Macros

Some Diagonal Line Hacks

Amy Hendrickson
TEXnology, Inc.

Occasionally the need arises for a slanted or diagonal line. How to do it in TEX? This macro makes it possible and at the same time demonstrates some of the new features of TEX82: loops, counters and conditionals.

The basic mechanism involved is to use a loop to move a dot over and down a specified amount a number of times until the condition that determines the width is met, at which time the loop is discontinued. The distance the dot has moved each time is sufficiently small that the single dot is no longer distinguishable and a slanted line is produced.

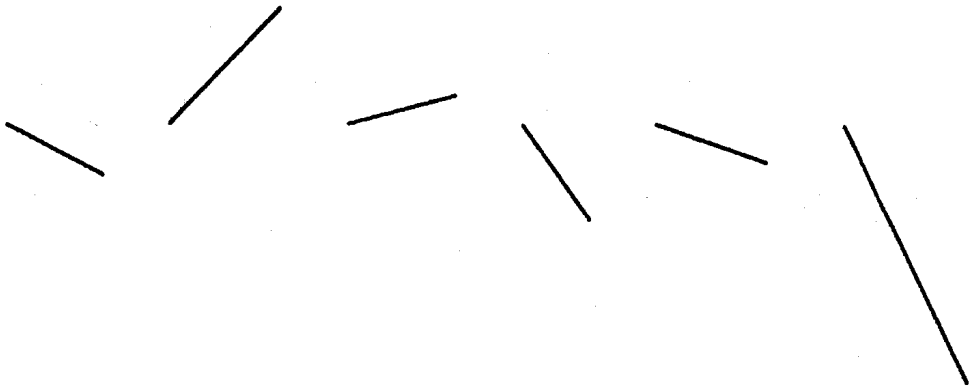
The user determines the slant of the line and the final width of the horizontal space the line will occupy in the \newline macro by specifying the width of the horizontal move each time, (#1), the length of the vertical move each time, (#2) and the final width desired, in points (#3).

Scaled points (sp) are used so that the dot can be moved a fraction of a point (remembering that counters will operate only with whole numbers). Here is the annotated code for \newline:

```
% 65536sp=1pt 72.27pt=1in
%Setting up counters
\newcount\originalvmove
\newcount\hwidth\newcount\numtimes
\newcount\hmove\newcount\vmove
%%%\vmove and \hmove are multiplied by 1000 to increase the
%%% distance traveled each time the loop is iterated
\def\newline#1#2#3{\numtimes=0 \vmove=1000\hmove=1000
\multiply\hmove #1 \multiply\vmove #2 \originalvmove=\the\vmove
%%%Following is the calculation to arrive at the number of
%%% times \hmove must be invoked to arrive at the desired
%%% width (\#3), given to the counter'\hwidth'
\hwidth=#3
\multiply\hwidth by65536%%%(number of sp in pt)
\divide\hwidth by\the\hmove
%%%Now the loop starts, and continues until the counter
%%%'\numtimes' is as large as the counter '\hwidth'
\loop\ifnum\numtimes<\hwidth
\hskip\the\hmove sp\lower\the\vmove sp\hbox
to 0pt{\hss.\hss}\advance\numtimes by 1\advance\vmove by\originalvmove
\repeat}
```

The values of the second argument can be negative as well as positive, yielding a line moving up from its origin instead of down. An example:

```
\newline{40}{20}{35}\qquad\newline{20}{-20}{40}\qquad
\newline{60}{-15}{40}\qquad\newline{30}{40}{25}\qquad\newline{30}{10}{40}
\qquad\newline{30}{60}{45}
```



You can tailor the line to fit a specified space, for instance in a table, by measuring the width you want and playing with the slant of the line until it fits the space correctly. A `-\vskip` is used after `\newline` to make it appear within the table.

First Symbol / Second Symbol		Nonfloating Insertion Symbols									Floating Insertion Symbols						Other Symbols				
		B	O	/	,	.	{+} { }	{+} { } {−}	{CR} { } {−}	CS {DB}	{Z} { }	{Z} { } {*}	{+} { } {*}	{+} { } {−}	CS {−}	A CS	9	S	V	P	P
																	X				
Non-floating Insertion Symbols	B	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	
	O	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	
	/	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	
	,	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	
	.	Y	Y	Y	Y		Y			Y	Y		Y		Y						
	{+−}																				
	{+−} cs	Y	Y	Y	Y	Y Y			Y	Y	Y			Y	Y	Y		Y	Y	Y	

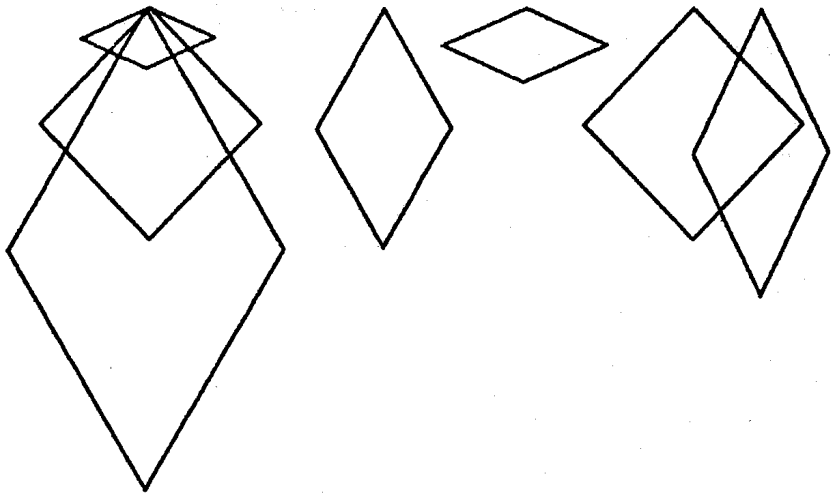
Making Diamond Shapes

A variation on the previous macro will make a diamond shape. This is useful for people making programming flow charts. Once again, the values given to it are for horizontal move (#1), vertical move, (#2), and the width of the whole diamond, in points (#3). If you copy and use this macro, one caution—at some point you can use up all the memory TEX has, so if you need a *really* big diamond shape, or need to use it many times in one file, you may need to increase your system memory.

Here is the code for \diamondline, made by combining four loops, one for each side of the diamond, and with one parameter of the specifications for the height or depth changed for each side, as noted in the comments.

```
\newcount\hwidth
% 65536sp=1pt
\newcount\hmove\newcount\vmove \newcount\originalvmove
\newcount\hwide
\newcount\numtimes
\def\diamondline#1#2#3{\numtimes=0 \vmove=1000\hmove=1000
\hwidth=#3
\multiply\hwidth by65536
\divide\hwidth by2
\multiply\hmove #1 \multiply\vmove #2
\divide\hwidth by\the\hmove
\originalvmove=\the\vmove
\loop\ifnum\numtimes<\hwidth
\hskip\the\hmove sp\lower\the\vmove sp\hbox to
0pt{\hss.\hss}\advance\numtimes by 1\advance\vmove
by\originalvmove \repeat
\numtimes=1
\loop\ifnum\numtimes<\hwidth
\hskip-\the\hmove sp\lower\the\vmove    %%%Notice \hskip-\the\hmove
sp\hbox to 0pt{\hss.\hss}\advance\numtimes %%%to move to the left
by 1\advance\vmove by\originalvmove
\repeat
\numtimes=0
\loop\ifnum\numtimes<\hwidth
\hskip-\the\hmove sp\lower\the\vmove sp\hbox to  %%%\hskip-\the\hmove
0pt{\hss.\hss}\advance\numtimes by           %%%to move to the left,
1\advance\vmove by -\originalvmove %%% This time we also need
\repeat              %%% -\originalmove to make the line move upwards
\numtimes=0
\loop\ifnum\numtimes<\hwidth       %%% This time \hskip is positive but
\hskip\the\hmove sp\lower\the\vmove sp\hbox to %%% we use -\originalmov
0pt{\hss.\hss}\advance\numtimes by  %%% to move up, which finishes
1\advance\vmove by-\originalvmove   %%% the diamond.
\repeat}
```

And here are some sample diamonds...



```
\diamondline{30}{50}{100}\diamondline{70}{30}{50}\diamondline{30}{30}{80}
\hskip80pt\diamondline{30}{50}{50}\hskip50pt\diamondline{70}{30}{60}
\hskip60pt\diamondline{30}{30}{80}\qquad\diamondline{30}{60}{50}
```

To which can be added horizontal and vertical arrows if they are to be used for a flowchart.

* * * * * * * * *

A (Possibly) Totally Useless Macro

Inspired by D. Knuth's prime number generator, I've put together a smaller, and humbler, number generator—this time for numbers in the Fibonacci series. Each new number in the Fibonacci series is made by combining the previous two numbers. It is known as a way of describing certain growth patterns, such as spirals in the cross-section of some seashells and as a measure of the 'golden section', a relationship thought to be especially harmonious by some artists and musicians. This macro will print out the Fibonacci series to as many instances as you call for in the argument (up to 47 that is, then the numbers get too large for TEX to manage).

Here is an example–`\fibonacci{26}`: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, and 121393.

```
\newcount\numbertimes \newcount\numone \newcount\numtwo
\newcount\savenumone \def\fibonacci#1{1, \numbertimes=2  \numone=0
\numtwo=1\loop \advance\numone by \numtwo \the\numone,
\savenumone=\the\numone \numone=\numtwo \numtwo=\savenumone
\advance\numbertimes by1\ifnum\numbertimes<#1\repeat
\ifnum\numbertimes=#1 \advance\numone by \numtwo\fi and \the\numone.}
```