psgo

Typesetting Go Diagrams with PSTricks

Victor Bos
email: v.bos@tue.nl

October 3, 2001

1 Introduction

The psgo package provides functionality to typeset Go diagrams in LaTeX 2_ε. It is built on top of the PSTricks package, which is nowadays available in many LaTeX distributions. Although psgo does not understand the *Smart Game Format* (SGF), it has support for all graphical markup properties of SGF FF[4], see `http://www.red-bean.com/sgf/`.

2 Download and installation

The psgo package can be downloaded from my website:

`http://members.chello.nl/~v.bos01/`

To install the package, download the files `psgo.sty` and `psgomanual.tex` and put them in a directory where LaTeX can find them. After that, test the installation by running LaTeX on `psgomanual.tex`.

3 Go boards

The interface of psgo is based on how Go diagrams are usually displayed in books. That is, the rows of a 19×19 board are indexed by $1, 2, \ldots, 19$ and the columns are indexed by $A, B, \ldots, T$ (skipping I). Further, the lower left corner has index $(A, 1)$ and the upper right corner has index $(T, 19)$. Figure 1 shows a 19×19 board and a 9×9 board. Note that the sizes of the boards are reduced in order to fit in one figure. To re-size a Go board, the command `\setgounit` can be used. This command takes the desired horizontal unit distance as an argument. The default horizontal unit distance is 0.6cm. The vertical unit distance is computed by the psgo package. For the diagrams in this document, except for the diagrams of Figure 1, we have set the horizontal unit distance to 0.4cm (`\setgounit{0.4cm}`).

Go boards are defined in the `psboard` environment of psgo. This environment takes 1 optional parameter which indicates the size of the board (default size is 19). For instance, the boards of Figure 1 were defined by:

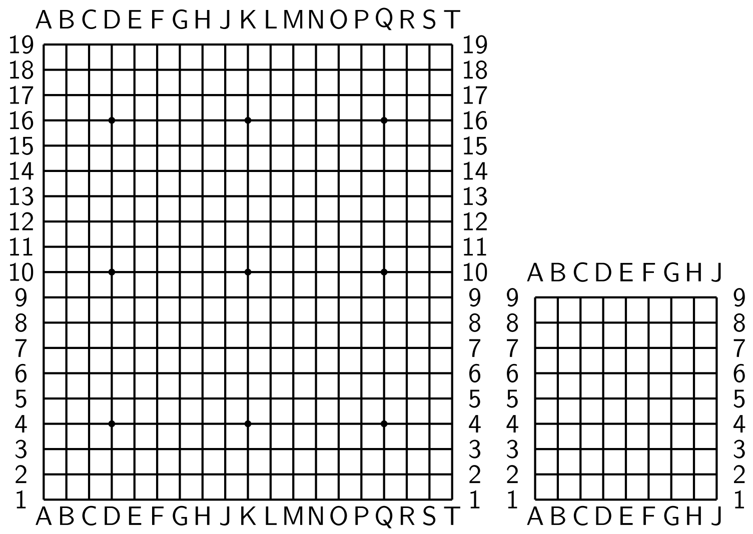


Figure 1: Different size Go boards

```
\begin{psgoboard}
\end{psgoboard}
\begin{psgoboard}[9]
\end{psgoboard}
```

If the indexes are not desired, the starred version `psgoboard*` of the environment should be used, as in

```
\begin{psgoboard*}
\end{psgoboard*}
\begin{psgoboard*}[9]
\end{psgoboard*}
```

4 Stones and moves

There are two commands to put stones on the board. The first one is `\stone` which takes three parameters: the color, the column, and the row of the stone. For example, `\stone{black}{c}{4}` puts a black stone at position $(C, 4)$ (note that in the LaTeX code, the columns are indicated by lower case characters). The `\stone` command can be used to setup a particular configuration. For instance, the configuration of Figure 2 is defined as follows.

```
\begin{psgoboard}[9]
\stone{white}{c}{3}
\stone{white}{e}{3}
\stone{white}{d}{2}
\stone{white}{d}{4}
\stone{black}{f}{3}
\stone{black}{e}{2}
\stone{black}{e}{4}
```

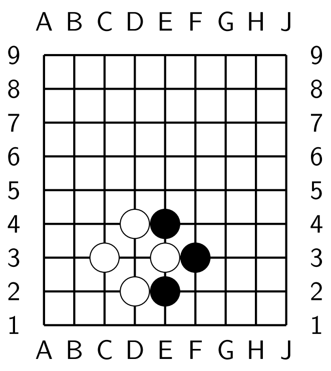


Figure 2: Setting up a configuration

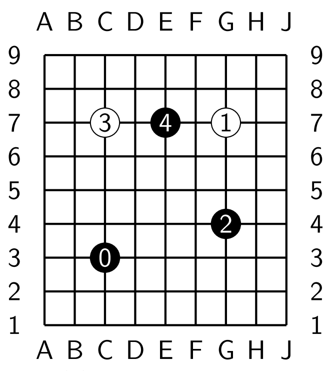
```
\end{psgoboard}
```

The second command is `move` which takes two parameters: the column and the row of the next move. Moves are usually numbered. The counter that keeps track of the move number is called `gomove`. This is a normal LaTeX counter and can be changed using ordinary LaTeX-counter commands. The color of the stones placed by the `\move` command alternates between successive moves. For example, `\move{b}{3}` puts a stone on position $(B, 3)$. If this was a black move, the stone is black, otherwise it is white. It is time for an example. The following code generates a 9×9 board with six moves. The result is depicted in Figure 3(a).

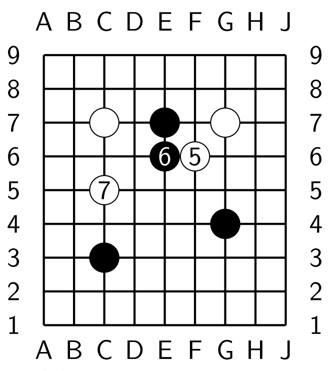
```
\begin{psgoboard}[9]
\move{c}{3}
\move{g}{7}
\move{g}{4}
\move{c}{7}
\move{e}{7}
\end{psgoboard}
```

As can be seen, the move numbers are displayed on the stones. The `gomove` counter is never reset by the psgo package. So, the move numbers just continue in subsequent diagrams. Of course, it is possible to reset the value of `gomove` manually. In that case, one should know that the `\move` command increases the `gomove` counter *before* it draws the stone. So, if a diagram should start with move 0, one should issue the command `\setcounter{gomove}{-1}` just before the diagram.

If a diagram continues another diagram, the numbers on the stones played so far are usually not desirable. Therefore, psgo has defined a starred version the move command: `\move*`. This command does not decorate stones with move numbers and it does not increase the `gomove` counter. For instance, if the game of Figure 3(a) is continued, all we have to do is copy&paste the game played so far, replace the existing `\move` commands by `\move*`, and add some new `\move` (unstarred!) commands to it. The code is given below and the result displayed in Figure 3(b).



(a) First 5 moves.



(b) Successive moves.

Figure 3: Moves on a board

```
\begin{psgoboard}[9]
\move*{c}{3} % old \move commands replaced by \move*
\move*{g}{7}
\move*{g}{4}
\move*{c}{7}
\move*{e}{7}
\move{f}{6}  % new \move commands
\move{e}{6}
\move{c}{5}
\end{psgoboard}
```

5 Markers

Empty positions on the board can be marked with the command `\markpos`. This command takes three parameters, being, the marker, the column, and the row. Available markers and the commands to generate them are listed in Table 1. Each marker is illustrated at position $(B, 2)$ on a 3×3 board. Note that the label marker command, `\marklb`, takes one argument, being the label. It is possible, though not advisable, to add more than one marker to an empty position.

Stones can be marked too. To this end, the commands `\stone` and `\move` take an optional parameter representing the marker of the stone. For example, `\stone[\markma]{black}{b}{4}` puts a black stone marked with a cross at position $(B, 4)$. Table 2 shows how stones can be marked. It is possible, though not advisable, to add more than one marker to a stone or to add markers to numbered stones.

6 Lines and arrows

In addition to markers, it is possible to add lines and arrows to the diagrams. The command `\goline` draws a line and the command `\goarrow` draws an arrow. Both commands take four parameters indicating the column and the

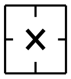
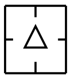
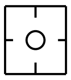
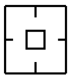
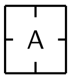
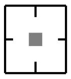
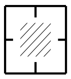
Diagram	psgo Command	Description	Example
	`\markma`	Cross	`\markpos{\markma}{b}{2}`
	`\marktr`	Triangle	`\markpos{\marktr}{b}{2}`
	`\markcr`	Circle	`\markpos{\markcr}{b}{2}`
	`\marksq`	Open square	`\markpos{\marksq}{b}{2}`
	`\marklb{#1}`	Label	`\markpos{\marklb{A}}{b}{2}`
	`\marksl`	Filled square	`\markpos{\marksl}{b}{2}`
	`\markdd`	Hatched lines	`\markpos{\markdd}{b}{2}`

Table 1: Markers on empty positions

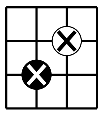
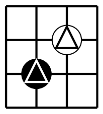
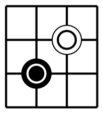
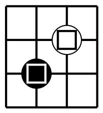
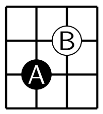
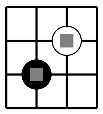
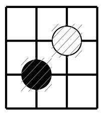
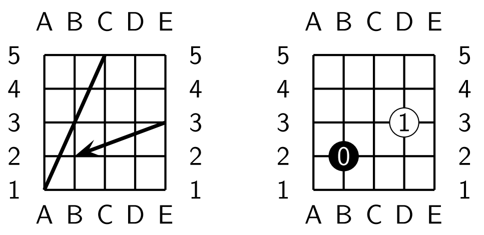
Diagram	psgo Command	Description	Example
	`\markma`	Cross	`\stone[\markma]{black}{b}{2}` `\stone[\markma]{white}{c}{3}`
	`\marktr`	Triangle	`\stone[\marktr]{black}{b}{2}` `\stone[\marktr]{white}{c}{3}`
	`\markcr`	Circle	`\stone[\markcr]{black}{b}{2}` `\stone[\markcr]{white}{c}{3}`
	`\marksq`	Open square	`\stone[\marksq]{black}{b}{2}` `\stone[\marksq]{white}{c}{3}`
	`\marklb{#1}`	Label	`\stone[\marklb{A}]{black}{b}{2}` `\stone[\marklb{B}]{white}{c}{3}`
	`\marksl`	Filled square	`\stone[\marksl]{black}{b}{2}` `\stone[\marksl]{white}{c}{3}`
	`\markdd`	Hatched lines	`\stone[\markdd]{black}{b}{2}` `\stone[\markdd]{white}{c}{3}`

Table 2: Markers on stones



(a) Lines and arrows (b) Labeling stones

Figure 4: More psgo features: lines, arrows, and labels

row of the start position and the column and the row of the end position. That is, `\goline{a}{1}{c}{5}` draws a line from position $(A, 1)$ to position $(C, 5)$ and `\goarrow{e}{3}{b}{2}` draws an arrow from position $(E, 3)$ to position $(B, 2)$, as illustrated in Figure 4(a).

7 Stones in text

In comments on a Go diagram, one often sees graphical representations of the stones on the board. Therefore, psgo has facilities to include stones in running text. For instance, we can refer to stone ❹ of Figure 3(a) with the command `\stone[4]{black}`. As can be seen, in running text (i.e., outside the `psgoboard` environment), `\stone` takes two parameters. The first one is optional and denotes a number or a marker to be placed on the stone. The second parameter denotes the color of the stone. As another example, here is a list of various stones: ⊗, ⊗, ▲, △, ◉, ◎, ■, □, Ⓐ, Ⓐ, ■, ▪, ●, ▨. This list was generated by the following code:

```
\stone[\markma]{black}, \stone[\markma]{white},
\stone[\marktr]{black}, \stone[\marktr]{white},
\stone[\markcr]{black}, \stone[\markcr]{white},
\stone[\marksq]{black}, \stone[\marksq]{white},
\stone[\marklb{A}]{black}, \stone[\marklb{A}]{white},
\stone[\marksl]{black}, \stone[\marksl]{white},
\stone[\markdd]{black}, \stone[\markdd]{white}.
```

It is possible to attach ordinary LaTeX labels to *moves* on a board. For instance, the digram of Figure 4(b) was generated by the following code:

```
\begin{psgoboard}[5]
\move{b}{2}
\move{d}{3}\label{funny:go:move}
\end{psgoboard}
```

As can be seen, a label `{funny:go:move}` is defined after the second move. To refer in the text to this move, the ordinary LaTeX `\ref` command can be used. In this case, we type `\stone[\ref{funny:go:move}]{white}`, which results in the stone ①, as expected. Of course, the color should still be defined manually.