The Pptalk package : sample and manual

Palash B. Pal
pbpal@theory.saha.ernet.in
Saha Institute of Nuclear Physics
1/AF Bidhan-Nagar, Calcutta 700064, INDIA

First release: February 2004
This version: March 2005

Set it up first!

You are now viewing a PDF file. I recommend that you see it with **Acroread** (called **Acrobat Reader** in Windows), 5.0 or later. Please read the instructions on this slide carefully before going to the next slide.

1. If you find that the file appears 90° rotated as you open it, use the **View** menu to rotate it clockwise by 90°. If the file appears in normal orientation, ignore this step.
2. Use the **full screen view** in the Acrobat. This can be accessed in the pull-down menu under the option **View**, or alternatively, by pressing **Ctrl-L**.
3. Use the $\boxed{\text{left mouse}}$ button, or the $\boxed{\text{PageDown}}$ button on the keyboard to go forward in the file. In what follows, this hit or click will be denoted by $\boxed{\bullet}$.
4. The $\boxed{\text{right mouse}}$ / $\boxed{\text{PageUp}}$ buttons can be used for going backward if you have to.
5. Even if you are not using **full screen view** for some reason, <u>never use the scroll bar</u> to navigate within the file. Rather, click on the next-page arrow (which looks like $\boxed{\triangleright}$, a triangle pointing right) with your mouse.

Okay, now use $\boxed{\bullet}$ to go to the next slide.

What Pptalk does

The package Pptalk is a LaTeX-based style file which allows you to achieve the following things:

1. Using a multi-media projector, you can project your talk on the screen in different slides. The slides can be divided into different segments which can appear one at a time, at the push of one key on the keyboard or one click of the mouse.
2. If you change your mind and decide to give the talk with transparencies, you can use the same file, with one announcement in the preamble, to generate a hard copy which will not divide a page into segments.
3. If you further want to print out the talk where you would not care about the division of the text into slides, you can again do it with a different announcement in the preamble.

Pptalk vis-à-vis Power Point

As you noticed, the objective is similar to the program *Power Point* used in Windows OS. Although I have never used *Power Point*, I must admit that Pptalk has been inspired by *Power Point* talk presentations that I have seen. Indeed, some people argue that the letters "pp" in the name Pptalk is merely an acronym of this famous software, though some others say that they are the initials of the person who wrote Pptalk.

Like *Power Point*, segments appear with clicks in Pptalk. However, it does not have animation like *Power Point*. On the other hand, you can use the rich formatting features of LaTeX, including equations and tables etc. If you are used to using LaTeX, you can use all its features in Pptalk.

Pptalk vis-à-vis other LaTeX-based presentation packages

I have seen two more LaTeX-based talk presentation packages: **prosper** and **pdfscreen**. I have not used either, and have come to know about the latter one only after writing most of Pptalk. Both these packages are much larger than Pptalk, and they can do many things that Pptalk cannot. However, there is a flip side to it. Both these packages have many commands as opposed to only a few for Pptalk. Pptalk consists of only a style file, a few kilobytes long, which can be easily transported and installed on another system if you are away on a trip. The other packages have to use separate class files, which means that the input file looks very different from what you would write for an ordinary **article** class file. In short, Pptalk, though rudimentary, has the most important features needed for a talk presentation, and is easier to deal with. This, by the way, is not the only advantage of Pptalk. The user is left to judge for others.

How to use Pptalk

Create the LaTeX file following the instructions given later in this document. Then use the standard commands to produce the dvi file, the ps file and the pdf file. For example, if the input file is called **xyz.tex**, issue the following commands in successsion:

```
latex   xyz
dvips   xyz   -o
ps2pdf   xyz.ps
```

The final stage produces a file called **xyz.pdf**. You can also create this file by using the alternative commands

```
latex   xyz
dvipdf   xyz.dvi   >   xyz.pdf
```

Open the pdf file on the screen using any standard pdf reader like acroread (or Acrobat Reader). As emphasized earlier, Acrobat 5.0 or later versions should be used.

If you find that the file appears 90° rotated as you open it, use the VIEW menu to rotate it clockwise by 90°. If the file appears in normal orientation, ignore this suggestion.

Now use the full screen view in the Acrobat, as discussed earlier.

How to organize the file

You don't need to use a new class file in order to write your talk with pptalk. You can use the **article** class file. Use the package pptalk. In other words, start your file as follows:

```
\documentclass[12pt]{article}
\usepackage{pptalk}
```

and of course, with the commands for beginning and ending the document. In the middle, you can write LaTeX in the usual way, with a few special commands for pptalk which are discussed in the rest of this document. These special commands will appear in red in this document. Any other command, either a shell command or commands belonging to standard LaTeX or to any existing style file, will appear in blue.

Be careful to include the **12pt** option in the **\documentclass** declaration, as shown above. The sizes of fonts have been set with this option in such a way that appears nice on the projected screen.

Do not use any of the page style commands like **\textheight** or **\textwidth**. These have already been set by pptalk to some optimum values which will suit a screen presentation.

Dividing things into slides

You need to decide how much (or how little) of text you want to put in a particular slide. When you want to begin a particular slide, type

\beginslide

Then continue with whatever you want to put there, typing things just as in a normal LaTeX file.

Divide things into paragraphs as you want, put equations as you want:

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} .$$

When you finish typing the contents of the slide, type

\endslide

After that, use the **\beginslide** command to start a new slide.

If you have ended up putting too little material on the slide, you will obtain a warning message while running LaTeX. You may choose to ignore this message, or want to put in some more material on the slide.

On the other hand, if the material is a bit too much to fit on a slide, Pptalk will mildly complain. While normally the page header (containing the slide number) appears above the slide box, Pptalk will now put the header on an otherwise empty page, followed by a page containing the slide box. This means that you need to roughly shorten the material by one line in order to make it fit.

When the material becomes so long that even that does not work, Pptalk will give you a page with an error message. Now some more drastic changes are necessary. Here are your options:

1. You can transfer some of the contents of the slide to another slide. This is the best option.

2. You can reduce the font size. The default font size is **\LARGE**. You can change it to, e.g., **\Large** by issuing the command

   ```
   \def\defaultsize{\Large}
   ```

   anywhere within the file. This is not recommended unless you are giving the talk to a small audience, in a small room.

3. In principle, you can increase the size of the slides by putting suitable values of **\textheight** and/or **\textwidth** in the preamble, like you can do in any LaTeX file. This is highly discouraged since the resulting slide may not fit on a computer screen.

Dividing things into segments

Now come the novel things. Suppose you want to present the contents of a page in a number of segments, one segment appearing at a time. What do you do?

You decide how the segmentation will have to be made in a slide. Start writing the slide with the first segment, after the **beginslide** command. When one segment is done and you want to start the next segment, just type the command

newsegment

Then write the text for the next segment. Continue this process until you reach the end of the slide with **endslide**.

Let us now see an example. Notice that there is an empty portion at the bottom of this slide. This is because there are more segments down there. Use $\boxed{\bullet}$ now.

Dividing things into segments

Now come the novel things. Suppose you want to present the contents of a page in a number of segments, one segment appearing at a time. What do you do?

You decide how the segmentation will have to be made in a slide. Start writing the slide with the first segment, after the **beginslide** command. When one segment is done and you want to start the next segment, just type the command

newsegment

Then write the text for the next segment. Continue this process until you reach the end of the slide with **endslide**.

Let us now see an example. Notice that there is an empty portion at the bottom of this slide. This is because there are more segments down there. Use $\boxed{\bullet}$ now.

Now you see the next segment. You can continue this way, introducing more segments. In principle, the number of segments on a page can be unlimited. However, the length of the page and the size of the font introduces some obvious restrictions.

From now on, hit $\boxed{\bullet}$ whenever you want to see some more material on the screen: whether material continued on the same slide or on a different slide.

You can put the command **\newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit ⊡ now.

You can put the command **newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit $\boxed{\bullet}$ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-

You can put the command **\newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit $\boxed{\bullet}$ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-calli

You can put the command **newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit $\boxed{\bullet}$ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-callifragi

You can put the command **newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit ⊡ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-callifragilistic

You can put the command **newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit $\boxed{\bullet}$ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-callifragilisticexpi

You can put the command **newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit $\boxed{\bullet}$ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-callifragilisticexpialli

You can put the command **newsegment** almost anywhere in the file. You can put it between two paragraphs, as you have done once earlier and will do again at the end of this paragraph. You will realize that the command can be put at the end of a sentence if you hit ⊡ now. You can put the command before or after an equation, before or after a table, or even in the middle of a word like (let's see if I can spell it right) "super-callifragilisticexpiallidocious".

You can also put the command within groups. By **group**, I mean anything which is between a matching beginning and ending command. For example, it can be a table, an environment like quote or enumerate, an equation, a command in the math mode which uses matching brackets.

However, in this case you will have to do a bit of extra work. The command **\newsegment** works only within a group. So, when the group ends, if you want to continue with the same segment, put the command

> **\samesegment**

and keep writing. For example, in writing

$$x =$$

You can also put the command within groups. By **group**, I mean anything which is between a matching beginning and ending command. For example, it can be a table, an environment like quote or enumerate, an equation, a command in the math mode which uses matching brackets.

However, in this case you will have to do a bit of extra work. The command **\newsegment** works only within a group. So, when the group ends, if you want to continue with the same segment, put the command

\samesegment

and keep writing. For example, in writing

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} ,$$

I had to put the **\samesegment** command right after the equation. And when I wrote

$$x = \left[\frac{1}{a} + \right.$$

You can also put the command within groups. By **group**, I mean anything which is between a matching beginning and ending command. For example, it can be a table, an environment like quote or enumerate, an equation, a command in the math mode which uses matching brackets.

However, in this case you will have to do a bit of extra work. The command **\newsegment** works only within a group. So, when the group ends, if you want to continue with the same segment, put the command

\samesegment

and keep writing. For example, in writing

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} ,$$

I had to put the **\samesegment** command right after the equation. And when I wrote

$$x = \left[\frac{1}{a} + \frac{1}{b}\right](y+2)$$

using **\left[**$\cdots$**\right]**, I had to put the **\samesegment** command after the closing square bracket.

How to use colors

You have already seen colors being used in this sample file. This has been done by using the package **color**. Pptalk itself has this package as an input, so you should not introduce it explicitly when you are using Pptalk.

How to use colors

You have already seen colors being used in this sample file. This has been done by using the package **color**. Pptalk itself has this package as an input, so you should not introduce it explicitly when you are using Pptalk.

The **color** package defines two commands for implementing colored text. One of them is **\textcolor**, used for a local change of color. The syntax is

\textcolor{*colorname*}{Text material}

which makes the *"Text material"* to appear in the color given as *"colorname"*. This command has been suitably redefined in Pptalk, and will work exactly the same way within Pptalk. On the other hand, you can use the command

\Color{*colorname*}

which will change the color of the text for the rest of the document. This is a Pptalk special, and is a variation of the command **\color** defined in the **color** package.

Color names

The basic color names appear in the file **color.sty**, which is part of the color package. They are: red, blue, green, cyan, magenta, yellow, and the non-colors white and black.

Color names

The basic color names appear in the file **color.sty**, which is part of the color package. They are: red, blue, green, cyan, magenta, yellow, and the non-colors white and black.

Additional colors, defined by the file **color.pro** in the color package, can also be used. Take, e.g., the name "Plum" from this file. If you issue the command

```
\definecolor{plum}{named}{Plum}
```

you will be able to use *'plum'* for a *'colorname'*.

Color names

The basic color names appear in the file **color.sty**, which is part of the color package. They are: red, blue, green, cyan, magenta, yellow, and the non-colors white and black.

Additional colors, defined by the file **color.pro** in the color package, can also be used. Take, e.g., the name “Plum” from this file. If you issue the command

\definecolor{*plum*}{named}{Plum}

you will be able to use *‘plum’* for a *‘colorname’*.

You can also create new colors. For example, a new color called **RONG** can be defined by any of the following commands, where each p_i is a number between 0 and 1:

\definecolor{*RONG*}{rgb}{p_1, p_2, p_3}
\definecolor{*RONG*}{cmyk}{p_1, p_2, p_3, p_4}
\definecolor{*RONG*}{gray}{p_1}

The last command produces only shades of gray. Test with various values of p_i to create your own pallette.

Background color

There can be two different meanings of the words "background color" in the context of Pptalk. Any slide appears as a box on a page. If you want to change the color of the page outside the box, use the standard command defined in the color package:

\pagecolor{*colorname*}

where *'colorname'* is the name of a color already defined, either in the file **color.sty** or by yourself. The default color is white, which is what has been used in this document. You can change it anywhere within the document. The command takes effect from the slide on which it appears.

You might also want to change the color of the box containing text, and the color of the border. For this, you need to use the Pptalk command

\slidebox{*color1*}{*color2*}

before any **\beginslide** command. Once you use this, *'color1'* will be the color of the border, and *'color2'* will be the base color within the borders, starting from the slide which starts after this declaration. If you don't like the border, *'color1'* must be the same as the background color of the page, or the base color set by *'color2'*.

If you also want to change the width of the border, you can use an optional argument. For example, before this page, the border has been set to blue, the base color to beige, and the borderwidth to 1mm by issuing the command

```
\slidebox[1mm]{blue}{beige}
```

where the color beige has been defined by

```
\definecolor{beige}{rgb}{.9,.9,.7}
```

earlier in the document. Note that the base color was beige earlier as well. But it does not matter. You have to supply the names of both colors in the **\slidebox** command even if you want to change only one of them, or even if you want to just change the borderwidth without changing the colors at all.

If you also want to change the width of the border, you can use an optional argument. For example, before this page, the border has been set to blue, the base color to beige, and the borderwidth to 1mm by issuing the command

```
\slidebox[1mm]{blue}{beige}
```

where the color beige has been defined by

```
\definecolor{beige}{rgb}{.9,.9,.7}
```

earlier in the document. Note that the base color was beige earlier as well. But it does not matter. You have to supply the names of both colors in the **\slidebox** command even if you want to change only one of them, or even if you want to just change the borderwidth without changing the colors at all.

This command can also be utilized if you do not want the border around the text. Just set the width to zero. Remember that the width must have a unit of length. For example, you can write

```
\slidebox[0mm]{blue}{beige}
```

Instead of *‘mm’*, you can use any other unit recognized by LaTeX.

I have a few recommendations on the use of colors.

1. Do not use the global color change command **color** defined in the package **color.sty**. Use **textcolor** or **Color** instead.

2. Use your judgement for color contrasts. Needless to say, the contents of a slide will not be very legible if you use orange letters on a yellow background.

3. Use light and dull shades for background. Typeset most of the things in black. No other color appears as beautifully on the screen. Use other colors only for emphasis.

4. It is best not to define a new color unless you absolutely need to do so. Use, as much as possible, the names defined in **color.pro** in order to retain some kind of universality in the color names. It contains a lot of names, as shown next. Try them out.

Colors defi ned in color.pro

GreenYellow	**Yellow**	**Goldenrod**	**Dandelion**
Apricot	**Peach**	**Melon**	**YellowOrange**
Orange	**BurntOrange**	**Bittersweet**	**RedOrange**
Mahogany	**Maroon**	**BrickRed**	**Red**
OrangeRed	**RubineRed**	**WildStrawberry**	**Salmon**
CarnationPink	**Magenta**	**VioletRed**	**Rhodamine**
Mulberry	**RedViolet**	**Fuchsia**	**Lavender**
Thistle	**Orchid**	**DarkOrchid**	**Purple**
Plum	**Violet**	**RoyalPurple**	**BlueViolet**
Periwinkle	**CadetBlue**	**CornflowerBlue**	**MidnightBlue**
NavyBlue	**RoyalBlue**	**Blue**	**Cerulean**
Cyan	**ProcessBlue**	**SkyBlue**	**Turquoise**
TealBlue	**Aquamarine**	**BlueGreen**	**Emerald**
JungleGreen	**SeaGreen**	**Green**	**ForestGreen**
PineGreen	**LimeGreen**	**YellowGreen**	**SpringGreen**
OliveGreen	**RawSienna**	**Sepia**	**Brown**
Tan	**Gray**	**Black**	**White**

The running head on a slide

Each slide contains a header. For example, the header of the present slide contains a line which says *'Created with pptalk'*, followed by the slide number. This is the default header, and will appear on all slides unless you change it.

To change the text appearing in the header to, say, *'My favorite header'*, you need to issue the command

```
\def\runningheadtext{My favorite header}
```

In addition, you can also change the color of the running header to *'color1'* by issuing the command

```
\def\runningheadcolor{color1}
```

You can use these commands at the beginning of the file, or between any two slides.

The slide number will appear automatically. You need not do anything for it. Its color will be the same as the color of the text in the header.

Some technical points

- On a segmented slide, do not use objects which are automatically numbered by LaTeX. The segmentation will interfere with the numbering. If you have to start a section, you can do it with the usual **section** command, which I have redefined to appear without a number. If you want to have a displayed equation, use the **eqnarray*** or the **equation*** environments (with the asterisk marks), or begin and end equations with the double dollar ($$) signs, which will prevent the numbering of equations.
- The contents of a slide are put into a vbox by Pptalk. Environments like **enumerate**, **quote**, and **itemize** cannot work across different vboxes, and therefore cannot be continued on different slides. For long lists, it is advisable not to use **enumerate** at all. Instead, use **itemize**. That way, you can put a number of items on one slide, end the **itemize** environment on that slide, and continue with a new **itemize** on the next slide. Looking at the result, no one will know that the same **itemize** is not continuing.

Displaying tables

- Do not use the **table** environment that LaTeX provides. Apart from putting a number of the table, it also puts the table as a floating object, and LaTeX decides where to put it. Just for creating the table, the **tabular** environment is good enough. If the entries contain too many math symbols, you can also use the **array** environment, putting the table within an equation.
- Do not use the \newsegment command in the middle of a table. I haven't been able to make it work. This is a bug, which will be fixed shortly. Meanwhile, if you really want to highlight something in a table, use colors to do that.

Displaying figures

- There are a number of packages for inserting postscript or EPSF files into a LaTeX file. In pptalk, I use **epsf.sty**, which must not be introduced by the user because it is already included in pptalk.
- For reasons described in the context of the **table** environment, the **fi gure** environment should not be used. Just insert your figure by the usual command **\epsfbox**, with one specification mentioned below.
- The normal syntax for **\epsfbox** is as follows:

  \epsfxsize=*len1* \epsfysize=*len2* \epsfbox{*figfile*}

  where ***len1, len2*** are lengths in any units recognized by LaTeX, and ***figfile*** is a ps or eps file for a figure. The size declarations are normally optional. However, for pptalk, you must have an **\epsfysize** declaration. In fact, pptalk decides on the size of the slide from this declaration. The **\epsfxsize** declaration still remains optional.

- If you prefer some other figure package, you can still use it, but then you will have to announce it by the **\usepackage** command explicitly. And I guess that the figures will then be incompatible with the segmentation commands of Pptalk. This means that you will have to put the figures in a page which is not divided into segments.
- But if you use the **epsf** package, you can use segmentation commands on pages containing figures. An example is given on this page.
- On a segmented slide, avoid using very large figure files which will take a long time to load.

- If you prefer some other figure package, you can still use it, but then you will have to announce it by the **\usepackage** command explicitly. And I guess that the figures will then be incompatible with the segmentation commands of Pptalk. This means that you will have to put the figures in a page which is not divided into segments.
- But if you use the **epsf** package, you can use segmentation commands on pages containing figures. An example is given on this page.
- On a segmented slide, avoid using very large figure files which will take a long time to load.

On defaults

The default font family for LaTeX is **cmr**. These fonts do not appear well in pdf conversions. So I have reset the default to **times**. The **palatino** and the **helvetica** families also work quite well. If you want to use **palatino**, for example, put the command

```
\usepackage{palatino}
```

in the preamble of the file after the pptalk package has been included. For **helvetica**, the corresponding package is called **helvetic**.

Other defaults are summarized in the following table:

Parameter	Default value	Can be changed by
Font size	\LARGE	\defaultsize
Text color	black	\Color
Background color	white	\pagecolor
Slide base color	cyan	\slidebox
Slide border color	red	\slidebox
Slide border width	2mm	\slidebox

Presentation tips not related to Pptalk

- ♠ Acrobat reader has some special modes for appearance of a slide or a segment which appear in the full-screen view mode. For example, in the PREFERENCES menu, you can set up such that new slides appear in the DISSOLVE mode. Test to see what it does.
- ♠ The LaTeX package called **pstricks** has many beautiful features which can aid your talk presentation. For example, you can draw an arrow from one place in the slide to another.
- ♠ You can use a variety of symbols to announce an **\item** in a list like this one. For example, the list on this slide has been initiated by writing

  \begin{list}{\textcolor{red}{$\spadesuit$}}{}

  followed by the usual **\item** command and the command to end the list. Some fancy symbols for item markers are available in different packages.

Printing out

In case you want to print out the talk, you would presumably want all segments of a page to appear simultaneously. Two options have been provided for this. If you type

```
\transparency
```

in the preamble of the file, you will obtain a version which you can use on transparencies. On the other hand, if you type

```
\onpaper
```

in the preamble, you will obtain a more condensed version, where text will not be divided into slides, but will rather appear continuously, as is usual for printing out an article. You cannot use both at the same time. If you put both commands, only the last one will take effect.

If later you want to produce a multimedia talk again, just remove all of these declarations from the preamble.

Summary of commands

Here is a summary of the special commands for pptalk.

\beginslide · · · **\endslide** Denotes the beginning and end of a slide. The contents of the slide goes in the middle, in place of the three dots.

\newsegment Denotes the beginning of a new segment in a slide. Not required for the first segment in a slide, which begins with the **\beginslide** declaration.

\samesegment If the **\newsegment** appears within a group, this command has to be used at the end of the group to reassert the continuation of the segment after the group.

\slidebox Defines the border and base colors of all slides which start after this announcement, unless changed by the same command again. An optional argument defines the width of the border.

\Color Changes the color of the text for the rest of the document.

\transparency To be put in the preamble if you want all segments to appear simultaneously on a page, as e.g. is usual for a transparency talk.

\onpaper To be put in the preamble if you want a continuous printout, like you would want for an article to be sent to a journal.

Then there are commands which need to be set with a leading **\def**. These are:

\def\defaultsize Redefines the default size of letters in the text.

\def\runningheadtext Redefines the running head on the slides.

\def\runningheadcolor Redefines the color of the letters in the running head on the slides.

Also useful are the following commands, which are not Pptalk specials but are not a standard LaTeX commands either. The following two are parts of the package **color**:

\defi necolor Used for defining colors.

\textcolor Redefines the text color locally.

And then the command from the **epsf** pacakge:

\epsfbox Used for inserting figures. Remember that the **\epsfysize** command is mandatory here.

Acknowledgements

I thank Soumitro Banerjee and Amitabha Lahiri, who made many important comments and refused to be satisfied by earlier versions of Pptalk. I hope they are still dissatisfied, which will provide me with motivations for further improvements of Pptalk.

If you find any bug, or have any suggestion for improvements without substantially complicating the program, please contact me.