No hands — the dictation of LaTeX

Mike Roberts

Abstract

This article gives a brief introduction to a combination of open source extensions to Dragon Professional Individual dictation software which allow for relatively easy dictation of LaTeX syntax and mathematical formulae.

While the primary use case is for people with disabilities which prevent them from typing (repetitive strain injuries caused by regular computer use are surprisingly common), dictation may provide a realistic alternative for normal users as well.

1 Introduction

1.1 Me

As an Economics undergraduate with a spinal injury which precludes me from using a keyboard, I've been using dictation software throughout my degree for all exams, assignments and essays. This article will focus on the workflow I've developed and the tools that I use for dictating LaTeX documents and mathematical formulae relatively painlessly.

First, though, I will briefly describe the problem space: what would ideally be made possible by a dictation system and the limitations of what is currently available. I will then describe the technical details and user experience of my setup.

1.2 Dictation software

If you are totally unfamiliar with dictation software, the first thing I should say is that broadly speaking it works well. With a good microphone, in a quiet room, speaking clearly, it's not unreasonable to expect in excess of 99% accuracy when dictating full sentences.

The leader in the voice recognition industry has for many years been Nuance (though with the current rate of progress in machine learning their technological lead may soon evaporate). Their product, Dragon, is marketed mainly for corporate, medical and legal document preparation and works excellently when dictating into Microsoft Word. For completing the full range of academic work without a keyboard though, this basic dictation functionality is necessary but not sufficient, and for those with disabilities who cannot use a mouse, navigating a graphical interface to access formatting options is an active hindrance to getting things done.

For dictating mathematics, another commercial product is available — MathTalk [3] is the industry standard for mathematical dictation (the client list on their website includes the Department of Defense and Federal Aviation Authority, as well as many universities). When I first tried it I was very disappointed and quickly found it to be practically unusable. Despite costing over $300 it is incredibly slow to recognise and execute commands, and will not accept more than two or three commands at a time. This can easily be seen by watching any of the videos on their website. Even with extra time granted, the chances of being able to do well in exams when you have to wait half a second between every character are minimal.

1.3 The problem

There is certainly room for improvement then, both for normal documents and for mathematics. An ideal dictation system of this kind should have a number of features which are lacking from the commercial offerings detailed above. Firstly, it should be able to interpret commands as fast as the user can say them, without the need to wait for output to appear or to pause between commands. Secondly, it should be customisable so that users can modify and add to their grammar to suit it to their own needs. Finally, of course, it should as far as possible be free and open source.

2 Implementation

As I finish my degree and after a lot of experimentation I have settled on a solution which is as close to this ideal as I can imagine. It can be used for almost anything, can interpret commands as quickly as they can be dictated, and is modifiable and extensible.

2.1 Building blocks

Although it is somewhat expensive, Dragon [4] is by far the best speech recognition engine currently available, and while its built-in tools for creating custom commands are limited, it is thankfully hackable. This is done using Natlink [2], a free tool originally created in 1999 by Joel Gould — then working at Dragon — which allows for custom command sets written in Python 2.7 to be imported into Dragon.

These custom commands are defined using an open source Python library called Dragonfly [1], which simplifies the process of creating grammars and provides easy access to frequently used functionality like the typing of text and the execution of keystrokes.

Together, these elements — Dragon, Natlink and Dragonfly — allow for any combination of keystrokes or Python scripts to be mapped to voice commands which can be interpreted and executed fluidly and with only minimal delay. While these tools have so far primarily been used to enable voice programming, they can easily be repurposed for voice-enabling virtually anything.

2.2 Mathfly

Mathfly [5] is my own contribution, and comprises command sets for dictating raw LaTeX as well as using WYSIWYG editors like LyX.

Within Mathfly, commands are organised into modules, each with a different purpose, which can be enabled and disabled at will. For basic operations like creating a new file there are also context specific commands which will only be recognised when a particular program is active.

To provide usability for non-programmers, predictable and common structures (like the begin, end tags in LaTeX) are hardcoded in Python while the lists of options themselves are defined in plain text configuration files which can all be opened and added to with voice commands.

3 LaTeX

LaTeX represents an obvious and favourable alternative to dictating into word processing software for a number of reasons. Writing everything in plain text means that entire documents can be produced by replicating keystrokes, without ever having to navigate an awkward GUI. This provides the ability to automate fiddly tasks like the creation of tables, insertion of images and organisation of references — a major win for those without the use of a mouse.

Dictating using Mathfly's LaTeX module is intended to be as intuitive as possible and to work largely as one would expect it to. Most commands consist of a memorable prefix, which helps to avoid over-recognition during dictation, followed by the name of the desired item.

3.1 Basic commands

For example, saying "begin equation" produces:

```
\begin{equation}
\end{equation}
```

Similarly, "insert author" and "insert table of contents" produces `\author{}` and `\tableofcontents`, respectively, and "use package geometry" will produce `\usepackage{geometry}`.

LaTeX commands can often be a little cryptic and non-obvious. For example, to create a bulleted list, you need:

```
\begin{itemize}
```

This is reasonably memorable once you have used it a few times, but is not easily guessable, especially for those of us living in countries which resist the encroachment of the letter Z.

Mathfly attempts to make things as easy as possible in cases like these by often providing multiple voice commands for the same thing. In this case, "begin itemize", "begin list" and "begin bulleted list" will all produce an itemize environment.

3.2 Mathematics

By default, all mathematical symbols are prefixed with "symbol", so "symbol integral" produces `\int`, but there is also a mode specifically for dictating symbols which does not require the prefix. Thus in mathematics mode, "sine squared greek theta plus cosine squared greek theta equals one" produces:

```
\sin ^{2} \theta +\cos ^{2} \theta =1
```

that is,

$$\sin^2\theta + \cos^2\theta = 1 \tag{1}$$

3.3 Templates

For including larger sections of text, or sections which don't fit into any of the predefined commands, there are templates — arbitrary strings which are pasted with a voice command. For example, the command "template graphic" pastes:

```
\begin{figure}[h!]
\centering
\includegraphics[width=0.8\textwidth]{}
\caption{}
\label{}
\end{figure}
```

Not only does this save a lot of time and repetition, but as a novice user it is useful to be able to outsource the task of remembering what settings you like to use and how common command blocks are constructed.

3.4 Configuration

As I mentioned above, it is easy for users to add to the available commands by modifying the configurations files. The command definitions for the LaTeX module look like this:

```
[environments]
"equation" = "equation"
...
[command]
"author" = "author"
...
```

and can be easily added to or modified.

3.5 Scripting

There are also intriguing possibilities for the integration of Python scripting. I've only scratched the surface so far, but to give an example I can highlight the title of a book or paper, say "add paper to bibliography" and a script will run which searches Google Scholar for the title and appends the resulting BibTeX citation to my `.bib` file.

4 WYSIWYG mathematics

For technical homework assignments and especially exams, formatting is of far less importance than getting what you know onto the paper as quickly and easily as possible, so a what you see is what you get (WYSIWYG) editor makes more sense.

Mathfly includes grammars for both LyX, an open source LaTeX editor, and Scientific Notebook, a proprietary alternative which is often provided for free by universities. They both function similarly and allow for natural dictation of mathematical formulae with immediately visible output.

For example, the command

```
integral
one over x-ray
right
delta
x-ray
equals
natural logarithm
x-ray
plus
charlie
```

can all be interpreted in one go and will produce the desired output.

$$\int \frac{1}{x} dx = \ln x + c \tag{2}$$

The only deviation from natural speech is the requirement for a command (a right keypress) to signal the end of the fraction.

I don't have any hard data comparing the speed of dictation like this to that of normal writing. I can say, though, that I am technically allowed 50% extra time for all exams but have never needed to make use of it, suggesting that the two methods are fairly comparable.

5 Limitations

The major limitations of dictation are currently not functional — it performs about as well as could reasonably be expected — but are related to platforms and compatibility. Dragon and Natlink are only available on Windows (with a limited and soon to be discontinued version of Dragon available for Mac OS X), so the only feasible way of using software like Mathfly on other operating systems is to run Dragon in a Windows Virtual Machine, using remote procedure calls to send instructions to the host.

The long-term prospects for a completely free and platform agnostic dictation and voice command framework are reasonably good, however. Dragonfly is under active development with the aim of integrating new speech engines like Carnegie Mellon University's PocketSphinx and Mozilla's DeepSpeech, although these have a fairly long way to go before they reach Dragon's level of maturity and accuracy.

6 Getting started

Anybody interested can visit the Mathfly website [5], which contains links to the documentation, installation instructions and a few short video demonstrations. If you have any questions or requests then feel free to email me, post in the Gitter chat room or on the GitHub issues page.

References

[1] C. Butcher. Dragonfly, 2007. `pythonhosted.org/dragonfly`

[2] Q. Hoogenboom. About Natlink, Unimacro and Vocola. `qh.antenna.nl/unimacro`

[3] mathtalk.com. MathTalk — speech recognition math software. `mathtalk.com`

[4] nuance.com. Dragon — the world's no. 1 speech recognition software (nuance UK). `nuance.com/en-gb/dragon.html`

[5] M. Roberts. Mathfly — dragonfly/caster scripts for dictating mathematics fluidly, 2019. `mathfly.org`

◇ Mike Roberts
`mike (at) mathfly dot org`
`mathfly.org`