MATCHMAKER: A PYTHON LIBRARY FOR REAL-TIME MUSIC ALIGNMENT

Jiyun Park[1] **Carlos Cancino-Chacón**[2] **Taegyun Kwon**[1] **Juhan Nam**[1]

[1] Graduate School of Culture Technology, KAIST, South Korea

[2] Institute of Computational Perception, Johannes Kepler University, Linz

june@kaist.ac.kr, carlos_eduardo.cancino_chacon@jku.at

ABSTRACT

Music alignment is a fundamental MIR task, and real-time music alignment is a necessary component of many interactive applications (e.g., automatic accompaniment systems, automatic page turning). This paper introduces Matchmaker, an open source Python library for real-time music alignment. Unlike offline alignment methods, for which state-of-the-art implementations are publicly available, real-time (online) methods have no standard implementation, forcing researchers and developers to build them from scratch for their projects. We aim to provide efficient reference implementations of score followers for use in real-time applications which can be easily integrated into existing projects.

1. INTRODUCTION

Automatic music alignment refers to the task of linking or matching two musical signals of the same musical work. This alignment can involve matching different performances of the same piece or matching the performance of a piece with its musical score. Music alignment is one of the fundamental tasks in Music Information Retrieval (MIR) and serves as the basis for several applications, ranging from the analysis of music performance [1, 2], real-time interactive systems [3] or powering visualizations [4, 5]

There are two main types of music alignment:

- **Offline**: Alignment of two recordings or documents, such as audio recordings, MIDI performances, or MusicXML scores.
- **Online**: Alignment of a live (i.e., real-time) performance to the music encoded in a target document (e.g., a pre-annotated audio recording or a symbolic score). The problem of real-time online alignment is known in the MIR literature as *score following* and can be particularly useful in live interactive settings, such as automatic accompaniment systems.

While there have been several recent works addressing offline music alignment for both symbolic and audio formats, there have not been many new contributions to real-time alignment since the mid-2010s. In recent years, there have some important efforts in providing robust open source toolboxes for offline music alignment. In the audio domain, Müller et al. [6] introduced the Sync Toolbox, which provides a reference implementation of DTW-based alignment. In the symbolic domain, Parangonar [1] and Nakamura et al.'s [7] tools for symbolic music. No open-source libraries for real-time score following.

This work introduces *Matchmaker*, an open-source Python library for real-time music alignment. The library provides robust reference implementations and can be easily integrated with other software. This library also addresses the case of both audio and symbolic alignment.

The rest of this paper is structured as follows

2. MUSIC REPRESENTATIONS

2.1 Audio

Our implementation uses two main categories of audio features: Traditional signal processing features, represented by chromagrams, and handcrafted features such as `LogLinearSpectralOnset` and `ChromaOnset`. Chromagram is the most basic feature used in score following, which ensures fast and high performance for non-percussive pieces such as piano pieces. However, the most important timepoint for alignment if the timing of the onsets of tones. We added handcraft feature in order to consider the energy peak of onset event and decay of subsequent signals. In specific, we perform a first-order difference followed by half-wave rectification on a given sequence. `LogLinearSpectralOnset` is an implementation of the function used in [8], which is designed to be a scale that captures the information of polyphony better than mel-spectrogram.

2.2 Symbolic

In this work, we use two common symbolic music processing features for capturing information from MIDI performances: piano rolls and pitch class distributions. Pi-

Late-Breaking / Demo Session Extended Abstract, ISMIR 2024 Conference

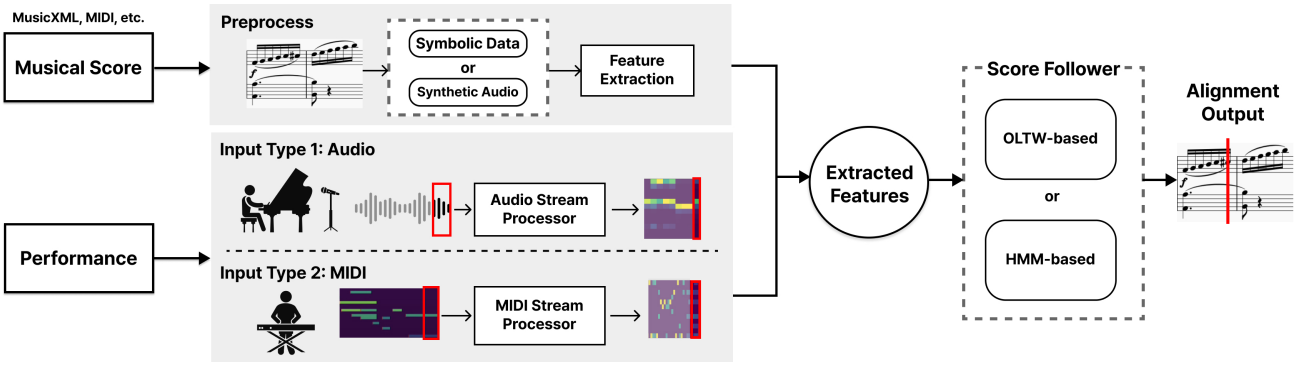

Figure 1. Overview of Matchmaker

ano rolls are a 2D arrays that have pitch and time information, and can be considered the symbolic equivalent of magnitude spectrograms from audio. Pitch class distributions are the symbolic equivalent of chromagrams. The user can specify the frame rate for both of these representations, and we use a default frame rate of 100 frames per second. Figure **??** shows a comparison of these features.

3. ALIGNMENT METHODS

3.1 Dynamic Time Warping

3.2 Hidden Markov Model

4. MATCHMAKER

In this section we provide a description of the Matchmaker library. The library was born out of the need for modular and flexible real-time alignment methods on different applications (e.g., accompaniment systems).

4.1 Package Structure

The package is structured as illustrated in Figure **??**. The three main elements

1. `Stream` objects that handle input real-time inputs. For audio there is
2. Feature extractors `Processor`
3. Online alignment `OnlineAlignment`

The following is example code to run `Matchmaker` in simulation mode with audio input:

For efficiency, some methods are implemented in Cython [9], a superset of Python designed for C-like performance by incorporating C data types and optimizing Python code execution.

4.1.1 Simulating Mode for Input Stream

In the development and testing phase of the Matchmaker library, a `MockStream` can be utilized to simulate real-time stream processing. This approach allows for the evaluation of the system's performance and robustness without the need for live stream input. By using pre-recorded audio files or MIDI files, all the alignment methods can be tested under controlled conditions.

```python
from matchmaker import Matchmaker

score_path = "path/to/score.mid"
performance_path = "path/to/input.wav"

# Specify input stream
audio_stream = AudioStream()

# specify a score follower
matchmaker = Matchmaker(
    score_path,
    performance_path,
    algorithm="oltw_arzt",
    mock=True,
)

# Run the matchmaker with stream
audio_stream.start()
matchmaker.start()

audio_stream.stop()
matchmaker.stop()
```

Figure 2. Matchmaker example

4.1.2 Evaluation

Given the proper pair of annotation files for the score and performance audio, the performance of the matchmaker library can be evaluated. The annotated labels from the score and performance audio, respectively, are considered as ground truth and the matchmaker's output is calculated as a prediction. The annotations file can be at any granularity the user defines, from beat-, measure-, or even note-level, but must be consistent across the file pair. By comparing the aligned output with the annotated reference, various evaluation metrics such as precision, reliability, and workload can be calculated.

4.1.3 Visualization

5. ACKNOWLEDGMENTS

This work has been partially supported by the Austrian Science Fund (FWF), grant agreement PAT 8820923 ("*Rach3: A Computational Approach to Study Piano Rehearsals*").

6. REFERENCES

[1] S. D. Peter, C. E. Cancino-Chacón, F. Foscarin, A. P. McLeod, F. Henkel, E. Karystinaios, and G. Widmer, "Automatic Note-Level Score-to-Performance Alignments in the ASAP Dataset," *Transactions of the International Society for Music Information Retrieval*, vol. 6, no. 1, pp. 27–42, Jun. 2023. [Online]. Available: http://transactions.ismir.net/articles/10.5334/tismir.149/

[2] C. E. Cancino-Chacón, M. Grachten, W. Goebl, and G. Widmer, "Computational Models of Expressive Music Performance: A Comprehensive and Critical Review," *Frontiers in Digital Humanities*, vol. 5, p. 25, Oct. 2018. [Online]. Available: https://www.frontiersin.org/article/10.3389/fdigh.2018.00025/full

[3] C. Cancino-Chacón, S. Peter, P. Hu, E. Karystinaios, F. Henkel, F. Foscarin, N. Varga, and G. Widmer, "The ACCompanion: Combining Reactivity, Robustness, and Musical Expressivity in an Automatic Piano Accompanist," in *Proceedings of the 32nd International Joint Conference on Artificial Intelligence IJCAI-23*, Macao, S. A. R., 2023.

[4] O. Lartillot, C. Cancino-Chacón, and C. Brazier, "Real-Time Visualisation of Fugue Played by a String Quartet," in *Proceedings of the Sound and Music Computing Conference (SMC 2020)*, Online, 2020.

[5] A. Arzt, H. Frostel, T. Gadermaier, M. Gasser, M. Grachten, and G. Widmer, "Artificial Intelligence in the Concertgebouw," in *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI-15)*, Buenos Aires, Argentina, 2015.

[6] M. Müller, Y. Özer, M. Krause, T. Prätzlich, and J. Driedger, "Sync toolbox: A python package for efficient, robust, and accurate music synchronization," *Journal of Open Source Software*, vol. 6, no. 64, p. 3434, 2021. [Online]. Available: https://doi.org/10.21105/joss.03434

[7] E. Nakamura, K. Yoshii, and H. Katayose, "Performance Error Detection and Post-Processing for Fast and Accurate Symbolic Music Alignment," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, Suzhou, China, 2017.

[8] S. Dixon, "Live tracking of musical performances using on-line time warping," in *Proceedings of the 8th International Conference on Digital Audio Effects*, vol. 92. Citeseer, 2005, p. 97.

[9] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, and K. Smith, "Cython: The best of both worlds," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 31–39, 2011.