

---

# KinemaTikZ



A  $\LaTeX$  package for Kinematics

Vitor Santos  
([vitor@ua.pt](mailto:vitor@ua.pt))

Version 1.0  
21 December 2021

---

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	About . . . . .	3
1.2	License . . . . .	3
1.3	Loading the package . . . . .	3
1.4	Purpose of the package . . . . .	3
1.5	ISO symbols and disclaimer . . . . .	4
<b>2</b>	<b>Description</b>	<b>4</b>
2.1	Principle and basics . . . . .	4
2.2	Types of elements . . . . .	7
2.3	Types of kinematic chains . . . . .	8
2.4	Parameters and options for pics . . . . .	9
2.5	Design styles in KinemaTikZ . . . . .	9
2.6	Connecting lines and components . . . . .	10
2.7	Scaling pic objects . . . . .	11
2.8	Placement of components . . . . .	12
2.9	Examples of TikZ-like parameterization . . . . .	15
<b>3</b>	<b>Creating kinematic chains</b>	<b>16</b>
3.1	Examples of planar mechanisms . . . . .	17
3.2	Examples of non planar (spatial) mechanisms . . . . .	19
3.3	Other examples of kinematic chains . . . . .	20
<b>4</b>	<b>Reference listing of elements available in the package</b>	<b>21</b>
4.1	The <i>cylindrical pair</i> . . . . .	21
4.2	The <i>frame</i> . . . . .	22
4.3	The <i>frame 3D</i> . . . . .	22
4.4	The <i>frame dual pivot slide</i> . . . . .	23
4.5	The <i>frame pivot flat</i> . . . . .	23
4.6	The <i>frame pivot trapezium</i> . . . . .	24
4.7	The <i>frame pivot triangle</i> . . . . .	24
4.8	The <i>frame pivot rounded</i> . . . . .	25
4.9	The <i>gripper</i> . . . . .	26
4.10	The <i>helical pair</i> . . . . .	27
4.11	The <i>linear 3D</i> . . . . .	28
4.12	The <i>linear joint bar</i> . . . . .	30
4.13	The <i>linear piston</i> . . . . .	31
4.14	The <i>link bar generic</i> . . . . .	31
4.15	The <i>link polygon</i> . . . . .	33
4.16	The <i>planar contact pair</i> . . . . .	34

4.17	The <i>prismatic pair</i> . . . . .	34
4.18	The <i>revolute pair planar</i> . . . . .	35
4.19	The <i>revolute pair spatial</i> . . . . .	36
4.20	The <i>rotating cam plate</i> . . . . .	36
4.21	The <i>cam follower</i> . . . . .	37
4.22	The <i>rotational 3D H</i> . . . . .	38
4.23	The <i>rotational 3D P</i> . . . . .	39
4.24	The <i>rotational 3D V</i> . . . . .	41
4.25	The <i>rotational colinear</i> . . . . .	43
4.26	The <i>spherical pair</i> . . . . .	44
<b>5</b>	<b>List of available symbols</b>	<b>45</b>

# 1 Introduction

## 1.1 About

KinemaTikZ was developed mainly to support the production of a book on robotics and its associated course.

## 1.2 License

Copyright © 2021 by Vitor Santos

This package is author-maintained. Permission is granted to copy, distribute and/or modify this software under the terms of the  $\TeX$  Project Public License, version 1.3.1, or the GNU Public License. This software is provided ‘as is’, without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

## 1.3 Loading the package

Loading the package follows the usual procedures in  $\TeX$ :

```
\usepackage{kinematikz}
```

The package itself loads the `tikz` package along with several associated libraries such as the following: `backgrounds`, `angles`, `arrows`, `calc`, `decorations.pathmorphing`, `decorations.pathreplacing`, `fit`, `hobby`, `intersections`, `math`, `matrix`, `patterns`, `positioning`, `scopes`, `shadows`, `shapes`, `shapes.multipart`, among other.

## 1.4 Purpose of the package

This package provides functionalities to draw kinematic diagrams for mechanisms using dedicate symbols (some ISO standard and others). It does not perform kinematics solving (either direct or inverse).

Moreover, the intention is not to represent realistic mechanical drawings of mechanisms and robots, but only to represent 2D and 3D kinematic chains.

The package provides links, joints and other symbols, mostly in the form of TikZ `\pic` objects, which are some kind of very special nodes with custom code. Placement of these pics follows the usual TikZ processes and, hopefully, all TikZ facilities and huge potentialities were not affected, despite some complex interactions that can occur.

These pics can be placed in the canvas using the usual TikZ approaches, either by a central point for joints, and start and end points for some links, as described later.

All these nodes are actually internal TikZ coordinates defined to ease the process of placement and establishing links, arbitrarily or possibly in arrangements as described further.

Concerning some terms used throughout this document, when applied to joints and links, this document considers synonyms the following:

- revolute = rotational
- linear = prismatic

Another common term is *kinematic pair*, which is a formal denomination of two rigid bodies in contact in such a way that their relative motion is mutually constrained. The two bodies are named links, and the connection between them, the joint. The nature of the joint, hence the type and name of the kinematic pair, depends on the actual contact between the bodies: through a surface, a line, or a point.

## 1.5 ISO symbols and disclaimer

Kinematic diagrams are regulated by the ISO 3952 standard named "Kinematic diagrams — Graphical symbols" with the most well known versions from 1981 and 1995. That norm is divided in 3 main parts plus a fourth part with examples of miscellaneous mechanisms. These parts are divided in the following sub-parts:

### Part 1 (ISO 3952-1)

- Motion of links of mechanisms
- Kinematic pairs
- Links and connection of their components
- N-bar linkages and their components

### Part 2 (ISO 3952-2)

- Friction and gear mechanisms
- Cam mechanisms

### Part 3 (ISO 3952-3)

- Geneva and ratchet mechanisms
- Couplings and breaks

### Part 4 (ISO 3952-4)

- Miscellaneous mechanisms and their components

This package does not cover all ISO symbols from the 3952 standard and focuses mainly on Part 1, especially on its second section, the **kinematic pairs**, but elements from other parts are included, and coverage of remainder symbols and notations are expected to evolve in future editions of the package. That is the case for many elements of the ISO standard that are not yet explicitly addressed with the standard names, but can easily be reproduced by commands from TikZ, such as for example the ISO expression "trace of motion" which is actually a line (straight or curved) easily sketchable by a "`\draw [thick] (P1) -- (P2);`" or similar commands. Other ISO items are more elaborate, like for example "one-sided motion with instantaneous stop", but they are perhaps not relevant for the scope for which this package was conceived which is to draw kinematic chains. Also, not all variants of symbols (dubbed *Permissible symbol* in the standard) are covered yet.

On the other hand, as mentioned, the package includes other non-ISO symbols that are common in the literature.

## 2 Description

### 2.1 Principle and basics

Mechanisms are made of links and joints arranged and connected among them.

The package allows the plot of usual TikZ constructions (lines, nodes, coordinates, etc.), and specific objects representing links and joints properly drawn, using the `\pic` construction.

One of the simplest implementations of an element of a mechanism using a `\pic` construction is something like the following to represent a fixed triangular frame (base) with a pivot point:



```
\begin{tikzpicture}
\pic {frame pivot triangle};
\end{tikzpicture}
```

Alternatively, although recommend only for simpler graphics, the same drawing can also be done inline with the command `\tikz \pic {frame pivot triangle};` that yields the same graphic

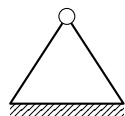
inline, as given by this example:  along the flow of the text.

Naming of the `\pic` elements can seem too verbose in some cases, but that was done either to match ISO definitions or to ensure clarity of what element was involved. But, users can create their own aliases, as  $\LaTeX$  is so versatile for. For example, if users find `frame pivot triangle` too verbose, an alias like `"\frameT"` could be created in some global scope and then used like this:



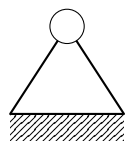
```
\begin{tikzpicture}
\def\frameT{frame pivot triangle}
\pic {\frameT};
\end{tikzpicture}
```

Many elements possess parameters. An example of a parameter that changes the triangular frame side width for 1.5 cm (any  $\TeX$  measurement is acceptable) is the following:



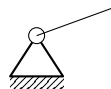
```
\begin{tikzpicture}
\pic {frame pivot triangle=1.5cm};
\end{tikzpicture}
```

The previous case is different from a scale change that results in slightly different representation as shown next; the issues of scaling upon `\pics` will be addressed later.



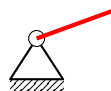
```
\begin{tikzpicture} %[scale=2.15,transform shape]
\pic[scale=2.15] {frame pivot triangle};
\end{tikzpicture}
```

An example of a most simple mechanism is a bar (link) pivoting around a fixed base. For a simple bar hooked on a triangular base and extend with length 30pt in the direction of  $20^\circ$ , the following can be done:



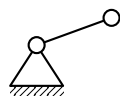
```
\begin{tikzpicture}
\pic (MyBase A) {frame pivot triangle};
\draw (MyBase A-center) -- (20:30pt);
\end{tikzpicture}
```

The first operation places the triangular base in default point (0,0) because it is omitted. The element is named `MyBase A` (but could be any name allowed by `TikZ`). The second command draws a simple line between the center `MyBase A` and some point, here expressed as the absolute radial point `(20:30pt)`, that is at an angle of  $20^\circ$  and a radius of 30pt. Of course, this and all parameters usable in `TikZ` can be added to most operations, as in this variation:



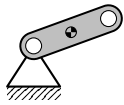
```
\begin{tikzpicture}
\pic (MyBase A) at (0,0) {frame pivot triangle};
\draw [ultra thick, red] (MyBase A-center) -- (20:30pt);
\end{tikzpicture}
```

However, this package offers multiple graphic possibilities for links and joints, like this more accurate link:



```
\begin{tikzpicture}
\pic (MyBase A) {frame pivot triangle};
\pic at (MyBase A-center) {revolute pair planar={20:30pt}};
\end{tikzpicture}
```

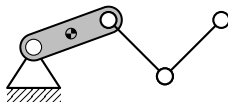
Or this even more expressive bar link that is configurable, as shown further:



```
\begin{tikzpicture}
\pic (MyBase A)          {frame pivot triangle};
\pic at (MyBase A-center) {link bar generic={20:30pt}};
\end{tikzpicture}
```

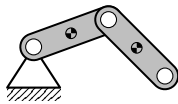
In these last two variants, (especially in the second one) the link is drawn with a richer image, and a simple line drawing is replaced by a second `\pic` object defined by two points: the same start as before, but now the `\pic` object (`revolute pair planar` and the `link bar generic`) has an argument (`={...}`) that contains the end point of the bar. Naturally, the object adjusts its size automatically. The braces involving the arguments are optional and only in very special situations are they expected to be needed.

The methodology can be further extended at will and more links can be concatenated and mixed, even if some combinations may be not very graphically consistent!



```
\begin{tikzpicture}
\pic (MyB)          {frame pivot triangle};
\pic (MyL1) at (MyB-center) {link bar generic=20:30pt};
\pic (MyL2) at (MyL1-end)   {revolute pair planar=-45:30pt};
\pic (MyL3) at (MyL2-end)   {revolute pair planar=45:30pt};
\end{tikzpicture}
```

Coordinates passed as options to `\pic` objects can emulate the effect as being either in the global frame or in the local frame of the starting point of the link. To use global coordinates it is enough to create the point (using `\coordinate` command, for example) and use it as argument; to express the point in relative local coordinates, simply use the full extension coordinates. Follow the two examples of this situation:



```
%Local: -45:30pt is relative to the \pic starting point
\begin{tikzpicture}
\pic (MyB)          {frame pivot triangle};
\pic (MyLX) at (MyB-center) {link bar generic=20:30pt};
\pic (MyLZ) at (MyLX-end)   {link bar generic=-45:30pt};
\end{tikzpicture}
```



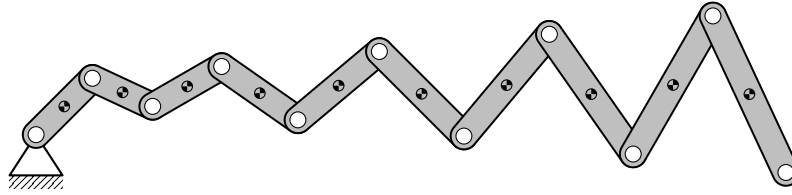
```
%Global: point P2 (with -45:30pt) reports to calling tikzpicture
\begin{tikzpicture}
\coordinate (P2) at (-45:30pt);
\pic (MyB)          {frame pivot triangle};
\pic (MyLY) at (MyB-center) {link bar generic=20:30pt};
\pic (MyLZ) at (MyLY-end)   {link bar generic=P2};
\end{tikzpicture}
```

Actually, when passing the coordinate name, such as the `P2` in the example, it is the token `P2` that is passed without any meaning and it is expanded inside the `\pic` code (with an operation similar to `\coordinate (-end) at (#1);`, where `#1` is the passed argument, the token `P2`, in this case), where it inherits the meaning created in the context of the `tikzpicture` where it was defined.

NOTE: assigning a name to the link when using a predefined point as argument for the `\pic` gives an error in version 3.0 of TikZ; it needs to use explicit coordinates. This is solved in version 3.1.5 of TikZ!

The option `anchor`, if passed to a `KinemaTikZ \pic` has no effect because the `pic`, by default, attaches internally to the current point, that is `(0,0)`, although some `pics` can attach to other points.

Some elaborate constructions can be made with a configurable code as shown in next example. The `\tikzmath` context was used to perform calculations and to better define  $\TeX$  macros for dynamic names, because it is easier and more versatile than performing the automation inside the `\pic` description! Other remarks include the usage of a  $\TeX$  dimension (`\mylen`) to use it appropriately when required in the argument of the `\pic` instruction.



```

\centering
\begin{tikzpicture}
\pic (MyBase A)                {frame pivot triangle};
\pic (MyNewLink 1) at (MyBase A-center) {link bar generic={45:30pt}};

\newdimen\mylen %create a TeX dimension to use below
\foreach \x in {1,...,9}
{
  \tikzmath
  {
    integer \nn,\cont;
    \nn=(-1)^\x*(20+5*\x);
    \mylen=20pt+\x*5pt;
    \cont=\x+1;
  }
  \pic (MyNewLink \cont) at (MyNewLink \x-end) {link bar generic={\nn:\the\mylen}};
}
\end{tikzpicture}

```



## 2.2 Types of elements





As mentioned, drawing with KinemaTikZ can use the powerful techniques and tools provided by TikZ and related libraries and packages; beyond that, it uses three types of specific objects that can be combined following the logic of kinematic chains.

Besides some accessory objects that will appear throughout this text (like *center of mass*, *grippers*, etc.), the main building objects are of the following types:

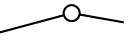

- frames
- links
- joints


**Frames** represent the fixed parts of a structure, usually grounds, walls of other fixing parts; there are cases where the frames include a pivoting point to further attach links. Several geometries

exist, like the **frame pivot triangle**  presented earlier, or the simple **frame** , or other to be presented later. By convention, all frames are supposed to be attached together, even if it does not appear on the diagrams; but that is irrelevant anyway do draw with the KinemaTikZ package.

**Links** are the solid elements (bodies) that connect joints. The **link bar generic**  presented earlier is an example of link that has two joint attachments, and can even have additional ornaments like a COM, or **center of mass** symbol () or crosshairs in the joints: . This link is also configurable to the particular case of a terminal link that does not connect to any joint in the tip, like this: . But, links can be simple lines (straight or curved according to the ISO norm) or, as shown ahead, polygonal blocks with several joint attachment points.



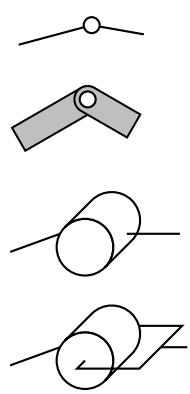
**Joints** are representations of the contact established between two bodies (links) and can have 1 or more degrees of freedom. This set of one joint plus its two attached links is called kinematic pair. The simplest representation of a kinematic pair is the **revolute pair planar**  or, if we want the absolute simplest representation of the rotational joint without links:  $\bigcirc$ . There are other joint representations, both in ISO and non ISO formats, for planar or spatial (3D) chains. In most of those cases, those links are made of simple lines since the important is to emphasize the nature of the joint because of the wealth of information that it contains, such is the case of this rotational joint symbol (non ISO) with the axis along the horizontal in the page: .

**Other elements** include accessories to attach to links and joints, or some graphic facilities. That includes for example the above mentioned **center of mass** symbol () , or coordinate axis and joint variables for further illustrations on the kinematic chain.

## 2.3 Types of kinematic chains

There are two main philosophies when representing kinematic chains: link-centered and joint-centered. In the link-centered approach, the link is defined by its extremes (normally two points, but can be otherwise). The dimensions and geometry are important; representation may be simple lines or more elaborate objects, as mentioned in the beginning of this document. In this approach, the joints emerge naturally from the interconnection of the links.

On the other hand, in the joint-centered approach, what is central is the type of joint that has normally a more elaborate representation, and the links are usually symbolic. This representation is more conceptual than realistic, although link dimensions can still be respected and properly illustrated. Next image illustrates essentially the same situation of a rotational joint defined by two links, ranging from a simple **revolute pair planar** passing by two **link bar generic** (that define the same revolute a joint), up to two common representations (that use **rotational 3D P** symbols) stressing on the joint itself rather than on the links, although they are both present.



**Four representations of the same case of a joint and two links**

```
\begin{tikzpicture}[knLinkStyle]
\pic {revolute pair planar};
\pic at (L1-out) {link bar generic=30:30pt/0/0/1};
\pic at (L1-out) {link bar generic=-30:20pt/0/1/0};
}
\pic at (J1) {rotational 3D P=0};
\draw (J1-west) -- ++(-160:20pt) (J1-end) -- ++(20pt,0);
}
\pic at (J2) {rotational 3D P=1};
\draw (J2-in) -- ++(-160:20pt) (J2-end) -- ++(10pt,0);
}
\end{tikzpicture}
```

Moreover, it is possible to mix representations and have both realistic links with specific joint representations in a same schematics, although the result may generate confusion and difficulty in the reading.

Independently of that, for the scope of this KinemaTikZ package, the constructions for the kinematic chains can be considered of three main kinds:

1. Pure planar chains
2. Spatial chains using 2D symbols
3. Spatial chains using 3D orthogonal perspective symbols

Types 1 and 2 are mainly associated to the ISO norm (but symbols outside the norm are also common), and type 3 are more common from non ISO conforming literature, but can give better insight of the actual 3D mechanism, as shown in the last two examples of the previous figure.

Some literature also include non orthogonal perspective (e.g. isometric) symbols, but representing kinematic chains with them would require true 3D geometry which is not fully developed in TikZ; although some limited possibilities exist, a full 3D solution would be better managed with languages such as Asymptote.

## 2.4 Parameters and options for pics

As mentioned, using this package follows the TikZ philosophy where `\pics` are central. To keep it simple, the created `\pics` may support specific parameters or arguments for configuration, but arguments are optional in which case a default drawing is created.

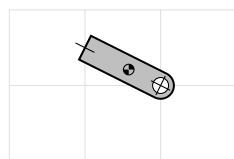
The general syntax is:

```
\pic [tikz options] (pic name) at (starting point) {pic type={arg1/arg2/...}};
```

The arguments `arg1`, `arg2`, etc. may be omitted totally or partially. Arguments are separated by "/" and in some cases empty arguments are ignored, and in other cases empty arguments are replaced by defaults.

The order of the general syntax just presented is flexible as stated by the TikZ manual; the separation of arguments with a "/" is specific of KinemaTikZ and braces around the list of arguments is most of the times optional since the argument separator is the "/"; this was used because other usual separators such as `,` `.` `:` `;`, among others, are reserved or have special usages either in  $\text{\LaTeX}$  or TikZ. This argument separation is compelling because the package uses the valuable `listofitems` package to parse the arguments.

If any argument is expected to be a point, created by TikZ `\node`, `\coordinate` or else, no parenthesis ( ) are to be used to delimit it, only the point in Cartesian or polar coordinates. For example: the command `\pic (my Link 1) at (2,1) {link bar generic={-1,0.5/1//0/1}};` places a **link bar generic** starting at point (2,1), names it `my Link 1`, has the end point at (-1,0.5) (relatively to the starting point of the object), has a COM drawn on the link body center, uses default option for pivot in **first** extremity, does not include a pivot in **second** extremity, and has **cross hairs** in extremities/pivots. Follows the result with grid lines added for clarity:



```
\begin{tikzpicture}
\draw [help lines,gray!25] (0,0) grid (3,2);
\pic (my Link 1) at (2,1) {link bar generic={-1,0.5/1//0/1}};
\end{tikzpicture}
```

For some more complex `\pic`, namely those with variable number of parameters like the **link polygon**, it is possible to predefine macros used inside the `\pic` code in case they are defined.

## 2.5 Design styles in KinemaTikZ

For enhanced flexibility, the design of components uses custom styles that can be overridden or appended. They affect essentially line width, line endings (cap) and line joining.

The `knLinkStyle` style is used to draw most objects made of lines. It is the recommended base style to draw other lines in the schematics for uniformity, but user is free to use other properties, or even modify locally, or globally the style.


The `knLineCap` style is used to enforce that some lines have alternative endings, namely because of the issues presented in section 2.6.


The `knJointStyle` is used to draw parts of joints that are not connecting lines or related. By default, the style is similar to `knLinkStyle` (all parts of the kinematic chain have the same line weight), but that can be customized locally or globally any time in the code.


### Some drawing styles in KinemaTikZ

```
\tikzset
{
  knLinkStyle/.style={% main style for links
    thick,
    miter limit=5, %avoid long sharp corners in special edges
  },
  knLineCap/.style={ %extra option for line cap in some drawings
    line cap=round,
  },
  knJointStyle/.style={ %Style for drawing joint parts
    knLinkStyle,
    line join=miter, %Reinforce the default
  },
}
```

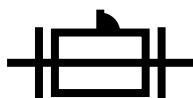
## 2.6 Connecting lines and components

For larger scales, especially for larger line thickness, drawing the elements separately can originate discontinuities in lines or connections like this example: . This is most noticeable when connecting lines (links) with other elements of a kinematic chain that terminate in open lines. For that purpose, those elements were drawn with the option `line cap=round` for the terminals prone

to have further connections like in this (exaggerated) example: . If however that is not desired, the option `line cap=flat` can be used to override that effect in the style `knLineCap`

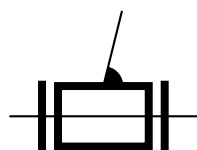
resulting in the following design: .

In practice, that operation can be used by appending or redefining the styles defined by KinemaTikZ. This last version can be obtained with the following code:



```
\begin{tikzpicture}
\tikzset{
  knLinkStyle/.append style={line width=3pt}, %style for links
  knLineCap/.style={line cap=flat}, %style for line caps
}
\pic[scale=2] {revolute pair spatial=90:10pt/0/0};
\end{tikzpicture}
```

Some joint symbols have the possibility of distinguishing the line width of the "body" part of the joint and the "connecting" parts of the joints (link parts, actually). The latest is regulated by the `knLinkStyle` style, as just seen. The first is regulated by the `knJointStyle` style. By default, `knJointStyle` is similar to `knLinkStyle`, but it can be changed to stress the differences:



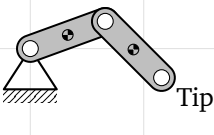
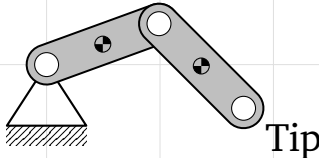
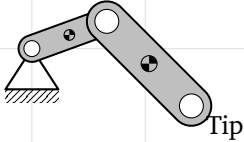
```
\begin{tikzpicture}
\tikzset{knJointStyle/.append style={line width=3pt}}
\pic[scale=2] {revolute pair spatial=80:20pt/0/0};
\end{tikzpicture}
```

## 2.7 Scaling pic objects

Most TikZ objects can be scaled locally by the `scale` option. But when there are many objects to be scaled, it is more practical to use a global `scale` option when declaring the `tikzpicture`. However, as it is intrinsic to TikZ, the `scale` option issued as a global action in TikZ pictures does not affect the size of pics or nodes, unless the option `transform shape` is used.

But using `transform shape` changes also the size of text in nodes. To suppress that effect locally, the node can have the option `transform shape=false`: this option has been dubbed TSF in the KinemaTikZ package for easier and faster typing since it is frequently used. Follows an example illustrating that situation with three cases all under the same global scale factor but with different outcomes: 1) no effect at all; 2) affecting everything; 3) affecting only the non-TSF protected.

Issues when scaling pics

1) scale has no effect	2) scale affects everything	3) scale does not affect the TSF protected
		

---

```

\centering
\begin{tikzpicture}[scale=1.5] %Global scale
\tikzset{myT/.style={anchor=north west,xshift=2pt}};
\draw [help lines,gray!25,step=1] (-1,-1) grid ++(9,2);
%--Example on the left
\pic (MyBaseA) {frame pivot triangle};
\pic (L1) at (MyBaseA-center) {link bar generic=20:30pt};
\pic (L2) at (L1-end) {link bar generic=-45:30pt} node [at=(L2-end),myT] {Tip};
%--Example in the middle
{[transform shape,xshift=3cm] %a new scope
\pic (MyBaseA) {frame pivot triangle};
\pic (L1) at (MyBaseA-center) {link bar generic=20:30pt};
\pic (L2) at (L1-end) {link bar generic=-45:30pt} node [at=(L2-end),myT] {Tip};
}
%--Example on the right
{[transform shape,xshift=6cm] %another scope
\pic [TSF](MyBaseA) {frame pivot triangle};
\pic [TSF](L1) at (MyBaseA-center) {link bar generic=20:30pt};
\pic (L2) at (L1-end) {link bar generic=-45:30pt} node [at=(L2-end),myT,TSF] {Tip};
}
%--The caption text for each example
{[inner sep=0,anchor=north west,font=\footnotesize] %scope for descriptions
\node at (0,0.85) {1) scale has no effect};
\node at (3,0.85) {2) scale affects everything};
\node at (6,0.85) [align=left] {3) scale does not affect\\the TSF protected};
}
\end{tikzpicture}

```

All `link bar generic` pics were called with one argument only, which is the end point; all the remainder arguments were assumed the default value by the `\pic` code.

As mentioned, independent control on each pic can be ensured by using locally the desired scale factor option.

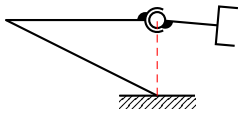
## 2.8 Placement of components

Link objects are placed anchored on their **start** and **end** points. The start point is specified in the `\pic` command syntax by a `at` instruction or equivalent, or by default at the  $(0,0)$  coordinate in the current `\tikzpicture`, and the end point (or points, for multi-end links) are specified in the arguments, or a default value for illustration purposes in case no end point is passed as argument.

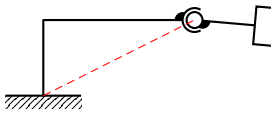
Frames and Joints are placed anchored mostly on their center (`-center`), or rotating pivot, when applicable. In some elements it is possible to add an additional argument to force the placement anchor at the `-in` node instead of the default `-center`; check in each case. In some of those resulting cases using `knLineCap` line style may be advised when connecting elements.

The difference between using `-in` or `-center` as the anchor point can be understood from the following example where defaults, or a specific parameter (in this case the parameter number 5 for a `spherical pair`), controls which is the case or anchoring.

Different anchor points in the relative placement of the spherical pair



Default anchor at **-center** node



Anchor at **-in** node

---

```

\centering
\begin{tikzpicture}
  %anchor at -center (default)
  \pic (B) {frame};
  \pic (C) [above of=B-out] {spherical pair=-5//2cm};
  \pic [scale=0.5,rotate=-5] at (C-out) {gripper};
  \draw [knLinkStyle,knLineCap] (B-out) -- (C-in);
  \node [below of=B-in] {Default anchor at \textbf{-center} node};
  \draw [thin, red, densely dashed] (B-out) -- (C-center);
  {[xshift=5cm] %anchor at -in (argument5=1 for a spherical pair)
  \pic (B) {frame};
  \pic (C) [above of=B-out] {spherical pair=-5//2cm//1};
  \pic [scale=0.5,rotate=-5] at (C-out) {gripper};
  \draw [knLinkStyle,knLineCap] (B-out) -- (C-in);
  \node [below of=B-in] {Anchor at \textbf{-in} node};
  \draw [thin, red, densely dashed] (B-out) -- (C-center);
  }
\end{tikzpicture}

```

The definition of the placement points for nodes and pics can be absolute numeric or using named coordinates (like `\nodes`). But we can also use relative positions, which is most of the times preferred for more automatic adjusting.

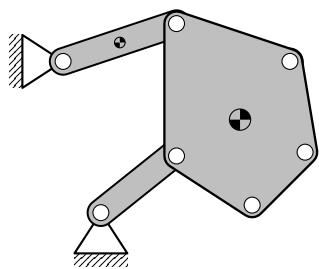
The order of growing flexibility is:

- Absolute point coordinates: e.g. `at (3,4)`
- Named coordinates or nodes: e.g. `at (P1)`
- Relative placement using node names: e.g. `above=of Joint1-center`

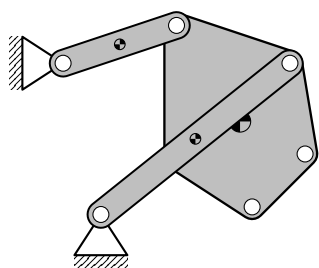
In any case, fine tuning adjustments can be added using the usual TikZ parameters like `xshift=` or `yshift=`, minding however that these instructions operate in each `\pic` reference frame, which may yield unexpected layouts when option `rotate` is also included!

In some cases, when mixing symbols at different scales, improper edge alignment may occur because thinner lines of one object may mismatch lines of an adjacent object, or sometimes a

symbol that is in the foreground should be in the background. The order of placement of the elements can be a solution to overcome that issue. But sometimes it is not possible or convenient to change that order in the code. The solution may be using TikZ layers. Follows an example where a link placed later in the code is sent to the background, which in this case also solves a potential ambiguity due to the options left on the `link polygon` object.

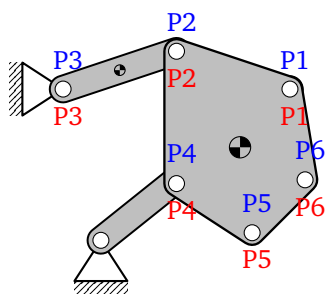


```
\begin{tikzpicture}
\coordinate (P1) at (2.5,2); \coordinate (P2) at (1,2.5);
\coordinate (P3) at (-0.5,2); \coordinate (P4) at (1,0.75);
\coordinate (P5) at (2,0.1); \coordinate (P6) at (2.7,0.8);
\pic (BA) {frame pivot triangle};
\pic[rotate=-90] (BB) at (P3) {frame pivot triangle};
\pic (L1) at (BA-out) {link bar generic=P1};
\pic (L2) at (BB-out) {link bar generic=P2};
%\begin{pgfonlayer}{background}
\pic (L3) at (P1) {link polygon=P2/P4/P5/P6};
%\end{pgfonlayer}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\coordinate (P1) at (2.5,2); \coordinate (P2) at (1,2.5);
\coordinate (P3) at (-0.5,2); \coordinate (P4) at (1,0.75);
\coordinate (P5) at (2,0.1); \coordinate (P6) at (2.7,0.8);
\pic (BA) {frame pivot triangle};
\pic[rotate=-90] (BB) at (P3) {frame pivot triangle};
\pic (L1) at (BA-out) {link bar generic=P1};
\pic (L2) at (BB-out) {link bar generic=P2};
\begin{pgfonlayer}{background}
\pic (L3) at (P1) {link polygon=P2/P4/P5/P6};
\end{pgfonlayer}
\end{tikzpicture}
```

Just for illustration purposes, and to save some typing time, taking advantage of TikZ facilities may be advised, especially when repetitive code seems to be necessary. For instance, in the previous examples, if more actions would be desired for each point (like adding a label), it would be possible to use the `\foreach` construction. Next code shows more than one different usage of it:



```
\begin{tikzpicture}
\def \pnts
{(2.5,2),(1,2.5),(-0.5,2),(1,0.75),(2,0.1),(2.7,0.8)}
\foreach \P [count=\i] in \pnts \coordinate [at=\P] (P\i);
\pic (BA) {frame pivot triangle};
\pic[rotate=-90] (BB) at (P3) {frame pivot triangle};
\pic (L1) at (BA-out) {link bar generic=P1};
\pic (L2) at (BB-out) {link bar generic=P2};
\pic (L3) at (P1) {link polygon=P2/P4/P5/P6};
\foreach \P [count=\i] in \pnts
\node [above=4pt,xshift=2pt,blue] at \P {P\i};
\foreach \i in {1,...,6}
\node [below=3pt,xshift=2pt,red] at (P\i) {P\i};
\end{tikzpicture}
```

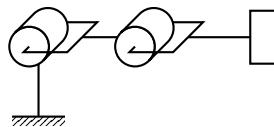
The previous examples show the placement of components at fixed coordinates manually established. However, the kinematic chain used to illustrate is quite oriented for such approach because the dominant elements are links, which are defined by their extremities. That is, the coordinates given are the centers of the joints that are defined automatically by the connections of two (or more) links.

If, instead of links, joints are to be placed first (the other philosophy of describing kinematic chains), and then links are drawn, other placement approaches emerge as advantageous.

To illustrate the absolute and relative techniques to place joints, along with an additional technique that uses a TikZ matrix, a same simple kinematic chain will be implemented in the 3 approaches.

Absolute coordinates are obvious, meaning that the placement of pics are made with at operations with absolute explicit coordinates or, possibly, named coordinates. It is an easy way but hard to adjust if changing global element spacing. In the example, a scale factor was imposed to the overall image just for illustration purposes.

#### Absolute coordinates



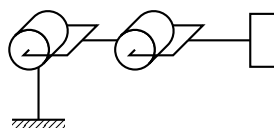
```

\centering
\begin{tikzpicture}[scale=0.7,transform shape]
\pic (BaseR) at (0,-0.5) {frame};
\pic (J1) at (0, 1) {rotational 3D P=1};
\pic (J2) at (2, 1) {rotational 3D P=1};
\pic (Grp) at (4, 1) {gripper};
\draw [knLinkStyle]
(BaseR-out) -- (J1-south)
(J1-out) -- (J2-in)
(J2-out) -- (Grp-in)
;
\end{tikzpicture}

```

More flexible than **absolute coordinates** is the usually preferred way of **relative placement** where pics are placed relatively to other objects in the scene. The usual left, right, above, etc. modifiers allow a very easy relative placement. Most of the times, changing a single object in the chain (as long as its name does not change) is an easy task, and may not require other modifications. Changing spacing globally is easy with the node distance option. As in the previous example, a global scale factor was used to illustrate the possibilities.

#### Relative positioning



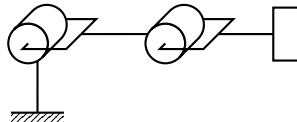
```

\centering
\begin{tikzpicture}[scale=0.7, transform shape, node distance=1.5cm and 2cm]
\pic (BaseR) {frame};
\pic (J1) [above=of BaseR-north] {rotational 3D P=1};
\pic (J2) [right=of J1-center] {rotational 3D P=1};
\pic (Gripp) [right=of J2-center] {gripper};
\draw [knLinkStyle]
(BaseR-out) -- (J1-south)
(J1-out) -- (J2-in)
(J2-out) -- (Gripp-in)
;
\end{tikzpicture}

```

A last approach is to use a matrix to place the pics. This has also a large flexibility in changing elements of the kinematic chain and can even have an advantage of visual perception where the pics are. However, specific global options for the pics have to be passed in each one, scale factors included! In the example, the properties [scale=0.7, transform shape] that were used globally in the former examples were fed into a local style (named my), which is then passed to each pic in the matrix of pics. Most likely, TikZ offers other ways to overcome this limitation, but this one works well. Additionally, a definition was created to make the matrix less verbose and more legible:

#### Matrix of pics



```

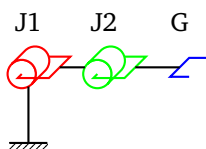
\centering
\def\RDP{rotational 3D P} %Alias to shorten the entries in matrix
\begin{tikzpicture}
\tikzset{my/.style={scale=0.7, transform shape}}
\matrix [column sep = 0.5cm, row sep = 0.5cm]
{
\pic[my] (J1)    {\RDP=1};& \pic[my] (J2) {\RDP=1};& \pic[my] (Gr) {gripper};\\
\pic[my] (BaseR) {frame}; & & \\
};
\draw [knLinkStyle]
(BaseR-out) -- (J1-south)
(J1-out)    -- (J2-in)
(J2-out)    -- (Gr-in)
;
\end{tikzpicture}

```

In all cases, after the placement of the elements (frames, end-effectors and joints) comes the drawing of the connections or links. In all the previous cases, the solution is the same with simple draw commands. Notice that when building kinematic chains with ISO (or non-ISO) joint components, we usually connect from node **-end** (or **-out**) from one component to node **-start** (or **-in**) of another component.

## 2.9 Examples of TikZ-like parameterization

In case of convenience illustrative or descriptive purposes, it is possible to add labels and drawing properties to joints (and links), as in the following example:



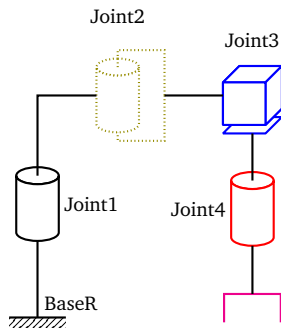
```

\begin{tikzpicture}[scale=0.5, transform shape, node distance=2cm]
\pic (Bas)                {frame};
\pic (J1) [above=of Bas-north,red] {rotational 3D P=1}
  \node [TSF, above=10pt of J1-center] {J1};
\pic (J2) [right=of J1-center,green] {rotational 3D P=1}
  \node [TSF, above=10pt of J2-center] {J2};
\pic (Gr) [right=of J2-center,blue] {gripper=0/1}
  \node [TSF, above=10pt of Gr-end] {G};
\draw [knLinkStyle]
(Bas-out) -- (J1-south)(J1-out) -- (J2-in)(J2-out) -- (Gr-in);
\end{tikzpicture}

```

or this more complete case where more drawing parameters are used:





```

\begin{tikzpicture} [scale=0.75, node distance=2cm,
transform shape]
\pic (BaseR) {frame} node [above right]{BaseR};
\pic (Joint1) [above=of BaseR-center] {rotational 3D V=0}
node [right=10pt of Joint1-center] {Joint1};
\pic [olive,densely dotted, above right=of Joint1-north]
(Joint2) {rotational 3D V=1} node [above=10pt of
Joint2-north] {Joint2};
\pic [blue, right=of Joint2-east] (Joint3) {linear 3D=2}
node [above=10pt of Joint3-north] {Joint3};
\pic [red,below=of Joint3-center] (Joint4) {rotational 3D
V=0} node [left=10pt of Joint4-center] {Joint4};
\pic [magenta,below=of Joint4-north] (Grp) {gripper=3};
\draw[knLinkStyle]
(BaseR-north) -- (Joint1-south)
(Joint1-north) |- (Joint2-west)
(Joint2-out) -- (Joint3-west)
(Joint3-south) -- (Joint4-north)
(Joint4-south) -- (Grp-in)
;
\end{tikzpicture}

```

### 3 Creating kinematic chains

Kinematic chains can have a quite variable complexity, and the designer has several variables and decision options to perform the drawing. In summary, those main options are as described next.

#### When the focus elements to draw are links:

- Extremities or attach points are defined to delimit the links;
- The joints appear naturally at these link extremities;
- These are most commonly used for 2D kinematic chains. Being 2D, these representations transmit a good geometric match with the real mechanism.

#### When the focus elements to draw are the joints:

- This is a more symbolic representation suited both for 2D and 3D kinematic chains;
- The links are drawn between joints, usually from the node `-end` or `-out` of one component (joint or frame) to node `-start` or `-in` of another component (joint or end effector);
- This approach is commonly found in 3D more advanced mechanisms, like robot manipulators.

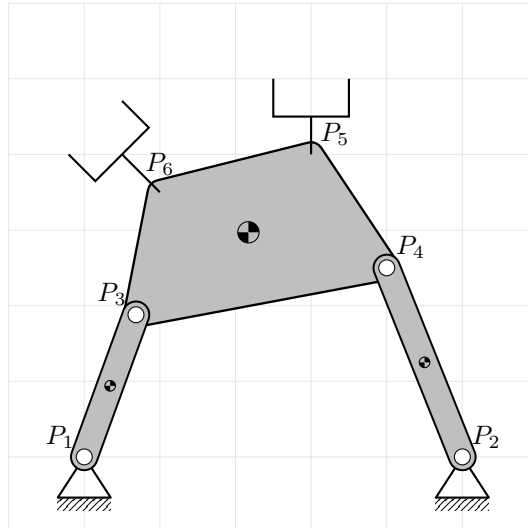
#### When placing components, their position (center or extremities) can be specified in:

- absolute places or coordinates (clear but less flexible to later adjustments);
- in relative positions of others components' internal nodes (best flexibility);
- in a matrix of nodes (flexible, but not so common due to some restrictions);

These issues were described and detailed earlier, and just for illustration purposes follow some examples of planar and non-planar mechanisms, and in this later case focusing mainly in the ISO notation, since non-ISO notation examples were given earlier.

### 3.1 Examples of planar mechanisms

Follow a couple of examples of planar kinematic chains for illustration purposes.



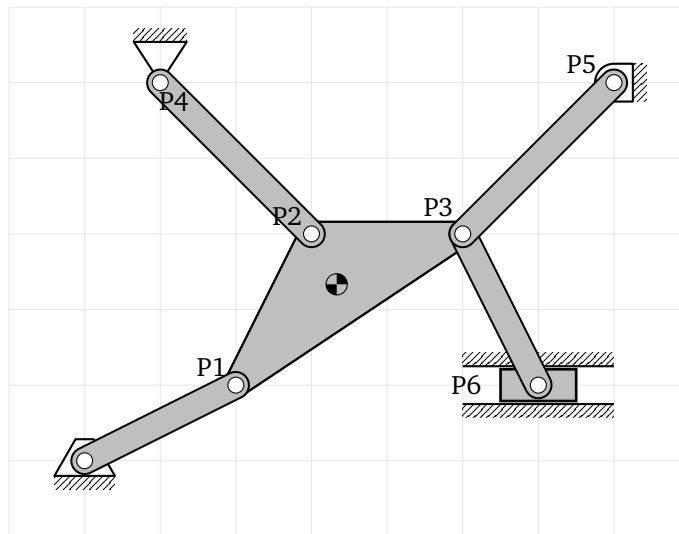
```

\centering
\begin{tikzpicture}
\draw [help lines,gray!20] (-1,-1) grid (6,6);
%
\coordinate (P1) at (0,0)   node at (P1) [above left]       {$P_1$};
\coordinate (P2) at (5,0)   node at (P2) [above right]      {$P_2$};
\coordinate (P3) at (70:2)  node at (P3) [above left]       {$P_3$};
\coordinate (P4) at (4,2.5) node at (P4) [above right]      {$P_4$};
\coordinate (P5) at (3,4)   node at (P5) [above right]      {$P_5$};
\coordinate (P6) at (1,3.5) node at (P6) [above,yshift=3pt] {$P_6$};

\pic (base1) at (P1) {frame pivot triangle};
\pic (base2) at (P2) {frame pivot triangle};

\def\ListOfPivotPointsToDraw{1,0,0} %macro used by link polygon
\pic (link3) at (P3) {link polygon={P4/P5/P6}};
\pic (link1) at (P1) {link bar generic={P3}};
\pic (link2) at (P2) {link bar generic={P4}};
\draw [knLinkStyle] (P5) -- ++(0,0.5) node (tip) {};
\pic at (tip) {gripper=1};
\draw [knLinkStyle] (P6) -- ++(-0.5,0.5) node (tip2) {};
\pic [rotate=135] at (tip2) {gripper};
\end{tikzpicture}

```



```

\centering
\begin{tikzpicture}[scale=1, transform shape]
  \draw [help lines,gray!20] (-1,-1) grid (8,6);

  \coordinate (P0) at (0,0);
  \coordinate (P1) at (2,1);
  \coordinate (P2) at (3,3);
  \coordinate (P3) at (5,3);
  \coordinate (P4) at (1,5);
  \coordinate (P5) at (7,5);
  \coordinate (P6) at (6,1);
  \coordinate (P7) at (4,2);

  % \pic at (P1) {link triangular={P2}{P3}} ;
  \pic at (P1) {link polygon=P2/P3} ;

  \pic [rotate=180](B0) at (P4) {frame pivot triangle};
  \pic [rotate=0] (B1) at (P0) {frame pivot trapezium};
  \pic [rotate=90] (B2) at (P5) {frame pivot rounded};
  \pic [rotate=0] (B3) at (P6) {frame dual pivot slide};

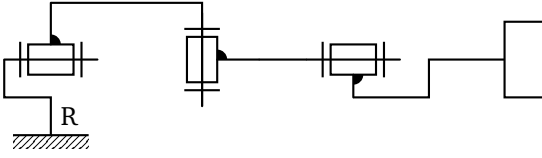
  \pic (myB) at (B1-center) {link bar generic={P1/0/1/1} };
  \pic (link3) at (P3) {link bar generic={B3-center/0/1/1}};
  \pic (link3B) at (P3) {link bar generic={B2-center/0/1/1}};
  \pic (link2) at (P2) {link bar generic={B0-center/0/1/1}};

  \node [anchor=south east,TSF] at (P1) {P1};
  \node [anchor=south east,TSF] at (P2) {P2};
  \node [anchor=south east,yshift=3pt,TSF] at (P3) {P3};
  \node [anchor=north west,xshift=-4pt,TSF] at (P4) {P4};
  \node [anchor=south east,xshift=-3pt,TSF] at (P5) {P5};
  \node [anchor=east,xshift=-18pt,TSF] at (P6) {P6};
\end{tikzpicture}

```

### 3.2 Examples of non planar (spatial) mechanisms

This section illustrates several cases of non planar (3D or spatial) mechanisms using the ISO notation and illustrating some variations and configurations allowed by the KinemaTikZ package.

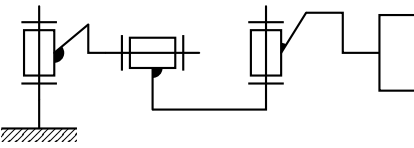


```

\centering
\begin{tikzpicture}[knLinkStyle,node distance=1cm and 2cm]
\pic (J0) {frame} node [above right] {R};
\pic (J1) [above=of J0-out] {revolute pair spatial=90:0.75/0/0};
%\node [anchor=west,at=(J1-in2)] {J1};
\pic [rotate=90] (J2) [right=of J1-center] {revolute pair spatial=-90:0.75/1/0};
%\node [anchor=west,at=(J2-in2)] {J2};
\pic (J3) [right=of J2-center] {revolute pair spatial=-90:0.5/1/0};
%\node [anchor=west,at=(J3-in2)] {J3};
\pic (GR) [right=of J3-center] {gripper};
%\node [anchor=west,at=(GR-in)] {H};
\draw [] (J0-out) -- ($(J0-out)!0.5!(J1-center)$) -| (J1-in);
\draw [] (J1-out) -| (J2-in);
\draw [] (J2-out) -| (J3-in);
\draw [] (J3-out) -| ($(J3-center)!0.5!(GR-in)$) -- (GR-in);
\end{tikzpicture}

```

The **revolute pair spatial** can have its arm (**-out** or **-end**) created as terminating in an existing point instead of being left with the default. That point can either be local (as in the examples given) or some existing point from other pics or nodes. Normally, it may be easier to define a local destination and then connect with some links.



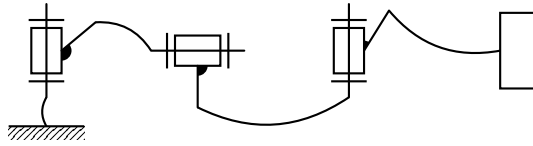
```

\centering
\begin{tikzpicture}[knLinkStyle, node distance=1cm and 1.5cm]
\pic (J0) {frame};
\pic [rotate=-90] (J1) [above=of J0-out] {revolute pair spatial=120:0.75/0/0};
\pic [rotate=0] (J2) [right=of J1-center] {revolute pair spatial=-90:0.75/1/0};
\pic [rotate=90] (J3) [right=of J2-center] {revolute pair spatial=-45:0.75/1/0};
\pic (GR) [right=of J3-center] {gripper};
\draw [] (J0-out) -- (J1-in2);
\draw [] (J1-out) |- (J2-in);
\draw [] (J2-out) -| (J3-in);
\draw [] (J3-out) -| ($(J3-out)!0.5!(GR-in)$) |- (GR-in) ;
\end{tikzpicture}

```

Connections between joints (i.e. the links) are normally polygonal lines either straight `--`, with a perpendicular expression, `-|` or `|-`, or a dual perpendicular composite line between points P1 and P2, with `(P1)-| ($(P1)!0.5!(P2)$) |- (P2)`, or other variations on it. Additionally, although less frequent, links can be curved lines like in the following example where `to` or `edge` instructions are used.

### Example of kinematic chain with curved links



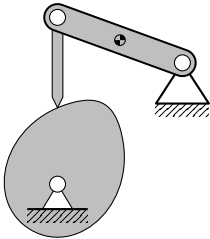
```

\centering
\begin{tikzpicture}[knLinkStyle, node distance=1cm and 2cm]
\pic (J0) {} {frame};
\pic [rotate=-90] (J1) [above=of J0-out] {revolute pair spatial=120:0.75/0/0};
\pic [rotate=0] (J2) [right=of J1-center] {revolute pair spatial=-90:0.75/1/0};
\pic [rotate=90] (J3) [right=of J2-center] {revolute pair spatial=-45:0.75/1/0};
\pic (GR) [right=of J3-center] {gripper};
\draw [bend left] (J0-out) edge (J1-in2);
\draw [bend left] (J1-out) edge (J2-in);
\draw [bend right] (J2-out) edge (J3-in);
\draw [bend right] (J3-out) edge (GR-in);
\end{tikzpicture}

```

### 3.3 Other examples of kinematic chains

The kinematic chains allowed by the package cover even some higher kinematic pairs, such as cams and other arrangements. Far from being exhaustive, follow an example of a not so common mechanism. For a full list of the joints and links available in the package and their configurations report to section 4.



```

\begin{tikzpicture}
\pic (mycam1) {} {camlink}; %rotating cam plate
\pic (mypin1) at (mycam1-touchtop) {campin}; %cam follower
\pic (mybar) at (mypin1-end) {} {link bar generic=-20:50pt};
\begin{pgfonlayer}{background}
\pic (myf2) at (mybar-end) {} {frame pivot triangle};
\end{pgfonlayer}
\end{tikzpicture}

```

## 4 Reference listing of elements available in the package

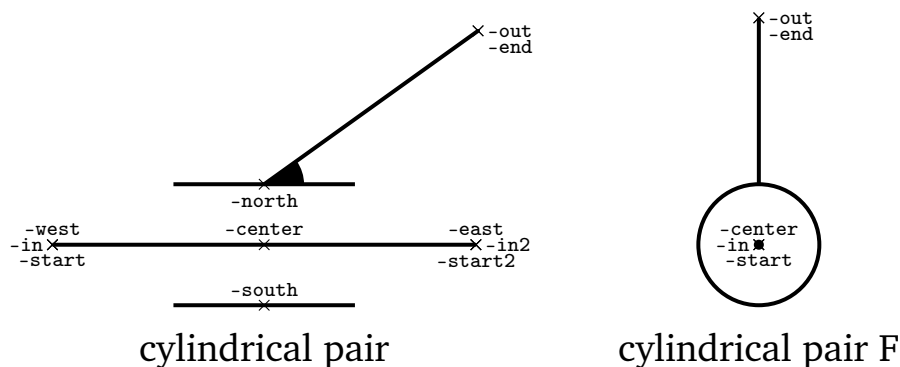
This section describes one by one all the links and joints available in the KinemaTikZ package. For each case a large image is included with most of the default parameters and illustrating the main node points of the `\pic`, such as as `-center`, `-in`, `-out` and others. In some cases, multiple node points coincide but they are all kept for easier usage. To avoid cluttering of the same point with different names, occasionally some names were omitted and only the most relevant were are shown.

Besides the geometry and the node attaching points, the arguments are enumerated, and examples using different parameters are shown in illustrative examples and their associate code.

For ISO compliant elements, the actual ISO names were used although all lower case letters; nonetheless, in some cases, there are some alternative names that correspond to the same element. For elements outside the ISO standard, the term "3D" is used in the name to indicate a 3D graphic representation (perspective), and it is omitted when the symbol (non-ISO) has a 2D graphical expression.

### 4.1 The *cylindrical pair*

The *cylindrical pair* is the element 2.2.1 from the ISO standard. It is a 2 DOF kinematic pair and it has two representations: one with the axis in the plane of the page, and another, here named *cylindrical pair F*, which has the axis orthogonal to the plane of the page.



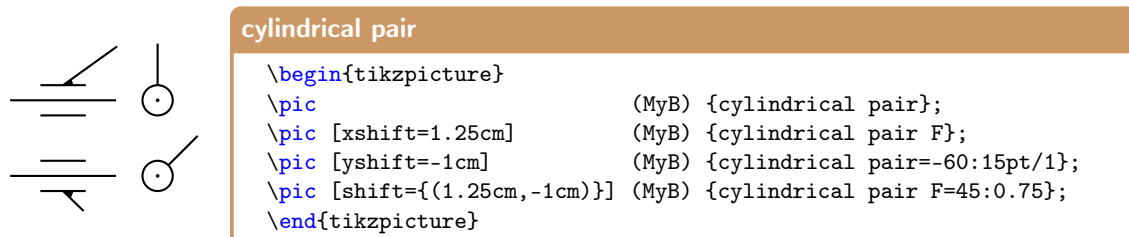
#### Parameters for *cylindrical pair*

1. point: point for link end (default: 45:1cm);
2. boolean: start of end arm: 0 top, 1 bottom (default: 0);

#### Parameters for *cylindrical pair F*

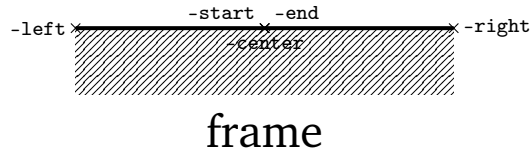
1. point: point for link end (default: 90:0.75cm);

Illustration of usage:



## 4.2 The *frame*

The *frame* is a simple horizontal frame. It is the element 3.1 in the ISO standard, which accepts some variants. This representation is the simplest frame of all possibilities offered by the KinemaTikZ package



### Parameters

1. length: base width

Illustration of usage:

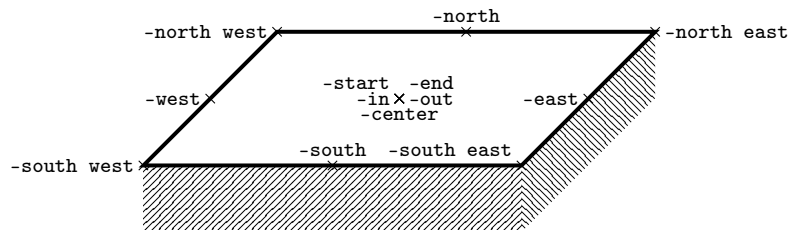


#### frame

```
\begin{tikzpicture}
\pic (MyB) {frame};
\pic [yshift=-0.75cm] (MyB) {frame=1.5cm};
\pic [yshift=-1.5cm] (MyB) {frame=2cm};
\end{tikzpicture}
```

## 4.3 The *frame 3D*

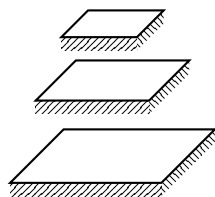
The *frame 3D* is an adaptation of the simple frame to emulate a 3D layout and useful for 3D variants of kinematic chains. It is not present in the official ISO standard.



### Parameters

1. length: base width

Illustration of usage:

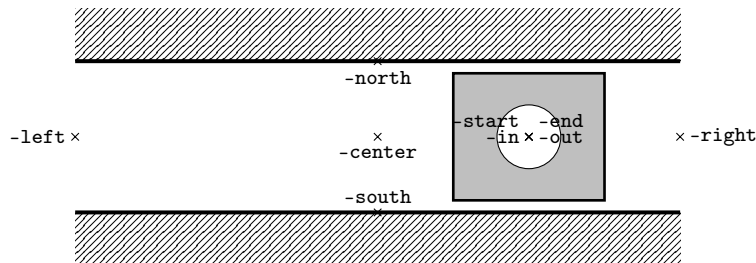


#### frame 3D

```
\begin{tikzpicture}
\pic (MyB) {frame 3D};
\pic [yshift=-0.75cm] (MyB) {frame 3D=1.5cm};
\pic [yshift=-1.75cm] (MyB) {frame 3D=2cm};
\end{tikzpicture}
```

#### 4.4 The *frame dual pivot slide*

The *frame dual pivot slide* is a frame with two fixed parts with an internal sliding pivot. This is actually a joint with two degrees of freedom attached to a dual frame, above and below.

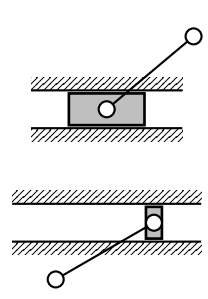


frame dual pivot slide

#### Parameters

1. A measurement: the length of the frame (default 2cm)
2. A number: relative position of cursor (from -0.5 to 0.5; 0 is at the center which is the default). Values outside this range are clipped.
3. A number: relative size of cursor regarding the frame. Default is 0.5. Values are clipped to a minimum of the diameter of the pivot and a maximum of 1cm.

Next picture shows two cases of application where an additional link was added to illustrate the functionality. In the first case, the defaults are applied, in the second case a longer frame is used, the cursor is at 30% to the right (0.3), and the cursor dimension was attempted to be 1% (0.01) the size of the frame, but was clipped to the minimum size allowed, which is the size of the pivot!



**frame dual pivot slide**

```

\begin{tikzpicture}
\pic (MyB) {frame dual pivot slide};
\pic at (MyB-out) {revolute pair planar=40:1.5cm};
{[yshift=-1.5cm]
\pic (MyB) {frame dual pivot slide={2.5cm/0.3/0.01}};
\pic at (MyB-out) {revolute pair planar=-150:1.5cm};
}
\end{tikzpicture}

```

#### 4.5 The *frame pivot flat*

The *frame pivot flat* is a horizontal frame with a pivot at the very ground level.



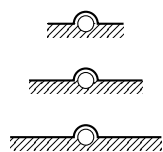
frame pivot flat



## Parameters

1. measurement: base width

The base width parameter reflects in the overall size of the block (but not the pivot point as a scale change would affect), as illustrated next:

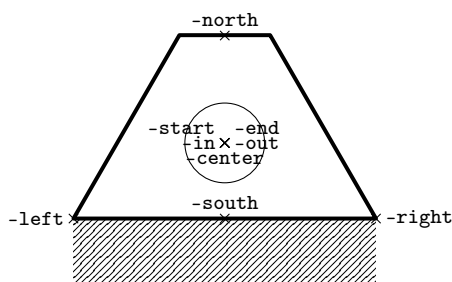


### frame pivot flat

```
\begin{tikzpicture}
\pic (MyB) {frame pivot flat};
\pic [yshift=-0.75cm] (MyB) {frame pivot flat={1.5cm}};
\pic [yshift=-1.5cm] (MyB) {frame pivot flat={2cm}};
\end{tikzpicture}
```

## 4.6 The frame pivot trapezium

The **frame pivot trapezium** is a frame with a trapezoidal geometry.

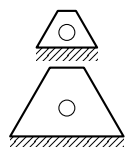


### frame pivot trapezium

## Parameters

1. measurement: base width

The base width parameter reflects in the overall size of the block (but not the pivot point as a scale change would affect), as illustrated next:

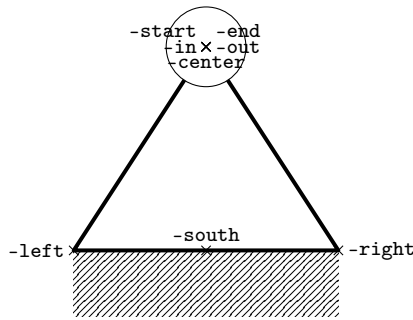


### frame pivot trapezium

```
\begin{tikzpicture}
\pic (MyB) {frame pivot trapezium};
\pic [yshift=-1cm] (MyB) {frame pivot trapezium=1.5cm};
\end{tikzpicture}
```

## 4.7 The frame pivot triangle

The **frame pivot triangle** is a frame with a triangle geometry.

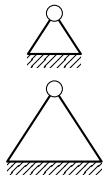


frame pivot triangle

### Parameters

1. measurement: base width

The base width parameter reflects in the overall size of the block (but not the pivot point as a scale change would affect), as illustrated next:



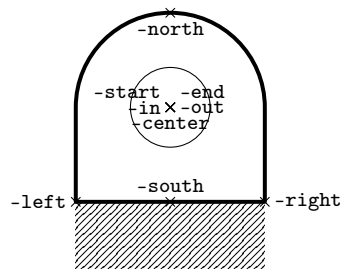
```

frame pivot triangle
\begin{tikzpicture}
\pic (MyB) {frame pivot triangle};
\pic [yshift=-1cm] (MyB) {frame pivot triangle=1.25cm};
\end{tikzpicture}

```

## 4.8 The frame pivot rounded

The **frame pivot rounded** is a frame with the shape of a small pole with a rounded top and a pivot point.

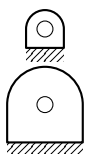


frame pivot rounded

### Parameters

1. measurement: base width

The base width parameter reflects in the overall size of the block (but not the pivot point as a scale change would affect), as illustrated next:



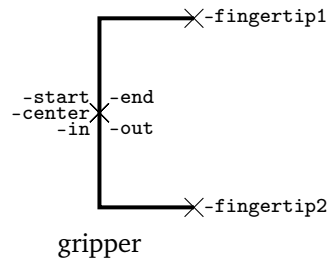
```

frame pivot rounded
\begin{tikzpicture}
\pic (MyB) {frame pivot rounded};
\pic [yshift=-1cm] (MyB) {frame pivot rounded=1cm};
\end{tikzpicture}

```

## 4.9 The gripper

The `gripper` is a simple representation of a two finger gripper as an end-effector. It accepts two parameters intended to mimic different orientations with or without perspective views. The attaching nodes are the following.

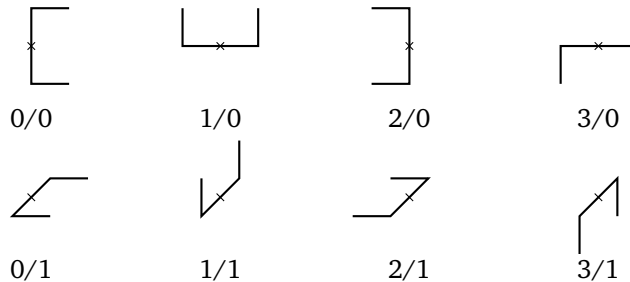


### Parameters

The two parameters control the orientation and whether or not the representation is to be made in perspective. They are the following, and the absence of parameters is equivalent to 0/0:

1. Number: multiple of 90° of fingers' direction. Default is 0 (horizontal right).
2. Boolean: 1 to draw in perspective (0, no perspective, is the default)

#### gripper - standard cases (orientation/perspective)



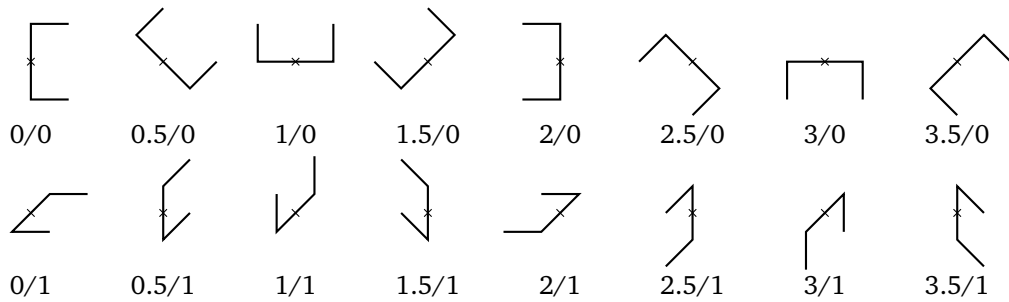
```

\centering
\begin{tikzpicture}[scale=1,transform shape]
\foreach \persp in {0,1}
  \foreach \dir in {0,1,...,3}
  {
    \def\fType{\dir/\persp}
    \begin{scope}[xshift=\dir*2.5cm, yshift=-\persp*2cm]
      \pic (MyPic) {gripper=\fType};
      \draw [shift=(MyPic-center)] plot [mark=x,scale=1] coordinates{(0,0)};
      \node [at=(MyPic-center),yshift=-2.75em, TSF] {\fType};
    \end{scope}
  }
\end{tikzpicture}

```

Despite these "standard" representations, the pic actually accepts any number in the first argument, which will translate as fractional parts of 90°. In case they are useful, they can be used. Follows a representation of several possibilities:

### gripper - extended set of cases (orientation/perspective)



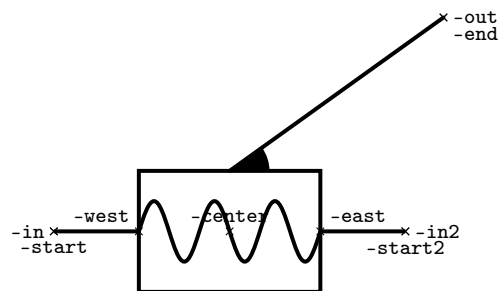
```

\centering
\begin{tikzpicture}[scale=1,transform shape]
\foreach \persp in {0,1}
  \foreach \dir in {0,0.5,...,3.5}
  {
    \def\fType{\dir/\persp}
    \begin{scope}[xshift=\dir*3.5cm, yshift=-\persp*2cm]
      \pic (MyPic) {gripper=\fType};
      \draw [shift=(MyPic-center)] plot [mark=x,scale=1] coordinates{(0,0)};
      \node [at=(MyPic-center),yshift=-2.75em,TSF] {\fType};
    \end{scope}
  }
\end{tikzpicture}

```

## 4.10 The helical pair

The **helical pair**, also named **screw pair**, represents the item 2.1.3 from the ISO standard.

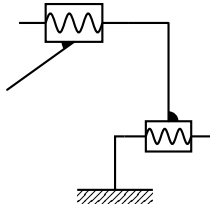


helical pair

### Parameters

1. point: point for link end (default: 45:1cm);
2. boolean: start of end arm: 0 top, 1 bottom (default: 0);

Illustration of usage:



```
helical pair
\begin{tikzpicture}[knLinkStyle]
\pic (MyB) {frame};
\pic [above right of=MyB-out] (MyH1) {helical pair=90:1.5cm/0};
\pic [scale=-1.25,right of=MyH1-out] (MyH2) {helical pair};
\draw (MyB-out) |- (MyH1-in);
\draw (MyH1-out)--(MyH2-in);
\end{tikzpicture}
```

The option `knLinkStyle` applied to the entire figure ensures a uniform style for the drawing, although specific style variations (like color, line thickness changes or scale, among others) may occur locally.

### 4.11 The *linear 3D*

The *linear 3D* is a representation of a linear/prismatic joint exhibiting a 3D look. It accepts 6 different configurations to reflect different 3D layouts depending on the preferred option. There is an additional configuration (default), which is similar to one of the others except that it has small links attached mainly for illustration purposes. Among many other attaching nodes, the joint has a `-in` and a `-out` node whose positions depend on the configuration adopted. The six (seven) configurations allowed are defined by the first argument of the `pic` which describes the direction of the motion.

#### Parameters

1. number: type of configuration
  - No argument: defaults to the right (5) with simple ends drawn (for illustration). The remainder options do not draw end segments and simply add a plane for their direction of motion.
  - 1: top
  - 2: bottom
  - 3: near (front/closer)
  - 4: far (back of page)
  - 5: right
  - 6: left

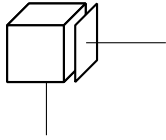
**linear 3D**

---

```
\centering
\begin{tikzpicture}
\pic (MyLO) {linear 3D};
\foreach \n in {1,...,6}
{
\pic[xshift=\n*2cm] (MyL\n) {linear 3D=\n};
\node at (MyL\n-label) {\n};
}
\end{tikzpicture}
```

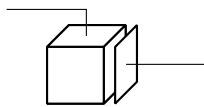
The six (seven) possible configurations and their respective node points are shown next. The node names, like `-in` of `-out` and others, change accordingly with the context to reflect the actual position in the schematic.

Anyway, the programmer will decide which node suits best the purpose of the application for the sake of anchoring points and attaching links. For example if the programmer requires a linear joint with orthogonal links, the following could be issued:

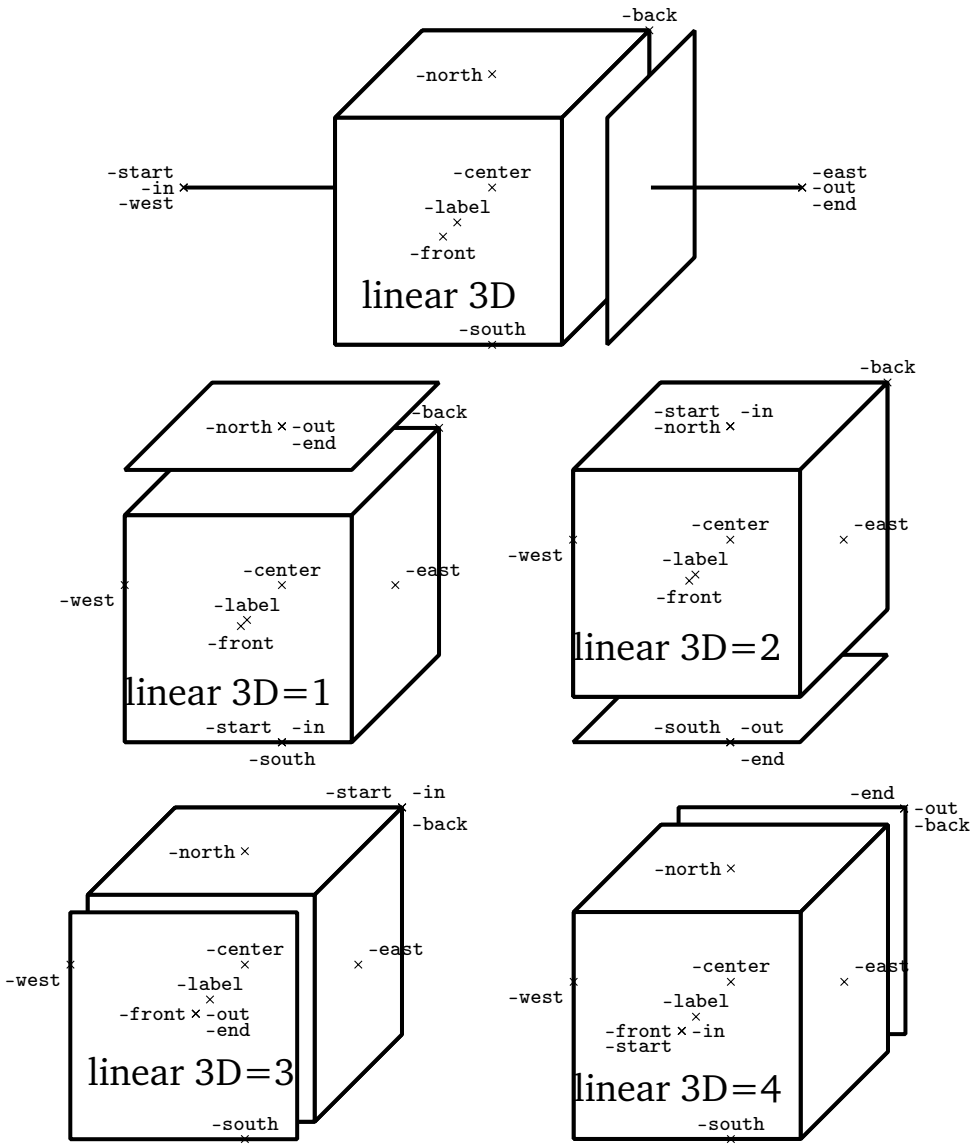


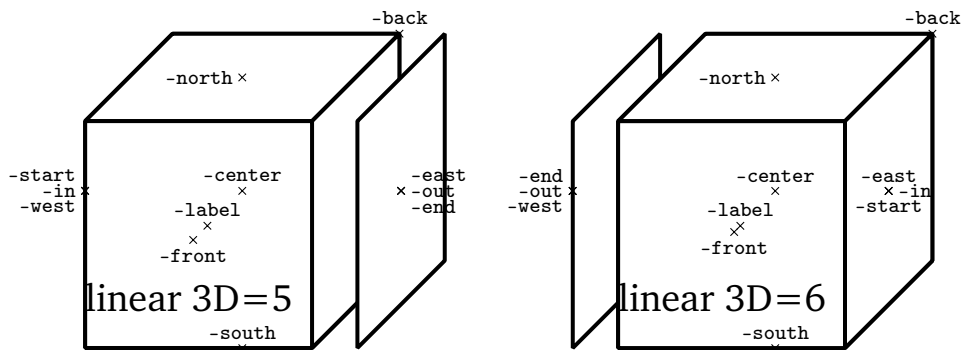
```
\begin{tikzpicture}
\pic (MyL) {linear 3D=5};
\draw (MyL-south) -- ++ (0,-20pt);
\draw (MyL-out) -- ++ (30pt,0);
\end{tikzpicture}
```

Or even something like this:



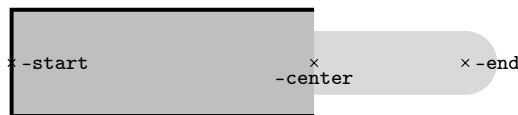
```
\begin{tikzpicture}
\pic (MyL) {linear 3D=5};
\draw (MyL-north) |- ++(-30pt,10pt);
\draw (MyL-out) -- ++ (30pt,0);
\end{tikzpicture}
```





## 4.12 The *linear joint bar*

The **linear joint bar** is an alternative representation in 2D of a linear or prismatic joint.



linear joint bar

The object accepts parameters to define its length, fraction of insertion and the existence of round or flat tips to adapt to variate applications.

### Parameters

1. number from 0 to 1: fraction of insertion of piston (default: 0.5);
2. length : length of fixed part (default: 1cm)
3. boolean : has curved base and pivot (default 0: no)
4. boolean : has curved tip (default 1: yes)

Representations with different fractions of insertion and with flat or curved tips and base:

**linear joint bar**

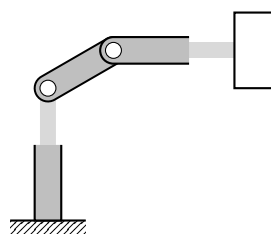
---

```

\centering
\begin{tikzpicture}
\foreach \n/\b/\t in {0.2/0/0, 0.4/0/1, 0.6/1/1, 0.8/1/0, 0.95/0/1}
  \pic [xshift=12*\n*1.3cm] {linear joint bar=\n//\b/\t};
\end{tikzpicture}

```

Illustration of usage:



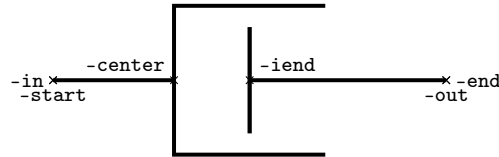
```

\begin{tikzpicture} %[knLinkStyle]
\pic (F) {frame};
\pic [rotate=90] (P) {linear joint bar=0.25};
\pic at (P-out) (Q) {link bar generic=30:1cm/0};
\pic at (Q-out) (R) {linear joint bar=0.4//1/0};
\pic at (R-out) (G) {gripper};
\end{tikzpicture}

```

### 4.13 The *linear piston*

The *linear piston* is an alternative representation in 2D of a linear or prismatic joint.



linear piston

#### Parameters

1. number from 0 to 1: fraction of insertion of piston (default: 0.5);
2. measurement: width of piston plate (default: 10pt);

Representations with different fractions of insertion and with non-null and null piston plate widths:

**linear piston**

0.15	0.3	0.45	0.6	0.75	0.9

---

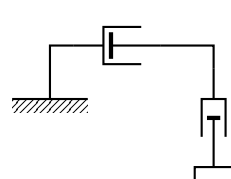
```

\newcommand*{\MyNum}[1]{\pgfmathprintnumber[precision=2, fixed]{#1}}
\centering
\begin{tikzpicture}[font=\small]
\foreach \n in {0.15,0.3,...,1}
{
  \pic [xshift=12*\n*1.3cm] {linear piston=\n};
  \node [shift={(12*\n*1.3cm,-0.80cm)}] {\MyNum{\n}};
  \pic [shift={(12*\n*1.3cm,-1.5cm)}] {linear piston=\n/0};
}
\end{tikzpicture}

```

In the previous code, a macro (`\MyNum`) was created to ensure a proper decimal representation of numbers due to the small resolution of fractional numbers in  $\text{T}_\text{E}_\text{X}$ .

Illustration of usage:



```

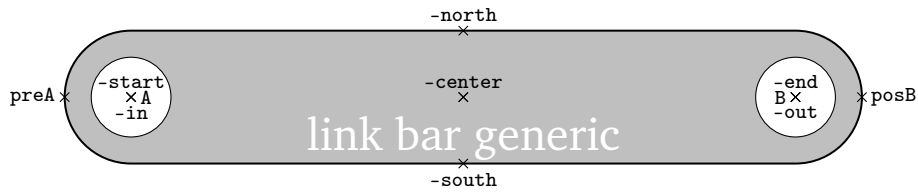
\begin{tikzpicture}[knLinkStyle]
\pic (F) {frame};
\pic [above right of=F-out] (P) {linear piston=0.2};
\pic [below right of=P-out,rotate=-90] (Q) {linear piston=/5pt};
\pic [scale=0.5] at (Q-out) (G) {gripper=3};
\draw (F-out) |- (P-in) (P-out) -| (Q-in);
\end{tikzpicture}

```

### 4.14 The *link bar generic*

The *link bar generic* is a general purpose planar link with multiple applications that has several internal nodes and variate configurations through parameters.





**Parameters**

1. point: end point coordinates or named point without parenthesis ( )
2. boolean: 1 includes the COM symbol in the **-center**
3. boolean: 1 has pivot in **-start**
4. boolean: 1 has pivot in **-end**
5. boolean: 1 has cross hairs on pivots/extremities

The defaults for the 5 parameters are: 0:30pt/1/1/1/0

Follow some illustrations of configurations, but all combinations are allowed.

**link bar generic**

```

\begin{tikzpicture}
\pic (MyB) {frame pivot triangle};
\pic (MyL1) at (MyB-center) {link bar generic};
{[yshift=-2cm] %A new scope to draw below with relative coordinates
  \pic (MyB) {frame pivot triangle};
  \pic (MyL1) at (MyB-center) {link bar generic={30:40pt/0/1/0}};
}
{[yshift=-4cm] %another scope...
  \pic (MyB) {frame pivot triangle};
  \pic (MyL1) at (MyB-center) {link bar generic={30:40pt/1/1/1/1}};
}
{[yshift=-6cm] %... and a last scope.
  \pic [rotate=-90] (MyB) {frame};
  \pic (MyL1) at (MyB-center) {link bar generic={0:40pt/0/0/1}};
}
\end{tikzpicture}

```

The **link bar generic** even accepts a configuration of a fixed bar, that is, without any joints in the extremes like the following example:

```

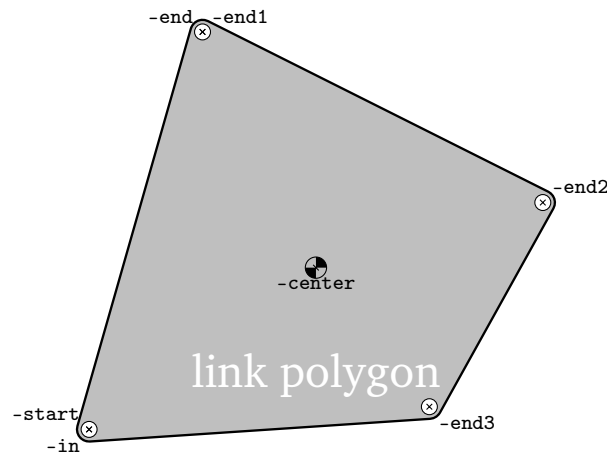
\begin{tikzpicture}
\pic [rotate=-90] (MyB) {frame};
\pic [rotate=90, right=2cm] (MyB2) {frame};
\pic (MyL1) at (MyB-center) {link bar generic={MyB2-out/0/0/0}};
\end{tikzpicture}

```

It is worth observe that the second frame on the figure was placed 2 cm on the right (of the placement point that is the (0,0) because it is omitted), and we did not use the TikZ option `xshift=2cm` because the `\pic` is rotated and the horizontal direction is no longer `x`; but we could have used `yshift=-2cm` for the same effect, with the negative sign occurring because when rotating  $+90^\circ$  the `y` direction grows to the left in the page!

## 4.15 The *link polygon*

The *link polygon* is a link with polygonal shape with one input pivot and virtually any number of end joint pivots to attach to other links. These end pivots can be suppressed by using a special predefined macro. This is a generalization of the ISO standard name *ternary link* (item 4.4) and likely similar to the ISO designation *multi-element link* (item 4.5), although no representation of it is given in the document of ISO.

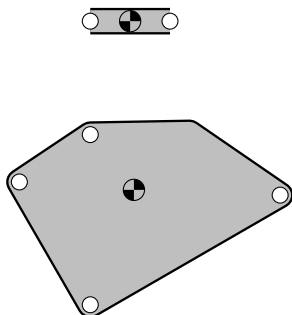


The *link polygon* accepts a variable number of parameters which are the additional pivot points besides the start pivot.

### Parameters

1. point: First additional pivot named (-end1) (default: 0:30pt)
2. point: Second additional pivot named (-end2)
- ...
- $n$ . point:  $n^{\text{th}}$  additional pivot named (-end $n$ )

Without parameters a default is used for a single end point, yielding something similar to a link bar as a particular case. To selectively suppress pivot points (except the start which is always present), the macro `\ListOfPivotPointsToDraw` can be defined as a boolean list of 1s or 0s separated by commas like for example `\def\ListOfPivotPointsToDraw{1,0,1,1}` that indicates that in the next polygon the `-end2` will have no pivot but `-end1`, `-end3` and `-end4` will have pivots. The COM, which is always present, is calculated automatically and defines also the `-center`.

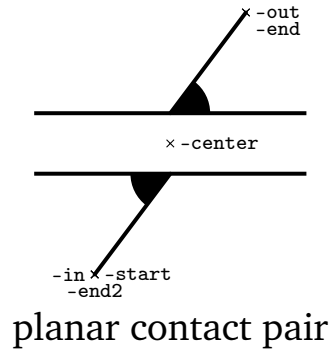


### link polygon

```
\begin{tikzpicture}
\pic (MyP) {link polygon};
{[yshift=-3.75cm]
\def\ListOfPivotPointsToDraw{1,0,1,1}
\foreach \i in {1,...,4} {%
\pgfmathsetmacro{\tA}{\i*30} ;
\pgfmathsetmacro{\tR}{(0.75*cos(\tA) + 2.25) } ;
\coordinate (P\i) at (\tA:\tR);
}
\pic (MyP) {link polygon=P1/P2/P3/P4};
}
\end{tikzpicture}
```

#### 4.16 The *planar contact pair*

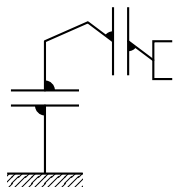
The **planar contact pair** represents the item 2.3.2 from the ISO standard and is a two DOF kinematic pair.



#### Parameters

1. point: point for link end (top) (default: 60:0.5cm);
2. point: point for the other link end (bottom) (default: -120:0.5cm);

Illustration of usage:



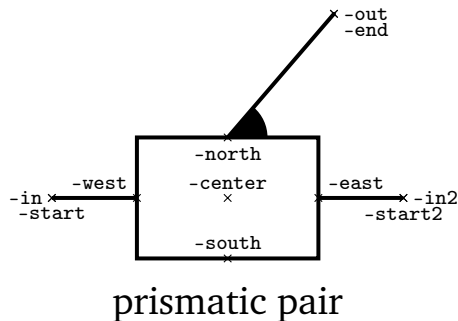
```

planar contact pair
\begin{tikzpicture} [knLinkStyle]
\pic (B) {frame};
\pic [above of=B-out] (L1) {planar contact pair=90:.75cm/-90:.25cm};
\pic [right of=L1-out,rotate=-90] (L2) {planar contact pair};
\pic [scale=0.5] (Gr1) at (L2-out) {gripper};
\draw (B-out)--(L1-in) (L1-out)--(L2-in);
\end{tikzpicture}

```

#### 4.17 The *prismatic pair*

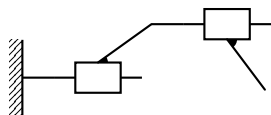
The **prismatic pair**, also named **linear pair**, represents the item 2.1.2 from the ISO standard. Both names are available (and perform the same) in KinemaTikZ.



## Parameters

1. point: point for link end (default: 45:1cm);
2. boolean: start of end arm: 0 top, 1 bottom (default: 0);

Illustration of usage:

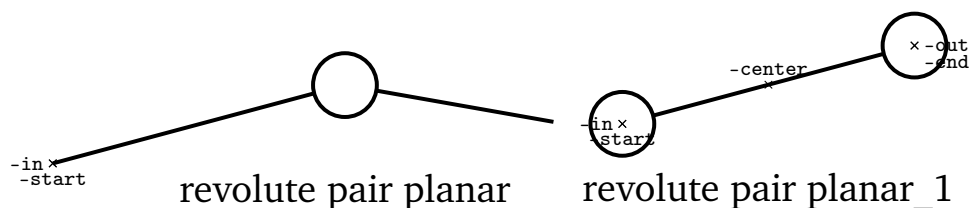


### prismatic pair

```
\begin{tikzpicture}[knLinkStyle]
\pic [rotate=-90] (MyB) {frame};
\pic [right of=MyB-out] (MyL1) {prismatic pair};
\pic [right of=MyL1-out] (MyL2) {linear pair=-60:1cm/1};
\draw (MyB-out) --(MyL1-in);
\draw (MyL1-out)--(MyL2-in);
\end{tikzpicture}
```

## 4.18 The revolute pair planar

The **revolute pair planar** represents the item 2.1.1 a) from the ISO standard. It is one of the most common elements in kinematic chains. It represents the two links and the joint between them. Nonetheless, the KinemaTikZ package allows some additional configurations.

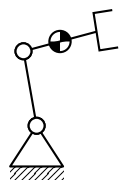


The object accepts three parameters that control the existence of terminal pivot or the presence of COM (center of mass in the link middle point).

## Parameters

1. point: destination point (default: 15:1cm)
2. number: 1 add a COM, 0 do not add COM, -1 draw a special case (default: -1)
3. boolean: add or not pivot in destination (default: 1 for yes)

Illustration of usage:

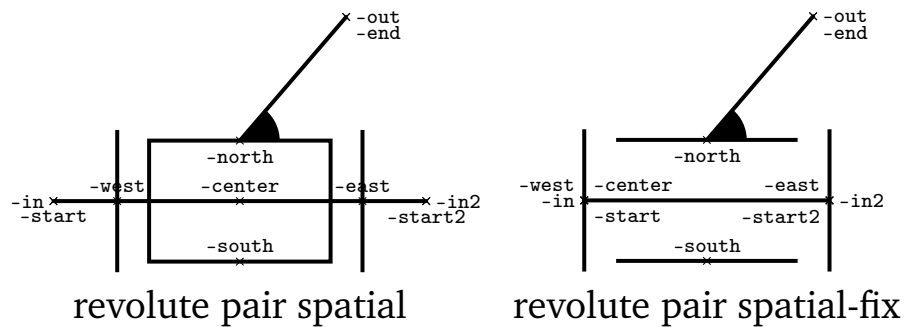


### revolute pair planar

```
\begin{tikzpicture} [knLinkStyle]
\pic (MB) {frame pivot triangle};
\pic at (MB-out) (J1) {revolute pair planar=100:1cm/0/1};
\pic at (J1-out) (J2) {revolute pair planar=/1/0};
\pic [scale=0.5] at (J2-out) {gripper=0.1/0};
\end{tikzpicture}
```

## 4.19 The *revolute pair spatial*

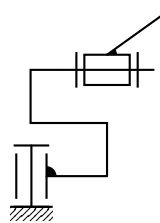
The *revolute pair spatial* represents the item 2.1.1 b) from the ISO standard. According to that document, this joint can have a special representation when the rotation axle is fixed or attached to some frame. In that case, the symbol simplifies a little. To ease the attachment in those circumstances, the reference point *-center* is shifted to the left extreme.



### Parameters

1. point: point for link end (default: 45:1cm);
2. boolean: start of end arm: 0 top, 1 bottom (default: 0);
3. boolean: axle fixed to some frame: 0 not fixed, 1 fixed (default: 0)

Illustration of usage:

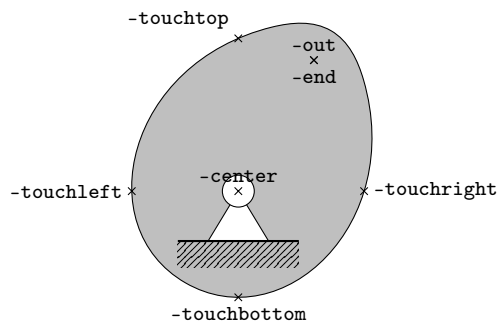


#### revolute pair spatial

```
\begin{tikzpicture} [knLinkStyle]
  \pic (MB) {frame=16pt};
  \pic [rotate=90] at (MB-out) (J1) {revolute pair spatial=-90:1cm/1/1};
  \pic [above right=of J1-east] (J2) {revolute pair spatial};
  \draw (J1-out) |- ($(J1-out)!0.5!([xshift=-40pt]J2-in)$) |- (J2-in);
\end{tikzpicture}
```

## 4.20 The *rotating cam plate*

The *rotating cam plate* is the item 6.1 of the ISO standard, which has also a synonym in KinemaTikZ package named *camlink*. It is a special link that rotates around a point and is mainly intended to create a high kinematic pair with other links, namely the *cam follower*. The official ISO symbol does not include the pivot point on a frame, although there is another related symbol (*eccentric*, item 4.3.1.3 of the standard) that includes it.

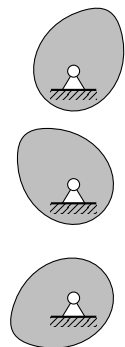


## rotating cam plate (camlink)

### Parameters

1. number: orientation relatively to the vertical (default is  $-30^\circ$ )

Illustration of usage:



#### rotating cam plate / camlink

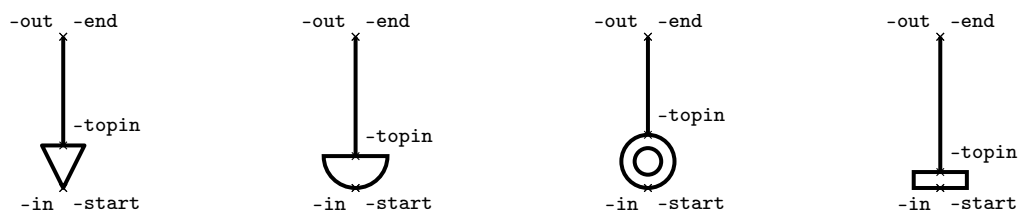
```
\begin{tikzpicture}[scale=0.75,transform shape]
\pic (MyC) {rotating cam plate}; %camlink
\pic [yshift=-2cm] (MyC) {rotating cam plate=45};
\pic [yshift=-4cm] (MyC) {rotating cam plate=120};
\end{tikzpicture}
```

Its boundary touchpoints are calculated automatically with the orientation along the horizontal and vertical axis that cross the `-center`.

Although not expected, and probably not very useful, this implementation also includes an anchor `-end` point that can be used to attach other elements or links.

### 4.21 The cam follower

The **cam follower**, item 6.5 from the ISO standard, is an auxiliary link to be used with a **rotating cam plate** to build a high kinematic pair altogether. The ISO standard accepts 4 variants: knife-edge, arcuate, roller or flat-faced.



cam follower=0   cam follower=1   cam follower=2   cam follower=3

knife-edge

arcuate

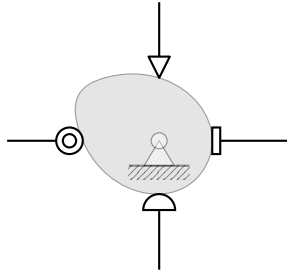
roller

flat-faced

## Parameters

- number: type of follower. Default: 0.
  - 0 – knife-edge
  - 1 – arcuate
  - 2 – roller
  - 3 – flat-faced
- boolean: the layout as described in the standard. 0: vertical (default), and 1: horizontal.

Next follows an illustration of the 4 variants of cam followers in contact with a cam:



### cam follower

```
\begin{tikzpicture}
\pic [opacity=0.4] (mycam1) {rotating cam plate=60};
\pic at (mycam1-touchtop) {cam follower};
\pic [rotate=180] at (mycam1-touchbottom){cam follower=1};
\pic [rotate=90] at (mycam1-touchleft) {cam follower=2};
\pic [rotate=-90] at (mycam1-touchright) {cam follower=3};
\end{tikzpicture}
```

Horizontal variants of the cam follower are as shown next:



### cam follower

```
\begin{tikzpicture}
\pic [opacity=0.4] (mycam1) {rotating cam plate=180};
\pic at (mycam1-touchtop) {cam follower=0/1};
\pic [xscale=-1,rotate=180] at (mycam1-touchbottom) {cam follower=1/1};
\end{tikzpicture}
```

There is also a 5<sup>th</sup> variant which is not in the ISO norm, but that can be found in some literature, and is here named **campin**, also obtainable with **cam follower=4**:



### The campin variant of cam follower

```
\begin{tikzpicture}
\pic {campin};
\pic [xshift=1cm] {cam follower=4}; %the same as campin
\end{tikzpicture}
```

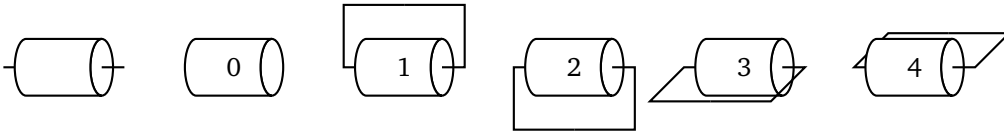
## 4.22 The rotational 3D H

The **rotational 3D H** is a representation of a horizontal rotational/revolute joint exhibiting a 3D look. It accepts 6 different configurations to reflect different 3D layouts depending on the preferred option: with or without an extra arm structure, as occurs in variate literature. Among many other attaching nodes, the joint has a **-in** and a **-out** node whose positions depend on the configuration.

## Parameters

- number: type of configuration with these values:
  - No argument: simple joint with two extremes
  - 0: no ends. A simple cylinder
  - 1: arm at top
  - 2: arm at bottom
  - 3: arm at near (front/closer)
  - 4: arm at far (back of page)

### rotational 3D H



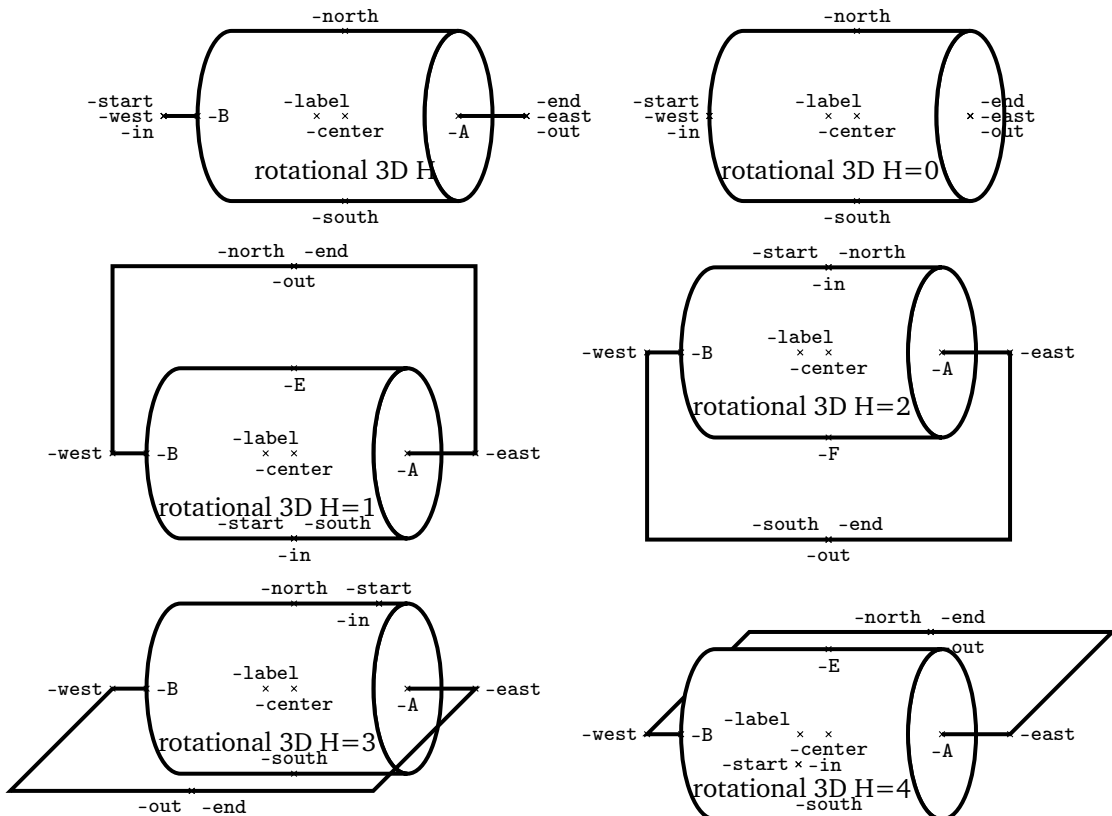
```

\centering
\begin{tikzpicture}
\pic (MyR0) {rotational 3D H};
\foreach \n in {0,...,4}
{
\pic [xshift=(\n+1)*2.25cm] (MyR{\n+1}) {rotational 3D H=\n};
\node at (MyR{\n+1}-center) {\n};
}
\end{tikzpicture}

```

The six possible configurations and their respective node points are shown next. Many of those node names remain unchanged throughout the variants, but some, like `-in` or `-out`, change accordingly with the context.

Anyway, the programmer will decide which node suits best the purpose of the application for the sake of anchoring points and attaching links.



### 4.23 The rotational 3D P

The **rotational 3D P** is a representation of a rotational/revolute joint exhibiting a 3D look in perspective. It accepts 6 different configurations to reflect different 3D layouts depending on the

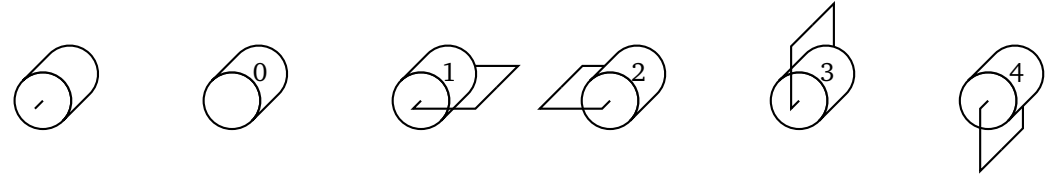


preferred option: with or without an extra arm structure, as occurs in variate literature. Among many other attaching nodes, the joint has a `-in` and a `-out` node whose positions depend on the configuration adopted. The six configurations allowed are defined by the first argument of the `pic`.

### Parameters

- number: type of configuration with these values:
  - No argument: simple joint with the extremes
  - 0: no ends. A simple cylinder without joint extremes
  - 1: arm at right
  - 2: arm at left
  - 3: arm at top
  - 4: arm at bottom

rotational 3D P




---

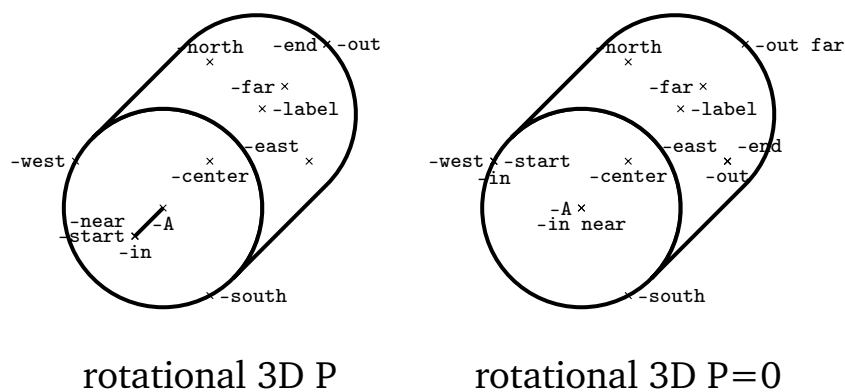
```

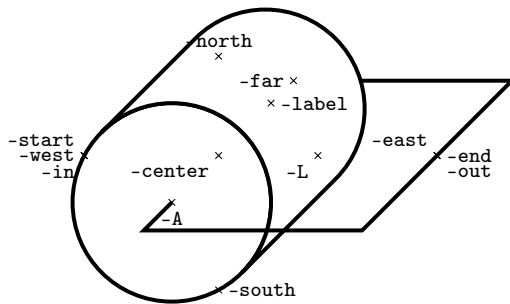
\centering
\begin{tikzpicture}
\pic (MyRO) {rotational 3D P};
\foreach \n in {0,...,4}
{
  \pic[xshift=(\n+1)*2.5cm] (MyR{\n+1}) {rotational 3D P=\n};
  \node at (MyR{\n+1}-label) {\n};
}
\end{tikzpicture}

```

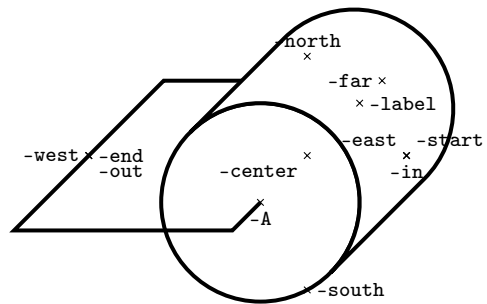
The six possible configurations and their respective node points are shown next. Many of those node names remain unchanged throughout the variants, but some, like `-in` or `-out`, change accordingly with the context. Many nodes have equivalent synonyms but not all of them are shown simultaneously in the images.

Anyway, the programmer will decide which node suits best the purpose of the application for the sake of anchoring points and attaching links.

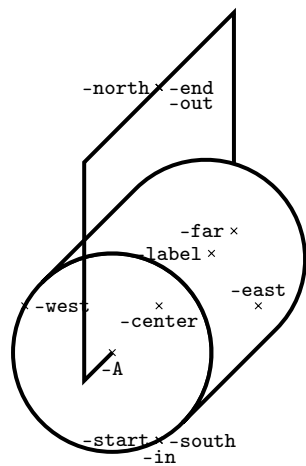




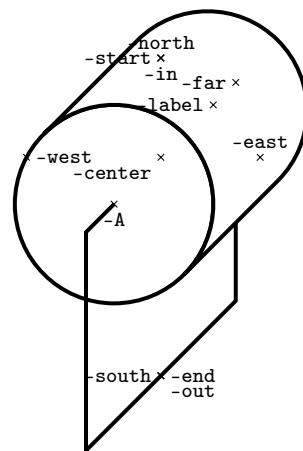
rotational 3D P=1



rotational 3D P=2



rotational 3D P=3



rotational 3D P=4

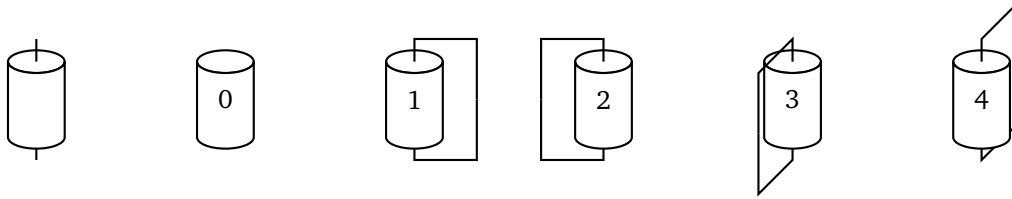
#### 4.24 The *rotational 3D V*

The *rotational 3D V* is a representation of a rotational/revolute joint exhibiting a 3D look in vertical arrangement. It is actually a 90° rotation of the *rotational 3D V* symbol, but anchor nodes are renamed to better match the orientation. It accepts 6 different configurations to reflect different 3D layouts depending on the preferred option: with or without an extra arm structure, as occurs in variate literature. Among many other attaching nodes, the joint has a *-in* and a *-out* node whose positions depend on the configuration adopted. The six configurations allowed are defined by the first argument of the pic.

##### Parameters

1. number: type of configuration with these values:
  - No argument: simple joint with two extremes
  - 0: no ends. A simple cylinder
  - 1: arm at left
  - 2: arm at right
  - 3: arm at near (front/closer)
  - 4: arm at far (back of page)

## rotational 3D V



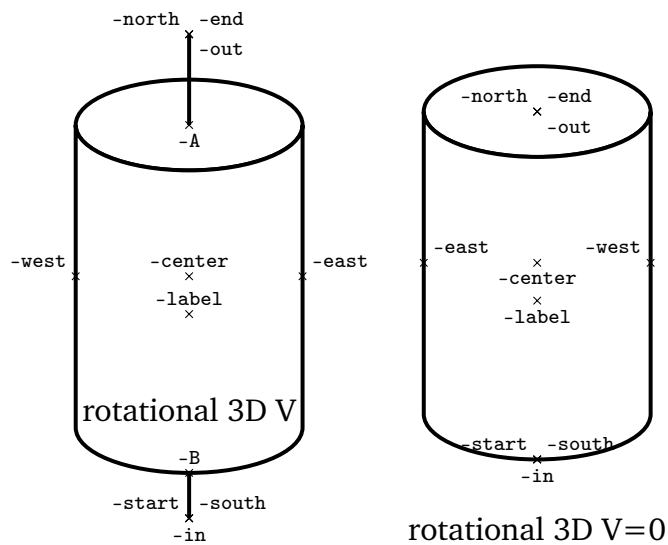
```

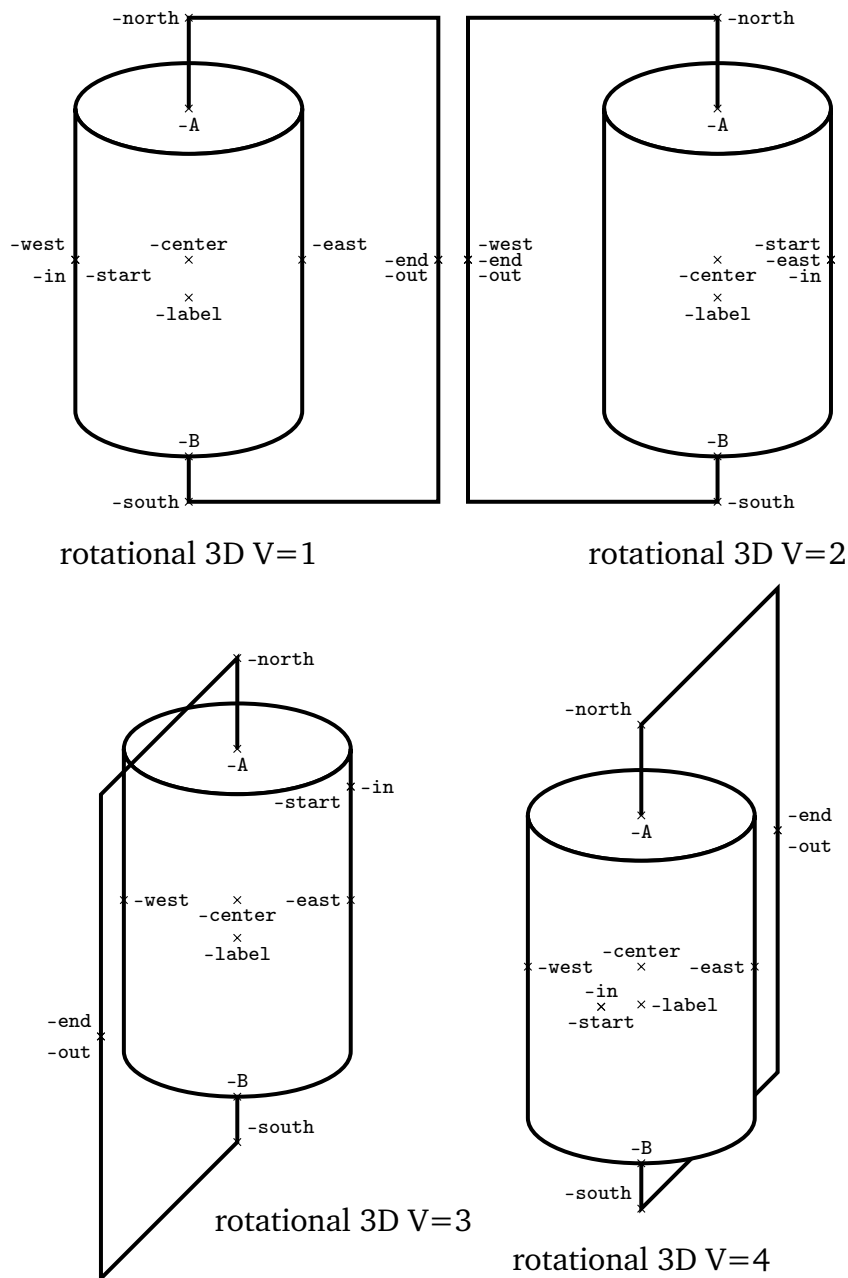
\centering
\begin{tikzpicture}
\pic (MyR0) {rotational 3D V};
\foreach \n in {0,...,4}
{
  \pic[xshift=(\n+1)*2.5cm] (MyR{\n+1}) {rotational 3D V=\n};
  \node at (MyR{\n+1}-center) {\n};
}
\end{tikzpicture}

```

The six possible configurations and their respective node points are shown next. Many of those node names remain unchanged throughout the variants, but some, like `-in` or `-out`, change accordingly with the context. Many nodes have equivalent synonyms but not all of them are shown simultaneously in the images.

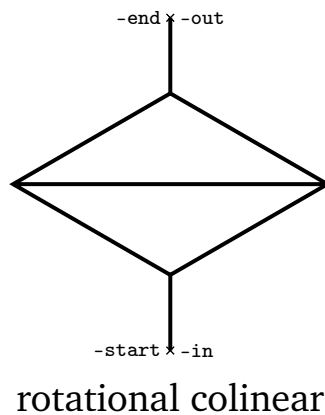
Anyway, the programmer will decide which node suits best the purpose of the application for the sake of anchoring points and attaching links.





#### 4.25 The *rotational colinear*

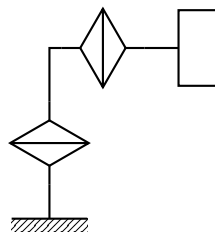
The **rotational colinear** is an alternative representation of a rotational joint with the axis colinear with the links or, more explicitly, a rotational joint with the axis in the plane of the paper. It is a rarer representation in the literature of the most common **rotational 3D V** or **rotational 3D H** described in this package. The default representation is in vertical mode but a simple TikZ rotation of any angle value will re-orient in at will.



### Parameters

The object has no specific parameters.

Illustration of usage:



```

rotational colinear
\begin{tikzpicture}[node distance=1cm and 0.5cm, knLinkStyle]
\pic (B) {frame};
\pic [above of=B-out] (J1) {rotational colinear};
\pic [above right of=J1-out, rotate=-90] (J2) {rotational colinear};
\pic [right of=J2-center] (GR) {gripper};
\draw (B-out) -- (J1-in)
      (J1-out) |- (J2-in)
      (J2-out) -- (GR-in);
\end{tikzpicture}

```

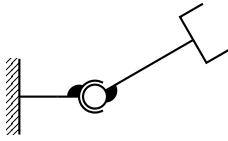
## 4.26 The spherical pair

The **spherical pair** is the item 2.3.1 of the ISO standard, which is a lower kinematic pair with 3 DOF. It is sometimes also called **ball-and-socket**.



### Parameters

1. number: angle orientation of end arm relatively to the input arm (default: 15).
2. number: an optional scale factor of ball diameter (default: 1).
3. dimension: length of input arm (default: 0.8cm).
4. dimension: length of output arm (default: 0.8cm).
5. boolean: (1) use `-in` internal node as anchor point instead of `-center` (default: 0).



### spherical pair

```








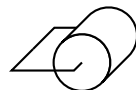

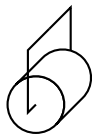

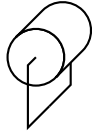
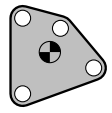
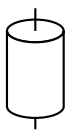

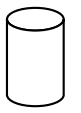
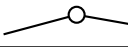
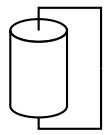

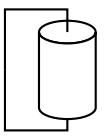
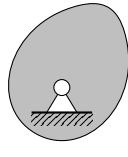
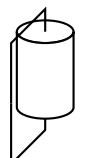



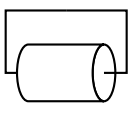
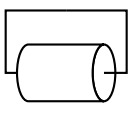
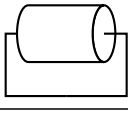
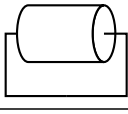


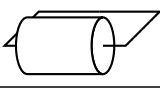
\begin{tikzpicture}[knLinkStyle]
\pic [rotate=-90] (F) {frame};
\pic [right of=F-out] (MyL) {spherical pair=30/1.5/0.5cm/1.5cm};
\pic [rotate=30,scale=0.7] at (MyL-out) {gripper};
\draw (F-out) -- (MyL-in);
\end{tikzpicture}

```

## 5 List of available symbols

The list of symbols includes the main symbol (without any arguments) plus the variations that occur in several symbols, especially in the 3D variants.

cam follower		frame pivot flat	
cam follower=1		frame pivot rounded	
cam follower=2		frame pivot trapezium	
cam follower=3		frame pivot triangle	
cam follower=4		gripper	
campin		helical pair	
center of mass		linear 3D	
cylindrical pair		linear 3D=1	
cylindrical pair F		linear 3D=2	
frame		linear 3D=3	
frame 3D		linear 3D=4	
frame dual pivot slide		linear 3D=5	

linear 3D=6		rotational 3D P	
linear joint bar		rotational 3D P=0	
linear joint bar=/1/1		rotational 3D P=1	
linear piston		rotational 3D P=2	
link bar generic		rotational 3D P=3	
link polygon		rotational 3D P=4	
link polygon=20:1/60:1/90:1		rotational 3D V	
prismatic pair		rotational 3D V=0	
revolute pair planar		rotational 3D V=1	
revolute pair spatial		rotational 3D V=2	
rotating cam plate		rotational 3D V=3	
rotational 3D H		rotational 3D H=0	
rotational 3D H=0		rotational 3D H=1	
rotational 3D H=1		rotational 3D H=2	
rotational 3D H=2		rotational 3D H=3	
rotational 3D H=3		rotational 3D H=4	
rotational 3D H=4	