

\textit{You need to know possible options of the \texttt{angles} library}
28.17.1. Angle with TikZ

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B}
\tkzDefTriangle[right,swap](A,B) \tkzGetPoint{C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints[below](B,A)
\tkzLabelPoints[above right](C)
\tkzPicAngle["$\alpha$",draw=orange,\]
\tkzPicRightAngle[draw,red,thick,\]
\end{tikzpicture}
Part VI.

Labelling
29. Labelling

29.1. Label for a point

It is possible to add several labels at the same point by using this macro several times.

\begin{tabular}{|c|c|}
\hline
arguments & example \\
\hline
point & \tkzLabelPoint(A){$A_1$} \\
\hline
options & default definition \\
\hline
TikZ options & colour, position etc. \\
\hline
\end{tabular}

Optionally, we can use any style of Ti\textsc{k}Z, especially placement with above, right, dots...

29.1.1. Example with \tkzLabelPoint

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
\tkzDefPoint(0,3){C}
\tkzDrawSegments(A,B,B,C,C,A)
\tkzDrawPoints(A,B,C)
\tkzLabelPoint[left,red](A){$A$}
\tkzLabelPoint[right,blue](B){$B$}
\tkzLabelPoint[above,purple](C){$C$}
\end{tikzpicture}

29.1.2. Label and reference

The reference of a point is the object that allows to use the point, the label is the name of the point that will be displayed.

\begin{tikzpicture}
\tkzDefPoint(2,0){A}
\tkzDrawPoint(A)
\tkzLabelPoint[above](A){$A_1$}
\end{tikzpicture}

29.2. Add labels to points \tkzLabelPoints

It is possible to place several labels quickly when the point references are identical to the labels and when the labels are placed in the same way in relation to the points. By default, below right is chosen.

\begin{tabular}{|c|c|}
\hline
arguments & example \\
\hline
list of points & \tkzLabelPoints(A,B,C) Display of A, B and C \\
\hline
\end{tabular}

This macro reduces the number of lines of code, but it is not obvious that all points need the same label positioning.
29.2.1. Example with \texttt{\tkzLabelPoints}

\begin{tikzpicture}
\tkzDefPoint(2,3){A}
\tkzDefShiftPoint[A](30:2){B}
\tkzDefShiftPoint[A](30:5){C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B,C)
\end{tikzpicture}

29.3. Automatic position of labels \texttt{\tkzAutoLabelPoints}

The label of a point is placed in a direction defined by a center and a point center. The distance to the point is determined by a percentage of the distance between the center and the point. This percentage is given by dist.

\begin{verbatim}
\tkzLabelPoints[(local options)]((A_1,A_2,...))
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>list of points</td>
<td>\tkzLabelPoint(A,B,C)</td>
<td>Display of A, B and C</td>
</tr>
</tbody>
</table>

29.3.1. Label for points with \texttt{\tkzAutoLabelPoints}

Here the points are positioned relative to the center of gravity of A, B, C and O.

\begin{verbatim}
\begin{tikzpicture}[scale=1]
\tkzDefPoint(2,1){O}
\tkzDefRandPointOn[circle=center O radius 1.5]\tkzGetPoint{A}
\tkzDefPointBy[rotation=center O angle 100](A)\tkzGetPoint{C}
\tkzDefPointBy[rotation=center O angle 78](A)\tkzGetPoint{B}
\tkzDrawCircle(O,A)
\tkzDrawPoints(O,A,B,C)
\tkzDrawSegments(C,B B,A A,O O,C)
\tkzDefTriangleCenter[centroid](A,B,C) \tkzGetPoint{O}
\tkzDrawPoint(tkzPointResult)
\tkzLabelPoints(O,A,C,B)
\end{tikzpicture}
\end{verbatim}

30. Label for a segment

\begin{verbatim}
\tkzLabelSegment[(local options)]((pt1,pt2)){(label)}
\end{verbatim}

This macro allows you to place a label along a segment or a line. The options are those of \texttt{TikZ} for example pos.

<table>
<thead>
<tr>
<th>argument</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>\tkzLabelSegment(A,B){5}</td>
<td>label text</td>
</tr>
<tr>
<td>(pt1,pt2)</td>
<td>(A,B)</td>
<td>label along [AB]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pos</td>
<td>.5</td>
<td>label's position</td>
</tr>
</tbody>
</table>
30.9.1. First example

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(6,0){B}
\tkzDrawSegment(A,B)
\tkzLabelSegment[above,pos=.8](A,B){$a$}
\tkzLabelSegment[below,pos=.2](A,B){$4$}
\end{tikzpicture}

30.9.2. Example : blackboard

\begin{tikzpicture}[show background rectangle,scale=.4]
\tkzDefPoint(0,0){O}
\tkzDefPoint(1,0){I}
\tkzDefPoint(10,0){A}
\tkzDefPointWith[orthogonal normed,K=4](I,A)
\tkzGetPoint{H}
\tkzDefMidPoint(O,A) \tkzGetPoint{M}
\tkzInterLC(I,H)(M,A)\tkzGetPoints{B}{C}
\tkzDrawSegments[color=white,line width=1pt](I,H O,A)
\tkzDrawPoints[color=white](O,I,A,B,M)
\tkzMarkRightAngle[color=white,line width=1pt](A,I,B)
\tkzDrawArc[color=white,line width=1pt,style=dashed](M,A)(O)
\tkzLabelSegment[white,right=1ex,pos=.5](I,B){$\sqrt{a}$}
\tkzLabelSegment[white,below=1ex,pos=.5](O,I){$1$}
\tkzLabelSegment[pos=.6,white,below=1ex](I,A){$a$}
\end{tikzpicture}

30.9.3. Labels and option : swap

\begin{tikzpicture}[rotate=-60]
\tkzSetUpStyle[red,auto]{label style}
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDefSpcTriangle[orthic](A,B,C){N,O,P}
\tkzDefTriangleCenter[circum](A,B,C) \tkzGetPoint{O}
\tkzDrawPolygon[green!60!black](A,B,C)
\tkzDrawLine[dashed,color=magenta](C,P)
\tkzLabelSegment(B,A){$c$}
\tkzLabelSegment[swap](B,C){$a$}
\tkzLabelSegment[swap](C,A){$b$}
\tkzMarkAngles[size=1,color=cyan,mark=\|](C,B,A A,C,P)
\tkzMarkAngle[size=6.75,color=orange,mark=\|](P,C,B)
\tkzMarkAngle[size=6.75,color=orange,mark=\|](B,A,C)
\tkzMarkRightAngles[german](A,C,B B,P,C)
\tkzAutoLabelPoints[center = 0,dist = .1](A,B,C)
\tkzLabelPoint[below left](P){$P$}
\end{tikzpicture}
31. Add labels on a straight line \texttt{\textbackslash{}tkzLabelLine}

\begin{tikzpicture}[scale=1]
  \tkzDefPoints{0/0/O,2/2/A,4/0/B,6/2/C}
  \tkzDrawSegments(O,A,A,B)
  \tkzDrawPoints(O,A,B)
  \tkzDrawLine(O,B)
  \tkzLabelSegments[color=red,above=4pt](O,A,A,B){$a$}
\end{tikzpicture}

\tkzLabelLine{\langle pt1,pt2 \rangle}{\langle label \rangle}

\begin{tabular}{lll}
  arguments & default & definition \\
  \tkzLabelLine(A,B){\$\Delta\$} & & \\
\end{tabular}

\begin{tabular}{lll}
  options & default & definition \\
  pos & .5 & pos is an option for Ti\textsc{kZ}, but essential in this case… \\
  & & As an option, and in addition to the pos, you can use all styles of Ti\textsc{kZ}, especially the placement with above, right, … \\
\end{tabular}

31.0.1. Example with \texttt{\textbackslash{}tkzLabelLine}

An important option is pos, it’s the one that allows you to place the label along the right. The value of pos can be greater than 1 or negative.

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/0/B,1/1/C}
  \tkzDefLine[perpendicular=through C,K=-1](A,B)
  \tkzGetPoint{c}
  \tkzDrawLines(A,B,C,c)
  \tkzLabelLine[pos=1.25,blue,right](C,c){$(\delta)$}
  \tkzLabelLine[pos=-0.25,red,left](C,c){again $(\delta)$}
\end{tikzpicture}

31.1. Label at an angle : \texttt{\textbackslash{}tkzLabelAngle}

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/0/B,1/1/C}
  \tkzDefLine[perpendicular=through C,K=-1](A,B)
  \tkzGetPoint{c}
  \tkzDrawLines(A,B,C,c)
  \tkzLabelLine[pos=1.25,blue,right](C,c){$(\delta)$}
  \tkzLabelLine[pos=-0.25,red,left](C,c){again $(\delta)$}
\end{tikzpicture}

There is only one option, dist (with or without unit), which can be replaced by the TikZ’s pos option (without unit for the latter). By default, the value is in centimeters.
31. Add labels on a straight line \texttt{\textbackslash \tkzLabelLine}

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pos</td>
<td>1</td>
<td>or dist, controls the distance from the top to the label.</td>
</tr>
</tbody>
</table>

It is possible to move the label with all TikZ options: rotate, shift, below, etc.

31.1.1. Example

\begin{tikzpicture}[scale=.75]
\tkzDefPoint(0,0){C}
\tkzDefPoint(20:9){B}
\tkzDefPoint(80:5){A}
\tkzDefPointsBy[projection=onto B--C](A){a}
\tkzDrawPolygon[thick,fill=yellow!15](A,B,C)
\tkzDrawSegment[dashed, red](A,a)
\tkzDrawSegment[style=red, dashed, dim={$10$,15pt,midway,font=\scriptsize, rotate=90}](A,a)
\tkzMarkAngle(B,C,A)
\tkzMarkRightAngle(A,a,C)
\tkzMarkRightAngle(C,A,B)
\tkzFillAngle[fill=blue!20, opacity=0.5](B,C,A)
\tkzFillAngle[fill=red!20, opacity=0.5](A,B,C)
\tkzLabelAngle[pos=1.25](A,B,C){$\beta$}
\tkzLabelAngle[pos=1.25](B,C,A){$\alpha$}
\tkzLabelPoints(B,C)
\tkzLabelPoints[above](A)
\end{tikzpicture}

31.1.2. With pos

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/O,5/0/A,3/4/B}
\tkzMarkAngle[size = 4,mark = ||, arc=ll,color = red](A,O,B)
\tkzFillAngle[fill=blue!20, opacity=0.5](B,C,A)
\tkzFillAngle[fill=red!20, opacity=0.5](A,B,C)
\tkzLabelAngle[pos=1.25](A,B,C){$\beta$}
\tkzLabelAngle[pos=1.25](B,C,A){$\alpha$}
\tkzMarkAngle(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(B,C)
\tkzLabelPoints[above](A)
\end{tikzpicture}
31.3.1. pos and \tkzLabelAngles

\begin{tikzpicture}[rotate=30]
\tkzDefPoint(2,1){S}
\tkzDefPoint(7,3){T}
\tkzDefPointBy[rotation=center S angle 60](T)
\tkzGetPoint{P}
\tkzDefLine[bisector,normed](T,S,P)
\tkzGetPoint{s}
\tkzDrawPoints(S,T,P)
\tkzDrawPolygon{color=blue}(S,T,P)
\tkzDrawLine[dashed,color=blue,add=0 and 3](S,s)
\tkzLabelPoint[above right](P){$P$}
\tkzLabelPoints(S,T)
\tkzMarkAngle[size = 1.8,mark = |,arc=ll, color = blue](T,S,P)
\tkzMarkAngle[size = 2.1,mark = |,arc=1, color = blue](T,S,s)
\tkzMarkAngle[size = 2.3,mark = |,arc=1, color = blue](s,S,P)
\tkzLabelAngle[pos = 1.5](T,S,P){$60^\circ$}
\tkzLabelAngles[pos = 2.7](T,S,s,s,S,P){$30^\circ$
\end{tikzpicture}

\tkzLabelAngles[(local options)]((A,O,B)((A',O',B'))etc.

With common options, there is a macro for multiple angles.

It finally remains to be able to give a label to designate a circle and if several possibilities are offered, we will see here \tkzLabelCircle.

31.2. Giving a label to a circle

\tkzLabelCircle[(tikz options)]((0,0)(angle){(label)}

<table>
<thead>
<tr>
<th>options</th>
<th>default definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tikz options</td>
<td>circle O center through A</td>
</tr>
</tbody>
</table>

We can use the styles from TikZ. The label is created and therefore "passed" between braces.
31.2.1. Example

\begin{tikzpicture}
\tkzDefPoint(0,0){O} \tkzDefPoint(2,0){N}
\tkzDefPointBy[rotation=center O angle 50](N)
\tkzGetPoint{M}
\tkzDefPointBy[rotation=center O angle -20](N)
\tkzGetPoint{P}
\tkzDefPointBy[rotation=center O angle 125](N)
\tkzGetPoint{P'}
\tkzLabelCircle[above=4pt](O,N)(120){$\mathcal{C}$}
\tkzDrawCircle(O,M)
\tkzFillCircle[color=blue!10,opacity=.4](O,M)
\tkzLabelCircle[draw,
text width=2cm,text centered,left=24pt](O,M)(-120)
\tkzDrawPoints(M,P)
\tkzLabelPoints[right](M,P)
\end{tikzpicture}

32. Label for an arc

This macro allows you to place a label along an arc. The options are those of TikZ for example pos.

\begin{tabular}{|l|l|l|}
\hline
label & \tkzLabelArc(A,B){5} & label text \\
(pt1,pt2,pt3) & (O,A,B) & label along the arc $\widearc{AB}$ \\
\hline
options & default & definition \\
pos & .5 & label's position \\
\hline
\end{tabular}

32.0.1. Label on arc

\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\pgfmathsetmacro\r{2}
\tkzDefPoint(30:\r){A}
\tkzDefPoint(85:\r){B}
\tkzDrawCircle(O,A)
\tkzDrawPoints(B,A,O)
\tkzLabelArc[right=2pt](O,A,B){\widearc{AB}}
\tkzLabelPoints(A,B,O)
\end{tikzpicture}
Part VII.

Complements
33. Using the compass

33.1. Main macro \tkzCompass

This macro allows you to leave a compass trace, i.e. an arc at a designated point. The center must be indicated. Several specific options will modify the appearance of the arc as well as TikZ options such as style, color, line thickness etc.

You can define the length of the arc with the option \texttt{length} or the option \texttt{delta}.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{delta}</td>
<td>0 (deg)</td>
<td>Increases the angle of the arc symmetrically</td>
</tr>
<tr>
<td>\texttt{length}</td>
<td>1 (cm)</td>
<td>Changes the length (in cm)</td>
</tr>
</tbody>
</table>

33.1.1. Option \texttt{length}

\begin{tikzpicture}
\tkzDefPoint(1,1){A}
\tkzDefPoint(6,1){B}
\tkzInterCC[R](A,4)(B,3)
\tkzGetPoints{C}{D}
\tkzDrawPoint(C)
\tkzCompass[length=1.5](A,C)
\tkzCompass(B,C)
\tkzDrawSegments(A,B A,C B,C)
\end{tikzpicture}

33.1.2. Option \texttt{delta}

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,0){B}
\tkzInterCC[R](A,4)(B,3)
\tkzGetPoints{C}{D}
\tkzDrawPoints(A,B,C)
\tkzCompass[delta=20](A,C)
\tkzCompass[delta=20](B,C)
\tkzDrawPolygon(A,B,C)
\tkzMarkAngle(A,C,B)
\end{tikzpicture}

33.2. Multiple constructions \tkzCompasses

Attention the arguments are lists of two points. This saves a few lines of code.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{delta}</td>
<td>0</td>
<td>Modifies the angle of the arc by increasing it symmetrically</td>
</tr>
<tr>
<td>\texttt{length}</td>
<td>1</td>
<td>Changes the length</td>
</tr>
</tbody>
</table>
33.2.1. Use \tkzCompass

\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(2,2){A} \tkzDefPoint(5,-2){B} \tkzDefPoint(3,4){C} \tkzDrawPoints(A,B)
  \tkzDrawPoint[shape=cross out](C)
  \tkzCompass[new](A,B (A,C B,C C,B)
  \tkzShowLine[mediator,new,dashed,length = 2](A,B)
  \tkzShowLine[parallel = through C, color=purple,length=2](A,B)
  \tkzDefLine[mediator](A,B)
  \tkzPoints{i}{j}
  \tkzDefLine[parallel=through C](A,B)
  \tkzGetPoint{D}
  \tkzDrawLines(add=.6 and .6)(C,D A,C B,D)
  \tkzDrawLines(i,j) \tkzDrawPoints(A,B,C,i,j,D)
  \tkzLabelPoints(A,B,C,i,j,D)
\end{tikzpicture}

34. The Show

34.1. Show the constructions of some lines \tkzShowLine

\begin{tikzpicture}
  \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-1.5/2/C}
  \tkzDrawLine(A,B)
  \tkzDefLine[parallel=through C](A,B) \tkzGetPoint{c}
  \tkzShowLine[parallel=through C](A,B)
  \tkzDrawLine(C,c) \tkzDrawPoints(A,B,C,c)
\end{tikzpicture}

These constructions concern mediatries, perpendicular or parallel lines passing through a given point and bisectors. The arguments are therefore lists of two or three points. Several options allow the adjustment of the constructions. The idea of this macro comes from Yves Combe.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mediator</td>
<td>mediator</td>
<td>displays the constructions of a mediator</td>
</tr>
<tr>
<td>perpendicular</td>
<td>mediator</td>
<td>constructions for a perpendicular</td>
</tr>
<tr>
<td>orthogonal</td>
<td>mediator</td>
<td>idem</td>
</tr>
<tr>
<td>bisector</td>
<td>mediator</td>
<td>constructions for a bisector</td>
</tr>
<tr>
<td>K</td>
<td>1</td>
<td>circle within a triangle</td>
</tr>
<tr>
<td>length</td>
<td>1</td>
<td>in cm, length of a arc</td>
</tr>
<tr>
<td>ratio</td>
<td>.5</td>
<td>arc length ratio</td>
</tr>
<tr>
<td>gap</td>
<td>2</td>
<td>placing the point of construction</td>
</tr>
<tr>
<td>size</td>
<td>1</td>
<td>radius of an arc (see bisector)</td>
</tr>
</tbody>
</table>

You have to add, of course, all the styles of TikZ for tracings...

34.1.1. Example of \tkzShowLine and parallel

\begin{tikzpicture}
  \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-1.5/2/C}
  \tkzDrawLine(A,B)
  \tkzDefLine[parallel=through C](A,B) \tkzGetPoint{c}
  \tkzShowLine[parallel=through C](A,B)
  \tkzDrawLine(C,c) \tkzDrawPoints(A,B,C,c)
\end{tikzpicture}
34. The Show

34.1.2. Example of \texttt{\texttt{tkzShowLine} and perpendicular}

\begin{tikzpicture}
  \tkzDefPoints{0/0/A, 3/2/B, 2/2/C}
  \tkzDefLine[perpendicular=through C,K=-.5](A,B) \tkzGetPoint{c}
  \tkzShowLine[perpendicular=through C,K=-.5,gap=3](A,B)
  \tkzDefPointBy[projection=onto A--B](c)\tkzGetPoint{h}
  \tkzMarkRightAngle[fill=lightgray](A,h,C)
  \tkzDrawLines[add=.5 and .5](A,B C,c)
  \tkzDrawPoints(A,B,C,h,c)
\end{tikzpicture}

34.1.3. Example of \texttt{\texttt{tkzShowLine} and bisector}

\begin{tikzpicture}[scale=1.25]
  \tkzDefPoints{0/0/A, 4/2/B, 1/4/C}
  \tkzDrawPolygon(A,B,C)
  \tkzSetUpCompass[color=brown,line width=.1 pt]
  \tkzDefLine[bisector](B,A,C) \tkzGetPoint{a}
  \tkzDefLine[bisector](C,B,A) \tkzGetPoint{b}
  \tkzInterLL(A,a)(B,b) \tkzGetPoint{I}
  \tkzDefPointBy[projection = onto A--B](I)
  \tkzGetPoint{H}
  \tkzShowLine[bisector,size=2,gap=3,blue](B,A,C)
  \tkzShowLine[bisector,size=2,gap=3,blue](C,B,A)
  \tkzDrawCircle[color=blue,line width=.2pt](I,H)
  \tkzDrawSegments[color=red!50](I,tkzPointResult)
  \tkzDrawLines[add=0 and -.3,color=red!50](A,a B,b)
\end{tikzpicture}

34.1.4. Example of \texttt{\texttt{tkzShowLine} and mediator}

\begin{tikzpicture}
  \tkzDefPoint(2,2){A}
  \tkzDefPoint(5,4){B}
  \tkzDrawPoints(A,B)
  \tkzShowLine[mediator,color=orange,length=1](A,B)
  \tkzGetPoints{i}{j}
  \tkzDrawLines[i,j]
  \tkzDrawLines(A,B)
  \tkzLabelPoints[below =3pt](A,B)
\end{tikzpicture}

34.2. Constructions of certain transformations \texttt{\texttt{tkzShowTransformation}}

\texttt{\texttt{tkzShowTransformation}[(\texttt{local options})](\texttt{pt1,pt2}) or (\texttt{pt1,pt2,pt3})}

These constructions concern orthogonal symmetries, central symmetries, orthogonal projections and translations. Several options allow the adjustment of the constructions. The idea of this macro comes from Yves Combe.
34. The Show

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>reflection= over pt1--pt2</td>
<td>reflection</td>
<td>constructions of orthogonal symmetry</td>
</tr>
<tr>
<td>symmetry=center pt</td>
<td>reflection</td>
<td>constructions of central symmetry</td>
</tr>
<tr>
<td>projection=onto pt1--pt2</td>
<td>reflection</td>
<td>constructions of a projection</td>
</tr>
<tr>
<td>translation=from pt1 to pt2</td>
<td>reflection</td>
<td>constructions of a translation</td>
</tr>
<tr>
<td>K</td>
<td>1</td>
<td>circle within a triangle</td>
</tr>
<tr>
<td>length</td>
<td>1</td>
<td>arc length</td>
</tr>
<tr>
<td>ratio</td>
<td>.5</td>
<td>arc length ratio</td>
</tr>
<tr>
<td>gap</td>
<td>2</td>
<td>placing the point of construction</td>
</tr>
<tr>
<td>size</td>
<td>1</td>
<td>radius of an arc (see bisector)</td>
</tr>
</tbody>
</table>

34.2.1. Example of the use of \tkzShowTransformation

\begin{tikzpicture}[scale=.6]
\tkzDefPoint(0,0){O} \tkzDefPoint(2,-2){A}
\tkzDefPoint(70:4){B} \tkzDrawPoints(A,O,B)
\tkzLabelPoints(A,O,B)
\tkzDrawLine[add= 2 and 2](O,A)
\tkzDefPointBy[translation=from O to A](B)
\tkzGetPoint{C}
\tkzDrawPoint[color=orange](C) \tkzLabelPoints(C)
\tkzShowTransformation[translation=from O to A,color=orange,size=2](B)
\tkzDefPointBy[reflection=over O--A](B) \tkzGetPoint(E)
\tkzDrawSegment[blue](B,E)
\tkzDefPointBy[reflection=over O--A](B)
\tkzGetPoint{F}
\tkzDrawSegment[green](B,F)
\tkzShowTransformation[reflection=over O--A,color=green,size=2,gap=-2](F)
\tkzDefPointBy[projection=onto O--A](C)
\tkzGetPoint(H)
\tkzDrawSegments[color=magenta](C,H)
\tkzDrawPoint[color=magenta](H) \tkzLabelPoints(H)
\tkzShowTransformation[projection=onto O--A,color=red,size=3,gap=-2](C)
\end{tikzpicture}

34.2.2. Another example of the use of \tkzShowTransformation

You’ll find this figure again, but without the construction features.
35. Protractor

Based on an idea by Yves Combe, the following macro allows you to draw a protractor. The operating principle is even simpler. Just name a half-line (a ray). The protractor will be placed on the origin O, the direction of the half-line is given by A. The angle is measured in the direct direction of the trigonometric circle.

35.1. The macro \tkzProtractor

\begin{tkzpicture}[scale=.5]
  \tkzDefPoint(2,0){A}\tkzDefShiftPoint[A](31:5){B}
  \tkzDefShiftPoint[A](158:5){C}
  \tkzDrawPoints(A,B,C)
  \tkzDrawSegments[color = red, line width = 1pt](A,B,A,C)
  \tkzProtractor[color = red, size = 3, gap = -3](I)
  \tkzDrawPoints[color = red](M,N)
  \tkzDrawPoints[color = blue](O,A,B,I,M)
  \tkzLabelPoints(O)
  \tkzLabelPoints[above right](N,I)
  \tkzLabelPoints[below left](M,A)
\end{tkzpicture}

35.1.1. The circular protractor

Measuring in the forward direction

\begin{tkzpicture}[scale=.6]
  \tkzDefPoints{0/0/A,8/0/B,3.5/10/I}
  \tkzDefMidPoint(A,B) \tkzGetPoint{O}
  \tkzDefPointBy[projection=onto A--B](I)
  \tkzGetPoint{J}
  \tkzInterLC(I,A)(O,A) \tkzGetPoints{M}{M'}
  \tkzInterLC(I,B)(O,A) \tkzGetPoints{N}{N'}
  \tkzDefMidPoint(A,B) \tkzGetPoint{M}
  \tkzDrawSemiCircle(M,B)
  \tkzDrawSegments(I,A I,B A,B B,M A,N)
  \tkzMarkRightAngles(A,M,B A,N,B)
  \tkzDrawSegment[style=dashed,color=blue](I,J)
  \tkzShowTransformation[projection=onto A--B, color=red, size=3, gap=-3](I)
  \tkzDrawPoints[color = red](M,N)
  \tkzDrawPoints[color = blue](O,A,B,I,M)
  \tkzLabelPoints(O)
  \tkzLabelPoints[above right](N,I)
  \tkzLabelPoints[below left](M,A)
\end{tkzpicture}
35.1.2. The circular protractor, transparent and returned

\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(2,3){A}
  \tkzDefShiftPoint[A](31:5){B}
  \tkzDefShiftPoint[A](158:5){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzProtractor[return](A,C)
\end{tikzpicture}

36. Miscellaneous tools and mathematical tools

36.1. Duplicate a segment

This involves constructing a segment on a given half-line of the same length as a given segment.

\begin{verbatim}
\tkzDuplicateSegment((pt1,pt2),(pt3,pt4),(pt5))
\end{verbatim}

This involves creating a segment on a given half-line of the same length as a given segment. It is in fact the definition of a point. \texttt{\tkzDuplicateSegment} is the new name of \texttt{\tkzDuplicateLen}.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)(pt3,pt4){pt5}</td>
<td>\tkzDuplicateSegment(A,B)(E,F){C}</td>
<td>AC=EF and C ∈ [AB)</td>
</tr>
</tbody>
</table>

The macro \texttt{\tkzDuplicateLength} is identical to this one.

36.1.1. Use of \texttt{\tkzDuplicateSegment}

\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/A,2/-3/B,2/5/C}
  \tkzDuplicateSegment(A,B)(A,C)
  \tkzGetPoint{D}
  \tkzDrawSegments[new](A,B A,C)
  \tkzDrawSegment[teal](A,D)
  \tkzDrawPoints[new](A,B,C,D)
  \tkzLabelPoints[above right=3pt](A,B,C,D)
\end{tikzpicture}
36. Miscellaneous tools and mathematical tools

36.1.2. Proportion of gold with \tkzDuplicateSegment

\begin{tikzpicture}[rotate=-90,scale=.4]
\tkzDefPoints{0/0/A,10/0/B}
\tkzDefMidPoint(A,B)
\tkzGetPoint(I)
\tkzDefPointWith[orthogonal,K=-.75](B,A)
\tkzGetPoint(C)
\tkzInterLC(B,C)(B,I) \tkzGetSecondPoint{D}
\tkzDuplicateSegment(B,D)(D,A) \tkzGetPoint{E}
\tkzInterLC(A,B)(A,E) \tkzGetPoints{N}{M}
\tkzDrawArc[orange,delta=10](D,E)(B)
\tkzDrawArc[orange,delta=10](A,M)(E)
\tkzDrawLines(A,B B,C A,D)
\tkzDuplicateSegment(B,D)(D,A) \tkzGetPoint{E}
\tkzInterLC(A,B)(A,E) \tkzGetPoints{N}{M}
\tkzDrawArc[orange,delta=10](D,E)(B)
\tkzDrawArc[orange,delta=10](A,M)(E)
\tkzDrawLines(A,B B,C A,D)
\tkzGetLength(A,B) \dAB
\end{tikzpicture}

36.1.3. Golden triangle or sublime triangle

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,5/0/C,0/5/B}
\tkzDefMidPoint(A,C)\tkzGetPoint{H}
\tkzDuplicateSegment(H,B)(H,A)\tkzGetPoint{D}
\tkzDuplicateSegment(A,D)(A,B)\tkzGetPoint{E}
\tkzDuplicateSegment(A,D)(B,A)\tkzGetPoint{G}
\tkzInterCC(A,C)(B,G)\tkzGetSecondPoint{F}
\tkzDrawLine(A,C)
\tkzDrawArc(A,C)(B)
\tkzDrawSegment[dashed](H,B)
\tkzDrawArc(H,B)(D)
\tkzCompass(B,F)
\tkzDrawPolygon[new](A,B,F)
\tkzDrawPoints(A,...,H)
\tkzLabelPoints{below left}(A,...,H)
\end{tikzpicture}

36.2. Segment length \tkzCalcLength

There's an option in TikZ named \texttt{veclen}. This option is used to calculate AB if A and B are two points.
The only problem for me is that the version of TikZ is not accurate enough in some cases. My version uses the \texttt{xfp} package and is slower, but more accurate.

\begin{verbatim}
\tkzCalcLength[(local options)]((pt1,pt2))
\end{verbatim}

You can store the result with the macro \texttt{tkzGetLength} for example \texttt{tkzGetLength(dAB)} defines the macro \texttt{dAB}.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)</td>
<td>\tkzCalcLength(A,B)</td>
<td>\texttt{dAB} gives AB in cm</td>
</tr>
</tbody>
</table>
36. Miscellaneous tools and mathematical tools

Only one option

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
</table>
| cm      | true    | \(\text{tkzCalcLength}(A,B)\) After \(\text{tkzGetLength}(dAB)\) \(dAB\) gives \(AB\) in cm

36.2.1. Compass square construction

\begin{tikzpicture}[scale=1]
\tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
\tkzCalcLength(A,B) \tkzGetLength{dAB}
\tkzDefLine[perpendicular=through A](A,B)
\tkzGetPoint{D}
\tkzDefPointWith[orthogonal,K=-1](B,A)
\tkzGetPoint{F}
\tkzGetPoint{C}
\tkzDrawLine[add=.6 and .2](A,B)
\tkzDrawLine(A,D)
\tkzShowLine[orthogonal=through A,gap=2](A,B)
\tkzMarkRightAngle(B,A,D)
\tkzCompasss(A,D,D,C)
\tkzDrawArc[R](B,\dAB)(80,110)
\tkzDrawPoints(A,B,C,D)
\tkzDrawSegments[color=gray,style=dashed](B,C,C,D)
\tkzLabelPoints[below left](A,B,C,D)
\end{tikzpicture}

36.2.2. Example

The macro \(\text{tkzDefCircle}[\text{radius}](A,B)\) defines the radius that we retrieve with \(\text{tkzGetLength}\), this result is in cm.

\begin{tikzpicture}[scale=.5]
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,-4){B}
\tkzDefMidPoint(A,B) \tkzGetPoint{M}
\tkzCalcLength(M,B) \tkzGetLength{rAB}
\tkzDrawCircle(A,B)
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
\tkzDrawSegment[dashed](A,B){\pgfmathprintnumber{\rAB}}
\end{tikzpicture}

36.3. Transformation from pt to cm or cm to pt

Not sure if this is necessary and it is only a division by 28.45274 and a multiplication by the same number. The macros are:

\texttt{\text{tkzpttocom}(number){name of macro}}

The result is stored in a macro.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(number){name of macro} \texttt{\text{tkzpttocom}(128){len}}</td>
<td>len gives a number of \texttt{tkznamecm}</td>
<td></td>
</tr>
</tbody>
</table>

\texttt{tkz-euclide} AlterMundus
36. Miscellaneous tools and mathematical tools

You’ll have to use \len along with cm.

36.4. Change of unit

\tkzcmtopt\{number\}\{name of macro\}

The result is stored in a macro.

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(number){name of macro}</td>
<td>\tkzcmtopt(5){len}</td>
<td>\len length in pts</td>
</tr>
</tbody>
</table>

The result can be used with \len pt

36.5. Get point coordinates

\tkzGetPointCoord(\{A\})\{name of macro\}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(point){name of macro}</td>
<td>\tkzGetPointCoord(A){A}</td>
<td>\Ax and \Ay give coordinates for A</td>
</tr>
</tbody>
</table>

Stores in two macros the coordinates of a point. If the name of the macro is p, then \px and \py give the coordinates of the chosen point with the cm as unit.

36.5.1. Coordinate transfer with \tkzGetPointCoord

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=3]
\tkzGrid[sub,orange]
\tkzDrawX \tkzDrawY
\tkzDefPoint(1,0){A}
\tkzDefPoint(4,2){B}
\tkzGetPointCoord(A){a}
\tkzGetPointCoord(B){b}
\tkzDefPoint(\ax,\ay){C}
\tkzDefPoint(\bx,\by){D}
\tkzDrawPoints[red](C,D)
\end{tikzpicture}

36.5.2. Sum of vectors with \tkzGetPointCoord

\begin{tikzpicture}[>=latex]
\tkzDefPoint(1,4){a}
\tkzDefPoint(3,2){b}
\tkzDefPoint(1,1){c}
\tkzDrawSegment[->,red](a,b)
\tkzGetPointCoord(c){c}
\draw[blue,->](a) -- ([shift=(b)]\cx,\cy) ;
\draw[purple,->](b) -- ([shift=(b)]\cx,\cy) ;
\tkzDrawSegment[->,blue](a,c)
\tkzDrawSegment[->,purple](b,c)
\end{tikzpicture}
36.6. Swap labels of points

\[ \text{\texttt{\textbackslash tkzSwapPoints\{pt1,pt2\}}} \]

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)</td>
<td>\texttt{\textbackslash tkzSwapPoints{A,B}} now A has the coordinates of B</td>
<td>\textit{The points have exchanged their coordinates.}</td>
</tr>
</tbody>
</table>

36.6.1. Use of \texttt{\textbackslash tkzSwapPoints}

\begin{tikzpicture}
\tkzDefPoints{0/0/O,5/-1/A,2/2/B} \\
\tkzSwapPoints(A,B) \\
\tkzDrawPoints(O,A,B) \\
\tkzLabelPoints(O,A,B)
\end{tikzpicture}

36.7. Dot Product

In Euclidean geometry, the dot product of the Cartesian coordinates of two vectors is widely used.

\[ \text{\texttt{\textbackslash tkzDotProduct\{pt1,pt2,pt3\}}} \]

The dot product of two vectors \( \vec{u} = [a, b] \) and \( \vec{v} = [a', b'] \) is defined as: \( \vec{u} \cdot \vec{v} = aa' + bb' \)

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2,pt3)</td>
<td>\texttt{\textbackslash tkzDotProduct{A,B,C}} the result is ( \overrightarrow{AB} \cdot \overrightarrow{AC} )</td>
<td>\textit{The result is a number that can be retrieved with \texttt{\textbackslash tkzGetResult}.}</td>
</tr>
</tbody>
</table>
36.7.1. Simple example

\begin{tikzpicture}
\tkzDefPoints{-2/-3/A,4/0/B,1/3/C}
\tkzDefPointBy[projection= onto A--B](C)
\tkzGetPoint{H}
\tkzDrawSegment(C,H)
\tkzMarkRightAngle(C,H,A)
\tkzDrawSegments[vector style](A,B,A,C)
\tkzDrawPoints(A,H) \tkzLabelPoints(A,B,H)
\tkzLabelPoints[above](C)
\tkzDotProduct(A,B,C) \tkzGetResult{pabc}
% \pgfmathparse{round(10*pabc)/10}
\let\pabc\pgfmathresult
\node at (1,-3) {$\overrightarrow{PA}\cdot \overrightarrow{PB} = \pabc$};
\tkzDotProduct(A,H,B) \tkzGetResult{phab}
% \pgfmathparse{round(10*phab)/10}
\let\phab\pgfmathresult
\node at (1,-4) {$PA \times PH = \phab$};
\end{tikzpicture}

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{1/2/O,5/2/B,2/2/P,3/3/Q}
\tkzInterLC[common=B](O,B)(O,B) \tkzGetFirstPoint{A}
\tkzInterLC[common=B](P,Q)(O,B) \tkzGetPoints{C}{D}
\tkzDrawCircle(O,B)
\tkzDrawSegments(A,B C,D)
\tkzDrawPoints(A,B,C,D,P)
\tkzLabelPoints(P)
\tkzLabelPoints[below left](A,C)
\tkzLabelPoints[above right](B,D)
\tkzDotProduct(P,A,B) \tkzGetResult{pab}
\pgfmathparse{round(10*pab)/10}
\let\pab\pgfmathresult
\tkzDotProduct(P,C,D) \tkzGetResult{pcd}
\pgfmathparse{round(10*pcd)/10}
\let\pcd\pgfmathresult
\node at (1,-3) {$\overrightarrow{PA}\cdot \overrightarrow{PB} = \overrightarrow{PC}\cdot \overrightarrow{PD}$};
\node at (1,-4) {$\overrightarrow{PA}\cdot \overrightarrow{PB} = -15.0$};
\node at (1,-5) {$\overrightarrow{PC}\cdot \overrightarrow{PD} = -15.0$};
\end{tikzpicture}

36.7.2. Cocyclic points
36.8. Power of a point with respect to a circle

\begin{itemize}
  \item \texttt{\tkzPowerCircle\((pt1)\)(pt2,pt3)}
\end{itemize}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1)(pt2,pt3)</td>
<td>\tkzPowerCircle((A))(((0,M)))</td>
<td>power of A with respect to the circle ((0,A)) \textit{The result is a number that represents the power of a point with respect to a circle.}</td>
</tr>
</tbody>
</table>

36.8.1. Power from the radical axis

In this example, the radical axis \((EF)\) has been drawn. A point \(H\) has been chosen on \((EF)\) and the power of the point \(H\) with respect to the circle of center \(A\) has been calculated as well as \(PS^2\). You can check that the power of \(H\) with respect to the circle of center \(C\) as well as \(HS^2, HT, HT^2\) give the same result.

\begin{tikzpicture}[scale=.5]
\tkzDefPoints{-1/0/A,0/5/B,5/-1/C,7/1/D}
\tkzDrawCircles(A,B,C,D)
\tkzDefRadicalAxis(A,B)(C,D) \tkzGetPoints{E}{F}
\tkzDrawLine[add=1 and 2](E,F)
\tkzDefPointOnLine[pos=1.5](E,F) \tkzGetPoint{H}
\tkzDefLine[tangent from = H](A,B) \tkzGetPoints{T,T'}
\tkzDefLine[tangent from = H](C,D) \tkzGetPoints{S,S'}
\tkzDrawSegments(H,T,H,T',H,S,H,S')
\tkzDrawPoints(A,B,C,D,E,F,H,T,T',S,S')
\tkzPowerCircle(H)(A,B) \tkzGetResult{pw}
\tkzDotProduct(H,S,S) \tkzGetResult{phtt}
\node {Power \approx \pw \approx \phtt};
\end{tikzpicture}

\begin{itemize}
  \item \texttt{\tkzDefRadicalAxis\((pt1,pt2)\)(pt3,pt4)}
\end{itemize}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2)(pt3,pt4)</td>
<td>\tkzDefRadicalAxis((A,B))(C,D)</td>
<td>Two circles with centers A and C \textit{The result is two points of the radical axis.}</td>
</tr>
</tbody>
</table>

36.9. Radical axis

In geometry, the radical axis of two non-concentric circles is the set of points whose power with respect to the circles are equal. Here \texttt{\tkzDefRadicalAxis\((A,B)\)(C,D)} gives the radical axis of the two circles \(\mathcal{C}(A,B)\) and \(\mathcal{C}(C,D)\).
36.9.1. Two circles disjointed

\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{-1/0/A,0/2/B,4/-1/C,4/0/D}
  \tkzDrawCircles(A,B,C,D)
  \tkzDefRadicalAxis(A,B)(C,D)
  \tkzGetPoints{E}{F}
  \tkzDrawLine[add=1 and 2](E,F)
  \tkzDrawLine[add=.5 and .5](A,C)
\end{tikzpicture}

36.10. Two intersecting circles

\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{-1/0/A,0/2/B,3/-1/C,3/-2/D}
  \tkzDrawCircles(A,C,B,D)
  \tkzDefRadicalAxis(A,C)(B,D)
  \tkzGetPoints{E}{F}
  \tkzDrawPoints(A,B,C,D,E,F)
  \tkzLabelPoints(A,B,C,D,E,F)
  \tkzDrawLine[add=.5 and 1](E,F)
  \tkzDrawLine[add=.25 and .25](A,B)
\end{tikzpicture}

36.11. Two externally tangent circles

\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/A,4/0/B,6/0/C}
  \tkzDrawCircles(A,B,C,B)
  \tkzDefRadicalAxis(A,B)(C,B)
  \tkzGetPoints{E}{F}
  \tkzDrawPoints(A,B,C,D,E,F)
  \tkzLabelPoints(A,B,C,E,F)
  \tkzDrawLine[add=1 and 1](E,F)
  \tkzDrawLine[add=.5 and .5](A,B)
\end{tikzpicture}
36. Miscellaneous tools and mathematical tools

36.12. Two circles tangent internally

\begin{tikzpicture} [scale=.5]
\tkzDefPoints{0/0/A,3/0/B,5/0/C}
\tkzDrawCircles(A,C,B,C)
\tkzDefRadicalAxis(A,C)(B,C)
\tkzGetPoints{E}{F}
\tkzDrawPoints(A,B,C,E,F)
\tkzLabelPoints[below right](A,B,C,E,F)
\tkzDrawLine[add=1 and 1](E,F)
\tkzDrawLine[add=.5 and .5](A,B)
\end{tikzpicture}

36.12.1. Three circles

\begin{tikzpicture} [scale=.4]
\tkzDefPoints{0/0/A,5/0/a,7/-1/B,3/-1/b,5/-4/C,2/-4/c}
\tkzDrawCircles(A,a,B,b,C,c)
\tkzDefRadicalAxis(A,a)(B,b) \tkzGetPoints{i}{j}
\tkzDefRadicalAxis(A,a)(C,c) \tkzGetPoints{k}{l}
\tkzDefRadicalAxis(C,c)(B,b) \tkzGetPoints{m}{n}
\tkzDrawLines[new](i,j,k,l,m,n)
\end{tikzpicture}

36.13. \texttt{\tkzIsLinear}, \texttt{\tkzIsOrtho}

\begin{verbatim}
\tkzIsLinear((pt1,pt2,pt3))
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2,pt3)</td>
<td>\tkzIsLinear(A,B,C)</td>
<td>A,B,C aligned?</td>
</tr>
</tbody>
</table>
\tkzIsLinear allows to test the alignment of the three points pt1,pt2,pt3.

\begin{verbatim}
\tkzIsOrtho((pt1,pt2,pt3))
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pt1,pt2,pt3)</td>
<td>\tkzIsOrtho(A,B,C)</td>
<td>(AB) \perp (AC)?</td>
</tr>
</tbody>
</table>
\tkzIsOrtho allows to test the orthogonality of lines (pt1pt2) and (pt1pt3).
36.13.1. Use of \texttt{\tkzIsOrtho} and \texttt{\tkzIsLinear}

\begin{tikzpicture}
tkzDefPoints{1/-2/A,5/0/B}
tkzDefCircle[diameter](A,B) \tkzGetPoint{O}
tkzDrawCircle(O,A)
tkzDefPointBy[rotation= center O angle 60](B) \tkzGetPoint{C}
tkzDefPointBy[rotation= center O angle 60](A) \tkzGetPoint{D}
tkzDrawCircle(O,A)
tkzDrawPoints(A,B,C,D)
tkzIsOrtho(C,A,B)
\iftkzOrtho
  tkzDrawPolygon[blue](A,B,C)
tkzDrawPoints[blue](A,B,C,D)
\else
  tkzDrawPoints[red](A,B,C,D)
\fi
\tkzIsLinear(O,C,D)
\iftkzLinear
  tkzDrawSegment[orange](C,D)
\fi
\end{tikzpicture}
Part VIII.

Working with style
37. Predefined styles

The way to proceed will depend on your use of the package. A method that seems to me to be correct is to use as much as possible predefined styles in order to separate the content from the form. This method will be the right one if you plan to create a document (like this documentation) with many figures. We will see how to define a global style for a document. We will see how to use a style locally.

The file `tkz-euclide.cfg` contains the predefined styles of the main objects. Among these the most important are points, lines, segments, circles, arcs and compass traces. If you always use the same styles and if you create many figures then it is interesting to create your own styles. To do this you need to know what features you can modify. It will be necessary to know some notions of TikZ.

The predefined styles are global styles. They exist before the creation of the figures. It is better to avoid changing them between two figures. On the other hand these styles can be modified in a figure temporarily. There the styles are defined locally and do not influence the other figures.

For the document you are reading here is how I defined the different styles.

\begin{verbatim}
\tkzSetUpColors[background=white,text=black]
\tkzSetUpPoint[size=2,color=teal]
\tkzSetUpLine[line width=.4pt,color=teal]
\tkzSetUpCompass[color=orange, line width=.4pt,delta=10]
\tkzSetUpArc[color=gray,line width=.4pt]
\tkzSetUpStyle[orange]{new}
\end{verbatim}

The macro `\tkzSetUpColors` allows you to set the background color as well as the text color. If you don’t use it, the colors of your document will be used as well as the fonts. Let’s see how to define the styles of the main objects.

38. Points style

This is how the points are defined:

\begin{verbatim}
\tikzset{point style/.style = {%
    draw = \tkz@euc@pointcolor,
    inner sep = \tkz@euc@pointsize,
    shape = \tkz@euc@pointshape,
    minimum size = \tkz@euc@pointsize,
    fill = \tkz@euc@pointcolor}}
\end{verbatim}

It is of course possible to use `\tikzset` but you can use a macro provided by the package. You can use the macro `\tkzSetUpPoint` globally or locally, let’s look at this possibility.

38.1. Use of `\tkzSetUpPoint`

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>point color</td>
</tr>
<tr>
<td>size</td>
<td>3</td>
<td>point size</td>
</tr>
<tr>
<td>fill</td>
<td>black!50</td>
<td>inside point color</td>
</tr>
<tr>
<td>shape</td>
<td>circle</td>
<td>point shape circle, cross or cross out</td>
</tr>
</tbody>
</table>

tkz-euclide AlterMundus
38. Points style

38.1.1. Global style or local style

First of all here is a figure created with the styles of my documentation, then the style of the points is modified within the environment \texttt{tikzpicture}.

You can use the macro \texttt{\tkzSetUpPoint} globally or locally. If you place this macro in your preamble or before your first figure then the point style will be valid for all figures in your document. It will be possible to use another style locally by using this command within an environment \texttt{tikzpicture}.

Let’s look at this possibility.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,5/0/B,3/2/C,3/1/D}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above right](C)
\end{tikzpicture}

38.1.2. Local style

The style of the points is modified locally in the second figure

\begin{tikzpicture}
\tkzSetUpPoint[size=4,color=red,fill=red!20]
\tkzDefPoints{0/0/A,5/0/B,3/2/C,3/1/D}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzDrawPoint[shape=cross out,thick](D)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above right](C)
\end{tikzpicture}

38.1.3. Style and scope

The points get back the initial style. Point D has a new style limited by the environment \texttt{scope}. It is also possible to use \{\ldots\} or \texttt{\begingroup \ldots \endgroup}. The points get back the initial style. Point D has a new style limited by the environment \texttt{scope}. It is also possible to use \{\ldots\} or \texttt{\begingroup \ldots \endgroup}.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,5/0/B,3/2/C,3/1/D}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\begin{scope}
\tkzSetUpPoint[size=4,color=red,fill=red!20]
\tkzDrawPoint(D)
\end{scope}
\tkzLabelPoints(A,B)
\tkzLabelPoints[above right](C)
\end{tikzpicture}

38.1.4. Simple example with \texttt{\tkzSetUpPoint}

\begin{tikzpicture}
\tkzSetUpPoint[shape = cross out, color=blue]
\tkzDefPoint(2,1){A}
\tkzDefPoint(4,0){B}
\tkzDrawLine(A,B)
\tkzDrawPoints(A,B)
\end{tikzpicture}
38.1.5. Use of `\tkzSetUpPoint` inside a group

\begin{tikzpicture}
\tkzDefPoints{0/0/A,2/4/B,4/0/C,3/2/D}
\tkzDrawSegments(A,B A,C A,D)
\tkzSetUpPoint[shape=cross out,
  fill= teal!50,
  size=4, color=teal]
\tkzDrawPoints(A,B)
\tkzSetUpPoint[fill= teal!50, size=4, color=teal]
\tkzDrawPoints(C,D)
\tkzLabelPoints(A,B,C,D)
\end{tikzpicture}

39. Lines style

You have several possibilities to change the style of a line. You can modify the style of a line with `\tkzSetUpLine` or directly modify the style of the lines with `\tikzset{line style/.style = ... }`

Reminder about line width: There are a number of predefined styles that provide more “natural” ways of setting the line width. You can also redefine these styles.

<table>
<thead>
<tr>
<th>predefined style</th>
<th>value of line width</th>
</tr>
</thead>
<tbody>
<tr>
<td>ultra thin</td>
<td>0.1 pt</td>
</tr>
<tr>
<td>very thin</td>
<td>0.2 pt</td>
</tr>
<tr>
<td>thin</td>
<td>0.4 pt</td>
</tr>
<tr>
<td>semithick</td>
<td>0.6 pt</td>
</tr>
<tr>
<td>thick</td>
<td>0.8 pt</td>
</tr>
<tr>
<td>very thick</td>
<td>1.2 pt</td>
</tr>
<tr>
<td>ultra thick</td>
<td>1.6 pt</td>
</tr>
</tbody>
</table>

39.1. Use of `\tkzSetUpLine`

It is a macro that allows you to define the style of all the lines.

\begin{center}
\begin{tabular}{lll}
<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>colour of the construction lines</td>
</tr>
<tr>
<td>line width</td>
<td>0.4pt</td>
<td>thickness of the construction lines</td>
</tr>
<tr>
<td>style</td>
<td>solid</td>
<td>style of construction lines</td>
</tr>
<tr>
<td>add</td>
<td>.2 and .2</td>
<td>changing the length of a line segment</td>
</tr>
</tbody>
</table>
\end{tabular}
\end{center}
39. Lines style

39.1.1. Change line width

\begin{tikzpicture}[scale=.75]
  \tkzSetUpLine[line width=1pt]
  \begin{scope}[rotate=90]
    \tkzDefPoints{0/6/A,10/0/B,10/6/C}
    \tkzDefPointBy[projection = onto B--A](C)
    \tkzGetPoint{H}
    \tkzMarkRightAngle[size=.4, fill=teal!20](B,C,A)
    \tkzMarkRightAngle[size=.4, fill=orange!20](B,H,C)
    \tkzDrawPolygon(A,B,C)
    \tkzDrawSegment[new](C,H)
    \tkzLabelSegment[below](C,B){$a$}
    \tkzLabelSegment[right](A,C){$b$}
    \tkzLabelSegment[left](A,B){$c$}
    \tkzLabelSegment[color=red](C,H){$h$}
    \tkzDrawPoints(A,B,C)
    \tkzLabelPoints[above left](H)
    \tkzLabelPoints(B,C)
    \tkzLabelPoints[above](A)
  \end{scope}
\end{tikzpicture}

39.1.2. Change style of line

\begin{tikzpicture}[scale=.5]
  \tikzset{line style/.style = {color = gray, style=dashed}}
  \tkzDefPoints{1/0/A,4/0/B,1/1/C,5/1/D}
  \tkzDefPoints{1/2/E,6/2/F,0/4/A',3/4/B'}
  \tkzCalcLength(C,D)
  \tkzGetLength{rCD}
  \tkzCalcLength(E,F)
  \tkzGetLength{rEF}
  \tkzInterCC[R](A',\rCD)(B',\rEF)
  \tkzGetPoints{I}{J}
  \tkzDrawLine(A',B')
  \tkzCompass(A',B')
  \tkzDrawSegments(A,B,C,D,E,F)
  \tkzDefCircle[R](A',\rCD)(B',\rEF)\tkzGetPoint{a'}
  \tkzDrawCircles(A',a' B',b')
  \begin{scope}
    \tkzSetUpLine[color=red]
    \tkzDrawSegments(A',I B',I)
  \end{scope}
  \tkzDrawPoints(A,B,C,D,E,F,A',B',I,J)
  \tkzLabelPoints(A,B,C,D,E,F,A',B',I,J)
\end{tikzpicture}
39.1.3. Example 3: extend lines

```
\begin{tikzpicture}[scale=.75]
\tkzSetUpLine[add=.5 and .5]
\tkzDefPoints{0/0/A,4/0/B,1/3/C}
\tkzDrawLines(A,B B,C A,C)
\tkzDrawPolygon[red,thick](A,B,C)
\tkzSetUpPoint[size=4,circle,color=red,fill=red!20]
\tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

40. Arc style

40.1. The macro \tkzSetUpArc

```
\tkzSetUpArc[⟨local options⟩]
```

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>colour of the lines</td>
</tr>
<tr>
<td>line width</td>
<td>0.4pt</td>
<td>thickness of the lines</td>
</tr>
<tr>
<td>style</td>
<td>solid</td>
<td>style of construction lines</td>
</tr>
</tbody>
</table>

40.1.1. Use of \tkzSetUpArc

```
\begin{tikzpicture}
\def\r{3} \def\angle{200}
\tkzSetUpArc[delta=10,color=purple,line width=.2pt]
\tkzSetUpLabel[font=\scriptsize,red]
\tkzDefPoint(0,0){O}
\tkzDefPoint(\angle:\r){A}
\tkzInterCC(O,A)(A,O) \tkzGetPoints{C'}{C}
\tkzInterCC(O,A)(C,O) \tkzGetPoints{D'}{D}
\tkzInterCC(O,A)(D,O) \tkzGetPoints{X'}{X}
\tkzDrawCircle(O,A)
\tkzDrawArc(A,C')(C)
\tkzDrawArc(C,O)(D)
\tkzDrawArc(D,O)(X)
\tkzDrawLine[add=.1 and .1](A,X)
\tkzDrawPoints(O,A)
\tkzSetUpPoint[size=3,color=purple,fill=purple!10]
\tkzDrawPoints(C,C',D,X)
\tkzLabelPoints[below left](O,A)
\tkzLabelPoints[below](C')
\tkzLabelPoints[below right](X)
\tkzLabelPoints[above](C,D)
\end{tikzpicture}
```
41. Compass style, configuration macro \tkzSetUpCompass

The following macro will help to understand the construction of a figure by showing the compass traces necessary to obtain certain points.

41.1. The macro \tkzSetUpCompass

\begin{tabular}{llp{5cm}}
\texttt{\tkzSetUpCompass} & \texttt{\[} & \texttt{\langle} \texttt{local options}\texttt{\rangle}\texttt{]} \\
\hline
\textbf{options} & \textbf{default} & \textbf{definition} \\
\hline
\texttt{color} & \texttt{black} & \texttt{colour of the construction lines} \\
\texttt{line width} & \texttt{0.4pt} & \texttt{thickness of the construction lines} \\
\texttt{style} & \texttt{solid} & \texttt{style of lines: solid, dashed, dotted,...} \\
\texttt{delta} & \texttt{0} & \texttt{changes the length of the arc} \\
\end{tabular}

41.1.1. Use of \tkzSetUpCompass

\begin{tikzpicture}
\tkzSetUpCompass[\texttt{color=red}, \texttt{delta=15}]
\tkzDefPoint(1,1){A}
\tkzDefPoint(6,1){B}
\tkzInterCC[R](A,4)(B,4) \tkzGetPoints{C}{D}
\tkzCompass(A,C)
\tkzCompass(B,C)
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}

41.1.2. Use of \tkzSetUpCompass with \tkzShowLine

\begin{tikzpicture}[\texttt{scale=.75}]
\tkzSetUpStyle[\texttt{bisector, size=2, gap=3}]{\texttt{showbi}}
\tkzSetUpCompass[\texttt{\textcolor{teal}{color=teal}, \texttt{line width=}.3\ pt}]
\tkzDefPoints{0/1/A, 8/3/B, 3/6/C}
\tkzDrawPolygon(A,B,C)
\tkzDefLine[\texttt{bisector}](B,A,C) \tkzGetPoint{a}
\tkzDefLine[\texttt{bisector}](C,B,A) \tkzGetPoint{b}
\tkzShowLine[\texttt{\textcolor{showbi}{showbi}}](B,A,C)
\tkzShowLine[\texttt{\textcolor{showbi}{showbi}}](C,B,A)
\tkzInterLL(A,a)(B,b) \tkzGetPoint{I}
\tkzDefPointBy[\texttt{projection= onto A--B}](I)
\tkzGetPoint{H}
\tkzDrawCircle[\texttt{new}](I,H)
\tkzDrawSegments[\texttt{new}](I,H)
\tkzDrawLines[\texttt{add=0 and .2, new}](A,I,B,I)
\end{tikzpicture}

42. Label style

42.1. The macro \tkzSetUpLabel

The macro \tkzSetUpLabel is used to define the style of the point labels.
43. Own style

You can set your own style with `\tkzSetUpStyle`

43.1. The macro `\tkzSetUpStyle`

\begin{tikzpicture}
    \tkzSetUpStyle[color=blue!20!black,fill=blue!20]{mystyle}
    \tkzDefPoint(0,0){O}
    \tkzDefPoint(0,1){A}
    \tkzDrawPoints(O,A)
    \tkzLabelPoints(O,A)
\end{tikzpicture}

44. How to use arrows

In some countries, arrows are used to indicate the parallelism of lines, to represent half-lines or the sides of an angle (rays).

Here are some examples of how to place these arrows. `tkz-euclide` loads a library called `arrows.meta`.

\begin{tikzpicture}
\usetikzlibrary{arrows.meta}
\end{tikzpicture}
44. How to use arrows

44.1. Arrows at endpoints on segment, ray or line

Stealth, Triangle, To, Latex and …which can be combined with reversed. That’s easy to place an arrow at one or two endpoints.

1. Triangle and Ray

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[-Triangle](A,B)
\end{tikzpicture}

2. Stealth and Segment

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[Stealth-Stealth](A,B)
\end{tikzpicture}

3. Latex and Line

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawLine[red,Latex-Latex](A,B)
\end{tikzpicture}

4. To and Segment

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[To-To](A,B)
\end{tikzpicture}

5. Latex and Segment

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[Latex-Latex](A,B)
\end{tikzpicture}

6. Latex and Ray

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[Latex-](A,B)
\end{tikzpicture}

7. Latex and Several rays

\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B,5/-2/C}
  \tkzDrawSegments[-Latex](A,B A,C)
\end{tikzpicture}
44. How to use arrows

44.1.1. Scaling an arrow head

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B}
\tkzDrawSegment[Latex[scale=2]--Latex[scale=2]](A,B)
\end{tikzpicture}

44.1.2. Using vector style

\begin{tikzpicture}
\begin{scope}[vector style/.style={>=Latex,->}]
\tkzDefPoints{0/0/A,4/0/B}
\tkzDrawSegment(A,B)
\end{scope}
\end{tikzpicture}

44.2. Arrows on middle point of a line segment

Arrows on lines are used to indicate that those lines are parallel. It depends on the country, in France we prefer to indicate outside the figure that \((A, B) \parallel (D, C)\). The code is an adaptation of an answer by muzimuzhi Z on the site tex.stackexchange.com.

Syntax:

- \texttt{tkz arrow} (Latex by default)
- \texttt{tkz arrow=<arrow end tip>}
- \texttt{tkz arrow=<arrow end tip> at \texttt{<pos>} \texttt{<pos>} = .5 by default}
- \texttt{tkz arrow=<arrow end tip>[<arrow options>] at \texttt{<pos>}\texttt{option possible scale}}

Example usages:

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B,4/2/C}
\tkzDefParallelogram(A,B,C)
\tkzGetPoint{D}
\tkzDrawSegment[tkz arrow](A,B D,C)
\tkzDrawSegment(B,C D,A)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above right](C,D)
\end{tikzpicture}

44.2.1. In a parallelogram

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B,4/2/C}
\tkzDefParallelogram(A,B,C)
\tkzGetPoint{D}
\tkzDrawSegments[tkz arrow](A,B D,C)
\tkzDrawSegments(B,C D,A)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above right](C,D)
\tkzDrawPoints(A,...,D)
\end{tikzpicture}
44. How to use arrows

44.2.2. A line parallel to another one

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B,1/2/C}
\tkzDefPointWith[colinear= at C](A,B)
\tkzGetPoint{D}
\tkzDrawSegments[tkz arrow=Triangle](A,B C,D)
\tkzLabelPoints(A,B,C)
\tkzDrawPoints(A,...,C)
\end{tikzpicture}

44.2.3. Arrow on a circle

It is possible to place an arrow on the first quarter of a circle. A rotation allows you to move the arrow.

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B}
\begin{scope}[rotate=150]
\tkzDrawCircle[tkz arrow={Latex[ scale=2, red]}](A,B)
\end{scope}
\end{tikzpicture}

44.3. Arrows on all segments of a polygon

Some users of my package have asked me to be able to place an arrow on each side of a polygon. I used a style proposed by Paul Gaborit on the site tex.stackexchange.com.

\tikzset{tkz arrows/.style={postaction={on each path={tkz arrow={Latex[ scale=2, color=black]}}}}}

You can change this style. With \texttt{tkz arrows} you can an arrow on each segment of a polygon

44.3.1. Arrow on each segment with \texttt{tkz arrows}

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B}
\tkzDefSquare(A,B) \tkzGetPoints{C}{D}
\tkzDrawPolygon[tkz arrows](A,...,D)
\end{tikzpicture}
44.3.2. Using \texttt{tkz arrows} with a circle

\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B}
\tkzDrawCircle[tkz arrows](A,B)
\end{tikzpicture}
Part IX.

Examples
45. Different authors

45.1. Code from Andrew Swan

\begin{tikzpicture} [scale=1.25]
\def\radius{4}
\def\angle{40}
\pgfmathsetmacro{\htan}{\tan(\angle)}
\tkzDefPoint(0,0){A} \tkzDefPoint(0,\radius){F}
\tkzDefPoint(\radius,0){B}
\tkzDefPointBy[rotation= center A angle \angle](B)
\tkzGetPoint(C)
\tkzDefLine[perpendicular=through B,K=1](A,B)
\tkzGetPoint(b)
\tkzInterLL(A,C)(B,b) \tkzGetPoint{D}
\tkzDefLine[perpendicular=through C,K=-1](A,B)
\tkzGetPoint(c)
\tkzInterLL(C,c)(A,B) \tkzGetPoint{E}
\tkzDrawSector[fill=blue,opacity=0.1](A,B)(C)
\tkzDrawArc[thin](A,B)(F)
\tkzMarkAngle(B,A,C){$x$}
\tkzDrawPolygon(A,B,D)
\tkzDrawSegments(C,B)
\tkzDrawSegments[dashed,thin](C,E)
\tkzLabelPoints[below left](A)
\tkzLabelPoints[below right](B)
\tkzLabelPoints[above right](D)
\tkzLabelPoints[above left](C)
\begin{scope}[pgf/decoration/raise=5pt]
\draw [decorate,decoration={brace,mirror,amplitude=10pt},xshift=0pt,yshift=-4pt](A) -- (B) node [black,midway,yshift=-20pt]{\footnotesize $1$};
\draw [decorate,decoration={brace,amplitude=10pt},xshift=4pt,yshift=0pt](D) -- (B) node [black,midway,xshift=27pt]{\footnotesize $\tan x$};
\draw [decorate,decoration={brace,amplitude=10pt},xshift=4pt,yshift=0pt](E) -- (C) node [black,midway,xshift=-27pt]{\footnotesize $\sin x$};
\end{scope}
\end{tikzpicture}

45.2. Example: Dimitris Kapeta

You need in this example to use \texttt{mkpos=.2} with \texttt{tkzMarkAngle} because the measure of $\widehat{CAM}$ is too small. Another possibility is to use \texttt{tkzFillAngle}.
45.3. Example: John Kitzmiller

Prove that \( \frac{AC}{CE} = \frac{BD}{DF} \).

Another interesting example from John, you can see how to use some extra options like `decoration` and `postaction` from TikZ with `tkz-euclide`.
45. Different authors

45.4. Example 1: from Indonesia
45. Different authors

\begin{tikzpicture}[scale=3]
\tkzDefPoints{0/0/A,2/0/B}
\tkzDefSquare(A,B) \tkzGetPoints{C}{D}
\tkzDefPointBy[rotation=center D angle 45](C){G}
\tkzDefSquare(G,D) \tkzGetPoints{E}{F}
\tkzInterLL(B,C)(E,F) \tkzGetPoint{H}
\tkzFillPolygon[gray!10](D,E,H,C,D)
\tkzDrawPolygon(A,...,D) \tkzDrawPolygon(D,...,G)
\tkzDrawSegment(B,E)
\tkzMarkSegments[mark=|,size=3pt,color=gray](A,B B,C C,D D,A E,F F,G G,D D,E)
\tkzMarkSegments[mark=||,size=3pt,color=gray](B,E E,H)
\tkzLabelPoints[left](A,D)
\tkzLabelPoints[right](B,C,F,H)
\tkzLabelPoints[above](G)
\tkzLabelPoints[below](E)
\tkzMarkRightAngles(D,A,B D,G,F)
\end{tikzpicture}

45.5. Example 2: from Indonesia

\begin{tikzpicture}[pol/.style={fill=brown!40,opacity=.2},
seg/.style={tkzdotted,color=gray}, hidden pt/.style={fill=gray!40},
mra/.style={color=gray!70,tkzdotted,/tkzrightangle/size=.2},scale=2]
\tkzDefPoints{0/0/A,2.5/0/B,1.33/0.75/D,0/2.5/E,2.5/2.5/F}
\tkzDefLine[parallel=through D](A,B) \tkzGetPoint{I1}
\tkzDefLine[parallel=through B](A,D) \tkzGetPoint{I2}
\tkzDefLine[parallel=through D](A,E) \tkzGetPoint{I3}
\tkzDefLine[parallel=through E](A,D) \tkzGetPoint{I4}
\tkzInterLL(D,I1)(B,I2) \tkzGetPoint{C}
\tkzDefLine[parallel=through E](A,D) \tkzGetPoint{I5}
\tkzDefLine[parallel=through F](E,H) \tkzGetPoint{I6}
\tkzDefLine[parallel=through H](E,F) \tkzGetPoint{I7}
\tkzDefMidPoint(G,H) \tkzGetPoint{P} \tkzDefMidPoint(G,C) \tkzGetPoint{Q}
\tkzDefMidPoint(B,E) \tkzGetPoint{R} \tkzDefMidPoint(A,B) \tkzGetPoint{S}
\tkzDefMidPoint(A,E) \tkzGetPoint{T} \tkzDefMidPoint(E,H) \tkzGetPoint{U}
\tkzDefMidPoint(A,D) \tkzGetPoint{N} \tkzDefMidPoint(D,C) \tkzGetPoint{M}
\tkzInterLL(B,D)(S,R) \tkzGetPoint{L} \tkzInterLL(H,F)(U,P) \tkzGetPoint{K}
\tkzDefLine[parallel=through K](D,H) \tkzGetPoint{I8}
\tkzDefLine[parallel=through K](B,D) \tkzGetPoint{I9}
\tkzDefLine[parallel=through K](D,H) \tkzGetPoint{I10}
\tkzDefLine[parallel=through K](B,D) \tkzGetPoint{I11}
\tkzDefLine[parallel=through K](D,H) \tkzGetPoint{I12}
\tkzInterLL(B,D)(S,R) \tkzGetPoint{L} \tkzInterLL(H,F)(U,P) \tkzGetPoint{K}
\tkzDefLine[parallel=through K](D,H) \tkzGetPoint{I13}
\tkzDefLine[parallel=through K](B,D) \tkzGetPoint{I14}
\tkzDefMidPoint(G,H) \tkzGetPoint{P} \tkzDefMidPoint(G,C) \tkzGetPoint{Q}
\tkzDefMidPoint(B,E) \tkzGetPoint{R} \tkzDefMidPoint(A,B) \tkzGetPoint{S}
\tkzDefMidPoint(A,E) \tkzGetPoint{T} \tkzDefMidPoint(E,H) \tkzGetPoint{U}
\tkzDefMidPoint(A,D) \tkzGetPoint{N} \tkzDefMidPoint(D,C) \tkzGetPoint{M}
\tkzInterLL(B,D)(S,R) \tkzGetPoint{L} \tkzInterLL(H,F)(U,P) \tkzGetPoint{K}
\tkzDefLine[parallel=through K](D,H) \tkzGetPoint{I15}
\tkzDefLine[parallel=through K](B,D) \tkzGetPoint{I16}
\tkzDefLine[parallel=through K](D,H) \tkzGetPoint{I17}
\tkzDefLine[parallel=through K](B,D) \tkzGetPoint{I18}
\tkzDrawPolygon(A,B,F,E)
\tkzMarkRightAngle[mra](L,O,K)
\tkzMarkSegments[mark=|,size=1pt,thick,color=gray](A,S B,S B,R,C,R Q,C Q,G G,P H,P E,U H,U E,T A,T)
\tkzLabelAngle[pos=.3](K,L,O)\text{(}$\alpha$\text{)}
\tkzLabelPoints[left](O,A,S,B) \tkzLabelPoints[above](H,P,G)
\tkzLabelPoints[left](T,E) \tkzLabelPoints[right](C,Q)
\tkzLabelPoints[above left](U,D,M) \tkzLabelPoints[above right](L,N)
\tkzLabelPoints[below right](F,R) \tkzLabelPoints[below left](K)
\end{tikzpicture}
45.6. Illustration of the Morley theorem by Nicolas François

\begin{tikzpicture}
  \tkzInit[ymin=-3,ymax=5,xmin=-5,xmax=7]
  \tkzClip
  \tkzDefPoints{-2.5/-2/A,2/4/B,5/-1/C}
  \tkzFindAngle(C,A,B) \tkzGetAngle{anglea}
  \tkzDefPointBy[rotation=center A angle 1\times\text{angles}/3](C) \tkzGetPoint{TA1}
  \tkzDefPointBy[rotation=center A angle 2\times\text{angles}/3](C) \tkzGetPoint{TA2}
  \tkzDefAngle(C,A,B) \tkzGetAngle{angleb}
  \tkzDefPointBy[rotation=center B angle 1\times\text{angles}/3](A) \tkzGetPoint{TB1}
  \tkzDefPointBy[rotation=center B angle 2\times\text{angles}/3](A) \tkzGetPoint{TB2}
  \tkzDefAngle(A,B,C) \tkzGetAngle{anglec}
  \tkzDefPointBy[rotation=center C angle 1\times\text{angles}/3](B) \tkzGetPoint{TC1}
  \tkzDefPointBy[rotation=center C angle 2\times\text{angles}/3](B) \tkzGetPoint{TC2}
  \tkzInterLL(A,TA1)(B,TB2) \tkzGetPoint{U1}
  \tkzInterLL(A,TA2)(B,TB1) \tkzGetPoint{V1}
  \tkzInterLL(B,TB1)(C,TC2) \tkzGetPoint{U2}
  \tkzInterLL(B,TB2)(C,TC1) \tkzGetPoint{V2}
  \tkzInterLL(C,TC1)(A,TA2) \tkzGetPoint{U3}
  \tkzInterLL(C,TC2)(A,TA1) \tkzGetPoint{V3}
  \tkzDrawPolygons(A,B,C U1,U2,U3 V1,V2,V3)
  \tkzDrawLines[add=2 and 2,very thin,dashed](A,TA1 B,TA1 C,TC1 A,TA2 B,TA2 C,TC2)
  \tkzDrawPoints(U1,U2,U3,V1,V2,V3)
  \tkzLabelPoint[left](V1){$s_a$} \tkzLabelPoint[right](V2){$s_b$}
  \tkzLabelPoint[below](V3){$s_c$} \tkzLabelPoint[above left](A){$A$}
  \tkzLabelPoints[above right](B,C) \tkzLabelPoint[below left](U1){$t_a$}
  \tkzLabelPoint[below left](U2){$t_b$} \tkzLabelPoint[above](U3){$t_c$}
\end{tikzpicture}
45.7. Gou gu theorem / Pythagorean Theorem by Zhao Shuang

Pythagoras was not the first person who discovered this theorem around the world. Ancient China discovered this theorem much earlier than him. So there is another name for the Pythagorean theorem in China, the Gou-Gu theorem. Zhao Shuang was an ancient Chinese mathematician. He rediscovered the “Gou gu theorem”, which is actually the Chinese version of the “Pythagorean theorem”. Zhao Shuang used a method called the “cutting and compensation principle”, he created a picture of “Pythagorean Round Square” Below the figure used to illustrate the proof of the “Gou gu theorem.” (code from Nan Geng)
45. Different authors

45.8. Reuleaux-Triangle

A well-known classic field of mathematics is geometry. You may know Euclidean geometry from school, with constructions by compass and ruler. Math teachers may be very interested in drawing geometry constructions and explanations. Underlying constructions can help us with general drawings where we would need intersections and tangents of lines and circles, even if it does not look like geometry. So, here, we will remember school geometry drawings. We will use the tkz-euclide package, which works on top of TikZ. We will construct an equilateral triangle. Then we extend it to get a Reuleaux triangle, and add annotations. The code is fully explained in the LaTeX Cookbook, Chapter 10, Advanced Mathematics, Drawing geometry pictures. Stefan Kottwitz
46. Some interesting examples

46.1. Square root of the integers

**Square root of the integers**

*How to get $1, \sqrt{2}, \sqrt{3}$ with a rule and a compass.*
46.2. About right triangle

We have a segment \([AB]\) and we want to determine a point \(C\) such that \(AC = 8\) cm and \(ABC\) is a right triangle in \(B\).

46.3. Archimedes

This is an ancient problem proved by the great Greek mathematician Archimedes. The figure below shows a semicircle, with diameter \(AB\). A tangent line is drawn and touches the semicircle at \(B\). An other tangent line at a point, \(C\), on the semicircle is drawn. We project the point \(C\) on the line segment \([AB]\) on a point \(D\). The two tangent lines intersect at the point \(T\). Prove that the line \((AT)\) bisects \((CD)\).
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A} \tkzDefPoint(6,0){D}
  \tkzDefPoint(8,0){B} \tkzDefPoint(4,0){I}
  \tkzDefLine[orthogonal=through D](A,D)
  \tkzInterLC[R](D,tkzPointResult)(I,4) \tkzGetSecondPoint{C}
  \tkzDefLine[orthogonal=through C](I,C) \tkzGetPoint{c}
  \tkzDefLine[orthogonal=through B](A,B) \tkzGetPoint{b}
  \tkzInterLL(C,c)(B,b) \tkzGetPoint{T}
  \tkzInterLL(A,T)(C,D) \tkzGetPoint{P}
  \tkzDrawArc(I,B)(A)
  \tkzDrawSegments(A,B A,T C,D I,C) \tkzDrawSegment[new](I,C)
  \tkzDrawLine[add = 1 and 0](C,T) \tkzDrawLine[add = 0 and 1](B,T)
  \tkzMarkRightAngle(I,C,T)
  \tkzDrawPoints(A,B,I,D,C,T)
  \tkzLabelPoints(A,B,I,D) \tkzLabelPoints[above right](C,T)
  \tkzMarkSegment[pos=.25,mark=s|](C,D) \tkzMarkSegment[pos=.75,mark=s|](C,D)
\end{tikzpicture}
46.3.1. Square and rectangle of same area; Golden section

*Book II, proposition XI* _Euclid’s Elements_

To construct Square and rectangle of same area.
46. Some interesting examples

46.3.2. Steiner Line and Simson Line

Consider the triangle ABC and a point M on its circumcircle. The projections of M on the sides of the triangle are on a line (Steiner Line). The three closest points to M on lines AB, AC, and BC are collinear. It's the Simson Line.

\begin{tikzpicture}[scale=.75,rotate=-20]
  \tkzDefPoint(0,0){B}
  \tkzDefPoint(2,4){A} \tkzDefPoint(7,0){C}
  \tkzDefCircle[ circumcision](A,B,C)
  \tkzGetPoint{O}
  \tkzDrawCircle(O,A)
  \tkzCalcLength(O,A)
  \tkzGetLength{rOA}
  \tkzDefShiftPoint[O](40:\rOA){M}
  \tkzDefShiftPoint[O](60:\rOA){N}
  \tkzDefTriangleCenter[orthic](A,B,C)
  \tkzGetPoint{H}
  \tkzDefSpcTriangle[orthic,name=H](A,B,C\{a,b,c
  \tkzDefPointsBy[reflection=over A--B](M,N){P,P'}
  \tkzDefPointsBy[reflection=over A--C](M,N){Q,Q'}
  \tkzDefPointsBy[reflection=over C--B](M,N){R,R'}
  \tkzDefMidPoint(M,P)\tkzGetPoint{I}
  \tkzDefMidPoint(M,Q)\tkzGetPoint{J}
  \tkzDefMidPoint(M,R)\tkzGetPoint{K}
  \tkzDrawSegments[P,R M,P M,Q M,R N,P'\% N,Q' N,R' P',R' I,K]
  \tkzDrawPolygons(A,B,C)
\end{tikzpicture}
46. Some interesting examples

46.4. Lune of Hippocrates

From Wikipedia: In geometry, the lune of Hippocrates, named after Hippocrates of Chios, is a lune bounded by arcs of two circles, the smaller of which has as its diameter a chord spanning a right angle on the larger circle. In the first figure, the area of the lune is equal to the area of the triangle ABC. Hippocrates of Chios (ancient Greek mathematician,)

\begin{tikzpicture}
\tkzInit[xmin=-2,xmax=5,ymin=-1,ymax=6]
\tkzClip % allows you to define a bounding box
% large enough
\tkzDefPoint(0,0){A}\tkzDefPoint(4,0){B}
\tkzDefSquare(A,B)
\tkzGetFirstPoint{C}
\tkzDrawPolygon[fill=green!5](A,B,C)
\begin{scope}
\tkzClipCircle[out](B,A)
\tkzDefMidPoint(C,A) \tkzGetPoint{M}
\tkzDrawSemiCircle[fill=teal!5](M,C)
\end{scope}
\tkzDrawArc[delta=0](B,C)(A)
\end{tikzpicture}

46.5. Lunes of Hasan Ibn al-Haytham

From Wikipedia: The Arab mathematician Hasan Ibn al-Haytham (Latinized name Alhazen) showed that two lunes, formed on the two sides of a right triangle, whose outer boundaries are semicircles and whose inner boundaries are formed by the circumcircle of the triangle, then the areas of these two lunes added together are equal to the area of the triangle. The lunes formed in this way from a right triangle are known as the lunes of Alhazen.
\begin{tikzpicture}[scale=.5,rotate=180]
\tkzInit[xmin=-1,xmax=11,ymin=-4,ymax=7]
\tkzClip
\tkzDefPoints{0/0/A,8/0/B}
\tkzDefTriangle[pythagore,swap](A,B)
\tkzGetPoint{C}
\tkzDrawPolygon[fill=green!5](A,B,C)
\tkzDefMidPoint(C,A) \tkzGetPoint{I}
\begin{scope}
\tkzClipCircle[out](I,A)
\tkzDefMidPoint(B,A) \tkzGetPoint{x}
\tkzDrawSemiCircle[fill=teal!5](x,A)
\tkzDefMidPoint(B,C) \tkzGetPoint{y}
\tkzDrawSemiCircle[fill=teal!5](y,B)
\end{scope}
\tkzSetUpCompass[/tkzcompass/delta=0]
\tkzDefMidPoint(C,A) \tkzGetPoint{z}
\tkzDrawSemiCircle(z,A)
\end{tikzpicture}
46. Some interesting examples

46.6. About clipping circles

The problem is the management of the bounding box. First you have to define a rectangle in which the figure will be inserted. This is done with the first two lines.

```latex
\begin{tikzpicture}
\tkzInit[xmin=0,xmax=6,ymin=0,ymax=6]
\tkzClip
\tkzDefPoints{0/0/A, 6/0/B}
\tkzDefSquare(A,B) \tkzGetPoints{C}{D}
\tkzDefMidPoint(A,B) \tkzGetPoint{M}
\tkzDefMidPoint(A,D) \tkzGetPoint{N}
\tkzDefMidPoint(B,C) \tkzGetPoint{O}
\tkzDefMidPoint(C,D) \tkzGetPoint{P}
\begin{scope}
\tkzClipCircle[out](M,B) \tkzClipCircle[out](P,D)
\tkzFillPolygon[teal!20](M,N,P,O)
\end{scope}
\begin{scope}
\tkzClipCircle[out](N,A) \tkzClipCircle[out](O,C)
\tkzFillPolygon[teal!20](M,N,P,O)
\end{scope}
\begin{scope}
\tkzClipCircle(P,C) \tkzClipCircle(N,A)
\tkzFillPolygon[teal!20](N,P,D)
\end{scope}
\begin{scope}
\tkzClipCircle(O,C) \tkzClipCircle(P,C)
\tkzFillPolygon[teal!20](P,C,O)
\end{scope}
\begin{scope}
\tkzClipCircle(M,B) \tkzClipCircle(O,B)
\tkzFillPolygon[teal!20](O,B,M)
\end{scope}
\begin{scope}
\tkzClipCircle(N,A) \tkzClipCircle(M,A)
\tkzFillPolygon[teal!20](A,M,N)
\end{scope}
\tkzDrawSemiCircles(M,B N,A O,C P,D)
\tkzDrawPolygons(A,B,C,D M,N,P,O)
\end{tikzpicture}
```
46.7. Similar isosceles triangles

The following is from the excellent site Descartes et les Mathématiques. I did not modify the text and I am only the author of the programming of the figures. http://debart.pagesperso-orange.fr/seconde/triangle.html

The following is from the excellent site Descartes et les Mathématiques. I did not modify the text and I am only the author of the programming of the figures. http://debart.pagesperso-orange.fr/seconde/triangle.html

Bibliography:

– Géométrie au Bac - Tangente, special issue no. 8 - Exercise 11, page 11
– Elisabeth Busser and Gilles Cohen: 200 nouveaux problèmes du "Monde" - POLE 2007 (200 new problems of "Le Monde")
– Affaire de logique n° 364 - Le Monde February 17, 2004

Two statements were proposed, one by the magazine Tangente and the other by Le Monde.

Editor of the magazine "Tangente": Two similar isosceles triangles AXB and BYC are constructed with main vertices X and Y, such that A, B and C are aligned and that these triangles are "indirect". Let α be the angle at vertex AXB = BYC. We then construct a third isosceles triangle XZY similar to the first two, with main vertex Z and "indirect". We ask to demonstrate that point Z belongs to the straight line (AC).

Editor of "Le Monde": We construct two similar isosceles triangles AXB and BYC with principal vertices X and Y, such that A, B and C are aligned and that these triangles are "indirect". Let α be the angle at vertex AXB = BYC. The point Z of the line segment [AC] is equidistant from the two vertices X and Y.
At what angle does he see these two vertices?

The constructions and their associated codes are on the next two pages, but you can search before looking. The programming respects (it seems to me ...) my reasoning in both cases.
46. Some interesting examples

46.8. Revised version of "Tangente"

\begin{tikzpicture}[scale=.8,rotate=60]
\tkzDefPoint(6,0){X} \tkzDefPoint(3,3){Y}
\tkzDefShiftPoint[X](-110:6){A} \tkzDefShiftPoint[X](-70:6){B}
\tkzDefShiftPoint[Y](-110:4.2){A'} \tkzDefShiftPoint[Y](-70:4.2){B'}
\tkzDefPointBy[translation= from A' to B ](Y) \tkzGetPoint{Y}
\tkzDefPointBy[translation= from A' to B ](B') \tkzGetPoint{C}
\tkzInterLL(A,B)(X,Y) \tkzGetPoint{O}
\tkzDefMidPoint(X,Y) \tkzGetPoint{I}
\tkzDefPointWith[orthogonal](I,Y)
\tkzInterLL(I,tkzPointResult)(A,B) \tkzGetPoint{Z}
\tkzDefCircle[circum](X,Y,B) \tkzGetPoint{O}
\tkzDrawCircle(O,X)
\tkzDrawLines[add = 0 and 1.5](A,C) \tkzDrawLines[add = 0 and 3](X,Y)
\tkzDrawSegments(A,X B,X B,Y C,Y)
\tkzDrawSegments[\color=red](X,Z Y,Z)
\tkzDrawPoints(A,B,C,X,Y,O,Z)
\tkzLabelPoints(A,B,C,Z) \tkzLabelPoints[above right](X,Y,O)
\end{tikzpicture}
46. Some interesting examples

46.9. "Le Monde" version

\begin{tikzpicture}[scale=1.25]
\tkzDefPoint(0,0){A}
\tkzDefPoint(3,0){B}
\tkzDefPoint(9,0){C}
\tkzDefPoint(1.5,2){X}
\tkzDefPoint(6,4){Y}
\tkzDefCircle[circum](X,Y,B) \tkzGetPoint{O}
\tkzDefMidPoint(X,Y) \tkzGetPoint{I}
\tkzDefPointWith[orthogonal](I,Y) \tkzGetPoint{i}
\tkzDrawLines[add = 2 and 1,color=orange](I,i)
\tkzInterLL(I,i)(A,B) \tkzGetPoint{Z}
\tkzInterLC(I,i)(O,B) \tkzGetFirstPoint{M}
\tkzDefPointWith[orthogonal](B,Z) \tkzGetPoint{b}
\tkzDrawCircle(O,B)
\tkzDrawLines[add = 8 and 2, color=orange](B,b)
\tkzDrawSegments(A,B,B,X,B,Y,C,Y,A,C,X,Y)
\tkzDrawSegments[color=red](X,Z,Y,Z)
\tkzDrawPoints(A,B,C,X,Y,Z,M,I)
\tkzLabelPoints(A,B,C,Z)
\tkzLabelPoints[above right](X,Y,M,I)
\end{tikzpicture}
46.10. Triangle altitudes

Triangle altitudes

From Wikipedia: The following is again from the excellent site Descartes et les Mathématiques (Descartes and the Mathematics). http://debart.pagesperso-orange.fr/geoplan/geometrie_triangle.html. The three altitudes of a triangle intersect at the same H-point.
46. Some interesting examples

46.11. Altitudes - other construction

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(8,0){B}
\tkzDefPoint(5,6){C}
\tkzDefMidPoint(A,B)\tkzGetPoint{O}
\tkzDefPointBy[projection=onto A--B](C) \tkzGetPoint{P}
\tkzInterLC[common=A](C,A)(O,A) \tkzGetFirstPoint{M}
\tkzInterLC(C,B)(O,A) \tkzGetSecondPoint{N}
\tkzInterLL(B,M)(A,N) \tkzGetPoint{I}
\tkzDefCircle[diameter](A,B) \tkzGetPoint{x}
\tkzDefCircle[diameter](I,C) \tkzGetPoint{y}
\tkzDrawCircles(x,A y,C)
\tkzDrawSegments(C,A C,B A,B B,M A,N)
\tkzMarkRightAngles[fill=brown!20](A,M,B A,N,B A,P,C)
\tkzDrawSegment[style=dashed,color=orange](C,P)
\tkzLabelPoints(O,A,B,P)
\tkzLabelPoint[left](M){$M$}
\tkzLabelPoint[right](N){$N$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[above right](I){$I$}
\tkzDrawPoints[color=red](M,N,P,I)
\tkzDrawPoints[color=brown](0,A,B,C)
\end{tikzpicture}
46. Some interesting examples

46.12. Three circles in an Equilateral Triangle

From Wikipedia: In geometry, the Malfatti circles are three circles inside a given triangle such that each circle is tangent to the other two and to two sides of the triangle. They are named after Gian Francesco Malfatti, who made early studies of the problem of constructing these circles in the mistaken belief that they would have the largest possible total area of any three disjoint circles within the triangle. Below is a study of a particular case with an equilateral triangle and three identical circles.

\begin{tikzpicture}[scale=.8]
\tkzDefPoints{0/0/A,8/0/B,8/4/a,8/4/b,8/8/c}
\tkzDefTriangle[equilateral](A,B) \tkzGetPoint{C}
\tkzDefMidPoint(A,B) \tkzGetPoint{M}
\tkzDefMidPoint(B,C) \tkzGetPoint{N}
\tkzDefMidPoint(A,C) \tkzGetPoint{P}
\tkzInterLL(A,N)(M,a) \tkzGetPoint{Ia}
\tkzDefPointBy[projection = onto A--B](Ia) \tkzGetPoint{ha}
\tkzInterLL(B,P)(M,b) \tkzGetPoint{Ib}
\tkzDefPointBy[projection = onto A--B](Ib) \tkzGetPoint{hb}
\tkzInterLL(A,C)(M,C) \tkzGetPoint{Ic}
\tkzDefPointBy[projection = onto A--C](Ic) \tkzGetPoint{hc}
\tkzInterLL(A,Ia)(B,Ib) \tkzGetPoint{G}
\tkzDefSquare(A,B) \tkzGetPoints{D}{E}
\tkzDrawPolygon(A,B,C)
\tkzClipBB
\tkzDrawCircles[gray](Ia,ha Ib,hb Ic,hc)
\tkzDrawPolySeg(A,E,D,B)
\tkzDrawPoints(A,B,C,G)
\tkzDrawPoints(A,B,C,G,Ia,Ib,Ic)
\tkzDrawSegments[gray,dashed](C,M A,N B,P M,a M,b a a,b b,B A,D Ia,ha)
\end{tikzpicture}
46.13. Law of sines

From wikipedia: In trigonometry, the law of sines, sine law, sine formula, or sine rule is an equation relating the lengths of the sides of a triangle (any shape) to the sines of its angles.

\[
\frac{a}{\sin A} = \frac{b}{\sin B} = \frac{c}{\sin C} \quad (1)
\]
\[
\hat{C} = \hat{D}
\]
\[
\frac{c}{2R} = \sin D = \sin C
\]

Then
\[
\frac{c}{\sin C} = 2R
\]
Sacred geometry can be described as a belief system attributing a religious or cultural value to many of the fundamental forms of space and time. According to this belief system, the basic patterns of existence are perceived as sacred because in contemplating them one is contemplating the origin of all things. By studying the nature of these forms and their relationship to each other, one may seek to gain insight into the scientific, philosophical, psychological, aesthetic and mystical laws of the universe. The Flower of Life is considered to be a symbol of sacred geometry, said to contain ancient, religious value depicting the fundamental forms of space and time. In this sense, it is a visual expression of the connections life weaves through all mankind, believed by some to contain a type of Akashic Record of basic information of all living things.

\begin{tikzpicture}[scale=.75]
  \tkzSetUpLine[line width=2pt, color=teal!80!black]
  \tkzSetUpCompass[line width=2pt, color=teal!80!black]
  \tkzDefPoint(0,0){O} \tkzDefPoint(2.25,0){A}
  \tkzDrawCircle(O,A)
  \foreach \i in {0,...,5}{
    \tkzDefPointBy[rotation= center O angle 30+60*\i](A)\tkzGetPoint{a\i}
    \tkzDefPointBy[rotation= center {a\i} angle 120](O)\tkzGetPoint{b\i}
    \tkzDefPointBy[rotation= center {a\i} angle 180](O)\tkzGetPoint{c\i}
    \tkzDefPointBy[rotation= center {c\i} angle 120](a\i)\tkzGetPoint{d\i}
    \tkzDefPointBy[rotation= center {c\i} angle 60](d\i)\tkzGetPoint{f\i}
    \tkzDefPointBy[rotation= center {f\i} angle 60](d\i)\tkzGetPoint{g\i}
    \tkzDefPointBy[rotation= center {f\i} angle 180](b\i)\tkzGetPoint{k\i}
    \tkzDefPointBy[rotation= center {e\i} angle 180](b\i)\tkzGetPoint{h\i}
    \tkzDrawCircle(a\i,O)
    \tkzDrawCircle(b\i,a\i)
    \tkzDrawCircle(c\i,a\i)
    \tkzDrawArc[rotate](f\i,d\i)(-120)
    \tkzDrawArc[rotate](e\i,d\i)(180)
    \tkzDrawArc[rotate](d\i,c\i)(180)
    \tkzDrawArc[rotate](g\i,f\i)(60)
    \tkzDrawArc[rotate](h\i,d\i)(60)
    \tkzDrawArc[rotate](k\i,e\i)(60)
  }
  \tkzClipCircle(O,f0)
\end{tikzpicture}
46.15. Pentagon in a circle

To inscribe an equilateral and equiangular pentagon in a given circle.
46.16. Pentagon in a square

: To inscribe an equilateral and equiangular pentagon in a given square.

\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/O,-5/-5/A,5/-5/B}
\tkzDefSquare(A,B) \tkzGetPoints{C}{D}
\tkzDefMidPoint(A,B) \tkzGetPoint{F}
\tkzDefMidPoint(C,D) \tkzGetPoint{E}
\tkzDefMidPoint(B,C) \tkzGetPoint{G}
\tkzDefMidPoint(A,D) \tkzGetPoint{K}
\tkzInterLC(D,C)(E,B) \tkzGetSecondPoint{T}
\tkzDefMidPoint(D,T) \tkzGetPoint{I}
\tkzInterCC[with nodes](O,D,I)(E,D,I) \tkzGetSecondPoint{H}
\tkzInterLC(O,H)(O,E) \tkzGetSecondPoint{M}
\tkzInterCC(O,E)(E,M) \tkzGetFirstPoint{Q}
\tkzInterCC[with nodes](O,O,E)(Q,E,M) \tkzGetFirstPoint{P}
\tkzInterCC[with nodes](O,O,E)(P,E,M) \tkzGetFirstPoint{N}
\tkzCompasss(O,H E,H)
\tkzDrawArc(E,B)(T)
\tkzDrawPolygons[purple](A,B,C,D M,E,Q,P,N)
\tkzDrawCircle(O,E)
\tkzDrawSegments(T,I O,H E,H E,F G,K)
\tkzDrawPoints(T,M,O,P,N,I)
\tkzLabelPoints(A,B,0,N,O,P,Q,M,H)
\tkzLabelPoints[above right](C,D,E,I,T)
\end{tikzpicture}
46.17. Hexagon Inscribed

To inscribe a regular hexagon in a given equilateral triangle perfectly inside it (no boarders).

46.17.1. Hexagon Inscribed version 1

\begin{tikzpicture}[scale=.5]
\pgfmathsetmacro{\c}{6}
\tkzDefPoints{0/0/A,\c/0/B}
\tkzDefTriangle[equilateral](A,B)\tkzGetPoint{C}
\tkzDefTriangleCenter[centroid](A,B,C)
\tkzGetPoint{I}
\tkzDefPointBy[homothety=center A ratio 1./3](B)
\tkzGetPoint{c1}
\tkzInterLC(B,C)(I,c1) \tkzGetPoints{a1}{a2}
\tkzInterLC(A,C)(I,c1) \tkzGetPoints{b1}{b2}
\tkzInterLC(A,B)(I,c1) \tkzGetPoints{c1}{c2}
\tkzDrawPolygon(A,B,C)
\tkzDrawCircle[thin,orange](I,c1)
\tkzDrawPolygon[red,thick](a2,a1,b2,b1,c2,c1)
\end{tikzpicture}

46.17.2. Hexagon Inscribed version 2

\begin{tikzpicture}[scale=.5]
\pgfmathsetmacro{\c}{6}
\tkzDefPoints{0/0/A,\c/0/B}
\tkzDefTriangle[equilateral](A,B)\tkzGetPoint{C}
\tkzDefTriangleCenter[centroid](A,B,C)
\tkzGetPoint{I}
\tkzDefPointsBy[rotation= center I angle 60/](A,B,C){a,b,c}
\tkzDrawPolygon[fill=teal!20,opacity=.5](A,B,C)
\tkzDrawPolygon[fill=purple!20,opacity=.5](a,b,c)
\end{tikzpicture}
46.18. Power of a point with respect to a circle

$\overline{MA} \times \overline{MB} = \overline{MT}^2 = \overline{MO}^2 - \overline{OT}^2$

\begin{tikzpicture}
    \pgfmathsetmacro{\r}{2} \% \\
    \pgfmathsetmacro{\xO}{6} \% \\
    \pgfmathsetmacro{\xE}{\xO-\r} \% \\
    \tkzDefPoints{0/0/M,\xO/0/O,\xE/0/E} \\
    \tkzDefCircle[diameter](M,O) \\
    \tkzGetPoint{I} \\
    \tkzInterCC(I,O)(O,E) \tkzGetPoints{T}{T'} \\
    \tkzDefShiftPoint[O](45:2){B} \\
    \tkzInterLC(M,B)(O,E) \tkzGetPoints{A}{B} \\
    \tkzDrawCircle(O,E) \\
    \tkzDrawSemiCircle[dashed](I,O) \\
    \tkzDrawLine(M,O) \\
    \tkzDrawLines(M,T O,T M,B) \\
    \tkzDrawPoints(A,B,T) \\
    \tkzLabelPoints[above](A,B,O,M,T) \\
\end{tikzpicture}
46.19. Radical axis of two non-concentric circles

From Wikipedia: In geometry, the radical axis of two non-concentric circles is the set of points whose power with respect to the circles are equal. For this reason the radical axis is also called the power line or power bisector of the two circles. The notation radical axis was used by the French mathematician M. Chasles as axe radical.
46. Some interesting examples

46.20. External homothetic center

From Wikipedia: Given two nonconcentric circles, draw radii parallel and in the same direction. Then the line joining the extremities of the radii passes through a fixed point on the line of centers which divides that line externally in the ratio of radii. This point is called the external homothetic center, or external center of similitude (Johnson 1929, pp. 19-20 and 41).

\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/2/B,2/3/K}
\tkzDefCircle[R](A,1)\tkzGetPoint{a}
\tkzDefCircle[R](B,2)\tkzGetPoint{b}
\tkzDrawCircles(A,a B,b)
\tkzDrawLine(A,B)
\tkzDefShiftPoint[A](60:1){M}
\tkzDefShiftPoint[B](60:2){M'}
\tkzInterLL(A,B)(M,M') \tkzGetPoint{O}
\tkzDefLine[tangent from = O](B,M') \tkzGetPoints{X}{T'}
\tkzDefLine[tangent from = O](A,M) \tkzGetPoints{X}{T}
\tkzDrawPoints(A,B,O,T,T',M,M')
\tkzDrawLines[thick](O,B O,T' O,M')
\tkzDrawSegments[thick](A,M B,M')
\tkzLabelPoints(A,B,O,T,T',M,M')
\end{tikzpicture}
46.21. Tangent lines to two circles

For two circles, there are generally four distinct lines that are tangent to both if the two circles are outside each other. For two of these, the external tangent lines, the circles fall on the same side of the line; the external tangent lines intersect in the external homothetic center.

\begin{tikzpicture}
\pgfmathsetmacro{\r}{1}
\pgfmathsetmacro{\R}{2}
\pgfmathsetmacro{\rt}{\R-\r}
\tkzDefPoints{0/0/A,4/2/B,2/3/K}
\tkzDefMidPoint(A,B) \tkzGetPoint{I}
\tkzInterLC(R)(A,B)(B,\rt) \tkzGetPoints(E,F)
\tkzInterCC(I,B)(B,F) \tkzGetPoints{a}{a'}
\tkzInterLC(R)(B,a)(B,B) \tkzGetPoints{X'}{T'}
\tkzDefLine[tangent at=T'](B) \tkzGetPoint{h}
\tkzInterLL(T',h)(A,B) \tkzGetPoint{O}
\tkzInterLC(R)(O,T')(A,\R) \tkzGetPoints{T}{T'}
\tkzDefCircle[R](A,\r) \tkzGetPoints{a}
\tkzDefCircle[R](B,\R) \tkzGetPoint{b}
\tkzDefCircle[R](B,\rt) \tkzGetPoint{c}
\tkzDrawCircles(A,a)
\tkzDrawCircles[orange](B,b,B,c)
\tkzDrawCircle[orange,dashed](I,B)
\tkzDrawPoints(O,A,B,a,a',E,F,T',T)
\tkzDrawLines(O,B A,a B,T' A,T)
\tkzDrawLines[add= 1 and 8](T',h)
\tkzLabelPoints(O,A,B,a,a',E,F,T,T')
\end{tikzpicture}
46.22. Tangent lines to two circles with radical axis

As soon as two circles are not concentric, we can construct their radical axis, the set of points of equal power with respect to the two circles. We know that the radical axis is a line orthogonal to the line of the centers. Note that if we specify $P$ and $Q$ as the points of contact of one of the common exterior tangents with the two circles and $D$ and $E$ as the points of the circles outside $\{AB\}$, then $(DP)$ and $(EQ)$ intersect on the radical axis of the two circles. We will show that this property is always true and that it allows us to construct common tangents, even when the circles have the same radius.
46.23. Middle of a segment with a compass

This example involves determining the middle of a segment, using only a compass.

\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefRandPointOn[circle= center A radius 4](A) \tkzGetPoint{B}
\tkzDefPointBy[rotation= center A angle 180](B) \tkzGetPoint{C}
\tkzInterCC(A,B)(B,A) \tkzGetPoints{I}{I'}
\tkzInterCC(A,I)(I,A) \tkzGetPoints{J}{J'}
\tkzInterCC(B,A)(C,B) \tkzGetPoints{D}{E}
\tkzInterCC(D,B)(E,B) \tkzGetPoints{M}{M'}
\tkzSetUpArc[color=orange,style=solid,delta=10]
\tkzDrawArc(C,D)(E)
\tkzDrawArc(B,E)(D)
\tkzDrawCircle[color=teal,line width=.2pt](A,B)
\tkzDrawArc(D,B)(M)
\tkzDrawArc(E,M)(B)
\tkzCompass[color=orange,style=solid](B,I,J,J',C)
\tkzDrawPoints(A,B,C,D,E,M)
\tkzLabelPoints(A,B,M)
\end{tikzpicture}
46.24. Definition of a circle _Apollonius_

From Wikipedia: Apollonius showed that a circle can be defined as the set of points in a plane that have a specified ratio of distances to two fixed points, known as foci. This Apollonian circle is the basis of the Apollonius pursuit problem. ... The solutions to this problem are sometimes called the circles of Apollonius.

Explanation
A circle is the set of points in a plane that are equidistant from a given point O. The distance r from the center is called the radius, and the point O is called the center. It is the simplest definition but it is not the only one. Apollonius of Perga gives another definition: The set of all points whose distances from two fixed points are in a constant ratio is a circle.

With tkz-euclide is easy to show you the last definition

```latex
\begin{tikzpicture}[scale=1.5]
    % Firstly we defined two fixed point.
    % The figure depends of these points and the ratio K
    \tkzDefPoint(0,0){A}
    \tkzDefPoint(4,0){B}
    \begin{itemize}
    \item % tkz-euclide.sty knows about the apollonius's circle
      \item with K=2 we search some points like I such as IA=2 x IB
    \end{itemize}
    \tkzDefCircle[apollonius,K=2](A,B) \tkzGetPoints{K1}{k}
    \tkzDefPointOnCircle[through= center K1 angle 30 point k]{I}
    \tkzDefPointOnCircle[through= center K1 angle 280 point k]{J}
    \tkzDrawSegments[new](A,I I,B A,J J,B)
    \tkzDrawCircle[color = teal,fill=teal!20,opacity=.4](K1,k)
    \tkzDrawPoints(A,B,K1,I,J)
    \end{tikzpicture}
```

\textbf{tkz-euclide} AlterMundus
From Wikipedia: *In geometry, the Pappus chain is a ring of circles between two tangent circles investigated by Pappus of Alexandria in the 3rd century AD.*

\begin{tikzpicture}[ultra thin]
\pgfmathsetmacro{\xB}{6}%
\pgfmathsetmacro{\xC}{9}%
\pgfmathsetmacro{\xD}{(\xC*\xC)/\xB}%
\pgfmathsetmacro{\xJ}{(\xC+\xD)/2}%
\pgfmathsetmacro{\r}{\xD-\xJ}%
\pgfmathsetmacro{\nc}{16}%
\tkzDefPoints{0/0/A,\xB/0/B,\xC/0/C,\xD/0/D}
\tkzDefCircle[diameter](A,C) \tkzGetPoint{x}
\tkzDrawCircle[fill=teal!30](x,C)
\tkzDefCircle[diameter](A,B) \tkzGetPoint{y}
\tkzDrawCircle[fill=teal!30](y,B)
\foreach \i in {-\nc,...,0,...,\nc}
{\tkzDefPoint(\xJ,2*\r*\i){J}
 \tkzDefPoint(\xJ,2*\r*\i-\r){H}
 \tkzDefCircleBy[inversion = center A through C](J,H)
 \tkzDrawCircle[fill=teal](tkzFirstPointResult,tkzSecondPointResult)}
\end{tikzpicture}
46. Some interesting examples

46.26. Book of lemmas proposition 1 Archimedes

If two circles touch at A, and if (CD), (EF) be parallel diameters in them, A, C and E are aligned.

46.27. Book of lemmas proposition 6 Archimedes

Let AC, the diameter of a semicircle, be divided at B so that AC/AB = \( \phi \) or in any ratio. Describe semicircles within the first semicircle and on AB, BC as diameters, and suppose a circle drawn touching the all three semicircles. If GH be the diameter of this circle, to find relation between GH and AC.
Let GH be the diameter of the circle which is parallel to AC, and let the circle touch the semicircles on AC, AB, BC in D, E, F respectively.

Then, by Prop. 1 A, G and D are aligned, ainsi que D, H and C.

For a like reason A E and H are aligned, C F and G are aligned, as also are B E and G, B F and H.

Let (AD) meet the semicircle on [AC] at I, and let (BD) meet the semicircle on [BC] in K. Join CI, CK meeting AE, BF in L, M, and let GL, HM produced meet AB in N, P respectively.

Now, in the triangle AGB, the perpendiculars from A, C on the opposite sides meet in L. Therefore by the properties of triangles, (GN) is perpendicular to (AC). Similarly (HP) is perpendicular to (BC).

Again, since the angles at I, K, D are right, (CK) is parallel to (AD), and (CI) to (BD).

Therefore

\[
\frac{AB}{BC} = \frac{AL}{LH} = \frac{AN}{NP} \quad \text{and} \quad \frac{BC}{AB} = \frac{CM}{MG} = \frac{PC}{NP}
\]

hence

\[
\frac{AN}{NP} = \frac{NP}{PC} \quad \text{so} \quad NP^2 = AN \times PC
\]

Now suppose that B divides [AC] according to the divine proportion that is:

\[
\phi = \frac{AB}{BC} = \frac{AC}{AB} \quad \text{then} \quad AN = \phi NP \text{and} \quad NP = \phi PC
\]

We have

\[
AC = AN + NP + PC \quad \text{either} \quad AB + BC = AN + NP + PC \quad \text{or} \quad (\phi + 1)BC = AN + NP + PC
\]

we get

\[
(\phi + 1)BC = \phi NP + NP + PC = (\phi + 1)NP + PC = \phi(\phi + 1)PC + PC = \phi^2 + \phi + 1)PC
\]
as

$$\phi^2 = \phi + 1 \quad \text{then} \quad (\phi + 1)BC = 2(\phi + 1)PC \quad \text{i.e.} \quad BC = 2PC$$

That is, p is the middle of the segment BC.

Part of the proof from https://www.cut-the-knot.org

46.28. "The" Circle of APOLLONIUS

"The Apollonius circle of a triangle _Apollonius_."

The circle which touches all three excircles of a triangle and encompasses them is often known as "the" Apollonius circle (Kimberling 1998, p. 102)

Explanation

The purpose of the first examples was to show the simplicity with which we could recreate these propositions. With TikZ you need to do calculations and use trigonometry while with tkz-euclide you only need to build simple objects.

But don’t forget that behind or far above tkz-euclide there is TikZ. I’m only creating an interface between TikZ and the user of my package.

The last example is very complex and it is to show you all that we can do with tkz-euclide.
46. Some interesting examples
Some interesting examples

\begin{tikzpicture}[scale=.6]
\tkzDefPoints{A/6/0,B/0/6,C/8/4}
\tkzDefTriangleCenter[euler](A,B,C) \tkzGetPoint{N}
\tkzDefTriangleCenter[circum](A,B,C) \tkzGetPoint{O}
\tkzDefTriangleCenter[orthocenter](A,B,C) \tkzGetPoint{K}
\tkzDefTriangleCenter[centroid](A,B,C) \tkzGetPoint{H}
\tkzDefSpTriangle[excentral, name=J](A,B,C)\tkzGetPoint{J}
\tkzDefSpTriangle[centroid, name=M](A,B,C)\tkzGetPoint{M}
\tkzDefPointBy[reflection= over Ja--Jc](C')\tkzGetPoint{C''}
\tkzDefPointBy[reflection= over Ja--Jc](A')\tkzGetPoint{A''}
\tkzInterLC(A,B)(Q,Cb) \tkzGetFirstPoint{Ba}
\tkzInterLC(A,C)(Q,Cb) \tkzGetPoints{Ac}{Ca}
\tkzInterLC(B,C')(Q,Cb) \tkzGetFirstPoint{Bc}
\tkzInterLC[common=F'a](Sp,F'a)(Ja,F'a) \tkzGetFirstPoint{Fa}
\tkzInterLC[common=F'b](Sp,F'b)(Jb,F'b) \tkzGetFirstPoint{Fb}
\tkzInterLC[common=F'c](Sp,F'c)(Jc,F'c) \tkzGetFirstPoint{Fc}
\tkzInterLC(Mc,Sp)(Q,Cb) \tkzGetFirstPoint{A''}
\tkzDefCircle[euler](A,B,C) \tkzGetPoints{E}{e}
\tkzDefCircle[excircle](C,A,B) \tkzGetPoints{Ec}{c}
\tkzDefCircle[excircle](A,B,C) \tkzGetPoints{Eb}{b}
\tkzDefCircle[excircle](B,C,A) \tkzGetPoints{Ec}{c}
\tkzDrawCircles(Q,Cb E,e)
\tkzDrawCircles(Eb,b Ea,a Ec,c)
\tkzDrawPolygon(A,B,C)
\tkzDrawSegments[dashed](A',A' C',C' A',Zc Za,C' B,Cb B,Ab A,Ca)
\tkzDrawSegments[dashed](C,Ac Ja,Xa Jb,Yb Jc,Zc)
\begin{scope}
\tkzClipCircle(Q,Cb) % We limit the drawing of the lines
\tkzDrawLine[add=5 and 12,orange](K,O)
\tkzDrawLine[add=12 and 28,red!50!black](N,Sp)
\end{scope}
\tkzDefPoints{Ja,Jb,Jc,Xa,Xb,Yb,Yc}
\tkzDefLabelPoints{Ja,Jb,Jc,Xa,Xb,Yb,Yc,Mc}
\tkzDefLabelPoints{Ac,Ab,Ac,Ba,Ac,Ab,Ac,Ba,Ab}
\tkzDefLabelPoints{Ja,Jb,Jc,Xa,Xb,Ya,Yb,Yc,Mc}
\tkzDefLabelPoints{Ac,Ab,Ac,Ba,Ac,Ab,Ac,Ba,Ab}
\tkzDrawSegments(Fc,F'c Fb,F'b Fa,F'a)
\tkzDrawSegments[color=green!50!black](Mc,N,Mc,A'',A'',Q)
\tkzDrawSegments[color=red,dashed](Ac,Ab Ca,Cb Ba,Bc Ja,Jc A',Cb C',Ab)
\tkzDrawSegments[color=red](Cb,Ab Bc,Ac Ba,Ac,Ab)
\tkzDrawSegments[color=red,mark=+] (Cb,Ab Bc,Ac Ba,Ca)
\tkzMarkRightAngles(Jc,Zc,A Ja,Xa,B Jb,Yb,Mc)
\tkzDrawSegments[green,dashed](A,F'a B,F'b C,F'c)
\end{tikzpicture}
Part X.

FAQ
47. FAQ

47.1. Most common errors

For the moment, I’m basing myself on my own, because having changed syntax several times, I’ve made a number of mistakes. This section is going to be expanded. With version 4.05 new problems may appear.

– The mistake I make most often is to forget to put an "s" in the macro used to draw more than one object: like \tkzDrawSegment(s) or \tkzDrawCircle(s) ok like in this example \tkzDrawPoint(A,B) when you need \tkzDrawPoints(A,B);

– Don’t forget that since version 4 the unit is obligatorily the “cm” it is thus necessary to withdraw the unit like here \tkzDrawCircle[R](0,3cm) which becomes \tkzDrawCircle[R](0,3). The traditional options of TikZ keep their units example below right = 12pt on the other hand one will write size=1.2 to position an arc in \tkzMarkAngle;

– The following error still happens to me from time to time. A point that is created has its name in brackets while a point that is used either as an option or as a parameter has its name in braces. Example \tkzGetPoint(A) When defining an object, use braces and not brackets, so write: \tkzGetPoint{A};

– The changes in obtaining the points of intersection between lines and circles sometimes exchange the solutions, this leads either to a bad figure or to an error.

– \tkzGetPoint{A} in place of \tkzGetFirstPoint{A}. When a macro gives two points as results, either we retrieve these points using \tkzGetPoints{A}{B}, or we retrieve only one of the two points, using \tkzGetFirstPoint{A} or \tkzGetSecondPoint{A}. These two points can be used with the reference \tkzFirstPointResult or \tkzSecondPointResult. It is possible that a third point is given as \tkzPointResult;

– Mixing options and arguments; all macros that use a circle need to know the radius of the circle. If the radius is given by a measure then the option includes a R.

– The angles are given in degrees, more rarely in radians.

– If an error occurs in a calculation when passing parameters, then it is better to make these calculations before calling the macro.

– Do not mix the syntax of pgfmath and xfp. I’ve often chosen xfp but if you prefer pgfmath then do your calculations before passing parameters.

– Error "dimension too large" : In some cases, this error occurs. One way to avoid it is to use the "veclen" option. When this option is used in an scope, the "veclen" function is replaced by a function dependent on "xfp". Do not use intersection macros in this scope. For example, an error occurs if you use the macro \tkzDrawArc with too small an angle. The error is produced by the decoration library when you want to place a mark on an arc. Even if the mark is absent, the error is still present.