What You Say is What You Get: Handsfree Coding in 2022

Wolfram Wingerath

Buzzing Technologies
What You Say is What You Get

Handsfree Coding in 2022

code.talks 2022
Sep. 16, 2022
Wolle
It's Simple, Really!

The requirements:

✓ **Microphone**: Every notebook has one!

✓ **Speech Recognition Software (SR)**: Included in Windows since 2007!

✓ **Voice Command Execution**: Available in every SR software!

LET'S GOOOOOO!!!!!!
Let Me Just Show You How Easy It is

open (adult scrolls conflict for
delete adult scrolls conflict
for
Let Me Just Show You How Easy It is

Go Watch Emily‘s Talk!
Where’s the Challenge?

- **Automatic Speech Recognition (ASR):** optimized for **natural** languages
  1. Signal processing extracts features from audio recording
  2. Acoustic model recognizes phonemes
  3. Language model finds a matching sequence of words:
     - **Default:** Every utterance is interpreted as (spoken) text
       (Commands only through special keywords)

- **Voice Coding:** optimized for **actions & programming** languages
  - **Default:** Everything is interpreted as a command
    (Natural language through special keywords, e.g. say <utterance>)
ONE DOES NOT SIMPLY TALK TO A COMPUTER
Outline: What This Talk is Going to Cover

1. My Personal Background
   As data engineer & scientist, I use handsfree coding every day.

2. Demo & Usage Examples
   Handsfree coding is awesome and can be useful for everyone!

3. Setup & Best Practices
   No-cost base setup with optional upgrades (e.g. for eye tracking).

4. How to Get Started
   Videos, blogs, articles, support & community – engage now!
My Job is Data Science
I Am Wolle

I'm data engineer, not an ASR or HCI expert!

Research:
- Stream Processing
- Real-Time Databases
- NoSQL & Cloud Systems
- ...

Practice:
- Web Caching
- Big Data Analytics
- Anger Management
- ...

Carl von Ossietzky Universität Oldenburg

Baqend
Look,
No Hands!
Basic Voice Control

• Actions & Symbols:
  o Numbers, brackets, etc.: one space paren bang slash → 1_(!/
  o Modifiers, e.g.: shift touch or hold alt or release control

• Spelling through a phonetic alphabet:
  o NATO alphabet: alpha bravo charlie delta echo → abcde
  o Optimized alphabet: air bat cam drum each → abcde

• Navigation, e.g.:
  o Cursor movement: go left or go way up or home
  o Selection: mark left or mark way up or mark all
Basic Voice Control

• **Typing:**
  - **Dictation:** *say hey comma you* → *hey, you*
  - **Type recognized text:** *phrase hey comma you* → *hey comma you*

• **Formatters, e.g.:**
  - **Camel case:** *camel hey you* → *heyYou*
  - **Dashed:** *kebab hey you* → *hey-you*
  - **Dotted:** *dotted hey you* → *hey.you*
  - **Without whitespace:** *smash hey you* → *heyyou*
  - **Uppercase:** *-uppercase hey you* → *HEY YOU*
Basic Voice Control

- Command History & Help:
  - Manage history: `command history` or `command history clear`
  - Show & hide help: `help active` or `help close`

- Application management, e.g.:
  - Launch & focus: `launch Firefox` or `focus Firefox`

- Command Management for efficiency, e.g.:
  - Repetition: `one third` → `111`
  - Chaining: `paren close paren` `go left` `say hi` → `(hi)`
    1.) type: `(`
    2.) move cursor: `hi`
    3.) dictate: `hi`
On-the-fly prototyping for your grammar!
  - Python for live reloading
  - Voice coding inception: Work in the grammar with the grammar

Customization of format and spelling:
  - Example: etcetera → etc.

Settings, delays, speech engine, etc.

```python
def settings():
    key_hold = 150.0
    key_wait = 20.0
    speech.engine = 'dragon'
```
mode: sleep
-

hello code talks:
  key(down)
"now this command will execute a sequence"
Handsfree Coding

- Different behavior for different semantics, for example:
  - C#: funky test funk \(\rightarrow\) private void testFunk()
  - JavaScript: funky test funk \(\rightarrow\) function testFunk()

- Intuitive IDE shortcuts such as
  - "run code" instead of \(<\text{shift}-f10>\)
  - "find usage" instead of \(<\text{ctrl}-\text{alt}-f7>\)

- Powerful templates, e.g.:
  ```
  action(user.code_state_if):
  insert("if () {}")
  key(left enter up end left left left)
  ```
import React from 'react';
import styled from 'styled-components';

import Icon from '@components/Icon';

function IconButton({ icon, children }) {
  return (
    <Wrapper>
      <Icon icon={icon} />
      {children}
    </Wrapper>
  );
}

const Wrapper = styled.button``
  background-color: var(--color-primary);
  font-size: 2rem;
  cursor: pointer;
Handsfree Coding: Cursorless

- Available on GitHub: [github.com/cursorless-dev/](http://github.com/cursorless-dev/)
- VSCode extension
- Spoken language for **structural code editing**
  - Decorates every token on screen with a **mark**
  - Tokens can be selected via combination of mark and **scope**
  - **Actions** operate on the specified tokens

- Example:
  - `chuck funk red`
    - action  scope  mark
  - (delete the function marked with „r“)
Handsfree Coding: Cursorless

```javascript
const foo = 0;
const bar = "hello";
makeEven(foo, true ? 1);

function makeEven(increase: number, num: boolean = false) {
  if (num % 2 === 0) {
    return increase + 1 ? hello + 1 : num - 1;
  }
  return num;
}

const numbers = {
  [one]: "one",
  two: "two",
  [three]: "three",
  four: "four",
  five: "five",
  [six]: "six",
  seven: "seven",
  eight: "eight",
};

export {};
```

(moving code around)
Handsfree Coding: Cursorless

(selecting semantic entities)
Snappy Noise Control With Parrot

- Available on GitHub: [github.com/chaosparrot/parrot.py](https://github.com/chaosparrot/parrot.py)
- Noise-controlled actions with latency <50ms
- Workflow
  1. Record sounds
  2. Train model for recognition
  3. Map sounds to actions
- Compatible (and recommended in combination with) with other tooling:
  - Often used with Project IRIS (eye tracking)
  - Can be used to produce Talon-compatible models

Custom noises for your Talon grammar!
Popular Handsfree Coding Stacks: Overview

- Input (Hardware)
  - iPhone/iPad/Mac
  - Eye Tracker
  - Microphone
  - Wav2Letter (English)
  - Conformer (English)
  - Dragon (German)
- Speech-to-Text
- Magic
- Scripting Framework
- Custom Commands
  - KinesicMouse Live
  - Iris
  - Parrot
  - Talon
  - <noise model>
  - <grammar>
- NatLink
- Dragonfly
  - Caster
  - Vocola
  - Aenea, Silvius, VoiceCode, Unimacro, ...

Please note that this overview is NOT complete: On every level, there are MANY other options!

*Facial Action Handling

This overview was inspired by: https://dictation-toolbox.github.io/dictation-toolbox.org/ (accessed: January 4, 2021)
Upgrades & Add-Ons
• **Microphone** determines accuracy!
  - *Build quality*: built-in < gaming headset < stage mic
  - *Positioning*: consistent, close to your mouth, away from all noise
  - *Mixed bag*: Noise canceling via hardware or software (e.g. RTX Voice)

• **Environment**: Minimize noise for you and annoyance for others!
  - Suspend ASR / mute mic accordingly (e.g. via push-to-talk pedal)

• **Homophones** should be avoided, e.g. through:
  - Grammar optimization to avoid ambiguity
  - Clear pronunciation
General Issues

- **Multilanguage support** is still in its infancy
  - Non-English language models all have their problems
  - Designing command libraries for different languages means effort
- **Eye tracker stability** is an issue as it disconnects sporadically
  - You will have to restart Talon from time to time...
- **Random crashes** are rare, but do occur from time to time:
  - Fallback to manual input sometimes necessary...
Workflow & Anger Management

- **Beware the Trolls**: Having an audience generally does not help!
  - Prepare to hear „Format C“ from your colleagues a lot

- **Keep your calm**: Shouting at the computer will not help, either!
  - Stay in your neutral voice, even when raging inside...

- **Avoid Voice Strain**: Find a comfortable way to speak A LOT!
  - e.g. use your natural voice & drink a lot of tea

- **Command chaining**: Anticipate what is going to happen!
  - Practice, practice, practice!
Potential Privacy Issues

- **Watch Your Tongue**: Passwords & confidential info may be leaked ...  
  - ... through plain acoustics (beware eavesdroppers!)
  - ... as they are stored your *command history*!
  - ... to involved third parties (e.g. with Web Speech)

- **Watch Your Transmitter**: Wireless solutions are often not encrypted!

- **Watch Your Eyes**: Your eye movement may give away a lot  
  → perhaps avoid continuous eye tracking ;-)
Why This is Still Worth All the Hassle

**Productivity**
- Speed up input-heavy tasks
- Faster navigation through easy-to-remember shortcuts

**Convenience**
- Intuitive interfaces
- Relieve your hands

**Accessibility**
**Compensate handicaps:**
- Injuries (e.g. broken hand)
- Repetitive stress injury (RSI)
- Cubital Tunnel Syndrome
- ...

**General Awesomeness**
- Talk to your computer!!!

The Hoff approves!
Helpful Resources & Outlook
Recommended Tooling & Documentation

- Talon (Free of Charge): talonvoice.com / talon.wiki
  - Voice coding for Win / Linux / Mac!
  - Starter Grammar (English): github.com/knausj85/knausj_talon
- parrot.py (noise control): github.com/chaosparrot/parrot.py
- Cursorless (code editing for VSCode): github.com/cursorless-dev
- Paid Upgrades:
  - Talon Premium Support: patreon.com/join/lunixbochs
  - Dragon Speech Recognition: nuance.com/dragon/
Alternatives: Speech Recognition

- Speech Recognition
  - WSR (Windows Speech Recognition): Built into Windows
  - Kaldi: github.com/kaldi-asr/kaldi
  - Vosk (ASR on mobile devices!): github.com/alphacep/vosk-api
  - Web Speech API (compatible with Talon through Chrome or Firefox)

- Scripting:
  - NatLink: sourceforge.net/p/natlink/
  - Dragonfly: github.com/dictation-toolbox/dragonfly
  - Caster: github.com/dictation-toolbox/Caster
  - Vocola (Voice Command Language): vocola.net
Articles & Blogs

- Emily Shea: [whalequench.club/](http://whalequench.club/)
  - Talon user
  - Very good starter instructions
- James Stout: [handsfreecoding.org/](http://handsfreecoding.org/)
  - Dragonfly user
  - Huge collection of relevant blog posts
Recommended Talks


David Williams-King. Coding by Voice with Open Source Speech Recognition The Eleventh Hope (2016)

Closing Recommendations

- **Keep it simple**: Prioritize ease-of-use over efficiency at the start (in particular: get used to an existing grammar before optimizing it)

- **Keep it reasonable**: Try to find use cases that make sense *for you* (e.g.: I’m not giving this talk handsfree, since I can use my index finger)

- **Keep it in mind**: Handsfree coding might save you one day (revisit this talk when you struggle with RSI, broken hand, etc.)
Sprechen ist das neue Klicken

Dr. Wolfram Wingerath, Michaela Gebauer

Für die Bedienung des Computers brauchte man viele Jahre Maus und Tastatur – heute kann man mit Sprache, Gestik und Mimik sogar programmieren.
Thanks! So What Now?

Slack
talonvoice.slack.com

Ask questions!
Enjoy the community!

GI Initiative
handsfree-coding.gi.de

Try out handsfree coding!

Patreon
patreon.com/lunixbochs

Support Talon Development!

Subscribe to the mailing list!

Wolfram Wingerath  wolle@uol.de