The `liftarm` package

Draw liftarms with Ti*k*Z

Matthias Floré

Version 2.0 (2022/04/07)

Abstract

This package is based on the package `tikz` (see [5]) and can be used to draw liftarms with Ti*k*Z. It provides several options for the appearance of the liftarms, a command which connects two liftarms, an environment to describe a construction and a method to animate a construction with one or more traces.

Contents

1 Usage

The package `liftarm` can be used by putting the following in the preamble.

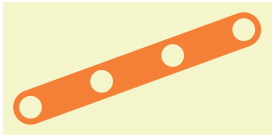
```
\usepackage{liftarm}
```

The package `liftarm` loads the packages `etoolbox`, `xcolor` with the option `dvipsnames`, `tikz` and the Ti*k*Z library `calc`. Since `xcolor` is loaded with the option `dvipsnames`, packages such as `pgfplots` and `tcolorbox` must be loaded *after* `liftarm`.

2 Drawing liftarms

`\liftarm[`⟨*options*⟩`]{`⟨*point*⟩`}{`⟨*length*⟩`}{`⟨*angle*⟩`}`

This command can be placed inside a `tikzpicture` environment. It draws a liftarm of ⟨*length*⟩ starting at ⟨*point*⟩. The angle between the liftarm and the x-axis can be specified by ⟨*angle*⟩ in degrees. The distance between the holes is 1.



```
\begin{tikzpicture}
\liftarm{1,2}{3}{20}
\end{tikzpicture}
```

Note that the number of holes is ⟨*length*⟩ + 1. The ⟨*options*⟩ can be given with the following keys.

`/liftarm/axle holes={`⟨*values*⟩`}` (no default)

This key defines the holes in the liftarm where axle holes will be drawn.

```
\begin{tikzpicture}
\liftarm[axle holes={0,4}]{0,1}{4}{0}
\end{tikzpicture}
```

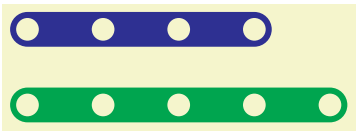
`/liftarm/brick=`⟨*boolean*⟩ (default `true`, `initially false`)

If true, a brick will be drawn instead of a liftarm.


```
\begin{tikzpicture}
\liftarm[brick]{0,1}{2}{0}
\end{tikzpicture}
```

`/liftarm/color={`⟨*name*⟩`}` (no default)

This key defines the color of the liftarm. The color can also be specified without key.



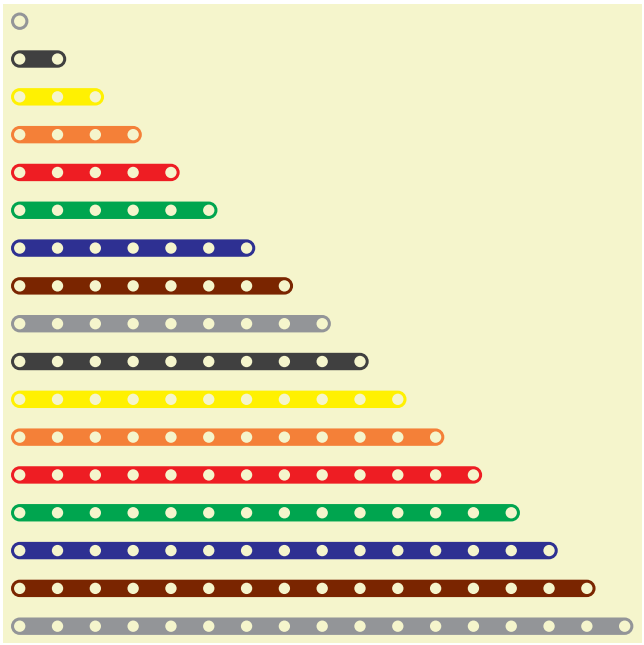
```
\begin{tikzpicture}
\liftarm[color=Green]{0,1}{4}{0}
\liftarm[Blue]{0,2}{3}{0}
\end{tikzpicture}
```

`/liftarm/color 0={`⟨*name*⟩`}` (no default, initially `Gray`)
`/liftarm/color 1={`⟨*name*⟩`}` (no default, initially `darkgray`)
`/liftarm/color 2={`⟨*name*⟩`}` (no default, initially `Yellow`)
`/liftarm/color 3={`⟨*name*⟩`}` (no default, initially `Orange`)
`/liftarm/color 4={`⟨*name*⟩`}` (no default, initially `Red`)
`/liftarm/color 5={`⟨*name*⟩`}` (no default, initially `Green`)
`/liftarm/color 6={`⟨*name*⟩`}` (no default, initially `Blue`)
`/liftarm/color 7={`⟨*name*⟩`}` (no default, initially `Brown`)

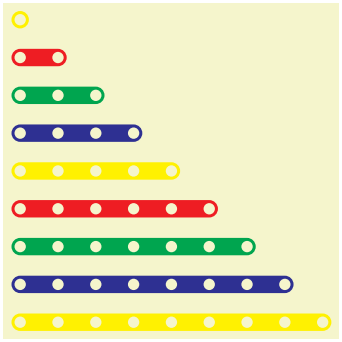
These keys define the colors of the liftarms which have as their length the number following `color`.

`/liftarm/color modulo={`⟨*number*⟩`}` (no default, initially `8`)

The default colors of the liftarms are determined by computing the length of the liftarm modulo the value of this key and selecting the color from the previous keys.



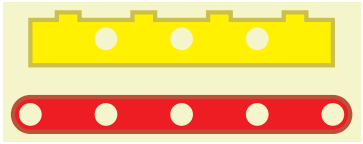
```
\begin{tikzpicture}[scale=0.5]
\foreach\n in {0,...,16}{
    \liftarm{0,-\n}{\n}{0}
}
\end{tikzpicture}
```



```
\begin{tikzpicture}[scale=0.5]
\pgfkeys{
    /liftarm,
    color 0=Yellow,
    color 1=Red,
    color 2=Green,
    color 3=Blue,
    color modulo=4
}
\foreach\n in {0,...,8}{
    \liftarm{0,-\n}{\n}{0}
}
\end{tikzpicture}
```

`/liftarm/contour`=⟨*boolean*⟩ (default `true`, `initially false`)

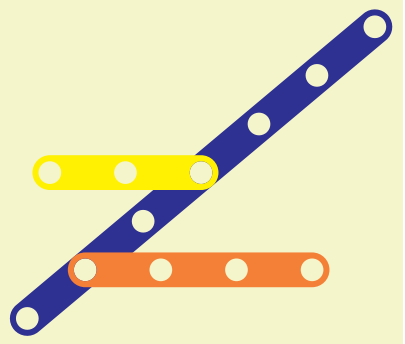
If true, a contour will be drawn around the liftarm.



```
\begin{tikzpicture}
\liftarm[contour]{0,1}{4}{0}
\liftarm[brick,contour]{1,2}{2}{0}
\end{tikzpicture}
```

`/liftarm/coordinate`={⟨*number 1*/*name 1*⟩}... (no default)

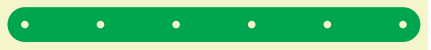
This key defines coordinates with name ⟨*name i*⟩ at hole ⟨*number i*⟩ of the liftarm.



```
\begin{tikzpicture}
\liftarm[
    coordinate={1/A,3/B}
]{0,1}{6}{40}
\liftarm{A}{3}{0}
\liftarm{B}{2}{180}
\end{tikzpicture}
```

`/liftarm/hole radius`={⟨*value*⟩} (no default, initially `0.3`)

The ⟨*value*⟩ of this key, multiplied with the ⟨*value*⟩ of the key `scalefactor` defines the radius of the holes.



```
\begin{tikzpicture}
\liftarm[hole radius=0.1]{0,0}{5}{0}
\end{tikzpicture}
```

`/liftarm/liftarm thickness`={⟨*value*⟩} (no default, initially `0.92`)

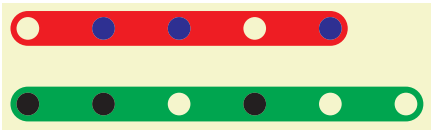
The ⟨*value*⟩ of this key, multiplied with the ⟨*value*⟩ of the key `scalefactor` defines the thickness of the liftarm.


```
\begin{tikzpicture}
\liftarm[
    hole radius=0.1,
    liftarm thickness=0.3
]{0,0}{5}{0}
\end{tikzpicture}
```

`/liftarm/mark color`={⟨*name*⟩} (no default, initially `Black`)

`/liftarm/mark holes`={⟨*values*⟩} (no default)

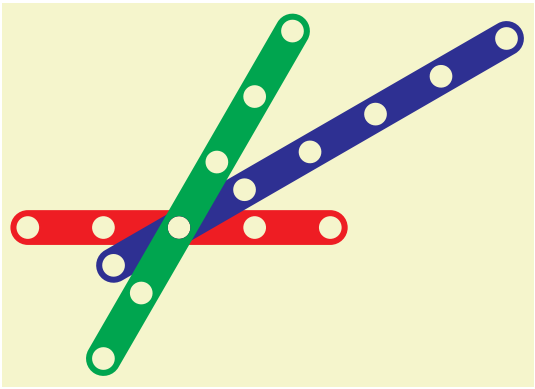
The key `mark holes` defines the holes in the liftarm which will be marked. The key `mark color` defines the color of these marks.



```
\begin{tikzpicture}
\liftarm[
    mark holes={0,1,3}
]{0,0}{5}{0}
\liftarm[
    mark holes={1,2,4},
    mark color=Blue
]{0,1}{4}{0}
\end{tikzpicture}
```

/liftarm/origin={⟨*number*⟩} (no default, initially 0)

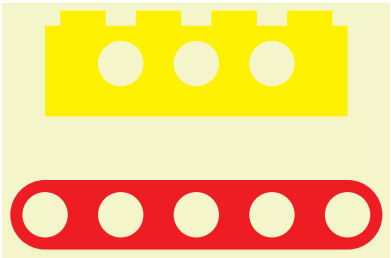
This key defines the number of the hole which will be placed at the coordinate given as argument to the liftarm.



```
\begin{tikzpicture}
\liftarm{-2,0}{4}{0}
\liftarm[origin=1]{0,0}{6}{30}
\liftarm[origin=2]{0,0}{5}{60}
\end{tikzpicture}
```

/liftarm/scalefactor={⟨*value*⟩} (no default, initially 0.5)

The ⟨*value*⟩ of this key defines the factor which scales the thickness of the liftarm and the radius of the holes.



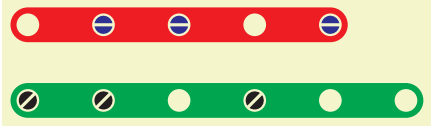
```
\begin{tikzpicture}
\liftarm[scalefactor=1]{0,0}{4}{0}
\liftarm[brick,scalefactor=1]{1,2}{2}{0}
\end{tikzpicture}
```

/liftarm/screw color={⟨*name*⟩} (no default, initially Black)

/liftarm/screw holes={⟨*values*⟩} (no default)

/liftarm/screw holes angle={⟨*angle*⟩} (no default, initially 45)

The key `screw holes` defines the holes in the liftarm where a screw will be drawn. The key `screw color` defines the color of these screws. The key `screw holes angle` defines the angle in degrees around which the screws are drawn.

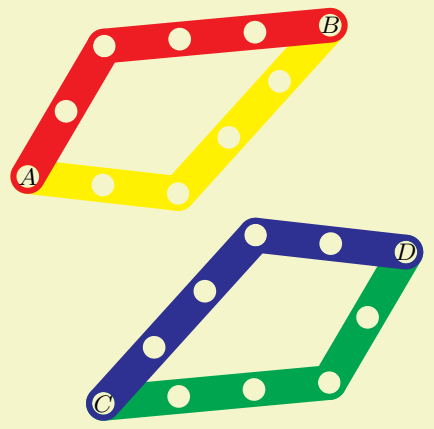


```
\begin{tikzpicture}
\liftarm[
    screw holes={0,1,3}
]{0,0}{5}{0}
\liftarm[
    screw holes={1,2,4},
    screw color=Blue,
    screw holes angle=0
]{0,1}{4}{0}
\end{tikzpicture}
```

3 Connecting liftarms

`\liftarmconnect[`⟨*options*⟩`]{`⟨*point1*⟩`}{`⟨*length1*⟩`}{`⟨*point2*⟩`}{`⟨*length2*⟩`}`

This command can be placed inside a `tikzpicture` environment. It draws a liftarm of ⟨*length1*⟩ starting at ⟨*point1*⟩ and a liftarm of ⟨*length2*⟩ starting at ⟨*point2*⟩ in such a way that their last holes have the same coordinate in case that such a point exists. If such a point does not exist then nothing is drawn. In case that there exist 2 such points then this point is chosen counterclockwise. In that case, the other configuration of the 2 liftarms can be obtained by simply swapping `{`⟨*point1*⟩`}{`⟨*length1*⟩`}` and `{`⟨*point2*⟩`}{`⟨*length2*⟩`}`. The keys for the command `\liftarm` can be given to the ⟨*options*⟩. In this case these keys will be passed to both liftarms.

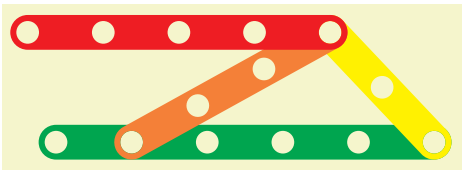



```
\begin{tikzpicture}
\coordinate (A) at (0,0);
\coordinate (B) at (4,2);
\coordinate (C) at (1,-3);
\coordinate (D) at (5,-1);
\liftarmconnect[Yellow]{A}{2}{B}{3}
\liftarmconnect[Red]{B}{3}{A}{2}
\liftarmconnect[Green]{C}{3}{D}{2}
\liftarmconnect[Blue]{D}{2}{C}{3}
\foreach\coord in {A,B,C,D}{
    \node at (\coord) {{\small $\coord$}};
}
\end{tikzpicture}
```

Additionally, the ⟨*options*⟩ can be given with the following keys.

`/liftarm/connect coordinate=`{⟨*name*⟩} (no default)

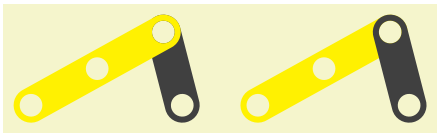
This key defines a coordinate with name ⟨*name*⟩ at the connection point of both liftarms.



```
\begin{tikzpicture}
\liftarm{-3,0}{5}{0}
\liftarmconnect[
    connect coordinate=A
]{2,0}{2}{-2,0}{3}
\liftarm{A}{4}{180}
\end{tikzpicture}
```

`/liftarm/connect reverse=`⟨*boolean*⟩ (default `true, initially false`)

If true, the first liftarm of `\liftarmconnect` will be drawn second and the second liftarm will be drawn first. This option can be used to change the appearance at the connection point of both liftarms.



```
\begin{tikzpicture}
\liftarmconnect{2,0}{1}{0,0}{2}
\liftarmconnect[
    connect reverse
]{5,0}{1}{3,0}{2}
\end{tikzpicture}
```

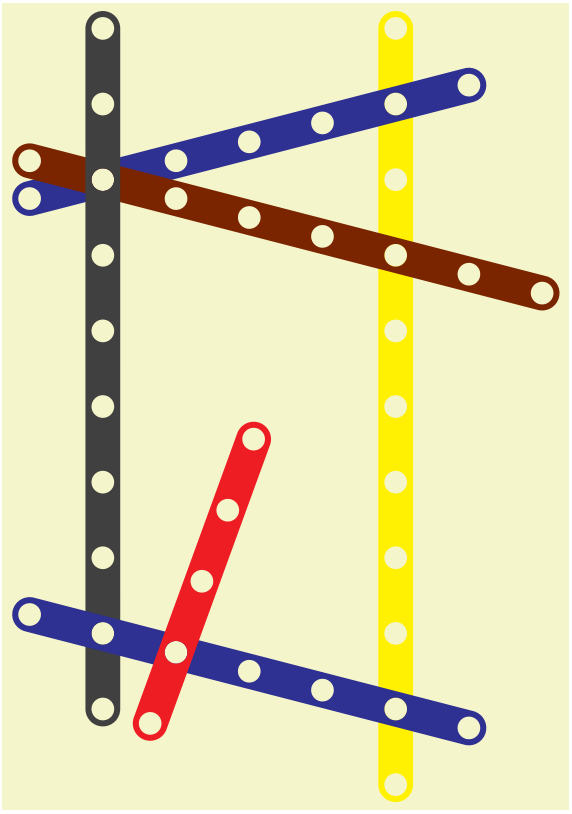
`/liftarm/liftarm 1=`{⟨*options*⟩} (style, no default, initially empty)

`/liftarm/liftarm 2=`{⟨*options*⟩} (style, no default, initially empty)

These keys accept a list of keys which will be applied to the first respectively second liftarm. These lists of keys accept the same options as the command `\liftarm`. Additionally, the key `connect` below can be given.

`/liftarm/connect=`{⟨*number*⟩} (no default)

This key defines the number of the hole which will be connected to the matching liftarm. If this key is not given then the last hole of the liftarm is taken as the connecting point.



```
\begin{tikzpicture}
\liftarm{0,-7}{10}{90}
\liftarmconnect[
    connect coordinate=A,
    liftarm 1={
        origin=1,
        connect=5
    },
    liftarm 2={
        origin=2,
        connect=6
    }
]{0,2}{6}{0,0}{7}
\liftarmconnect[
    liftarm 1={
        origin=2,
        connect=8
    },
    liftarm 2={
        origin=1,
        connect=5,
        coordinate=4/B
    }
]{A}{9}{0,-6}{6}
\liftarm[origin=1]{B}{4}{70}
\end{tikzpicture}
```

4 Describing a construction

If a construction involves many liftarms then it is convenient to describe this construction in separate steps and `tikzpictures`. Then the content of previous `tikzpictures` would need to be copied in each new `tikzpicture`. This process can be automated by using the `liftarmconstruction` environment and the command `\liftarmconstruct` below.

`\begin{liftarmconstruction}[`⟨*options*⟩`]`
⟨*environment contents*⟩
`\end{liftarmconstruction}`

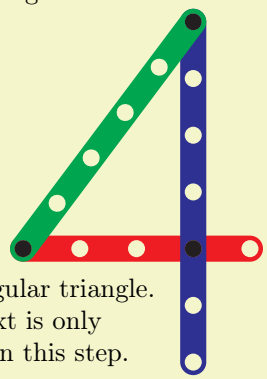
This environment is in fact an `enumerate` environment with the addition that it resets the content of the `tikzpicture` which is displayed by the command `\liftarmconstruct` below. Thus in particular, `\item` can be used inside the `liftarmconstruction` environment. The ⟨*options*⟩ will be passed to each `tikzpicture` drawn by the command `\liftarmconstruct` inside this environment. The following command can be used inside this environment.

`\liftarmconstruct[`⟨*options*⟩`]{`⟨*text*⟩`}{`⟨*commands*⟩`}`

This command starts an `\item` and shows ⟨*text*⟩. Then it displays a `tikzpicture` containing ⟨*commands*⟩ and also the ⟨*commands*⟩ of previous `\liftarmconstruct` commands inside the same `liftarmconstruction` environment. The ⟨*options*⟩ will be added to this `tikzpicture` but *only* in the current step.

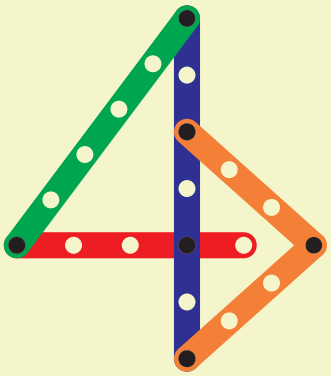
As an example, we describe below the construction of a regular pentagon from [1].

1. We start with 3 liftarms to form a rectangular triangle.

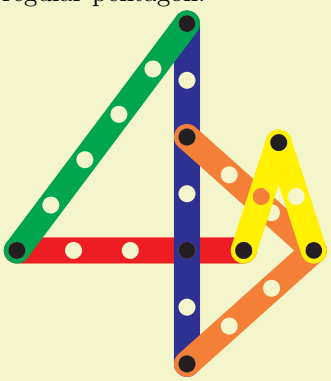


Rectangular triangle.
This text is only
visible in this step.

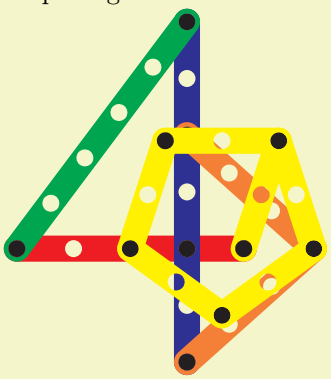
2. An `\item` can be added since this is an `enumerate` environment.
3. Now we add 2 liftarms of length 3.



4. In this step we construct the first side of the regular pentagon.



5. Now we finish the construction of the regular pentagon.



```
\begin{minipage}{0.5\linewidth}%only for
%usage in this manual%\linewidth-6pt
%\begin{multicols}{2}%only for
%usage in this manual
\begin{liftarmconstruction}[scale=0.75]
\liftarmconstruct[
    {\node[left,align=left]
        at (-0.5,-1.3)
        {Rectangular triangle.\\
        This text is only\\
        visible in this step.};}
]{
    We start with 3 liftarms to form
    a rectangular triangle.
}{
\liftarm{-3,0}{4}{0}
\liftarmconnect[
    liftarm 1={
        origin=2,
        mark holes={2,6}
    },
    liftarm 2={
        mark holes=0
    }
]{0,0}{6}{-3,0}{5}}
\item An |\item| can be added since this
    is an |enumerate| environment.
\liftarmconstruct{
    Now we add 2 liftarms of length $3$.
}{\liftarmconnect[
    connect coordinate=A,
    liftarm 1={
        mark holes={0,3}
    },
    liftarm 2={
        mark holes=0
    }
]{0,-2}{3}{0,2}{3}}
\liftarmconstruct{
    In this step we construct the first
    side of the regular pentagon.
}{\liftarmconnect[
    connect coordinate=B,
    liftarm 2={
        mark holes={0,2}
    }
]{A}{2}{1,0}{2}}
\liftarmconstruct{
    Now we finish the construction
    of the regular pentagon.
}{\liftarmconnect[
    liftarm 2={
        mark holes={0,2}
    }
]{B}{2}{-1,0}{2}
\liftarmconnect[
    liftarm 1={
        mark holes=2
    }
]{-1,0}{2}{A}{2}}
\end{liftarmconstruction}
%\end{multicols}
\end{minipage}
```

5 Animations

`\liftarmanimate[`⟨*options*⟩`]{`⟨*frame rate*⟩`}{`⟨*list*⟩`}{`⟨*command*⟩`}`

This command shows an animation using the **animateinline** environment of the package **animate**. The package **animate** is *not* loaded by default and needs to be loaded to use the command **\liftarmanimate**. The ⟨*options*⟩ are passed to the **animateinline** environment. The ⟨*frame rate*⟩ of the animation is described in the documentation of the package **animate**. The ⟨*command*⟩ must be a previously defined command with one mandatory argument. The ⟨*list*⟩ is passed to a **\foreach** loop. The frames of the animation consist of the ⟨*command*⟩ evaluated one by one in the result of the **\foreach** loop. The command **\liftarmanimate** creates a timeline which is used in the **animateinline** environment. This timeline is stored in the file **liftarm**⟨*number of the animation in the document*⟩**.tln**. It requires two compiler runs to create and use this timeline correctly.

`/liftarm/trace={`⟨*number/number of frames/code*⟩`}...` (no default)

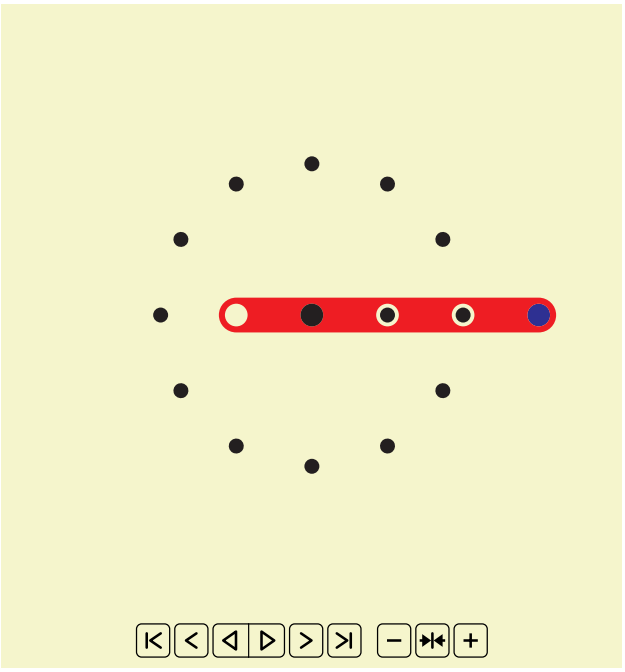
This key draws ⟨*code*⟩ at hole ⟨*number*⟩ of the liftarm on the frames of the animation determined by ⟨*number of frames*⟩.

If ⟨*number of frames*⟩ is 0 then the ⟨*code*⟩ is drawn starting at the current frame until the end of the animation. If ⟨*number of frames*⟩ is an integer greater than or equal to 1 then the ⟨*code*⟩ is drawn starting at the current frame and remaining during the next frames determined by ⟨*number of frames*⟩. If ⟨*number of frames*⟩ is left empty then the ⟨*code*⟩ is drawn starting at the beginning of the animation until the end of the animation.

The ⟨*code*⟩ can be some Ti*k*Z code. In this ⟨*code*⟩, (0, 0) is positioned at hole ⟨*number*⟩ of the liftarm. If ⟨*code*⟩ is left empty then the following code is used.

```
\fill[Black] (0,0) circle[radius=0.66*\liftarm@holeradius];
```

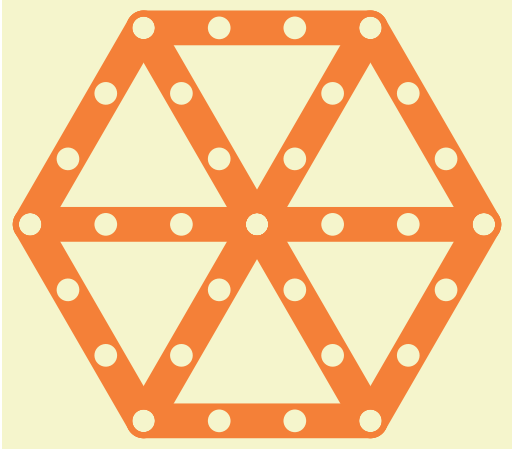
A list of multiple triples ⟨*number/number of frames/code*⟩ can be given to the key **trace**.



```
\usepackage {animate}
\newcommand{\exampleliftarmanimate}[1]{
    \liftarm[
        origin=1,
        mark holes=1,
        trace={
            2/0/,
            3//,
            4/3/{\fill[Blue] (0,0)
                circle[radius=0.15];}
        }
    ]{0,0}{4}{#1}
}
\liftarmanimate[
    autoplay,
    controls,
    loop,
    begin={
        \begin{tikzpicture}
        \useasboundingbox (-4,-4)
            rectangle (4,4);
    },
    end={\end{tikzpicture}}
]
{5}
{0,30,...,330}
{\exampleliftarmanimate}
```

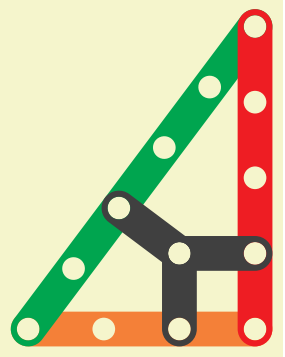
6 Additional examples

The following example shows a regular hexagon.



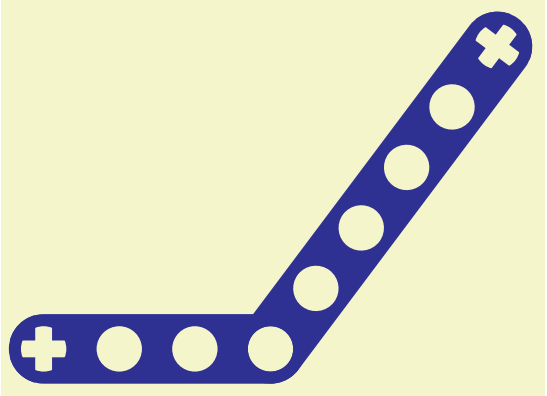
```
\begin{tikzpicture}
\def\r{3}
\foreach\n in {1,...,6}{
    \liftarmconnect{0,0}{\r}{\n*60:\r}{\r}
}
\end{tikzpicture}
```

The following example illustrates that $2\,\mathrm{atan}(\frac{1}{2}) = \mathrm{atan}(\frac{4}{3})$.



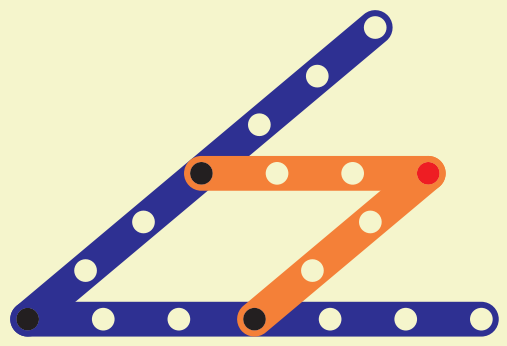
```
\begin{tikzpicture}
\liftarm{0,0}{3}{0}
\liftarm{0,0}{5}{atan(4/3)}
\liftarm{3,0}{4}{90}
\liftarm{2,0}{1}{90}
\liftarm{2,1}{1}{0}
\liftarm{2,1}{1}{90+atan(4/3)}
\end{tikzpicture}
```

Below is an example of an angled liftarm.



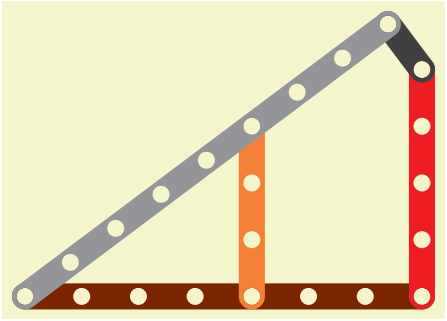
```
\begin{tikzpicture}
\pgfkeys{
    /liftarm,
    scalefactor=1,
    Blue
}
\liftarm[axle holes=0]{0,0}{3}{0}
\liftarm[axle holes=5]{3,0}{5}{atan(4/3)}
\end{tikzpicture}
```

The following example illustrates an angle bisection.



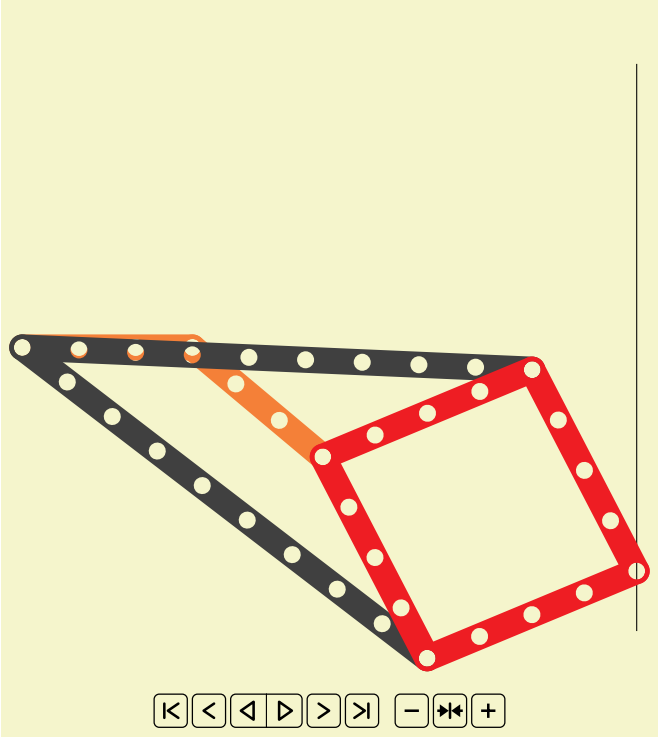
```
\begin{tikzpicture}
\def\ang{40}
\def\r{3}
\liftarm[mark holes={0,\r}]{0,0}{2*\r}{0}
\liftarm[mark holes=\r]{0,0}{2*\r}{\ang}
\liftarm[
    mark holes=\r,
    mark color=Red
]{\r,0}{\r}{\ang}
\liftarm{\ang:\r}{\r}{0}
\end{tikzpicture}
```

The following example illustrates that $7^2 + 4^2 = 8^2 + 1^2$.



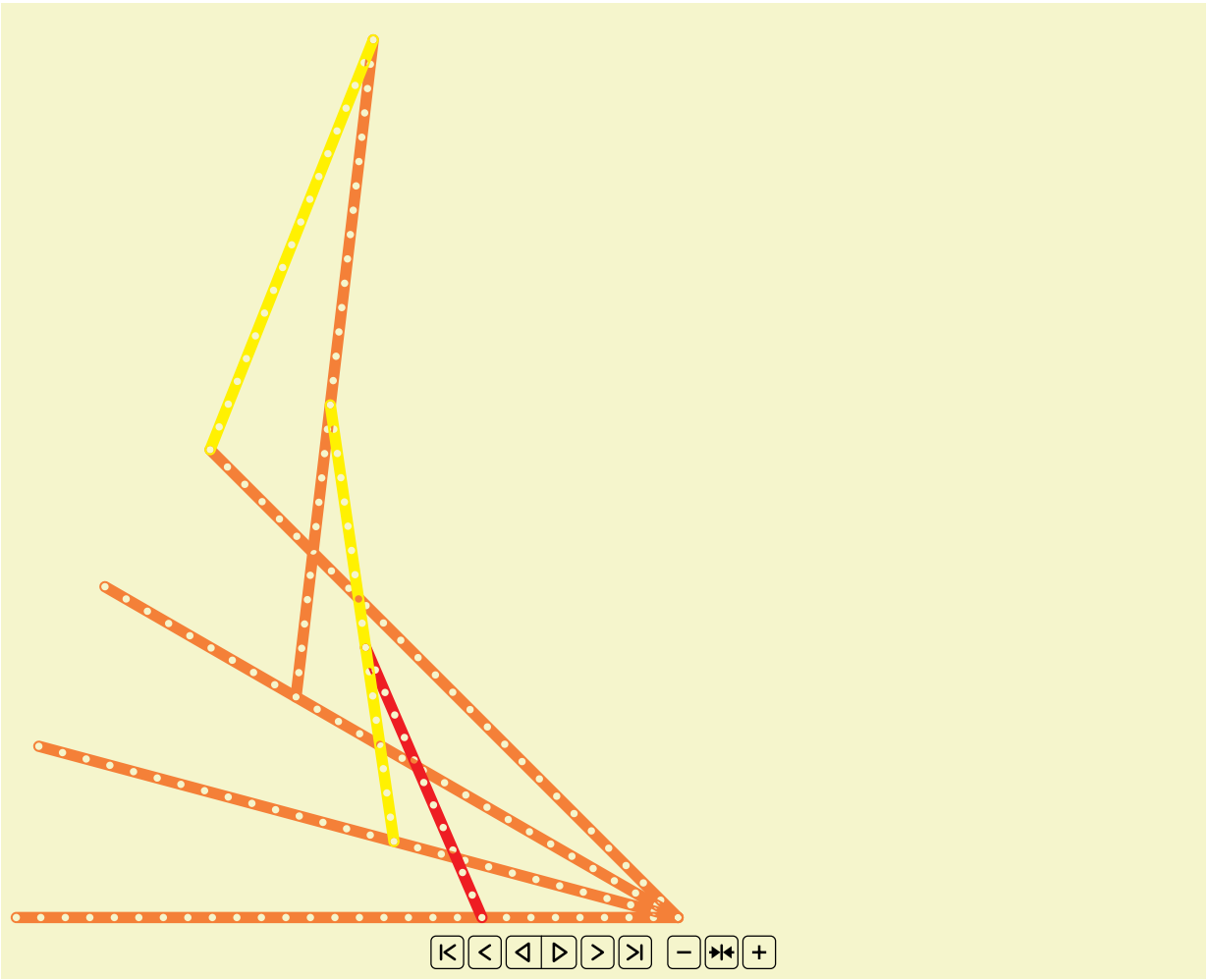
```
\begin{tikzpicture}[scale=0.75]
\def\a{4}
\def\b{7}
\def\c{1}
\def\d{8}
%\liftarm{0,0}{\b}{0}
%\liftarm{\b,0}{\a}{90}
\liftarmconnect{0,0}{\b}{\b,\a}{\a}
\liftarm{4,0}{3}{90}
%\liftarm{\b,\a}{1}{atan(\a/\b)+atan(\c/\d)+90}
%\liftarm{0,0}{\d}{atan(\a/\b)+atan(\c/\d)}
\liftarmconnect{\b,\a}{\c}{0,0}{\d}
\end{tikzpicture}
```

Below is an animation of the Peaucellier-Lipkin linkage, see e.g. [4].



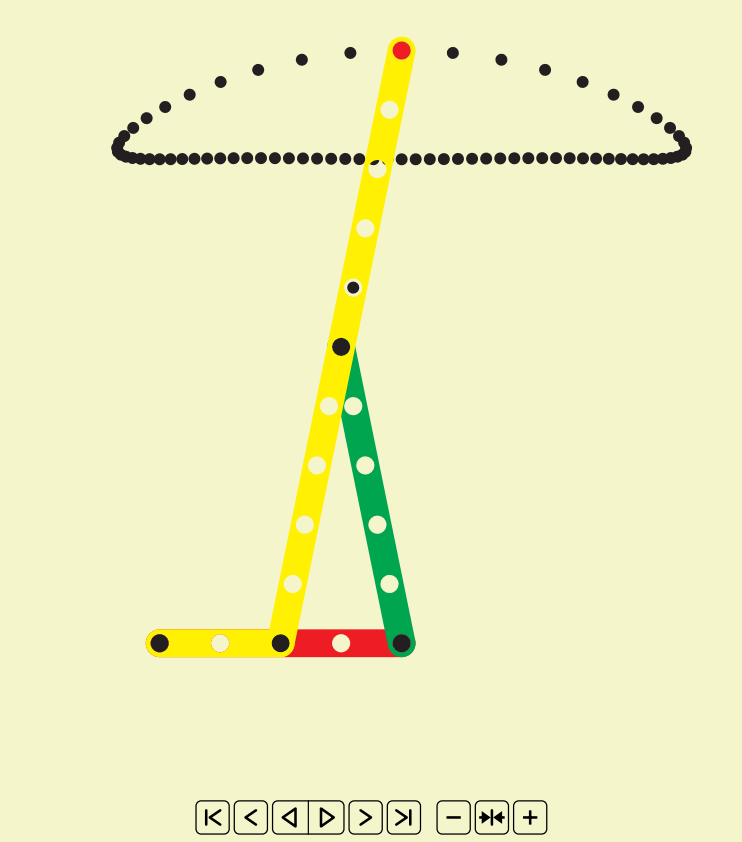
```
\usepackage {animate}
\newcommand{\PLlinkage}[1]{
\begin{tikzpicture}[scale=0.75]
\def\a{3}
\def\b{4}
\def\c{9}
\pgfmathsetmacro{\x}{
    2*\a+((\c^2-\b^2-(2*\a)^2)/(2*\a))
}
\useasboundingbox (-0.23,-6) rectangle
    ({\x+0.23},6);
\draw (\x,-5)--(\x,5);
\liftarm{0,0}{\a}{0}
\liftarm[coordinate=\a/A]{\a,0}{\a}{#1}
\liftarmconnect[
    connect coordinate=B,
    connect reverse
]{A}{\b}{0,0}{\c}
\liftarmconnect[
    connect coordinate=C
]{0,0}{\c}{A}{\b}
\liftarmconnect{C}{\b}{B}{\b}
\end{tikzpicture}
}
\begin{animateinline}[
    autoplay,
    controls,
    palindrome
]{30}
\multiframe{80}{rAng=-40+1}{
    \PLlinkage{\rAng}
}
\end{animateinline}
```

Below is an animation of Kempe's trisector, as shown in [3].



```
\usepackage {animate}
\newcommand{\trisector}[1]{
\begin{tikzpicture}[scale=0.33]
\useasboundingbox (-27.3,-0.5) rectangle (21.2,37);
\liftarm[coordinate=8/A]{0,0}{27}{180}
\liftarm[coordinate=12/B]{0,0}{27}{180-(#1)}
\liftarm[coordinate=18/C]{0,0}{27}{180-2*(#1)}
\liftarm[coordinate=27/D]{0,0}{27}{180-3*(#1)}
\liftarmconnect{C}{27}{D}{18}
\liftarmconnect[liftarm 2={connect=8}]{A}{12}{B}{18}
\end{tikzpicture}
}
\begin{animateinline}[autoplay,controls,palindrome]{5}
\multiframe{20}{rAng=15+1}{
\trisector{\rAng}
}
\end{animateinline}
```

Below is an animation of Chebyshev's Lambda Mechanism.



```
\usepackage {animate}
\newcommand{\CL}[1]{
\liftarm{0,0}{4*\r}{0}
\liftarm[
    mark holes={0,2*\r}
]{0,0}{2*\r}{#1}
\liftarmconnect[
    liftarm 1={mark holes={0,5*\r}},
    liftarm 2={
        connect=5*\r,
        mark holes=10*\r,
        mark color=Red,
        trace={6*\r/0/,10*\r//}
    }
]{4*\r,0}{5*\r}{#1:2*\r}{10*\r}
}
\liftarmanimate[
    autoplay,
    controls,
    loop,
    begin={
        \begin{tikzpicture}[scale=0.8]
        \def\r{1}
        \useasboundingbox
            (-2*\r-0.5,-2*\r-0.5)
            rectangle
            (10*\r-0.5,10*\r+0.5);
    },
    end={\end{tikzpicture}}
]
{20}
{0,5,...,355}
{\CL}
```

7 Version history

Version 1.0 (2022/03/08) First version.

Version 2.0 (2022/04/07) Removed some redundant ; in the code.[1] Added the command `\liftarmanimate` and the key `trace`.

References

[1] Gerard 't Hooft, *Meccano Math I*, https://webspace.science.uu.nl/~hooft101/lectures/meccano.pdf, 2006.

[2] Gerard 't Hooft, *Meccano Math II*, https://webspace.science.uu.nl/~hooft101/lectures/meccano2.pdf, 2008.

[3] Gerard 't Hooft, *Meccano Math III*, https://webspace.science.uu.nl/~hooft101/lectures/meccano3.pdf, 2014.

[4] Alfred Bray Kempe, *On a general method of producing exact rectilinear motion by linkwork*, 1875.

[5] Till Tantau, *The TikZ and PGF Packages*, Manual for version 3.1.9a, https://ctan.org/pkg/pgf, 2021.

[1] Thanks to Denis Bitouzé for pointing this out.

Index