Choir Singers Pilot – An online platform for choir singers practice

Matan Gover, Álvaro Sarasúa, Hector Parra, Jordi Janer, Oscar Mayor
Voctro Labs
Barcelona, Spain
alvarosarasua@gmail.com
hector@parra.cat
(matan.gover, jordi.janer,
oscar.mayor)@voctrolabs.com

Helena Cuesta, Maria Pilar Pascual, Aggelos Gkiokas, Emilia Gómez
Universitat Pompeu Fabra
Barcelona, Spain
(helena.cuesta, mariapilar.pascual,
aggelos.gkiokas, emilia.gomez)@upf.edu

ABSTRACT

We present the Choir Singers Pilot, a web-based system that assists choir singers in their individual learning and practice. Our system is built using modern web technologies and provides singers with an interactive view of the musical score along with an aligned audio performance created using state-of-the-art singing synthesis technology. The Web Audio API is used to dynamically mix the choir voices and give users control over sound parameters. In-browser audio latency compensation is used to keep user recordings aligned to the reference music tracks. The pitch is automatically extracted from user recordings and can then be analyzed using an assessment algorithm to provide intonation ratings to the user. The system also facilitates communication and collaboration in choirs by enabling singers to share their recordings with conductors and receive feedback. Our work, in the larger scope of the TROMPA project, aims to enrich musical activities and promote use of digital music resources. To that end, we also synthesize thousands of public-domain choir scores and make them available in a searchable repository alongside relevant metadata for public consumption.

1. INTRODUCTION

Classical music is one of the key elements of the European cultural heritage, which is still actively performed by many people today. Much of the classical repertoire is in the public domain by now, and massive amounts of musical scores and recordings have been digitized and become available to the general public. The TROMPA project[1] intends to provide means for large-scale enrichment of public-domain music archives by combining human and machine computation. The project involves five key user audiences and use cases: music scholars, choir singers, piano players, content owners, and music enthusiasts. Its main goal is to massively enrich publicly available musical heritage. This is done by fostering the interaction between state-of-the-art technology, which provides automatic processing of existing music content at a large scale, and music-loving citizens, who benefit from existing content and technology outputs, and provide feedback and high-level expert annotations.

This paper describes the Choir Singers Pilot (CSP). The goal of the CSP is to assist amateur choir singers during individual practice. In terms of state-of-the-art technologies, the pilot integrates singing synthesis and analysis techniques to support individual choir rehearsals. They allow, on the one hand, to synthesize existing public-domain scores, such as the ones available on IMSLP[2] or Choral Public Domain Library[3], generating audio material as a guide for rehearsal. On the other hand, singing analysis techniques provide singers feedback on their singing performance.

In terms of community features, it allows the choir conductor to create repertoires and to listen to performances by choir members, providing feedback to them. The current version of the system supports repertoire in five different languages: Latin, English, German, Spanish, and Catalan. We find similar systems for choir practice, which are mostly available as mobile apps. Some of these applications, such as Singerhood[4] or carus music[5], offer a fixed catalogue of professionally recorded performances available as separated tracks. Other web-based systems focus on performance assessment. For example, TuneIn [8] provides real-time intonation feedback for singers. On the other hand, My Choral Coach[6] is designed as a tool for conductors and instructors, permitting to upload new scores. It incorporates recording functionality to share performances for review and other choir management features. In both of these systems, scores are played back using synthetic instrumental sounds.

Developing a web-based solution for the CSP was motivated by several factors. First, our tool should be compatible with multiple devices, especially laptops and tablets, and allow rapid prototyping. Second, the CSP is framed within the TROMPA project which has a strong commitment in outreach activities. The web was the natural choice for its

[1] https://trompamusic.eu/

[2] https://imslp.org/

[3] http://www.cpdl.org/

[4] https://www.singerhood.com/

[5] https://www.carus-verlag.com/en/carus-plus/carus-music-the-choir-app/

[6] https://matchmysound.com/my-choral-coach/

Web Audio Conference WAC-2021, July 5–7, 2021, Barcelona, Spain.

widespread accessibility.

The main contribution of the CSP is combining data presentation and collection of multi-modal information for musical scores. Putting the digital music score in the center, we provide different visual representations (music notation and piano roll) alongside an accurate audio representation: singing synthesis containing both melody and lyrics. Furthermore, thanks to being web-based, we can easily collect user data in the form of synchronized voice recordings for the said digital music score, which can then be further analyzed, assessed, and distributed to other choir members.

A second contribution of our work is promoting accessibility of public-domain digital resources. The CSP can provide access to a large multi-lingual repertoire of choral music. The Choral Public Domain Library[3] has $34,531$ scores, from which we prepared a subset of $4,107$ scores in MusicXML format in the five supported languages. We synthesized these scores and uploaded them into the TROMPA Contributor Environment for public consumption. The CSP will in the future allow loading any of these public domain scores for playing and rehearsing.

2. GATHERING CHOIR REQUIREMENTS

In order to define the roadmap for development of the CSP, we ran a focus group workshop with 15 participants (13 singers and 2 conductors). The main goal was to identify the most useful functionalities for individual rehearsal, as well as to identify possible relevant non-functional requirements. The workshop consisted of a mock-up test, a requirements discussion session, and a final questionnaire.

We developed a mock-up of the CSP that participants interacted with by performing some simple tasks, such as browsing, visualizing, and listening to a piece. The mock-up was oriented towards evaluating the intuitiveness of the interface in terms of navigation and score visualization. It included a pre-loaded piece with piano roll and score visualizations.

All participants agreed on the usefulness of a technological tool for choir rehearsal for both amateur and professional musicians, and the majority said they would use it. Some participants were concerned by the piano roll visualization, since they believed it is important for musicians to learn music notation. Furthermore, some were concerned that users would imitate the peculiarities and expressiveness of the synthesized voices.

Participants also agreed that tablet devices would be the best fit for using the CSP due to their form factor. However, they also pointed out that desktop, laptop, and mobile versions would be useful. This was a strong motivation for deciding to develop the CSP as a web app.

3. SYSTEM DESCRIPTION

The CSP consists of several components. The overall system architecture is shown in Figure 1. The frontend is a single-page web application written in TypeScript and built using the React[7] library. The frontend communicates with a GraphQL[8] backend API that is built with Hasura[9] and stores its data in a PostgreSQL database.

The frontend consists of several pages. The Repertoire page allows users to browse their choir's repertoire and choose a piece. The Piece page (see Figure 2) displays the score of a specific piece and allows users to browse the score, play it back, practice it, and record their own rehearsal.

3.1 Web Audio functionality

The main functionality of the Piece page is playing back the score and recording a rehearsal. In order to play back scores, they are first synthesized and uploaded to cloud storage and their URLs are stored in the CSP backend alongside the piece metadata (for details on the synthesis see Section 4.1). In order to facilitate learning of choral parts, the CSP allows users to control the volume of each part separately. For example, if a user sings the Soprano part in the choir they may mute all other voices and listen to their part alone. For this reason, each part in the score is synthesized as a separate file, and the parts are dynamically mixed in the browser by the CSP frontend using the Web Audio API. We use the Tone.js[10] library that provides extra functionality on top of the Web Audio API.

In addition to adjusting the volume of each part, users may also adjust panning. By default, voices are panned to emulate the common stereo image of a real choir, that is, the first part (e.g., Soprano) is panned all the way to the left and the last part (e.g., Bass) all the way to the right. All the other parts are spaced equally in-between.

Furthermore, artificial reverb is added on the mixed audio to make the choir singing sound more natural, as choirs often sing in highly reverberant spaces such as churches and the raw synthesized audio files do not have any reverb. The reverb is implemented using `Tone.Reverb` which uses a Web Audio `ConvolverNode` behind the scenes with a dynamically generated impulse response.

The Piece page also allows users to activate a metronome which plays a tick sound on every beat, and an accented tick on the beginning of every measure. This is implemented by calculating the time of each beat in the score (see Section 3.2 below) and scheduling corresponding events using the Tone.js transport. The actual ticks are played using a Tone.js `Synth` which uses a Web Audio `OscillatorNode`.

3.1.1 *Recording and latency compensation*

The system allows users to record their practice sessions and store them as Rehearsal entities in the system. These rehearsals can later be played by the users themselves, or by the choir conductor (if the user so chooses). Since the recording can be played back with the rest of the voices, the system must ensure that each recording is aligned to the score while compensating for hardware and software latency issues. Specifically, we need to measure the time shift between audio that is played back in the user's output device and audio recorded by the input device, hence both input and output latency are important factors. Latency in Web Audio applications is often inconsistent and unpredictable due to the wide array of end user devices, browsers, and other factors [4].

In order to compensate for latency, we first measure each user's recording latency once and then store this measurement alongside each recorded rehearsal. The user can re-run the calibration at any time if necessary. The latency is measured using a method in which a chirp sound is played on the

[7] https://reactjs.org/

[8] https://graphql.org/

[9] https://hasura.io/

[10] https://tonejs.github.io/

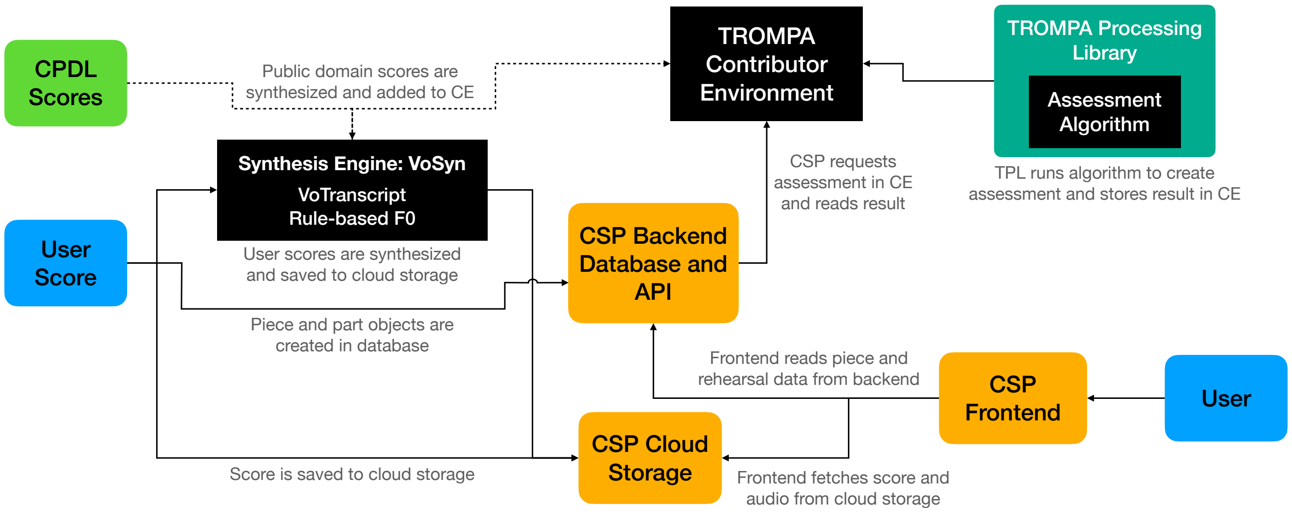
CPDL Scores
Public domain scores are synthesized and added to CE
TROMPA Contributor Environment
TROMPA Processing Library
Assessment Algorithm
Synthesis Engine: VoSyn
VoTranscript
Rule-based F0
User Score
User scores are synthesized and saved to cloud storage
CSP requests assessment in CE and reads result
TPL runs algorithm to create assessment and stores result in CE
CSP Backend Database and API
Piece and part objects are created in database
Frontend reads piece and rehearsal data from backend
CSP Frontend
User
CSP Cloud Storage
Score is saved to cloud storage
Frontend fetches score and audio from cloud storage

Figure 1: Overall system architecture. Scores are synthesized by the VoSyn engine (see Section 4.1) and inserted into cloud storage. Piece metadata is stored in the CSP backend database. For intonation rating, pitch is extracted on the CSP backend and sent to the TROMPA Contributor Environment for processing (see Section 4.2). Finally, the user accesses the CSP frontend which interacts with the backend using the API.

user's speakers, and simultaneously recorded using the microphone (the user is first asked to unplug their headphones, if any). The recorded chirp is then aligned to the played chirp using cross-correlation (computed in the frequency domain for increased speed), and taking the maximum point thereof. The chirp sound is generated in the browser by running an `OscillatorNode` inside an `OfflineAudioContext` and saving the result in a buffer.

3.2 Score functionality

The score is dynamically rendered onto the Piece page from the source score encoding by converting it to SVG using Verovio [7]. Verovio is compiled from C++ to WebAssembly and runs completely in the browser. It supports several score input encodings such as MEI and MusicXML (all input scores are internally converted to MEI on the fly). We use primarily MusicXML scores as they can be exported from most notation programs. To facilitate navigation in the score, it is automatically split into pages according to the size of the browser window and a slider allows users to turn pages.

Additionally, the score is independently parsed by the CSP frontend in order to extract information that is not exposed by Verovio. This is done by exporting the MEI raw string from Verovio and parsing it as an XML document. We implemented TypeScript wrapper classes for various MEI entities (e.g., `Note`, `Rest`, `Measure`) to facilitate type-safe access to the score information. This information is used to provide several features: for playback and recording, we must associate every measure number with a time offset in the audio. For pagination and bar selection, we need to count the total number of measures and associate each measure number to a page. For the metronome, we count the beats in each measure and differentiate between strong and weak beats. For highlighting the staff of the selected part, each staff in the score is matched with the corresponding part's audio.

In order to aid singers in following the score while playing and recording, notes are highlighted automatically in the score as they are played. This is done by extracting note onset and offset times from the parsed score and scheduling corresponding events on the Tone.js transport to highlight and de-highlight every note as appropriate by applying a class to the note's SVG element.

The most important feature enabled by our custom score parser is the Piano Roll view in the Piece page. By switching to the piano roll, users who are not necessarily familiar with traditional music notation can view the score in an intuitive display in which the horizontal axis denotes time and the vertical axis denotes pitch (see Figure 3). The piano roll shows each part's notes in a distinct color and highlights the active part. In addition, it shows each note's lyrics on the note itself and below the piano roll.

The piano roll is dynamically rendered to an SVG from the score. In addition to using the parsed MEI score, the rendering also uses the corresponding rendered MIDI, which is exported from Verovio. Using the rendered MIDI saves us the work of figuring out note pitches and durations (especially in non-trivial cases such as tied notes). However, the MIDI does not contain information such as lyrics and measure boundaries. Since we do display measure boundaries and lyrics in the piano roll, those must be extracted from the MEI.

4. SINGING SYNTHESIS AND ANALYSIS

4.1 Synthesis engine

In order to synthesize the scores collected from choirs, we use the VoSyn [6] system by Voctro Labs that is based on the neural-parametric singing synthesizer (NPSS) [1]. We train four different NPSS voice models, one model for each of the common choir voice types: soprano, alto, tenor, and bass. Training data was recorded by four professional choir singers from Cor Francesc Valls in Barcelona. The recorded

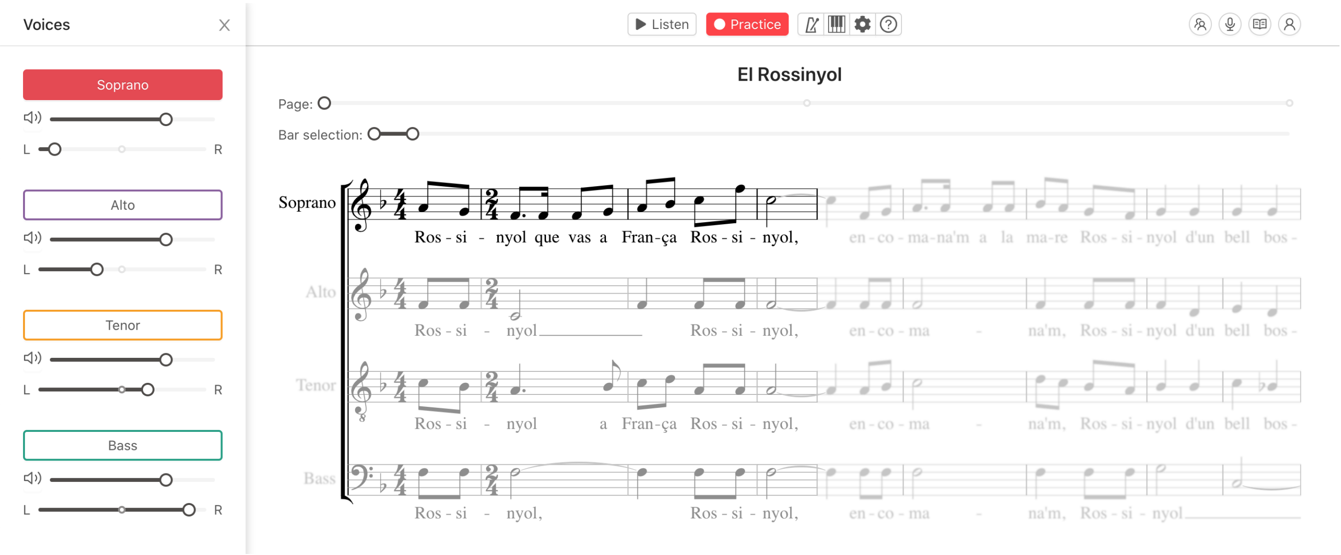
Voices
Soprano
L R
Alto
L R
Tenor
L R
Bass
L R
Listen
Practice
El Rossinyol
Page:
Bar selection:
Soprano
Ros - si - nyol que vas a Fran - ça Ros - si - nyol, en - co - ma - na'm a la ma - re Ros - si - nyol d'un bell bos -
Alto
Ros - si - nyol Ros - si - nyol, en - co - ma - na'm, Ros - si - nyol d'un bell bos -
Tenor
Ros - si - nyol a Fran - ça Ros - si - nyol, en - co - ma - na'm, Ros - si - nyol d'un bell bos -
Bass
Ros - si - nyol, Ros - si - nyol, en - co - ma - na'm, Ros - si - nyol

Figure 2: Screenshot of the CSP showing the piece El Rossinyol. Soprano is selected as the active voice. Measures 1–3 are selected for practice and are highlighted in the score.

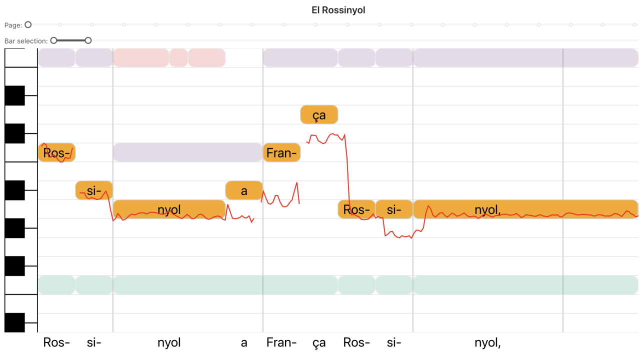

Figure 3: Piano roll display on the piece page, showing the first bars of El Rossinyol (the same score shown in Figure 2). The tenor part is selected, and a rehearsal pitch analysis is displayed on top of the sung notes.

material consisted of short musical phrases in four languages: English, German, Catalan, and Spanish. Each singer contributed approximately 40 minutes of singing for each of the four languages. Following the recording sessions, every musical excerpt was manually annotated to produce a label file containing the start and end times of each phone, using a phonetic dictionary specially designed for each language.

The input of NPSS is a smooth F0 curve and phonetic labels. To synthesize from a score, we use a rule-based system to devise the smooth F0 with human-like features such as smooth note transitions and vibrato. A separate module named VoTranscript transcribes the score's lyrics into phonetic labels using transcription rules and dictionaries. We then assign a set of phones to each note using a heuristic which matches the transcribed syllables onto the notes in the score. This system must take into account many edge cases such as melismas, tied notes, diphthongs which must be split into separate syllables, irregularities in the score encoding, and elisions (two syllables in one note).

4.2 Analysis and intonation rating

One of the requirements was to provide feedback about the user performances. In this case, we only focus on intonation aspects, for which we compare the F0 curve extracted from the user recording to the reference pitch from the music score. Additional performance aspects such as dynamics or timings are not considered in the current version.

The assessment mechanism consists of two main parts that correspond to two distinct systems. The first is the extraction of the F0 curve from a recorded voice track, which is subsequently fed to the second part, which is the intonation assessment algorithm. The F0 analysis is carried out on the server using the VoDesc API provided by Voctro Labs[11]. The F0 curve is stored in JSON format containing timestamps and F0 values at a frame rate of 5.8ms. This F0 curve is then input into the intonation assessment algorithm, which is integrated into the TROMPA Processing Library (TPL).

The TPL is a component of TROMPA for controlling and triggering algorithms that can be used by different systems. It uses the TROMPA Contributor Environment data infrastructure for scheduling jobs and storing results. Every performance assessment job is represented as a node in the database, which is linked to both the reference score node and the performance data node (i.e. the F0 data JSON file). Every time a new job node is created, the TPL reads the associated data and triggers the assessment algorithm. After the assessment is finished, the TPL uploads the result to an S3 bucket, creates a node with a reference to the result, and links this result with the specific job.

For the extraction of the F0 curve from a recorded voice track we use the signal processing based method Spectral-Amplitude Autocorrelation (SAC) [9, 3], which is an F0-estimation method that comprises several steps: multi-resolution spectral analysis, frame-by-frame F0 candidate estimation, voicing detection, and post-processing. This method gives comparable results to other more recent data-

[11] https://cloud.voctrolabs.com/docs/api/

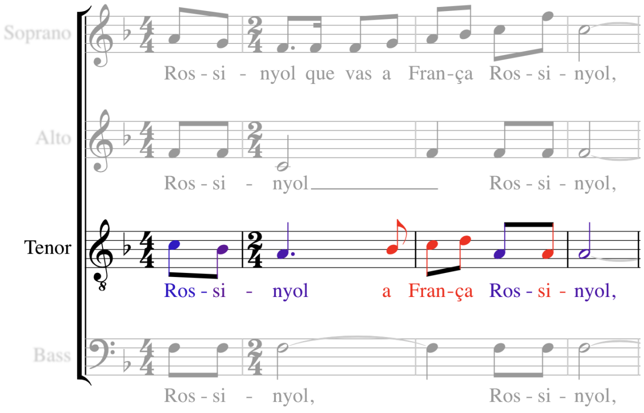
Soprano
Ros - si - nyol que vas a Fran - ça Ros - si - nyol,
Alto
Ros - si - nyol Ros - si - nyol,
Tenor
Ros - si - nyol a Fran - ça Ros - si - nyol,
Bass
Ros - si - nyol, Ros - si - nyol,

Figure 4: An example intonation assessment display corresponding to the rehearsal displayed in Figure 3. Each note is colored according to its intonation score, on a scale from most accurate (blue) to least accurate (red).

driven approaches such as CREPE [5].

The assessment algorithm is based on methods presented in [10, 2]. It has several input parameters: the F0 curve, the MusicXML score of the song, the start and end bars of the rehearsal, the estimated latency in seconds, and the voice part of the singer. It computes an intonation score for each note in the performance—a value between 0 (inaccurate intonation) and 1 (accurate intonation). Note that we explicitly measure this accuracy as the deviation from the score, thus obtaining an objective score. We do not consider any perceptual aspects, nor any temperament other than equal temperament in the current version. Also, we do not currently differentiate between a note that is sung flat and a note that is sung sharp. This will be addressed in the future.

The first step of the algorithm is selecting the excerpt of the score delimited by the start and end bars. As a result, we obtain a list of N triplets: `[note_onset, note_offset, pitch]`, where N is the number of notes in the performance. Then, we convert this list into a time series to ease further steps, using the frame rate from the F0 curve. Consequently, we obtain a list of tuples `[timestamp, score_pitch]`, where the F0 from the score is repeated for all frames within the same note.

Then, for each frame i of the F0 curve, we compute the intonation score, S_i, as the ratio between the target pitch (from the score, $F0_{ref}$) and the performance pitch (from the F0 curve, $F0_{perf}$) using the following equation:

$$S_i = 1200 \cdot log_2 \frac{F0_{perf_i}}{F0_{ref_i}} \tag{1}$$

which measures the difference between both values in *cents*. Note that we previously adjust the F0 curve according to the estimated latency (see Section 3.1.1). Using the note boundaries from the score, we compute the median of the frame-wise deviation values within each note. By default, we set a maximum deviation of 100 cents (one semitone), which defines the lowest intonation score.

The assessment algorithm outputs a JSON file with a list of N tuples `[start_time, intonation_score]`, one per note. Subsequently, the TPL uploads the result and updates the backend as described above. The result of the assessment is displayed in the CSP on the score by adjusting the color of each note according to its intonation score (see Figure 4).

5. USAGE AND FEEDBACK BY CHOIRS

The CSP is currently being actively used as a pilot by several selected choirs. We are gathering feedback from both singers and conductors to assess what remains to be done. The pilot participants appreciate the possibilities offered by the CSP. Particularly, they agree that individual work allows to unify the level of the choir and dedicate more rehearsal time to work on other musical aspects apart from “learning the notes”, such as dynamics, phrasing, and interpretation.

The COVID-19 pandemic has generated many threats to the choral field but also opportunities. Choirs’ schedules became unpredictable as many concerts have been cancelled, which caused cultural, social, and financial tensions. Choirs have been forced to cancel in-person rehearsals and lose the gratifying aspects of choral singing as a social experience. The CSP, though not comparable to an in-person rehearsal, still allows choirs to continue working on their repertoire in spite of the pandemic. In the process, singers improve their individual learning, listening, and score-reading skills, while also keeping in communication with other singers and the conductor. Thus, the CSP is seen more than ever as a tool that can help choirs in the present circumstances.

We collect feedback on the CSP from choirs by following up personally with choir representatives. Initial feedback can be differentiated into two types: problems with the content (errors in the scores, wrong pronunciation in synthesized voices, and so on), and usability (user interface suggestions, navigation, or other ideas proposed by users). Feedback has also revealed a technological gap between generations: the CSP generates some reluctance in older singers who are not accustomed to using digital scores, albeit this reluctance seems to be diminishing, perhaps in part due to the necessities of the pandemic. Furthermore, conversations with choir members have revealed uses for the CSP which were not initially intended. For example, one choir suggested using the synthesized voices from the CSP to replace a choir section that is absent from a sectional rehearsal.

6. CONCLUSIONS AND FUTURE DIRECTIONS

We presented the Choir Singers Pilot, a system that helps choir singers in their individual practice. Our desire to make the system as widely accessible as possible made web technologies a natural choice for implementation. As a pilot, the system is currently being actively used and evaluated by several selected choirs, and the gathered feedback is very positive overall. The CSP will be further developed as a commercial product under the name of Cantamus.[12]. We intend to open up the system to more choirs by implementing automatic synthesis of user-submitted scores. We also intend to connect the system with public-domain scores available in the TROMPA Contributor Environment. Another direction we intend to pursue is transferring more processing to the client side by compiling our algorithms such as pitch extraction to WebAssembly.

[12] https://cantamus.app

7. ACKNOWLEDGMENTS

This work is supported by the Towards Richer Online Music Public-domain Archives (TROMPA H2020 770376) project.

8. REFERENCES

[1] M. Blaauw and J. Bonada. A neural parametric singing synthesizer. In *Proceedings of the 18th Annual Conference of the International Speech Communication Association (Interspeech 2017)*, Stockholm, Sweden, 2017.

[2] H. Cuesta, E. Gómez, A. Martorell, and F. Loáiciga. Analysis of intonation in unison choir singing. In *Proceedings of the International Conference of Music Perception and Cognition (ICMPC)*, pages 125–130, Graz, Austria, 2018.

[3] E. Gómez and J. Bonada. Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing. *Computer Music Journal*, 37(2):73–90, 2013.

[4] W. Henderson. Latency and synchronization in web audio. In *Proceedings of the International Web Audio Conference*, WAC '18, Berlin, Germany, 2018.

[5] J. W. Kim, J. Salamon, P. Li, and J. P. Bello. CREPE: A convolutional representation for pitch estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 161–165, Calgary, Canada, 2018.

[6] O. Mayor, J. Janer, H. Parra, and Á. Sarasúa. Voiceful: Voice analysis, transformation and synthesis on the web. In *Proceedings of the International Web Audio Conference (WAC)*, Berlin, Germany, 2018.

[7] L. Pugin, R. Zitellini, and P. Roland. Verovio: A library for engraving MEI music notation into SVG. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[8] S. Rosenzweig, L. Dietz, J. Graulich, and M. Müller. TuneIn: A web-based interface for practicing choral parts. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Montreal, Canada, 2020.

[9] F. Villavicencio, J. Bonada, J. Yamagish, and M. Pucher. Efficient pitch estimation on natural opera-singing by a spectral correlation based strategy. Technical Report Vol.2015-SLP-107 No. 1, IPSJ SIG, 2015.

[10] S. Wager, G. Tzanetakis, S. Sullivan, C. Wang, J. Shimmin, M. Kim, and P. Cook. Intonation: A dataset of quality vocal performances refined by spectral clustering on pitch congruence. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 476–480, 2019.