Tim Bedford
System Paper Outline

1. Introduction
   (a) Initial inspirations
      i. Voyager by George E. Lewis
      ii. PfQ model by Blackwell and Young
   (b) Broad overview of my system
      i. Receives input from one monophonic sound source, outputs monophonic sound
         A. Input is human voice
         B. Output is synthesized instrument modeled after human voice
      ii. Based on PfQ
         A. Each of the three components is developed in a different programming language
         B. Components communicate through Open Sound Control (OSC) protocol
      iii. Project is personal
         A. Not developed with a particular theoretical model of creativity in mind
         B. Not intended to research cutting edge A.I. techniques
         C. Space to explore and refine my own programming practice and aesthetics
            - Similar systems are rarely developed with the intention to only be used by the designer themselves
2. ‘P’
   (a) Outline of the role of ‘P’
   (b) Previous attempts with different software
      i. librosa and aubio
         A. Both Python libraries
         B. Not well maintained
      ii. Marsyas
         A. Actively developed for over a decade
         B. Can be used through built-in scripting language or in C++
         C. Inconsistent, sparse documentation
            - Mainly used by lab it’s developed in
            - Research outside of their scope isn’t well covered
   (c) Why Max?
      i. Initial reservations
         A. I wanted all parts of system to use conventional text-based languages
         B. Max is proprietary
      ii. I have used it before
      iii. Stable and well-documented
      iv. Well supported objects for MIR
      v. Support for OSC
      vi. Reservations answered
         A. Decided text-based restriction was unnecessary
         B. Max Runtime
            - Free version of Max
            - Can run Max files normally, but can’t edit them
3. ‘f’
   (a) Outline of the role of ‘f’
   (b) Why Python?
      i. Initial reservations

A. It’s not a particularly fast language

- Speed is ideal with real-time audio

ii. Language I know the best

iii. Reservations answered

A. Speed hasn’t affected me

- Python isn’t directly working with audio
- 2010’s hardware is much more robust than early systems’ hardware

2. ‘Q’

(a) Outline of the role of ‘Q’

(b) Initially planned on abstract synthesis

(c) Switch to vocal synthesis

i. Reasons

A. Narrowed my options; therefore easier to focus

B. Interesting pairing my “real” voice with “fake” voice

ii. FOF synthesis

A. Various ways to analyze and synthesize sung speech

- List a few

B. FOF: explanation

C. FOF is well known and comparably simple conceptually

D. Sound isn’t as strong as other techniques, but synthesis isn’t focus of this project

(d) Why Csound?

i. Csound is the only option that already includes FOF synthesis

A. Max: only available as outdated (unusable) third party objects

ii. Opportunity to learn new language

iii. Syntax is relatively simple, if unconventional

3. Unifying the three components

(a) API

i. Built on top of OSC

ii. Simple information for each note

A. Originally more complex information retrieved in ‘P’ and passed to ‘f’

- Phrase density

B. Moved that retrieval process to ‘f’

iii. Originally built for communication between ‘P’ and ‘f’; reused for ‘f’ and ‘Q’

(b) Editing each component to better communicate with others

4. Refinement

(a) Through performance practice

(b) Adding features