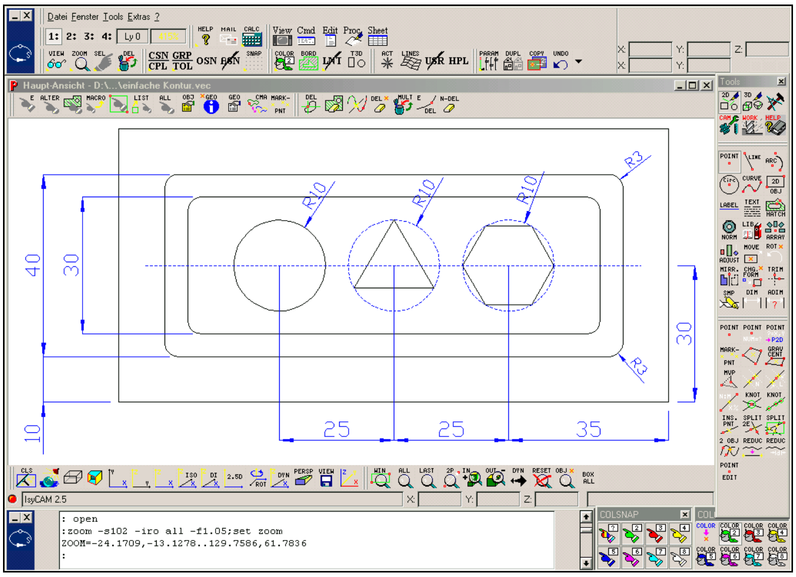
isel®

From Components to Systems


Datei Fenster Tools Extras ?
Haupt-Ansicht - D:\...\einfache Kontur.vec
R10
R10
R10
R3
R3
40
30
30
10
25
25
35
IsyCAM 2.5
COLSNAP

```
: open
:zoom -s102 -iro all -f1.05;set zoom
ZOOM=-24.1709,-13.1278..129.7586,61.7836
:
```

isyCAM 2.5 (light)

Manual Part 2
"The CAD Part"

isel Germany AG, D-36124 Eichenzell, Bürgermeister-Ebert-Str. 40 ☎ (06659)981-0 📠 (06659)981-776

About this manual:

Various symbols are to be found in this manual to alert you to important information.

CAUTION:	**NOTE:**	**EXAMPLE:**
!	☞	□○△

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the prior written consent of iselautomation KG.

We do not assume any liability or warranty that the manual, the program package
isyCAM 2.5 (light) or the supplementary tools provided are free of errors or suited for any special purpose. Any legal responsibility or liability for collateral damage shall be excluded.

All information provided in this manual is delivered without warranty. Subject to change without prior notice.

Manufacturer: ***isel Germany AG***
Buergermeister-Ebert-Straße 40
D-36124 Eichenzell

Phone: +49 (0) 6659 981-0
Fax: +49 (0) 6659 981-776
E-mail: automation@isel.com
http://www.isel.com

Version: 24/2004

Content

2D-Drawing

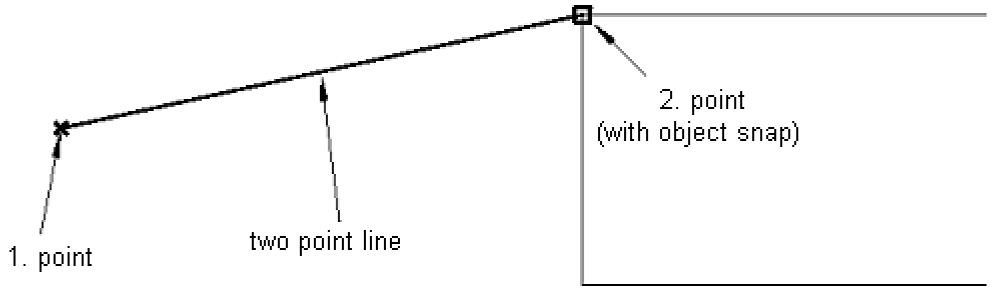
Lines

The construction of several 2D lines is carried out with the submenu in "Tools".

Menu - Button: 2D + LINE

Line between two points

Button: 2D + LINE + LINE 2P
Command: rb -2 -o? (-2 = 2-points, -o? = with object snap, -3 = 3D)



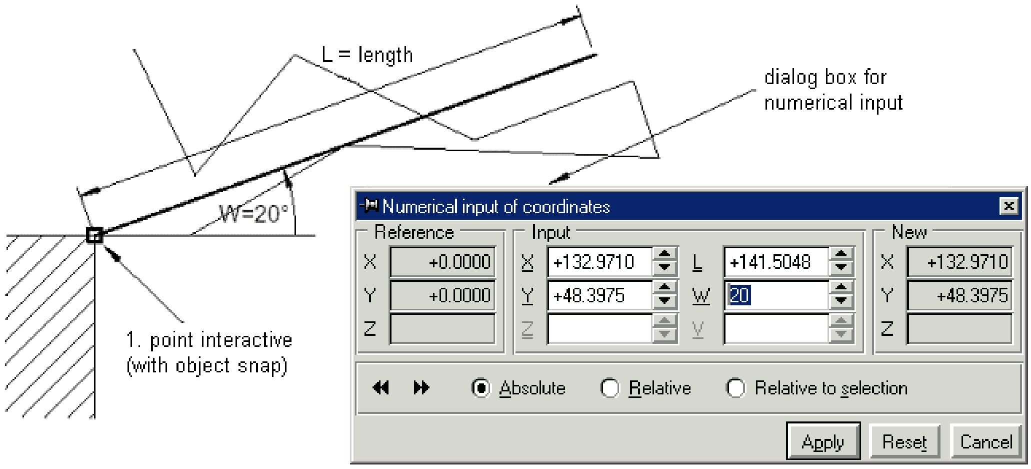
Proceeding:

1. Choose the command (with button)
2. Select the first point
3. Select the second point

Hints:
The point selection can be carried out numerically, key: n

Line from a point with specification of angle and length

Button: 2D + LINE + LINE 2P
Command: rb -2 -o? (-2 = 2 points, -o? = with object snap, -3 = 3D)



Numerical input of coordinates

Reference		Input				New	
X	+0.0000	X	+132.9710	L	+141.5048	X	+132.9710
Y	+0.0000	Y	+48.3975	W	20	Y	+48.3975
Z		Z		V		Z	

● Absolute ○ Relative ○ Relative to selection

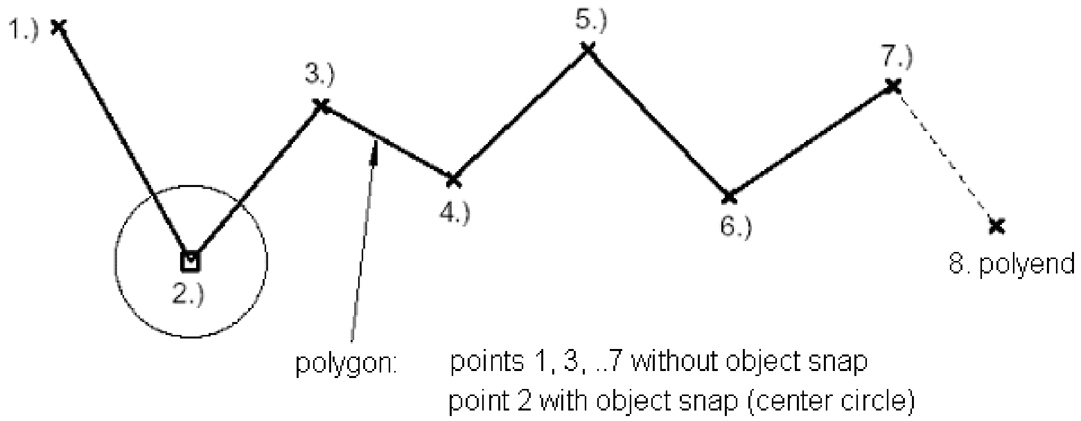
Apply Reset Cancel

Proceeding:
1. Choose the command (with button)
2. Interactive selection of the first point
3. Choose on keyboard "w"
4. Enter the angle in the dialog window and "Apply"

Hints:
The point selection can be carried out numerically, key: n

Polygon lines between selected points

Button: 2D + LINE + POLY
Command: rb -o? (-o? = with object snap)


1.)
2.)
3.)
4.)
5.)
6.)
7.)
8. polyend
polygon:
points 1, 3, ..7 without object snap
point 2 with object snap (center circle)

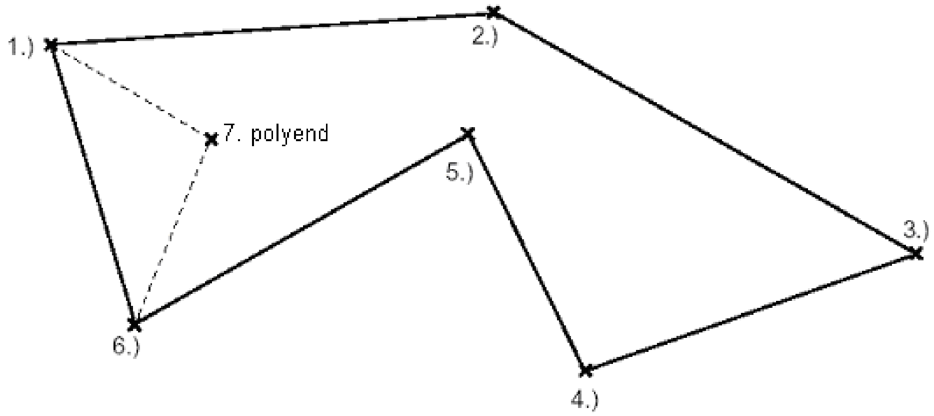
Proceeding:
1. Choose the command (with button)
2. Interactive selection of the first point
3. Selection of more points
4. Complete with "Polyend" (<F6>)

Hints:
The point selection can be carried out numerically, key: n

Closed polygon between selected points

Button: 2D + LINE + POLY

Command: rb -c -o? (-c = closed, -o? = with object snap)



Proceeding:

1. Choose the command (with button)
2. Interactive selection of the first point
3. Selection of more points
4. Complete with "Polyend" (<F6>)

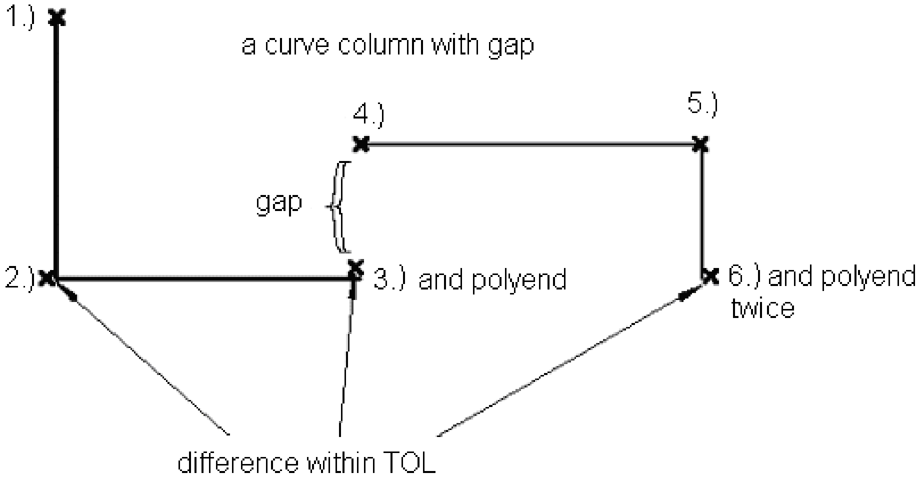
Hints:

The point selection can be carried out numerically, key: n

Perpendicular lines (multiple functions)

Button: 2D + LINE + ORTHO

Command: al -p (-p = perpendicular)



Proceeding:

1. Choose the command (with button)
2. Interactive selection of the first point
3. Complete the part with single "Polyend"
4. Complete the function with double "Polyend"

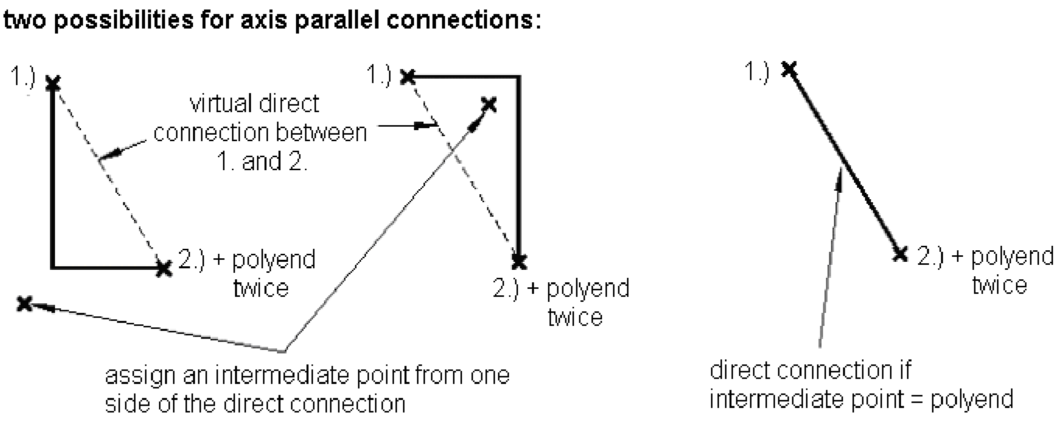
Hints:

Line is produced if the axis deviation of the each second point lies within the tolerance (TOL). Otherwise: Choose intermediate point!

Perpendicular lines with intermediate points

Button: 2D + LINE + ORTHO

Command: al -p (-p = (perpendicular)


two possibilities for axis parallel connections:
1.)
virtual direct connection between 1. and 2.
1.)
2.) + polyend twice
2.) + polyend twice
assign an intermediate point from one side of the direct connection
1.)
2.) + polyend twice
direct connection if intermediate point = polyend

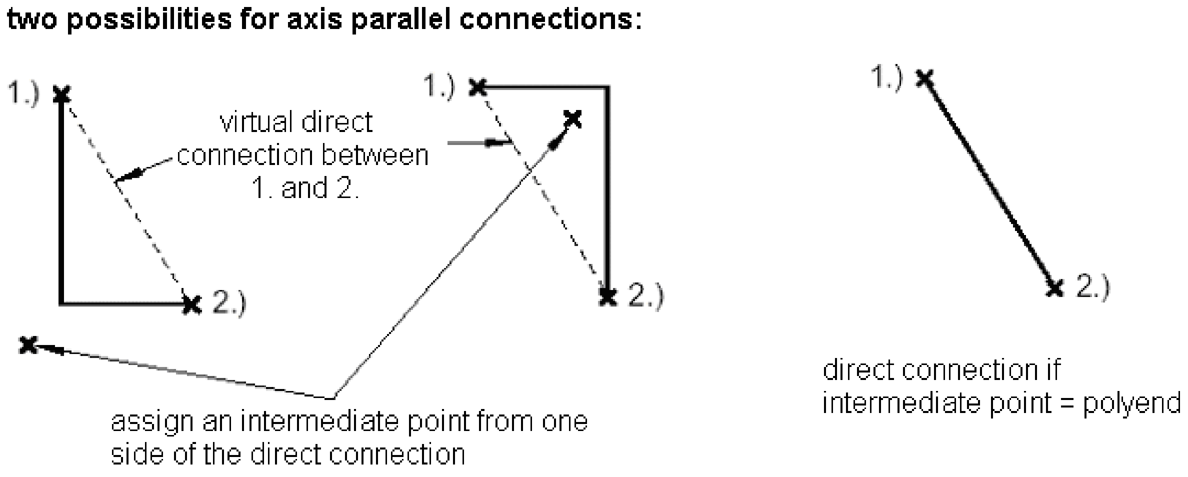
Proceeding:
1. Choose the command (with button)
2. Select first point
3. Select second point (outside from TOL)
4. Choose intermediate point, complete with double "Polyend"

Hints:
The choice of the intermediate point determines the position of perpendicular lines at tolerance overstepping.

Perpendicular 2-point line

Button: 2D + LINE + ORTHO 2P

Command: al -sp (-s = (single) 2-point line, -p = (perpendicular)


two possibilities for axis parallel connections:
1.)
virtual direct connection between 1. and 2.
1.)
2.)
2.)
assign an intermediate point from one side of the direct connection
1.)
2.)
direct connection if intermediate point = polyend

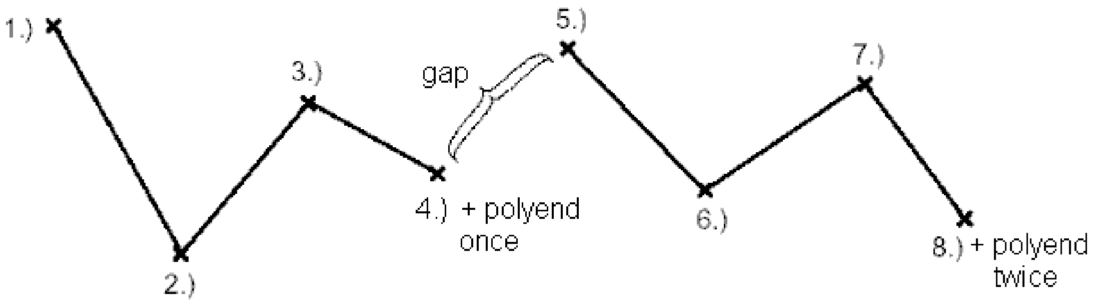
Proceeding:
1. Choose the command (with button)
2. Select first point
3. Select second point (outside from TOL)
4. If necessary, choose intermediate point

Hints:
The choice of the intermediate point determines the position of perpendicular lines at tolerance overstepping.

Connected separate lines

Button: 2D + LINE + AL

Command: al



separate lines with gaps lined each other in a column

Proceeding:

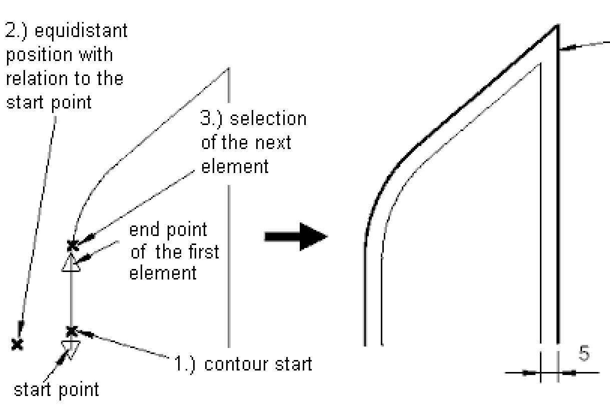
1. Choose the command (with button)
2. Interactive selection of the first point
3. Complete a part with single "Polyend"
4. Complete the function with double "Polyend"

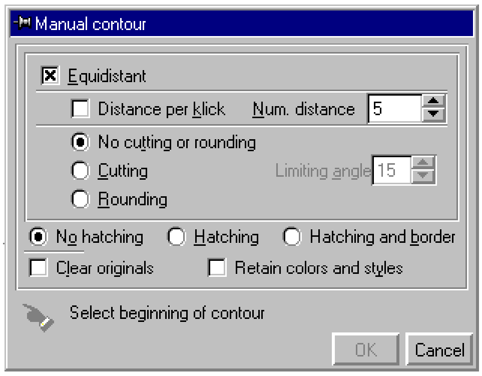
Hints:
Each line between two connected points is an object.

Equidistant line with manual contour offset

Button: 2D + LINE + EQUT

Command: econtourDLG -e (-e = equidistant)




Manual contour

☒ Equidistant

□ Distance per klick Num. distance 5

● No cutting or rounding

○ Cutting Limiting angle 15

○ Rounding

● No hatching ○ Hatching ○ Hatching and border

□ Clear originals □ Retain colors and styles

Select beginning of contour

Proceeding:

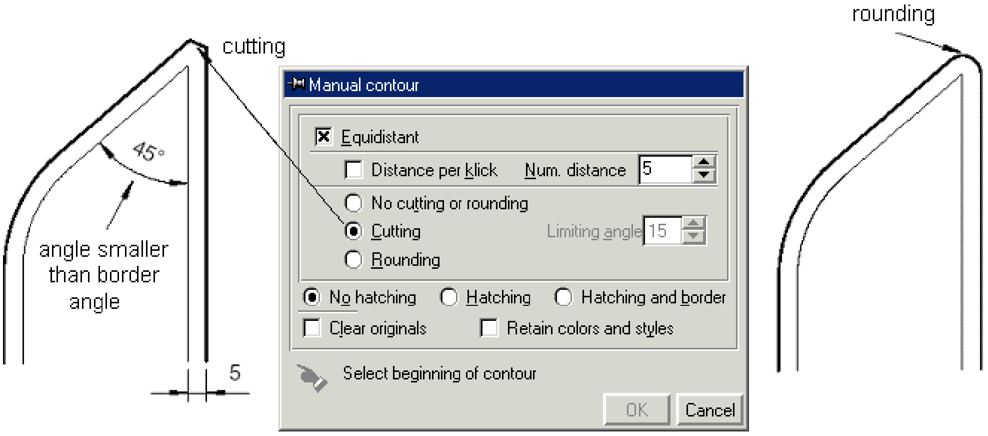
1. Choose command (with button)
2. Select the first contour element
3. Position of the equidistant with respect to the start point
4. Select the each next element, complete with "Polyend"

Hints:
Select the next element near the last end-point.

Equidistant with cutting or rounding

Button: 2D + LINE + EQUI

Command: econtourDLG -e (-e = equidistant)



Manual contour

☒ Equidistant
□ Distance per klick   Num. distance 5
○ No cutting or rounding
● Cutting   Limiting angle 15
○ Rounding
● No hatching ○ Hatching ○ Hatching and border
□ Clear originals □ Retain colors and styles
Select beginning of contour
OK Cancel

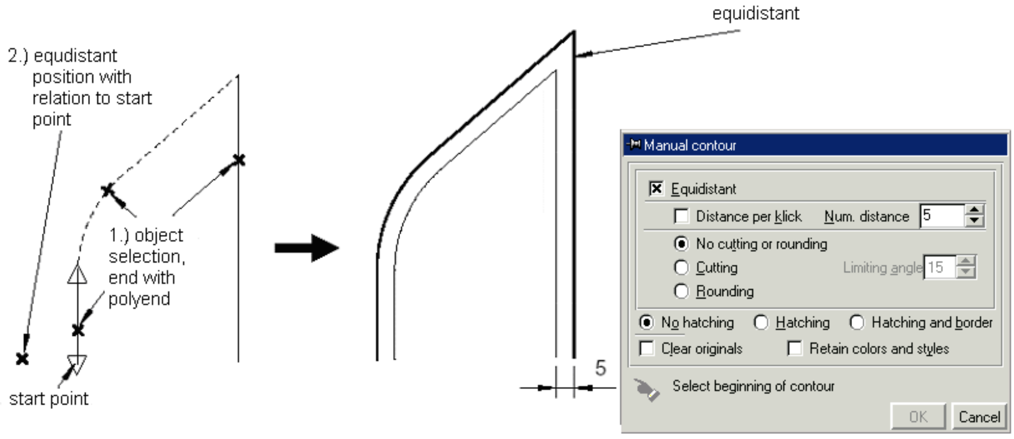
Proceeding:
1. Choose command (with button)
2. Enter parameter in the dialog window
3. Select according request
4. Complete with "Polyend" at contour end

Hints:
Select the next element near the last end point.

Automatic equidistant

Button: 2D + LINE + AUTO

Command: acontourDLG -e (-e = equidistant)


2.) equdistant position with relation to start point
1.) object selection, end with polyend
start point
equidistant
5

Manual contour

☒ Equidistant
□ Distance per klick   Num. distance 5
● No cutting or rounding
○ Cutting   Limiting angle 15
○ Rounding
● No hatching ○ Hatching ○ Hatching and border
□ Clear originals □ Retain colors and styles
Select beginning of contour
OK Cancel

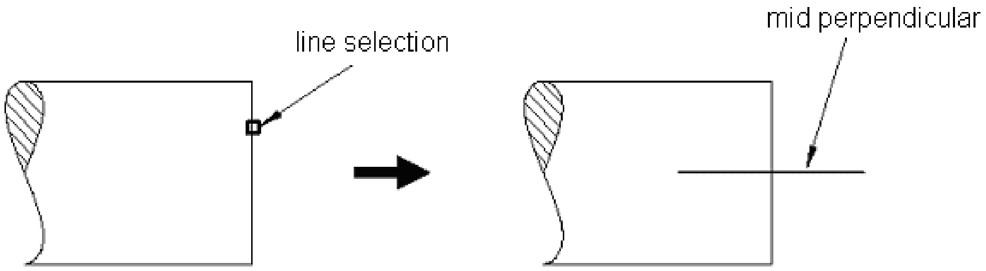
Proceeding:
1. Choose command (with button), enter options in dialog window
2. Select all contour objects (arbitrary order)
3. Complete the selection with „Polyend“
4. Position of the equidistant according to start point

Hints:
If there are gaps on the contour, the respective position of the equidistant lines must be decided for every connected part.

Mid perpendicular on a line

Button: 2D + LINE + ┼

Command: angpl -m (-m = mid perpendicular)



Proceeding:

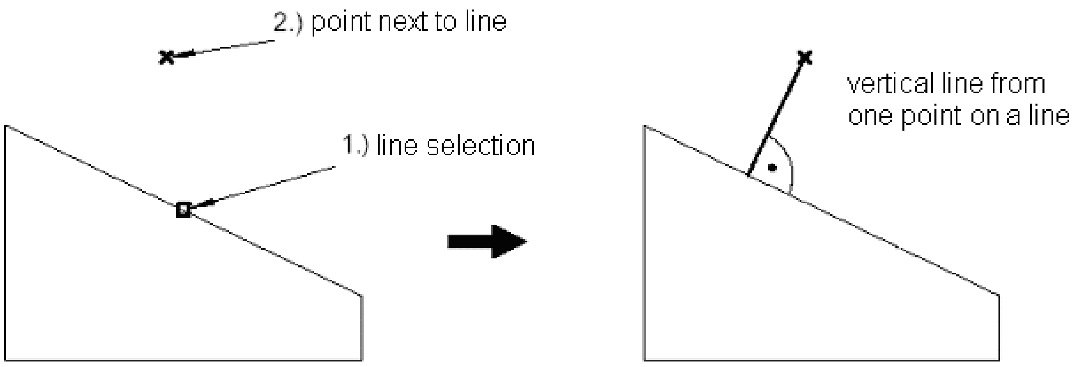
1. Choose command (with button)
2. Select a line

Hints:
The snap is only possible on lines independent of GROUP

Perpendicular on a line

Button: 2D + LINE + ⊥

Command: angpl



Proceeding:

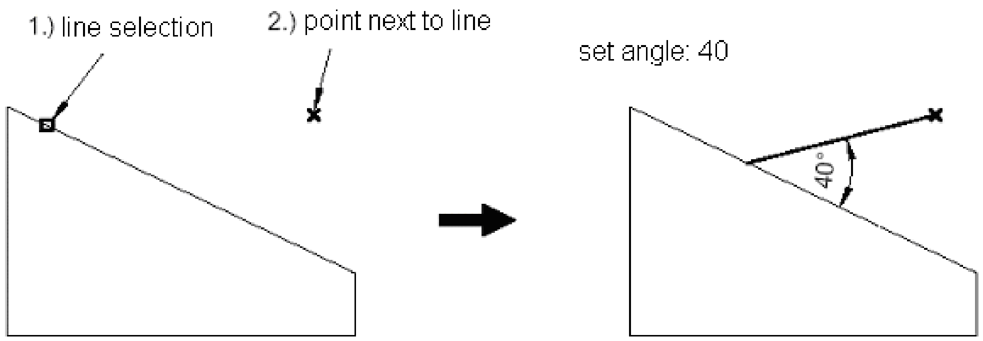
1. Choose command (with button)
2. Select a line
3. Select a point near the line

Hints:
Selecting the corresponding points, the perpendicular will be passed on the "imaginary" elongation of the line.

Inclined line from a point on a line

Button: 2D + LINE + α=?

Command: angpl " " ? (instead of "?" angle can be entered directly)



Proceeding:

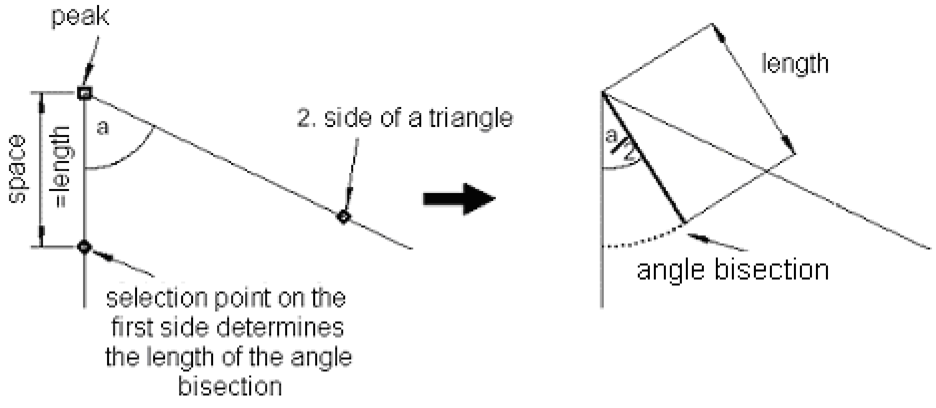
1. Choose command (with button)
2. Select a line
3. Select a point near the line

Hints:
Selecting the corresponding points, the inclined line will be drawn on the "imaginary" elongation of the line.

Draw a bisecting line

Button: 2D + LINE +

Command: halfang


peak
a
space
=length
2. side of a triangle
selection point on the first side determines the length of the angle bisection
length
a
a/2
angle bisection

Proceeding:

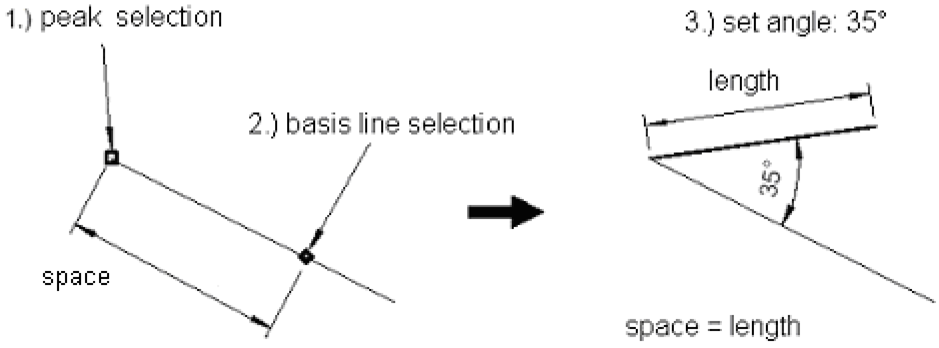
1. Choose command (with button)
2. Select the peak
3. Select the 1. element (determines length)
4. Select the 2. element

Hints:
Please notice: The order of selection has to be counter-clockwise !

Construction of an angle

Button: + +

Command: mkang (The angle can be entered directly as parameter.)



Proceeding:

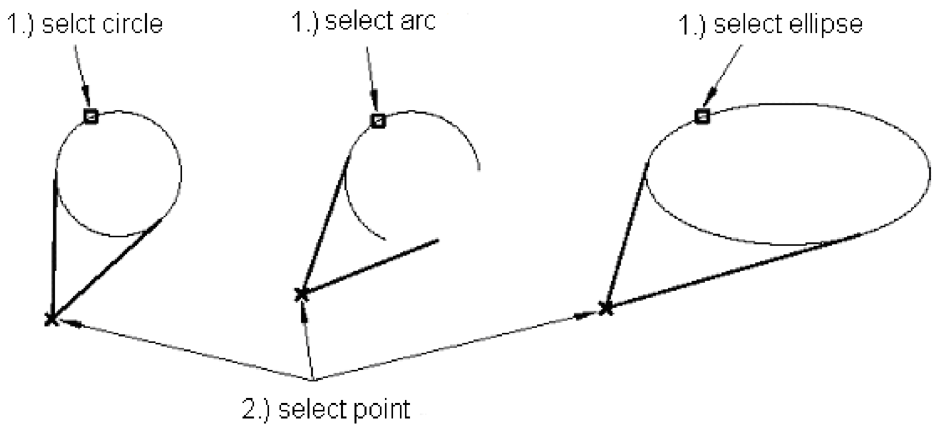
1. Choose command (with button)
2. Select the peak
3. Select the basic line
4. Enter the angle (relative)

Hints:
The angle designation is relative according to the position of the basic line.

Tangent from a point to a circle (ellipse)

Button: + +

Command: tang



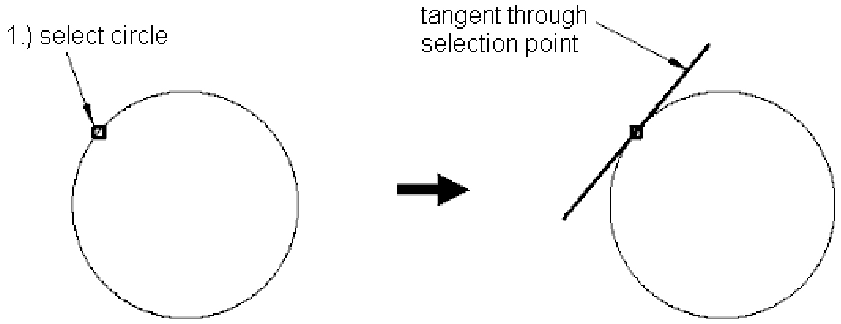
Proceeding:

1. Choose command (with button)
2. Select circle, ellipse or arc
3. Select point

Hints:
If one of the two tangents is not required, you can delete this tangent

Tangent through a circle point

Button: 2D + LINE +
Command: tang -c

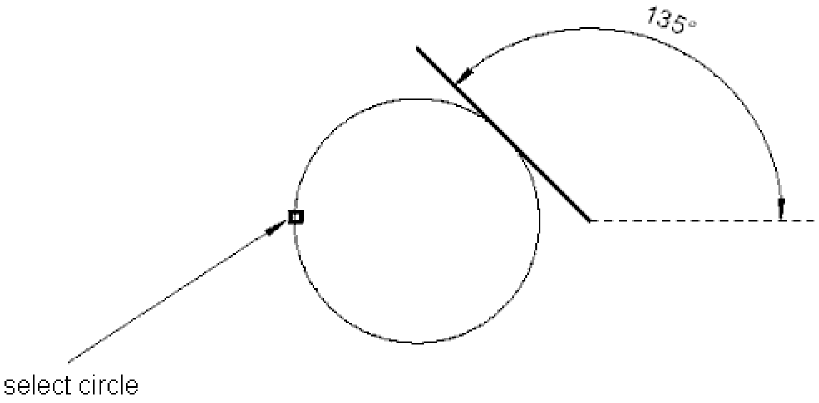


Proceeding:
1. Choose command (with button)
2. Select circle, ellipse or arc

Hints:
The length of the tangent corresponds to the diameter (for ellipses to the greater diameter).

Sloped tangent through a circle point

Button: 2D + LINE +
Command: tang -s 135 (-s = slope with parameter angle)

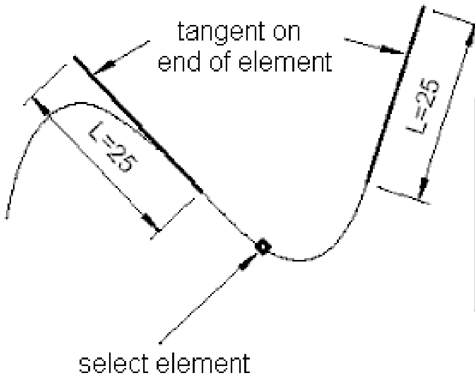


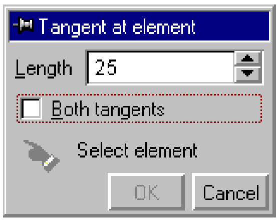
Proceeding:
1. Choose command (with button)
2. Enter angle (absolute) e. g. 135°
3. Select circle, ellipse or arc

Hints:
The length of the tangent corresponds to the diameter (for ellipses to the greater diameter).

Tangent(s) at the end of an element with length input

Button: + +
Command: eetang




Tangent at element

Length 25

□ Both tangents

Select element

OK Cancel

Proceeding:

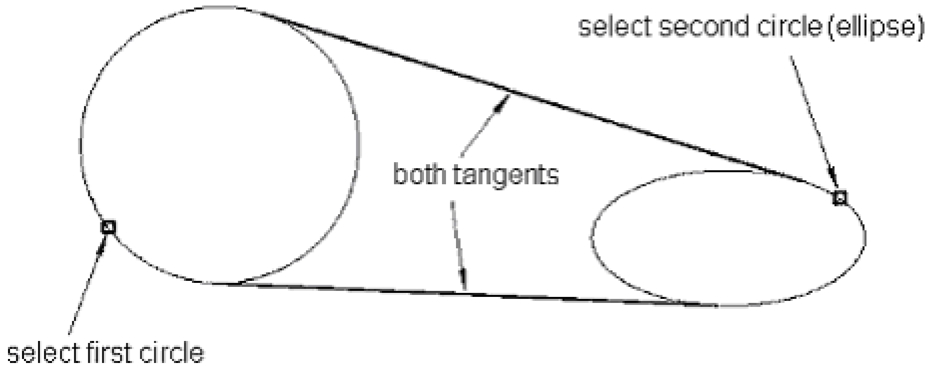
1. Choose command (with button)
2. Enter parameter into the dialog window
3. Select the element

Hints:

Please consider the difference between object and element!

Tangents to two circles

Button: + +
Command: tang2c



Proceeding:

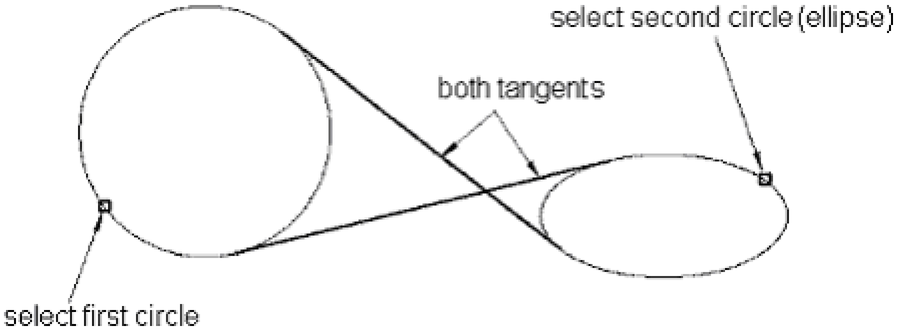
1. Choose command (with button)
2. Select first circle, ellipse or arc
3. Select second circle, ellipse or arc

Hints:

If one of the two tangents is not required, you can delete this tangent

Crossed tangents to two circles

Button: + +
Command: tang2c -x (-x = crossed tangents)



Proceeding:

1. Choose command (with button)
2. Select first circle, ellipse or arc
3. Select second circle, ellipse or arc

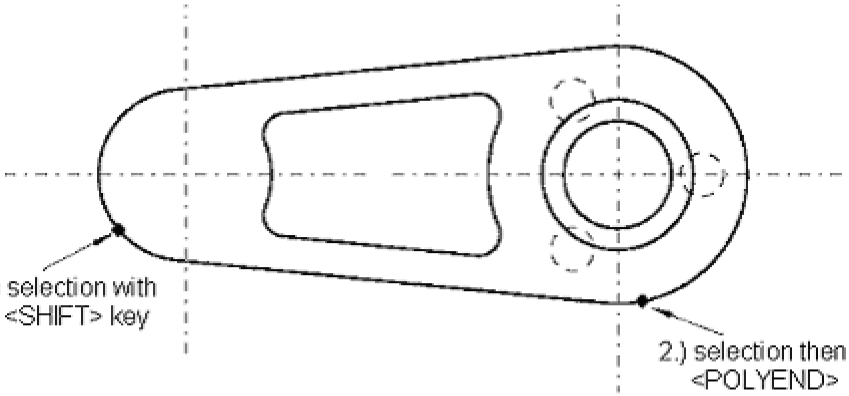
Hints:

If one of the two tangents is not required, you can delete this tangent

Centre lines to 2D-circle arcs

Button: + +

Example 1:
Command: centline

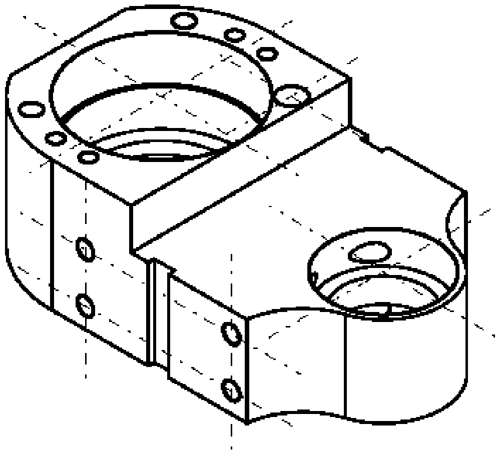


Proceeding:

1. Choose command (with button)
2. Select circle (-arc), several with <SHIFT>
3. Activate macro if necessary with <F10> (after <SHIFT>)
4. <POLYEND>, if selection with <SHIFT>, <F10>

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Example 2:
Command: centline



Proceeding:

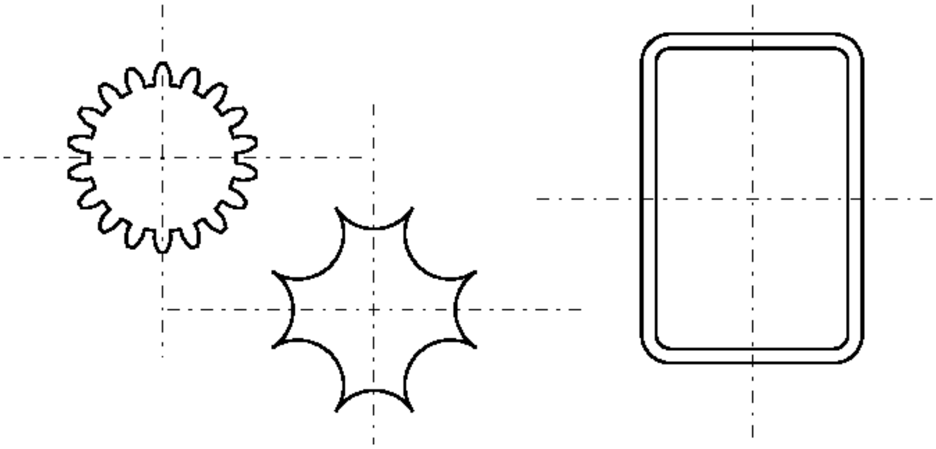
1. Choose command (with button)
2. Select circle (-arc), several with <SHIFT>
3. Activate macro if necessary with <F10> (after <SHIFT>)
4. <POLYEND>, if selection with <SHIFT>, <F10>

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Centre lines to an object (box)

Button: 2D + NORM +

Command: centline -o (-o: objects)



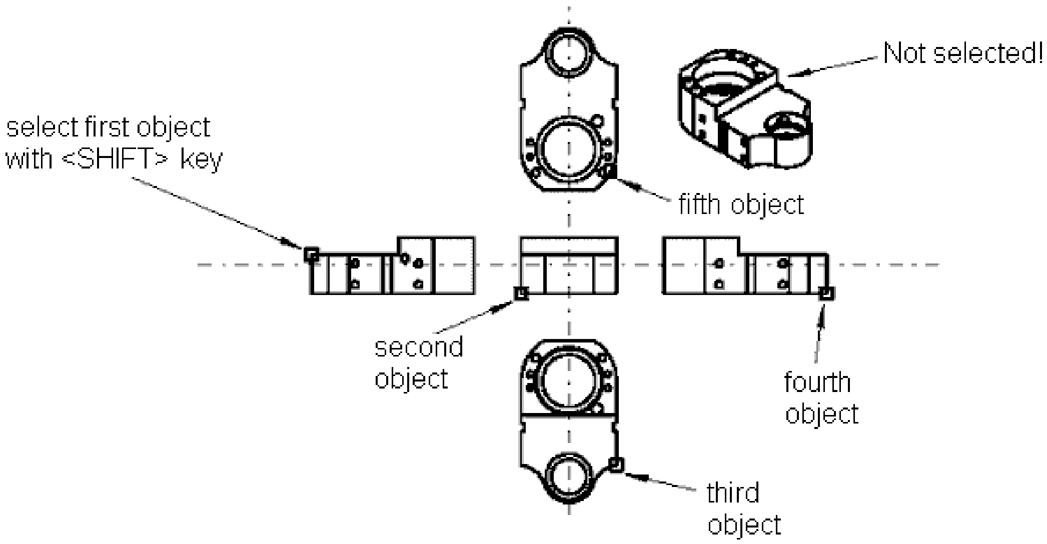
Proceeding:

1. Choose command (with button)
2. Select circle (-arc), several with <SHIFT>
3. Activate macro if necessary with <F10> (after <SHIFT>)
4. <POLYEND>, if selection with <SHIFT>, <F10>

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Axes of coordinates to a centre of several objects

Command: centline -l (-l: list of objects)



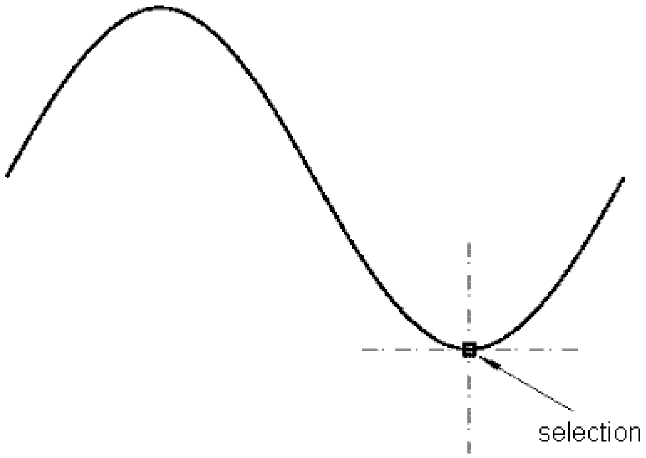
Proceeding:

1. Choose command (with button)
2. Select circle (-arc), several with <SHIFT>
3. Activate macro if necessary with <F10> (after <SHIFT>)
4. Complete with <POLYEND>

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Axes of coordinates to a arbitrary point

Command: centline -p (-p: point)



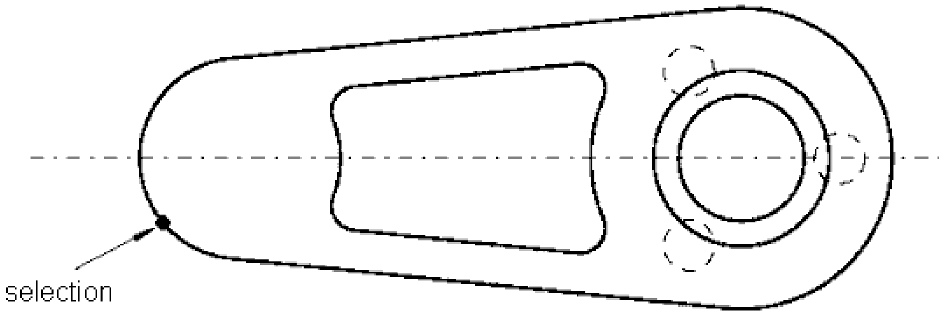
Proceeding:

1. Choose command (with button)
2. Select point

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Horizontal centre line of an object

Command: centline -oh (-o: object, -h: horizontal)



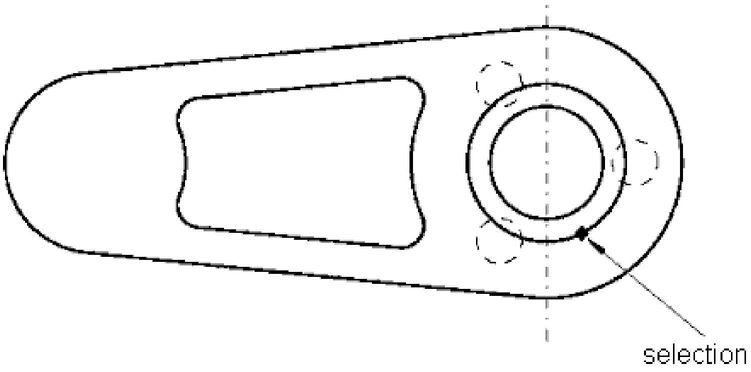
Proceeding:

1. Choose command
2. Select circle (-arc), several with <SHIFT>
3. Activate macro if necessary with <F10> (after <SHIFT>)
4. Complete with <POLYEND>

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Vertical centre line of an object

Command: centline -ov (-o: object, -v: vertical)



Proceeding:

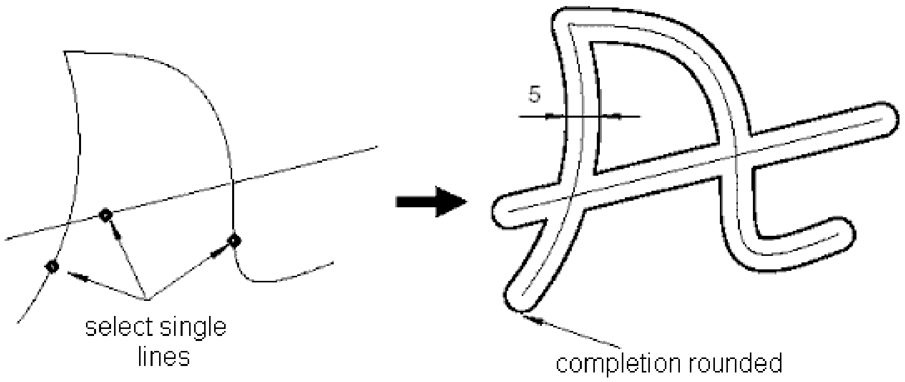
1. Choose command
2. Select circle (-arc), several with <SHIFT>
3. Activate macro if necessary with <F10> (after <SHIFT>)
4. Complete with <POLYEND>

Hints:
Line type is adjustable in variable CENTLINE_TYPE, colour in CENTLINE_COLOUR.

Automatic booth sided equidistant of single lines

Button: 2D + LINE + MORE LINE +

Command: wlines -w 5 (-w = width of the lines, without "-w": width will be queried)



Proceeding:

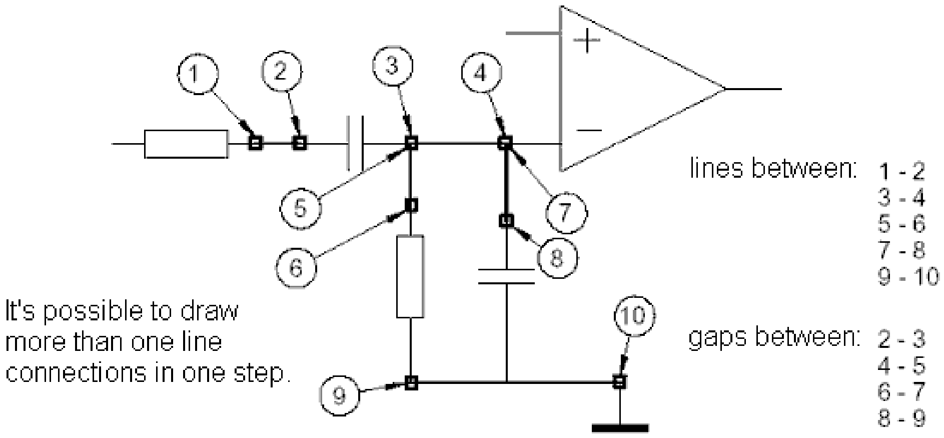
1. Choose command
2. Enter width: (e.g. 5)
3. Select lines
4. Complete with “POLYEND“

Hints:
Valid geometries ("lines"): Lines, polygon, circles, ellipses, arcs and 4 point bezier.

Discontinued polygon (line, gap, line etc.)

Button: 2D + LINE + MORE LINE +

Command: mpoly -m1 -o all (-m1 = discontinuation, -o = object snap)



It's possible to draw more than one line connections in one step.

Proceeding:

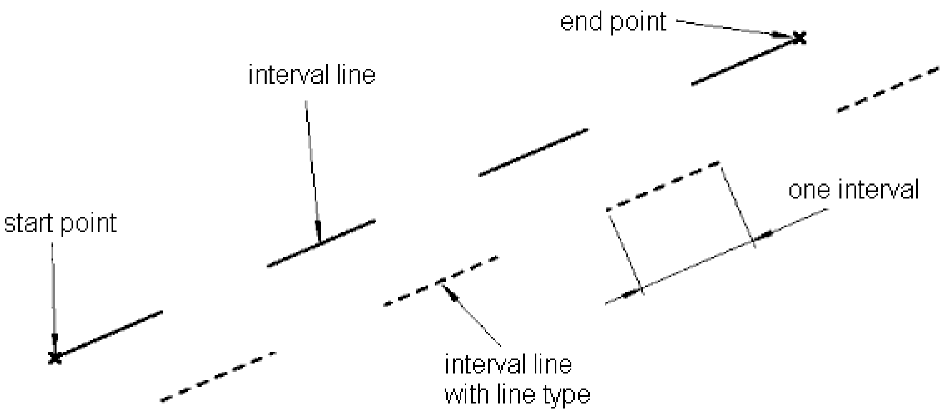
1. Choose command
2. Select points
3. Complete with “POLYEND“

Hints:
Lines between 1. and 2., 3. and 4., 5. and 6., etc. points. Gaps between 2. and 3., 4. and 5., etc. points.

Regular discontinued 2-point line with intervals

Button: 2D + LINE + MORE LINE + N=?

Command: linetype * # (# = uneven number of intervals, e.g.: 7)



Proceeding:

1. Choose command
2. Enter the number of intervals
3. Select start point
4. Select end point

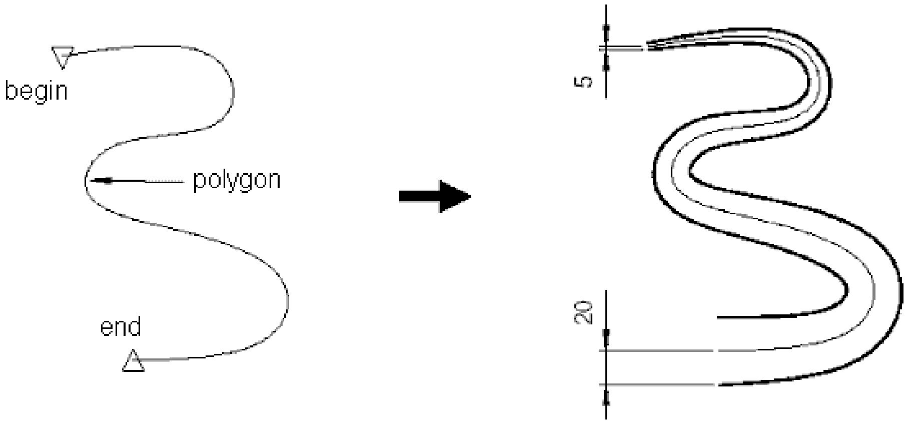
Hints:

Please do not mistake the interval line for a line with variable LINETYPE !

Widening line (double-sided) at a polygon

Button: 2D + LINE + MORE LINE +

Command: equi " " 5..20 -ms (-ms = begin and end are marked)



Proceeding:

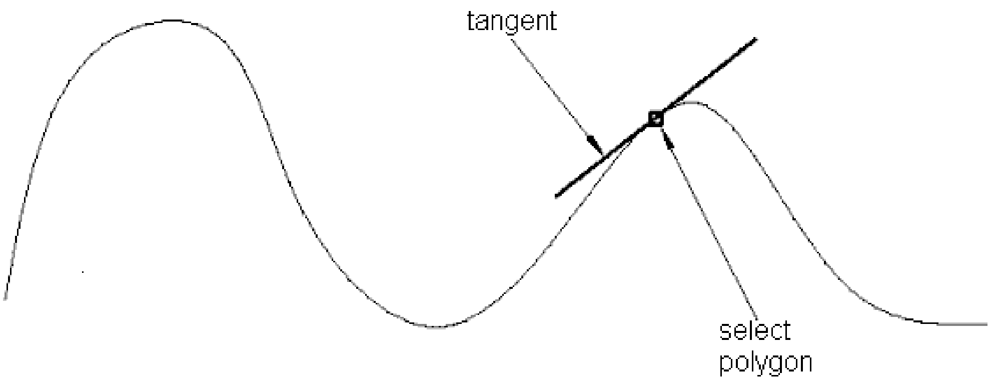
1. Choose command (with button)
2. Enter value of widening (e.g. 5 .. 20)
3. Select polygon
4. Complete with „Polyend“ (<F6> or right mouse key)

Hints:

The value for the widening refers of each side.

Tangent through a polygon point

Command: tangp -c (-c = curve)



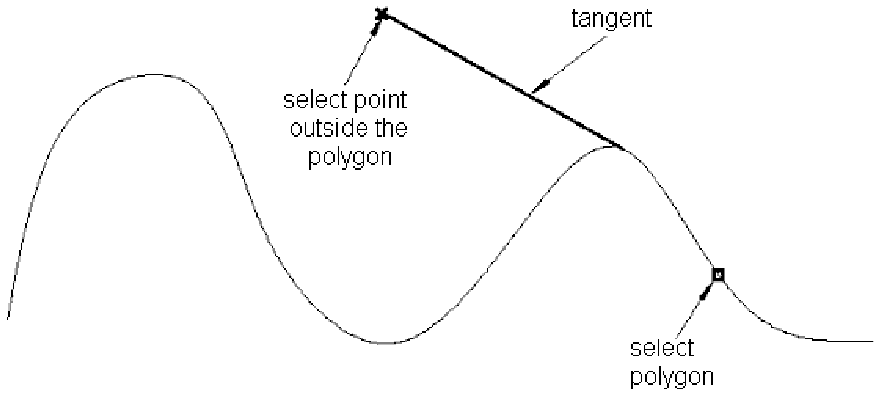
Proceeding:
1. Choose command
2. Select point on polygon

Hints:
The length of the tangent is dependent on the bending of the polygon. (high bending = short tangent)

Tangent to a smoothed polygon through point outside

Button: 2D + LINE + MORE LINE +

Command: tangp



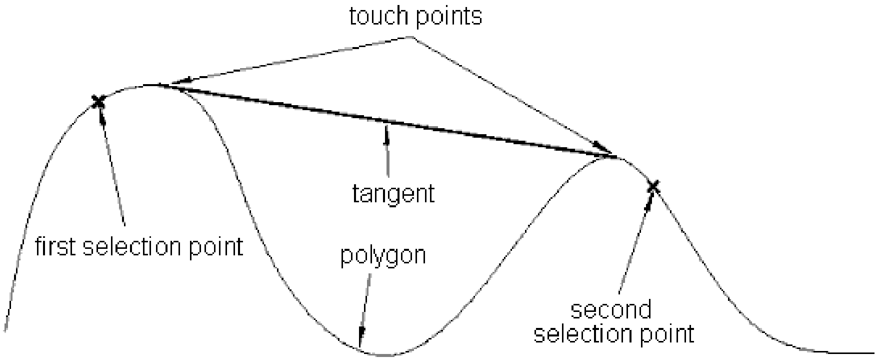
Proceeding:
1. Choose command (with button)
2. Select polygon
3. Select point outside of the polygon

Hints:
The choice of the selection points affects the position of the tangent.

Tangent to two elements (also polygon)

Button: 2D + LINE + MORE LINE +

Command: t2e



Proceeding:

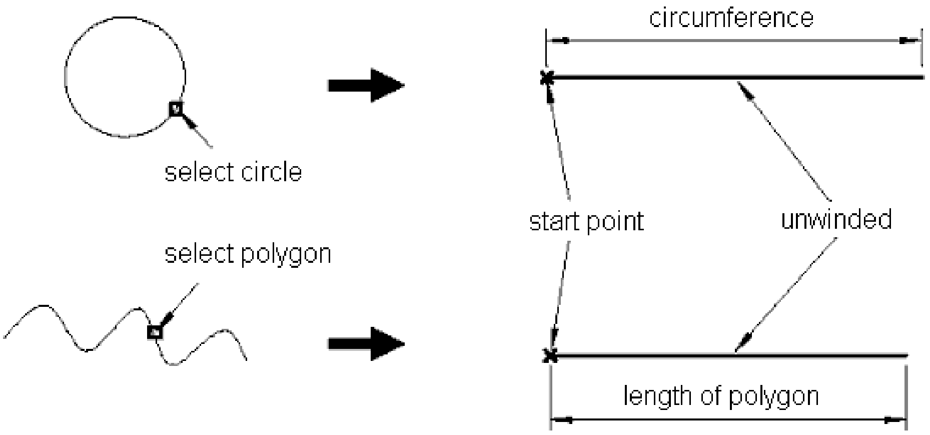
1. Choose command (with button)
2. Select first point
3. Select second point

Hints:
Please choose the selection points so that the touch points lie between the selection points.

Wind ellipses or polygons

Button: 2D + LINE + MORE LINE + spool

Command: wind



Proceeding:

1. Choose command (with button)
2. Select object
3. Determine start point to wind

Hints:
Valid objects: Circles and polygons

Curves

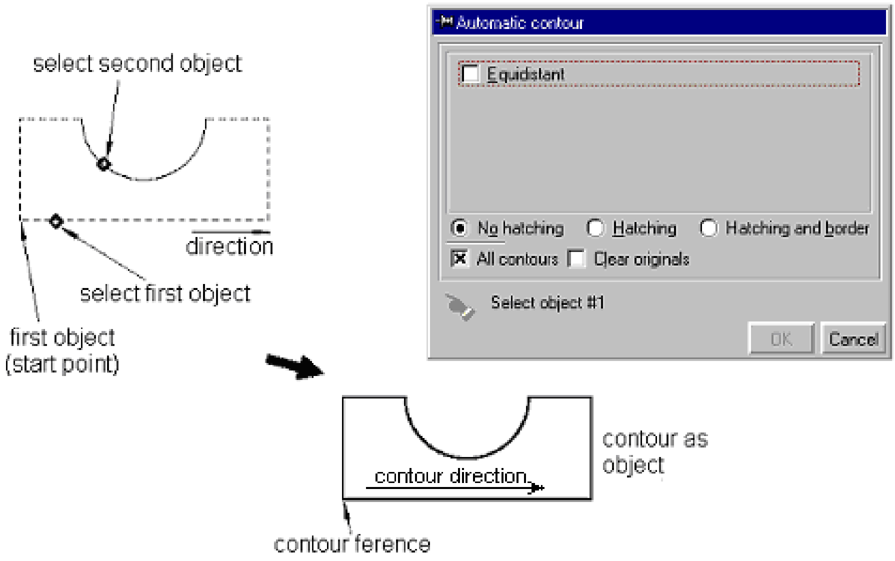
The construction of several curves is carried out with the submenu in "Tools".

Menu - Button: 2D + CURVE

Automatic contour generation

Button: 2D + CURVE + AUTO

Command: acontourDLG



Automatic contour

☐ Equidistant

● No hatching ○ Hatching ○ Hatching and border

☒ All contours ☐ Clear originals

Select object #1

OK Cancel

Proceeding:

1. Choose the function with button or enter the command
2. Select the 1. object
3. Select further objects (arbitrary order)
4. Complete with "POLYEND"

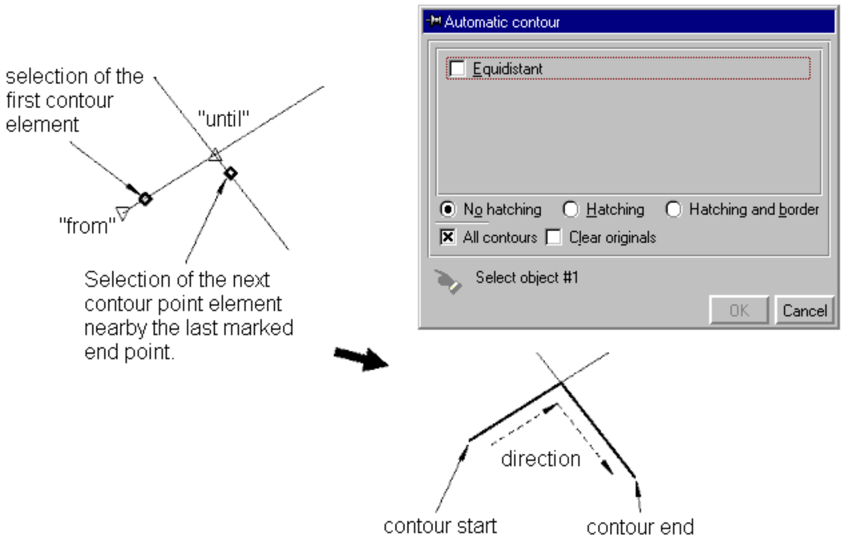
Hints:

The 1. object determines the direction and the start point of the contour. The contour objects may not have any forks.

Manual contour generation

Button: 2D + CURVE +

Command: econtourDLG


selection of the first contour element
"until"
"from"
Selection of the next contour point element nearby the last marked end point.
direction
contour start
contour end

Automatic contour

☐ Equidistant

● No hatching ○ Hatching ○ Hatching and border

☒ All contours ☐ Clear originals

Select object #1

OK Cancel

Proceeding:

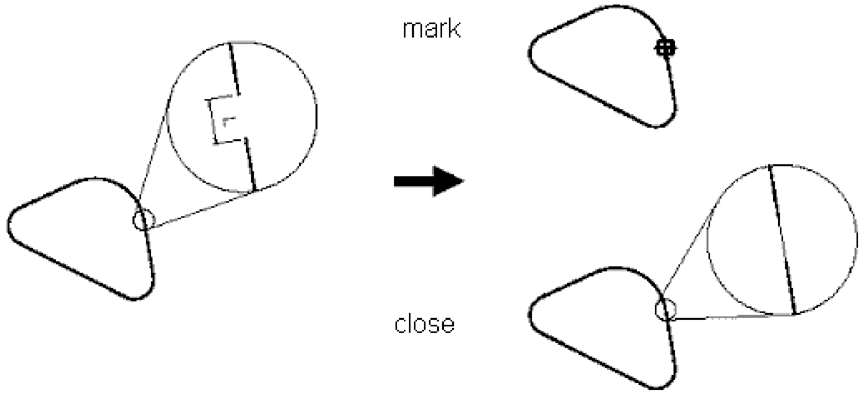
1. Choose the function with button or enter the command
2. Select the 1. contour object (near beginning of contour)
3. Select next contour object (and so on)
4. Complete contour with "POLYEND"

Hints:
The contour represents a new object. The elements. The elements must be split beforehand at the intersection or branch point.

Gaps at 2D-contours mark/close

Button: 2D + CURVE + GAP

Command: contgaps



After deleting the marked objects, the gap is still existing.

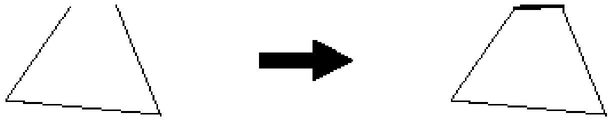
Proceeding:

1. Choose the function with button or enter the command
2. Adjust the parameter (more "Parameter for gaps")
3. Select the elements of the contour (<F10> -> for the complete object respectively macro)

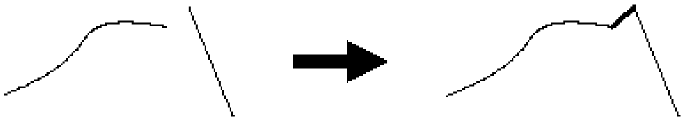
Hints:
All 2D-geometries are supported, gap size is shown.

Parameter for gaps

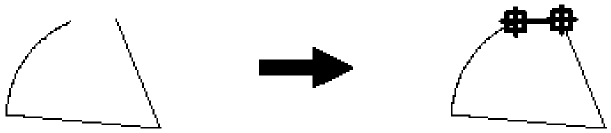
"Maximum gap":
Larger gaps are ignored (useful e.g. at opened contours)



"Mark":
Generation of a marking object, no connecting with line



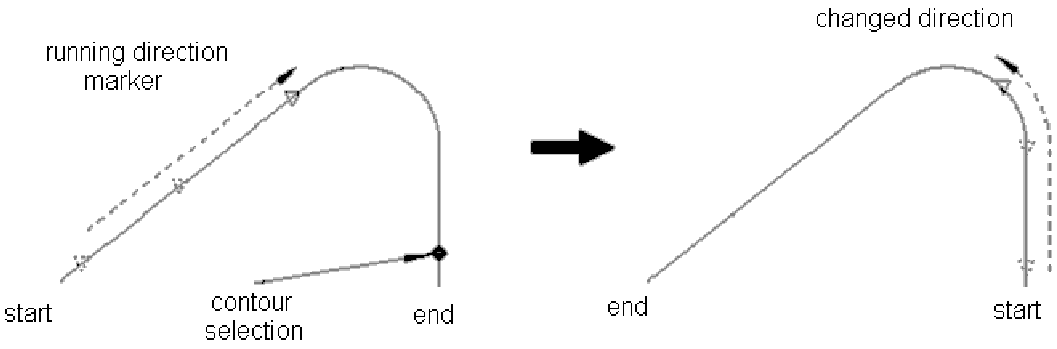
"Connect direct":
Smaller gaps are ignored, if “Mark" is deactivated, gaps are connected with lines until the limit of "maximal gap"



Reverse the direction of a contour

Button: 2D + CURVE + CONT. REV.

Command: revco



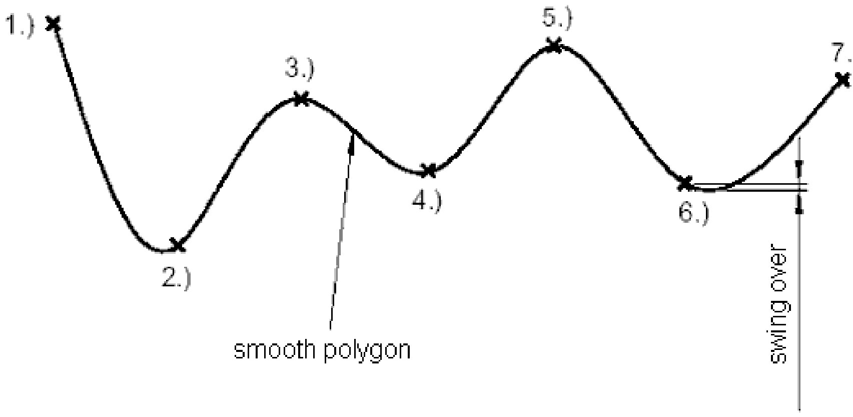
Proceeding:
1. Choose the function with button or enter the command
2. Select contour

Hints:
The direction of the selected contour is reversed. The new direction will be shown by a new marking.

Spline curve as polygon

Button: 2D + CURVE + POLY

Command: spl all (all = snap to all objects)



Proceeding:

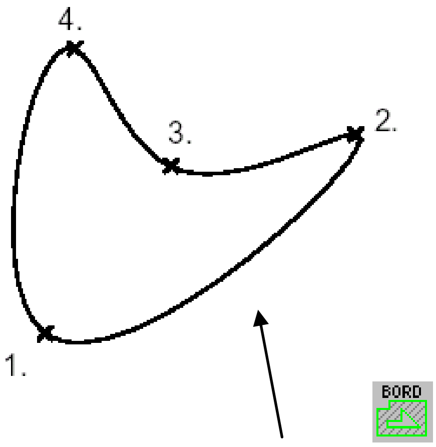
1. Choose command (with button)
2. Select the first point interactively
3. Select additional points (in all minimum three)
4. Complete with "POLYEND"

Hints:
The B-spline curve connects all selected points. However, it can also over swing!

Closed, periodical spline curve

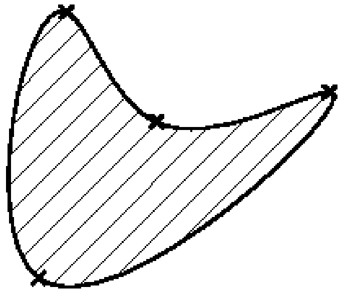
Button: 2D + CURVE + ○

Command: spl all -cp (-c = closed, -p = periodical, -hb = hatch with border)



BORD

periodical, closed smoothing polygon



with hatch option (-hb) BD&HT

Proceeding:

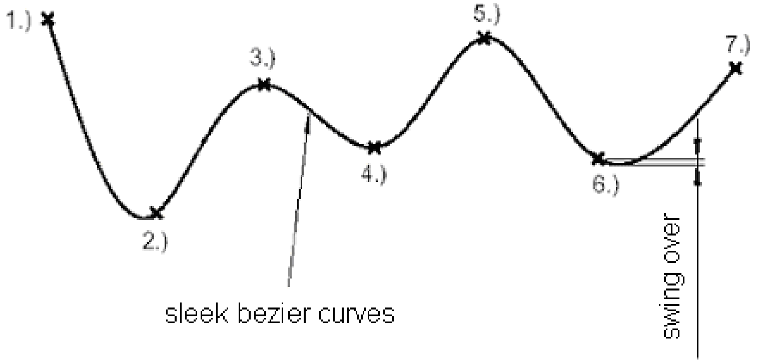
1. Choose command (with button)
2. Interactive selection of points (minimum three)
3. Complete with "POLYEND"

Hints:
The B-spline curve connects all selected points. However, it can also over swing!

Spline curve with 4-point beziers

Button: 2D + CURVE + 4PtBZ

Command: spl all -z (-z = Beziers)



Proceeding:

1. Choose command (with button)
2. Interactive selection of points (minimum three)
3. Complete with "POLYEND"

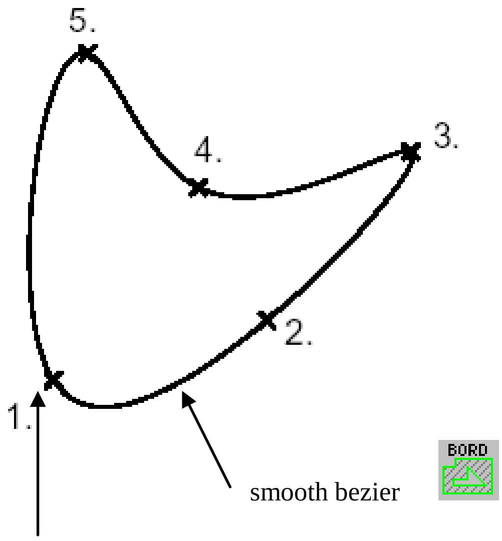
Hints:

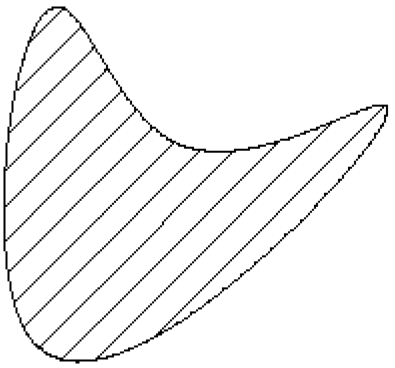
Continuous (smooth) bezier curves are generated between the point .

Periodical, closed spline curve with 4-point beziers

Button: 2D + CURVE + 4Pt BZ

Command: spl all -zcp (-z = bezier, -c = closed, -p = periodical)





smooth bezier BORD

with hatch option (-hb) BD&HT

periodical closed (smooth crossing between start- and end point)

Proceeding:

1. Choose command (with button)
2. Interactive selection of points (minimum three)
3. Complete with "POLYEND"

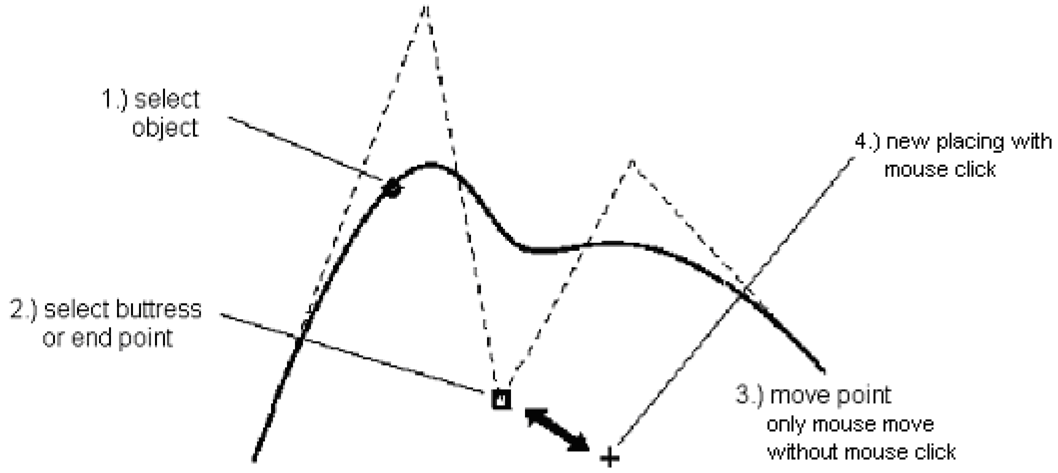
Hints:

Continuous (smooth) bezier curves are generated between the points.

MVP - Move point at nurbs or bezier

Button: 2D + CURVE + MVP

Command: mvp (option: -i = display information box)



Proceeding:

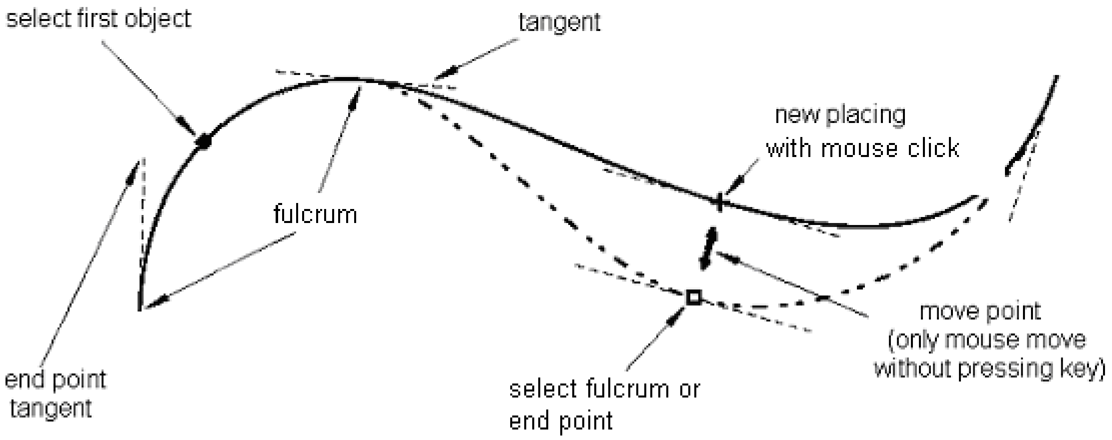
1. Choose command (with button) and select object
2. Select point of the object (with selection)
3. Move (mouse movement), place (mouse click)
4. Choose the other point of the object or "POLYEND"

Hints:
The curve carries along according to the base point moving.

MVP – Move point at 4-point-beziers

Button: 2D + CURVE + MVP

Command: mvp (option: -i = display of an information box)



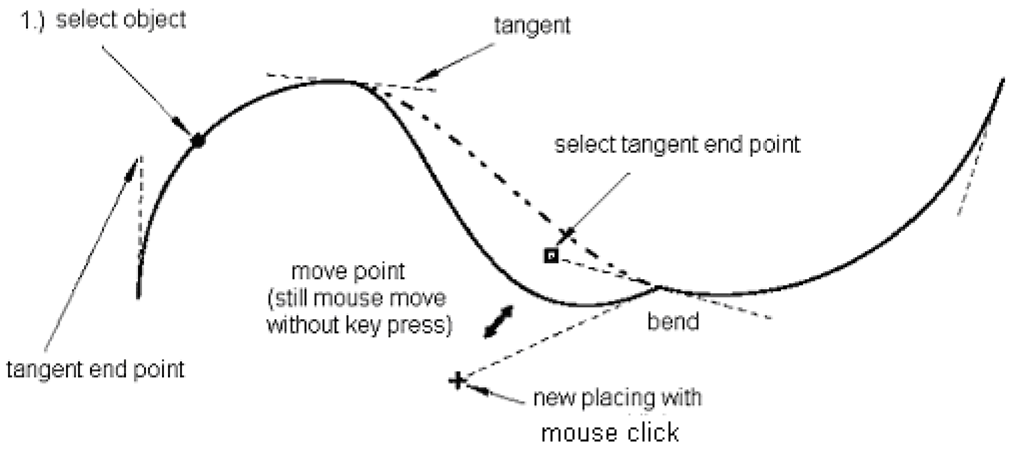
Proceeding:

1. Choose command (with button) and select object
2. Select point of the object (with selection)
3. Move (mouse movement), place (mouse click)
4. Choose the other point of the object or "POLYEND"

Hints:
The direction of tangents remains unchanged, if the base points are moved (at tangent endpoints : bend and kink)

MVP – Change bend at 4-point-bezier

Button: 2D + CURVE + MVP
Command: mvp (option: -i = display of an information box)

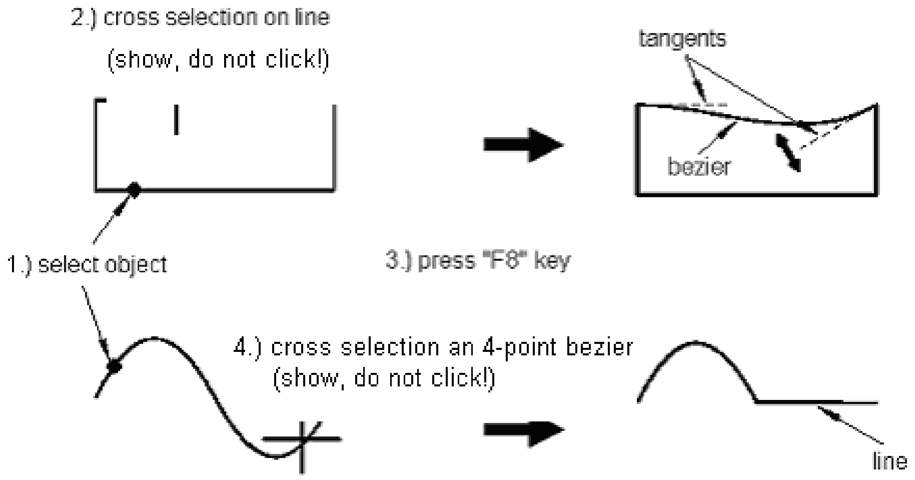


Proceeding:
1. Choose command (with button) and select object
2. Select point of the object (with selection)
3. Move (mouse movement), place (mouse click)
4. Choose the other point of the object or "POLYEND"

Hints:
The movement of tangent endpoints results in discontinuities on the curve (sharp bend)!

MVP - <F8> Conversion: Line <-> bezier (and back)

Button: 2D + CURVE + MVP
Command: mvp (option: -i = display of an information box)

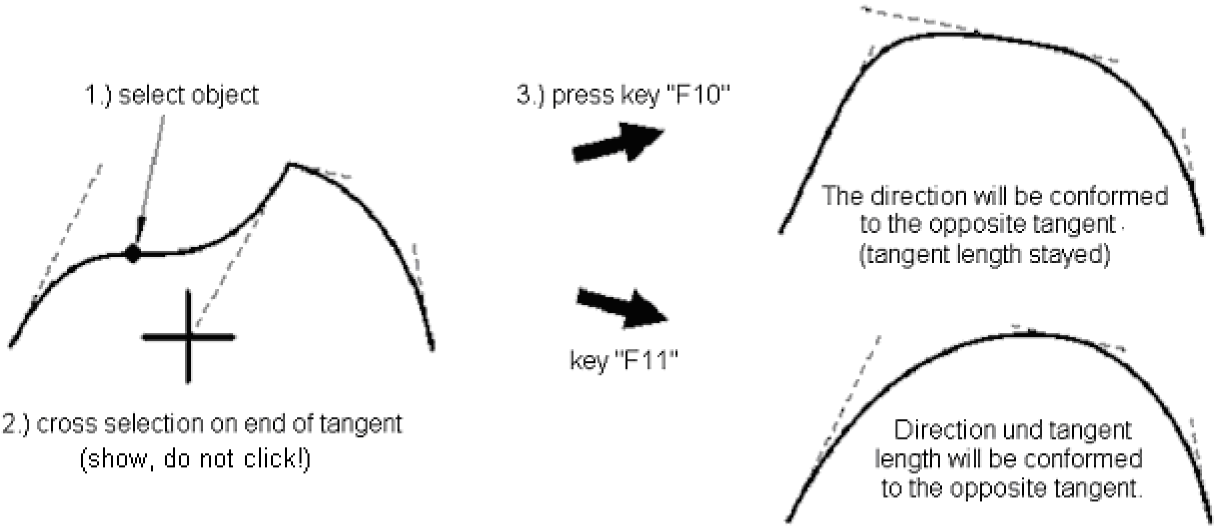


Proceeding:
1. Choose command (button) and select object
2. Show with crosshairs on the element (no click!)
3. Enter “F8“ on keyboard
4. Additional actions or "POLYEND"

Hints:
Converting a line into bezier, the tangents are located on the line. (Allowed: 4-point-bezier and polygon)

MVP - <F10>/<F11> Tangential compensation

Button: 2D + CURVE + MVP
Command: mvp (option: -i = display of an information box)



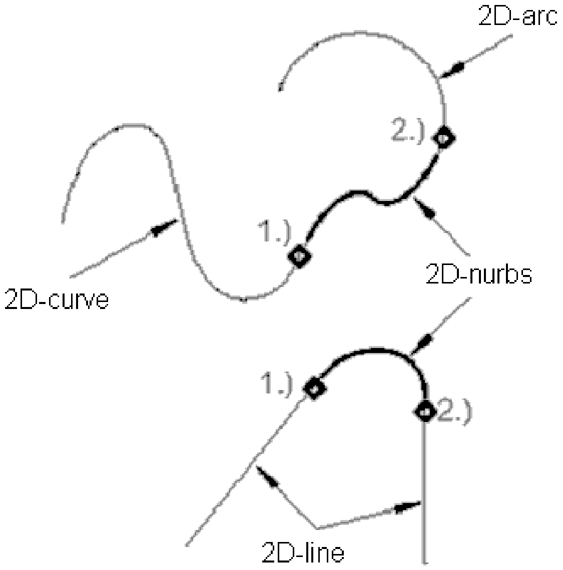
Proceeding:

1. Choose command (button) and select object
2. Show with crosshairs on the element (no click!)
3. Enter “F10“ or "F11" on keyboard
4. Additional actions or "POLYEND"

Hints:
Therewith “sharp bends“ on the 4-point-bezier curve can be compensated (blocked while moving points).

Connect two elements tangential with nurbs curve

Button: 2D + CURVE + NURBS
Command: icrv

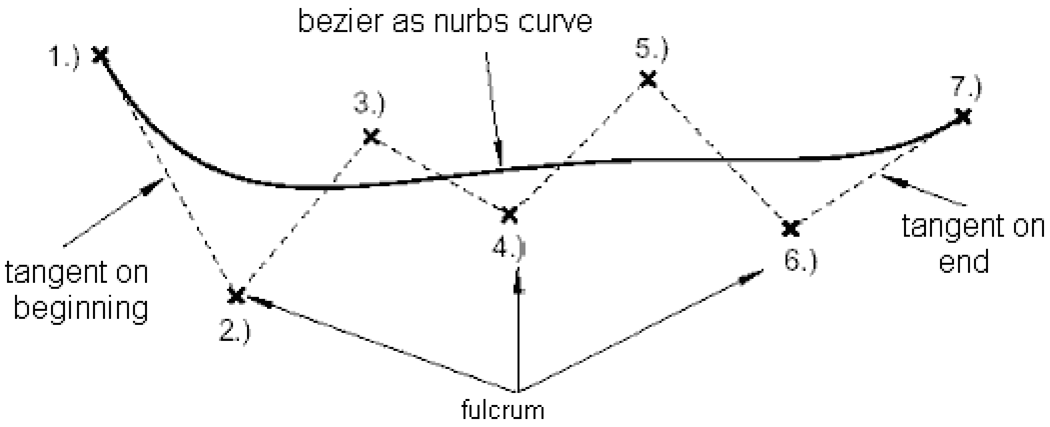


Proceeding:

1. Choose command (button) and select object
2. Select first element at connecting point
3. Select second element at connecting point

Bezier curve

Button: 2D + CURVE + BEZIER
Command: rbc -o?

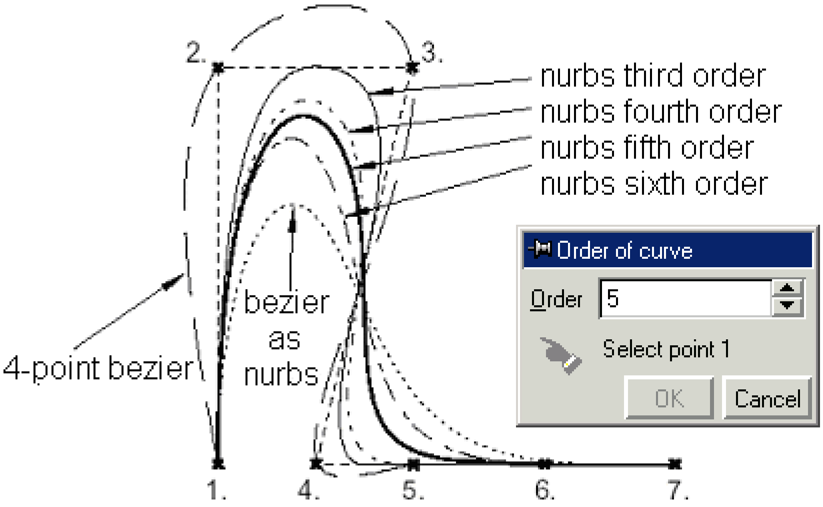


Proceeding:
1. Choose command (button) and select object
2. Select base points
3. Complete with "POLYEND"

Hints:
A smooth curve is drawn between the base points. The first and last point are included.

Nurbs curve with specifying orderliness

Button: 2D + CURVE + NURBS
Command: rbc -o? -c!

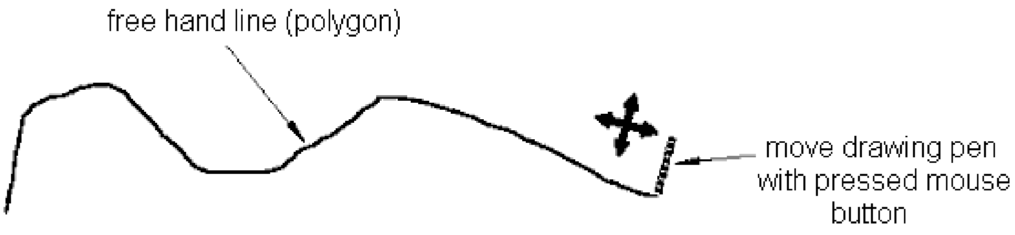


Proceeding:
1. Choose command (button) and select object
2. Enter order into dialog window
3. Select base points
4. Complete with "POLYEND"

Hints:
A nurbs curve is drawn between the base points. The first and last point are included.

Spline curve drawn with freehand

Button: 2D + CURVE +
Command: sketch * -m (-m = multiple)

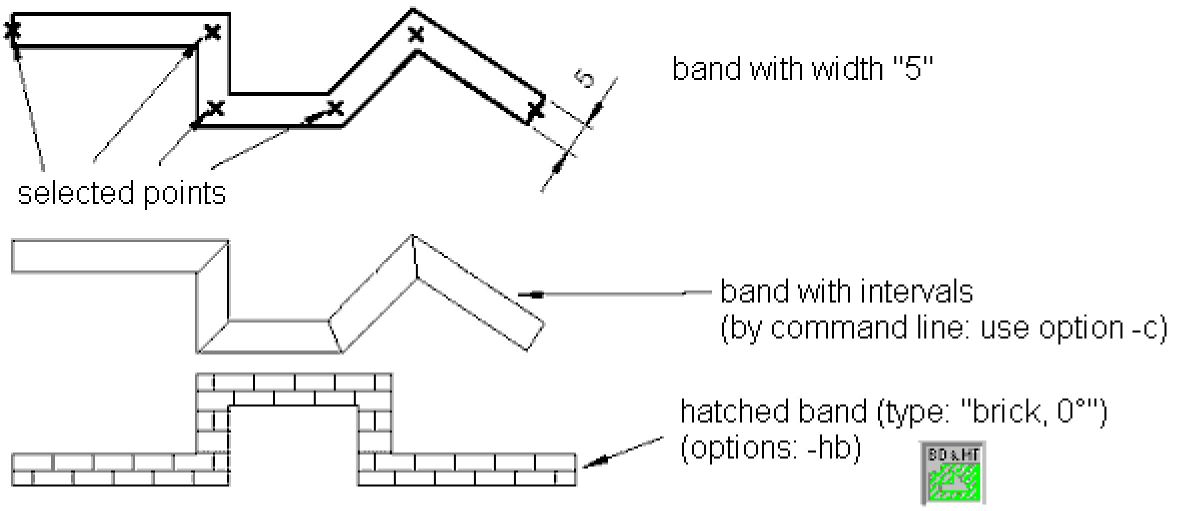


Proceeding:
1. Choose command (button) and select object
2. Draw with pressed mouse key
3. Complete with "POLYEND"

Hints:
A polygon is created. At “unclasp“ the mouse key a gap will be created.

Draw a band with certain width

Button: 2D + CURVE +
Command: trace * 5 (5 = width, options: -hb = hatched, -i = intervals)

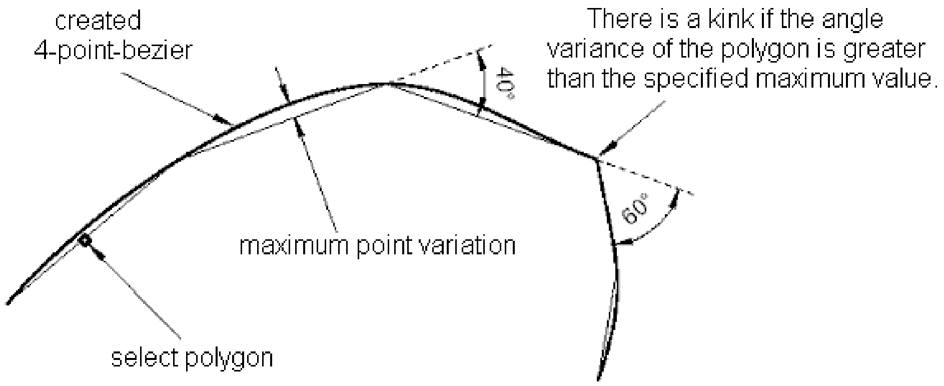


Proceeding:
1. Choose command (button) and select object
2. Enter width (e.g. “5“)
3. Select points (without object snap)
4. Complete with "POLYEND"

Hints:
Snap on adjusted grid (SNAP,GRID).

Bezier smoothing of a polygon with certain limit

Button: 2D + CURVE + POLY BZ
Command: pbez -d 3 -a 45 (-d=max. point tolerance, -a=max. angle tolerance)


created
4-point-bezier
There is a kink if the angle
variance of the polygon is greater
than the specified maximum value.
40°
60°
maximum point variation
select polygon

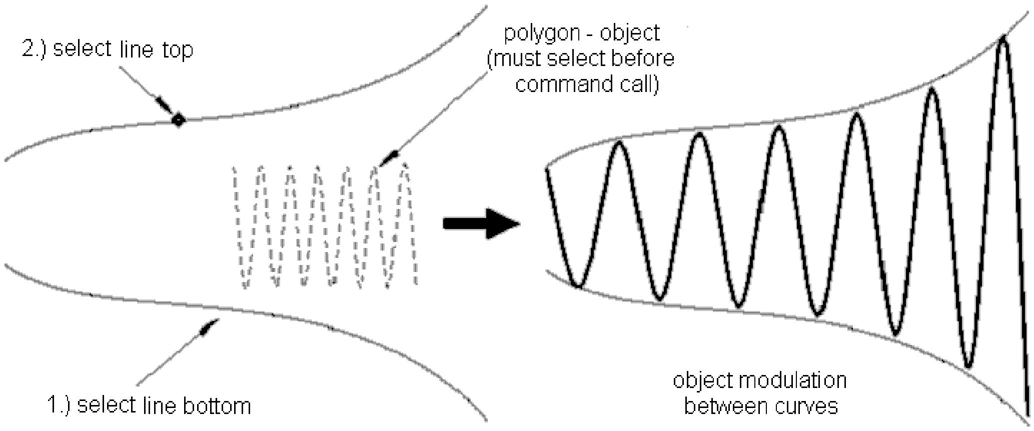
Proceeding:

1. Choose command (button) and select object
2. Enter maximum point tolerance (e.g. “3“)
3. Enter maximum angle tolerance (e.g. “45“)
4. Select polygon you want to smooth

Hints:
The smoothing curve is generated as duplicate.
ATTENTION: Bigger permitted deviations cause a "smoother" curve !

Polygon object modulate between two curves

Button: 2D + CURVE +
Command: mapobj *



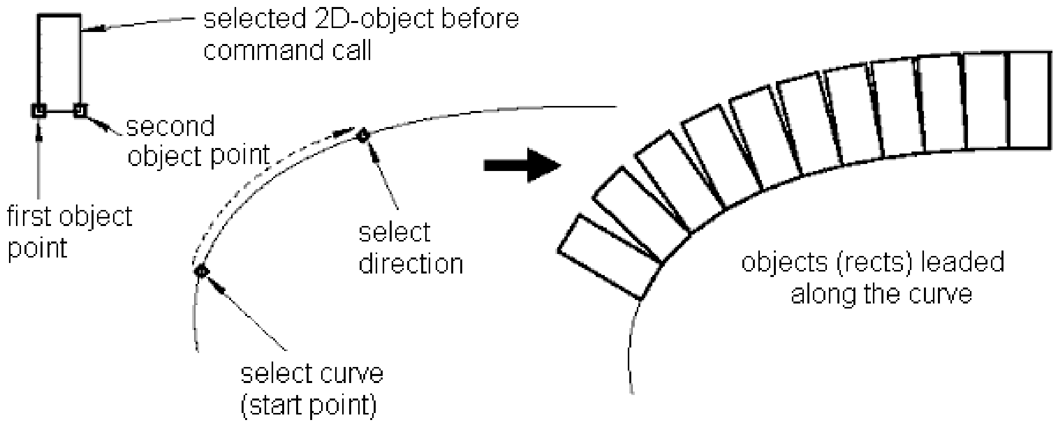
Proceeding:
1. Select 2D-object
2. Choose command (button)
3. Select the lower line (curve)
4. Select the upper line (curve)

Hints:

If necessary the objects must converted in polygons POLY. The right-angled area of the object box is mapped between the two curves.

2D-object drawn along a curve like a chain

Button: 2D + CURVE +
Command: band


selected 2D-object before command call
second object point
first object point
select direction
select curve (start point)
objects (rects) leaded along the curve

Proceeding:

1. Select 2D-object and choose command (button)
2. Select 1. and 2. object point of the 2D-object
3. Select curve (with that you determine the start point)
4. Enter the direction on the curve by choosing a point

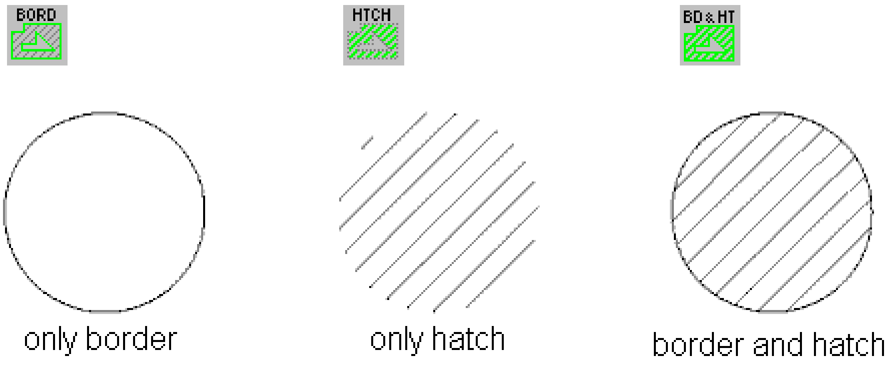
Hints:
The choice of the points on the 2D-object determines the position of the chained objects relating to the curve.

Circles

Border and hatching flags

Before creating closed 2D-objects such as circles, the border and/or hatching flag should be set for the view. You can set these flags about the variable HFL with help of the buttons in the tool bar "Flags". The meaning of these buttons can be explained most simply by an example of the circle generation:

Button: 2D + Circ + ○



Proceeding:

1. To generate the object, set the variable HFL
2. If necessary set hatching type beforehand (HATCH1)
3. Draw the 2D object (with help of buttons)
4. Radius is calculated by system

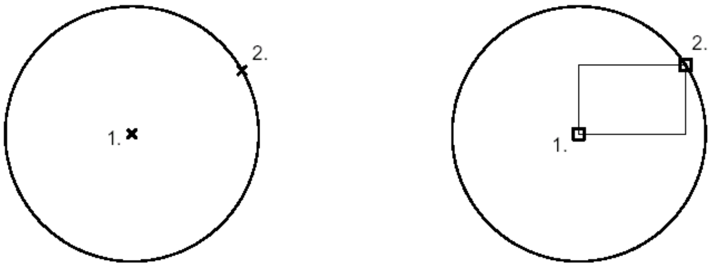
Hints:
Entering the command with keyboard, the option for creating border/hatch must be defined.

Circle (seek centre and border point)

Button: 2D + Circ + ○

Command: rbarc -t -o?

without object snap | with object snap



Proceeding:

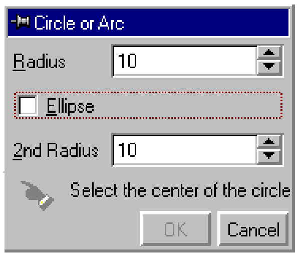
1. Before creating the object set variable HFL
2. Select the centre point
3. Select the border point

Hints:
The selection of the centre and border point can also happen with numerical input (key: n).

Ellipse/circle (numerical radius)

Button: 2D + Circ + Circ

Command: rbarc -t -o? -r!



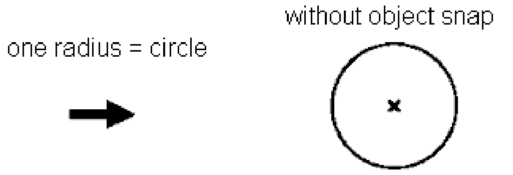
Circle or Arc

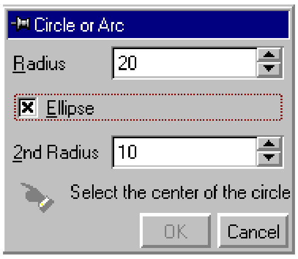
Radius: 10

☐ Ellipse

2nd Radius: 10

Select the center of the circle





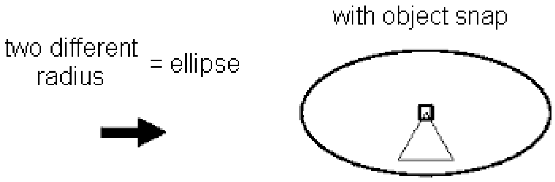
Circle or Arc

Radius: 20

☒ Ellipse

2nd Radius: 10

Select the center of the circle



Proceeding:

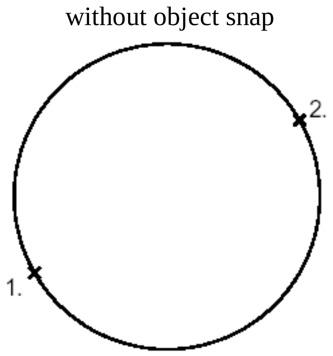
1. Choose function with button or enter the command
2. Determine radius
   (1.radius -> X, 2.radius -> Y)
3. Selection the centre point

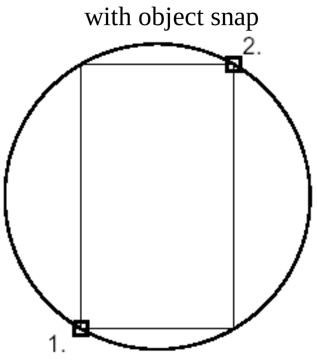
Hints:
The selection of the centre point can also happen with numerical input (key: n).

2-point circle

Button: 2D + Circ + 2P

Command: rbarc -t -o? -2





Proceeding:

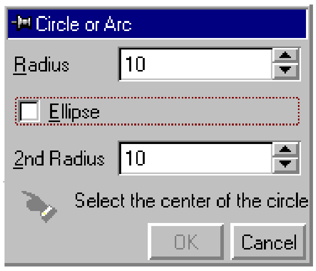
1. Choose function with button or enter command
2. Select 1. point (start point)
3. Select 2. point (end point)

Hints:
The selection of the points can also happen with numerical input (key: n).

Ellipse/circle of two points with numerical radius

Button: 2D + Circ + R=?

Command: rbarc -t -o? -2 -r!

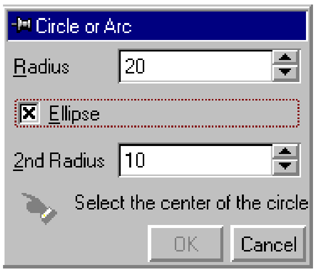

Circle or Arc

Radius: 10

☐ Ellipse

2nd Radius: 10

Select the center of the circle

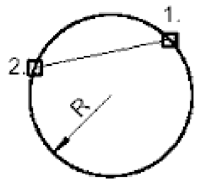

Circle or Arc

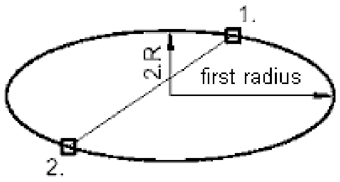
Radius: 20

☒ Ellipse

2nd Radius: 10

Select the center of the circle





Proceeding:
1. Choose function with button or enter command
2. Enter radius
3. Select 1. point (start point)
4. Select 2. point (end point)

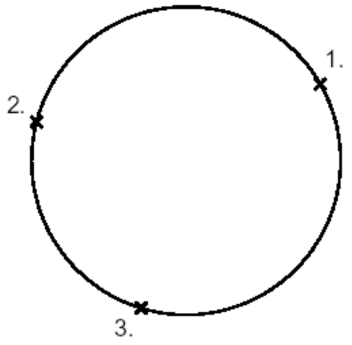
Hints:
The selection of the points can also happen with numerical input (key: n).

3-point circle

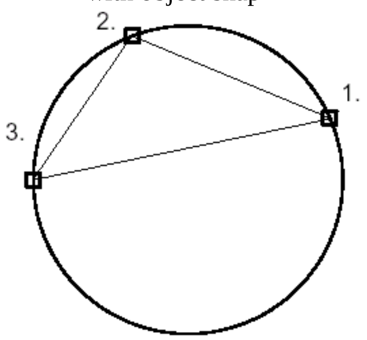
Button: 2D + Circ + 3P

Command: rbarc -t -o? -3

without object snap



with object snap



Proceeding:
1. Choose function with button or enter command
2. Select 1. point (start point)
3. Select 2. point (end point)
4. Select 3. point (intermediate point)

Hints:
The selection of the points can also happen with numerical input (key: n).

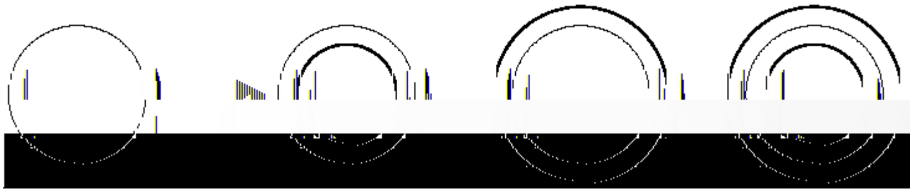
Concentric circle with numerical distance

Button: 2D + Circ + EQUI

Command: elequi

Example: elequi -i -d5 (-d distance)

elequi -i (inside) elequi -o (outside) elequi -b (both)



Proceeding:

1. Choose function with button or enter command
2. If necessary, enter distance and position
3. Select the circle

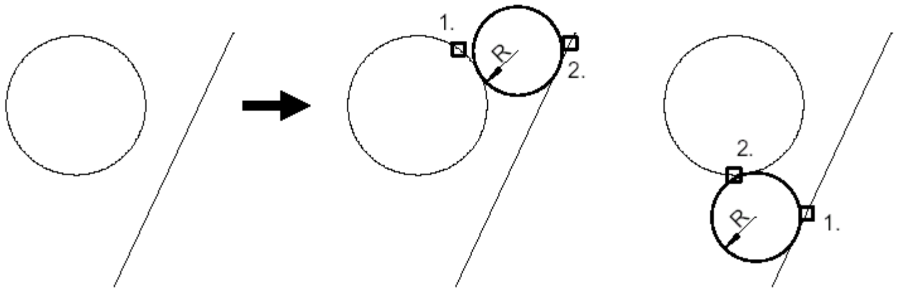
Hints:
Multiple execution with option -m.

Tangential circle with numerical radius

Button: 2D + Circ + 2E R=?

Command: ct2e "" "" ? -t (-t = total ; instead of ’?’ the radius can be entered)

The position of the tangential circle is dependent on the selection points.



Proceeding:

1. Choose function with button or enter command
2. If necessary, enter the radius
3. Select the elements

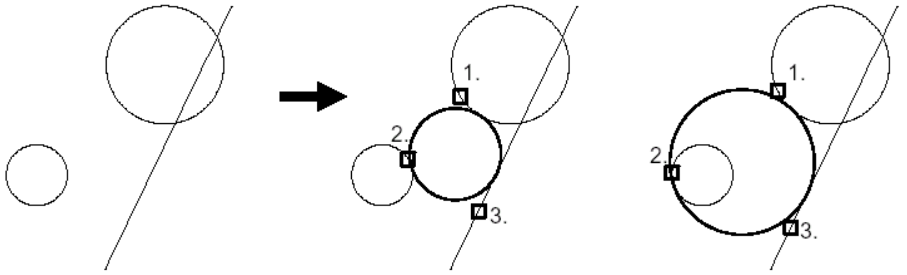
Hints:
Multiple execution with option -m.

Tangential circle at three elements

Button: 2D + Circ + 3E

Command: ct3e -t (-t = total)

The position of the tangential circle is dependent on the selection points.



Proceeding:

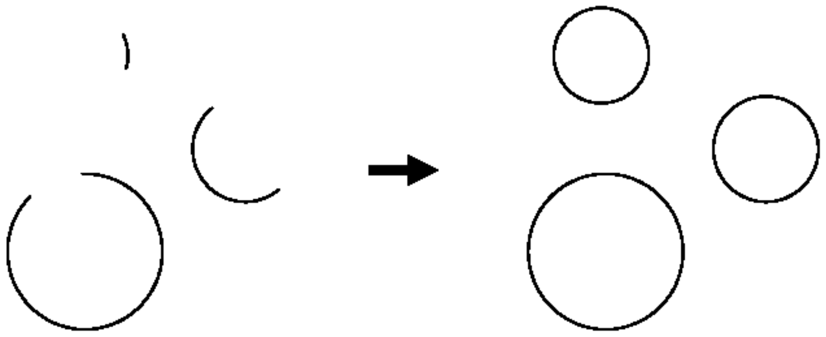
1. Choose function with button or enter command
2. Select the elements
3. Radius is calculated by the system

Hints:
Multiple execution with option -m.

Regeneration of a circle out of arcs

Button: 2D + Circ + ADD -t

Command: addarc -t -m (-t=total, -m=multiple)



Proceeding:

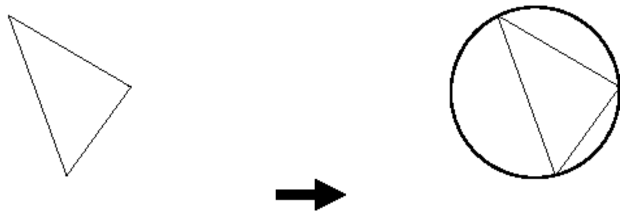
1. Choose function with button or enter command
2. Select the arc

Hints:
Multiple execution with option -m.

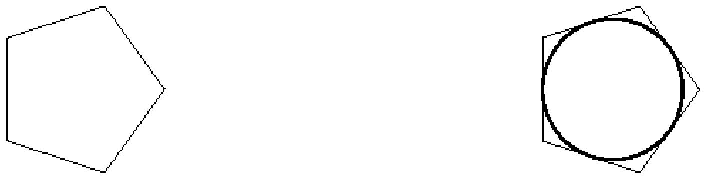
Generation of outer- and inner circle

Button: 2D + Circ + MORE CIRC + ?

Command: iocirc



Command: iocirc -i (-i = inner)



Proceeding:

1. Choose function with button or enter command
2. Select the 2D-object

Hints:

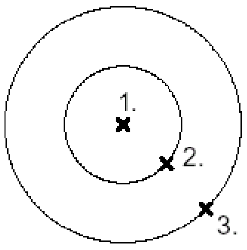
Triangles, parallelograms and regular polygons can be chosen as 2D objects.

Generation of circle rings

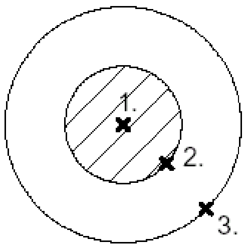
Button: 2D + Circ + MORE CIRC + ?

Command: ring

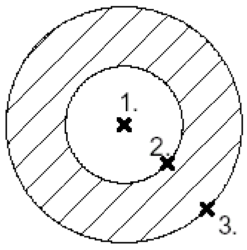
ring



ring -I
(-i=inside hatched)



ring -o
(-o=outside hatched)



Proceeding:

1. Choose function with button or enter command
2. Select the centre point and the radius
3. Complete with "POLYEND"

Hints:

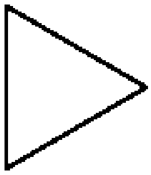
The selection of points can also happen with help of object snap or with numerical input. The definition of hatching is determined in the variable HATCH1.

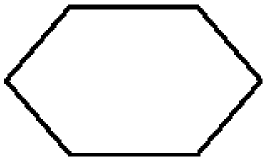
Special form of circles

I. Polygon circle (Circle is approached by a line)

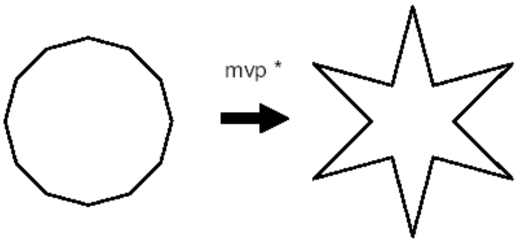
Command: arc * -t -p[#] (-t=total; -p=polygon circle with # edges (standard value: 90))

Triangle: arc * -tp3 **compressed hexagon:** arc * -tp6 -r rad1,rad2





Dodecagon with specification the numerical radius and the centre: arc * -tp12 -r8 -c10,20



Proceeding:

1. Choose function with command
2. If necessary, select the centre point and/or select the start point

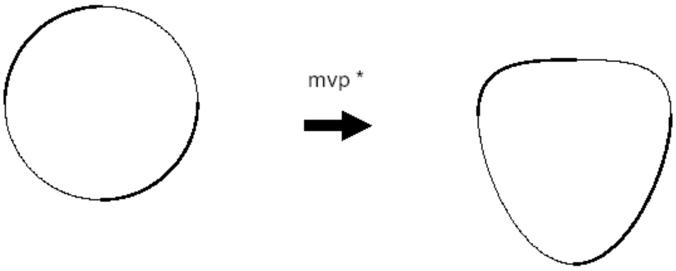
Hints:

The circular arc generation is carried out without the option t. The numerical input of the angle range is carried out via the option s a1 .a2.

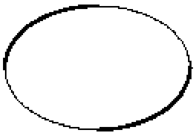
II. Bezier circle

Command: arc * -t -z (-t=total; -z=circle(arc) is building out of maximum 4 bezier curves)

Example: arc * -t -z -c0,0 -r25 (-c=num. declaration of the circle centre point, -r=radius)



Ellipse: arc * -tz -r rad1,rad2



Proceeding:

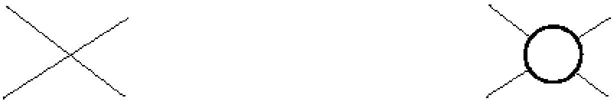
1. Choose function with command
2. If necessary, select the centre point and/or select the start point

Hints:
The circular arc generation is carried out without the option t. The numerical input of the angle range is carried out via the option s a1 .a2.

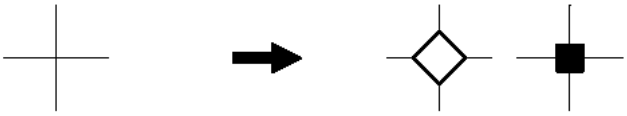
III. Insert knot point

Button: 2D + Circ + ?

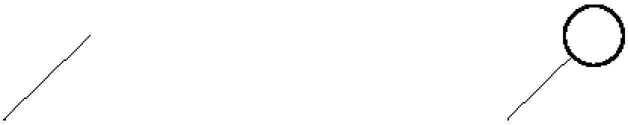
Command: knot [rad] (e.g. knot 5)



Command: knot -t2/knot 3 -t3 -f (f=fill (full hatching))



Command: knot 4 -s (s=single)



Proceeding:

1. Choose the function with button or enter the command
2. If necessary, please put in the radius
3. Select intersection point of two lines
   -> (at option -s the end point must be entered)

Hints:
The option -t2/-t3 is only useful at axially parallel lines. Multiple executions are possible with the option -m.

Arcs

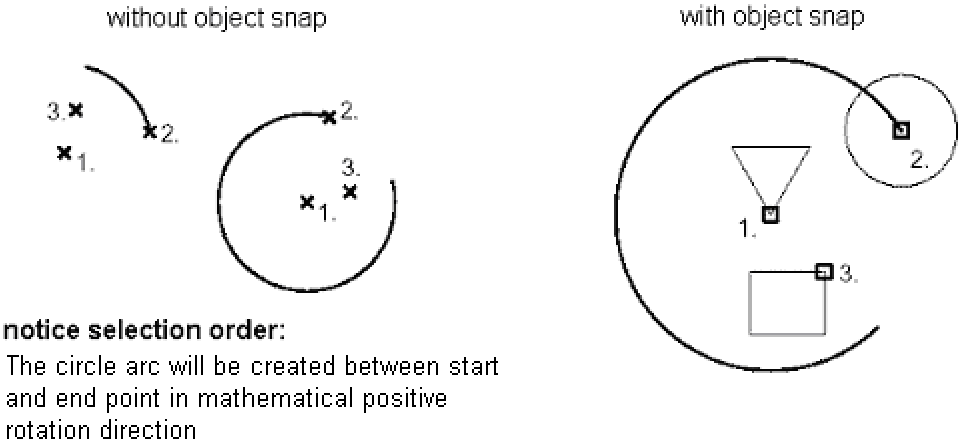
The construction of several 2D-arcs carried out with the submenu in the tool box „Tools“:

Menu-Button: 2D + ARC

Circular arc (Selection of centre-, start- and end point)

Button: 2D + ARC + ARC

Command: rbarc -o?


without object snap
3.
2.
1.
2.
3.
1.
notice selection order:
The circle arc will be created between start and end point in mathematical positive rotation direction
with object snap
2.
1.
3.

Proceeding:

1. Choose the function with button or enter the command
2.. Select the centre point
3. Select the arc start point
4. Fix the arc angle (select end point)

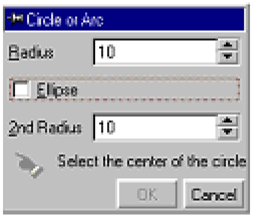
Hints:

Please pay attention to the order of selection:
The arc will be generated in a mathematical positive rotating direction (counter-clockwise) between start and end point.
The selection of point is also possible with the numerical input (key: n).

Elliptic arc/Arc (radius, centre)

Button: 2D + ARC + ARC R=?

Command: rbarc -o? -r!



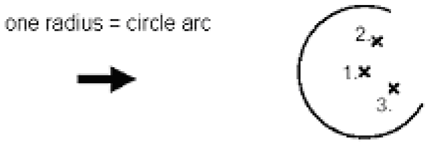
Circle or Arc

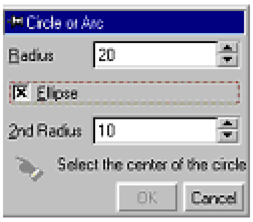
Radius 10

□ Ellipse

2nd Radius 10

Select the center of the circle





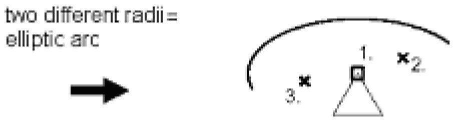
Circle or Arc

Radius 20

☒ Ellipse

2nd Radius 10

Select the center of the circle



The arc is generated in a mathematical positive rotation direction (counter clockwise).

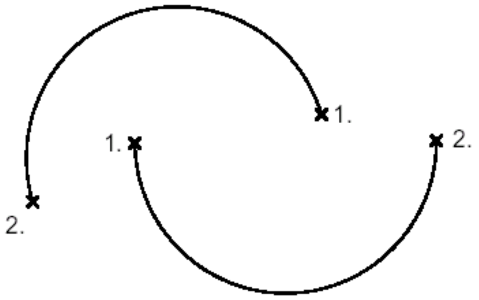
Proceeding:
1. Choose functions with button or enter command
2. Enter radius, select the centre point
3. Fix the start angle (select the start point)
4. Fix the end angle (select the end point)

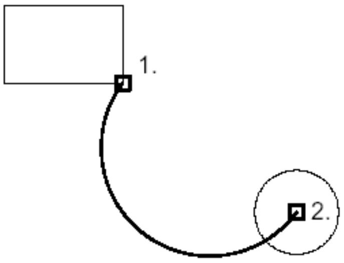
Hints:
The selection of points can also happen with numerical input (key: n).

Semi circle with two points

Button: + +

Command: rbarc -o? -2





The arc is generated in a mathematical positive rotating direction (counter clockwise).

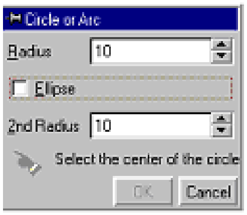
Proceeding:
1. Choose function with button or enter command
2. Select the start-point of arc
3. Select the end-point of arc
4. The radius is calculated by the system

Hints:
The selection of points can also happen with numerical input (key: n).

Elliptical arc/circular arc (radius, centre)

Button: 2D + ARC + 2P R=?

Command: rbarc -o? -2 -r!



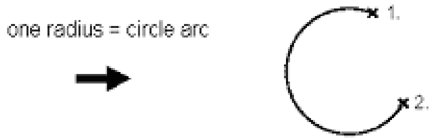
Circle or Arc

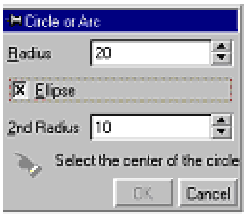
Radius: 10

☐ Ellipse

2nd Radius: 10

Select the center of the circle





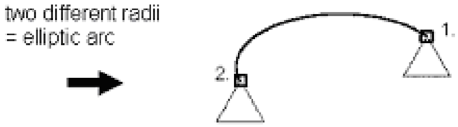
Circle or Arc

Radius: 20

☒ Ellipse

2nd Radius: 10

Select the center of the circle



The arc is generated in a mathematical positive rotating direction (counter-clockwise).

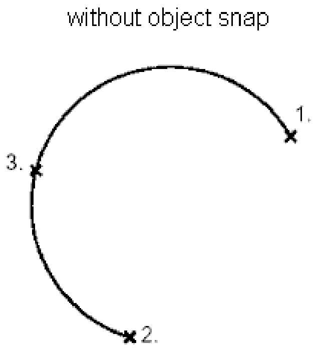
Proceeding:
1. Choose function with button or enter command
2. Enter the one or more radii
3. Select the start- and end-point of arc
4. The centre point is calculated by the system

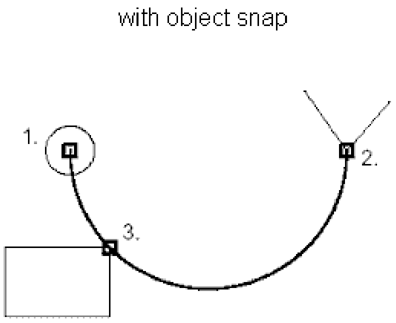
Hints:
The selection of points can also happen with numerical input (key: n).

Circular arc with three points

Button: 2D + ARC + 3P

Command: rbarc -o? -3





Proceeding:
1. Choose function with button or enter command
2. Select the 1. point (start point)
3. Select the 2. point (end point)
4. Select the 3. point (centre point)

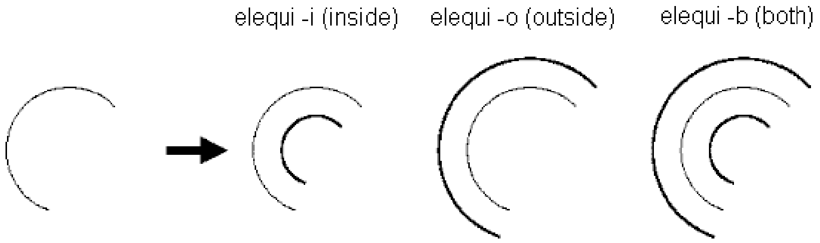
Hints:
The selection of points can also happen with numerical input (key: n).

Equidistant arc

Button: 2D + ARC + EQUI

Command: elequi

Example: elequi -i -d5 (-d distance)



Proceeding:
1. Choose function with button or enter command
2. If necessary, enter distance and position
3. Select the circular arc

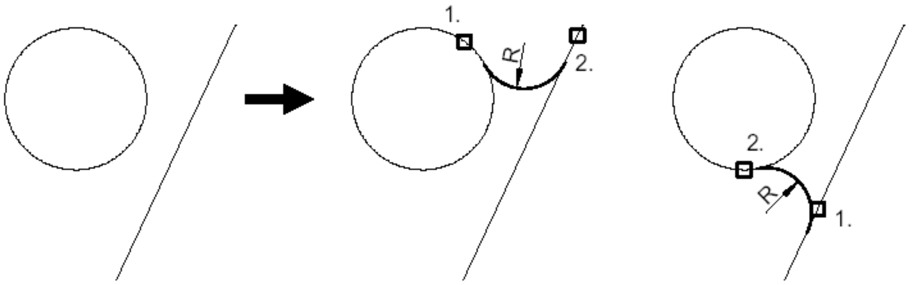
Hints:
Multiple execution with option -m.

Tangential circular arc, numerical radius

Button: 2D + ARC + 2E R=?

Command: ct2e "" "" ? (instead of ’?’ the radius can be entered)

The position of the tangential arc is dependent on the selection points.



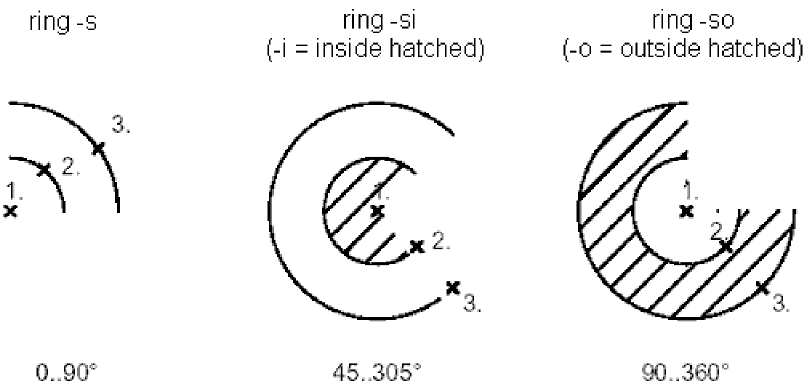
Proceeding:
1. Choose function with button or enter command
2. If necessary, enter the radius
3. Select the elements

Hints:
Multiple execution with option -m.

Generation arc-ring segments

Button: 2D + ARC + MORE ARC +

Command: ring -s (-s = segment, start- and end angle are inquired)



Proceeding:

1. Choose function with button or enter command
2. Select the centre point and the 1. radius
3. Enter the start-/end-angle, Select the 2. radius

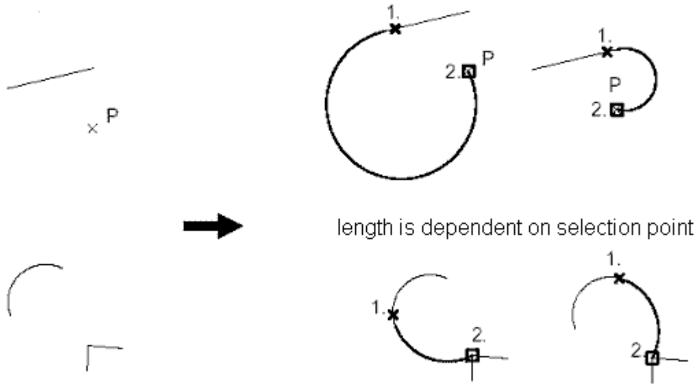
Hints:

- Complete with "POLYEND"
- Selection of the points are also possible with object snap or numerical input
- Hatching definition out of variable HATCH1

Tangential circular arc at element through a point (contact arc)

Button: 2D + ARC +

Command: ctep



Proceeding:

1. Choose function
2. Select the element
3. Select the point
4. Radius is calculated by system

Hints:

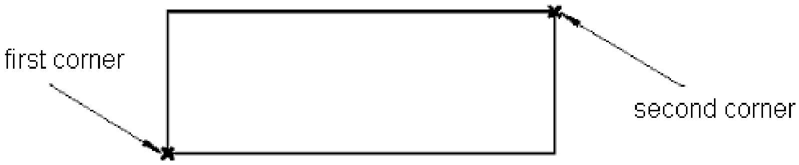
The selection of points can also happen with numerical input (key: n).

2D-Objects

Rectangles

Axially parallel rectangle

Button: 2D + 2D + RECT
Command: recta * (without object snap, Option: -bh = border and hatch)



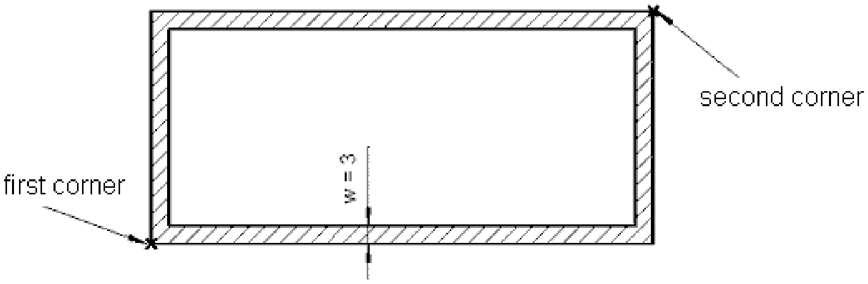
With BD&HT the geometry will be hatched (if calling button).

Proceeding:
1. Choose command (with button)
2.. Select first corner
3. Select second corner

Hints:
Calling the function with button, you can snap on objects. With the key <n> a numerical input of coordinates is possible.

Rectangle with (hatched) border

Button: 2D + 2D +
Command: recta * -bh -w 3 (-bh= border and hatch, -w 3= border with width 3)



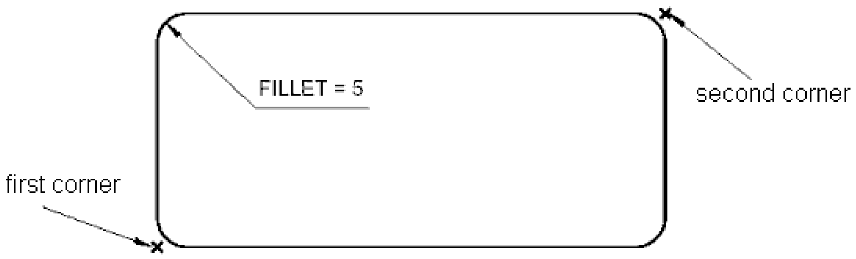
With BD&HT the border will be hatched (if calling button).

Proceeding:
1. Adjust and set hatching
2. Enter width of border
3. Select diagonal points (1. and 2. corner)

Hints:
Calling the function with button, you can snap on objects. With the key <n> a numerical input of coordinates is possible.

Rectangle with fillets

Button: 2D + 2D + RECT
Command: recta * -f (-f= fillet)



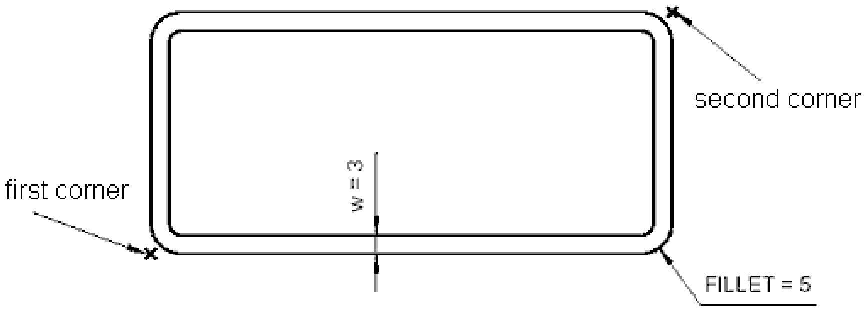
With BD&HT the border will be hatched (if calling the button).

Proceeding:
1. Choose command (with button)
2. Select first corner
3. Select second corner
4. Enter rounding radius (if FILLET empty)

Hints:
The rounding radius is the value in the variable FILLET.

Rectangle with fillets and border

Button: 2D + 2D + ▢
Command: recta * -fw 3 (-f= fillet, -w 3= border with width 3)



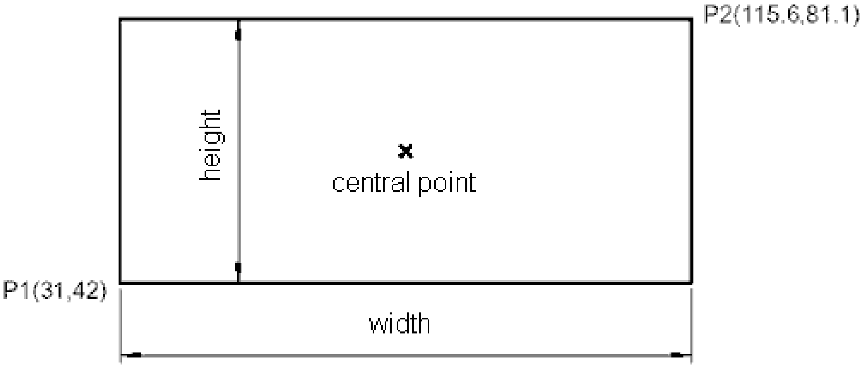
With BD&HT the border will be hatched (if calling the button).

Proceeding:
1. Choose command (with button)
2. Select first and second corner
3. Enter rounding radius (if FILLET empty)

Hints:
The rounding radius is the value in the variable FILLET.

Rectangle with numerical input of dimensions

Button: 2D + 2D + RECT NUM
Command: recta * 31,42..115.6,81.1 (parameter = coordinates of the left lower and the right upper corner)
e.g. recta * ($box)

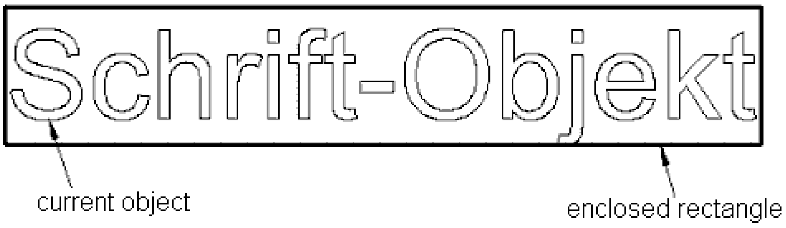


Proceeding:
1. Choose command (with button)
2. Enter width
3. Enter height
4. Enter coordinates of the centre point

Hints:
Using the command directly, the coordinates for the left lower edge and right upper corner are entered as parameter disconnected with "..".

Rectangle around a current object

Button: 2D + 2D +
Command: box *;recta * ($box) (The parameter of the variable BOX are used)

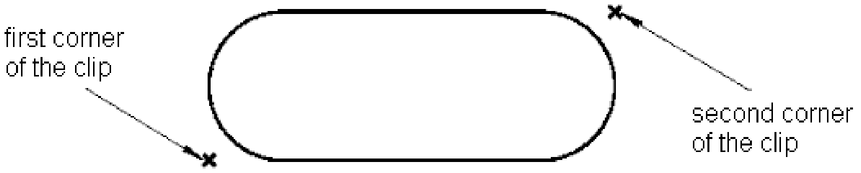


Proceeding:
1. Select object
2. Choose command (with button)

Hints:
If necessary, the object box surrounds also (invisible) tangent-end-points of the current object.

Rectangle oval

Button: 2D + 2D + OVAL
Command: recta * -o (-o= oval, arcs on the short sides)



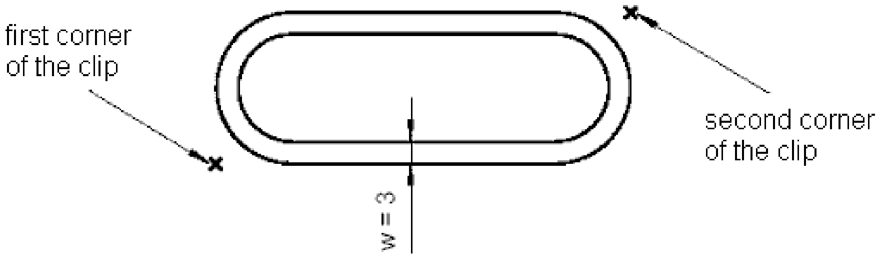
With BD&HT the border will be hatched (if calling the button).

Proceeding:
1. Choose command (with button)
2. Select the first corner of the part
3. Select the second corner of the part

Hints:
Calling the command directly, the option "-h" is not allowed.

Rectangle - oval with border

Button: 2D + 2D +
Command: recta * -o -w 3 (-o= oval, -w 3= border with width 3)



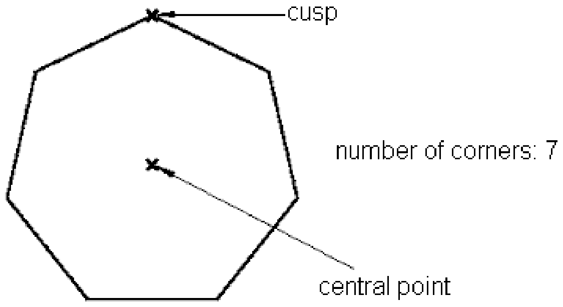
With BD&HT the border will be hatched (if calling the button).

Proceeding:
1. Choose command (with button)
2. Enter the border width
3. Select the first corner of the part
4. Select the second corner of the part

Hints:
Calling the command directly, the option "-h" is not allowed.

Regular polygon

Button: 2D + 2D + ⬡
Command: reg (-h= hatched, -hb= hatch with border)



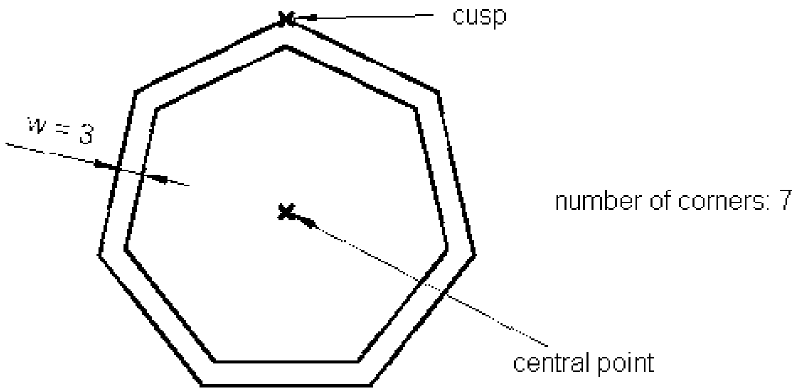
With BD&HT the border will be hatched (if calling the button).

Proceeding:
1. Choose command (with button)
2. Enter number of corners
3. Select centre point
4. Select zenith

Hints:
Calling the command directly, the number of corners can be entered as explicit parameters.

Regular polygon with border

Button: 2D + 2D +
Command: reg -w 3 (-w 3= border with width 3, -h= hatched, -hb= hatch.+ border)



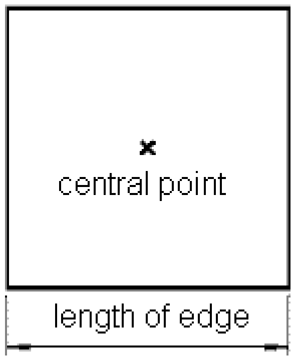
With BD&HT the border will be hatched (if calling the button).

Proceeding:
1. Choose command (with button)
2. Enter border width and number of corners
3. Select centre point
4. Select zenith

Hints:
Calling the command directly, the border width and the number of corners can be entered as explicit parameters.

Square at centre point with edge length

Button: 2D + 2D +
Command: {command order, if using of buttons}



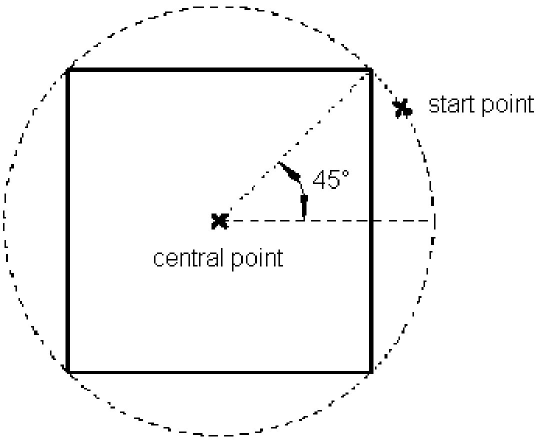
Proceeding:
1. Choose command (with button)
2. Select centre point
3. Enter length of edge

Hints:
Beginning with the corner, the drawing of a square is possible with “recta“.

With BD&HT the border will be hatched (if calling the button).

Square (axially parallel) at centre point with circle command

Command: arc * -s45..45 -p4 (-s= start- and end-angle, -p= number of point, -h= hatch)



With BD&HT the border will be hatched (if calling the button).

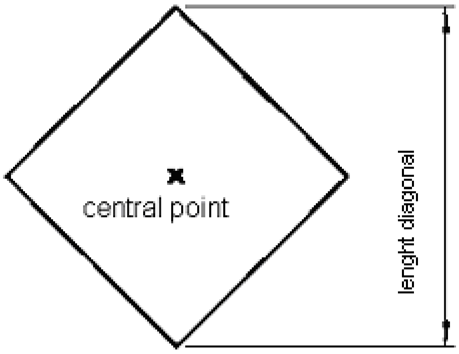
Proceeding:
1. Choose command (with keyboard)
2. Select centre point
3. Select start point

Hints:
Beginning with the corner, the drawing of a square is possible with “recta“.

Rhombus with diagonal length

Button: 2D + 2D +

Command: arc * -r 20 -s0..0 -p4 (-r= radius, -s= start- and. end-angle, -p4= 4 points)



Proceeding:

1. Choose command (with button)
2. Select centre point
3. Enter diagonal length

Hints:

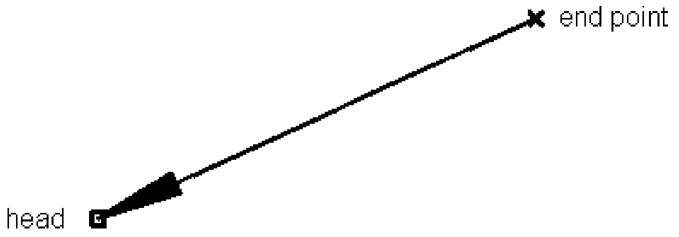
The radius at circle command corresponds to the half diagonal length!

With BD&HT the border will be hatched (if calling the button).

Arrows

Draw an arrow (multiple)

Button: 2D + 2D + ↖
Command: arrow -fm (-f= filled arrowheads, -m= multiple)



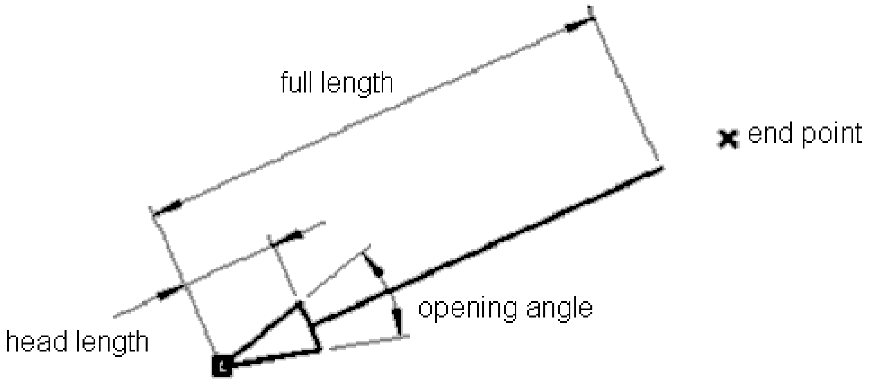
Proceeding:
1. Choose command
2. Select arrow head
3. Select end point
4. More with 2. or complete with "POLYEND"

Hints:
The arrow length is the value in the variable ARR.

With BD & HT the arrowheads are filled (if calling the button).

Draw a arrow, numerical (multiple)

Button: 2D + 2D + NUM
Command: arrow -m -p10 -t50 -w30 (-f= filled, -m= multiple, -p= arrow length, -t= total length, -w= aperture angle)



Proceeding:
1. Choose command (with button)
2. Enter head length, full length + angle
3. Select head and end point
4. More with 3. or complete with "POLYEND"

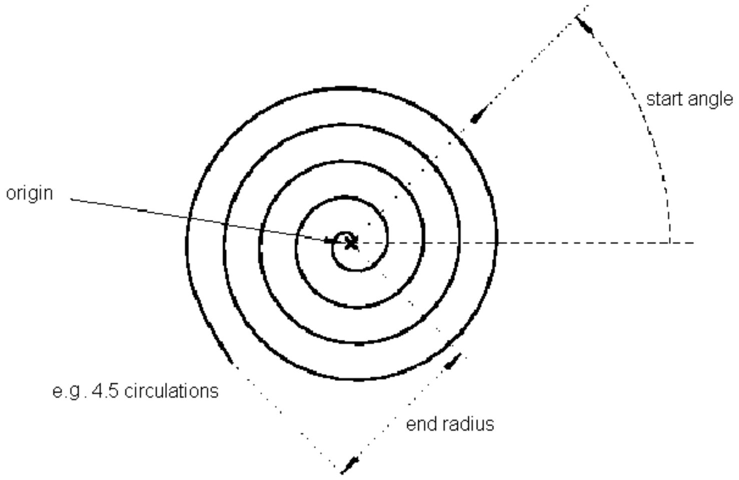
Hints:
The end point determines the direction of the arrow.

With BD & HT the arrowheads are filled (if calling the button).

Spirals

Arithmetical spirals

Button: + +
Command: spiral -a (-a= arithmetical)



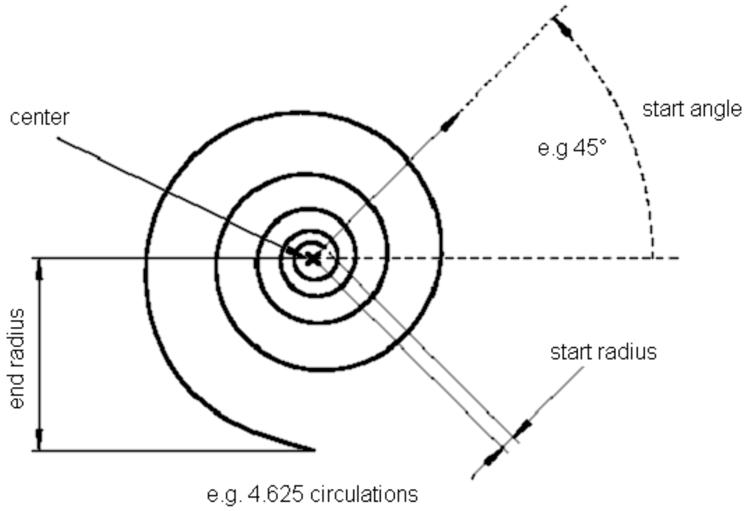
Proceeding:

1. Choose command (after button, please enter “a“)
2. Select origin
3. Enter start angle (between 0..360)
4. Enter end radius and number of circulations

Hints:
The spiral polygon can be smoothed (e.g. “smp * -z“).

Logarithmical spiral

Button: + +
Command: spiral -l (-l= logarithmical)



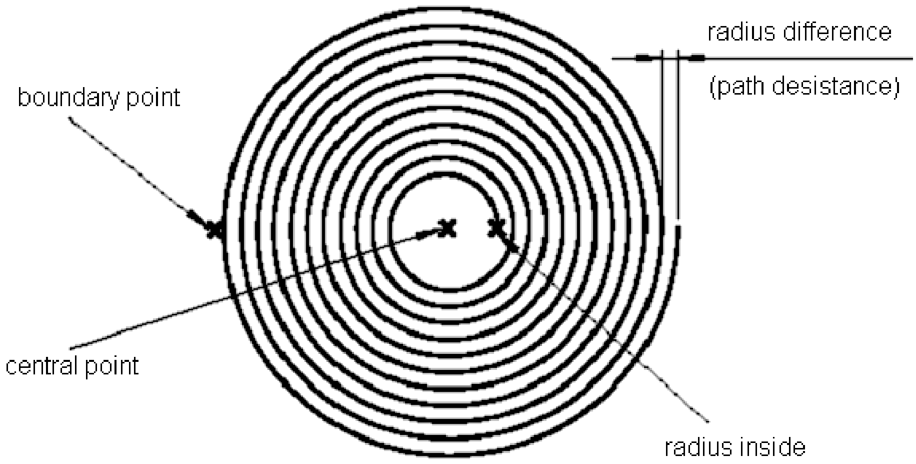
Proceeding:

1. Choose command (after button, please enter “a“)
2. Select origin
3. Enter start angle (between 0..360) and starting radius
4. Enter end radius and number of circulation

Hints:
The spiral polygon can be smoothed (e.g. “smp * -z“).

Spiral with radius difference (path desistance)

Button: + +
Command: spiral -a (-a= arithmetical)



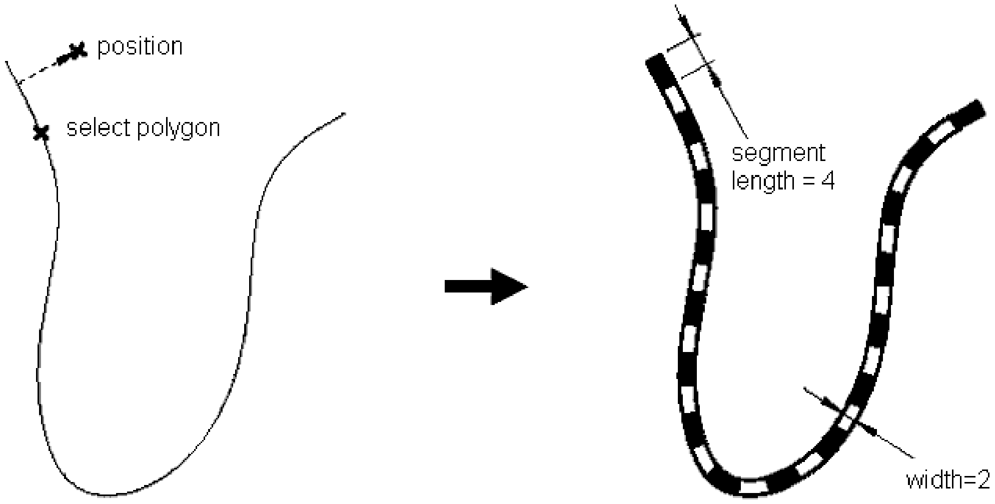
Proceeding:

1. Choose command (with button) + select centre
2. If necessary select radius inside, otherwise POLYEND
3. Select boundary point (outside radius)
4. Enter path desistance (radius difference)

Hints:
The spiral starts in the centre, if POLYEND at radius inside. The boundary point will never be reached completely (Reset of path desistance at whole circulation).

Hatch with pattern (cartographic railway line)

Button: + +
Command: rway



Proceeding:

1. Choose command (with button)
2. Enter width and segment length
3. Select polygon
4. Select position of equidistant relative to polygon

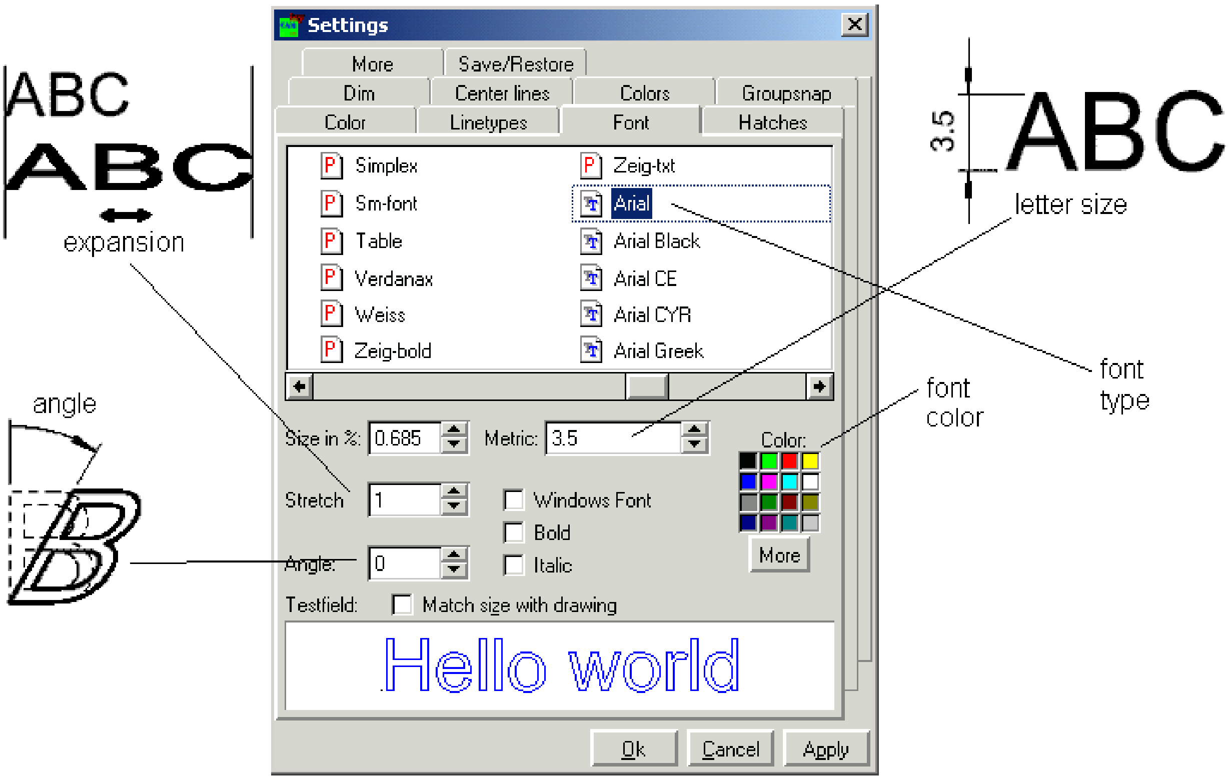
Hints:
The curve must be a polygon. If calling the function with button a full hatch will be drawn. (With command you can adjust beforehand)

2D-Labeling

Parameter

Set labeling parameter

You can set the labeling parameter with the button in the Tool “Extras“. A dialog window will be opened with the headline “Settings“. With click on the register card “Font“ you will see the current labeling parameter.



Size in %: 0.685 Metric: 3.5

Stretch: 1

Angle: 0

□ Windows Font
□ Bold
□ Italic

Testfield: □ Match size with drawing

Hello world

Proceeding:

1. Open dialog box, register card “Font“
2. Determine font
3. Determine parameter
4. Apply settings with “Ok“

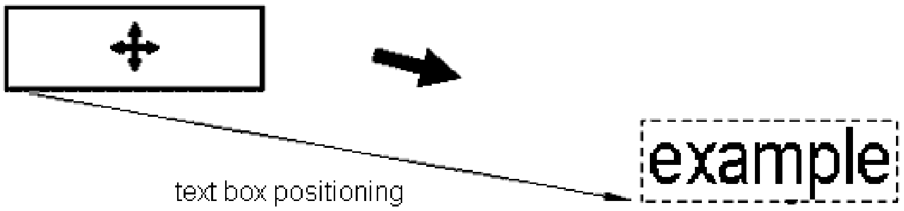
Hints:

The adjusted parameters are assigned to the variable TXTSPEC.

Enter text and position label with window

Button: 2D + LABEL + TEXT

Command: label *



Proceeding:

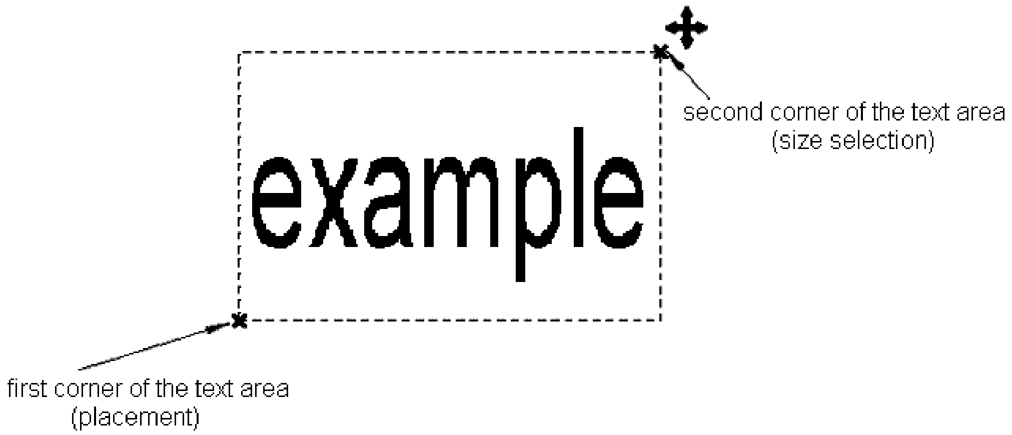
1. Choose command (with button)
2. Enter text (complete with <**ENTER>**“)
3. Select position of text (place the rectangle with mouse)

Hints:
Font, size and colour are applied from the variable TXTSPEC.

Enter text with scale label into window (proportional)

Button: 2D + LABEL + TEXT

Command: label * -s (-s = size by window-scaling)



Proceeding:

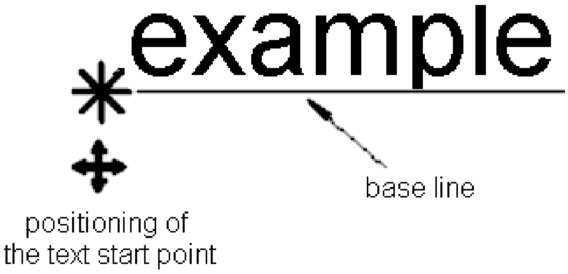
1. Choose command (with button)
2. Enter text (Complete with <**ENTER**>)
3. Select the first corner of text range
4. Select the second corner of text range

Hints:
The text is fitted into the spanned rectangle proportionally (Font and colour from the variable TXTSPEC).

Enter text left side of label text base line (left point)

Button: 2D + LABEL + TEXT

Command: label * -bp ? (-b = base line, -p = left point [?=interactive])



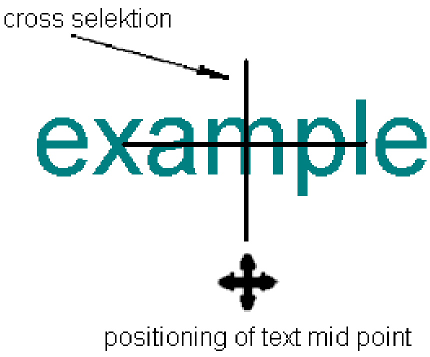
Proceeding:
1. Choose command (with button)
2. Enter text (+ <**ENTER**>)
3. Select text start point

Hints:
Font, size and colour from the variable TXTSPEC (no object snap).

Enter text, position centre of label text

Button: 2D + LABEL + TEXT

Command: label * -c ? (-c = centre [?= interactive])



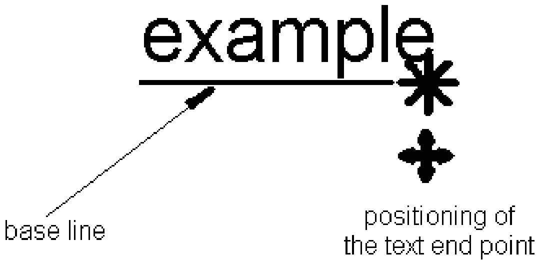
Proceeding:
1. Choose command (with button)
2. Enter text (+ <**ENTER**>)
3. Select centre of label text

Hints:
Font, size and colour from the variable TXTSPEC (no object snap).

Enter text right side of label text base line (right point)

Button: 2D + LABEL + TEXT

Command: label * -bp ? (-b = base line, -p = right point [?=interactive])



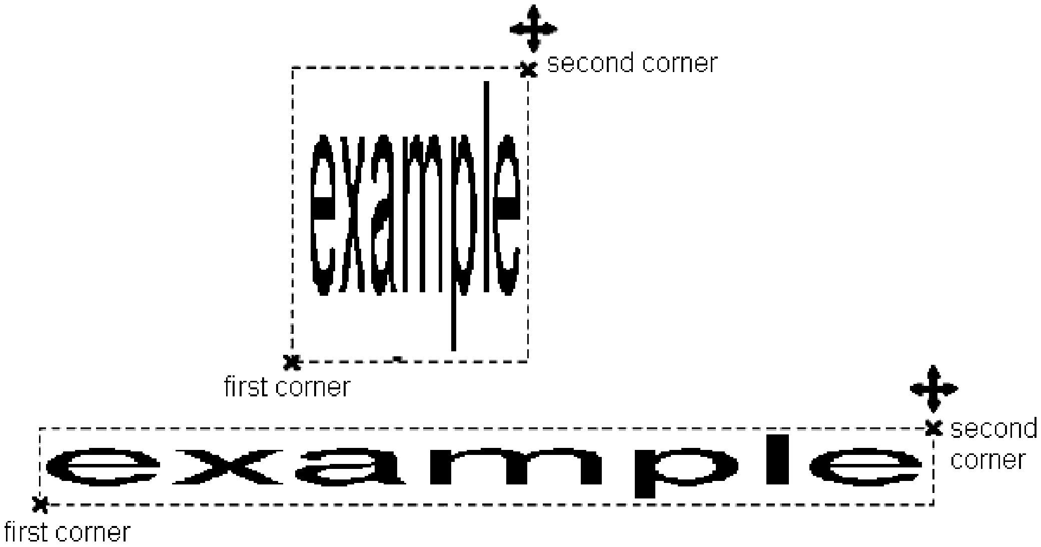
Proceeding:
1. Choose command (with button)
2. Enter text (+ <**ENTER**>)
3. Select text start point

Hints:
Font, size and colour from the variable TXTSPEC (no object snap).

Enter text with scale label into window (disproportional)

Button: 2D + LABEL + TEXT

Command: label * -s (-s = size by window-scaling), -d = disproportional)



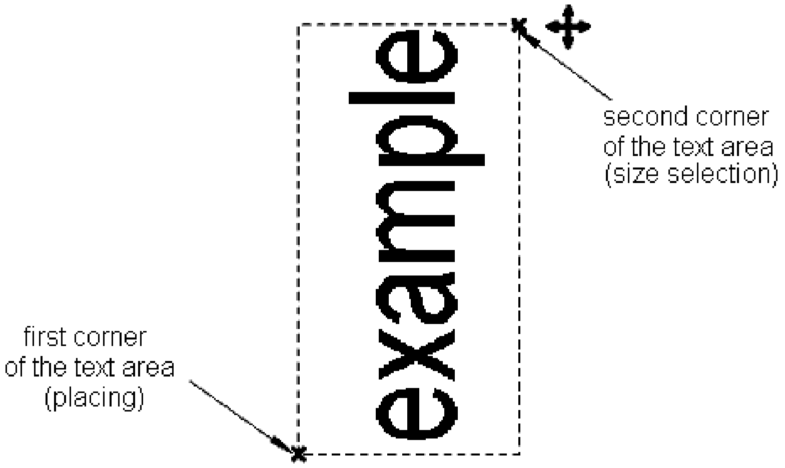
Proceeding:
1. Choose command (with button)
2. Enter text (Complete with <**ENTER**>)
3. Select the first corner of text range
4. Select the second corner of text range

Hints:
The text is fitted into the spanned rectangle exactly (Font and colour from the variable TXTSPEC).

90°- text with scale label into window (proportional)

Button: 2D + LABEL + TEXT

Command: label * -a90 -s (-s = size, window scaling, -a90 = angle: 90°)



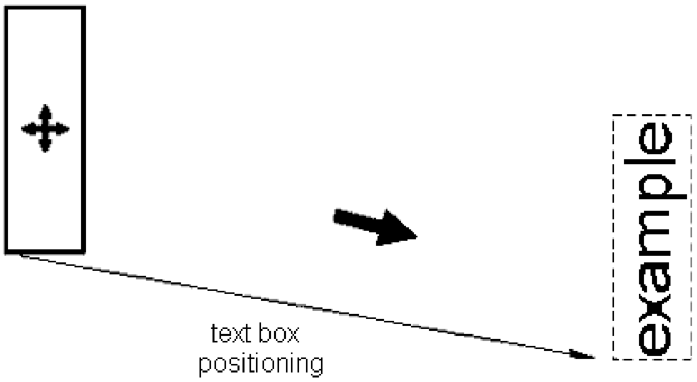
Proceeding:
1. Choose command (with button)
2. Enter text with keyboard (+ <**ENTER**>)
3. Select the first corner of text range
4. Select the second corner of text range

Hints:
The 90°-text is fitted into the spanned rectangle proportionally (Font and colour from the variable TXTSPEC).

90°- text and position label with window

Button: 2D + LABEL + TEXT

Command: label * -a90 (-a90 = angle: 90°)



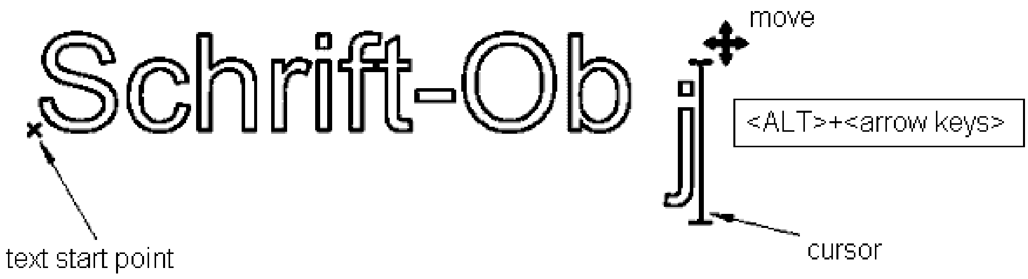
Proceeding:
1. Choose command (with button)
2. Enter text (complete with <**ENTER**>“)
3. Select position of text (place the rectangle with mouse)

Hints:
Font, size and colour are applied from the variable TXTSPEC.

Text with interactive kerning (move)

Button: 2D + LABEL + TEX

Command: kerned



- increase the sliding interval:	< **PgUp** >
- reduce the sliding-interval:	< **PgDn** >
- position the cursors in the text:	< **arrow keys >**

Proceeding:

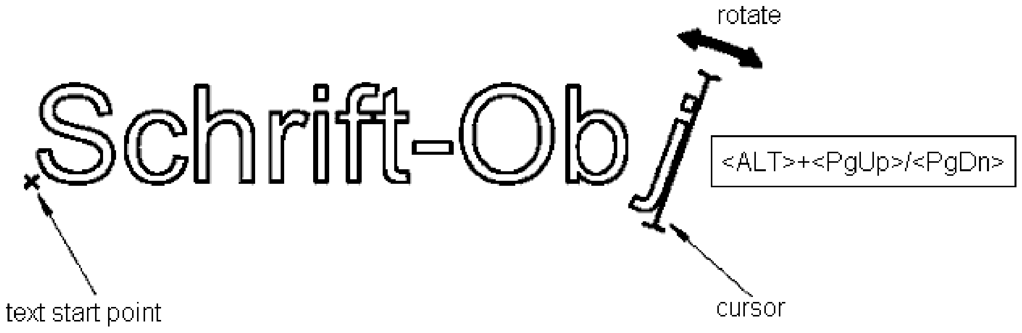
1. Choose command (with button)
2. Set text start-point and enter text
3. Move character (with <**ALT> + <arrow keys**>)
4. New line with: <**ENTER**>, complete with: <**TAB**> or <**F6**>

Hints:
The complete text is generated as object. Further characters are placed beginning from the moving point.

Text with interactive kerning (turn)

Button: 2D + LABEL + TEX

Command: kerned



- Position the cursors in the text with the <arrow keys>

Proceeding:

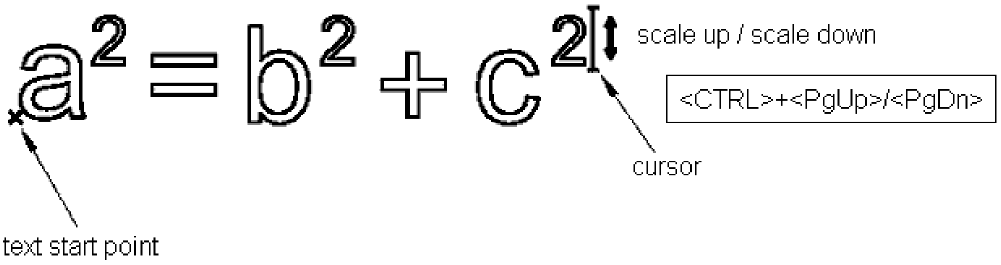
1. Choose command (with button)
2. Set text start point and enter text
3. Turn character (with <**ALT> + <PgUp/PgDn**>)
4. New line with: <**ENTER**>, complete with: <**TAB**> or <**F6**>

Hints:
The complete text is generated as object. Further characters are placed beginning from the moving point.

Text with interactive kerning (scaling)

Button: 2D + LABEL + TEXT

Command: kerned



- Combination with moving and turning is possible.
- Position the cursor in the text with the <arrow keys>

Proceeding:

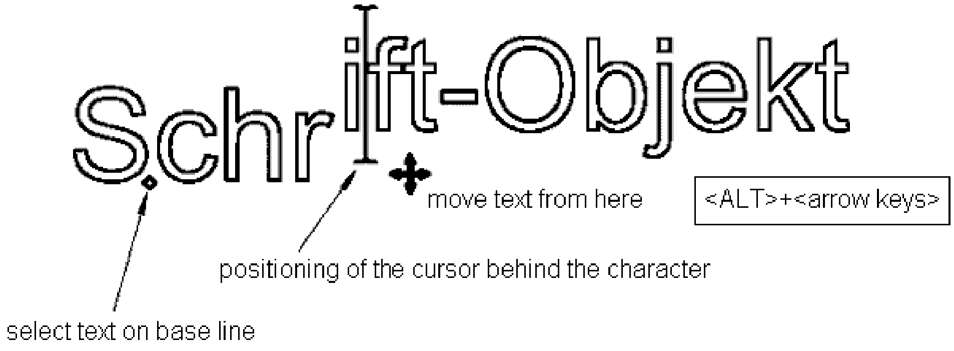
1. Choose command (with button)
2. Set text start-point and enter text
3. Scale character (with <**ALT> + <PgUp/PgDn**>)
4. New line with: <**ENTER**>, complete with: <**TAB**> or <**F6**>

Hints:
The complete text is generated as object. The characters are further placed beginning from the scaling.

Change text with interactive kerning (move)

Button: 2D + LABEL + EDIT

Command: kerned -o? (options: -o <object> , -p <point>)



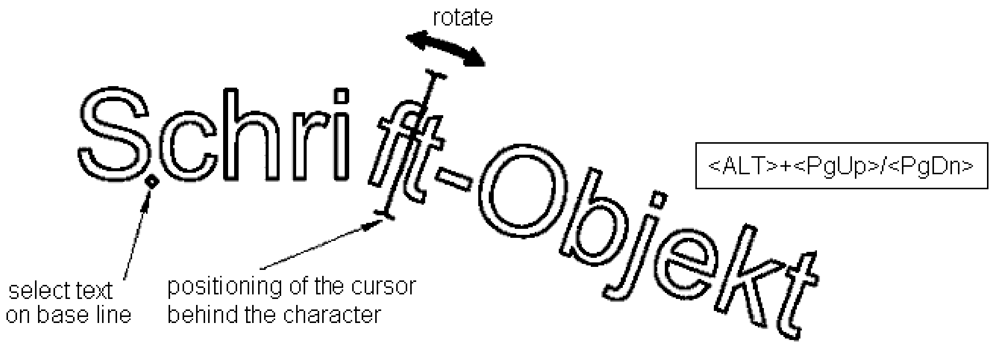
- increase the sliding interval:	< **PgUp** >
- reduce the sliding-interval:	< **PgDn** >
- change text with keys e. g.:	< **BkSp** >

Proceeding:
1. Choose command (with button)
2. Set the cursor to the desired place with arrow keys
3. From here move character (with <**ALT> + <arrow keys**>)
4. New line with: <**ENTER**>, complete with: <**TAB**> or <**F6**>

Hints:
The complete text remains as object. The text is moved beginning with the character in front of the cursor.

Change text with interactive kerning (turn)

Button: 2D + LABEL + EDIT
Command: kerned -o? (options: -o <object> , -p <point>)



- Change text with regular keys e.g.: <BkSp>

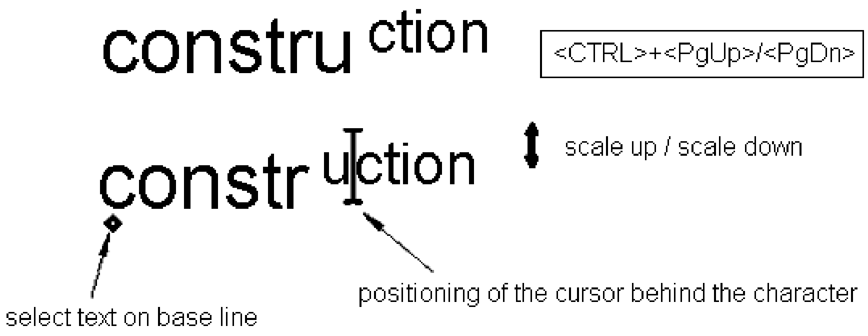
Proceeding:
1. Choose command (with button)
2. Set the cursor to the desired place with arrow keys
3. From here turn character (with <**ALT> + <PgUp/PgDn**>)
4. New line with: <**ENTER**>, complete with: <**TAB**> or <**F6**>

Hints:
The complete text remains as object. The text is turned beginning with the character in front of the cursor.

Change text with interactive kerning (scale)

Button: 2D + LABEL + EDIT

Command: kerned -o? (options: -o <object> , -p <point>)



- Change text with regular keys e.g.: <**BkSp**>
- Combination with moving and turning is possible.

Proceeding:

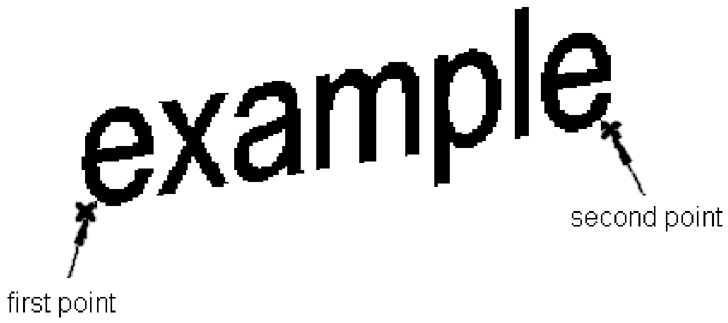
1. Choose command (with button)
2. Set the cursor to the desired place with arrow keys
3. From here scale character (with <**ALT> + <PgUp/PgDn**>)
4. New line with: <**ENTER**>, complete with: <**TAB**> or <**F6**>

Hints:
The complete text remains as object.
The text is scaled beginning with the character in front of the cursor.

Text between two points

Button: 2D + LABEL + TEXT

Command: kerned -o? (options: -o <object> , -p <point>)



Proceeding:

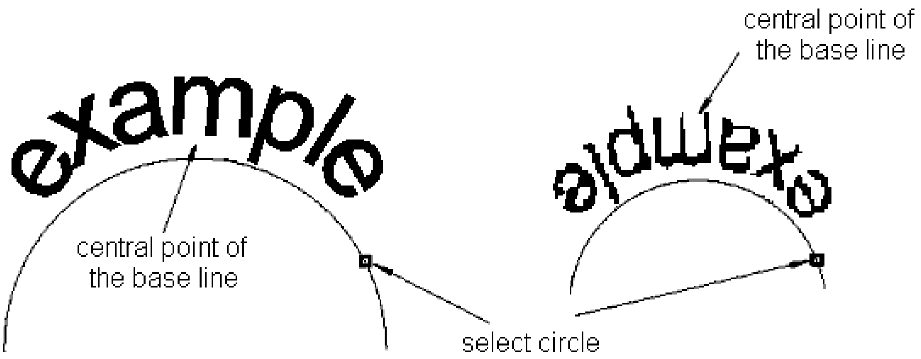
1. Choose command (with button)
2. Select first point
3. Select second point
4. Enter text

Hints:
The label will be placed between the points and the size will be adapted. (Font and colour are applied from the variable TXTSPEC.)

Circle labeling

Button: 2D + LABEL + CIRC

Command: lblcirc (-r = reverse: counter clockwise)



(n)ormal clockwise

(r)everse counter-clockwise

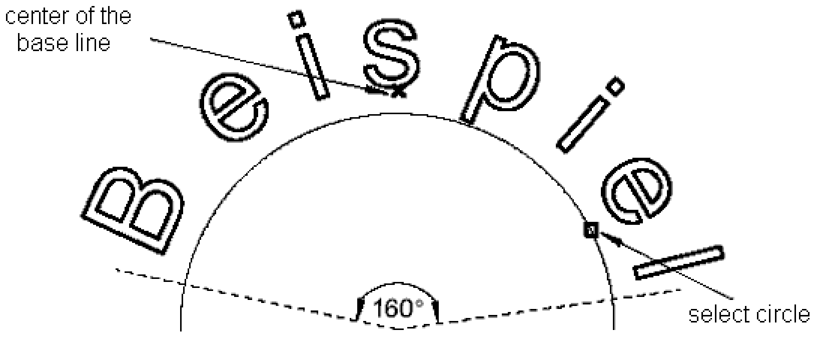
Proceeding:
1. After calling command: (n)ormal or (r)everse
2. Select object (circle)
3. Enter text
4. Determine centre of basic line (selection)

Hints:
Font, size and colour are applied from the variable TXTSPEC.

Text on circle segment

Button: 2D + LABEL + CIRC

Command: lblcirc -w160 (-r = reverse: counter-clockwise, -w = circle segment)



(n)ormal counter-clockwise

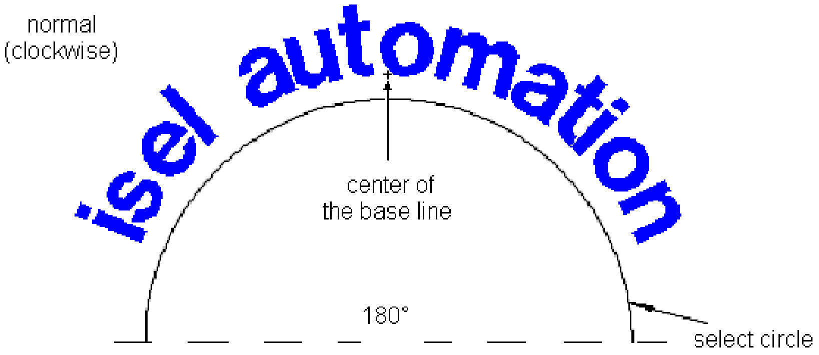
Proceeding:
1. After calling command: Enter angle
2. Enter (n)ormal or (r)everse
3. Select object (circle) and enter text
4. Determine centre of basic line (selection)

Hints:
Font, size and colour are applied from the variable TXTSPEC. The space between two letters will be adapted to the angle.

Text on semi-circle

Button: 2D + LABEL + CIRC

Command: lblcirc -h (-r = reverse: counter-clockwise, -h = semi-circle)



As opposed to the option “-h“ the text is located completely in the sector at the option "-w180" !

Proceeding:

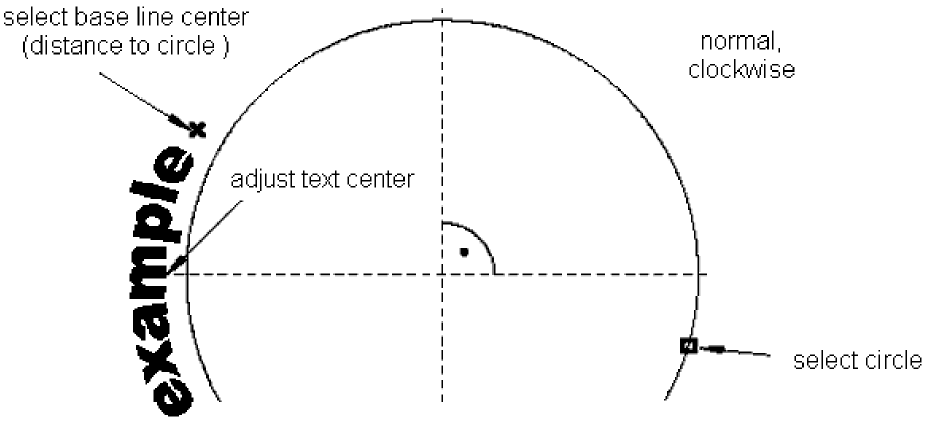
1. After calling command: (n)ormal or (r)everse
2. Select object (circle)
3. Enter text
4. Determine the centre of basic line (selection)

Hints:
Font, size and colour are applied from the variable TXTSPEC. The space between two letters will be adapted to the angle.

Text on circle with snap on quadrant points

Button: 2D + LABEL + CIRC

Command: lblcirc -q (-r = reverse: counter-clockwise, -q = quadrant)



Proceeding:

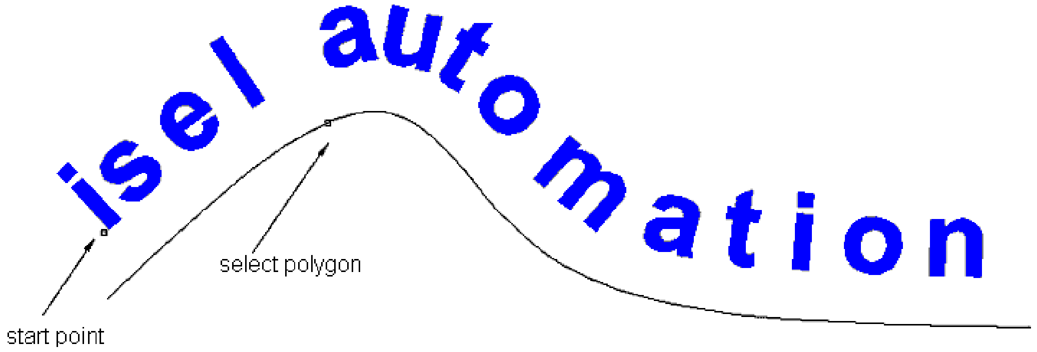
1. After calling command: (n)ormal or (r)everse
2. Select object (circle)
3. Enter text
4. Select distance to the circle (+ quadrant)

Hints:
Font, size and colour are applied from the variable TXTSPEC. The centre of the basic line will be moved to the next quadrant.

Text on curve (polygon)

Button: 2D + LABEL + EX

Command: lblcrv -iw (-i = interactive, -w = wide character pitch: 1,5-fach)



Character Placement:

<**ENTER**>:	Confirm placing of character
positive number "**n**":	Place the next "n" character
negative number "n":	Delete the last "n" character
<key "**d**">:	Determine a new distance of the curve
<key "**z**">:	New zoom sector

Proceeding:

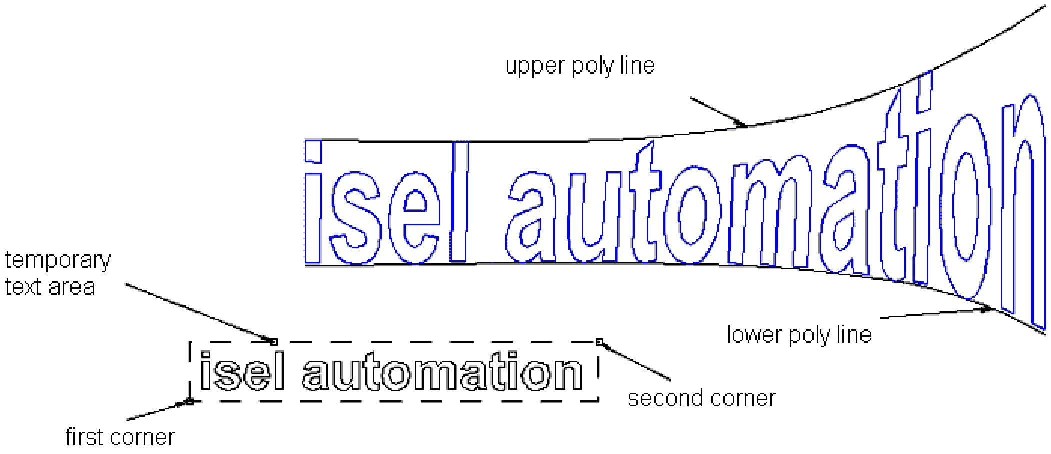
1. After calling command: Select curve (Poly)
2. Enter label
3. Choose start point (minimum distance to curve!)
4. Confirm single the placing of the letters <**ENTER**>

Hints:
Font, size and colour are applied from the variable TXTSPEC.

Text between two curves

Button: 2D + LABEL + EX

Command: pbmenu@MapText



Proceeding:

1. After calling command, enter text
2. Determine text field (is only used as tool and will be deleted again)
3. Select lower poly line
4. Select upper poly line

Hints:

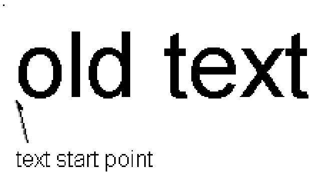
The labeling is changed to Poly automatically and is modulated between the curves.
Look out! The colour of the text corresponds then to the chosen drawing colour!

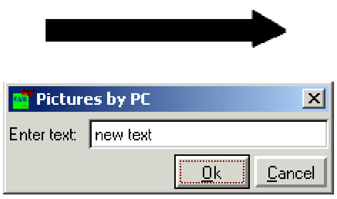
Format labeling

Change text content

Button: 2D + LABEL + TEXT STYLE + OLD NEWTX

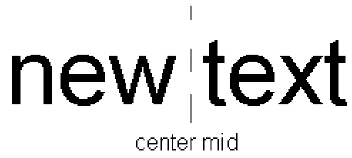
Command: newtext -ec (-e = edit, -c = centre: change centric)





Enter text: new text

Ok Cancel



Proceeding:

1. Choose command (with button)
2. Select object (text-start-point)
3. Edit text content (change)
4. Complete with “Ok“ or <**ENTER**>

Hints:

The changes can be stretched (into the old box) left, right, centric.

Variants to line up text:

left OLD NEWTX **Command:** newtext -el (-l = left)

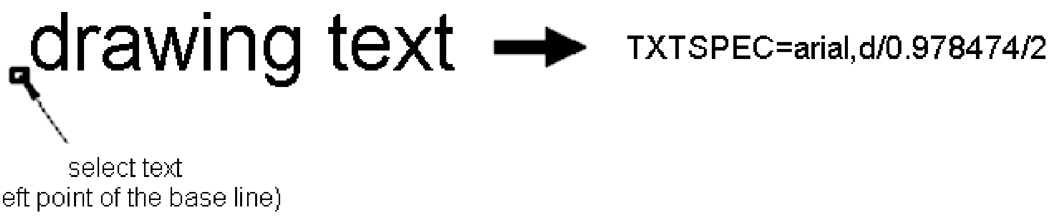
right OLD NEWTX **Command:** newtext -er (-r = right)

stretched OLDTX NEW **Command:** newtext -es (-s = stretched)

Use font, size and colour of the drawing

Button: 2D + LABEL + TEXT STYLE + OLD NEWTX

Command: txtsize -s (-s = store value into the variable TXTSPEC)



Proceeding:

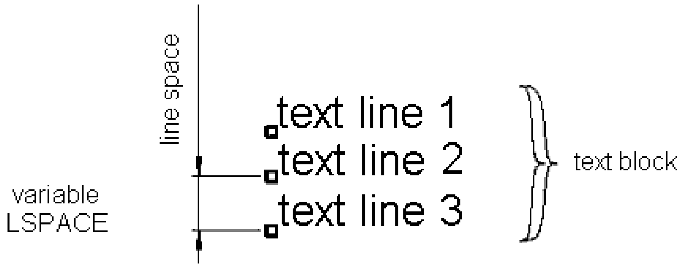
1. Choose command (with button)
2. Select text (on the left point of the base line)

Hints:
After this, all labels are created with these placements.
Cf. variable TXTSPEC

Line space in % of the font size (for text blocks)

Button: 2D + LABEL + TEXT STYLE + TEX TEX

Command: set lspace=120 (120% line space, standard: 190)



Proceeding:

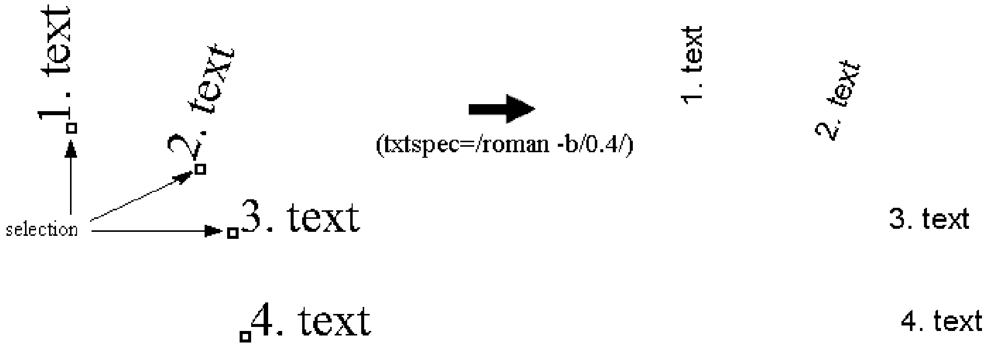
1. Choose command (with button)
2. Enter value of line space

Hints:
The line space for a text block is specified in %, according to the font size (cf. variable LSPACE).

Change font type and size of existing text

Button: 2D + LABEL + TEXT STYLE + TEX TEX

Command: scmmd "chgtxt * -ctso" (-c = centre, -t = from TXTSPEC, -s = size, -o = all text)



Proceeding:

1. Set font and size (TXTSPEC)
2. Choose command with button
3. Select text you want to change (left down !)
4. Complete: Right mouse key or <**F6**> (POLYEND)

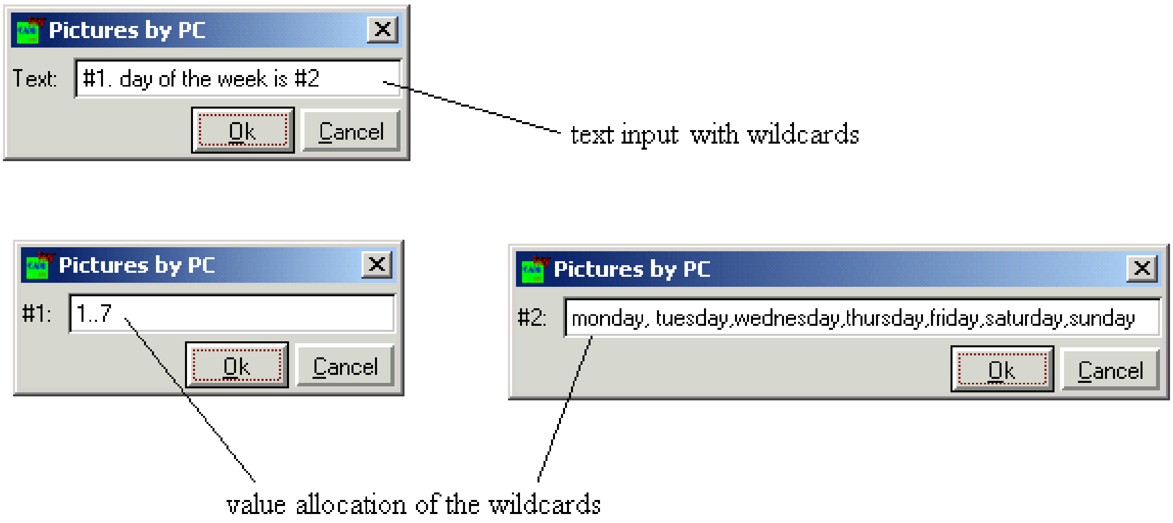
Hints:
Font and size are applied from the variable TXTSPEC.

More labeling

Text with numeration

Button: 2D + LABEL + MORE LABEL + P1,3 P2,2 P3,1

Command: nlabel



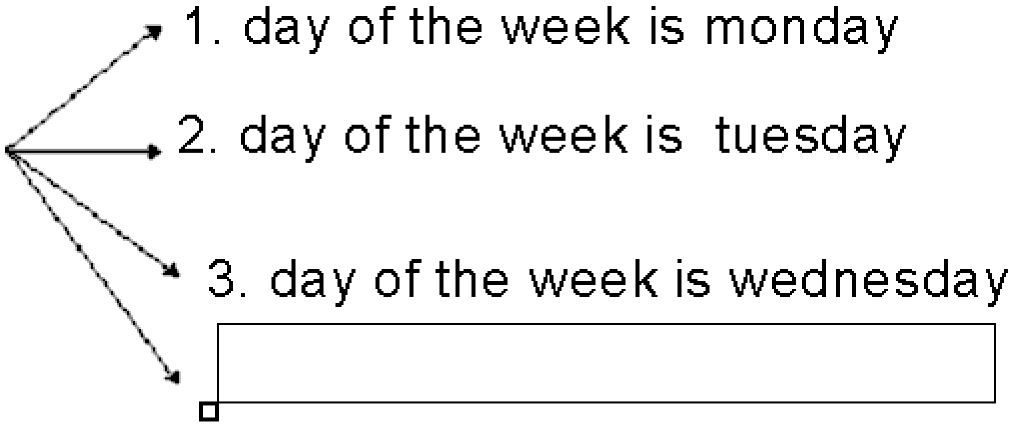
Text: #1. day of the week is #2

#1: 1..7

#2: monday, tuesday,wednesday,thursday,friday,saturday,sunday

Place the windows (continuous in this case)

Selection


1. day of the week is monday
2. day of the week is tuesday
3. day of the week is wednesday

Proceeding:

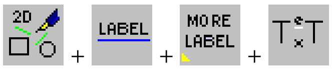
1. Choose command (with button)
2. Enter text (with wildcards, #1,#2, ...)
3. Assign value to the wildcards (end: <**ENTER**>)
4. Place text window, complete with "POLYEND"

Hints:

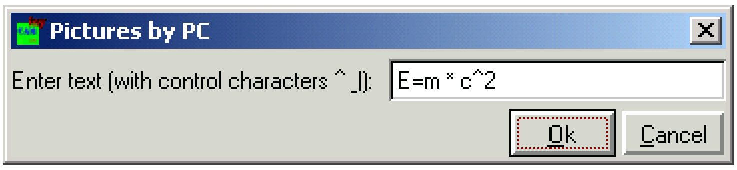
Number ranges (e.g.: 1..7, 3-6) or text fields (e.g.: A,B,C,D or Monday, Tuesday, ...) are the valid replacements for the wildcards.

Labeling with superscript and lowered text

Button:

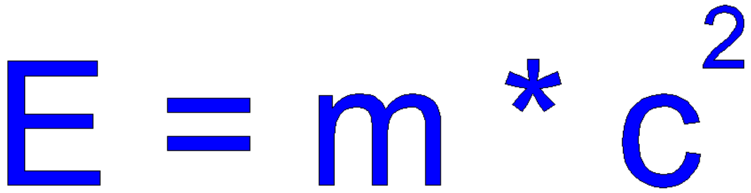


Command: susu



Enter text (with control characters ^ _|): E=m * c^2

After calling command
enter text with control character



Control character:
^ ... superscript text
| ... back to standard text
_ ... lowered text

Proceeding:

1. Choose command (with button)
2. Enter text with control character
3. Place text start point

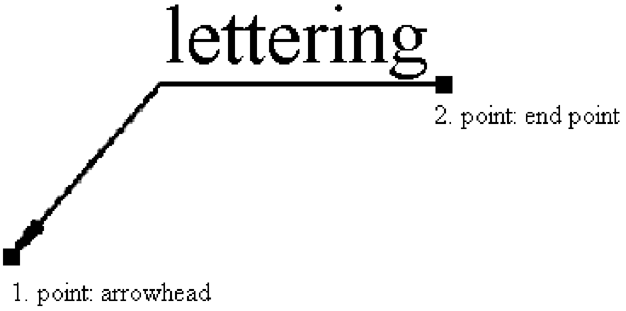
Hints:

Font, size and colour are applied from the variable TXTSPEC. The text will be generated as an object.

Labeling at broken arrow with start- and end-point

Button: 2D + LABEL + MORE LABEL + TEXT

Command: arrow -mfah (-m= multiple, -f= filled, -ah= horizontal basic line)



BD&HT + TEXT At calling button: Filled arrows with HATCH-setting !

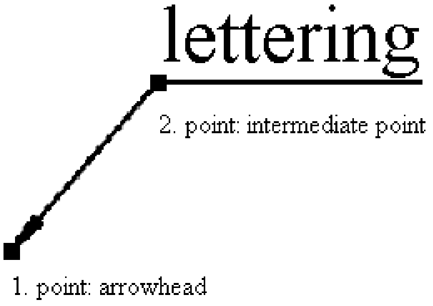
Proceeding:
1. Choose command (with button)
2. Select arrowhead
3. Place end point (of the basic line)
4. Enter text, complete with "POLYEND"

Hints:
Font, size and colour are applied from the variable TXTSPEC and the arrow length from the variable ARR.

Labeling at broken arrow with start-point and intermediate point

Button: 2D + LABEL + MORE LABEL + TEXT

Command: arrow -mfah (-m= multiple, -f= filled, -ah= horizontal basic line)



BD&HT + TEXT At calling button: Filled arrows with HATCH-setting !

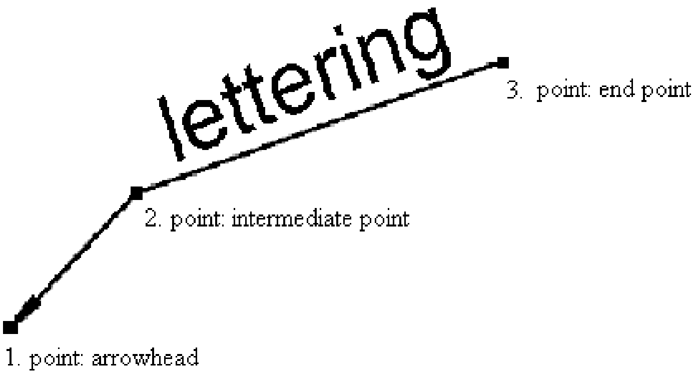
Proceeding:
1. Choose command (with button)
2. Select arrowhead
3. Place end point (of the basic line)
4. Enter text, complete with "POLYEND"

Hints:
Font, size and colour are applied from the variable TXTSPEC and the arrow length from the variable ARR.

Snaked arrow with text

Button: 2D + LABEL + MORE LABEL +

Command: arrow -mfcn (-m= multiple, -f= filled, -cn= count , num)



BD&HT + At calling button: Filled arrows with HATCH-setting !

Proceeding:

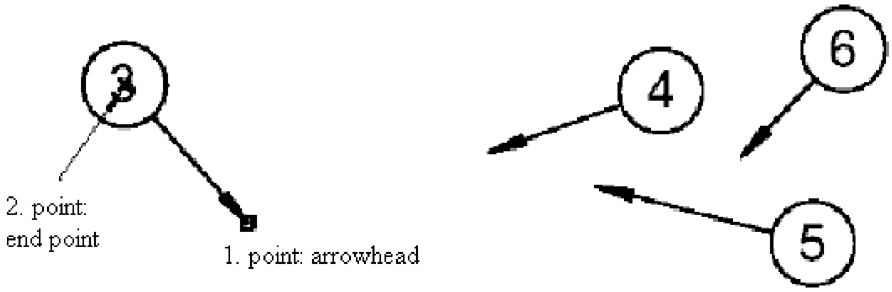
1. Choose command (with button)
2. Enter (start) number: e.g.: “3“ then <**ENTER**>
3. Select arrowhead
4. Place end point, go on with 3. or POLYEND

Hints:
Font, size and colour are applied from the variable TXTSPEC and the arrow length from the variable ARR. Numbering max. 3 digits!

Arrow with automatic numbering with circle

Button: 2D + LABEL + MORE LABEL +

Command: arrow -mfcn (-m= multiple, -f= filled, -cn= count , num)



BD&HT + At calling button: Filled arrows with HATCH-setting !

Proceeding:

1. Choose command (with button)
2. Enter (start) number e.g.: “3“ then <**ENTER**>
3. Select arrowhead
4. Place end point, go on with 3. or POLYEND

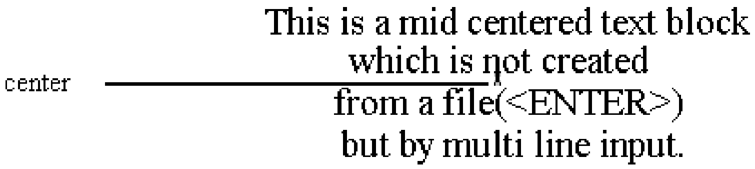
Hints:
Font, size and colour are applied from the variable TXTSPEC and the arrow length from the variable ARR. Numbering max. 3 digits!

Text block

Text block with keyboard (centric)

Button: + +

Command: bl0 -oc (-o= each line an object, block as macro; -c= centric)



This is a mid centered text block
which is not created
from a file(<ENTER>)
but by multi line input.

Proceeding:
1. Choose command (with button)
2. Press <**ENTER**> on keyboard (not from file!)
3. Select centre
4. Enter text, complete line with <**ENTER**>
5. Go on with 3. or <**ENTER**> if block end

Hints:
Font, size and colour are applied from the variable TXTSPEC and the line space from the variable LSPACE.

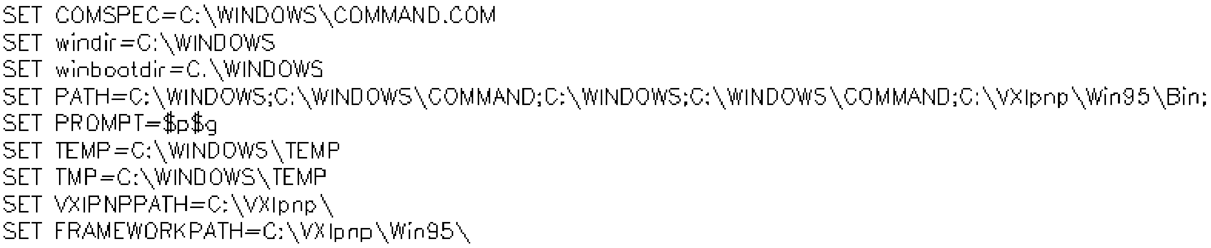
Variants to line up text:

centric		**Command:** bl0 -oc	(-c = centric)
left		**Command:** bl0 -ol	(-c = left)
right		**Command:** bl0 -or	(-r = right)

Read text block from ASCII file into drawing (left-justified)

Button: + +

Command: bl0 -ol (-o= each line an object, block as macro;-l= left)



```
SET COMSPEC=C:\WINDOWS\COMMAND.COM
SET windir=C:\WINDOWS
SET winbootdir=C.\WINDOWS
SET PATH=C:\WINDOWS;C:\WINDOWS\COMMAND;C:\WINDOWS;C:\WINDOWS\COMMAND;C:\VXIpnp\Win95\Bin;
SET PROMPT=$p$g
SET TEMP=C:\WINDOWS\TEMP
SET TMP=C:\WINDOWS\TEMP
SET VXIPNPPATH=C:\VXIpnp\
SET FRAMEWORKPATH=C:\VXIpnp\Win95\
```

Example: Import the content of the file “C:\AUTOEXEC.BAT“ – justification left

Proceeding:

1. Choose command (with button)
2. Enter ASCII file name in the command window: e.g. C:\AUTOEXEC.BAT
3. Select centre

Hints:

Font, size and colour are applied from the variable TXTSPEC and the line space from the variable LSPACE.

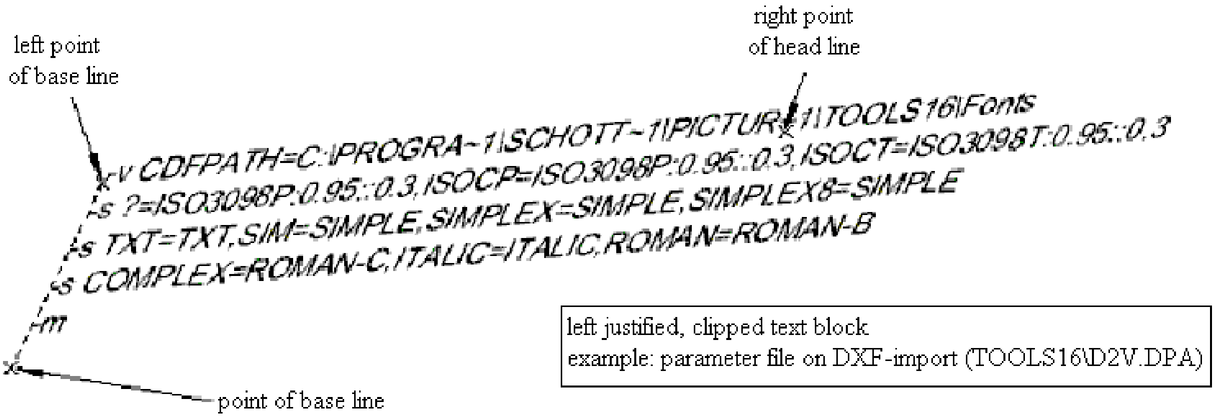
Variants to line up text:

centric	**Command:** bl0 -oc	(-c = centric)
left	**Command:** bl0 -ol	(-c = left)
right	**Command:** bl0 -or	(-r = right)

Text block left-justified, sheared

Button: + +

Command: bl1 -oi (-o= each line an object, block as macro; -i= italic)


left point
of base line
right point
of head line
-v CDFPATH=C:\PROGRA~1\SCHOTT~1\PICTUR~1\TOOLS16\Fonts
-s ?=ISO3098P:0.95::0.3,ISOCP=ISO3098P:0.95::0.3,ISOCT=ISO3098T:0.95::0.3
-s TXT=TXT,SIM=SIMPLE,SIMPLEX=SIMPLE,SIMPLEX8=SIMPLE
-s COMPLEX=ROMAN-C,ITALIC=ITALIC,ROMAN=ROMAN-B
-m
point of base line
left justified, clipped text block
example: parameter file on DXF-import (TOOLS16\D2V.DPA)

Proceeding:

1. Choose command (with button)
2. Enter ASCII file name (TOOLS16\D2V.DPA)
3. 1. and 2. point determine direction of headline
4. 3. point on basic line determines the shearing

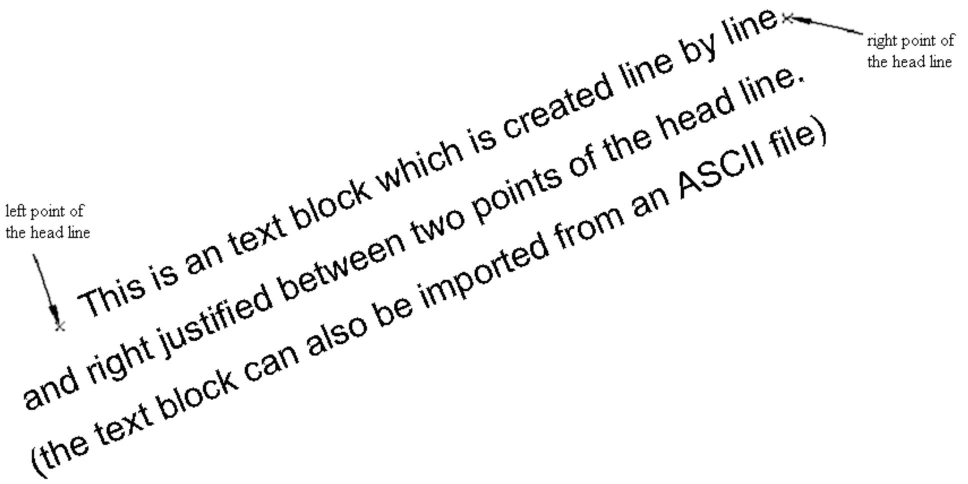
Hints:

Font, size and colour are applied from the variable TXTSPEC and the line space from the variable LSPACE!

Text block right-justified, free direction

Button: 2D + TEXT +

Command: bl1 -or (-o= each line an object, block as macro; -r= right-justified)


right point of
the head line
left point of
the head line
This is an text block which is created line by line
and right justified between two points of the head line.
(the text block can also be imported from an ASCII file)

Proceeding:

1. Choose command (with button)
2. <**ENTER**> (no file), then select 1. + 2. point
3. Enter text , complete line with <**ENTER**>
4. Go on with 3. or <**ENTER**> if block end

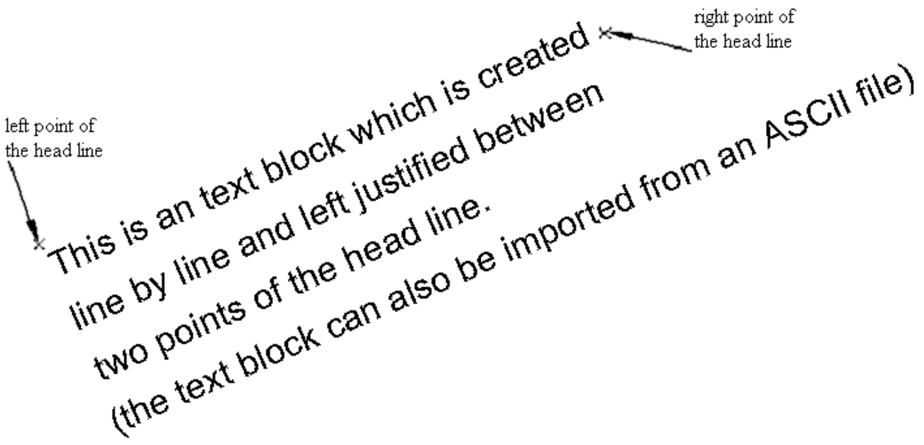
Hints:

Blank line with: <**SPACE**>+<**ENTER**>
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Text block right-justified, free direction

Button: 2D + TEXT +

Command: bl1 -ol (-o= each line is one object, block as macro; -l= left-justified)


right point of
the head line
left point of
the head line
This is an text block which is created
line by line and left justified between
two points of the head line.
(the text block can also be imported from an ASCII file)

Proceeding:

1. Choose command (with button)
2. <**ENTER**> (no file), then select 1. + 2. point
3. Enter text , complete line with <**ENTER**>
4. Go on with 3. or <**ENTER**> if block end

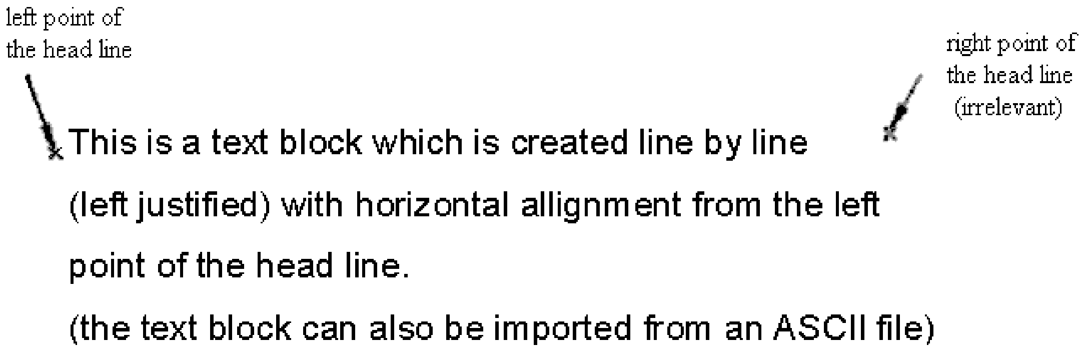
Hints:

Blank line with: <**SPACE**>+<**ENTER**>
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Text block left-justified, horizontal from left side

Button: 2D + TEXT + ≣

Command: bl1 -ol (-o= each line an object, block as macro; -l= left-justified)



This is a text block which is created line by line
(left justified) with horizontal allignment from the left
point of the head line.
(the text block can also be imported from an ASCII file)

Proceeding:
1. Choose command (with button)
2. <**ENTER**> (no file), then select 1. + 2. point
3. Enter text , complete line with <**ENTER**>
4. Go on with 3. or <**ENTER**> if block end

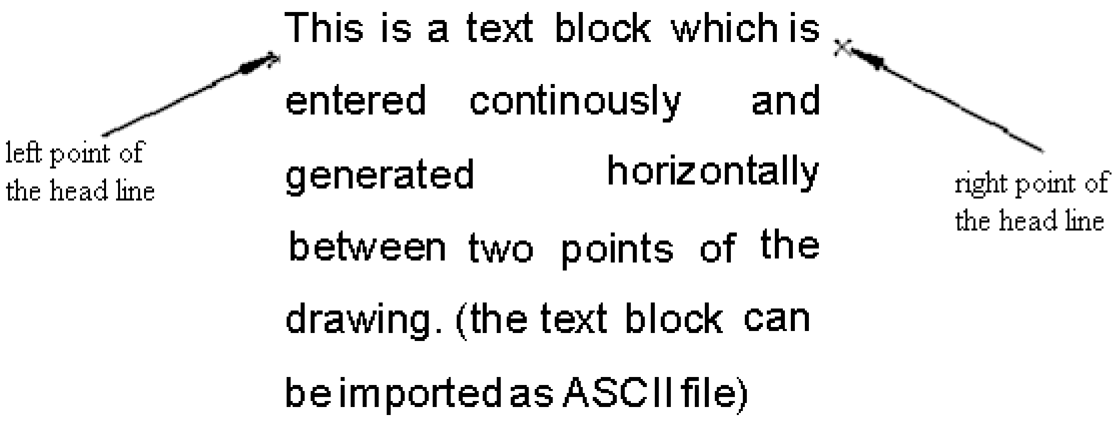
Hints:
Blank line with: <**SPACE**>+<**ENTER**>
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Block set

Text block set with separation, horizontal

Button: 2D + TEXT + ≣

Command: bl2 -oh (-o= each word is one object, block as macro; -h= horizontal)



This is a text block which is
entered continously and
generated horizontally
between two points of the
drawing. (the text block can
be imported as ASCII file)

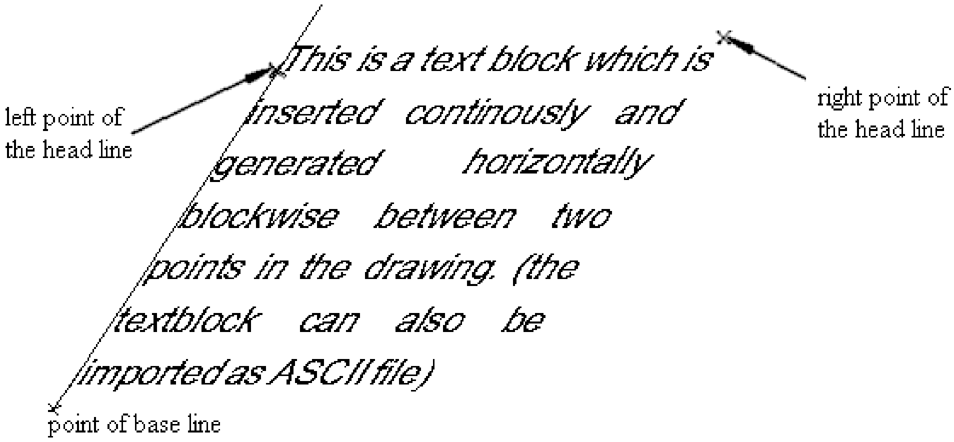
Proceeding:
1. Choose command (with button)
2. Select two points, for block width
3. Enter text continuously, complete with 2x <**ENTER**>
4. If necessary, correct position of hyphen “#“to left

Hints:
New paragraph: <**SPACE**>+<**ENTER**>
Blank line: 2x (<**SPACE**>+<**ENTER**>)
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Text block set inclined array, italic, horizontal

Button: 2D + TEXT +

Command: bl2 -oih (-o= words as objects in macro; -h= horizontal , -i= inclined array)



This is a text block which is
inserted continously and
generated horizontally
blockwise between two
points in the drawing. (the
textblock can also be
importedas ASCII file)

Proceeding:

1. Choose command (with button)
2. 1. and 2. point block width, 3. point block sloped
3. Enter text continuously, complete with 2x <**ENTER**>
4. If necessary, correct position of hyphen “#“to left

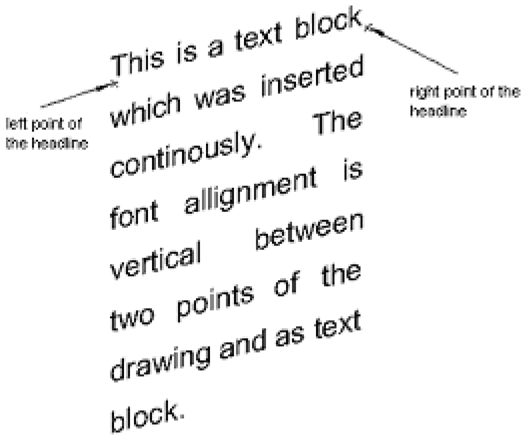
Hints:

New paragraph: <**SPACE**>+<**ENTER**>
Blank line: 2x (<**SPACE**>+<**ENTER**>)
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Text block set, vertical text between two points

Button: 2D + TEXT +

Command: bl2 -oj (-o= words as objects in macro; -j= vertical text)



This is a text block
which was inserted
continously. The
font allignment is
vertical between
two points of the
drawing and as text
block.

Proceeding:

1. Choose command (with button)
2. Select right and left point of headline
3. Enter text continuously, complete with 2x <**ENTER**>
4. If necessary, correct position of hyphen “#“to left

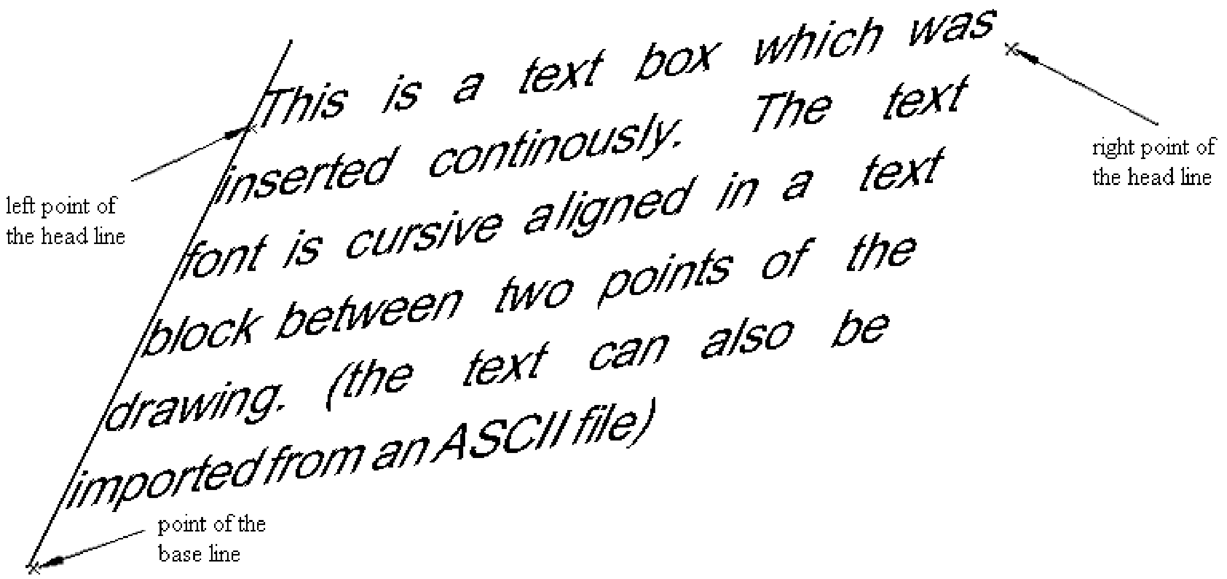
Hints:

New paragraph: <**SPACE**>+<**ENTER**>
Blank line: 2x (<**SPACE**>+<**ENTER**>)
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Text block set, inclined array, italic

Button: 2D + TEXT +

Command: bl2 -ij (-i= inclined array, -j= italic, line up on basic line)


This is a text box which was
inserted continously. The text
font is cursive aligned in a text
block between two points of the
drawing. (the text can also be
imported from an ASCII file)
right point of the head line
left point of the head line
point of the base line

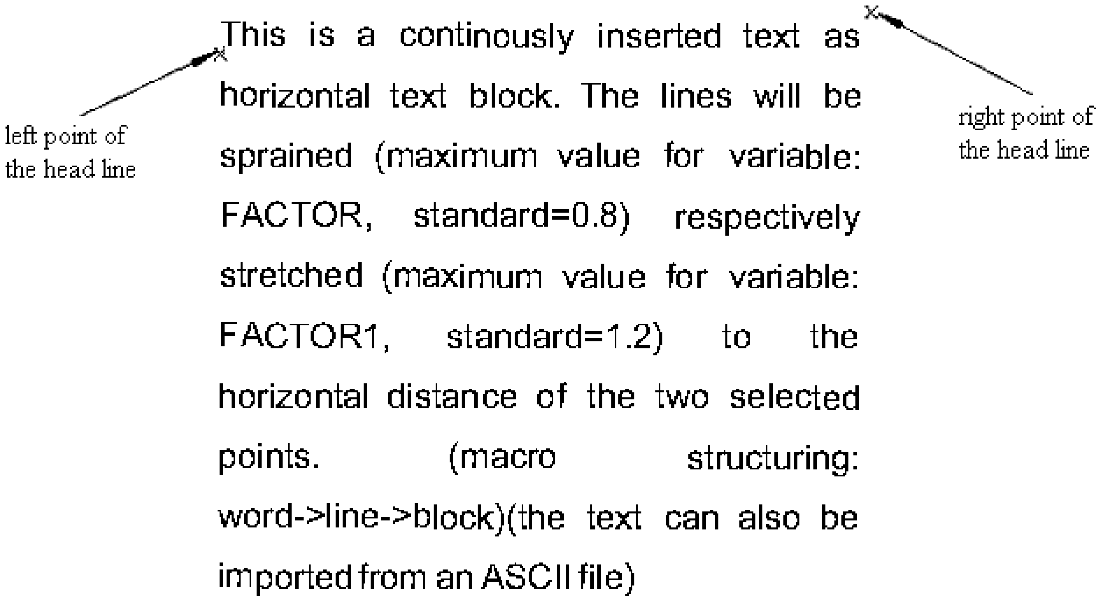
Proceeding:
1. Choose command (with button)
2. 1. and 2. point block width, 3. point block inclined
3. Enter text continuously, complete with 2x <**ENTER**>
4. If necessary, correct position of hyphen “#“to left

Hints:
New paragraph: <**SPACE**>+<**ENTER**>
Blank line: 2x (<**SPACE**>+<**ENTER**>)
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

Text block set, horizontal, line scaling

Button: 2D + TEXT +

Command: bl3 -oh (-o= word as object, lines and blocks as macro; -h= horizontal)



This is a continously inserted text as horizontal text block. The lines will be sprained (maximum value for variable: FACTOR, standard=0.8) respectively stretched (maximum value for variable: FACTOR1, standard=1.2) to the horizontal distance of the two selected points. (macro structuring: word->line->block)(the text can also be imported from an ASCII file)

Proceeding:

1. Choose command (with button)
2. Select two points, for block width
3. Enter text continuously, complete with 2x <**ENTER**>
4. If necessary, correct position of hyphen “#“to left

Hints:

New paragraph: <**SPACE**>+<**ENTER**>
Blank line: 2x (<**SPACE**>+<**ENTER**>)
The fonts are applied from the variable TXTSPEC and the line space from LSPACE!

2D-Manipulation

Overview move object

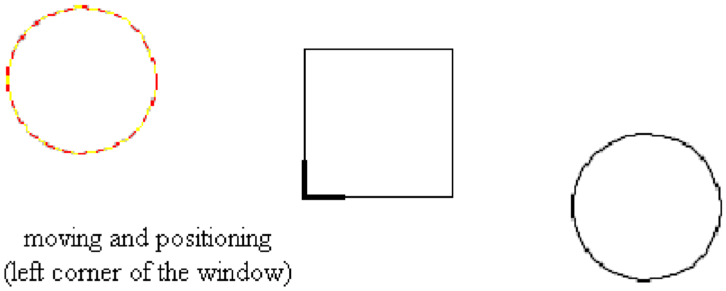
The movement of one or several objects is carried out with the submenu in the toolbar "Tools".

Menu - Button: 2D + MOVE

Move object with window (active object)

Button: 2D + MOVE + MOVE

Command: mvwin -d? *



Proceeding:

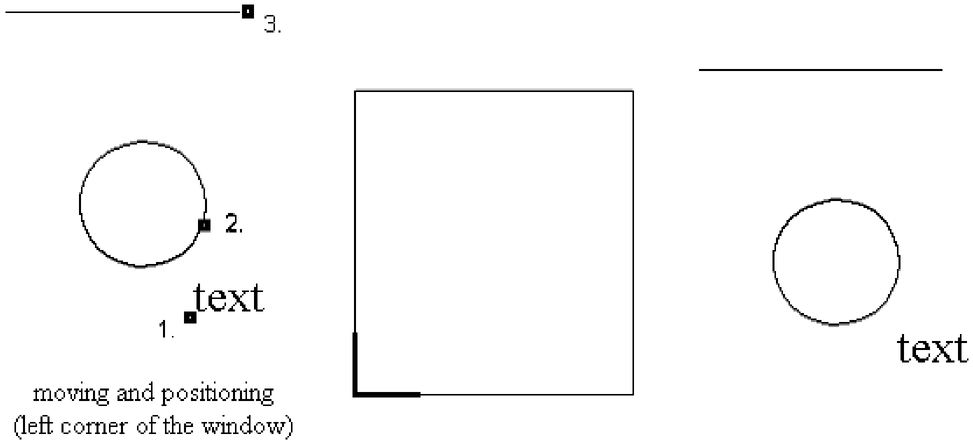
1. Choose function with button or enter command
2. Select the new position

Hints:
The selection of points can be done numerically, key: n.

Move with window (1 or several objects)

Button: 2D + MOVE + MOVE

Command: mvwin -d?



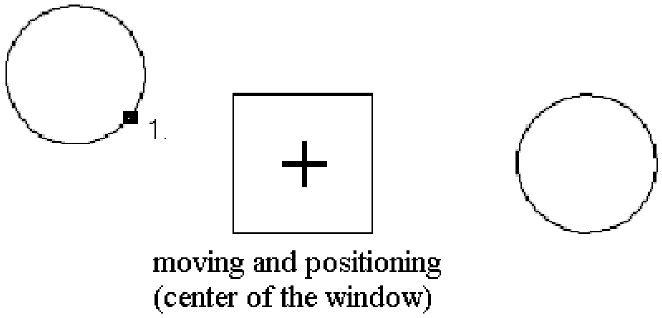
Proceeding:
1. Choose function with button or enter command
2. Object - Select (several with SHIFT-key, complete with POLYEND)
3. Select the new position

Hints:
Macro selection: Key "F10",
Numerical point selection e.g. with key "n".

Move object (centre of the box)

Button: 2D + MOVE +

Command: mvwin -d? *



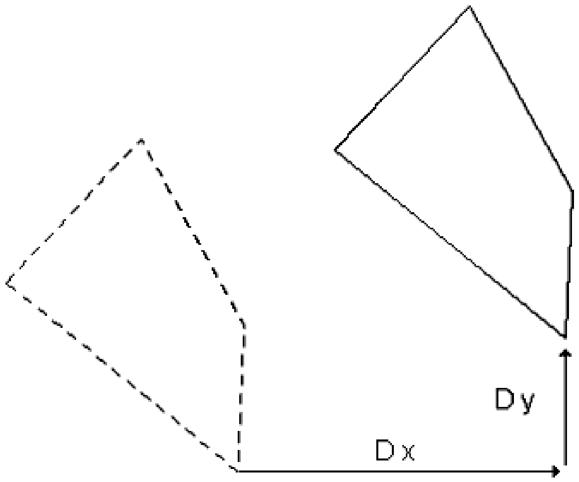
Proceeding:
1. Choose function with button or enter command
2. Select object (several with SHIFT-key, complete with POLYEND)
3. Select the new position

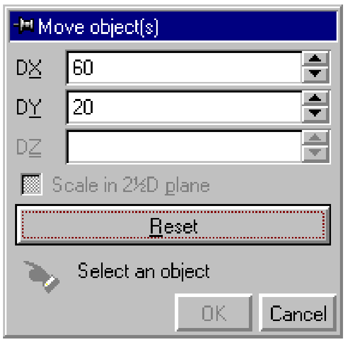
Hints:
Macro selection: Key "F10",
Numerical point selection e.g. with key "n".

Relative object movement (active object)

Button: 2D + MOVE + DY DX

Command: mvwin -d? *





Move object(s)

DX 60

DY 20

DZ

☐ Scale in 2½D plane

Reset

Select an object

Proceeding:

1. Choose function with button or enter command
2. Enter difference values D x, D y
3. Complete with <Ok>

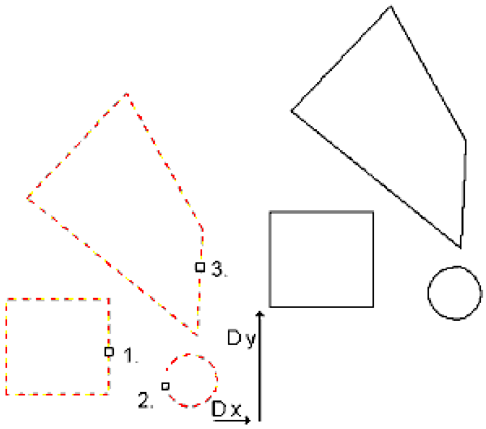
Hints:

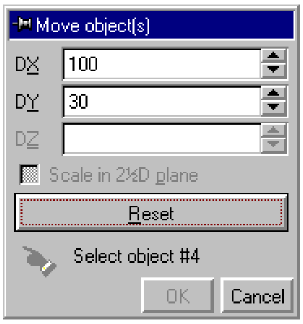
The difference values can be entered also as term (press space key).

Relative object movement (1 or several objects)

Button: 2D + MOVE + DY DX

Command: mv2p -n! -d?





Move object(s)

DX 100

DY 30

DZ

☐ Scale in 2½D plane

Reset

Select object #4

Proceeding:

1. Choose function with button or enter command
2. Enter the difference values D x, D y
3. Select object (several with SHIFT-key, complete selection with POLYEND)

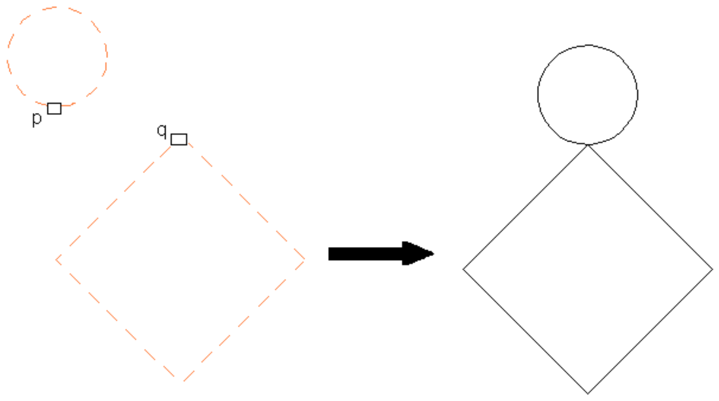
Hints:

The difference values can be entered also as term (press space key).
Macro selection with "F10".

Snapped movement point-point (active object)

Button: 2D + MOVE +

Command: mv2p * -d?



Proceeding:

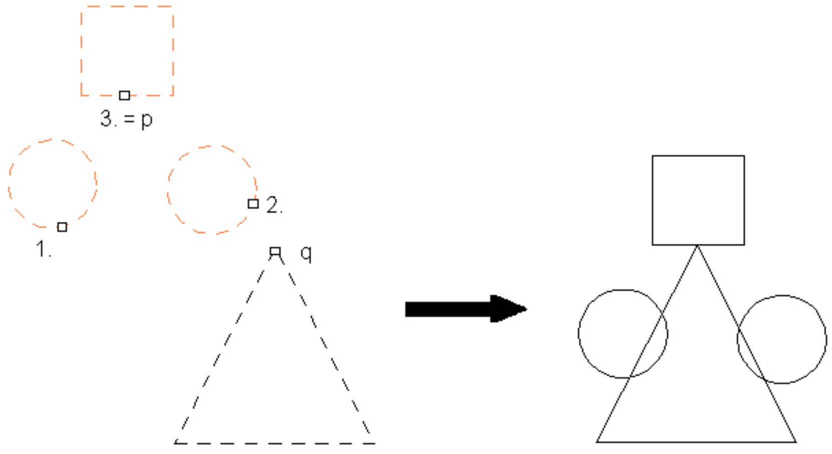
1. Choose function with button or enter command
2. Select the home point p
3. Select the target point q

Hints:
The point selection can be done numerically, key: n.

Snapped movement (1 or several objects)

Button: 2D + MOVE +

Command: mv2p -d? -q



Proceeding:

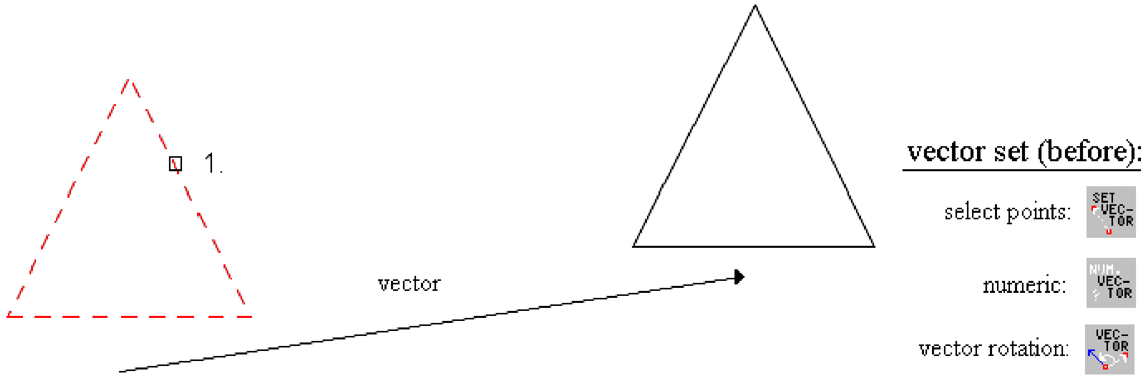
1. Choose function with button or enter command
2. Select object (several with SHIFT-key, complete selection with POLYEND)
3. Select the target point q

Hints:
The last selected point (p) is the home point of the movement.
Select object with SHIFT, then key "S".

Move object, set vector before

Button: 2D + MOVE + VEC-TOR

Command: mtrans $vector



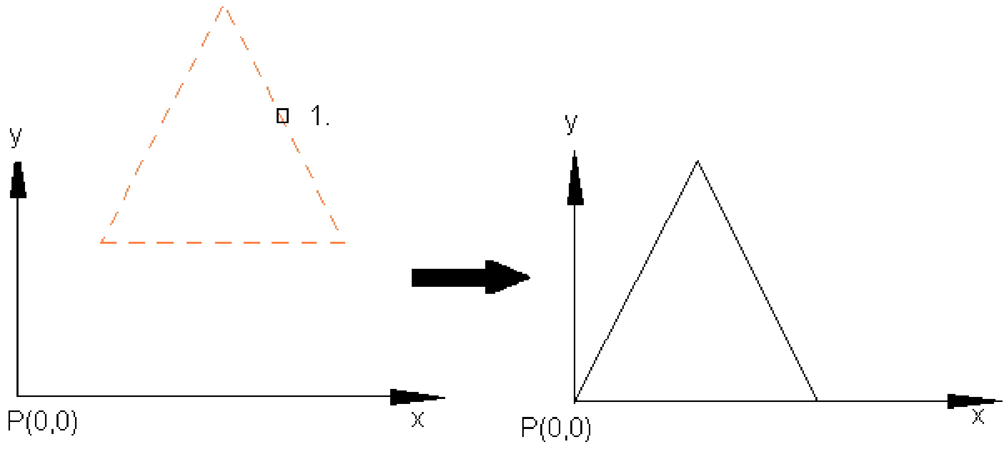
Proceeding:

1. Choose function with button or enter command
2. Select one object/several objects
3. Complete with POLYEND

Hints:
Please set the vector before moving, otherwise the relative moving will be questioned.

Move the object box to point of source

Button: 2D + MOVE + 0,0



Proceeding:

1. Choose function with button
2. Select object

Hints:
Simultaneous selection of several objects with SHIFT-key.

Overview rotate object

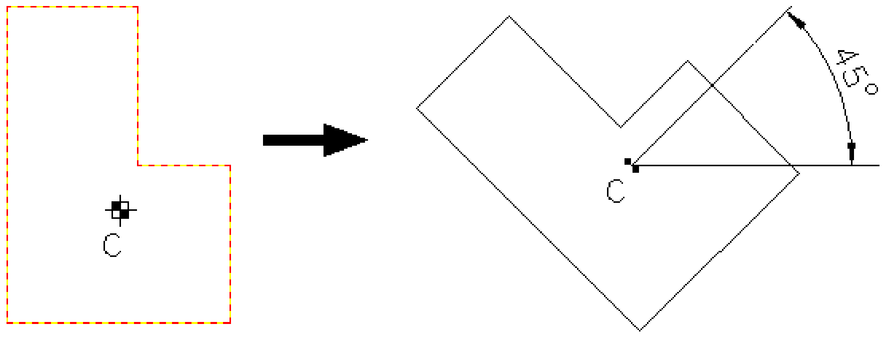
The rotation of one or several objects is carried out with the buttons in the menu “Rotation“ in the toolbar "Tools".

Menu - Button: 2D + ROT

Rotate object around a predefined angle (e.g. 45°)

Button: 2D + ROT + ROT 45

Command: rot * 45 -c * (-c * = centre of rotation is the centre of the object box)
resp. rot * $dpl 45 -hc * (with dialog box)



Proceeding:

1. Select the object
2. Choose function with button or enter command

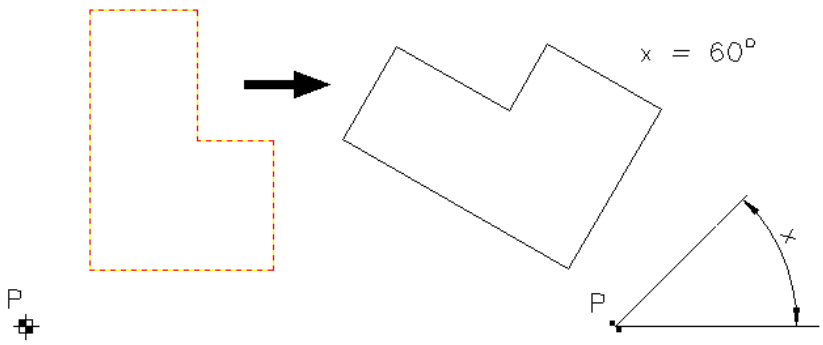
Hints:
Rotation counter-clockwise (mathematical positive).

Rotate object with angle and point selection

Button: 2D + ROT + ROT

Command: rot * a (a = number, enter negative numbers in brackets)

or: orot * -d? (with dialog box)



Proceeding:
1. Select the object
2. Choose function with button
3. Enter rotation angle a
4. Select the rotation point P

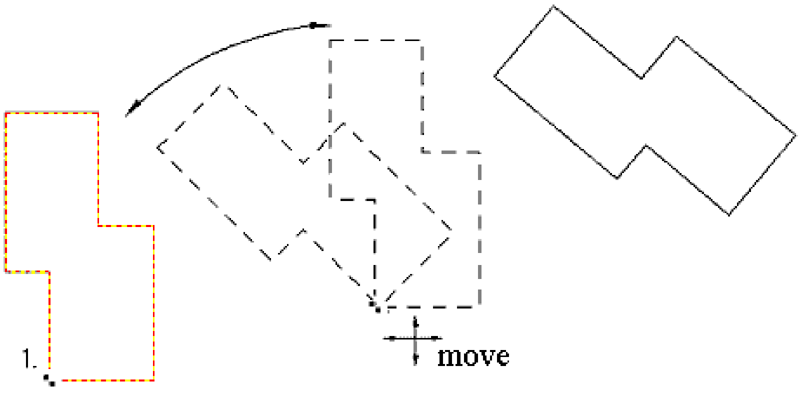
Hints:
Rotation counter-clockwise (mathematical positive).

Dynamic movement with rotation

Button: 2D + ROT +

Command: mvrt -d?

Rotation step-by-step with keys PgUp/PgDn: ↑ ↓



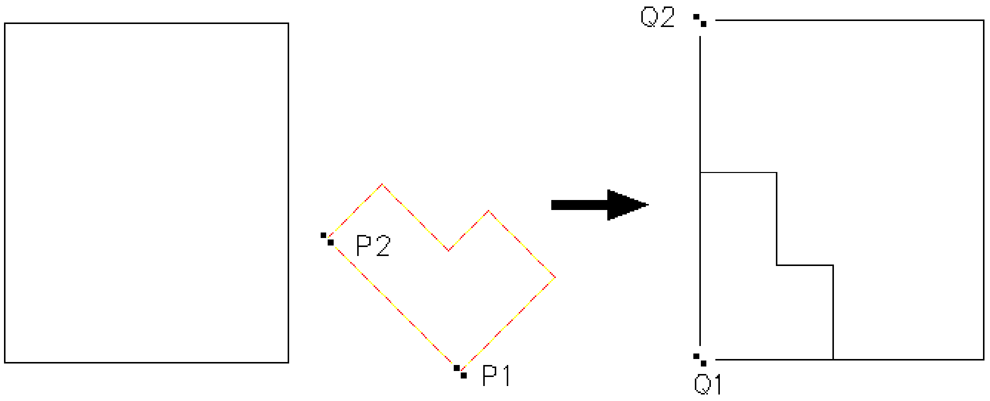
Proceeding:
1. Choose function with button or enter command
2. Select the object on rotation point
3. Move/rotate the object
4. Complete with left mouse key

Hints:
The default value of the rotation angle is 5°. The keys “+“ and “-“ doubles resp. halves the current value.

Rotate and move object (2 reference points)

Button: 2D + ROT +

Command: mv4p * -d? (no size adaption)



Proceeding:

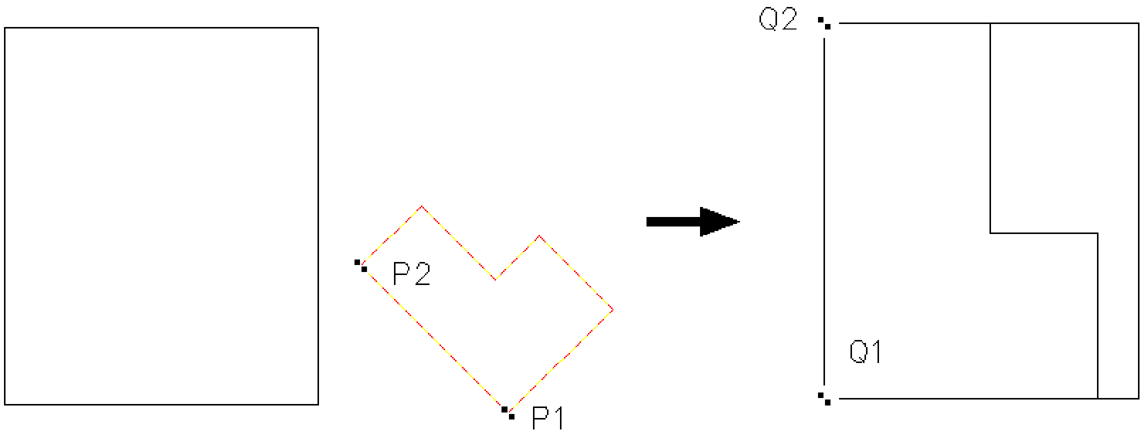
1. Select the object
2. Choose function with button or enter command
3. Select the start points (P1,P2)
4. Select the target points (Q1,Q2)

Hints:
The start point P1 is assigned to the target point Q1.
The second point pair determines only the direction.

Rotate, move, scale the object

Button: 2D + ROT +

Command: mv4p * -d? -s



Proceeding:

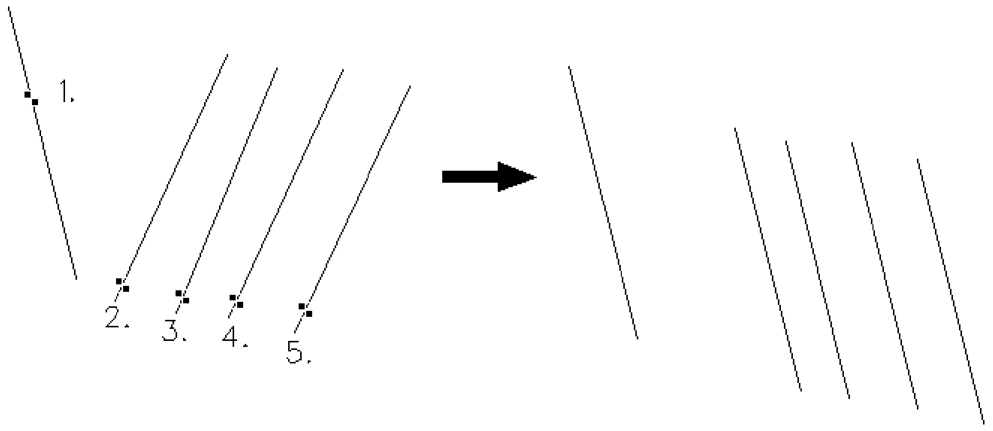
1. Select the object
2. Choose function with button or enter command
3. Select the start points (P1,P2)
4. Select the target points (Q1,Q2)

Hints:
The start point P1 is assigned to the target point Q1.
The second point pair determines only the direction.

Line up lines parallel to a reference line

Button: + +

Command: plines



Proceeding:

1. Choose function with button or enter command
2. Select the reference line
3. Select the line that will be lined up
4. Complete with right mouse key or POLYEND

Hints:

The rotation of the lined-up lines is carried out around its centre.

Overview Changing Object Form

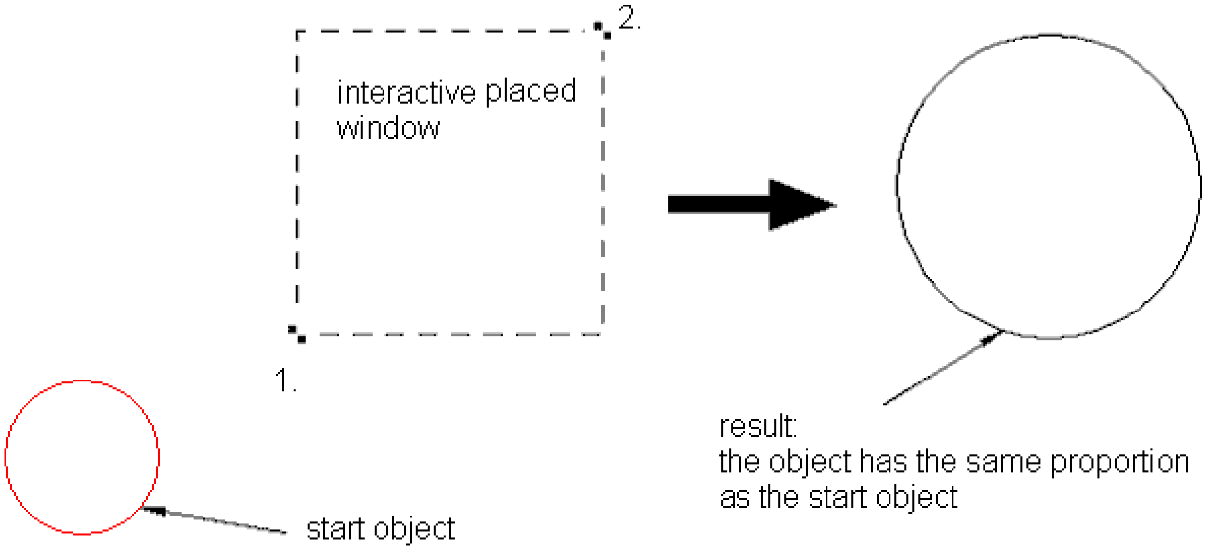
The form changing of objects is carried out with the buttons in menu “Change form“ in the toolbar “Tools“.

Menu - Button: 2D + CHG. FORM

Proportional scaling with interactive window

Button: 2D + CHG. FORM + SCALE

Command: scale * -r


2.
interactive placed window
1.
start object
result:
the object has the same proportion as the start object

Proceeding:

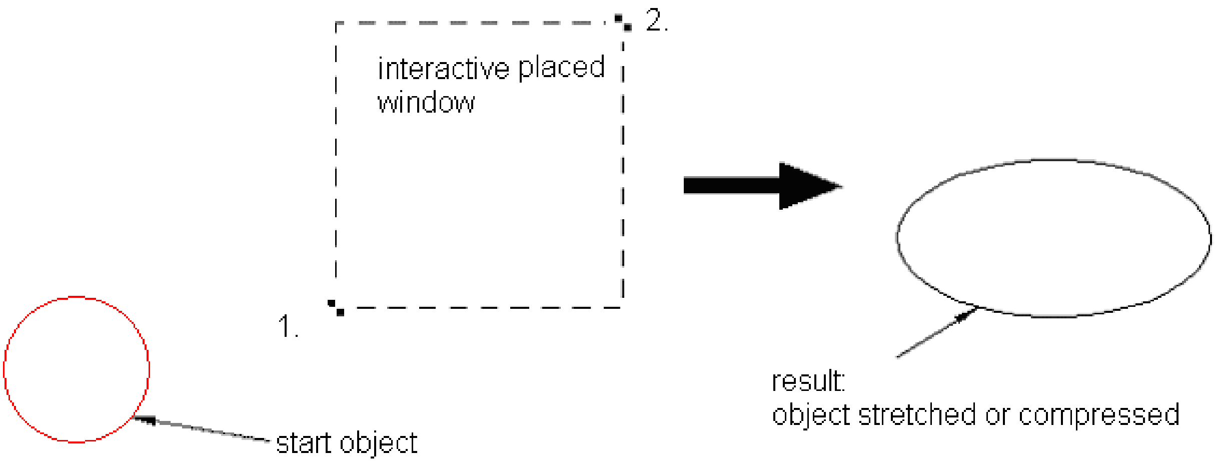
1. Object must be active (select it first)
2. Choose function with button or enter command
3. Determine position and size with circumscribed rectangle

Hints:
New position and size are determined by the circumscribed rectangle.

Disproportional scaling with interactive window

Button: 2D + CHG. FORM + DE-FORM

Command: scale *



Proceeding:

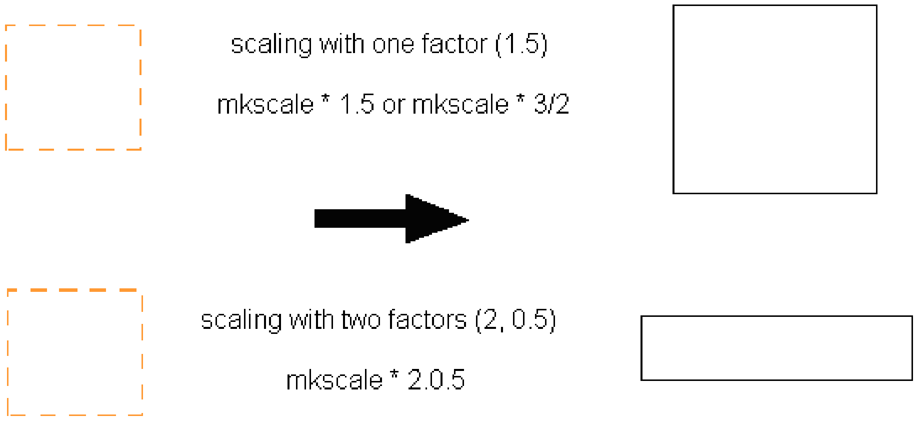
1. Object must be active (select it first)
2. Choose function with button or enter command
3. Determine position and size with circumscribed rectangle

Hints:
New position and size are determined by the circumscribed rectangle.

Numerical scaling with relative factor(s)

Button: 2D + CHG. FORM + SCALE NUM. REL.

Command: mkscale * rx [,ry]


scaling with one factor (1.5)
mkscale * 1.5 or mkscale * 3/2
scaling with two factors (2, 0.5)
mkscale * 2 0.5

Proceeding:

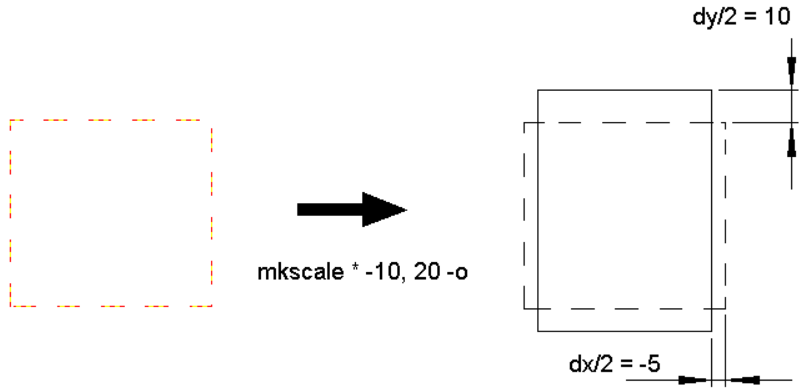
1. Object must be active (select it first)
2. Choose function with button or enter command
3. Enter the scaling factors

Hints:
The box centre remains unchanged. The procedure is self-calculating. Enter term as factor.

Numerical scaling with absolute size change

Button: 2D + CHG. FORM + SCALE NUM ABSOL

Command: mkscale * -o dx [,dy]



Proceeding:

1. Object must be active (select it first)
2. Choose function with button or enter command
3. Enter the absolute size changing

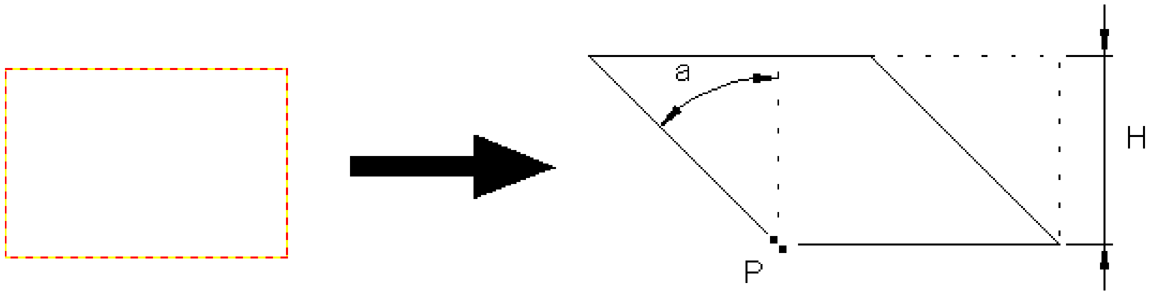
Hints:

The box centre remains unchanged. The procedure is self-calculating. Enter term as factor .One numerical value changes the horizontal and vertical dimensions to the equal value.

Horizontal parallel clipping (active object)

Button: 2D + CHG. FORM + ▱

Command: yaw * -hp (-h = horizontal, -p = parallel)



Proceeding:

1. Select object
2. Choose function with button or enter command
3. Select the reference point P
4. Enter the clipping angle a

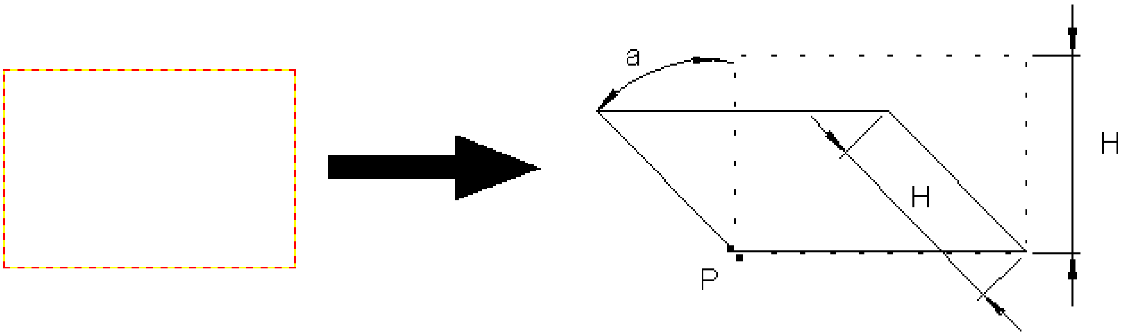
Hints:

Please pay attention: The angle is always entered counter-clockwise.

Horizontal clipping (active object)

Button: 2D + CHG. FORM +

Command: yaw * -hp (-h = horizontal, -p = parallel)



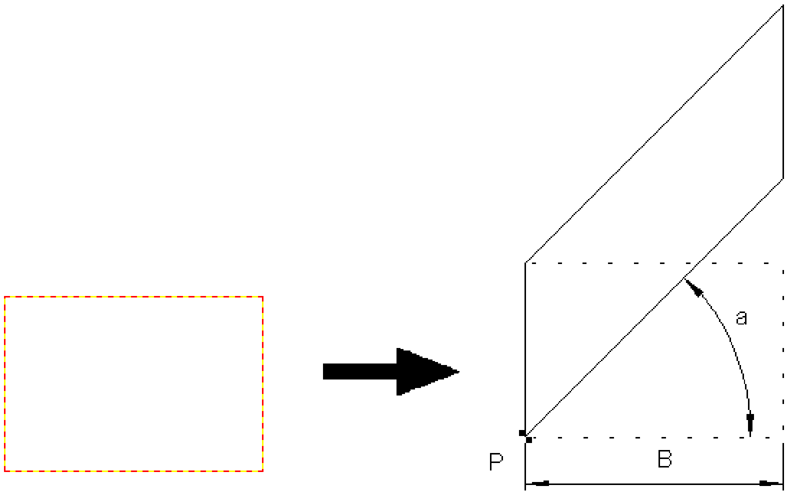
Proceeding:
1. Select object
2. Choose function with button or enter command
3. Select the reference point P
4. Enter the clipping angle a

Hints:
Please pay attention: The angle is always entered counter-clockwise.

Vertical parallel clipping (active object)

Button: 2D + CHG. FORM +

Command: yaw * -vp (-v = vertical, -p = parallel)



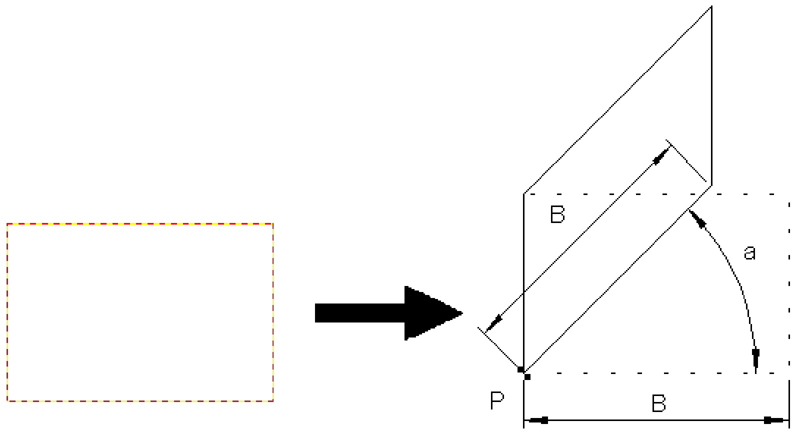
Proceeding:
1. Select object
2. Choose function with button or enter command
3. Select the reference point P
4. Enter the clipping angle a

Hints:
Please pay attention: The angle is always entered counter-clockwise.

Vertical clipping (active object)

Button: 2D + CHG. FORM +

Command: yaw * -vp (-v = vertical)



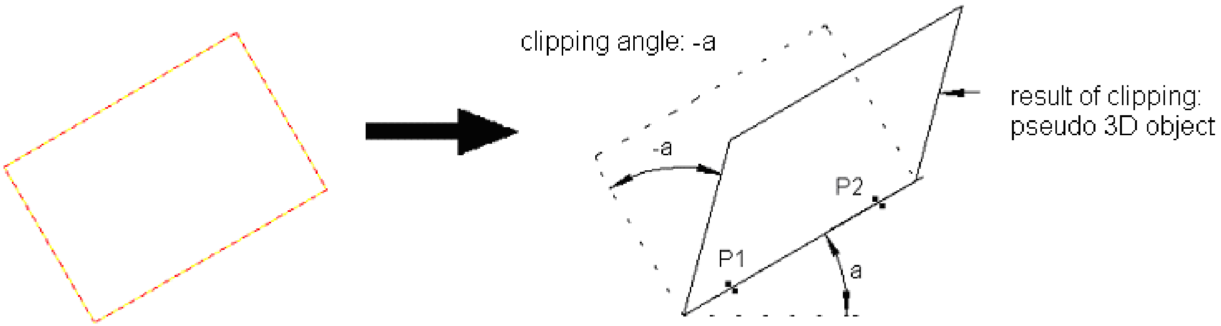
Proceeding:
1. Select object
2. Choose function with button or enter command
3. Select the reference point P
4. Enter the clipping angle a

Hints:
Please pay attention: The angle is always entered counter-clockwise.

Clipping on basic line (active object)

Button: 2D + CHG. FORM +

Command: yaw *



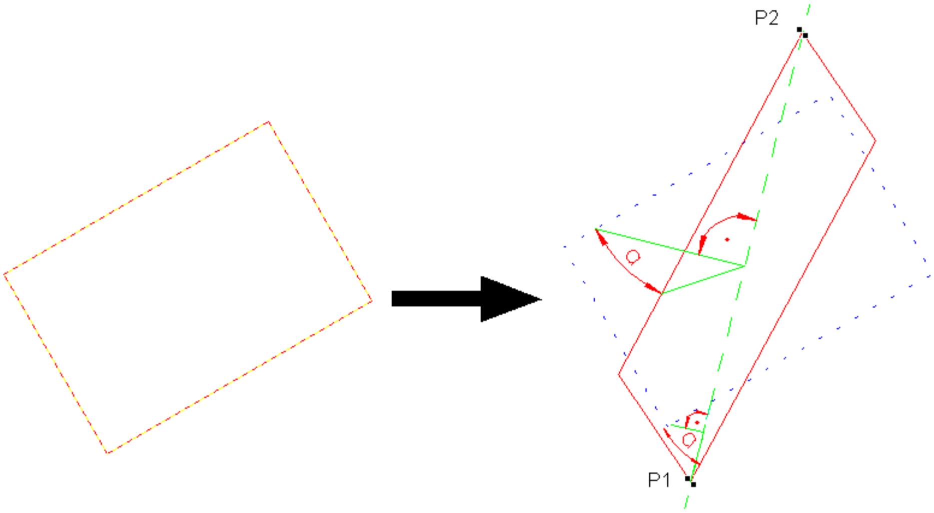
Proceeding:
1. Select object
2. Choose function with button or enter command
3. Select the reference points P1 and P2
4. Enter the clipping angle a

Hints:
Please pay attention: The angle is always entered counter-clockwise.

Clipping on arbitrary axis (active object)

Button: 2D + CHG. FORM +

Command: yaw *



Proceeding:

1. Select object
2. Choose function with button or enter command
3. Select the reference points P1 and P2
4. Enter the clipping angle a

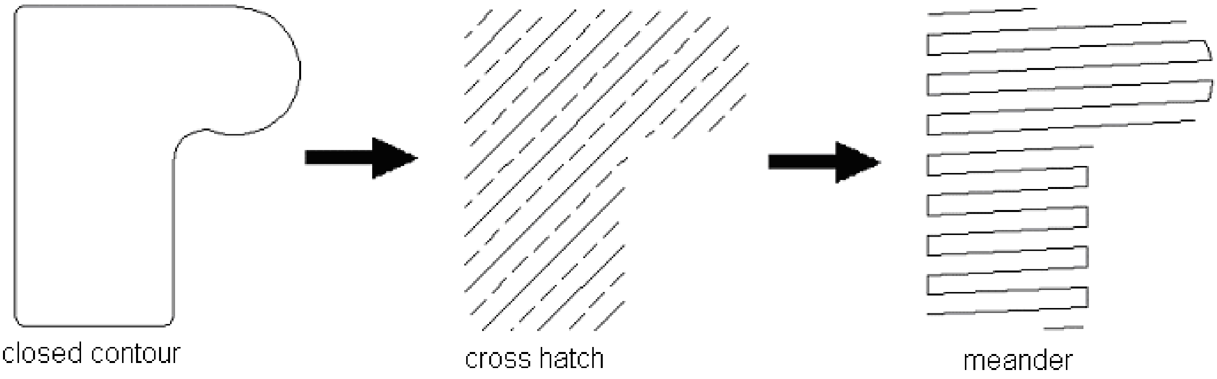
Hints:

Please pay attention: The angle is always entered counter-clockwise.

Converting inclined hatch to meander

Button: 2D + CHG. FORM + MORE CHGES +

Command: meander * (with option -d the source object will be deleted)



Proceeding:

1. If necessary, select hatch
2. Choose function with button or enter command
3. The meander polygon is generated as duplicate

Hints:

The hatch type must be "DIRCT". Meander helps e.g. to generate a specific milling paths.

Introduction mirror objects

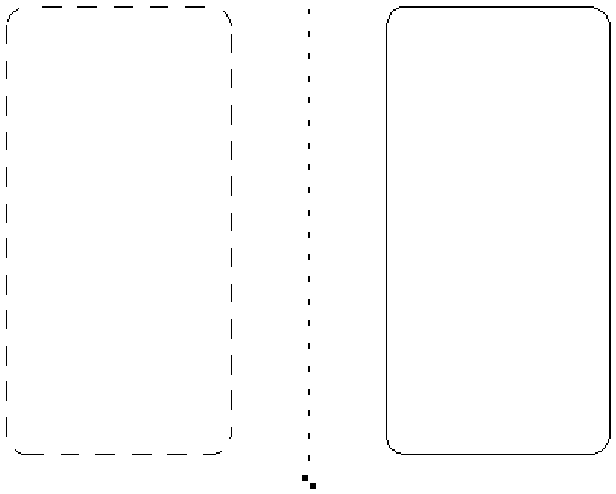
The mirroring of one or several objects is carried out with the button-functions in the menu “Rotate” of the tool bar "Tools".

Menu - Button: 2D + MIRR.

Horizontal mirroring at selection point

Button: 2D + MIRR. + MIRR×

Command: mirror * -y (-y = axis parallel to Y-axis)
or: mirror * * -y **or:** mirror * $dpl -y
Button-Menu uses **Bix-Routine:** mir2p * -y -d?



Proceeding:

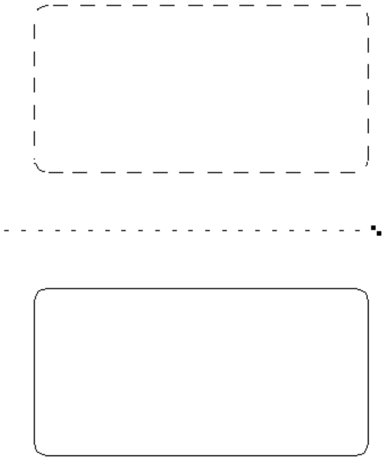
1. Object must be active, select it first
2. Choose function with button or enter command
3. Select a point on the axis (snap or numerical)

Hints:
The point selection can be carried out numerically.
Generate a duplicate with active duplicating button.

Vertical mirroring at selection point

Button: 2D + MIRR. +

Command: mirror * -x (-x = axis parallel to X-axis)
or: mirror * * -x **or:** mirror * $dpl -x
Button-Menu uses **Bix-Routine:** mir2p * -x -d?



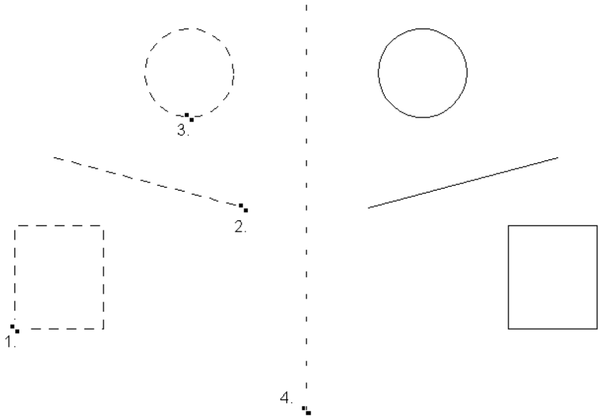
Proceeding:

1. Object must be active, select it first
2. Choose function with button or enter command
3. Select a point on the axis

Hints:
The point selection can be carried out numerically.
Generate a duplicate with active duplicating button.

Multiple horizontal mirroring at a point

Button: 2D + MIRR. + MIRR



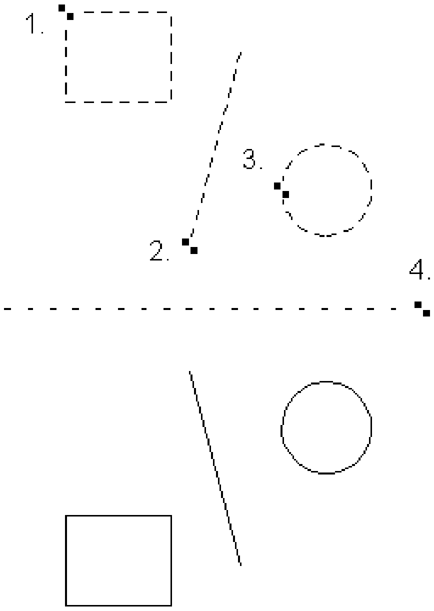
Proceeding:

1. Choose function with button
2. Select the objects (1. with SHIFT, complete with POLYEND)
3. Select a point of axis

Hints:
The point selection can be carried out numerically.
Generate a duplicate with active duplicating button.

Multiple vertical mirroring at a point

Button: 2D + MIRR. + ▨



Proceeding:

1. Choose function with button
2. Select the objects (1. with SHIFT, complete with POLYEND)
3. Select a point of axis

Hints:

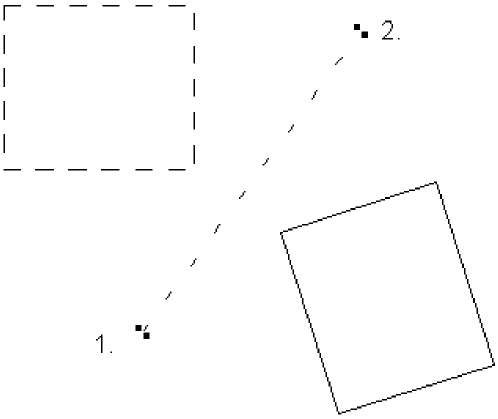
The point selection can be carried out numerically.

Generate a duplicate with active duplicating button.

Mirroring at a line

Button: 2D + MIRR. + 2P

Command: mir2p *



Proceeding:

1. Object must be active, select it first
2. Choose function with button or enter command
3. Select a point on the axis

Hints:

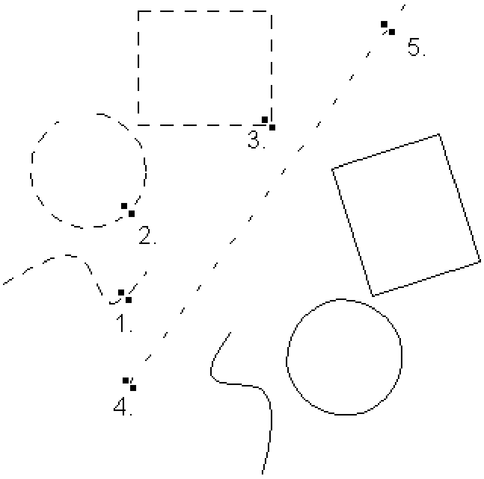
The point selection can be carried out numerically.

Generate a duplicate with active duplicating button.

Multiple mirroring at a line

Button: + +

Command: mir2p



Proceeding:

1. Choose function with button
2. Select the objects (1. with SHIFT, complete with POLYEND)
3. Select a point of axis

Hints:

The point selection can be carried out numerically.
Generate a duplicate with active duplicating button.

Overview Trim Functions

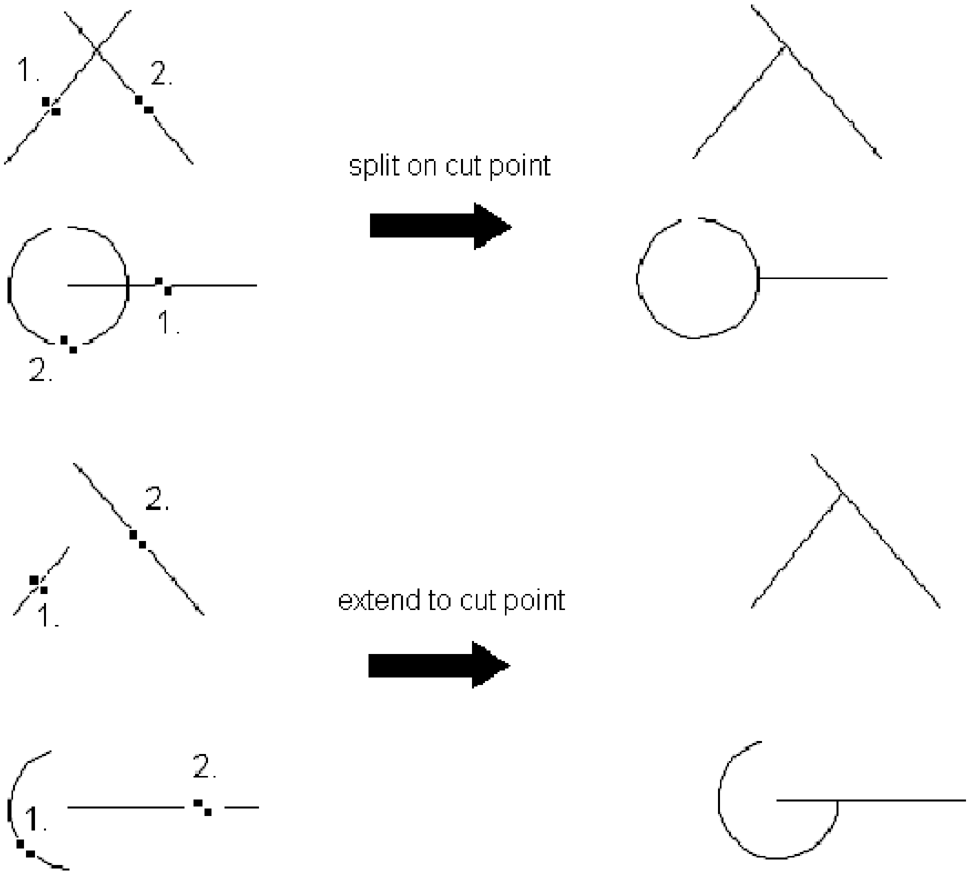
The trim functions are called with buttons in the tool bar "Tools".

Menu - Button: 2D + TRIM

Trim the first element at intersection

Button: 2D + TRIM + TRIM 1.

Command: trimx -am (-a=first element, -m=multiple)



Proceeding:

1. Choose function trimx
2. Select the elements to change
3. Selection the "boundary-" elements
4. Complete: Right mouse key or POLYEND

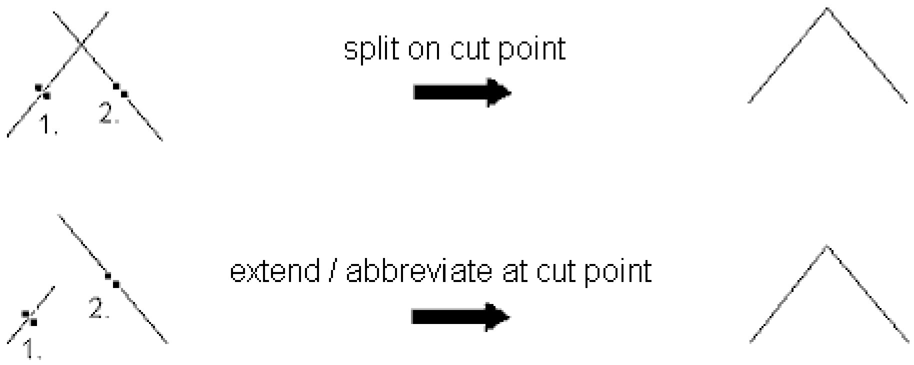
Hints:

Basically: Please select the elements on the part, that shall remain unchanged.

Trim two elements at intersection

Button: 2D + TRIM + TRIM

Command: trimx -m (-m=multiple)



Proceeding:

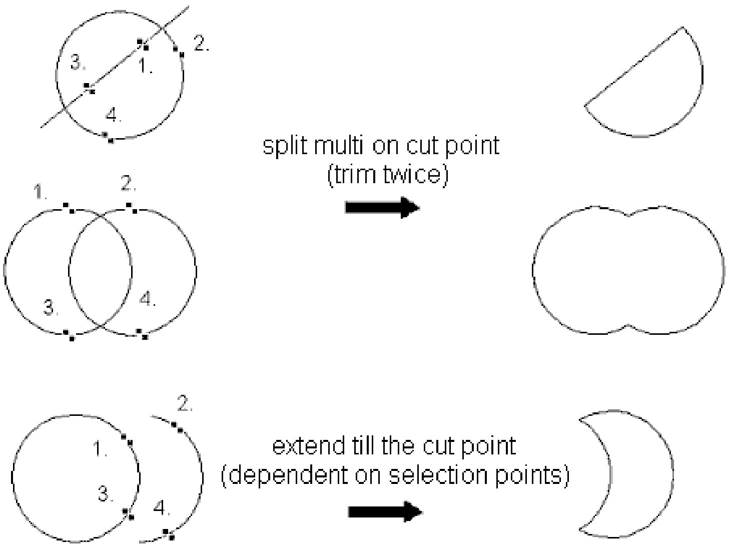
1. Choose function trimx
2. Select the 1. element
3. Select the 2. element
4. Complete: Right mouse key or POLYEND

Hints:
Basically: Please select the elements on the part, that shall remain unchanged.

Trim two elements at intersection(multiple)

Button: 2D + TRIM + TRIM

Command: trimx -m (-m=multiple)



Proceeding:

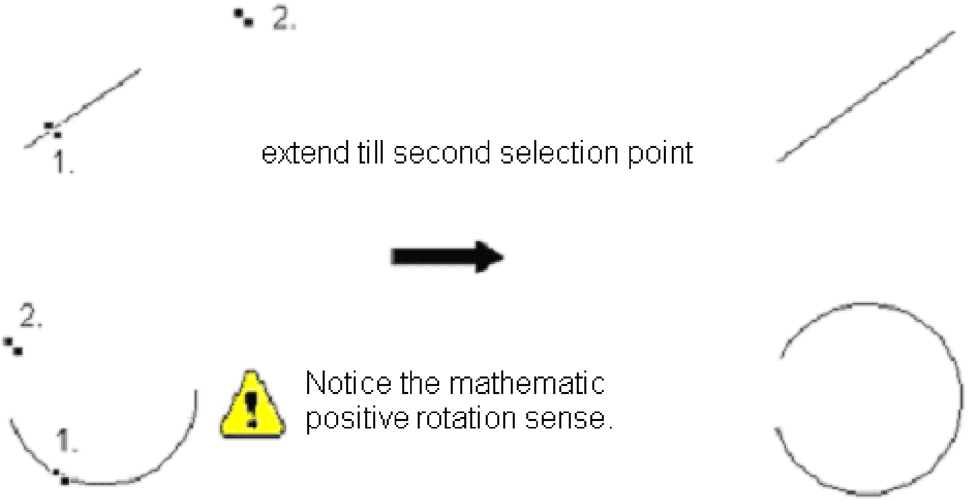
1. Choose function trimx
2. Select the elements near the 1. point
3. Select the elements near the 2. point
4. Complete: Right mouse key or POLYEND

Hints:
Basically: Please select the elements on the part, that shall remain unchanged.

Extend elements

Button: 2D + TRIM +

Command: trimx -lm (-l=length, -m=multiple)


2.
1.
extend till second selection point
2.
1.
Notice the mathematic positive rotation sense.

Proceeding:

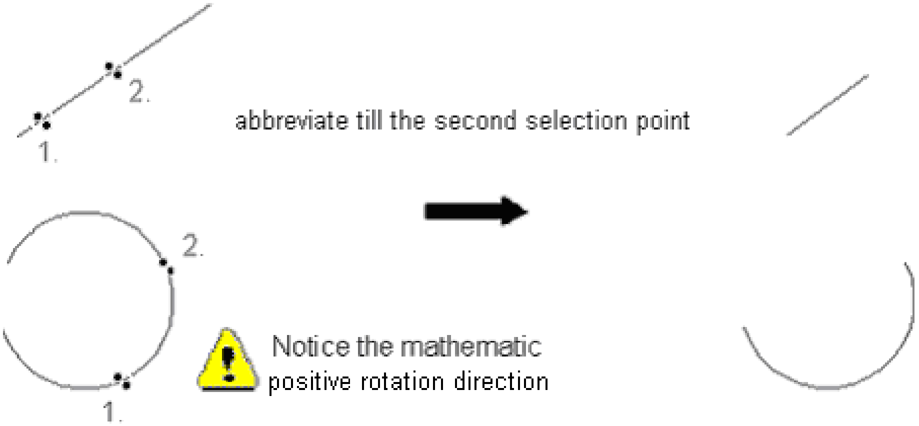
1. Choose function trimx
2. Select the element
3. Select the new element
4. Complete: Right mouse key or POLYEND

Hints:
Basically: Please select the elements on the part, that shall remain unchanged.

Abbreviate elements

Button: 2D + TRIM +

Command: trimx -lm (-l=length, -m=multiple)


2.
1.
abbreviate till the second selection point
2.
1.
Notice the mathematic positive rotation direction

Proceeding:

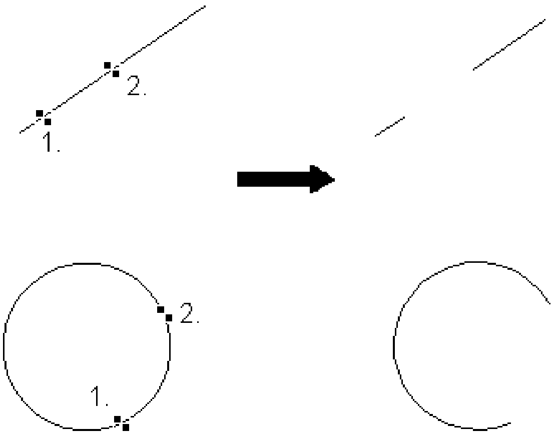
1. Choose function trimx
2. Select the element
3. Select the new end-point
4. Complete: Right mouse key or POLYEND

Hints:
Basically: Please select the elements on the part, that shall remain unchanged.

Cut a part out of an element

Button: 2D + TRIM + ✂

Command: trimx -pm (-p=part, -m=multiple)



Proceeding:

1. Choose function trimx
2. Select the element at 1. intersection
3. Select the element at 2. intersection
4. Complete: Right mouse key or POLYEND

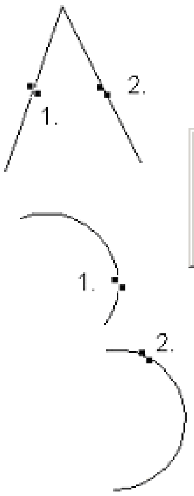
Hints:

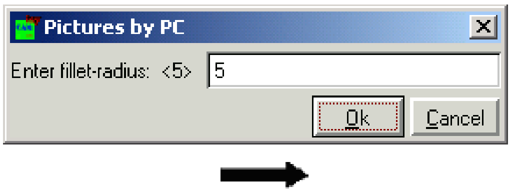
Please pay attention: For circles, cutting works counter clockwise!

Fillets with certain radius

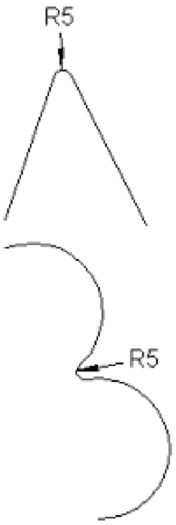
Button: 2D + TRIM + R=?

Command: trimx -fm (-f=fillet, -m=multiple)




Pictures by PC

Enter fillet-radius: <5> 5



Proceeding:

1. Choose function trimx
2. Enter curve radius, e. g. 5
3. Select the element near the fillet
4. Complete: Right mouse key or POLYEND

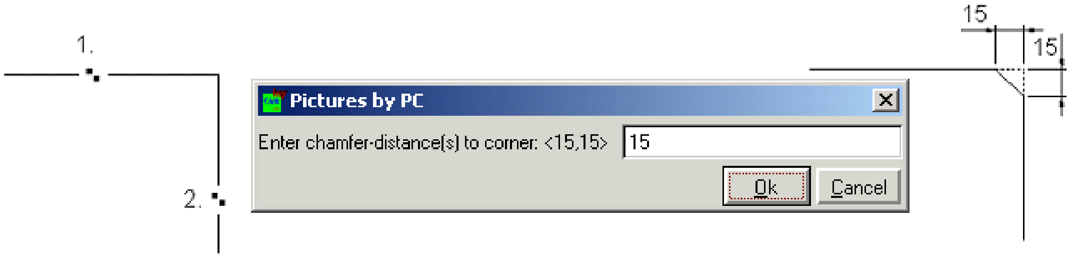
Hints:

Please pay attention: For circles, cutting works counter clockwise!

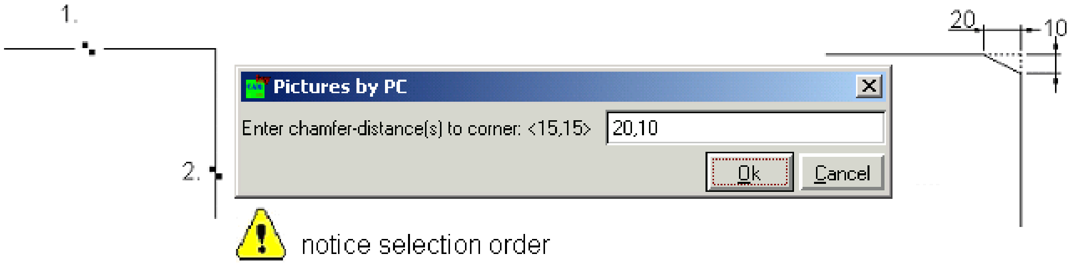
Generate chamfer - by defining distances to corner

Button: 2D + TRIM + D=?

Command: trimx -cm (-c=chamfer, -m=multiple)



Enter chamfer-distance(s) to corner: <15,15> 15



Enter chamfer-distance(s) to corner: <15,15> 20,10

Proceeding:

1. Choose function trimx (button)
2. Enter chamfer distance to corner (dialog box)
3. Select the elements near the chamfer
4. Complete: Right mouse key or POLYEND

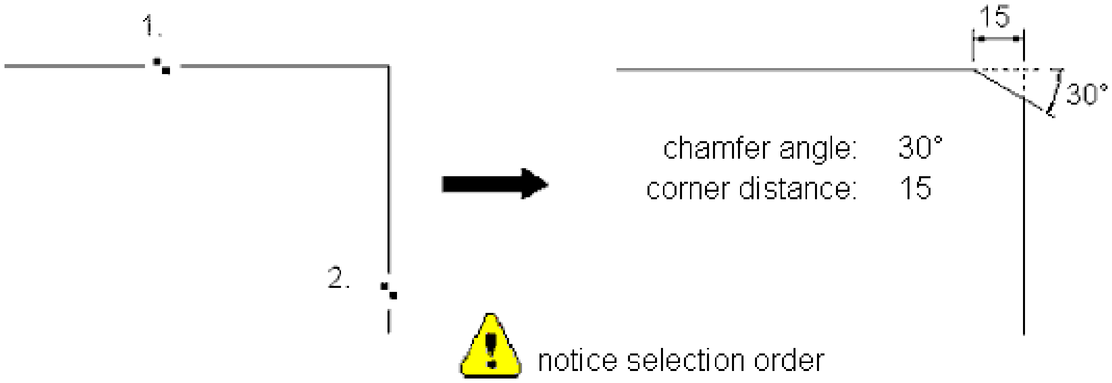
Hints:

Assign the chamfer distances to corner to the variable CHAMFER e.g. with chamfer = 20,10 or via menu.

Generate chamfer - definition of angle and distance

Button: 2D + TRIM + A=?

Command: trimx -cm (-c=chamfer, -m=multiple)
Example: set chamfer=15,$[(15*tand(30))%2]; trimx -cm



Proceeding:

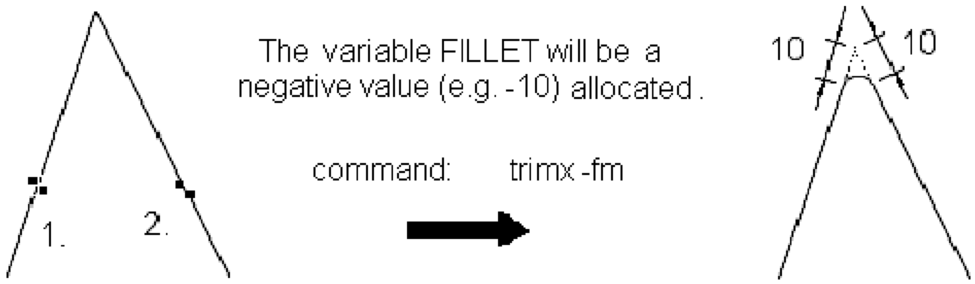
1. Choose function trimx (button)
2. Enter corner distances (dialog box)
3. Select the elements near the chamfer
4. Complete: Right mouse key or POLYEND

Hints:

During the input via menu the corner distance will be calculated and assigned to the variable CHAMFER.

Fillet – Added feature

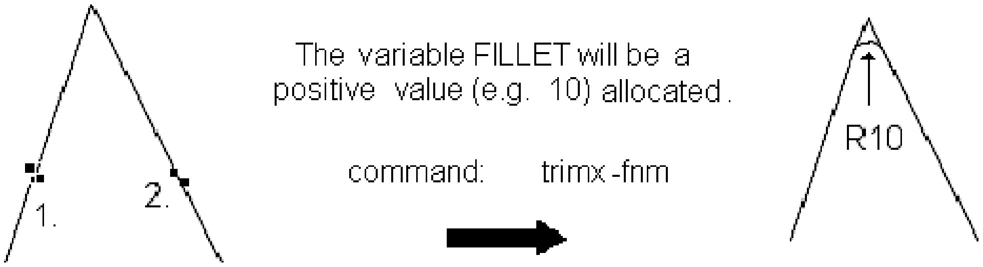
Fillet with defined space to intersection



The variable FILLET will be a negative value (e.g. -10) allocated.

command: trimx -fm

Fillet without cutting



The variable FILLET will be a positive value (e.g. 10) allocated.

command: trimx -fnm

Proceeding:

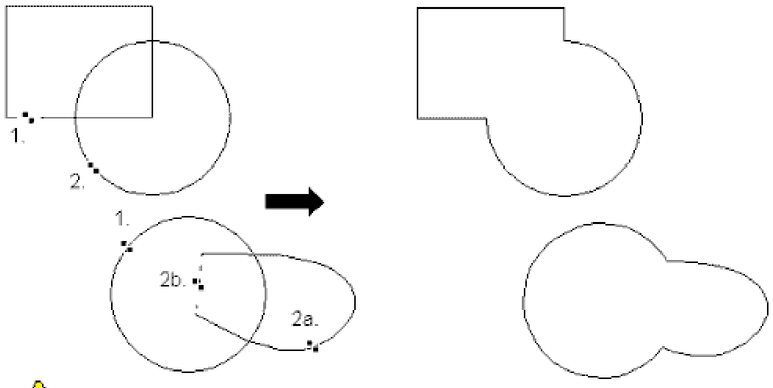
1. Set variable FILLET (command window: fillet=…)
2. Call function trimx (with parameters) about command window
3. Select the elements near the fillet
4. Complete: Right mouse key or POLYEND

Hints:
Assign the rounding radius to the variable FILLET e.g. set fillet = -10

Union of areas (Boolean operation)

Button: 2D + TRIM + 1+2

Command: eq -bu -d?



⚠ On open contours insert help lines if applicable.

Proceeding:

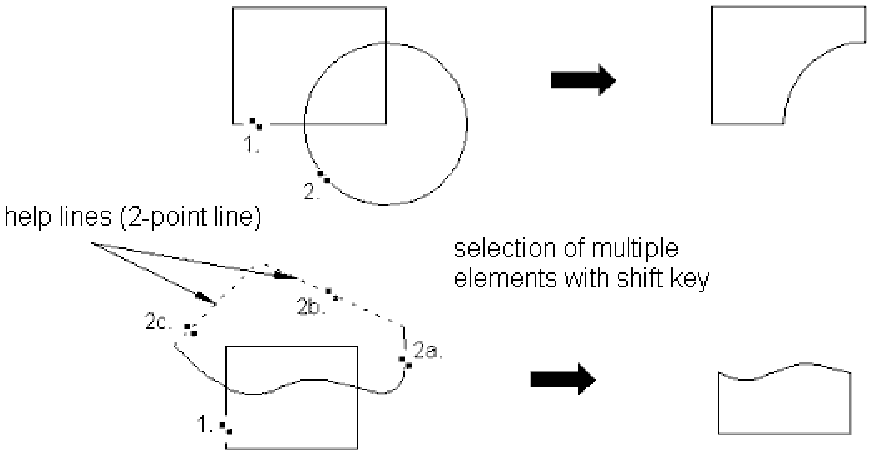
1. Choose function with button
2. Select contours (if objects consist of several elements, select with SHIFT-key, complete with POLYEND)

Hints:
With one operation you can edit only always two closed (!) contours, the result is one closed contour.

Subtraction of areas (Boolean operation)

Button: 2D + TRIM + 1-2

Command: eq -bs -d?



⚠ On open contours insert help lines if applicable.

Proceeding:

1. Choose function with button
2. Select contours (if objects consist of several elements, select with SHIFT-key, complete with POLYEND)

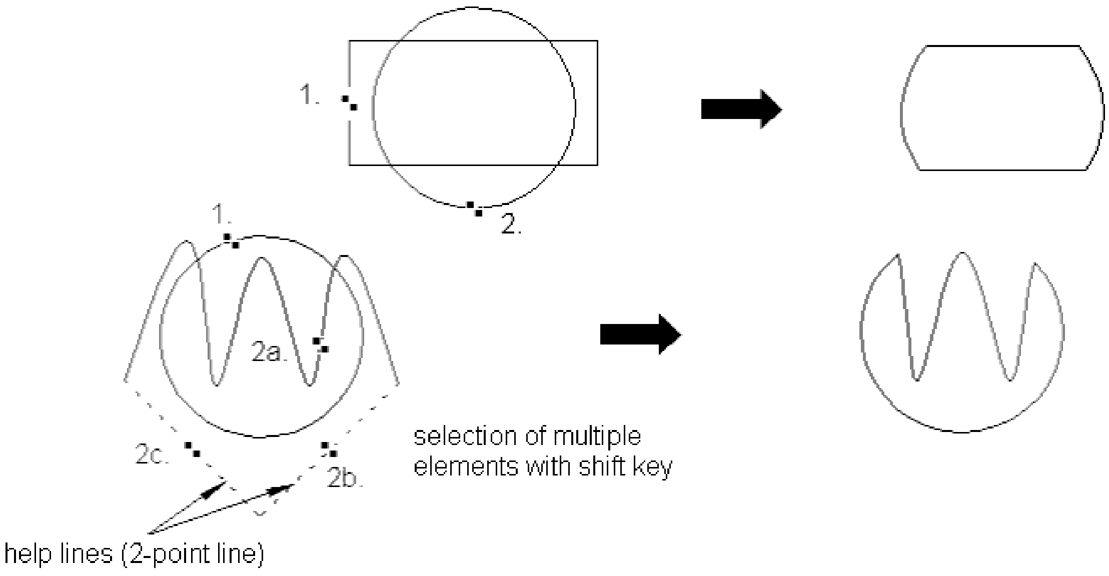
Hints:

With one operation you can edit only always two closed (!) contours, the result is one closed contour.

Intersection of areas (Boolean Operation)

Button: 2D + TRIM + 1⊘2

Command: eq -bi -d?



⚠ On open contours insert help lines if applicable.

Proceeding:

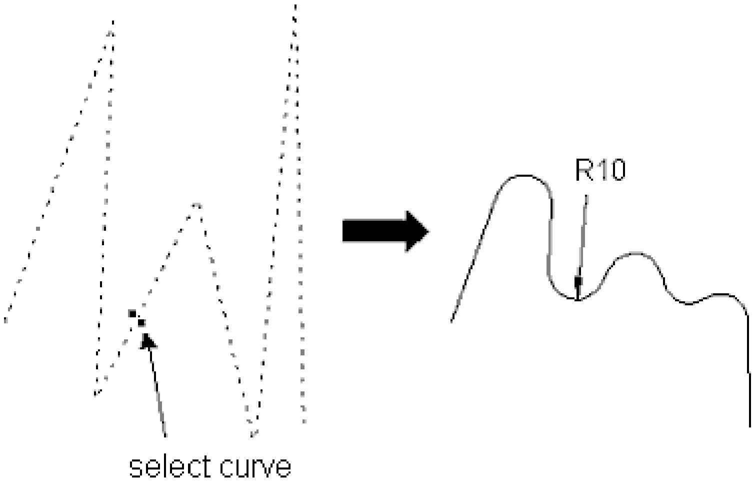
1. Choose function with button
2. Select contours (if objects consist of several elements, select with SHIFT-key, complete with POLYEND)

Hints:

With one operation you can edit only always two closed (!) contours, the result is one closed contour.

Generate contour with automatic fillet of all edges

Command: cfillet -r10 (-r = rounding radius 10)



Proceeding:

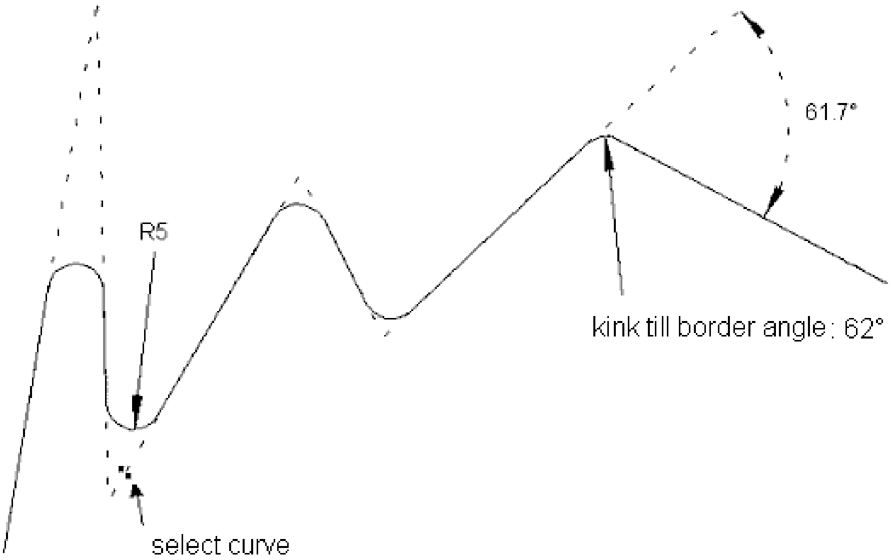
1. Choose command with corresponding options
2. Select curves (several: with SHIFT + POLYEND)

Hints:

The contour is generated as duplicate.

Generate contour with automatic fillet (up to critical angle)

Command: cfillet -r5 -a62 (-r5 = rounding radius 5, -a62 = critical angle 62°)



Proceeding:

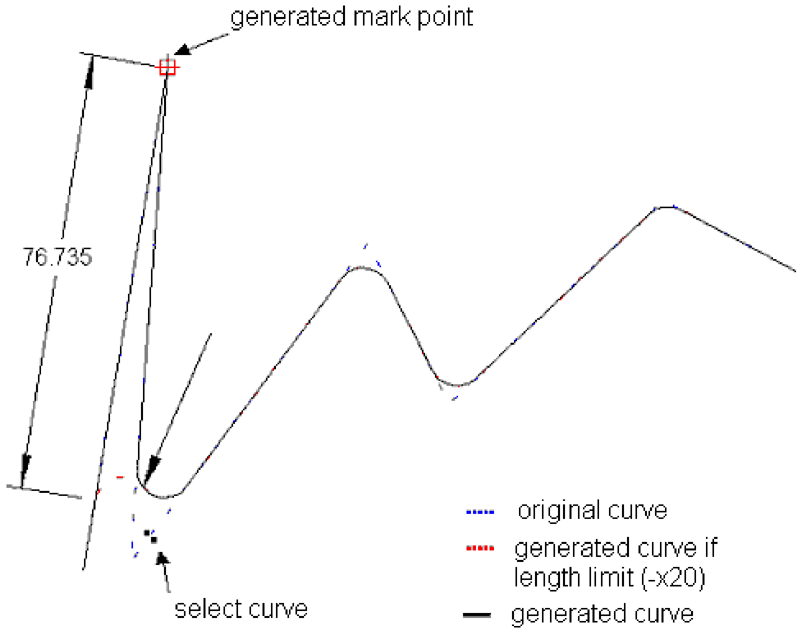
1. Choose command with corresponding options
2. Select curves (several: with SHIFT + POLYEND)

Hints:

The contour is generated as duplicate.

Generate contour with automatic fillet (dependent on length)

Command: cfillet -r5 -x20 -e (-r5 = rounding radius 5,
-x20 = max. length of elements to cut is 20,
-e = mark of edges, that can not be rounded)



Proceeding:

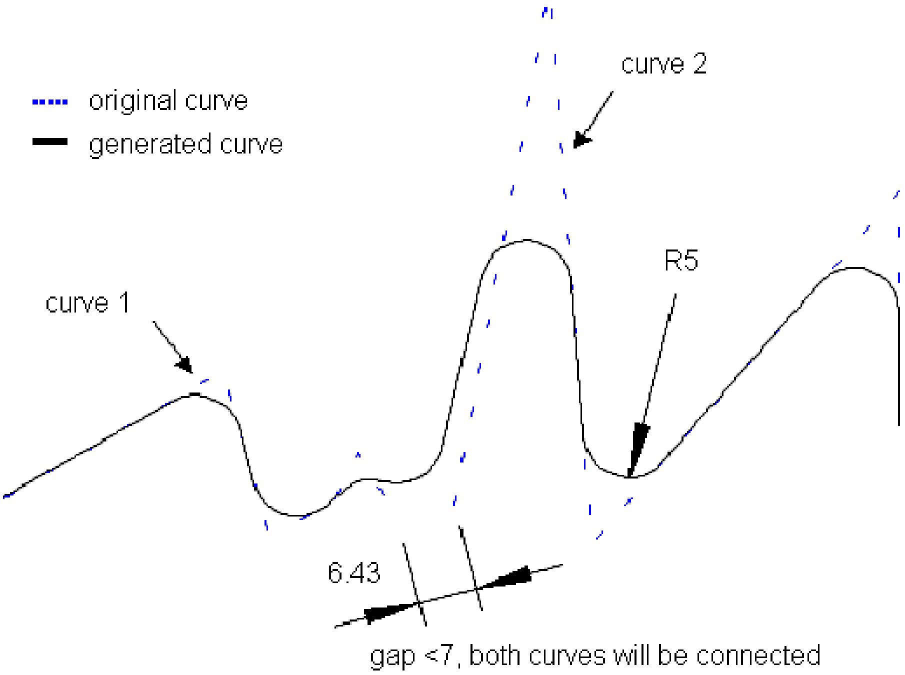
1. Choose command with corresponding options
2. Select curves (several: with SHIFT + POLYEND)

Hints:

The contour is generated as duplicate.

Generate contour with automatic fillet and gap compensation

Command: cfillet -r5 -g7 (-r5 = rounding radius 5,
-g7 = max. contour gap (otherwise new curve!)



Proceeding:

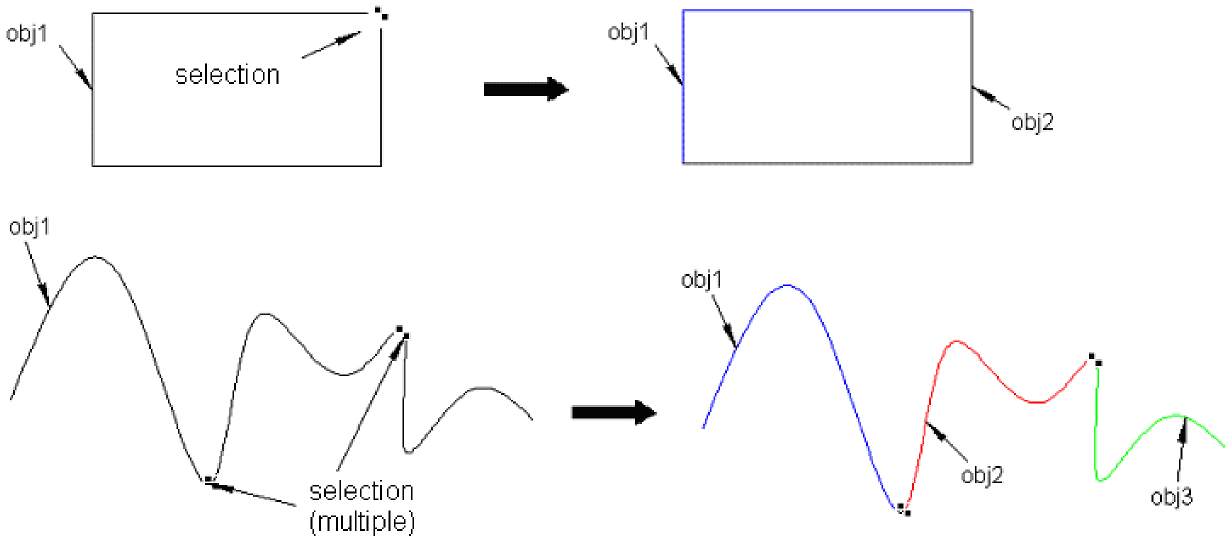
1. Choose command with corresponding options
2. Select curves (several: with SHIFT + POLYEND)

Hints:

The contour is generated as duplicate.

Split contours at split point into two objects

Command: splitobj (-m: multiple, -c #1,#2: alternating colours)



Proceeding:

1. Choose command
2. Select split point(s)

Hints:

Before choosing the command the contour must contain correlative split points.

Introduction Adjust Objects

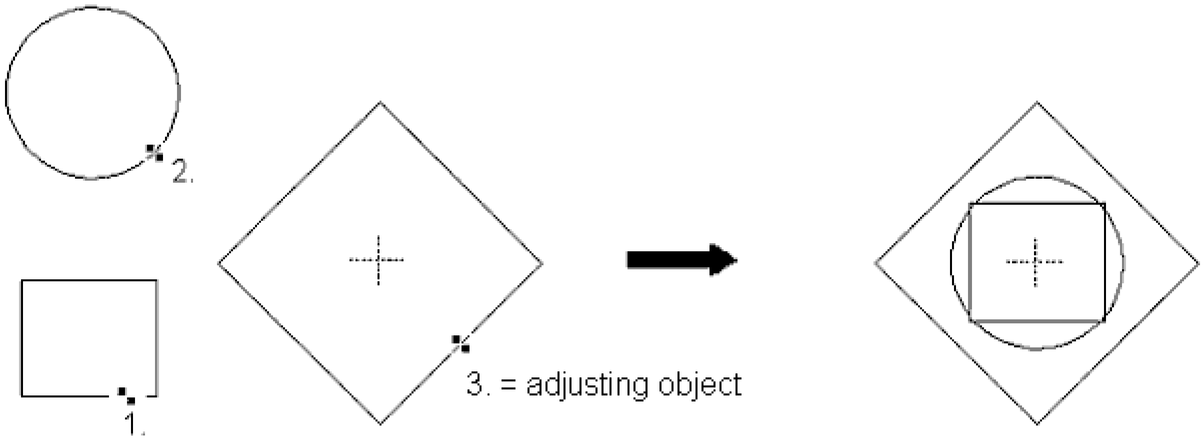
The adjust functions are called with buttons in the tool bar "Tools".

Menu - Button: 2D + ADJUST

Adjust object to centre of a reference object

Button: 2D + ADJUST + ADJ CENT

Command: adj -c -d? (-c = centric)



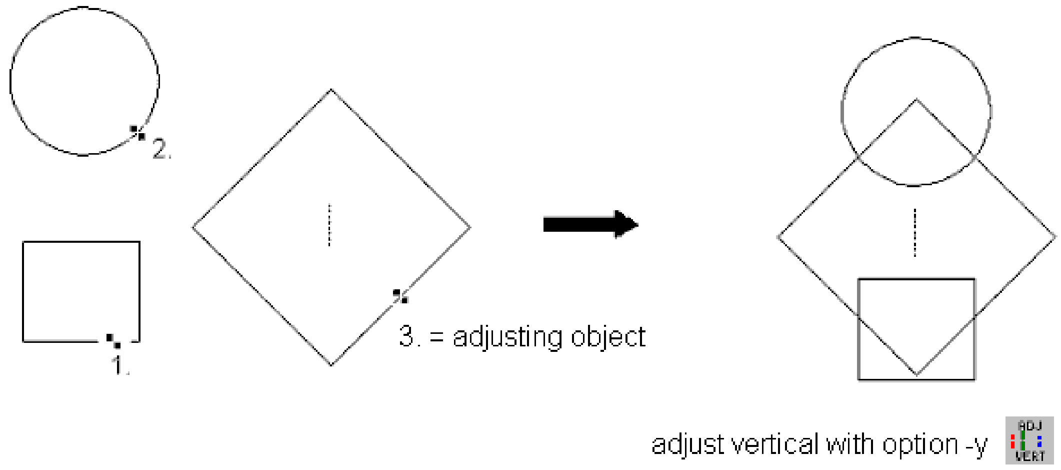
Proceeding:
1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the reference object

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust object horizontally to a reference object

Button: 2D + ADJUST + ADJ HOR

Command: adj -x -d? (-x = horizontal)



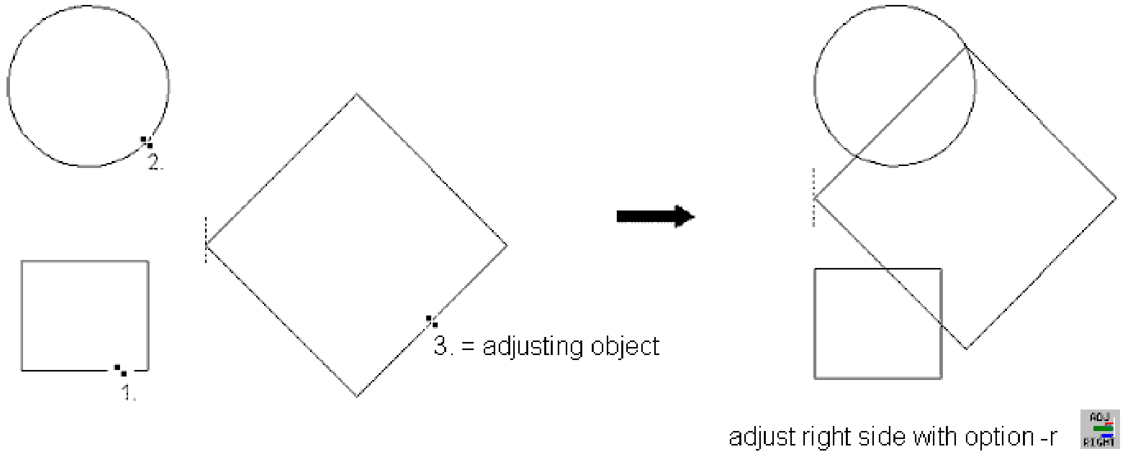
Proceeding:
1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the reference object

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust to the left side of a reference object

Button: 2D + ADJUST + ADJ LEFT

Command: adj -l -d? (-l = left)



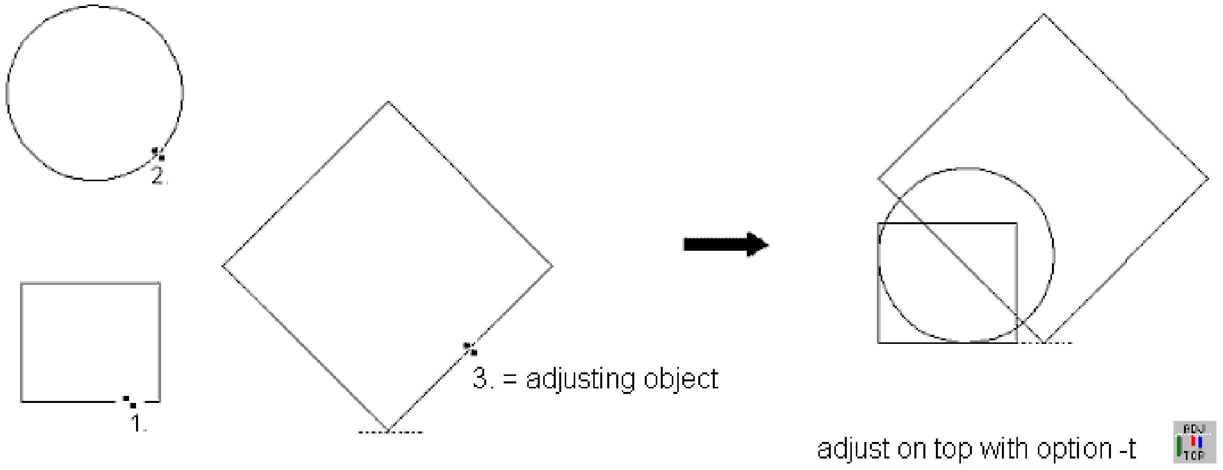
Proceeding:
1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the reference object

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust to the bottom of a reference object

Button: 2D + ADJUST + ADJ BOTT.

Command: adj -b -d? (-b = bottom)



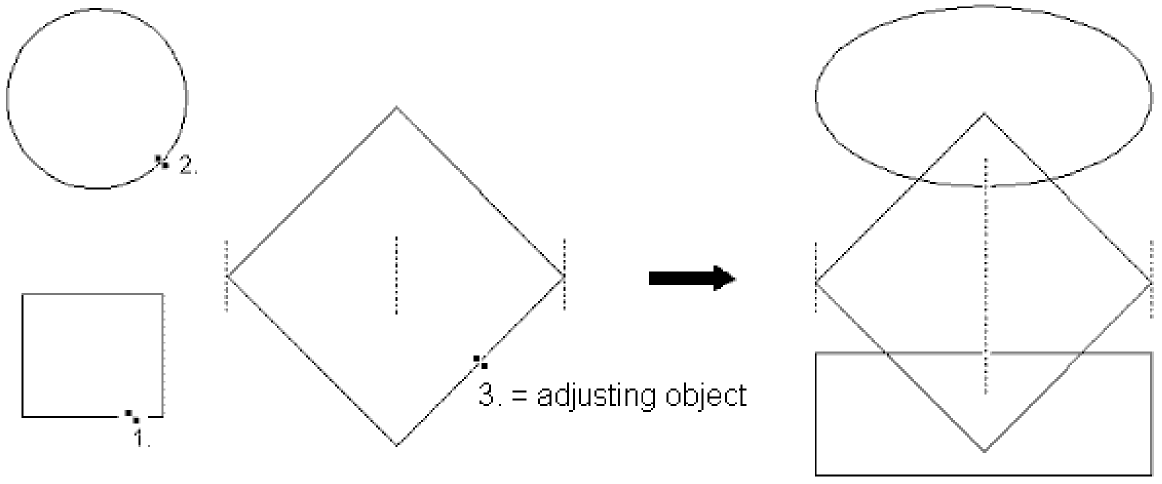
Proceeding:

1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the reference object

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust horizontal with adapted width, height unchanged

Command: adj -x3 -d? (-x3 = horizontal, adapted width)



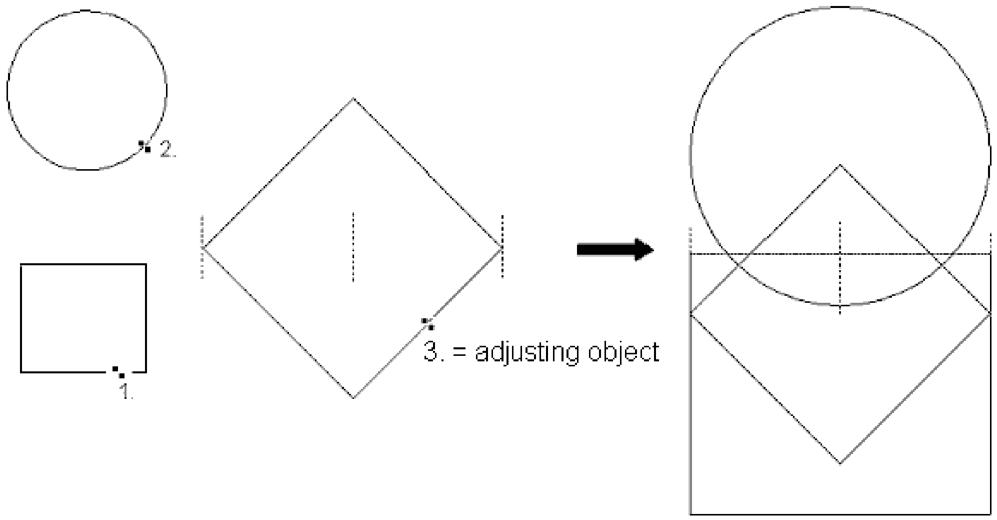
Proceeding:

1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the reference object

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust horizontal with adapted width, aspect ratio unchanged

Command: adj -x4 -d? (-x4 = horizontal, adapted width, aspect ratio unchanged)



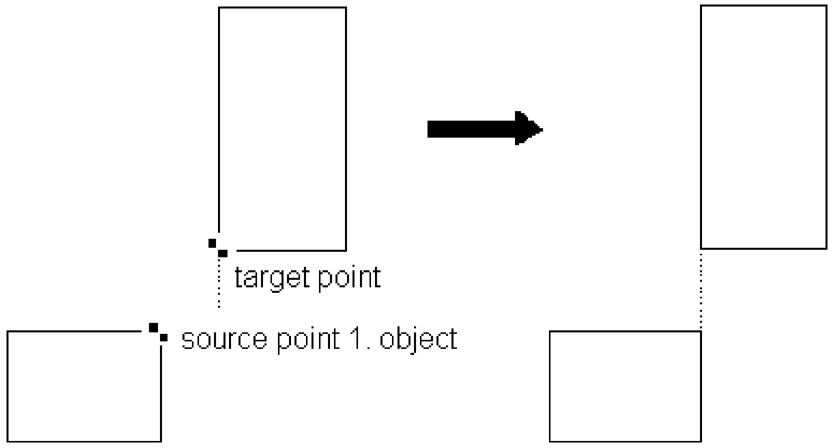
Proceeding:
1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the reference object

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust object(s) horizontal to a point

Button: 2D + ADJUST + ADJ R HOR

Command: adj -x -pq (-x = horizontal, -p = point. 1.obj., -q = point reference object)



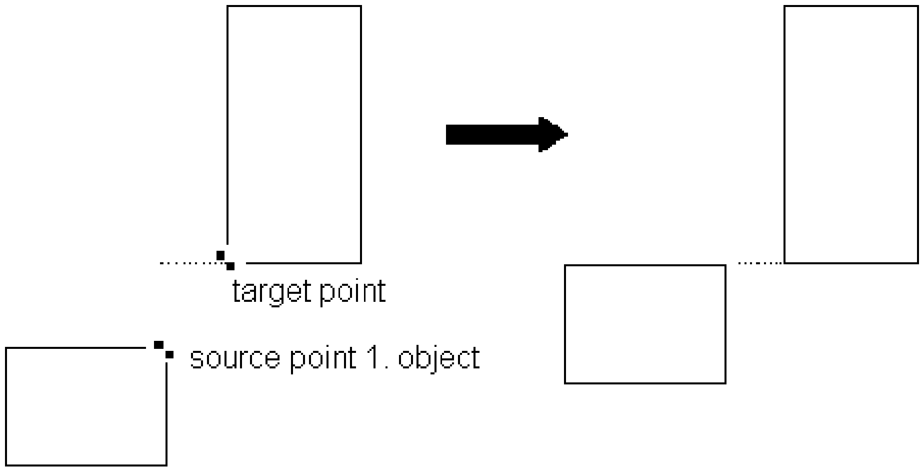
Proceeding:
1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the target point

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust object(s) vertical to a point

Button: 2D + ADJUST + ADJ P. VERT

Command: adj -y -pq (-y = vertical, -p = point. 1.obj., -q = target point reference object)



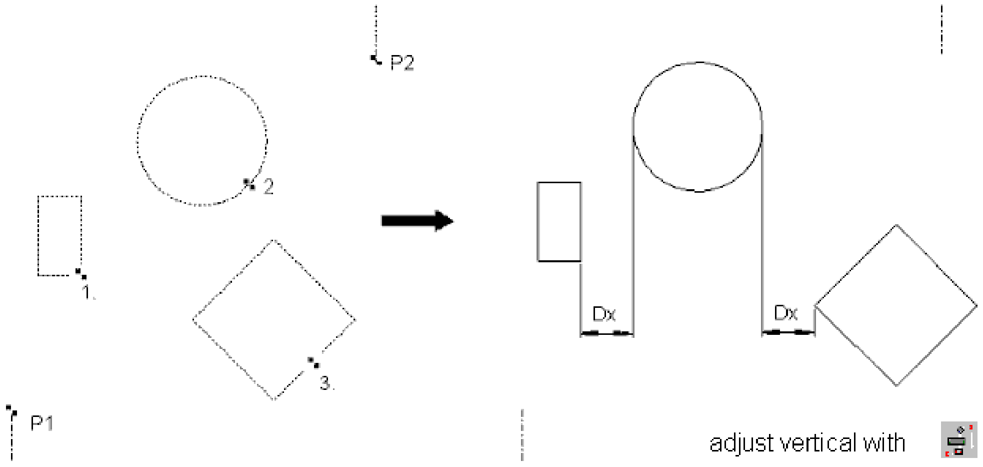
Proceeding:
1. Choose the function with button
2. Select object (several with <SHIFT>-key)
3. Complete the selection with POLYEND
4. Select the target point

Hints:
After selection of an object the appropriate macro object can be selected with key F10.

Adjust horizontal between 2 points without border space

Button: 2D + ADJUST +

Command: gap -h (-h = horizontal)



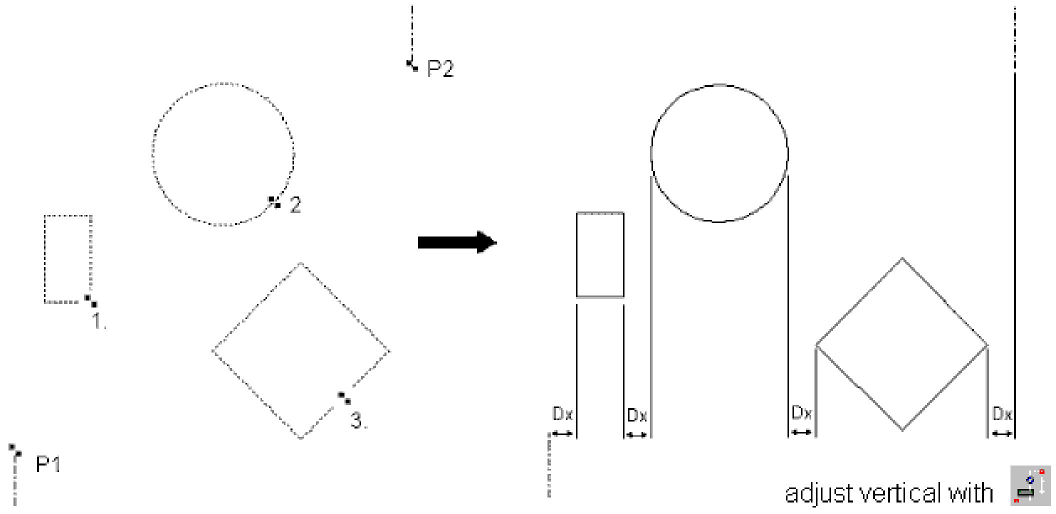
Proceeding:
1. Choose the function with button
2. Select object, complete with POLYEND
3. Select the reference points P1, P2

Hints:
The position of the objects is determined by the order of selection.

Adjust horizontal between 2 points with border space

Button: 2D + ADJUST +

Command: gap -hx (-h = horizontal)



Proceeding:

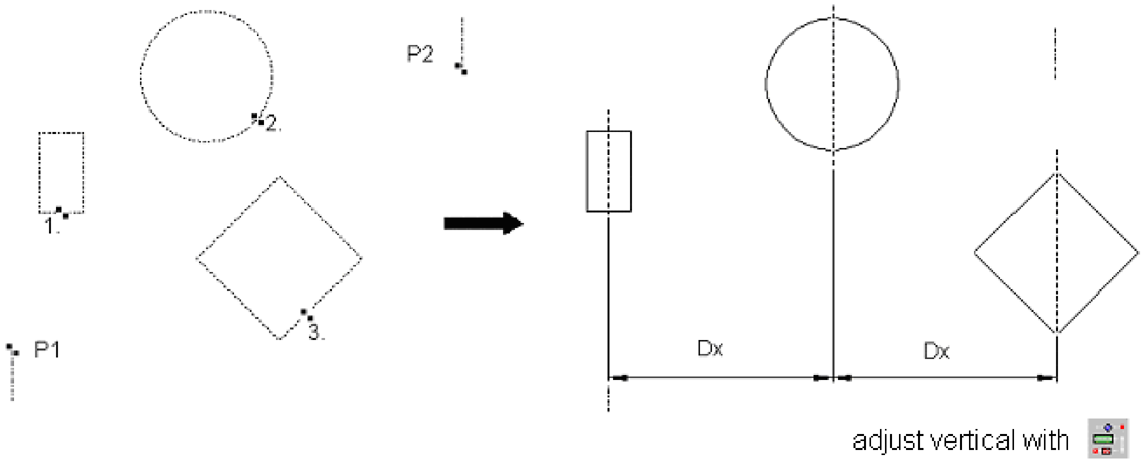
1. Choose the function with button
2. Select object, complete with POLYEND
3. Select the reference points P1, P2

Hints:
The position of the objects is determined by the order of selection.

Adjust horizontal between 2 points, centre, without border space

Button: 2D + ADJUST +

Command: gap -ch (-h = horizontal, -c = centre)



Proceeding:

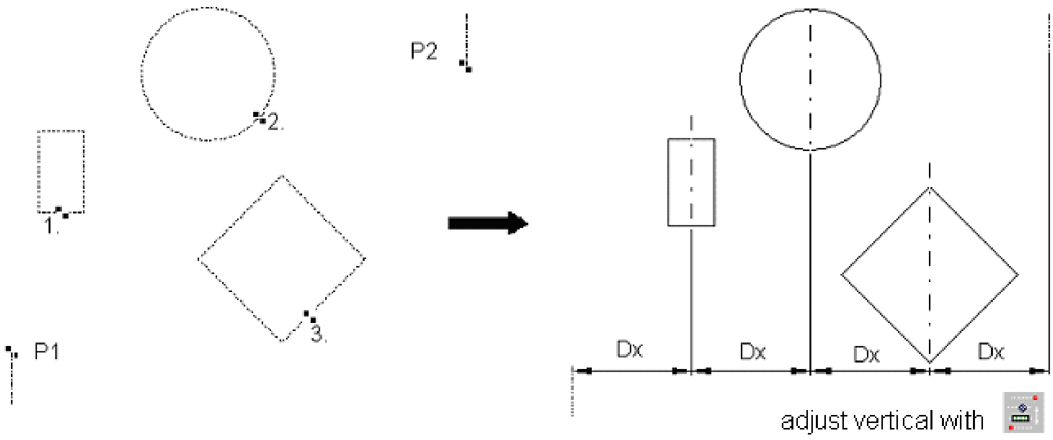
1. Choose the function with button
2. Select object, complete with POLYEND
3. Select the reference points P1, P2

Hints:
The position of the objects is determined by the order of selection.

Adjust horizontal between 2 points, centre with border space

Button: 2D + ADJUST +

Command: gap -chx (-h = horizontal, -c = centre)



Proceeding:

1. Choose the function with button
2. Select object, complete with POLYEND
3. Select the reference points P1, P2

Hints:
The position of the objects is determined by the order of selection.

Introduction Object – Array

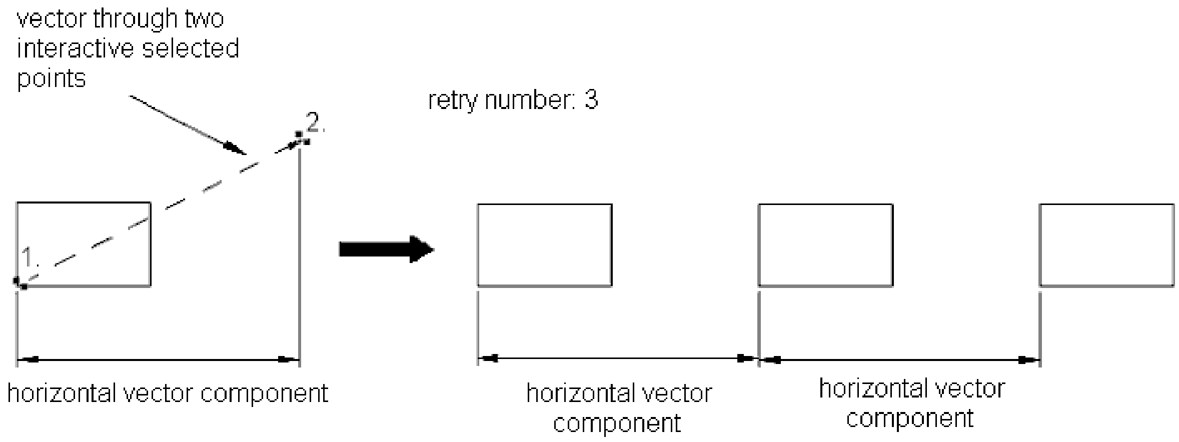
The functions of the menu “Array“ are used to array objects for direction and repetition.

Menu - Button: 2D + ARRAY

Horizontal array (translation vector by two points)

Button: 2D + ARRAY + HORIZ

Command: array -h


vector through two interactive selected points
retry number: 3
2.
1.
horizontal vector component
horizontal vector component
horizontal vector component

Proceeding:

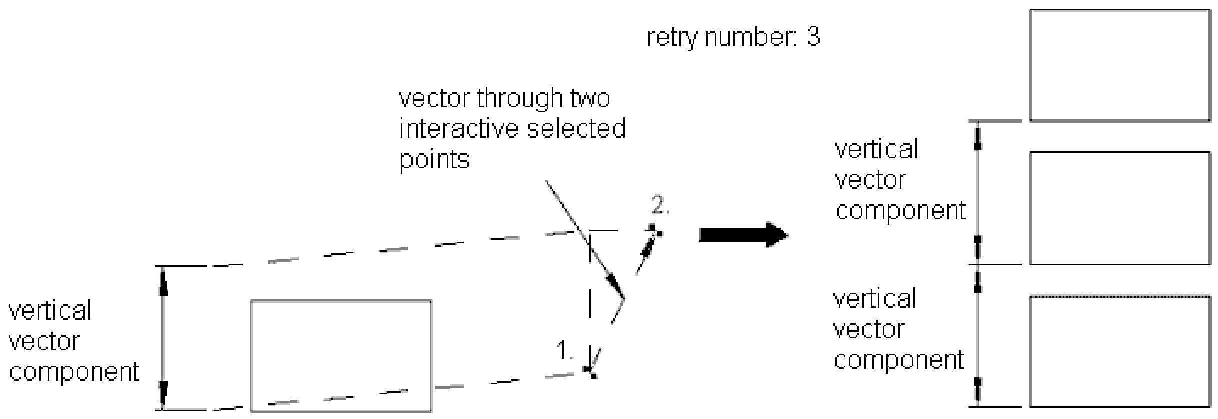
1. Choose function and select the home object
2. Enter the total number (repeat number): here 3
3. Select the source point(1.point of vector)
4. Select the target point (2.point of vector)

Hints:
At horizontal array, the horizontal part of the translation vector is considered exclusive!

Vertical array (translation vector by two points)

Button: 2D + ARRAY + VERT.

Command: array –v


retry number: 3
vector through two interactive selected points
vertical vector component
2.
1.
vertical vector component
vertical vector component

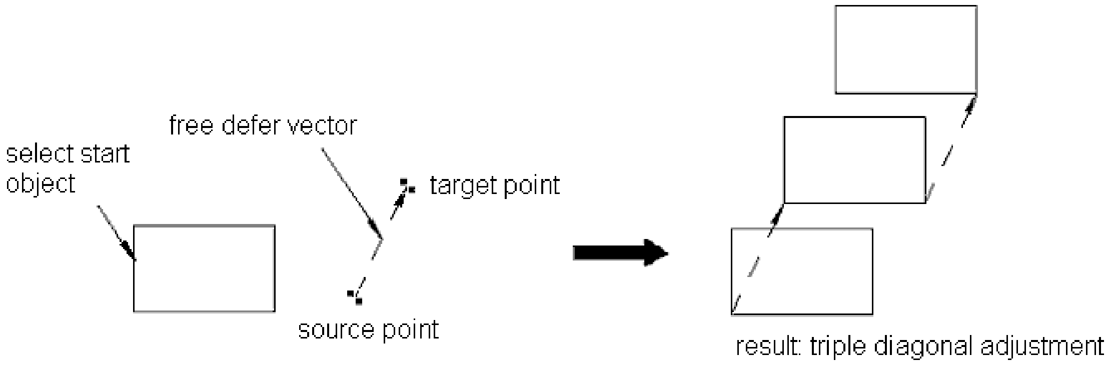
Proceeding:
1. Choose function and select the home object
2. Enter the total number (repeat number): here 3
3. Select the source point(1.point of vector)
4. Select the target point (2.point of vector)

Hints:
At vertical array, the vertical part of the translation vector is considered exclusive!

Diagonal array (translation vector by two points)

Button: 2D + ARRAY + DIAG.

Command: array



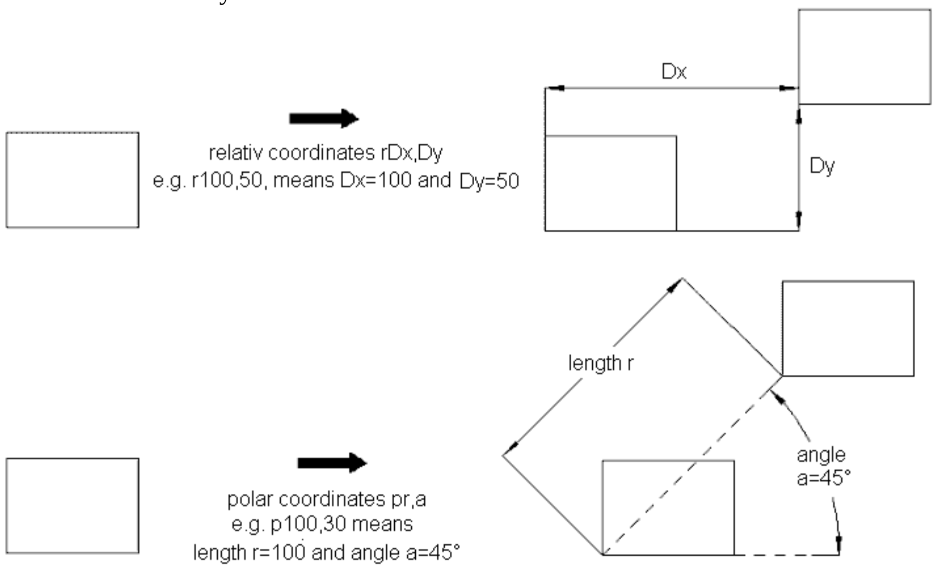
Proceeding:
1. Choose function and select the home object
2. Enter the total number (repeat number): here 3
3. Select the source point(1.point of vector)
4. Select the target point (2.point of vector)

Hints:
The repeat number is the total number including the home object, the 3. vector can be set with grid.

Numerical array (relative or polar)

Button: 2D + ARRAY + DIAG.

Command: array "" "" "" ""?


Dx
relativ coordinates rDx,Dy
e.g. r100,50, means Dx=100 and Dy=50
Dy
length r
angle
a=45°
polar coordinates pr,a
e.g. p100,30 means
length r=100 and angle a=45°

Proceeding:

1. Choose function
2. Select the home object
3. Enter the total number: here 2
4. Enter the translation vector numerically

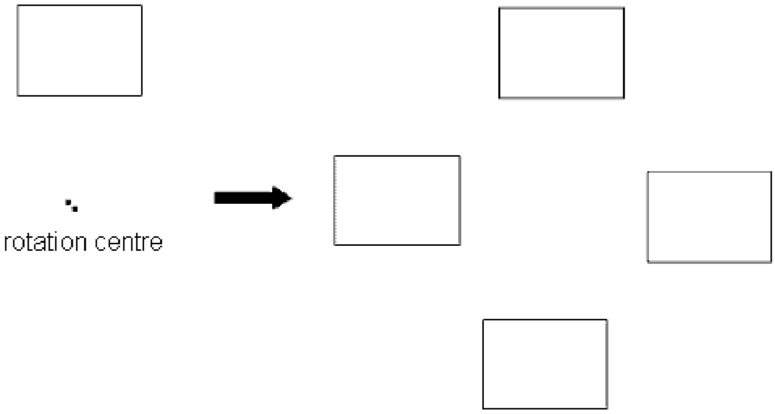
Hints:

Enter the translation vector relative (dx=horizontal, dy=vertical): -> rdx,dy polar (r=length, a=angle): ->pr,a without brackets

2D-rotatoric array, axial (across round angle)

Button: 2D + ARRAY +

Command: rotarr



Proceeding:

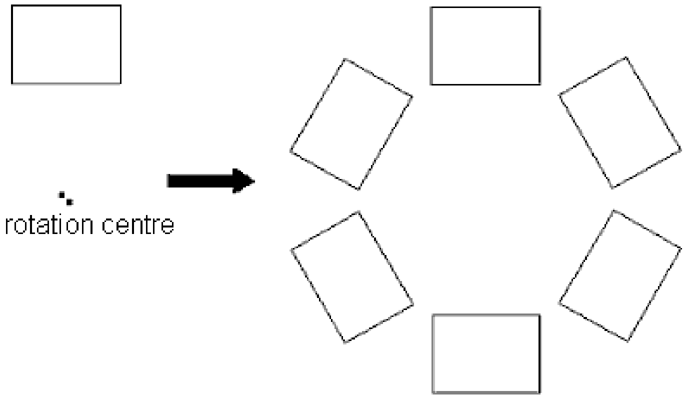
1. Choose function (button)
2. Select the object
3. Enter the total number: here 4
4. Select the rotation centre

Hints:
The function is also available with defined angle area and numerical input of rotation centre.

2D-rotatoric array, radial (across round angle)

Button: 2D + ARRAY +

Command: rotarr -r



Proceeding:

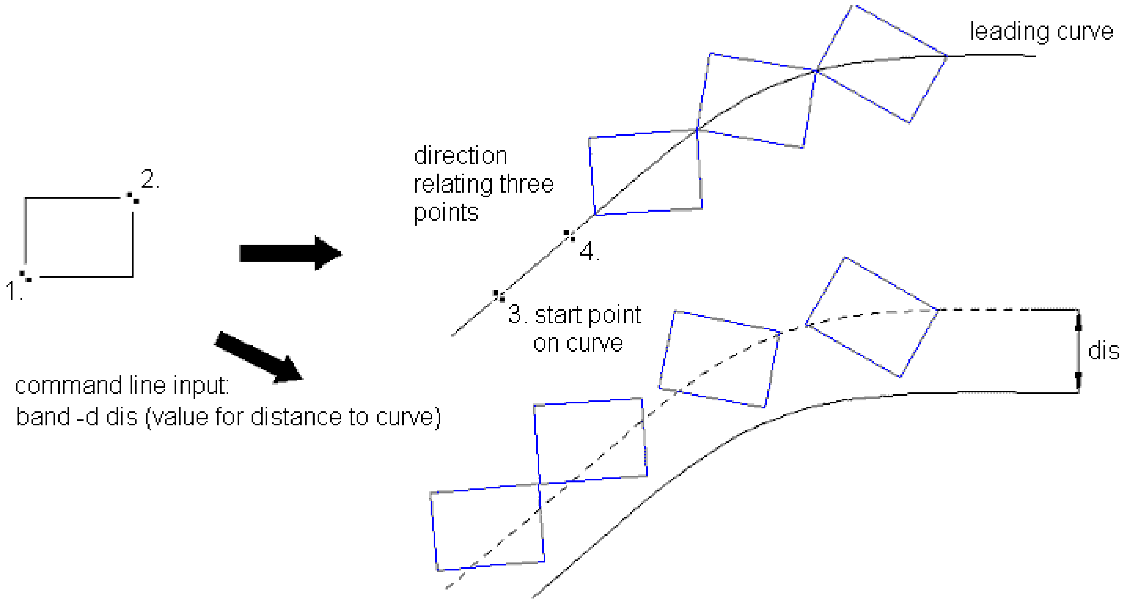
1. Choose function (button)
2. Select the object
3. Enter the total number: here 6
4. Select the rotation centre

Hints:
The function is also available with defined angle area and numerical input of rotation centre.

Array along a poly curve

Button: 2D + ARRAY +

Command: band



Proceeding:

1. Choose the function (with button) or enter command: band
2. Select the object points (1. and 2.)
3. Select the curve (3.) and direction (4.)
4. At option “band -d dis“, extra choice of the side

Hints:

Selecting the curve, the start point will be determined at the same time. An object overlapping can arise at a strong bend.

Introduction point reduction at polygons

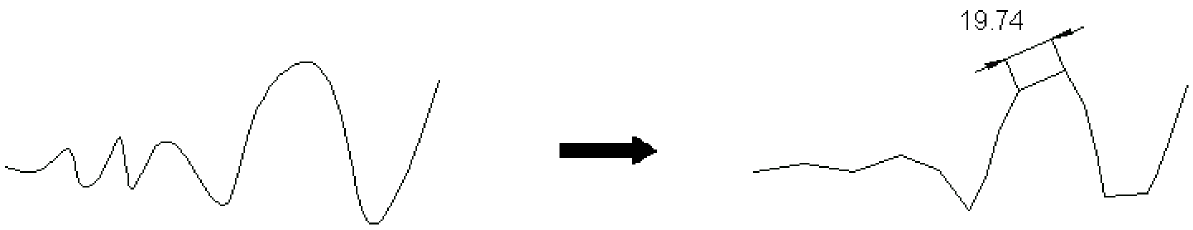
The menu “Converting“ on the menu “General Tools“ has functions for the conversion of polygons. Choose the following buttons to open this menu.

Button: + CON-VERT

Point reduction by maximum point distance

Button: + CON-VERT + REDUC

Command: pred -l 20 (-l 20 = The maximum point distance is 20)



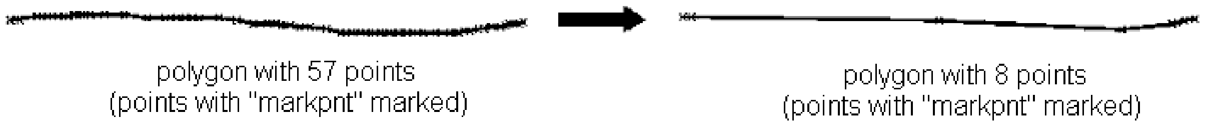
Proceeding:
1. Choose command (with button)
2. Enter maximum point distance: 20
3. Select polygon

Hints:
Points whose distance is smaller than indicated are removed. => "Anti-Smoothing"

Point reduction by input limit angle

Button: + CON-VERT + REDUC

Command: pred -a 30 (-a 30 = maximum angle difference of two polygon elements)


polygon with 57 points
(points with "markpnt" marked)
polygon with 8 points
(points with "markpnt" marked)

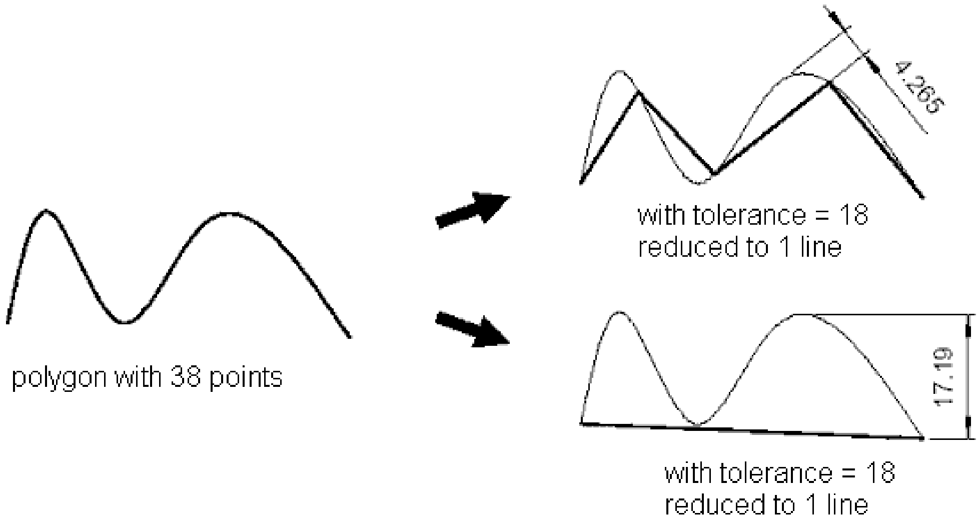
Proceeding:
1. Choose command (with button)
2. Enter maximum angle difference: 30
3. Select polygon

Hints:
Elements whose angle difference is smaller than given are summarized if possible. => "Anti-Smoothing"

Polygon point reduction with maximum distance

Button: + CON-VERT +

Command: redpnts -d 5 (-d 5 = tolerance is 5)



Proceeding:

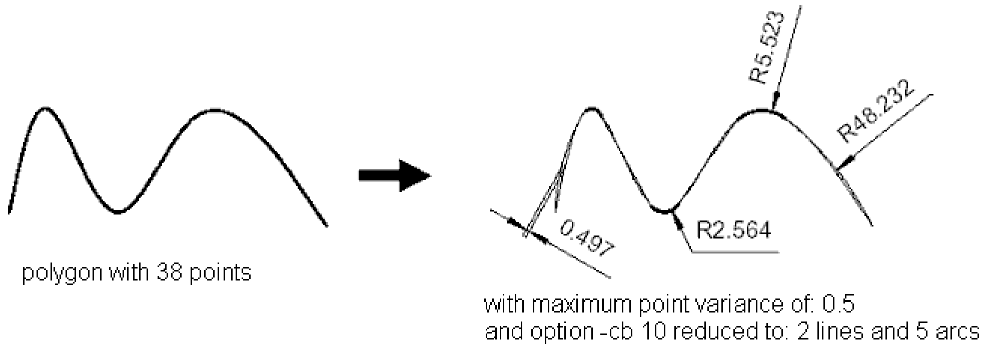
1. Choose command (with button)
2. Enter tolerance0
3. Select polygon (line)

Hints:
A duplicate is produced with a reduced point number. The permitted deviation corresponds to the entered tolerance.

Polygon-point reduction with circle approximation

Button: + CON-VERT +

Command: redpnts -d 0.5 -cb 10 (-d 0.5 = tolerance is 0.5; -c = circle arc; -b 10 = 10 % of radius at line centre)


R5.523
R48.232
0.497
R2.564
polygon with 38 points
with maximum point variance of: 0.5
and option -cb 10 reduced to: 2 lines and 5 arcs

Proceeding:

1. Choose command (with button)
2. Enter maximum point difference. 0.5
3. Select polygon (line)

Hints:
A duplicate is produced with a reduced point number. The permitted deviation corresponds to the entered tolerance.

Introduction Standardization

IsyCAM 2.5 (light) enables the drawing of standard parts for instance after DIN 933. Choose the following buttons to use the functions "Standardization".

Button: 2D + NORM

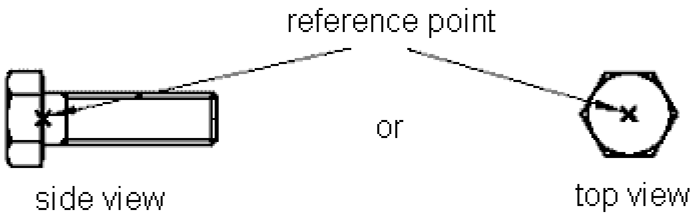
Create standard parts (e.g. DIN 933)

Button: 2D + NORM + 933

Command: din933

Example:

1. (s)ide view or (t)op view: <s>
2. Screw: M12
3. Select reference point
4. Enter length: 40
5. Rotation is counter-clockwise. Please enter degree: <0>



Proceeding:

1. Choose function
2. Enter parameter on key board
3. Place the reference point
4. If necessary, turn or move

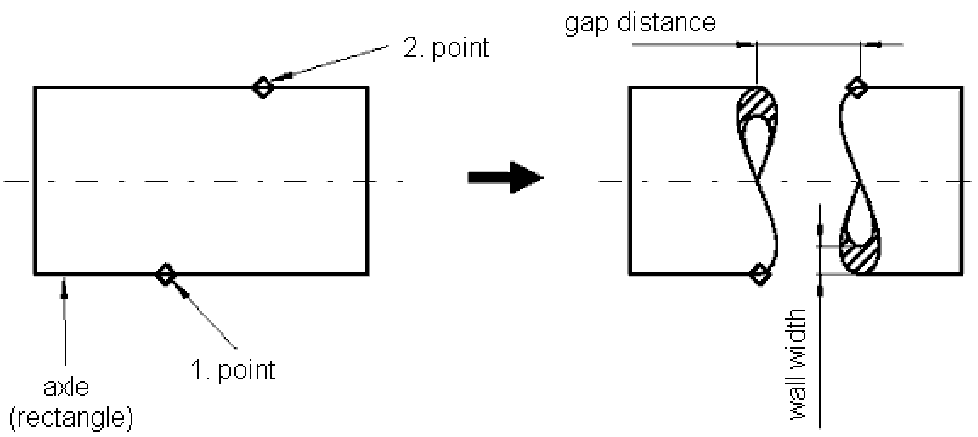
Hints:
If necessary, fit in of objects with: "fitin".

Generate break edge: e.g. hollow shaft

Button: 2D + NORM + HOLE

Command:

break -sh <dis> (-h hollow shaft, <dis> = distance)
break -sx (jagged break)
break -s (wavy break)



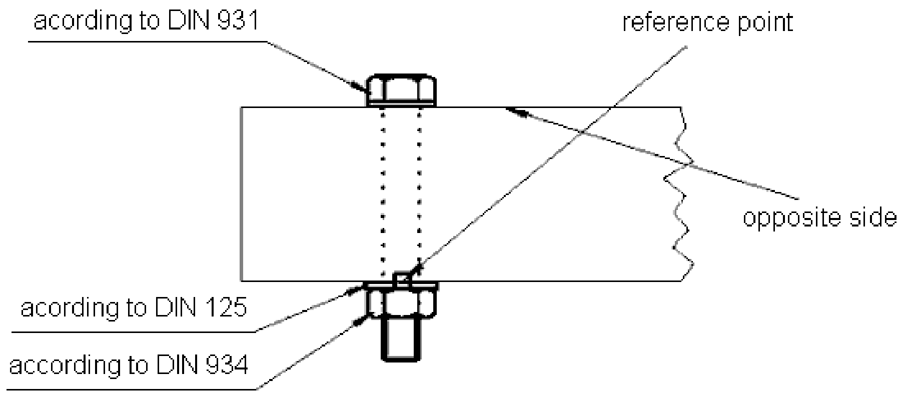
Proceeding:
1. Call function
2. Enter wall width
3. Select a point on the axis (gap start)
4. The second point (opposite) determines gap distance

Hints:
Other breaks can also brought to bear.
(jagged break and wavy break)

Screw, nut and washer

Button: 2D + NORM +

Command: screw



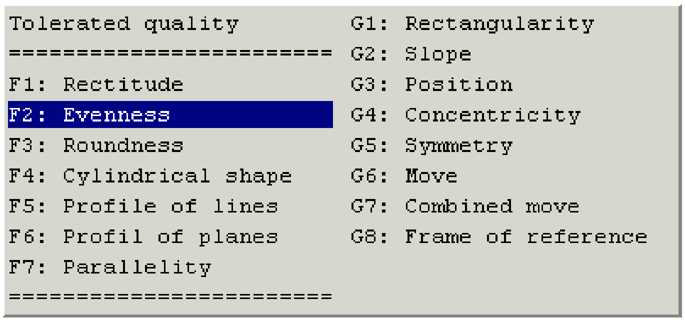
Proceeding:
1. Call function
2. Parameter dialog: Screw, washer, nut
3. Place the reference point for the variant
4. Next variant or finish procedure with “a“

Hints:
The screw is only generated if the dimensions correspond to the norm.

Form- and position tolerances (alike DIN 1101)

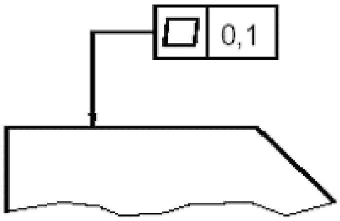
Button: 2D + NORM + 1101

Command: din1101

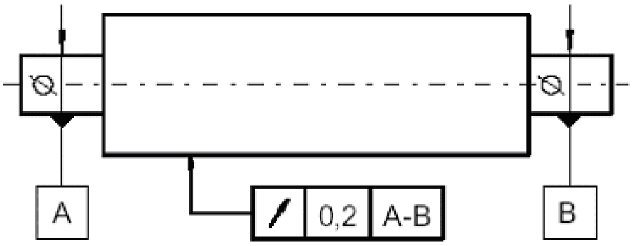


```
Tolerated quality           G1: Rectangularity
========================    G2: Slope
F1: Rectitude               G3: Position
F2: Evenness                G4: Concentricity
F3: Roundness               G5: Symmetry
F4: Cylindrical shape       G6: Move
F5: Profile of lines        G7: Combined move
F6: Profil of planes        G8: Frame of reference
F7: Parallelity
========================
```

example evenness



example roundness



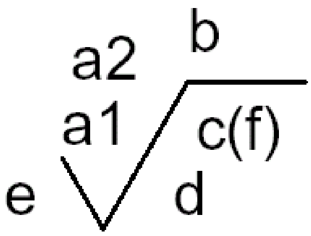
Proceeding:
1. Choose function with button or enter command
2. Select the tolerated attributes
3. Enter tolerance/possibly reference letters
4. Position tolerance border / select reference arrow

Hints:
If no reference letters have to be entered, the question about the 3rd boxes has to be completed with <enter>.

Symbol for surface quality (alike DIN1302)

Button: 2D + NORM + 1302

Command: din1302



Position of the single instruction at symbol:

a roughness value Ra in micrometer or roughness class N
b manufacturing process, surface treatment or layer
c reference line
d grove direction
e processing encore in mm
f other roughness measurement, e.g. Rz

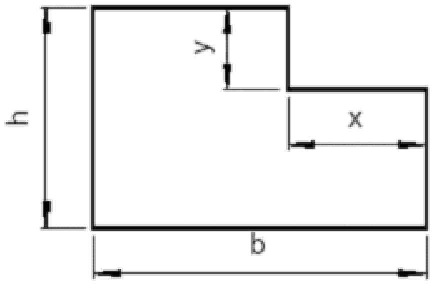
Proceeding:
1. Choose the function (with button) or enter command
2. Enter the values (skip with <ENTER>)
3. Select the position of the symbol (bottom peak, snap on object!)

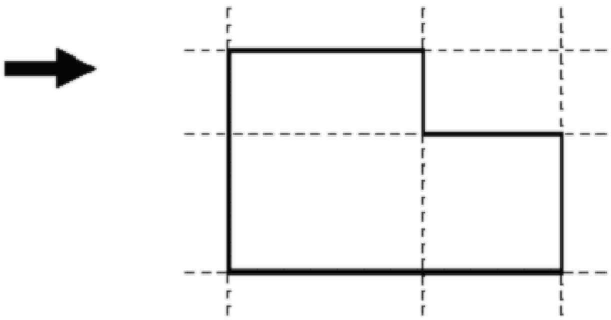
Layout technology: Define variant

Part 1 Definition of parameters

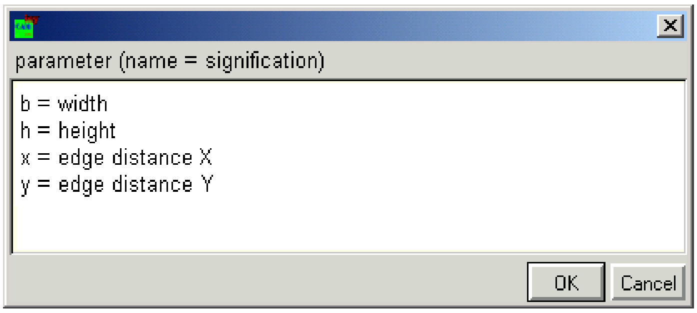
Button: 2D + NORM + MORE NORM + VARI

Command: mkv *





The geometry object must be chained as macro.



parameter (name = signification)

b = width
h = height
x = edge distance X
y = edge distance Y

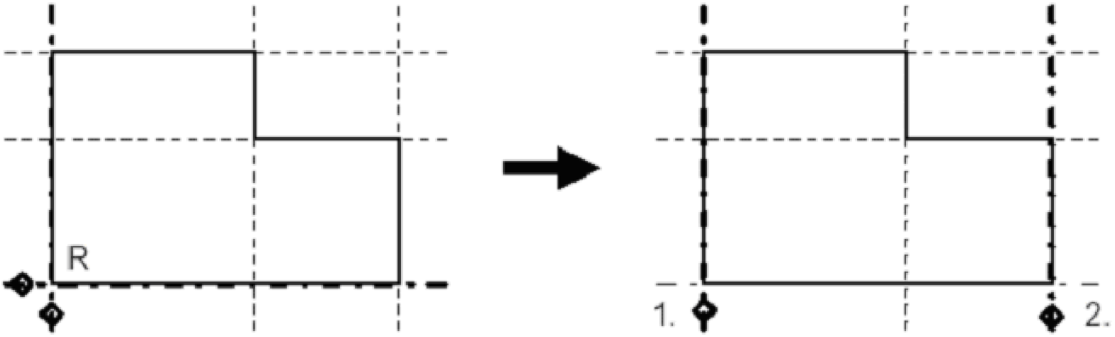
Proceeding:

1. Select macro
2. Choose the function (with button) or enter command
3. Determine name/meaning of distances

Hints:

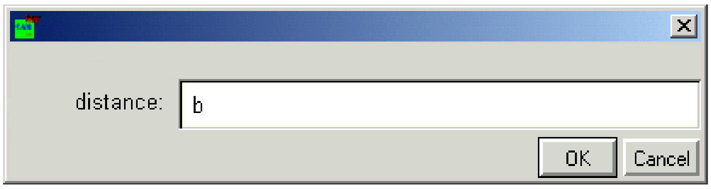
The defined "Name" is the short name, "Meaning" is the long version for subsequent generation of dialogs.

Part 2 Determine reference point and assign the parameter to the distances



determine reference point R
(cut point of the lines)

selection of the reference
lines (= distance "b")


distance: b

Proceeding:

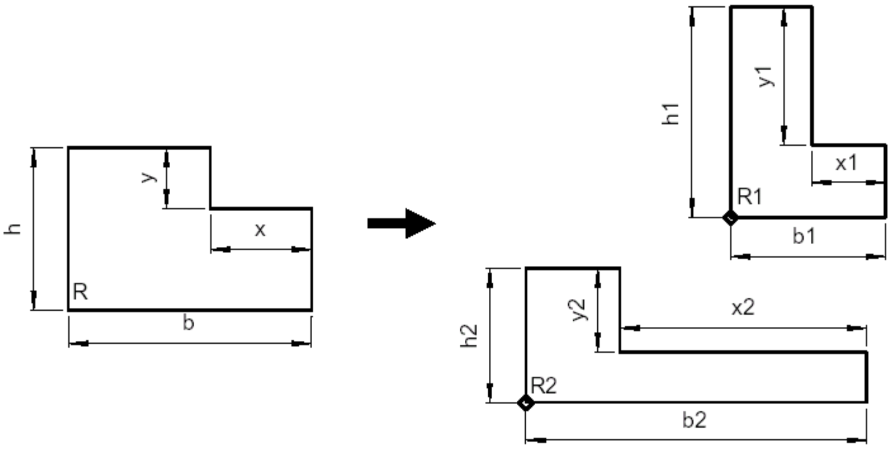
1. Fix reference point
2. Select the reference lines of a distance
3. Enter the name of the distance
4. Finish function after assignment of all parameter

Hints:
Changing the line type of correct selected lines into a dash dotted line, at selection of the distances please select the dash dotted line first.

Layout technology: Define new variant

Button: 2D + NORM + MORE NORM + OB VARI

Command: getv *



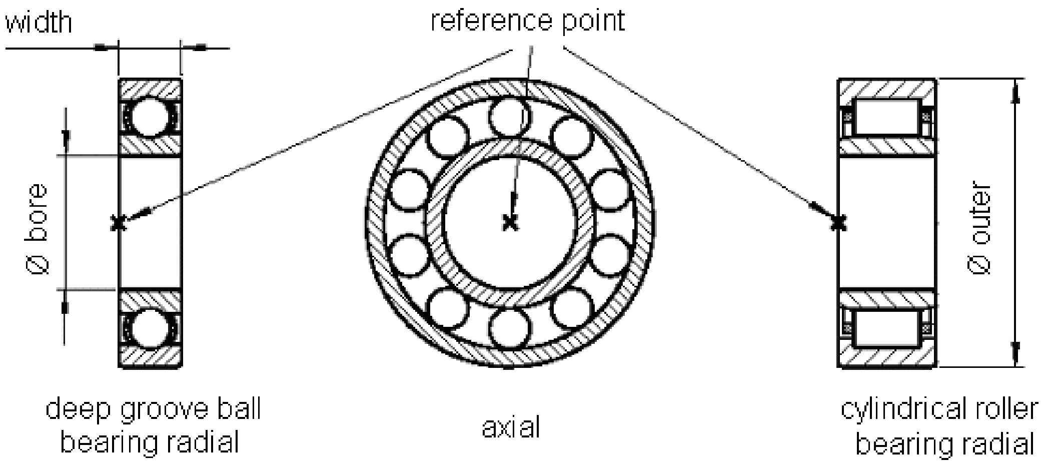
Proceeding:
1. Select macro with sample variant
2. Choose the function (with button) or enter command
3. Select the reference point
4. Enter numerical value for the parameter

Hints:
The parameter variant is created with the command "mkv".

Roller bearing as grooved ball bearing resp. cylinder roller bearing

Button: 2D + NORM +

Command: wlager

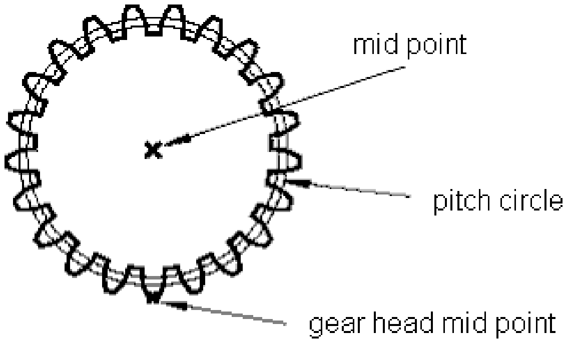

width
reference point
Ø bore
Ø outer
deep groove ball bearing radial
axial
cylindrical roller bearing radial

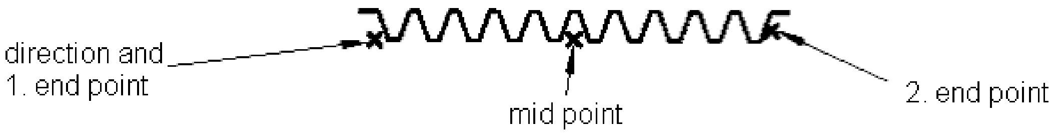
Proceeding:
1. Call function
2. Specify the bearing in the parameter dialog
3. Select reference point in drawing
4. Next bearing or finish procedure with "0"

Hints:
At scaling the hatch, the distance remains constant, if the variable SCALEHS=OFF !

Create cogwheels, cograils with procedure

Command: cogwheel
cograil





Proceeding:

1. Call function
2. Parameter dialog: Module, number of teeth, etc.
3. Place variant at centre point
4. Set tooth pitch centre resp. direction

Hints:
The pitch circles can be deleted as objects.

2D-Measuring

Overview Measuring

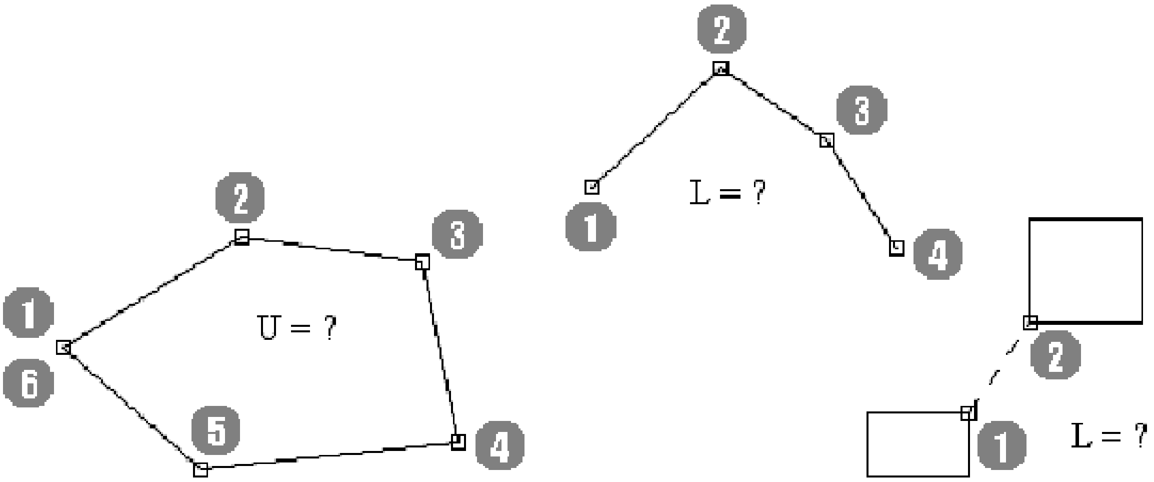
The menus for measuring in **isyCAD/CAM 2.5 (light)** can be called with the buttons of the menu bar "Tools".

Button: + MEA SUR

Length between selected points

Button: + MEA SUR + LEN.

Command: **length**



Point selection in one circulation sense during circumference measurement!

Proceeding:

1. Choose the function (with button) or enter command
2. Select the 1. point and the following points
3. Finish with POLYEND

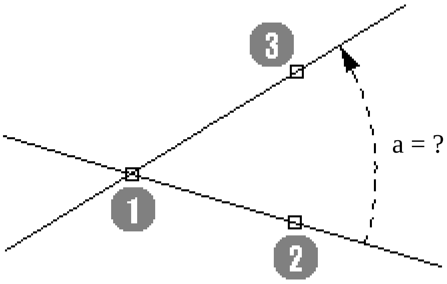
Hints:

Number of decimal digits from variable ROUND, display of total length and distance first/last point

Angle between two lines

Button: + MEA SUR + ?

Command: **slope**



Proceeding:

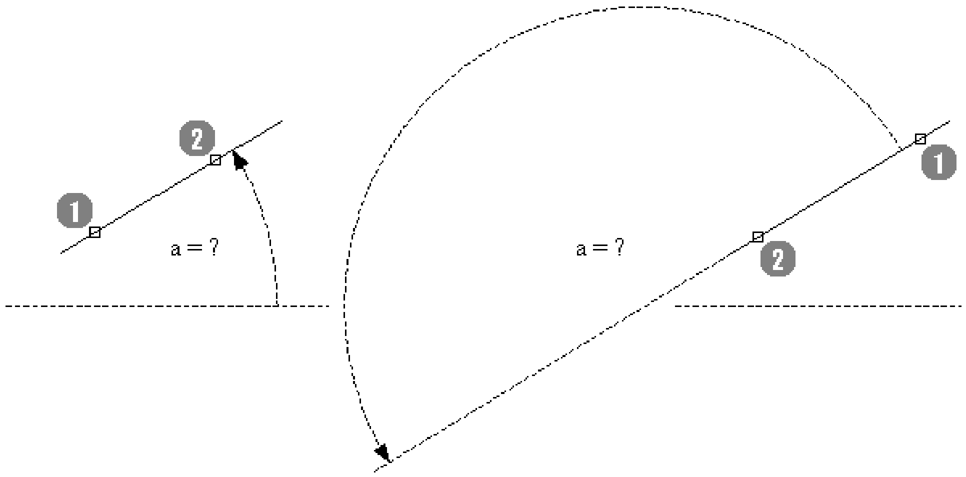
1. Choose the function (with button) or enter command
2. Select the vertex
3. Select both lines
4. Angle value is displayed

Hints:
Please observe direction counter-clockwise.

Sloped angle of a line

Button: + MEA SUR + ?

Command: **slope**



Proceeding:

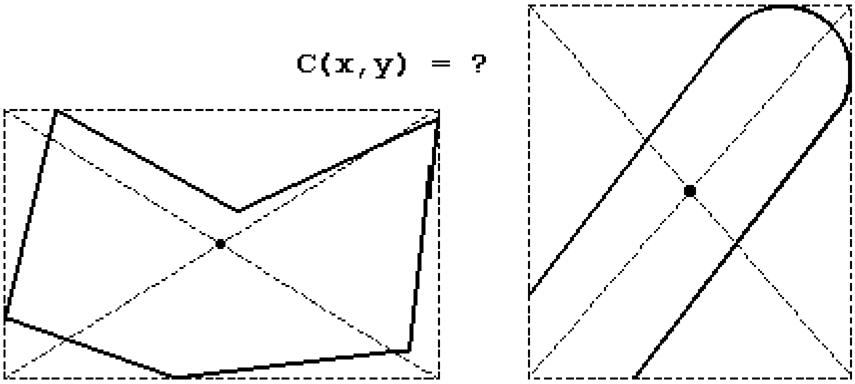
1. Choose the function (with button) or enter command
2. Select 1. point on line
3. Select 2. point on line
4. Angle value is displayed

Hints:
Please observe direction counter-clockwise.

Centre of an object box

Button: + MEA SUR +

Command: **centre ***



Proceeding:
1. Select the object
2. Choose the function (with button) or enter command
3. Centre coordinates are displayed

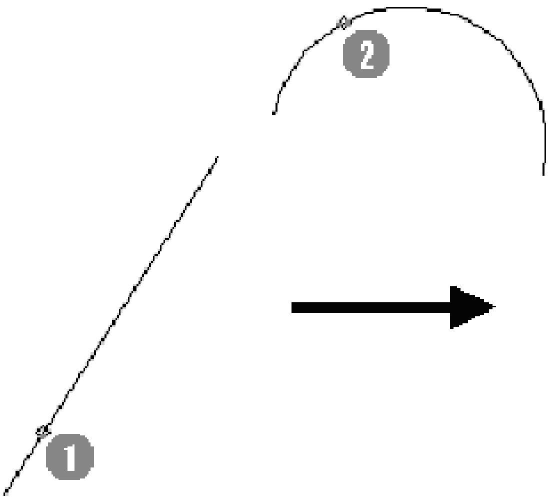
Hints:
Centre coordinates in variable P2D, inclusion of tangents at bezier curves and nurbs.

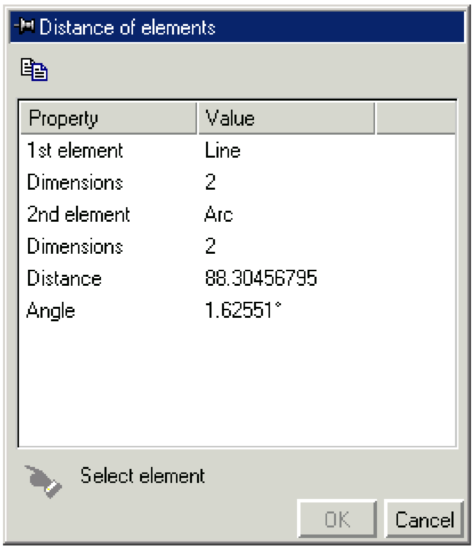
Distance and angle of two elements

Button: + MEASUR + DIFF

Command: **ediff***

Function is important help for contour checking!
(e.g. after error message: "open contour")





Property	Value	
1st element	Line	
Dimensions	2	
2nd element	Arc	
Dimensions	2	
Distance	88.30456795	
Angle	1.62551°	

Select element

Proceeding:

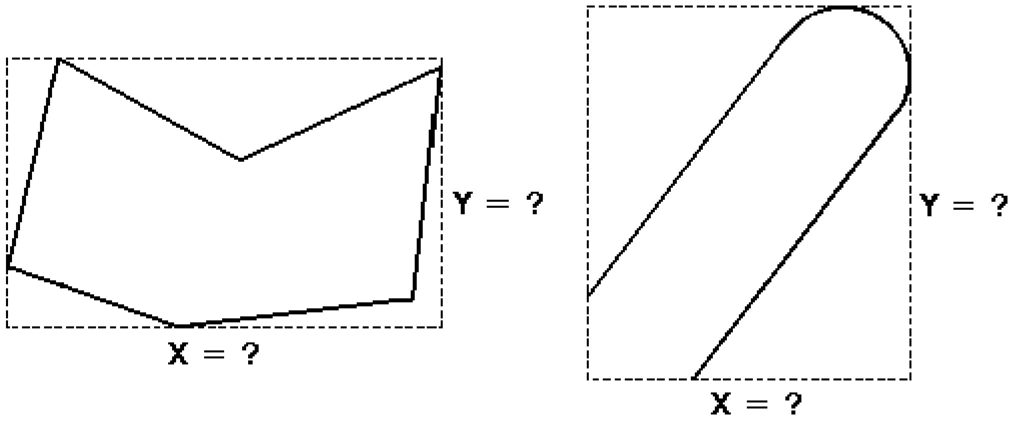
1. Select the object
2. Choose the function (with button) or enter command
3. Centre coordinates are displayed

Hints:
Centre coordinates in variable P2D, inclusion of tangents at bezier curves and nurbs.

Width and height of an object

Button: ⚒ + MEA SUR + ?

Command: **size ***



Proceeding:

1. Select the object
2. Choose the function (with button) or enter command
3. Values are displayed

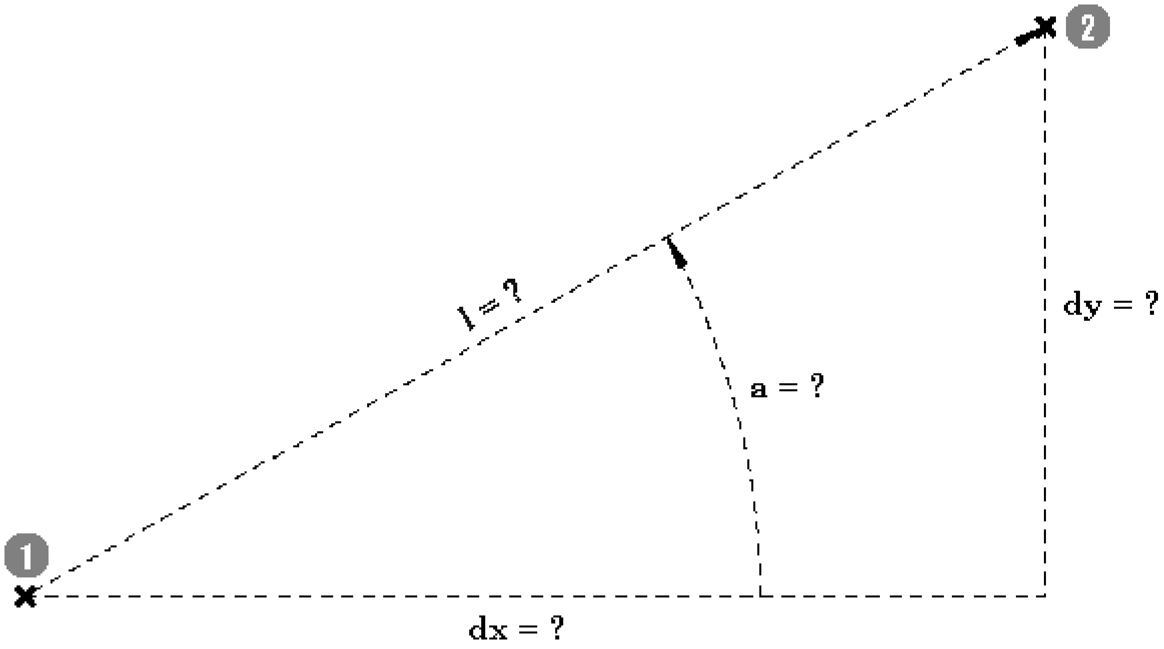
Hints:

Width, height and proportion (R = X/Y) are displayed, inclusion of tangents at bezier curves and nurbs.

Vector between two points

Button: + MEA SUR + 2P ?

Command: **vector -up**



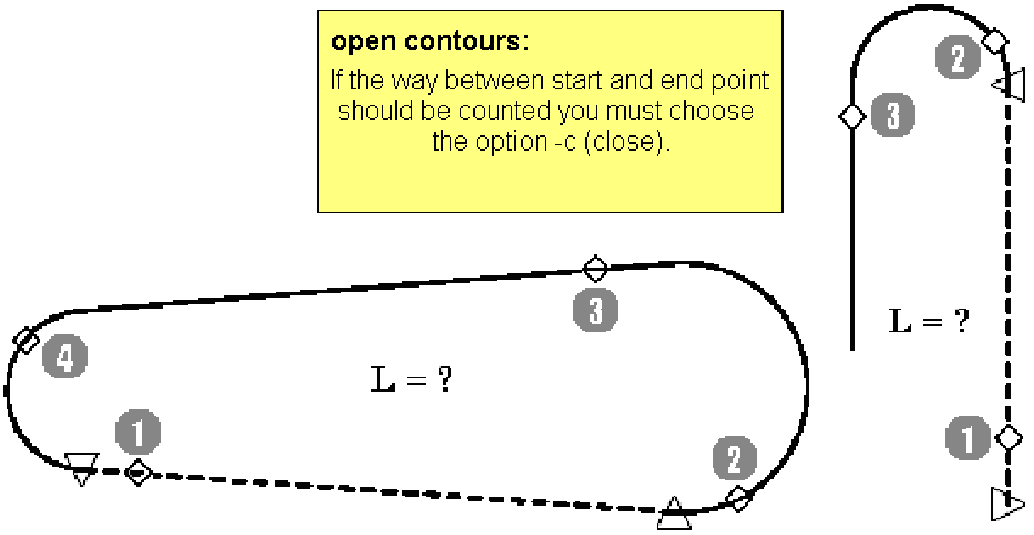
Proceeding:
1. Choose the function (with button) or enter command
2. Select 1. point on line
3. Select 2. point on line
4. Point coordinates and values are displayed

Hints:
Please observe direction counter-clockwise, point coordinates in variable VECTOR.

Length of a particular curve section

Button: + MEA SUR +

Command: **olen**



open contours:
If the way between start and end point should be counted you must choose the option -c (close).

Proceeding:

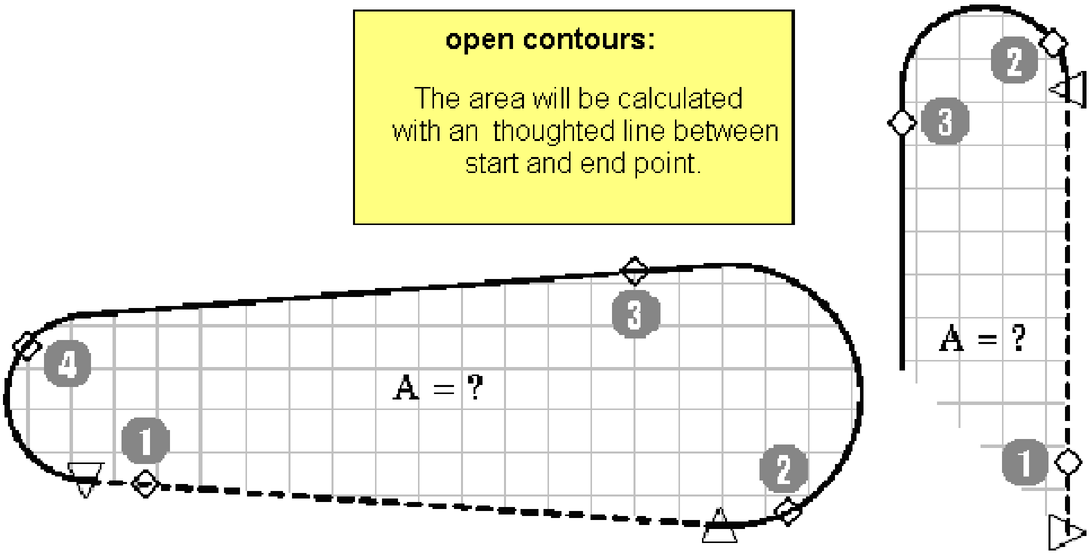
1. Choose the function (with button) or enter command
2. Select the 1. element
3. Select the following elements counter-clockwise
4. Complete opened contour with POLYEND

Hints:
Exclusively applicable to polygons and ellipses/circles, number of decimal digits from variable ROUND

Area of a particular curve section

Button: ⚒ + MEASUR + ?

Command: **olen -a**



open contours:

The area will be calculated with an thoughted line between start and end point.

Proceeding:

1. Choose the function (with button) or enter command
2. Select the 1. element
3. Select the following elements counter-clockwise
4. Complete opened contour with POLYEND

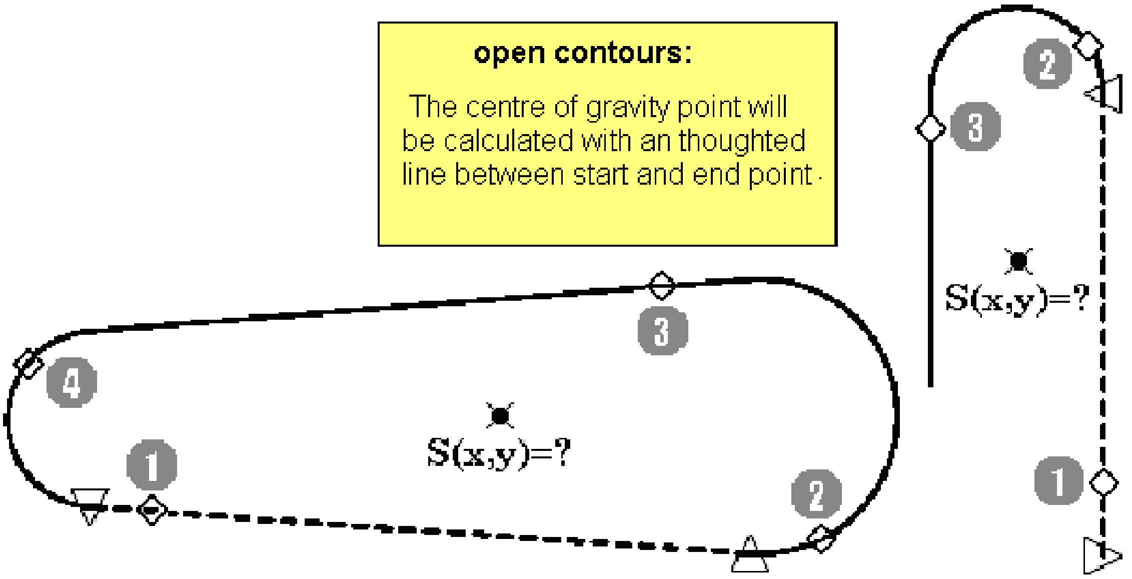
Hints:

Exclusively applicable to polygons and ellipses/circles, number of decimal digits from variable ROUND

Area centre of a particular curve section

Button: + +

Command: **olen -s**



open contours:

The centre of gravity point will be calculated with an thoughted line between start and end point.

Proceeding:

1. Choose the function (with button) or enter command
2. Select the 1. element
3. Select the following elements counter-clockwise
4. Complete opened contour with POLYEND

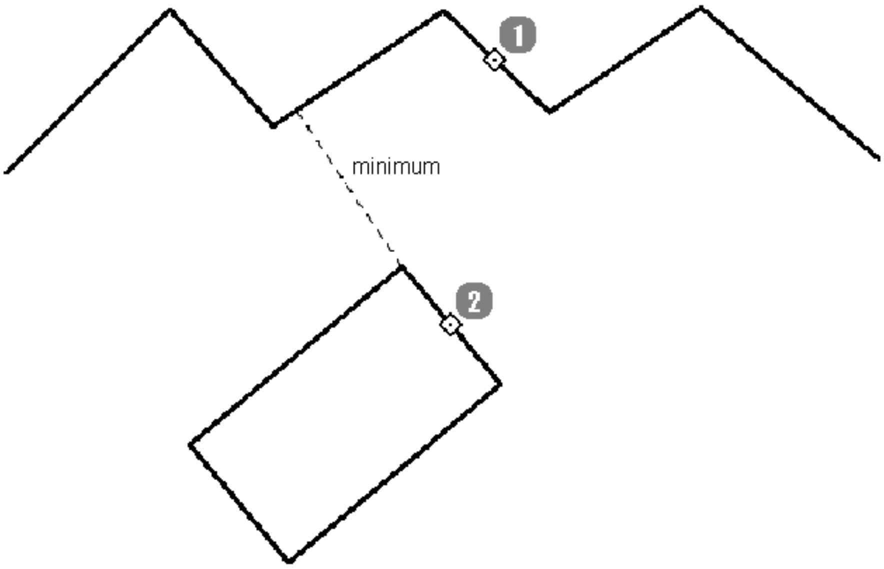
Hints:

Variable P2D contains coordinates of area centre, applicable to polygons and ellipses, number of decimal digits from variable ROUND

Minimum distance between two objects

Button: + MEA SUR + MIN

Command: **odis -oi**



Proceeding:

1. Choose the function (with button) or enter command
2. Select the 1. object
3. Select the 2. object

2D-Dimension

Overview Dimension

The menus for dimension after DIN in **isyCAD/CAM 2.5 (light)** can be called with the buttons of "2D-Tools".

Button: 2D + DIM oder 2D + ADIM

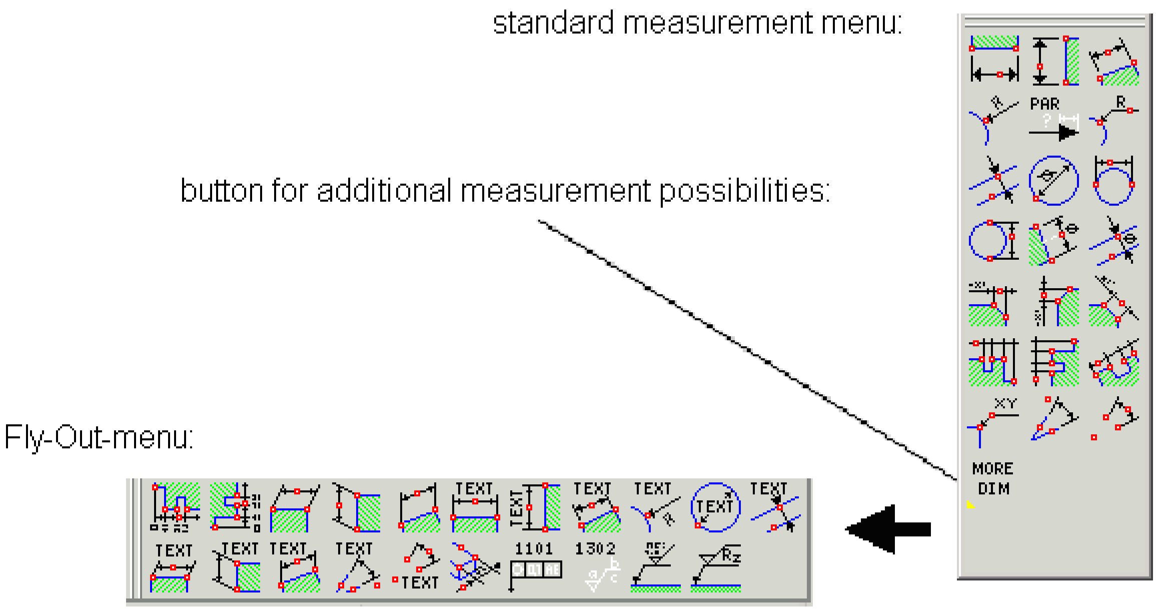
Generally, there are two dimension options:

- Standard dimension
- Associative dimension

Standard dimension

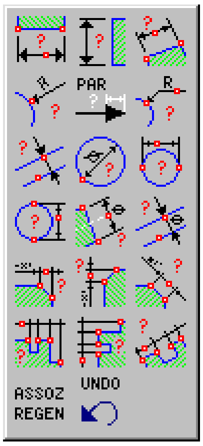
Button: 2D + DIM

After selecting the body edges which shall be dimensioned, the position of the dimension line is fixed with mouse. The dimension or the dimension text is usually derived from geometry.


standard measurement menu:
button for additional measurement possibilities:
Fly-Out-menu:
PAR
R
XY
MORE
DIM
TEXT
1101
1302

Associative dimension

Button: 2D + ADIM



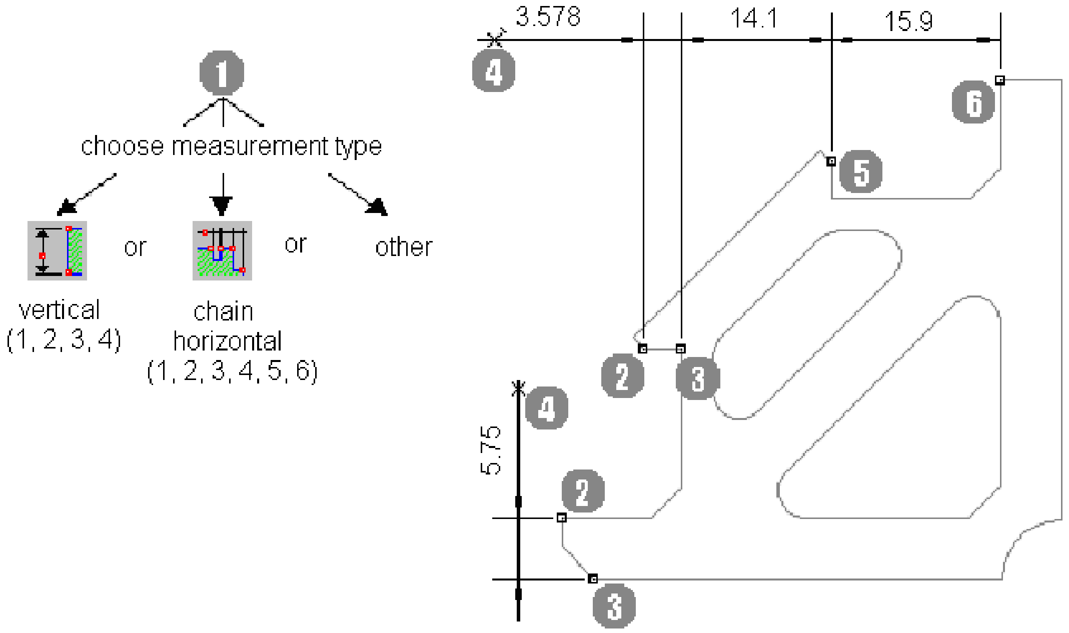
The dimensions produced associatively can be regenerated automatically after subsequent geometry changes.
The associative dimension is produced like the standard dimension, but this dimension is connected associatively with geometry.
Since all ways of dimension cannot be associative with geometry (e.g. text dimension), the performance range is limited as opposed to the standard dimension.
Therefore, a fly out menu is not offered.

Principle of interactive dimension

Button: 2D + DIM + ▦

Adjustable parameter:

Arrow length, kind	(ARR), dim_par1
Distance projection line	(DIML1), dim_par2
Overshoot projection line	(DIML2), dim_par3
Font, size	(TXTSPEC), dim_par4
Decimal digits	(ROUND), dim_par5



Interactions:

1. Call dimension command with button
2. Click on the edges of object which shall be dimensioned
3. Click on the position of the dimension line
4. More edges of object at incremental dimension or relative dimension

Hints:

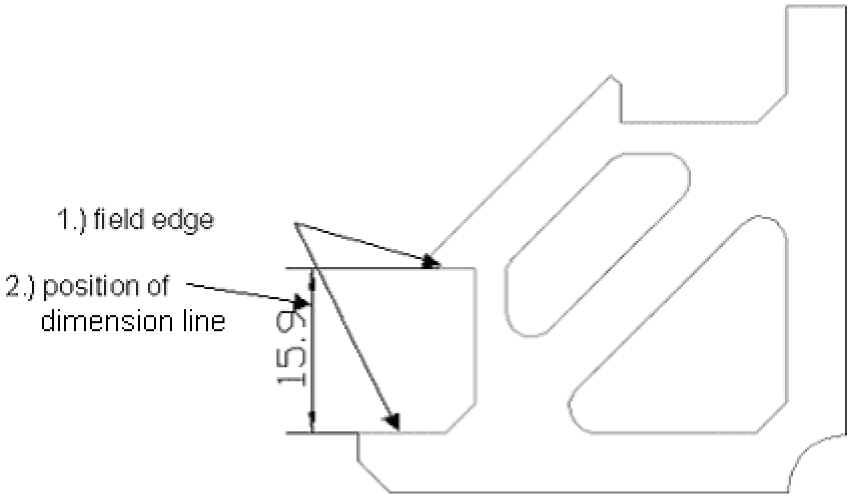
Equal distances between several dimension lines are generated with switched-on grid.

Adjustments, Parameter

General: Semiautomatic dimension

Button: 2D + DIM +

Principal selection order:



Proceeding:

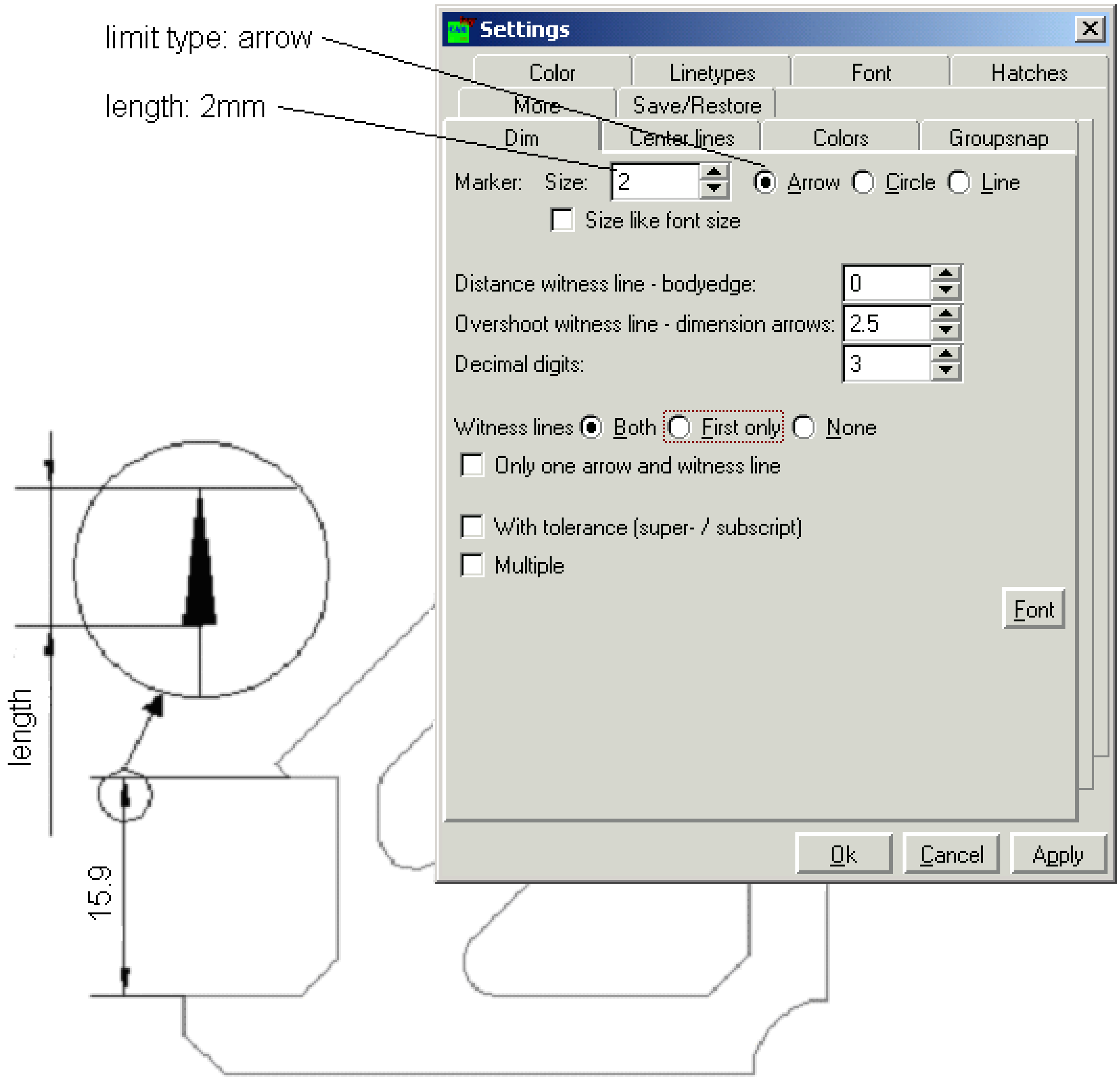
1. Call dimension command with button
2. Click on the edges of object which shall be dimensioned
3. Click on the position of the dimension line

Hints:

Equal distances between several dimension lines are generated with switched-on grid SNAP.

Dimension parameter: Limit symbol

Button: PARAM *or* 2D + DIM + PAR



Marker: Size: 2 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 0

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

□ Only one arrow and witness line

□ With tolerance (super- / subscript)

□ Multiple

Proceeding:

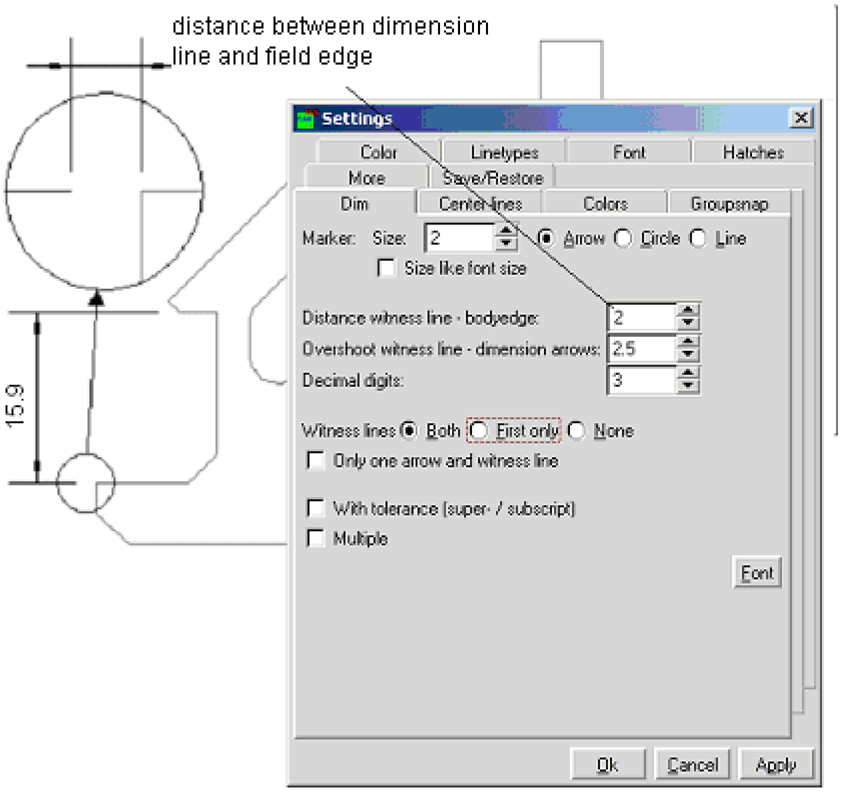
1. Open dialog box : Dimension
2. Enter arrow length (in mm)

Hints:

The dimension settings can be stored and loaded in a file (with setting dialog).

Dimension parameter: Distance of dimension lines

Button: PARAM *or* 2D + DIM + PAR


distance between dimension line and field edge

Marker: Size: 2 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 2

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

□ Only one arrow and witness line

□ With tolerance (super- / subscript)

□ Multiple

Proceeding:

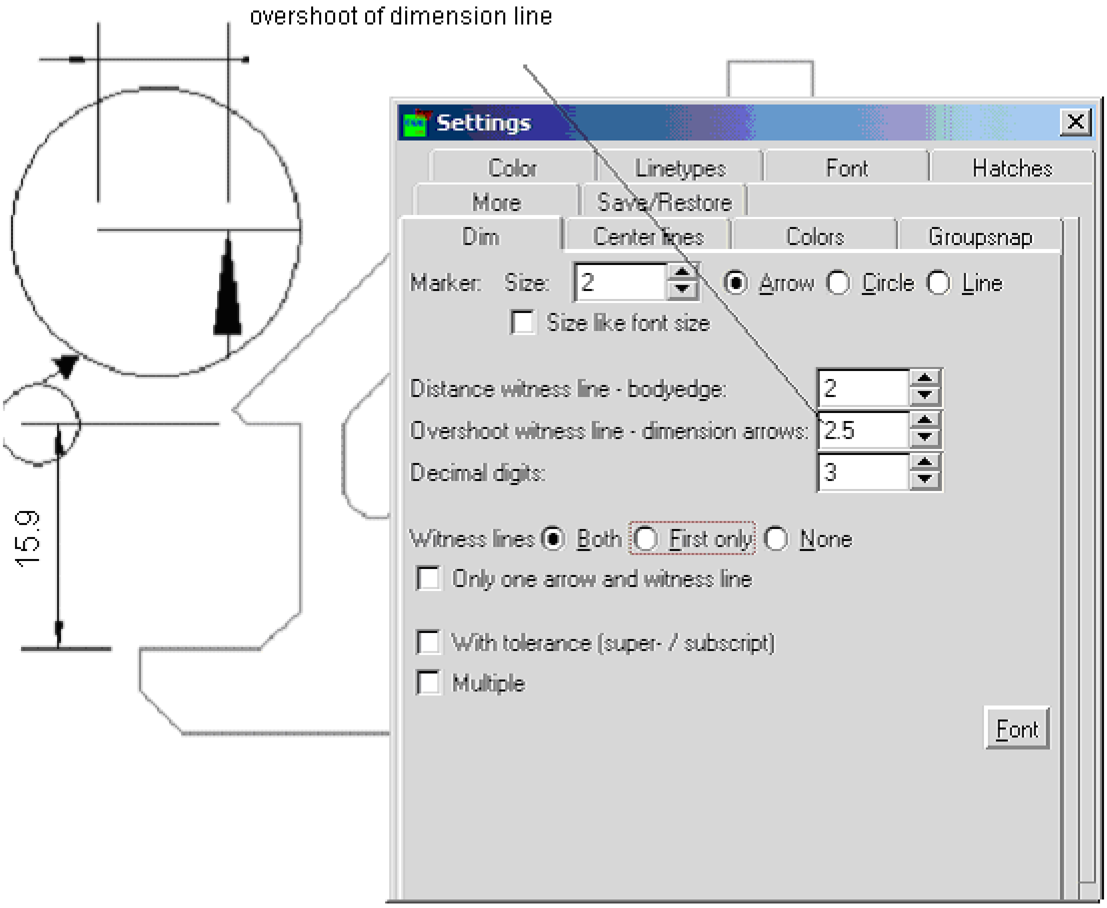
1. Open dialog box: Dimension parameter
2. Enter distance (in mm)

Hints:

The dimension settings can be stored and loaded in a file (with setting dialog).

Dimension parameter: Overshoot of the dimension line

Button: PARAM *or* 2D + DIM + PAR



Settings

Color | Linetypes | Font | Hatches | More | Save/Restore | Dim | Center lines | Colors | Groupsnap

Marker: Size: 2 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 2

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

□ Only one arrow and witness line

□ With tolerance (super- / subscript)

□ Multiple

Font

Proceeding:

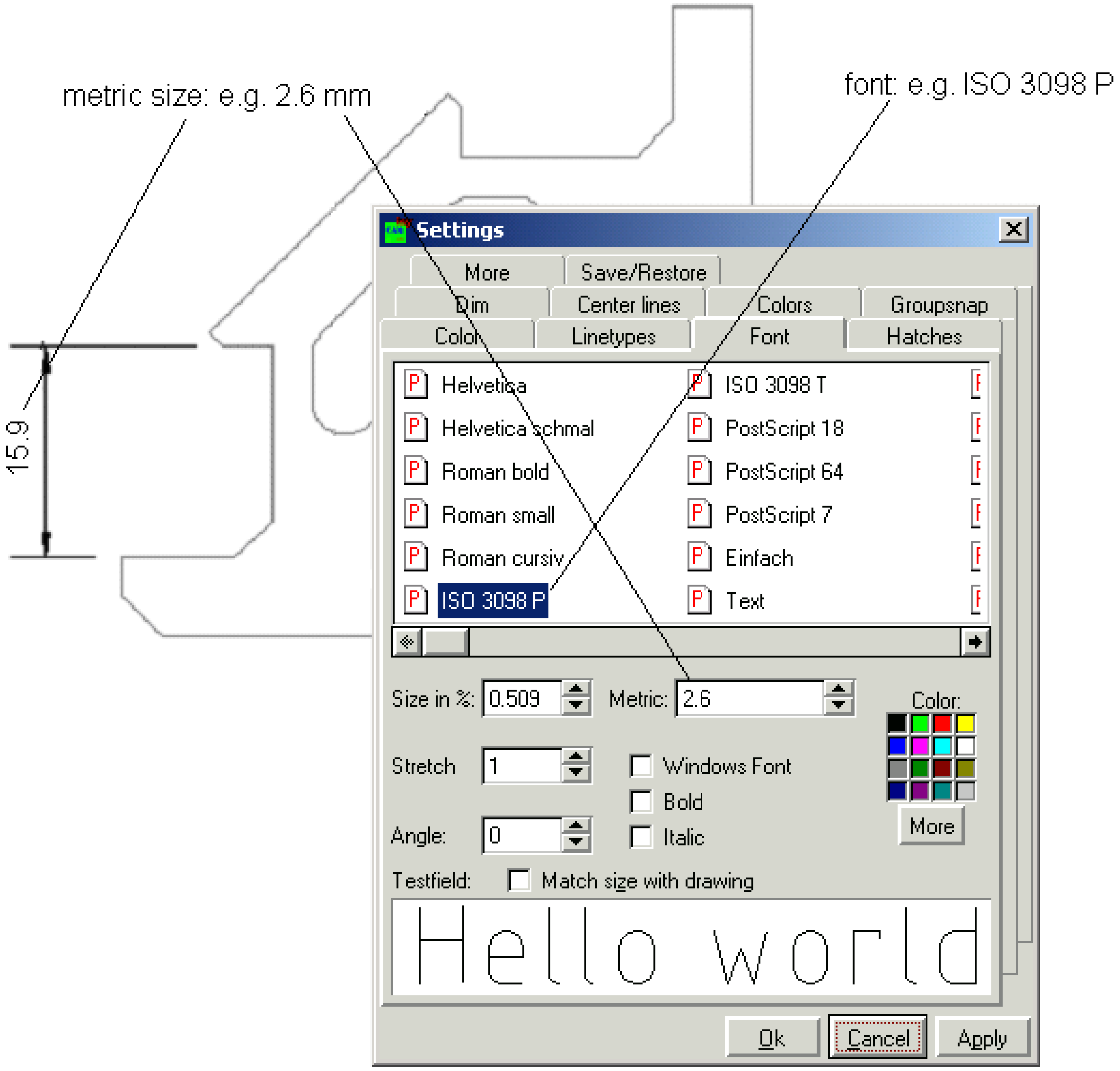
1. Open dialog box: Dimension parameter
2. Enter overshoot (in mm)

Hints:

The dimension settings can be stored and loaded in a file (with setting dialog).

Dimension parameter: Font and size

Button: PARAM *or* 2D + DIM + PAR



Size in %: 0.509
Metric: 2.6
Stretch 1
Angle: 0
□ Windows Font
□ Bold
□ Italic
□ Match size with drawing

Proceeding:

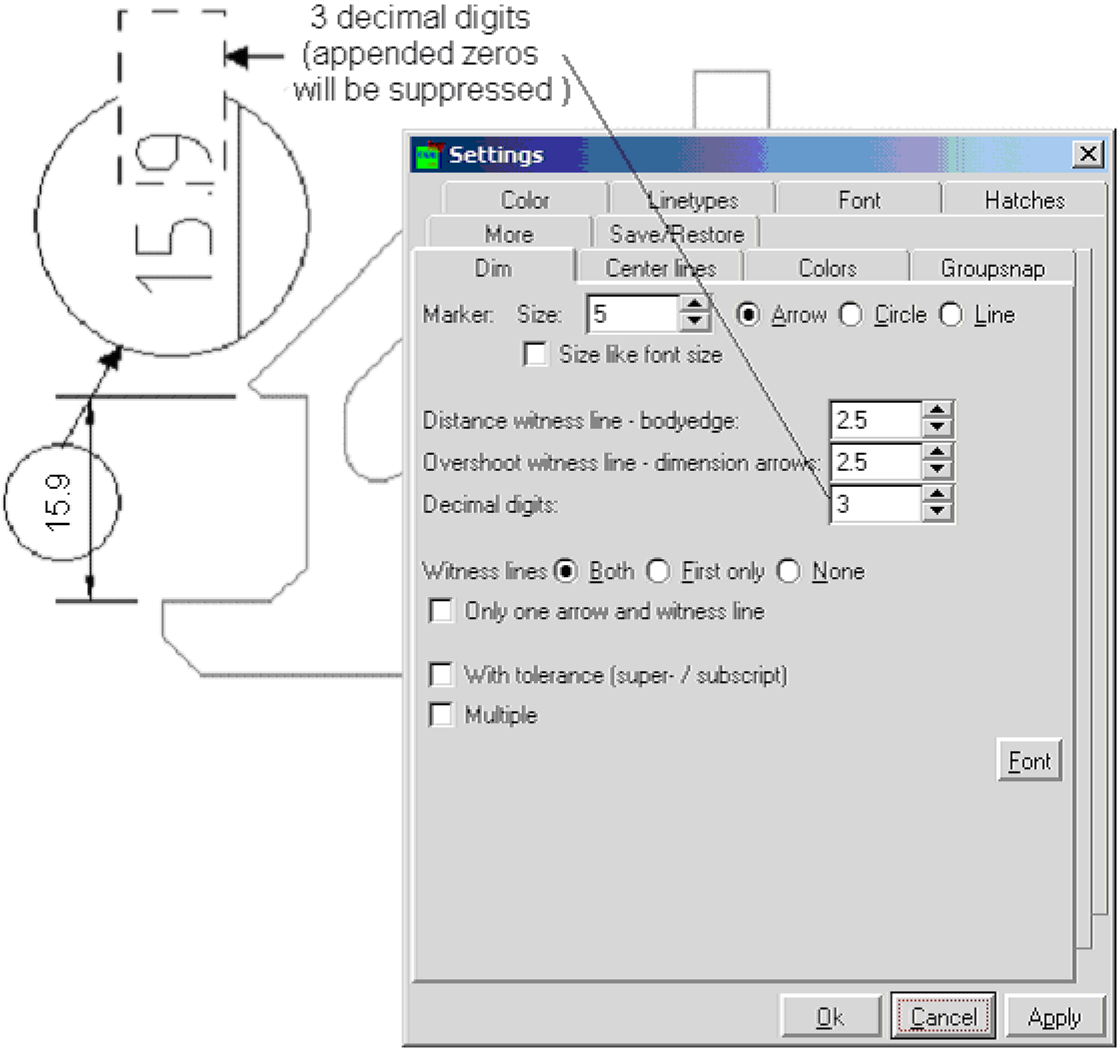
1. Open dialog box: Font
2. Enter font
3. Enter size (in mm)

Hints:

The colour is independent of the set colour of geometry objects.

Dimension parameter: Rounding of measured value

Button: PARAM *or* 2D + DIM + PAR



Marker: Size: 5 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 2.5

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

□ Only one arrow and witness line

□ With tolerance (super- / subscript)

□ Multiple

Proceeding:

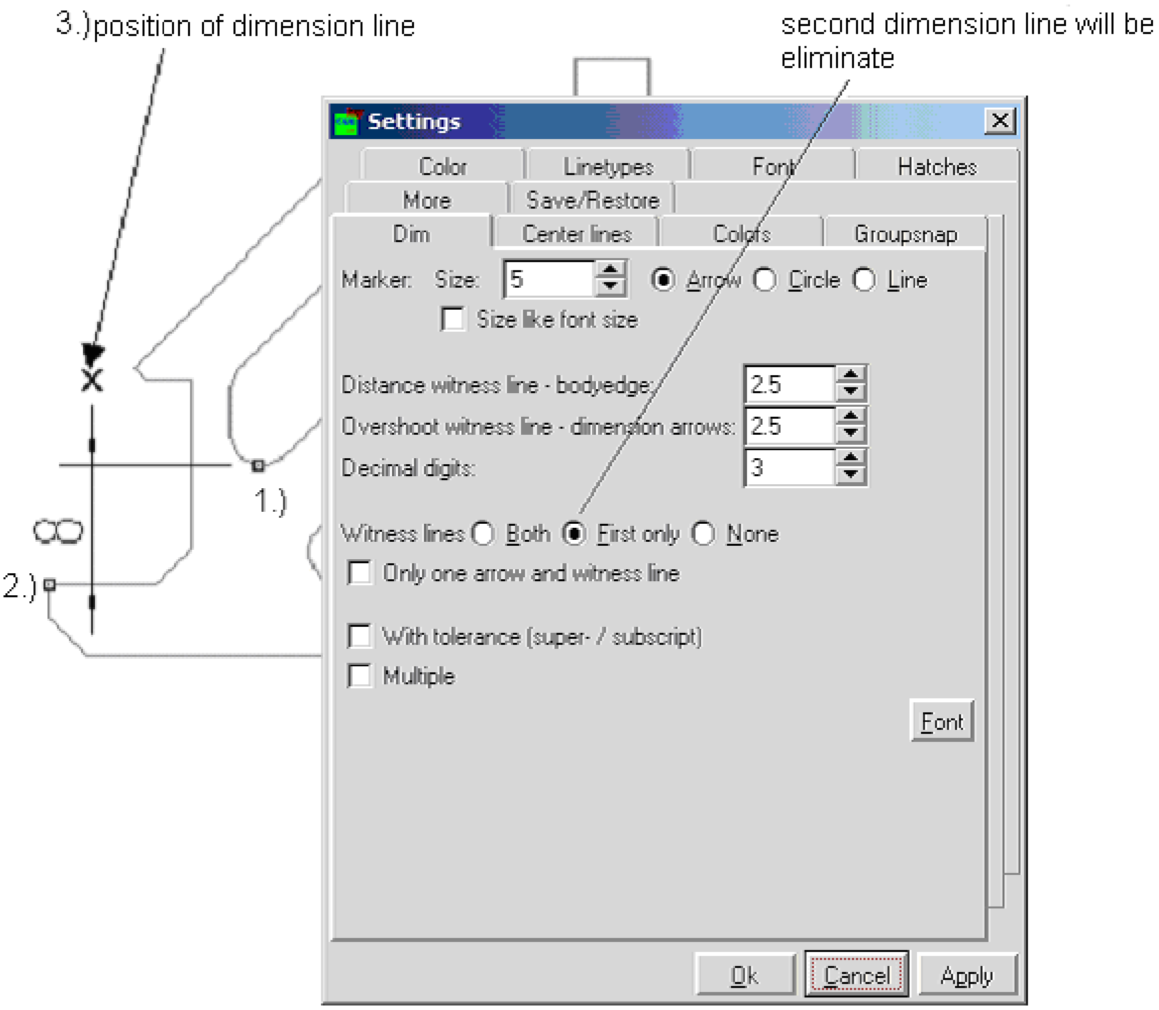
1. Open dialog box: Dimension
2. Enter number of decimal digits

Hints:

The precision of the drawing can be checked well if more than the required fractional digits are set.

Dimension parameter: Eliminate dimension lines

Button: PARAM *or* 2D + DIM + PAR



Marker: Size: 5 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 2.5

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ○ Both ● First only ○ None

□ Only one arrow and witness line

□ With tolerance (super- / subscript)

□ Multiple

Proceeding:

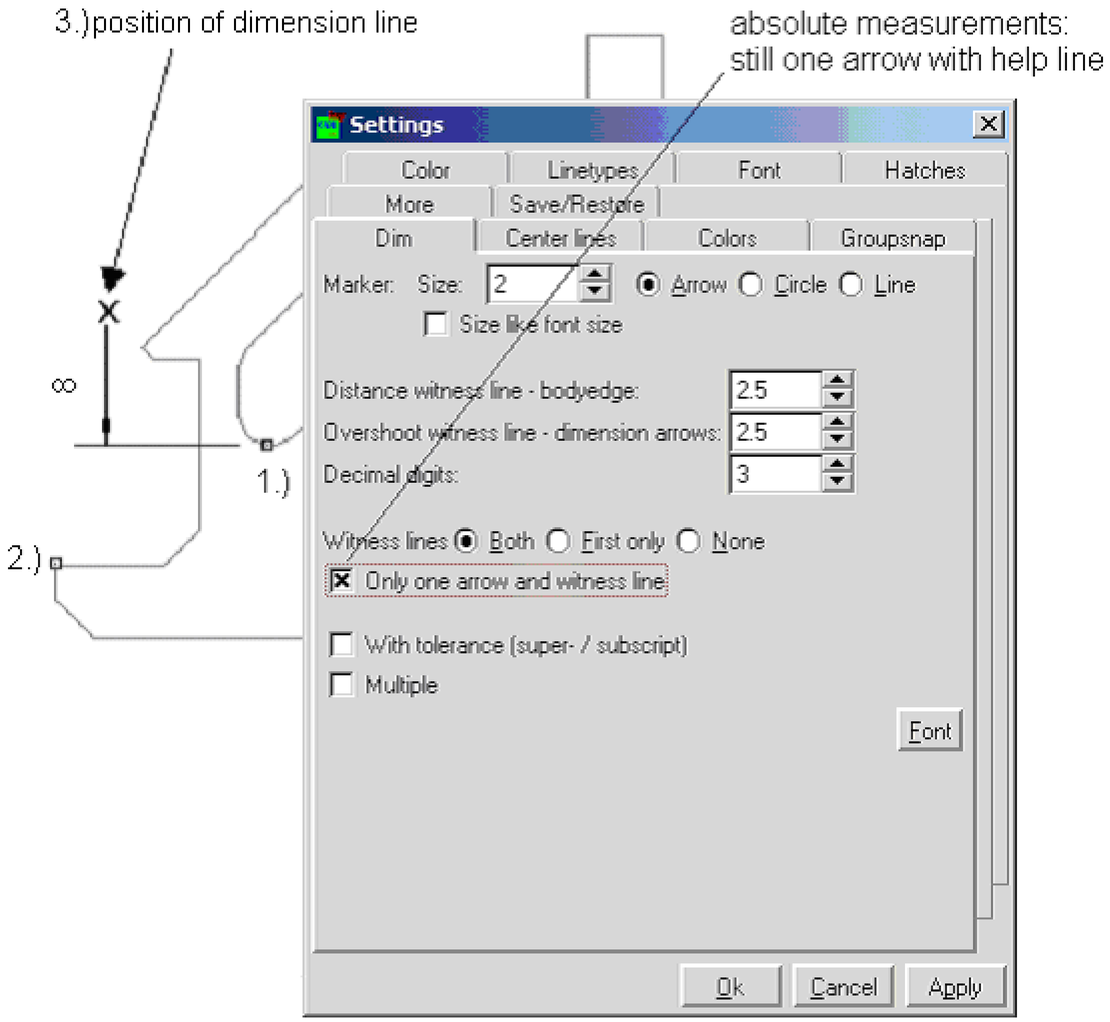
1. Open dialog box: Dimension
2. Enter kind of projection line

Hints:

With elimination of projection line "double lines" can be avoided.

Dimension parameter: Reference measure, single

Button: PARAM *or* 2D + DIM + PAR



3.)position of dimension line

absolute measurements:
still one arrow with help line

Marker: Size: 2 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 2.5

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

☒ Only one arrow and witness line

□ With tolerance (super- / subscript)

□ Multiple

Proceeding:

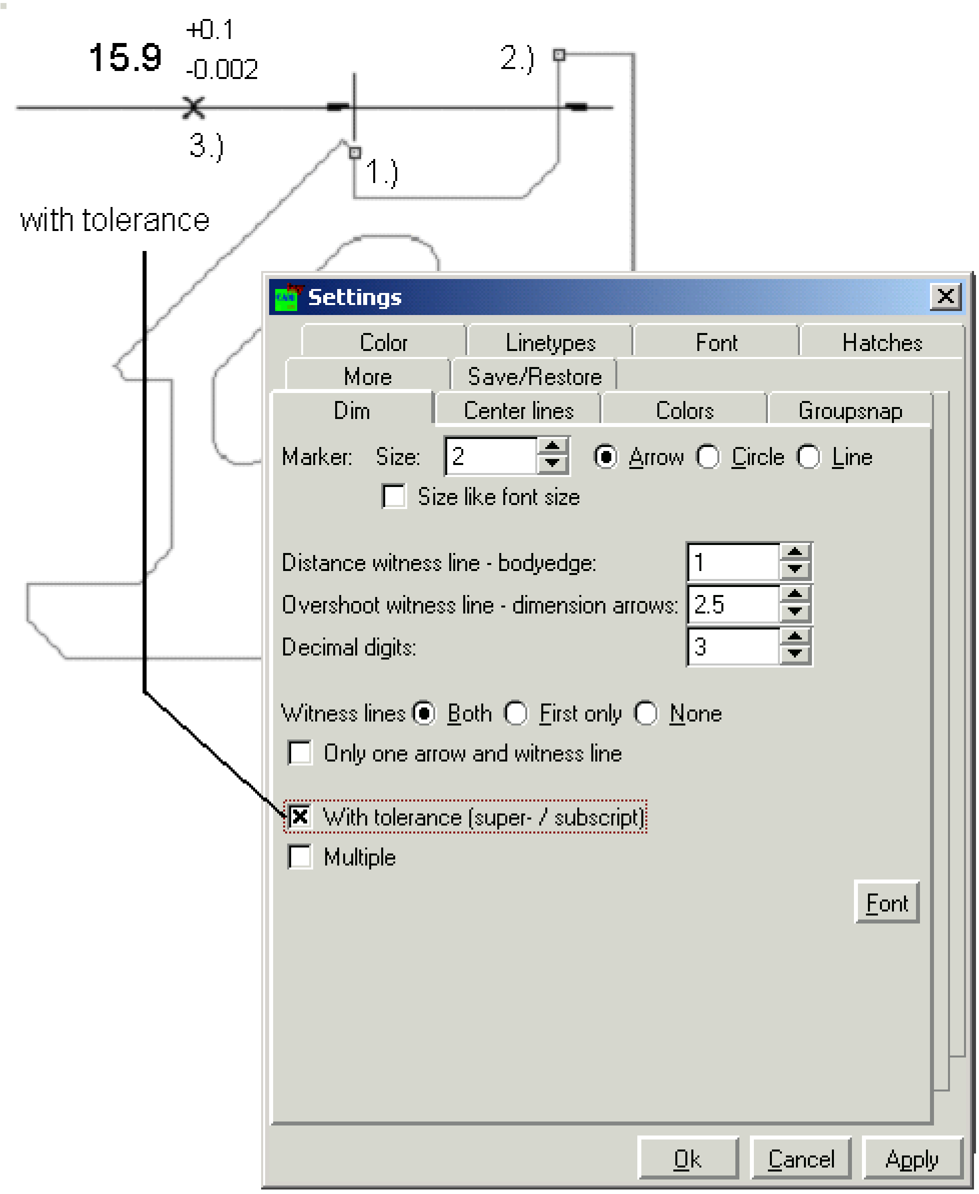
1. Open dialog box: Dimension
2. Enter kind of projection line

Hints:

With elimination of projection line "double lines" can be avoided.

Dimension parameter: Tolerance

Button: PARAM *or* 2D + DIM + PAR



Marker: Size: 2 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 1

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

□ Only one arrow and witness line

☒ With tolerance (super- / subscript)

□ Multiple

Proceeding:

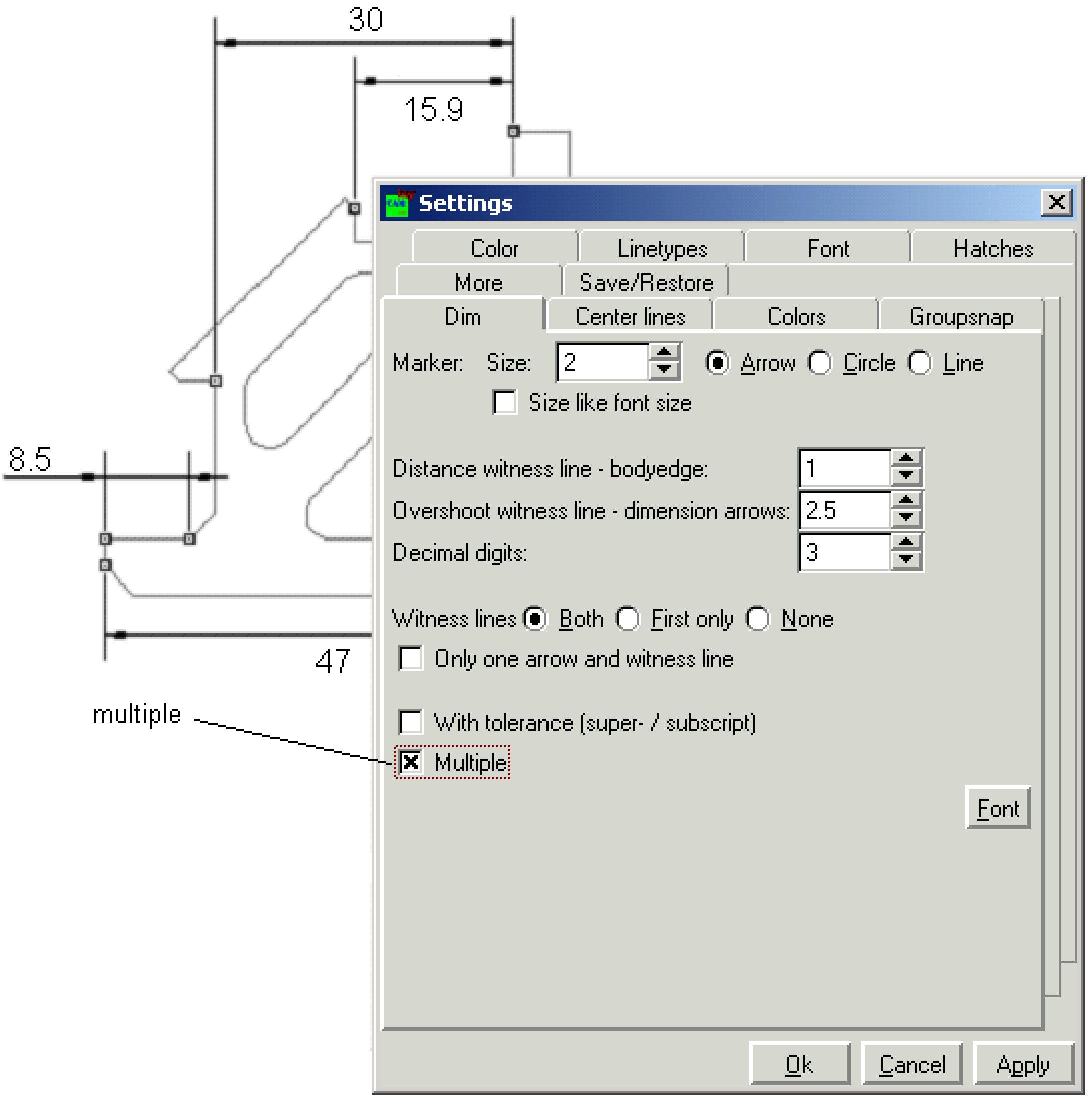
1. Select point according to the measure
2. Determine upper tolerance limit
3. Determine lower tolerance limit

Hints:

At each measure the tolerance will be asked for as text.
The text can be changed later with "newtext -el".

Dimension parameter: Multiple execution

Button: PARAM *or* 2D + DIM + PAR



Marker: Size: 2 ● Arrow ○ Circle ○ Line

□ Size like font size

Distance witness line - bodyedge: 1

Overshoot witness line - dimension arrows: 2.5

Decimal digits: 3

Witness lines ● Both ○ First only ○ None

□ Only one arrow and witness line

□ With tolerance (super- / subscript)

☒ Multiple

Proceeding:

1. Select "Multiple"
2. Select dimension
3. Generate all measures in one go
4. Complete with <POLYEND>

Hints:

The same distances between several dimension lines are obtained with active grid.

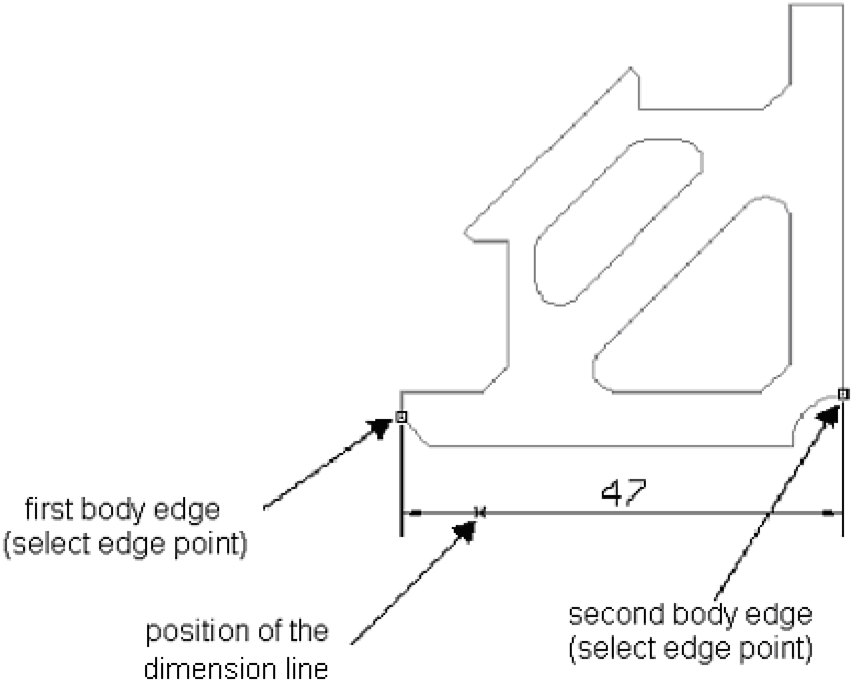
Standard dimension menu

Horizontal dimension

Button: 2D + DIM + ↔

Command: **dim all -hfs**

all	snap to all objects.
-h	horizontal dimension
-f	the arrowhead is filled (DIN).
-s	the measure number will be assumed from the geometry



Proceeding:

1. Call dimension command with button
2. Select the first edge of the object
3. Select the second edge of the object
4. Determine position (distance) of the measure line

Hints:

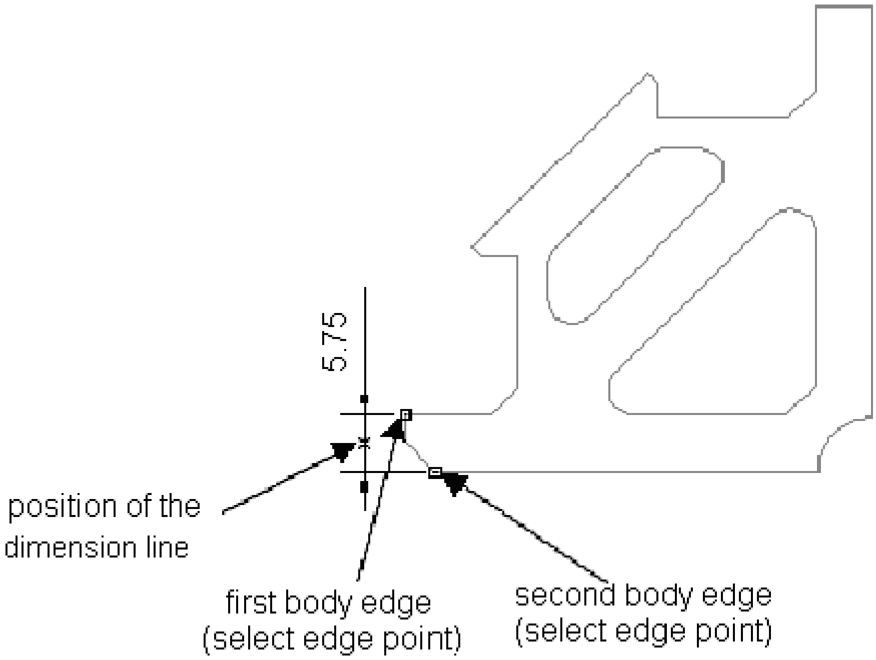
If the dimension doesn't match between the arrows, the dimension line is extended in direction of the first point.

Vertical dimension

Button: 2D + DIM + ↕

Command: **dim all -fsv**

all	snap to all objects.
-v	vertical dimension
-f	the arrowhead is filled (DIN).
-s	the measure number will be assumed from the geometry


5.75
position of the dimension line
first body edge (select edge point)
second body edge (select edge point)

Proceeding:

1. Call dimension command with button
2. Select the first edge of the object
3. Select the second edge of the object
4. Determine position (distance) of the measure line

Hints:

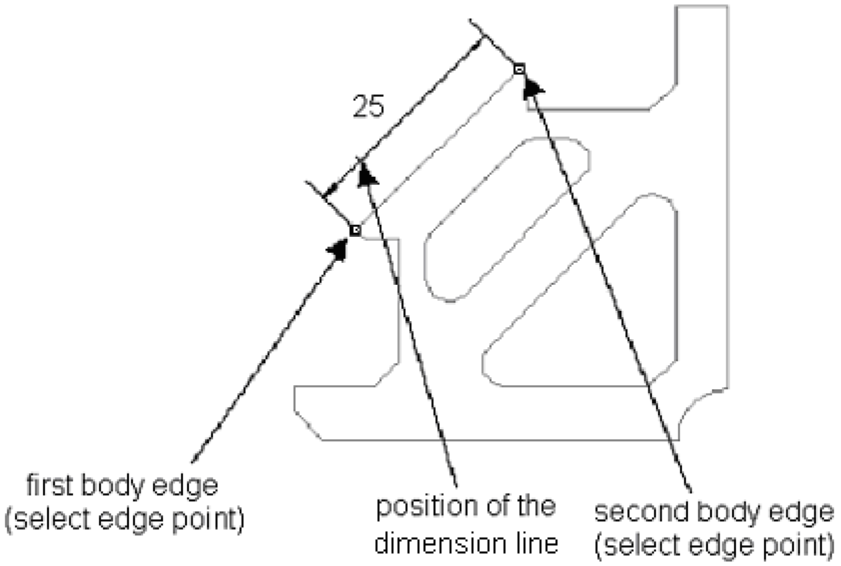
If the dimension doesn't match between the arrows, the dimension line is extended in direction of the first point.

Inclined dimension

Button: 2D + DIM +

Command: **dim all -fsx**

all	snap to all objects.
-x	inclined dimension
-f	the arrowhead is filled (DIN).
-s	the measure number will be assumed from the geometry



Proceeding:

1. Call dimension command with button
2. Select the first edge of the object
3. Select the second edge of the object
4. Determine position (distance) of the measure line

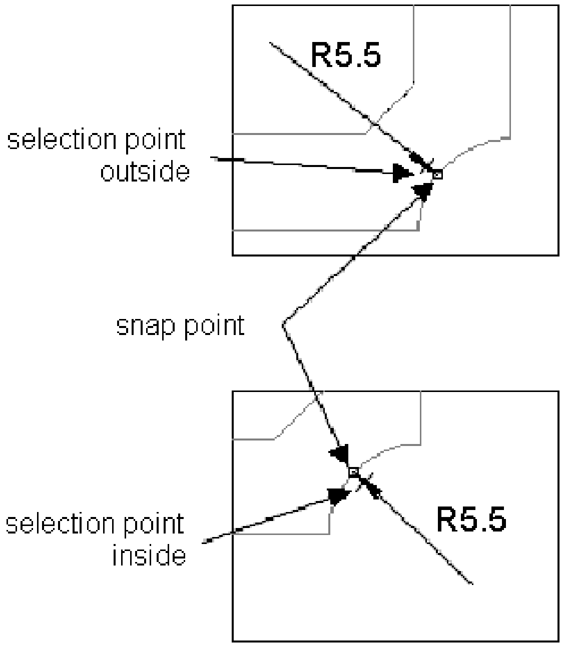
Hints:
If the dimension doesn't match between the arrows, the dimension line is extended in direction of the first point.

Radius dimension

Button: + +

Command: **dim all -fsr**

all	snap to all objects.
-r	radius dimension
-f	the arrowhead is filled (DIN).
-s	the measure number will be assumed from the geometry



Proceeding:

1. Call command with button
2. Select circle arc inside or outside

Hints:

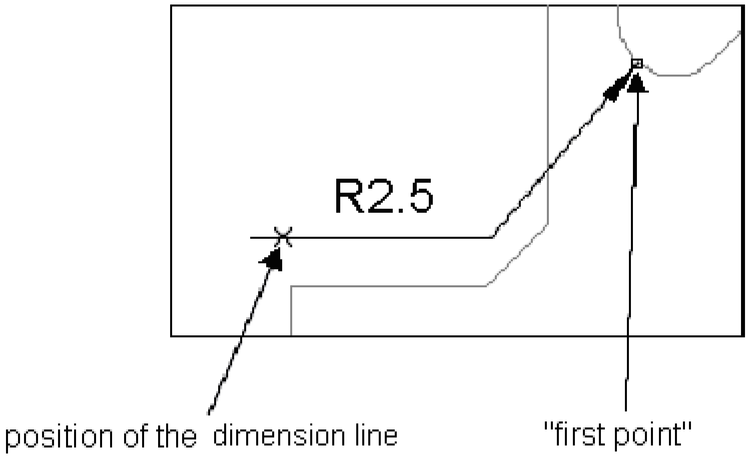
The second arrowhead marks the centre at inner radius if free space is available. The dimension line arises in the direction of the centre.

Radius dimension at cracked arrow

Button: 2D + DIM + R

Command: **dim all -fsrh**

all snap to all objects.
-r radius dimension
-f the arrowhead is filled (DIN).
-s the measure number will be assumed from the geometry
-h horizontal dimension
(-v vertical dimension)



Proceeding:

1. Call dimension command with button
2. Select horizontal or vertical text
3. Select circle arc inside or outside
4. Determine position of dimension line

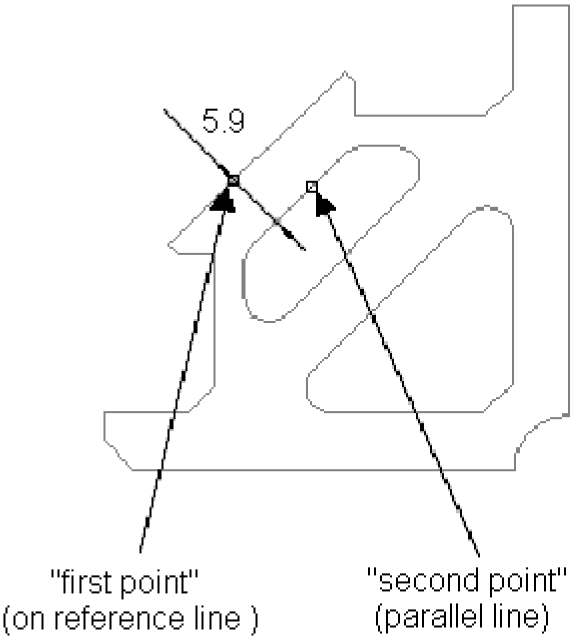
Hints:
The dimension arrow arises in direction of the centre and the dimension line goes horizontally or vertically through the second selection point.

Dimension of parallel lines

Button: + +

Command: **dim all -fsp**

all	snap to all objects.
-p	parallel lines
-f	the arrowhead is filled (DIN).
-s	the measure number will be assumed from the geometry



Proceeding:

1. Choose command with button
2. Place „First point“ at reference line
3. Select the parallel with „Second point“

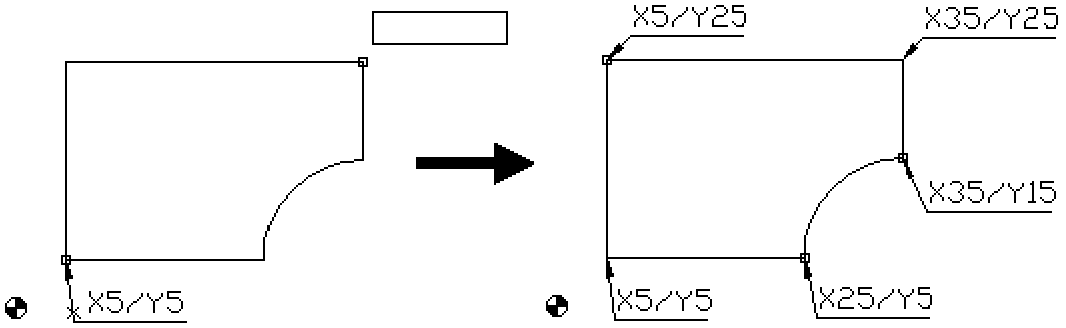
Hints:

The dimension line goes through the "first point" vertical to the reference line. Dimension lines are not generated.

Absolute and relative coordinate dimension

Button: 2D + DIM + XY

Command: **codim**
codim -r (relative)



Proceeding:

1. Call the function
2. If necessary, (option -r) select zero point
3. Select the dimensioning point and the text position (for all points).
4. Complete: Right mouse key or <F6> (POLYEND)

Hints:
There is a difference between right and left arrows.

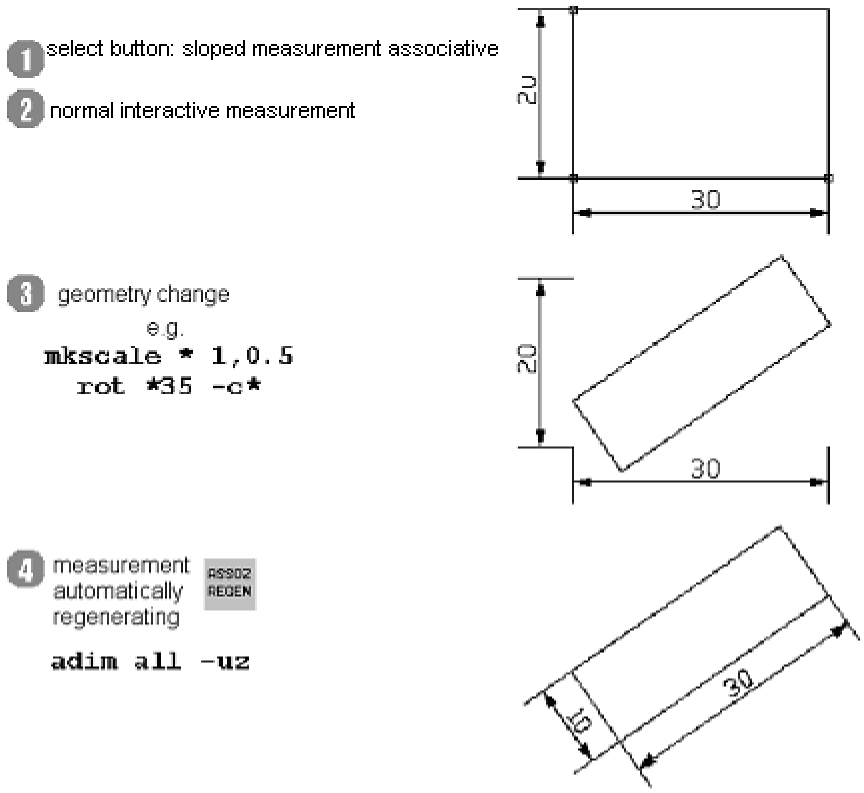
Associative dimension

Button: 2D + ADIM ? + ?

Command: **adim -fx**

-x inclined dimension

-f the arrowhead is filled (DIN).



1 select button: sloped measurement associative

2 normal interactive measurement

3 geometry change
e.g.

```
mkscale * 1,0.5
rot *35 -c*
```

4 measurement automatically regenerating

```
adim all -uz
```

Proceeding:

1. Choose dimension command with button
2. Carry out interactive dimension
3. Changing geometry
4. Adapt dimension to the changed geometry

Hints:

At later turn of geometry, you should already be using associative inclined dimension.

2D-Data Exchange

Introduction

Data exchange: Over the years, most different CAD, CAM and graphic systems were developed, propagated and applied (named in the following Cax system). The necessary information exchange between producer and supplier, and also the distribution of task between different branches of a company required the exchange and editing of data from different CAx systems.

Because of the different developments, a geometry model of Cax system A cannot be adapted straightaway into CAx system B. For this purpose, software interfaces are necessary, which can be used for data exchange with a little loss of information as possible.

Direct interfaces and system neutral interfaces are distinguished.

Direct interfaces: If the CAx system B has a direct interface to the CAx system A, system B can read (and if necessary also output) the original (native) data of the system A without a previous transformation. This method, simple at first sight, does, however, have serious disadvantages. Each direct interface must be purchased, and thus generates costs, and communication with many different CAx systems can cause considerable strain for the system.

More difficulties arise from the available updates (generally more than one per year). If the CAx system B has a direct interface to the CAx system A, there is no general guarantee that it will still work after the next update of the system A. An update (if available) will then also be necessary for system B.

System neutral interfaces: System neutral interfaces are data interchange formats which are defined, published and maintained by different institutions or which have gained worldwide acceptance through frequent use (e.g. like the English language).

The data interchange works in a way that the CAx system A changes and stores its original data in one of these formats. System B reads these data and changes them to fit into its structures. The essential advantage of this method is that a few interfaces are enough to exchange data with almost all others CAx systems.

Moreover, system-neutral interfaces are commonly used worldwide. There are different versions, but they are usually constant over many years.

IsyCAM 2.5 exclusively owns system-neutral interfaces which will be described in the next section.

In view of the variety of possibilities, coordination of the choice of the interface and the converting settings of the data exporter and the data importer is recommendable, and for permanent business relations, it is mandatory to find standard settings. All neutral interfaces have in common that they cannot realise the complete scope of services in each CAx system. Therefore, system-specific elements which cannot be transferred should be avoided.

Also, the knowledge of the users has considerable influence on the conversion results. On the one hand, the data quality of the origin system must correspond to the demands, on the other hand, the operation and configuration of the interface plays a decisive role. Frequently, operator errors are a reason for bad data quality and cause additional problems for further processing.

File Import

Overview: Open external format

Import of external files: The list with the available formats on the respective PC is identifiable with mouse click on

File -> Import ...

in Menu “File“ of the menu bar of isyCAM 2.5 (light).

In the following window you can scroll to „File type“ (bottom line) and select the desired format.

The following formats can be imported into isyCAM 2.5 (light).

Name	**Extension**	**Comment**
IsyCAM - Drawings	*.vec, *.vcz	Standard format of isyCAM 2.5
STL - Data	*.stl,*.sla	Area interface (angle stitches)
Drawing Exchange Format	*.dxf	DXF is the standard format that was defined by the producer of the program AutoCAD.
Adobe Illustrator Files	*.ai	Import of contours
Corel/Adobe Illustrator EPS	*.eps	Import of contours
HP-Graphical Language HPGL	*.hp	
ISEL - NCP/PGM files	*.ncp,*.pgm	ISEL Intermediate format after DIN 66025 Output of 2D NC-Programs with the CAM modules milling, drilling, engraving.
NC Code (G Syntax)	*.nc	Output of 2D NC-Programs with the CAM modules milling, drilling, engraving.
Bitmap to vector	*.tif,*.bmp, *.jpg, *.png	Insert a graphic as drawing object, dependent on the installed graphic filter

DXF and sometime also EPS, AI and HPGL are used for a widely loss-free data interchange between CAD/CAM systems for 2D data. All other formats require a higher processing cost or they are provided for other applications (like digitalisation of presentations, documentation, inserting pictures, word programs, controlling of machines etc.). Most formats are shown below in greater detail.

Data transfer

Electronic data transfer of CAD/CAM-data: Besides the traditional data transfer with data carrier like disk and CD, the transfer on electronic way e. g. with FTP, ISDN, email or special systems (e.g. ODETTE) has become increasingly important.

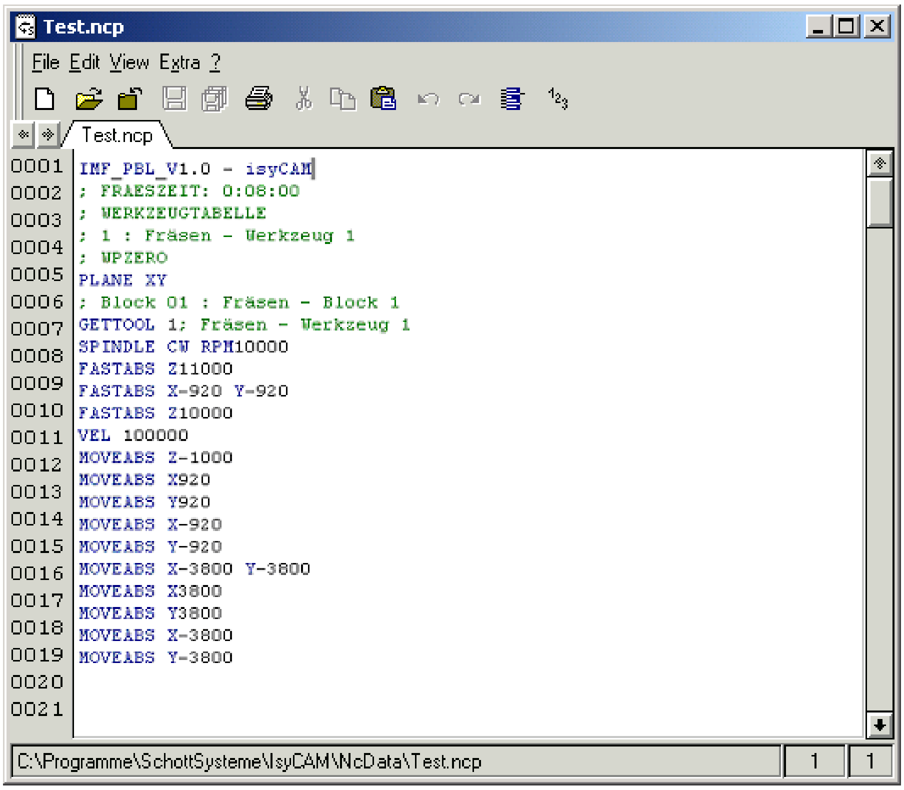
Working in the practice, it is possible, that data are changed or damaged partially or irreparably by the software transfer/process transfer. It is very advisable to send the data in compressed (packed) form particularly if transferred as email-appendix. If receive such data, please inform your partner about it.

Edit files

Edit of NCP-files:

With many data formats, it is possible to look and edit them with a text editor (e.g. isyCAM text editor). This has the advantage that smaller faults in the syntax (e.g. false file beginning or false file end) can be immediately eliminated by the receiver. With different formats you can get additional information (e.g. about senders, used software, the version number, precisions etc.) out of the header lines (header, global section).

The following picture shows a NCP file exemplarily.



```
IMF_PBL_V1.0 - isyCAM
; FRAESZEIT: 0:08:00
; WERKZEUGTABELLE
; 1 : Fräsen - Werkzeug 1
; WPZERO
PLANE XY
; Block 01 : Fräsen - Block 1
GETTOOL 1; Fräsen - Werkzeug 1
SPINDLE CW RPM10000
FASTABS Z11000
FASTABS X-920 Y-920
FASTABS Z10000
VEL 100000
MOVEABS Z-1000
MOVEABS X920
MOVEABS Y920
MOVEABS X-920
MOVEABS Y-920
MOVEABS X-3800 Y-3800
MOVEABS X3800
MOVEABS Y3800
MOVEABS X-3800
MOVEABS Y-3800
```

C:\Programme\SchottSysteme\IsyCAM\NcData\Test.ncp 1 1

Description, range of application and error handling of some formats

VEC/VCZ (*.vec, *.vcz):

VEC is the standard drawing format of isyCAM. The format VCZ is a compressed (packed) format. The formats VEC and VCZ are used with the programs PICTURES BY PC and their modules and isyCAM 2.5 (light)/3.0.

Apart from a few supplements, which are necessary additions in course of the development, the VEC format has not ben subjected to any changes, i.e. geometry data generated years ago can still be read and edited without problems.

Drawing Exchange Format (*.dxf)

The DXF standard (Drawing eXchange Format) is based on a definition of the company AUTODESK and is steady developed (perhaps discontinuously). It serves to data transfer of 2D-data like circles, polygons, hatchings, points and fonts. Curves (Bezier, NURBS) cannot be transferred but they are converted beforehand (usually internally).

At the moment, the version DXF12 is supported. Because the standard was changed

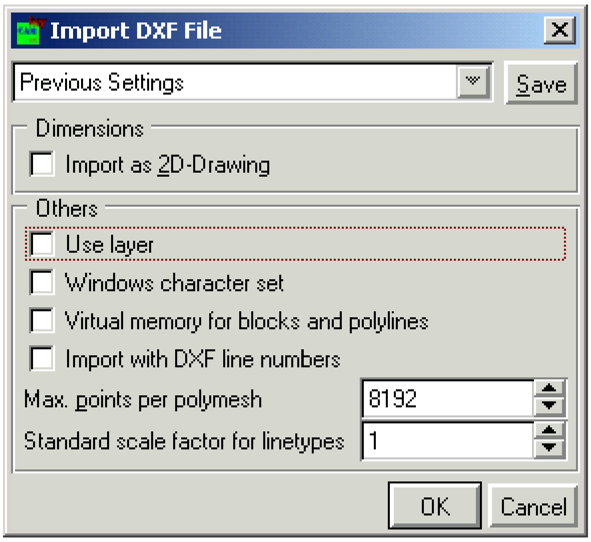
seriously thereafter, data always have to be required in this format.
For converting, special programs are used (export with v2d.exe, import with d2v.exe).
They are stored in the dictionary

\Programs\SchottSysteme\IsyCAM\Tools16

They can also be called directly from the DOS prompt with e.g.: *d2v name1*. With this example the converting of the file *name1.dxf* is started in *name1.vec*. There are various options for export/import which are defined in the control files *v2d.dpa* or *d2v.dpa*. Both files contain usual standard settings. However, they can be adapted with a text editor for special requirements. If editing, please pay attention that the last line will not be a blank line.

Import of DXF-files:

The following menu appears when importing DXF-files Some basic settings can be defined in this dialog. Further import options can be determined in the file *d2v.dpa*.



Import DXF File

Previous Settings

Dimensions
□ Import as 2D-Drawing

Others
□ Use layer
□ Windows character set
□ Virtual memory for blocks and polylines
□ Import with DXF line numbers

Max. points per polymesh: 8192
Standard scale factor for linetypes: 1

<u>List of the DXF-import options:</u>

-2	Translate into a 2D drawing
-3	2D-elements of the DXF-file are translated to 2,5D-elements (t2)
-b	Create sub-objects of blocks
-c <ca>=<cp>[,...]	the DXF-colour <ca> will be to <cp> in isyCAM 2.5
-d#	Decimal digit (2..8, Standard 4)
-e	Display the lines of the DXF-file during converting
-l	Adopt the layer
-m	Virtual memory for blocks and poly-meshes
-o1	Reverse the sloping style of labeling
-o2	Ignore POINT
-p#	Max. # edge points per poly-line resp. poly-meshes
-s<dFont>= <vFont>	The isyCAM-font <vFont> is used for the DXF-font <dFont> as

[:<w>:<h>:<b>],... substitution. If necessary text in this fonts are scaled with <w> in the width and <h> in the height. <b> is the sub length factor

-t <dCode>=<vCode>[,...] The character with the code <dCode> contains the code <vCode> in the PICTURES-drawing

-u <lts> Line factor (LTSCALE), if not defined in the DXF-file (standard 25.0)

-v <var>=<value> The variable <var> contains the value <value>; <var> can be CDFPATH

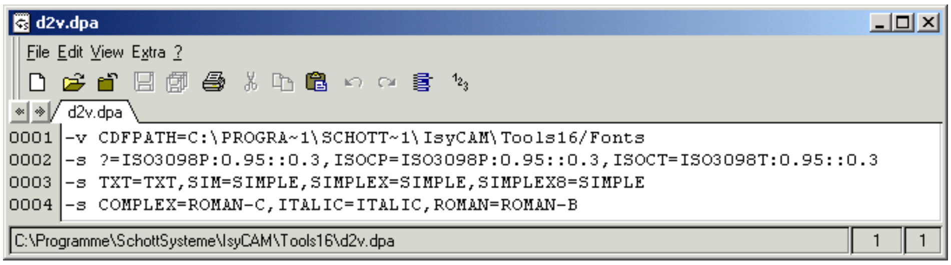
-w <width> Poly lines with less then <width> units width will become single lines

-x The line numbers from DXF-files will be inserted as 'xl ...'

The list of import options will be displayed, if you change in the above-mentioned dictionary and you enter the command **d2v ?** at DOS-prompt.

Error messages/ warnings at DXF-import:

- unknown fonts: Warning *„Font LetterXYZ unknown“*
-> refers that the font *Letter XYZ* defined in DXF-File was replaced. A according option (replacement of all not separate listed fonts, also if available, with *ISO3098P*) is adjusted in the file *d2v.dpa* (cf. picture).



```
0001 -v CDFPATH=C:\PROGRA~1\SCHOTT~1\IsyCAM\Tools16/Fonts
0002 -s ?=ISO3098P:0.95::0.3,ISOCP=ISO3098P:0.95::0.3,ISOCT=ISO3098T:0.95::0.3
0003 -s TXT=TXT,SIM=SIMPLE,SIMPLEX=SIMPLE,SIMPLEX8=SIMPLE
0004 -s COMPLEX=ROMAN-C,ITALIC=ITALIC,ROMAN=ROMAN-B
```

C:\Programme\SchottSysteme\IsyCAM\Tools16\d2v.dpa 1 1

If the script shall be replaced by a different one as ISO3098P, it has to be indicated extra (cf. e.g. line 3: an existing script TXT is used; or line 4: a substitute of *COMPLEX* by *ROMAN*-C). Documents can only be indicated, if a file *.cdf exists in the list

\Programs\SchottSysteme\IsyCAM\Tools16\Fonts

and a file *.pffexist* in the dictionary

\Programs\SchottSysteme\IsyCAM\Fonts

- Message: „*unknown obt type VIEWPORT"*
→ area views and qualities of view windows are defined in this part under AutoCAD; the loss of geometry elements isn't known, certain views can be at another place after converting

- Message: *"unknown group name xyz"*
→ points out that the data of AutoCAD13 or higher were exported; from release 13 the DXF version was changed non compatible to earlier versions and at the moment, they cannot be converted yet. Solution: The producer prepares the data in the form of DXF12.

- Message *„unexpected file end“*
-> *please check*, if the DXF-file is closed with *EOF* (last line, if necessary complete)

- General: look at faulty reported lines in an editor and delete perhaps faulty areas and blank lines at converting

STL/SLA (*.stl, *.sla)

STL (Standard Transformation Language) has asserted itself as exchange format for procedures of Rapid Prototyping such as Stereo Lithographie (SLA) or Lasersintern.

Because information is lost in the transformation, STL rarely plays a role in the data exchange between CAx systems (although many programs offer this interface).
The data are mainly not manipulable or only with very big effort. Since they are immediately useable, sometime STL is the only available "less-than-ideal solution" if using a high dissolution.

Graphic formats (e.g./BMP / JPG/PNG/TIF):

Graphic formats can be differentiated on principle in vector-oriented or pixel-oriented formats (bitmaps, raster pictures).

While a projection instruction for the describing geometry exists in a vector system, in a pixel-oriented system a line consists e. g. of a finite number of points. Point size and number are dependent on the dissolving.

The graphic formats are suitable for different purposes according to these qualities.

Direct import of geometry data:

With help of the interfaces AI, EPS and HPGL (version HPGL1) geometry data like lines and curves can be assumed out of vector-oriented systems.

Vectorisation of bitmaps:

Bitmaps with formats BMP and TIF can get so finished that vector oriented geometry data results again (e.g. after scanning). This method is named vectorisation.

Insert graphics in a drawing:

At isyCAM 2.5 it is possible to insert graphics in a drawing (as a rectangular picture). Type and number of formats are dependent on graphic filters installed on the PC.

Entering the command *ipict* (or click on the corresponding button " load picture into windows ... " in the menu "DIVERSE") the list of the available filters becomes visible. The filters are installed by other programs such as Microsoft Office programs (Winword etc.). But you must start the "user-defined set-up" and select the filters one by one.

Since the heights-/ sides relationship are not changed when inserting a graphic, you should select the edge points numerically if the size is known or you should use the options for width W or height H:

ipict-w100 > the graphic is inserted with a width of 100 GDU (mm).

File Export

Overview: Save in foreign formats

Export in foreign formats:

To see the list with the available formats on the respective PC, please click in menu "File" in the menu bar of isyCAM2.5 (light) :

File > Save as ...

This will open a window "Save file" and in the line File type (bottom line) you can scroll and select the desired format.
The following formats can be exported in isyCAM 2.5 (light):

Name	**Extension**	**Comment**
IsyCAM – drawings	*.vec, *.vcz	Standard formats of isyCAM 2.5
STL – data	*.stl,*.sla	Area interface (triangle-meshes),
Drawing Exchange Format	*.dxf	DXF it the standard format, which was defined by the producer of the program AutoCAD
Adobe Illustrator Files	*.ai	Import of contours
TIF, JPG, BMP, WMF, PNG, EMF	*.tif, *.jpg, *.bmp, *.wmf, *.png, *.emf	Common geometry-/picture output as graphic (bitmaps also useable for shading-pictures)

Description, ranges of application and debug of several formats

DXF-Export: Bezier curves are changed automatically in polygons, NURBS curves have to be manually converted before export. Some different formats are not supported, partly each CAx system has also formats of its own. All formats have to be replaced before export (manual in CAD) ore during export (option in v2d.dpa).

List of DXF-export options:

-a <tol>	Tolerance for transfer of Bezier curves and ellipses in poly lines
-b	A binary DXF-file is generated.
-d#	Number of decimal digits (S-a <tol> standard 6)
-c <cp>=<ca>[,...]	The isyCAM -colour <cp> will be to <ca> im DXF
-f#	Value for the variable FLATLAND (0 or 1)
-h <tol>	Tolerance for transfer of hatching-border curves, negative values are relative to hatching size
-l <LT>= <NAME>(<DEF>)	Assignment of line types. <LT> it the line type under isyCAM 2.5 and <NAME> the relevant name in the DXF-file. A definition can be specified in bracket.
-o	Combined objects are not spent as block
-s <vFont>=<dFont>[,...]	The isyCAM 2.5-font <vFont> is assigned to the font <dFont> in the DXF-file.
-t <vCode> =<dCode>[,...]	The code <dCode> is assigned to the character with the code <vCode>
-v <var>=<value>	The variable <var> get the value <value>, <var> can be CDFPATH, VECPATH or HTCHFILE
-x1	Colours are spent for each element

The list of import options will be displayed, if you change in the above-mentioned dictionary and you enter the command **d2v ?** at DOS-prompt.

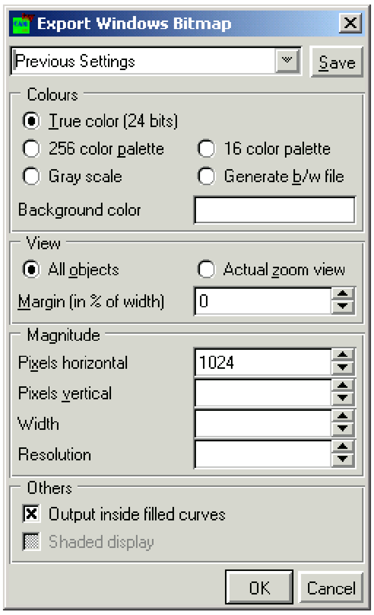
Error messages at DXF-export:

- Font: *v2d: no AutoCAD-Shape defined for 'LetterXYZ'*

The font has to be changed in the drawing or alternatively replaced by an entry in the file v2d.dpa (as much as possible with a font, which is known by the receiver).

Export of graphics:

For documentations, technical illustrations and presentations it is possible to distribute drawings or shading pictures so, that they can be reprocessed in a graphic or word processing program (comparable with a photo of the current screen view.) The corresponding settings are similarly for several formats and can look as shown below (example TIF).



Export Windows Bitmap

Previous Settings

Colours
● True color (24 bits)
○ 256 color palette
○ 16 color palette
○ Gray scale
○ Generate b/w file
Background color

View
● All objects
○ Actual zoom view
Margin (in % of width): 0

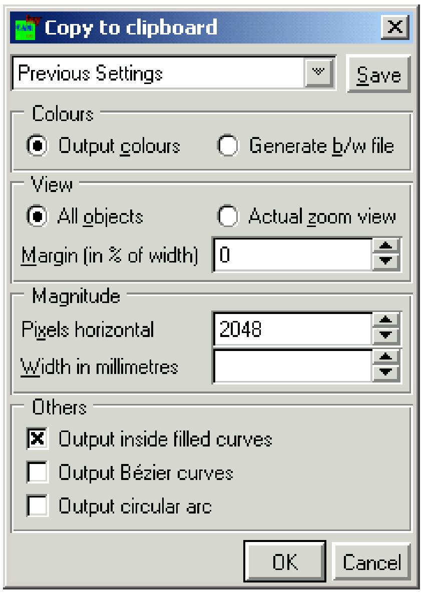
Magnitude
Pixels horizontal: 1024
Pixels vertical:
Width:
Resolution:

Others
☒ Output inside filled curves
□ Shaded display

BMP, JPG, PNG, TGA and TIF are available as bitmap output formats . Vector-oriented formats are EMF and WMF. With these formats the output of a shaded view is not possible.

Copy the drawing in the clipboard:

Another possibility to export isyCAM drawings in another program offers the use of the clipboard. The function "Drawing into the clipboard" can be found in the menu "File". After choosing, the drawing can be copied into the clipboard with the settings shown opposite and then it can be inserted directly into in another program.



Copy to clipboard

Previous Settings

Colours
● Output colours
○ Generate b/w file

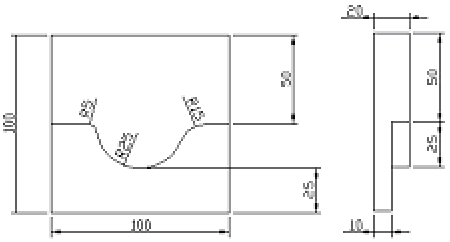
View
● All objects
○ Actual zoom view
Margin (in % of width): 0

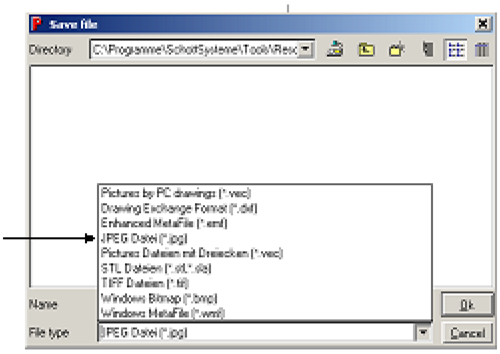
Magnitude
Pixels horizontal: 2048
Width in millimetres:

Others
☒ Output inside filled curves
□ Output Bézier curves
□ Output circular arc

An example shall be shown the export as JPG:

1. Load drawing in isyCAM 2.5
2. File → Save as → File type: JPEG file (*.jpg)




Save file
Pictures by PC drawings (*.vec)
Drawing Exchange Format (*.dxf)
Enhanced MetaFile (*.emf)
JPEG Datei (*.jpg)
Pictures Dateien mit Dreiecken (*.vec)
STL Dateien (*.stl;*.sla)
TIFF Dateien (*.tif)
Windows Bitmap (*.bmp)
Windows MetaFile (*.wmf)
Ok
Cancel

Directory C:\Programme\SchottSysteme\Tools\Resc

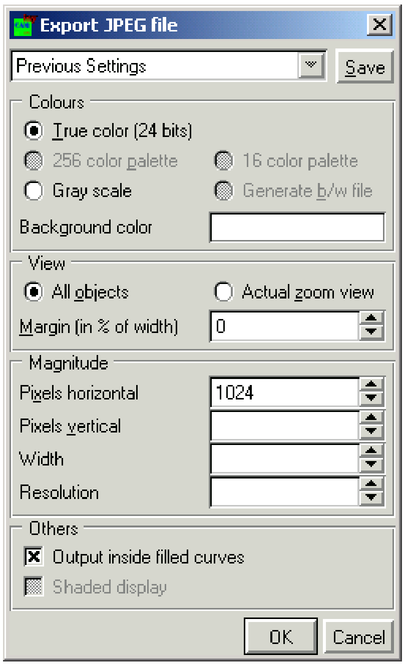
Name

File type JPEG Datei (*.jpg)

Hint:

Only edges and curves are produced as a TIF or JPG picture.
Where? Into the list which was determined in the dialog box**.**

The TIF or JPG picture is produced according to the details in the following dialog box. (Colours, detail, size: Width, dissolving etc.)



Previous Settings

Colours

● True color (24 bits)

○ 256 color palette ○ 16 color palette

○ Gray scale ○ Generate b/w file

Background color

View

● All objects ○ Actual zoom view

Margin (in % of width) 0

Magnitude

Pixels horizontal 1024

Pixels vertical

Width

Resolution

Others

☒ Output inside filled curves

☐ Shaded display

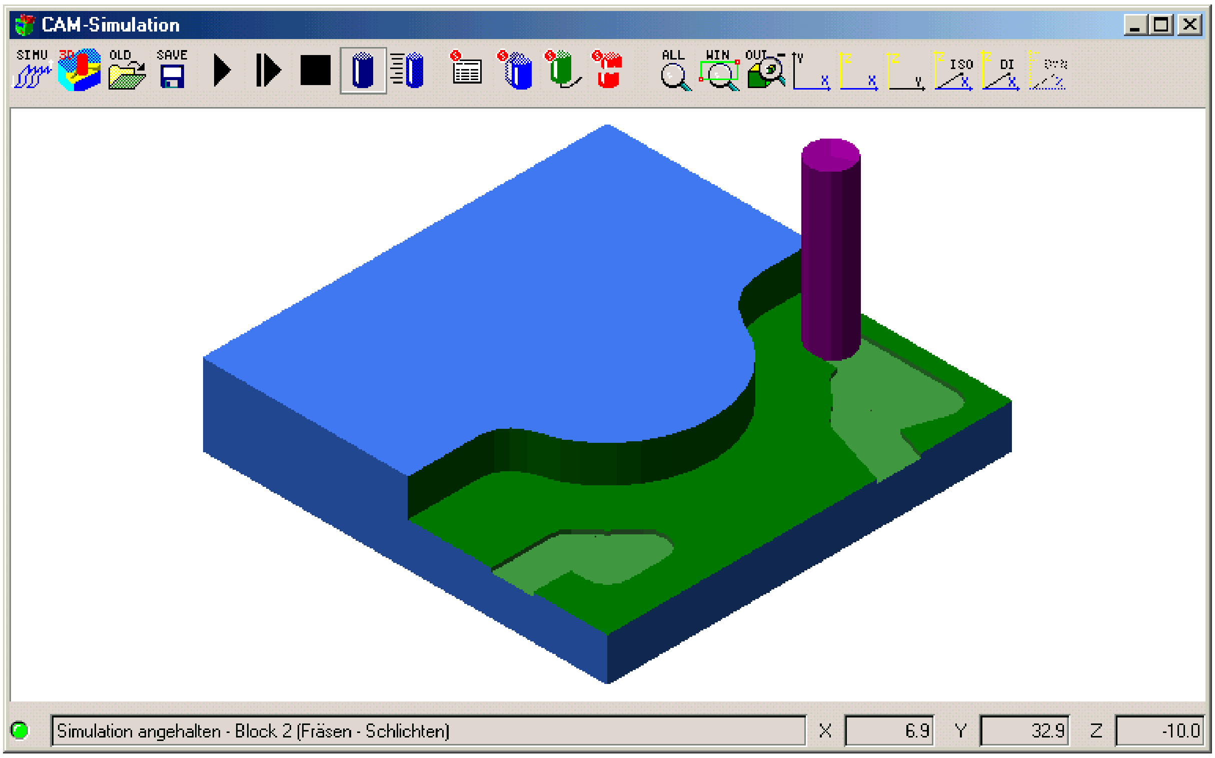
Hint:

A drawing can be saved alternatively also with help of the command line:

vbitmap <name>.tif → Drawing is saved as TIF picture.
vbitmap <name>.jpg → Drawing is saved as JPG picture.
vbitmap <name>.bmp → Drawing is saved as BMP picture.

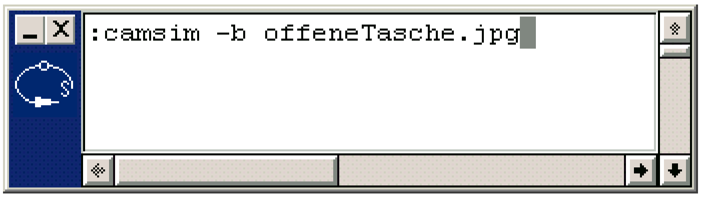
Save of single pictures out of the CAM simulation

1. Start CAM simulation of isyCAD/CAM 2.5 (light) CAM + SIMU and stop at the desired moment



Simulation angehalten - Block 2 (Fräsen - Schlichten)   X 6.9   Y 32.9   Z -10.0

2. Enter in the command line:



```
:camsim -b offeneTasche.jpg
```

`camsim -b <name>.jpg` ➔ The „Simulation“ picture is saved as JPG Picture .
`camsim -b <name>.tif` ➔ The „Simulation“ picture is saved as TIF Picture.

The file is saved in the system directory of isyCAD/CAM 2.5(light).

usually: **“C:\Programs\SchottSysteme\isyCAM\“**

Requirement for saving of the drawings is the corresponding Windows graphic export filter on your PC.

ATTENTION: The background gets independently of the set **background colour white automatically**!

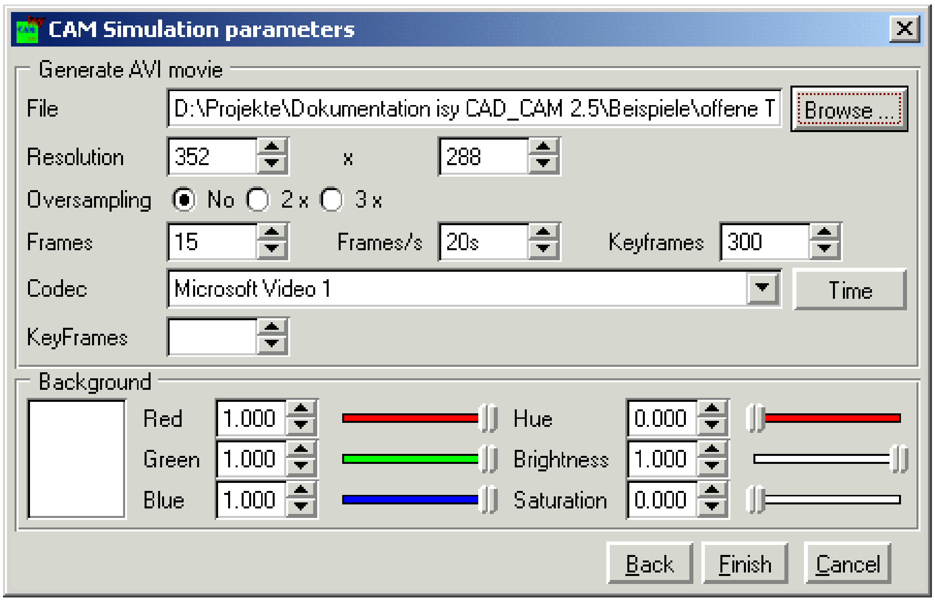
!

Save a CAM simulation as AVI movie (WINDOWS Video clip)

1. Load drawing with CAM blocks

2. Generate AVI (Videoclip) + +

A "regular" simulation is started in which you must select the wanted CAM blocks. Details on the video clip (e.g. dissolving, length of the simulation, number of the pictures etc.) can be made in an additional dialog box displayed on the screen. The window is closed with click on end. The window for the simulation appears now. At first e.g. the view or simulation parameters can be adjusted as in the case of every simulation. With click on the button "Simulation-generate movie file" the simulation starts and the video clip is generated.



CAM Simulation parameters

Generate AVI movie

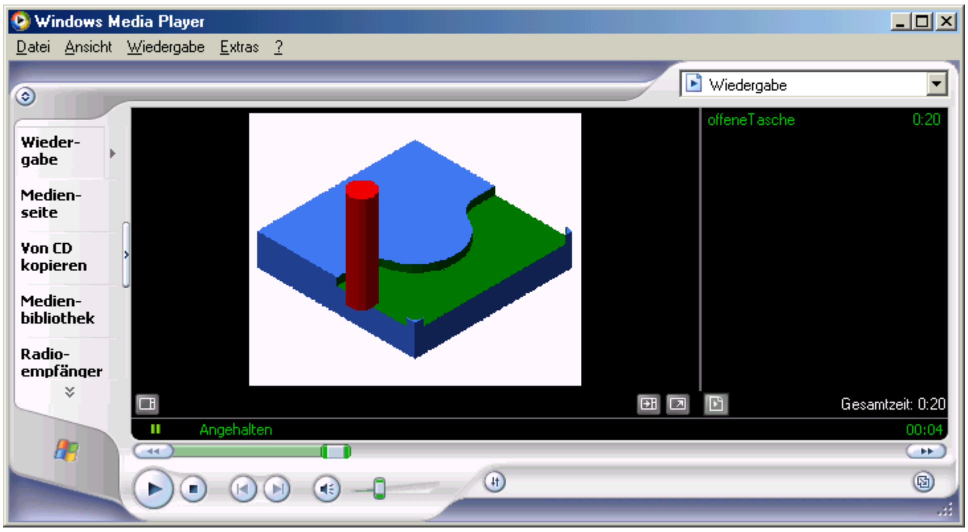
File	D:\Projekte\Dokumentation isy CAD_CAM 2.5\Beispiele\offene T				Browse ...
Resolution	352	x	288		
Oversampling	● No ○ 2 x ○ 3 x				
Frames	15	Frames/s	20s	Keyframes	300
Codec	Microsoft Video 1				Time
KeyFrames					

Background

Red	1.000	Hue	0.000
Green	1.000	Brightness	1.000
Blue	1.000	Saturation	0.000

3. Start AVI (video clip):

The simulation film is generated in the set dictionary as an AVI file. The AVI simulation can play back and start with the play back program (e. g. \Windows\MPLAYER.EXE). To start the saved simulation please click on the relevant AVI file in the "Windows Explorer" (there must be a relevant linking resp. setting that the start can happen automatically).



The simulation saved as AVI can be viewed independently of isyCAD/CAM 2.5 (light). This not very memory-intensive simulation can be sent compressed to other potential buyers (e.g. via Internet).

Index