DYNAMIC PITCH WARPING FOR EXPRESSIVE VOCAL RETUNING

Daniel Hernan Molina Villota
Institut Jean Le Rond d'Alembert
Equipe Lutheries-Acoustique-Musique
Sorbonne Université
Centre National de la Recherche Scientifique
Paris, France
daniel.molina_villota
@sorbonne-universite.fr

Christophe d'Alessandro
Institut Jean Le Rond d'Alembert
Equipe Lutheries-Acoustique-Musique
Sorbonne Université
Centre National de la Recherche Scientifique
Paris, France
christophe.dalessandro
@sorbonne-universite.fr

Olivier Perrotin
Université Grenoble Alpes
CNRS, Grenoble INP
GIPSA-lab
Grenoble, France
olivier.perrotin
@grenoble-inp.fr

ABSTRACT

This work introduces the use of the Dynamic Pitch Warping (DPW) method for automatic pitch correction of singing voice audio signals. DPW is designed to dynamically tune any pitch trajectory to a predefined scale while preserving its expressive ornamentation. DPW has three degrees of freedom to modify the fundamental frequency (f_0) signal: detection interval, critical time, and transition time. Together, these parameters allow us to define a pitch velocity condition that triggers an adaptive correction of the pitch trajectory (pitch warping). We compared our approach to Antares Autotune (the most commonly used software brand, abbreviated as ATA in this article). The pitch correction in ATA has two degrees of freedom: a triggering threshold (flextune) and the transition time (retune speed). The pitch trajectories that we compare were extracted from autotuned-in-ATA audio signals, and the DPW algorithm implemented over the f_0 of the input audio tracks. We studied specifically pitch correction for three typical situations of f_0 curves: staircase, vibrato, free-path. We measured the proximity of the corrected pitch trajectories to the original ones for each case obtaining that the DPW pitch correction method is better to preserve vibrato while keeping the f_0 free path. In contrast, ATA is more effective in generating staircase curves, but fails for not-small vibratos and free-path curves. We have also implemented an off-line automatic picth tuner using DPW.

1. INTRODUCTION

Pitch correction (or automatic pitch tuning) is nowadays one of the most commonly used digital audio effects for vocal music. Initially known as the "Cher" effect, the audible distortion produced by sharp pitch transition in retuned singing became appreciated on its own in popular electronic music. The sharp transition is a case of use where all minor expressive singing variations are flattened. Noticeable gliding appears often in the transitions between notes. The success of Autotune in the music industry has sparked much discussion and debate. Some argue that it is a tool that helps artists achieve a perfect pitch singing, while others criticise its use as it can lead to a loss of natural expression and emotion in the music. Despite this, Autotune has become a staple in modern music production and is used in various genres such as pop, hip-hop, and electronic music [1]. Although it is a common practice to use DAFx effects which involve perceptual features such as [2] melody (pitch), source (timbre, [3]), or space [4],pitch correction is one of the most commonly used. I became a stylistic signature for many popular music genre.

Antares Autotune (ATA)[1] is a digital audio effect developed by H. Hildebrand in 1997 [5] and its enduring popularity has spanned over 25 years. ATA uses an autocorrelation method that was initially developed for seismic imaging, with the help of short-time Fourier transform. Although the initial purpose of ATA was not to enrich the voice with a new vocoder-like audio effect but to correct out-of-tune melodies, the unique electronic texture produced has been embraced in popular music and has even become a hallmark of specific musical styles, often employed systematically. ATA offers two use cases: one the one hand pitch correction is used for better rendering of out of tune singing and on the other hand the distortion effect occurring extreme correction situations is appreciated on its own. The need for melodic correction also appeared in digital music instruments (DMI) [6, 7, 8, 9]. These DMIs use interfaces with particular features that involve learnability, explorability, and controllability [10]. A new pitch tuning correction, Dynamic Pitch Warping (DPW) [11], has been developed for performative vocal synthesis in Cantor Digitalis [8] where the fundamental frequency (pitch) is controlled in real-time with the help of a stylus on a graphic tablet. Pitch correction helps for singing accurate notes. However, it is very important to preserve small expressive ornaments like vibrato [12] without flattening the notes to preserve naturalness.

The purpose of this paper is to study the DPW pitch correction method. This method was designed to preserve expressive variations like vibrato while adjusting the main shape of the f_0 curve to a predefined scale. We identify three cases of particular interest: abrupt pitch transitions (staircase notes), notes with vibrato and free path curves that should not be corrected. The results of this paper allow us to open perspectives for developing dynamic and singer-controlled vocal digital audio effects that are able to preserve expressive ornaments in real-time. Section 2 presents a review of the pitch correction method studied (ATA and DPW). Section 3 compares DPW and ATA on typical pitch patterns. Section 4 presents the off-line implementation of DPW for audio signals.

2. PITCH CORRECTION SYSTEMS

An audio pitch correction system contains three parts: a pitch detection algorithm (PDA), a pitch correction algorithm, and finally

[1] https://www.antarestech.com/ last checked: 6 April 2023

a pitch warping modification (vocoder). The present paper aims to apply DPW as a pitch correction algorithm for vocal speech intonation. DPW offers three control parameters when other correction methods have one or two parameters. DPW uses an adaptive function, the term "adaptive" is related to the adaptive digital audio effects (aDAFx) that are recent solutions designed to respond to changes in the input signal and adjust specific audio parameters accordingly to it, thanks to specific denominated adaptive functions. These kind of effects are more dynamic and responsive that the traditional DAFx, some examples of aDAFx being the compressor, the expander and the limiter (auto-adaptive on loudness).

Along this line, several DMIs have introduced the use of pitch correction methods to improve the expressivity of musical user interfaces. That is the case for devices such as the Continuum Fingerboard [6, 7][2], the Seaboard[13][3], Garageband[4], TouchKeys[14], and Cantor Digitalis [8][5]. The latter is particularly interesting since it uses a Dynamic Pitch Warping method to correct the continuous position of the pitch controller relative to a pitch scale. The corresponding adaptive warping function proposed by Perrotin and d'Alessandro [11] attracts real pitch values towards integer values, using a MIDI scale. The integer values are tuned notes. DPW is based on a pitch velocity condition expressed as the pitch stability within a pitch interval during a critical time threshold before triggering the automatic correction. We will review the warping methods applied in ATA and DPW in the following two subsections.

2.1. Autotune Antares

Autotune was developed by H. Hildebrand using techniques originally developed for mapping the Earth's subsurface and is considered a time-domain vocoder that modifies the signal both on the frequency and time domain using a short-time Fourier transform with a window function to frame the inner transform. Autotune is a full pitch correction system including the three steps described above: pitch detection, pitch correction and pitch modification. We present in this section the pitch correction method. For this purpose, the sung notes are shifted to the closest note in a predefined scale, and the transition is carried out over a duration equal to a transition time (named "retune speed" on ATA). Autotune also includes the flextune parameter, which acts as a threshold for the correction and represents the size of the neighborhood of a note in which a pitch correction can be triggered.

Due to lack of detail in the patent [5], the ATA algorithm can only be reproduced for an extreme correction case, meaning a value 0 on the Decay parameter in the patent of ATA (internal parameter of the code, and related to the retune speed parameter). This case corresponds to force the input trajectory to match integer MIDI values, i.e., the target notes. For the non-zero Decay cases we cannot reproduce the algorithm as the patent doesn't describe exactly the configuration of the smoothing step. To treat cases with non-zero transition time we will apply the ATA VST on audio signals and then extract the retuned f_0 to study correction actually carried on.

[2] `https://www.hakenaudio.com/continuum-fingerboard` last checked: 25 may 2023

[3] `https://www.roli.com` last checked: 25 may 2023

[4] `https://www.apple.com/mac/garageband/` last checked: 25 may 2023

[5] `http://www.lam.jussieu.fr/cantordigitalis/` last checked: 25 may 2023

2.2. Dynamic Pitch Warping

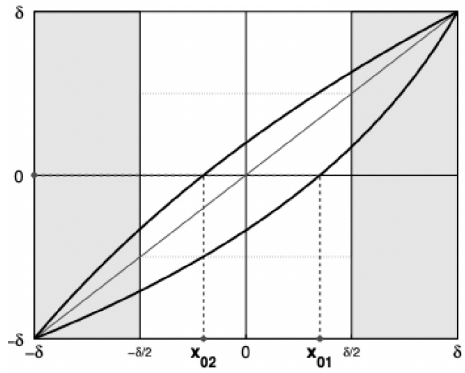

Figure 1: *The arc of curvature for the dynamic pitch correction method, took from [11]*

DPW is a real-time pitch correction method developed by Perrotin and d'Alessandro for Cantor Digitalis. Although it was originally designed to correct a driven by stylus pitch on a graphic tablet, we aim to use DPW for vocal correction. DPW relies on pitch velocity (speed) to trigger an adaptive correction that modifies the input f_0 curve gradually, enabling the output f_0 to converge to the nearest semitone on the MIDI scale. When pitch velocity falls below a threshold, DPW smoothly shifts subsequent f_0 values to converge to a tuned semitone, while preserving some expressive motion of the original f_0 value. The adaptive function remains static when the pitch velocity condition is not met, allowing intended notes to be corrected while retaining expressiveness and preserving all dynamics for non-corrected notes. To review the method, we first analyze the isolated adaptive function, as seen in Figure 1 that maps the input f_0 (x axis) to the output f_0 (y axis). On both axes, zero represents the closest target (ideal) pitch, and $-\delta$ and $+\delta$ correspond to the previous and next notes on the discrete target pitch scale, respectively. While it works on any arbitrary scale, $\delta = 1$ when working with semitones. For input pitch x_{01}, the closest target note is zero. Therefore, at the time the correction is triggered, the corresponding adaptive function that is initially diagonal will smoothly shift towards the lowest arc-shaped curve, to eventually map the input f_0 to the pitch target (zero) as output f_0. The adaptive function then becomes static until it is newly triggered. To avoid introducing a constant shift on the full pitch range, the adaptive function is arc-shaped so that if the input moves from the x_{01} value to the neighbour notes on the pitch scale ($-\delta$ or $+\delta$), the output f_0 will continuously reach $-\delta$ or $+\delta$. If those boundaries are reached, the adaptive function goes back to a linear mapping between input and output, until it is triggered again for a new input.

The adaptive function is derived from the analytic definition of an arc. To ease formulation, the inverse function is first defined:

$$x(y) = Ae^{\gamma(y+B)} + C \tag{1}$$

where the parameters A, B, C and γ can be calculated from the boundary conditions, i.e., the arc must satisfy $x(\pm\delta) = \pm\delta$. If we use this condition, we can write A and C in terms of γ, δ, and B

as follows:

$$C = -\delta\left(1 + \frac{2}{e^{2\gamma\delta} - 1}\right), A = 2\delta\frac{e^{\gamma(\delta - B)}}{e^{2\gamma\delta} - 1} \tag{2}$$

Replacing these values in the original equation 1, we find that the dependency on B disappears. Furthermore, the function is not defined for $\gamma = 0$, but it corresponds to an absence of correction, i.e., the mapping is linear. So the function of the arc curvature can be written as:

$$x(y) = \begin{cases} \delta\left[2\frac{e^{\gamma(\delta+y)}-1}{e^{2\gamma\delta}-1} - 1\right] & if \quad \gamma \neq 0 \\ y & if \quad \gamma = 0 \end{cases} \tag{3}$$

The adaptive warping function is defined as the inverse of 3:

$$y(x) = \begin{cases} 1/\gamma\left[\log\left[(e^{2\gamma\delta} - 1)(\frac{x}{\delta} + 1)\frac{1}{2} + 1\right]\right] - \delta & if \quad \gamma \neq 0 \\ x & if \quad \gamma = 0 \end{cases} \tag{4}$$

Where γ is the factor of correction, y is the output pitch after the correction, and x is the input pitch. When the correction is triggered (at that moment $x = x_o$) , the value of $\gamma = \gamma_0$ can be calculated from the input value x_0 to ensure that $y(x_0) = 0$ following the equation:

$$\gamma_0 = \frac{1}{\delta}\log\left(\frac{\delta - x_0}{\delta + x_0}\right) \tag{5}$$

The DPW has two stages that can be seen on Figure 2. One is the triggering part and the other is the warping stage. For the correction to be triggered, the pitch trajectory has to be stable enough, i.e., it has to stay within an interval of detection (ID) during a critical time (T_c) [11]. If these conditions are met, we can calculate the curvature γ_0 given the input pitch at triggering time (x_0 in the definition, f_0 for us). To ensure a smooth transition, γ is linearly interpolated from 0 (linear mapping) to γ_0. This transition spans a time interval denominated transition time (T_t). When the transition is completed, the input pitch has converged to the closest integer notes on the midi scale. This transition is carried out similarly to the static case of ATA, not over the frequency but over the γ value, then f_0 (input) is warped with the adaptive function.

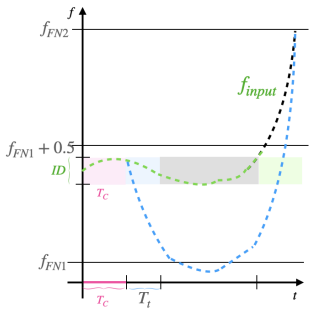

Figure 2: *Illustration of the dynamics of DPW. The input f_0 (green curve) is stable in a detection interval ID during the critical time T_c (pink region). The correction is triggered during the transition time T_t (blue region). The input f_0 can vary continuously during the transition time, until it reaches the next semitone on the pitch scale (integer, black).*

3. CASE STUDIES OF PITCH CORRECTION

In this section, we compare both ATA and DPW methods. Firstly, we want to show the difference between the methods through a simple case. We take as example a constant flat note (C♯) with a pitch shift of 0.15 semitone (ST), and we use both methods to correct it. In Figure 3, we see a DPW correction (blue) triggered with the following parameters: $ID = 0.1$ ST, $T_c = 0.5$ s, and $T_t = 0.5$ s. The ATA correction (red) has a retune speed equal to T_t. We have chosen a non-zero value for T_c to show the inclusion of the new parameter. The critical time is the main difference between both methods. While it introduces a triggering delay in DPW, we find similar results for both corrections once after that trigger.

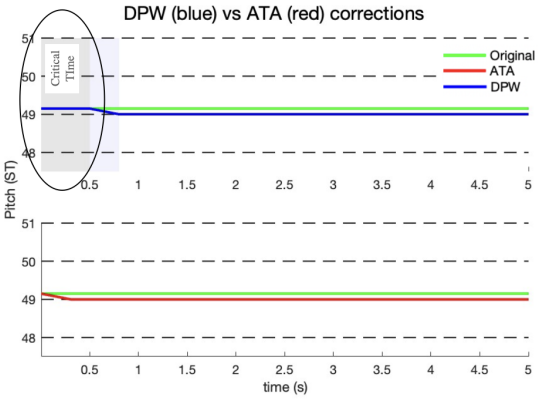

Figure 3: *DPW correction (blue curve) and ATA correction (red curve) of a constant input pitch (green curve).*

3.1. Extreme correction with zero transition time parameter

We denominate extreme correction to a full discretization of the input pitch trajectory. To check the extreme correction, we chose two typical examples: the first one is a glissando, and the second is a melody taken from [15]. After trying some configurations, we have found a combination of parameters that provides similar results with both methods. For DPW, we have chosen the parameters $T_c = 0$ s, $ID = 0.01$ ST, and $T = 0.001$ s (the minimal value). For ATA we choose just the zero retune speed the minimal value), that as described in the patent generates discrete notes (integers on ST scale). We can see the results in Figure 4 and 5. The f_o-signal treated with DPW is in blue, and the one treated with ATA is in red and the original is in green.

3.2. Expressive Correction with ATA

One of the most important artifacts of vocal expression is vibrato. Expressive Correction is the term we use here to refer to a fast transition within pitch correction that correspond to oscillatory ornaments, particularly vibrato. As we will see, a vibrato with a small amplitude can be shifted around the target pitch with DPW, while it is not well centered under the ATA correction. The expressive correction requires a non-zero transition time parameter. We don't have access to the full implementation of the transition time parameter in ATA (also referred to as the Decay parameter in the

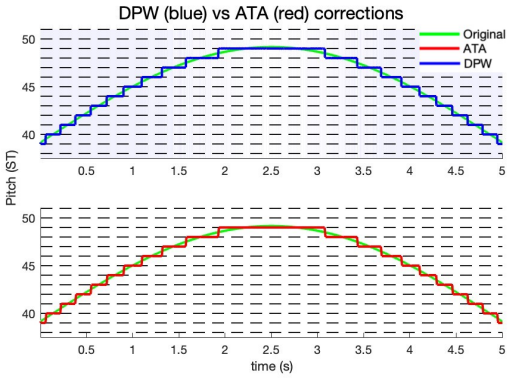

Figure 4: *Extreme correction for a glissando*

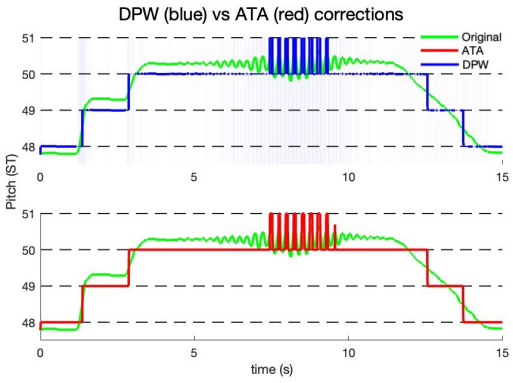

Figure 5: *Extreme correction for an expressive melody*

patent), so we cannot reproduce the exact expected pitch correction of ATA. Therefore, the most effective way to fully understand how the ATA pitch correction algorithm works is to utilize the ATA VST plugin to retune voice samples and extract the corrected f0 from the resulting audio using Praat software [6]. This curves are compared with the DPW correction. To generate the input audio samples, we use Cantor Digitalis (CaD), which is a continuous pitch input synthesizer. CaD takes the trajectory of a wacom stylus, the it generates f_0 and synthesizes a vocal sound. We modified its code to have purposely not-intonated sounds related to the original stylus trajectory. Audio examples can be found in soundcloud [7]. The non-intonated audio samples can be corrected with ATA vist but also with an off-line DPW implementation that we explain later in section 4. Now we proceed to the comparison of both both pitch correction methods.

The simplest case of correction is a shifted note with vibrato. Small vibratos can be effectively corrected with ATA using a retune speed of 50ms. For sustained notes, ATA performs very well and there is no difference with DPW, so we do not present this example here. The difference arises when we have a signal that contains flat notes, free paths, and vibratos. Therefore, it is important to demonstrate how a correction can be performed with ATA using different values of the retune speed parameter, refer to Figure 6. Going up to down we use a retune speed parameter from 0, 15, 50, 100 and 200 ms. The correction is effective at 50ms for the vibrato, but the pitch trajectory after the 12-second mark becomes lost and flattened. Only with a retune speed parameter set to 200ms is it possible to preserve some of the pitch trajectory, but at that configuration, the vibrato is not corrected.

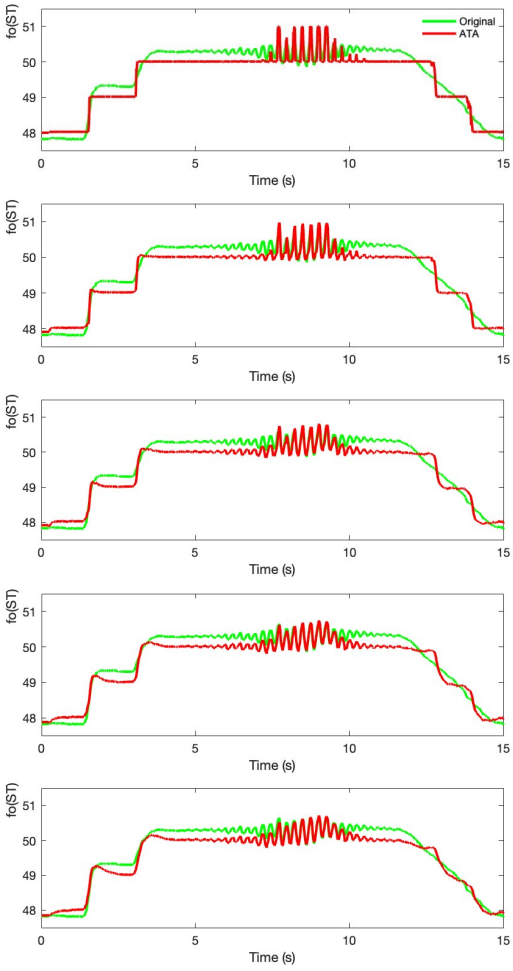

Figure 6: *Correction using different values of retune speed on ATA, RS= 0, 15, 50, 100, 200 ms (up to down)*

Now we will examine the functionality of the ATA flextune parameter. For a more general case, let's now observe what happens when we vary the retune speed while maintaining a specific value for flextune. We have done a configuration with zero retune speed and two values of flextune: zero (red) and 40 cents (violet), figure 7. As we can see, the flextune parameter allows for movement within the range defined by the flextune value after the correction, resulting in the production of smaller ornaments at the output.

In the following example, we will use a non zero value of retune speed, 15ms, and flextune values of zero (red) and 30 cents (violet), as shown in figure 8. As we can see, like the previous example, some ornaments smaller than the flextune value can be preserved at the output.

Now, we present a study with a transition time of 50ms and flextune values of 30 and 60 cents. As we can see in figure 9, a

[6] https://www.fon.hum.uva.nl/praat/

[7] https://on.soundcloud.com/b5NDp last checked: 25 may 2023

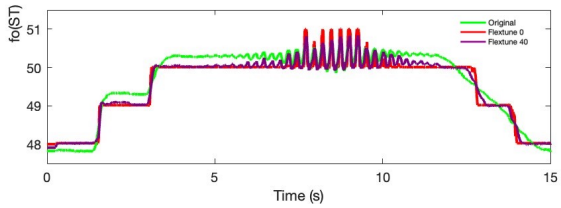

Figure 7: *Correction with ATA, at zero retune speed and flextune: 0 (red) and 40 cents (violet))*

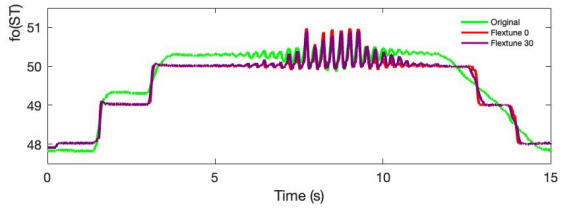

Figure 8: *Correction with ATA, at retune speed equal to 15 ms and flextune: 0 (red) and 30 cents (violet))*

larger value for flextune results in a lack of reactivity. This means the vibrato is not corrected but the path after time equal to 12 s is better preserved than in the other cases. In other words when the notes are well corrected, the general path may be more or less lost depending on the parameters.

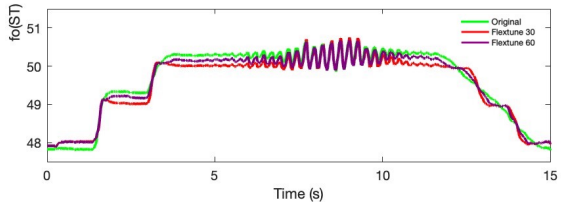

Figure 9: *Correction with ATA, at retune speed equal to 50 ms and flextune: 30 (red) and 60 cents (violet))*

Finally, we show what happens when varying the retune speed for the same flextune parameter. We have chosen a moderate flextune value of 40 cents, while the retune speed varies as follows: 50ms, 100ms, and 200ms. The result can be seen in figure 10. There is always a trade-off between preservation of the main path (free path) and vibrato correction. This means that ATA better preserves the vibrato, but regions such as the one after 12 seconds become staircase-like, resulting in the loss of the original pitch trajectory. In the other hand, parameter values that preserve the shape in that zone, does not correct the vibrato. As we can see in figure 10, the vibrato is not corrected for a retune speed higher than 50ms. On the other hand, when we use flextune at 40 cents and keep zero retune speed (figure 8) the vibrato is corrected but the path after time 12 s is flattened.

3.3. Expressive Correction with DPW

We will show several examples variations of the DPW parameter: critical time and transition time. For the first example, we do choose 100 ms as T_c, then we vary T_t, giving the results in figure

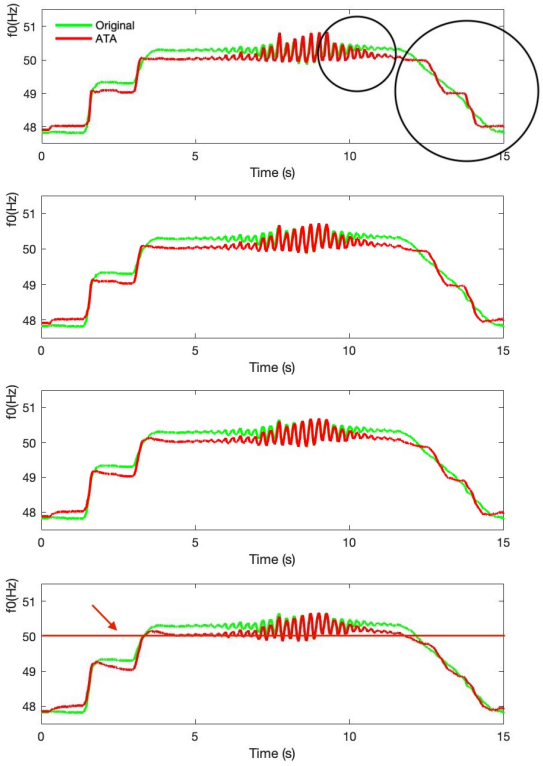

Figure 10: *Correction using different values of retune speed on ATA, RS= 0, 50, 100, 200 ms (up to down) for the same flextune value (40 cents)*

11. As we can see, varying T_t parameter allow us to "smooth" the pitch correction.

Also we have done a correction using a larger critical time equal to 250 ms (optimal according to [11]). It gives the results in figure 12. As we see, the critical time acts as trigger of the correction and the transition time acts as a smoother. The critical time (as parameter) adds an ornament at the beginning of each note step in the staircase region and the transition time modifies the shape of the ornament.

Finally, we have performed a correction using the same transition time (50 ms) while varying the critical time parameter (100 ms, 150 ms, 250 ms). It gives the results in figure 13. As we can see the critical time parameter acts like a trigger for the pitch correction algorithm and the transition time acts as the smoother.

Now we can compare the best configuration for each method. In the case of ATA, it is not possible to achieve good vibrato correction and good preservation of the free path simultaneously. Therefore, we preferred a moderate configuration that performs reasonably well for both purposes. A suitable ATA configuration is a retune speed of 100 ms and flextune of 40 cents (figure 13). For DPW the most suitable correction is done by choosing the critical time as 200 ms (DPW) and then we can choose for example a transition time equal to 50 ms (figure 10) . For simplicity we have put these two cases in the figure 14). This shows that DPW performs a better correction: Firstly the vibrato is well centered in DPW correction while not in ATA; and secondly the DPW preserve better the f_o-path after time 12 s, while ATA flatten it. In contrast, ATA seems visually better in the segment before 5 s while DPW

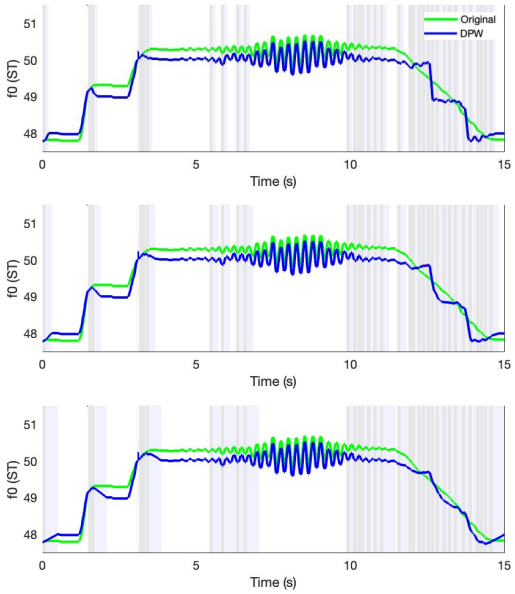

Figure 11: *Correction using different values of transition time in DPW (from up to down: 100,200,400 ms), for the same critical time (100 ms)*

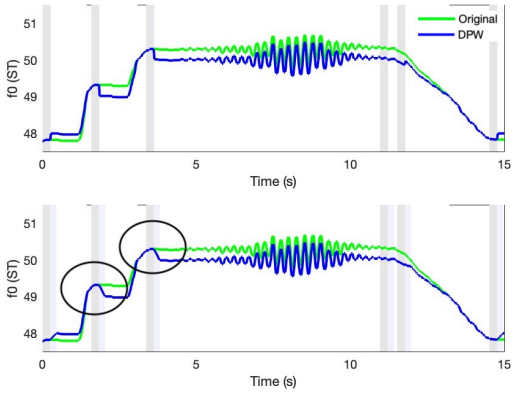

Figure 12: *Correction using different values of T_t in DPW (from up to down: 25,200 ms), for the same T_c (250 ms)*

present an more visible expressive ornament. In the subsequent subsection, we will showcase the measurements that are directly linked to the aforementioned observations, as we will see DPW is closer to the original f_o curve for all the regions.

3.4. Comparison through MSE and MAE

The difference between two curves can be measured in various ways, here we presented two. Firstly, the Mean Squared Error (MSE) that measures the sensitivity to quadratic errors; it is calculated through the difference of squares, which gives larger errors a greater impact on the overall result. MSE also provides a measure of variance between the curves. Secondly, the Mean Absolute Error (MAE) that provides a measure of the average difference in magnitude between the curves, unlike MSE, MAE does not amplify larger errors. We use the follwoing equations:

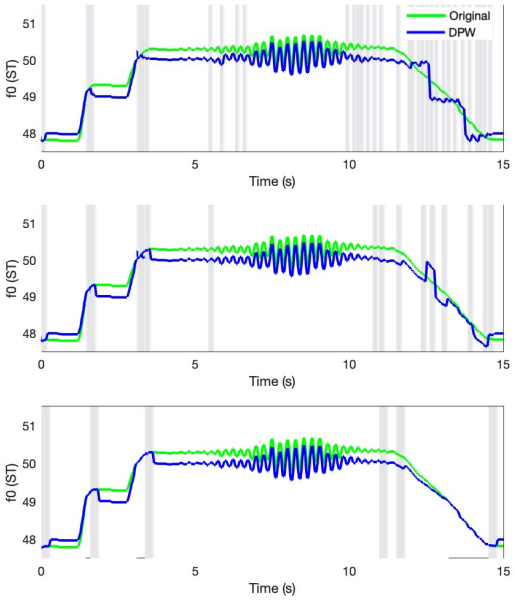

Figure 13: *Correction using different values of T_c in DPW (from up to down: 100,150,250 ms), for the same T_t (50 ms)*

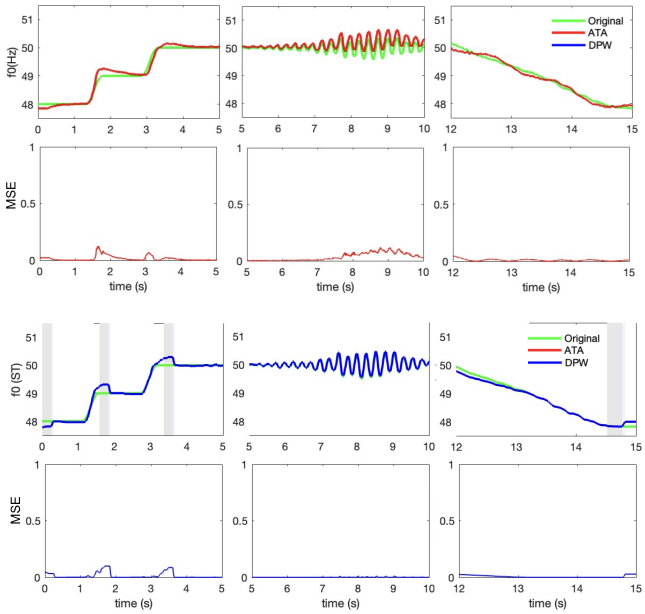

Figure 14: *Correction for the same T_t (50 ms) using flextune at 40 cents for ATA and T_c at 200 ms for DPW and the corresponding MSE.*

$$\text{Mean of MSE} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{n_j}\sum_{i=1}^{n_j}(y_{ij}-\hat{y}_{ij})^2\right) \tag{6}$$

$$\text{Mean of MAE} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{n_j}\sum_{i=1}^{n_j}|y_{ij}-\hat{y}_{ij}|\right) \tag{7}$$

Where N represents the number of samples, n_j is equal to 1, cause there always a comparison of one curve with the reference, j represents the curve to compare (ATA or DPW), y_{ij} are the values of the original curve j, and $\hat{y}_{ij}$ are the values of the comparison curve j.

Our example is helpful to highlight three types of pitch modification. The first part in $0 < t < 5$ where signal is like a staircase between the notes 48,49 and 50. The second part in $5 < t <$ 0represents the correction of a poorly intonated frequency modulation, similar to the human vibrato. And the third part is a soft path of a f_o trajectory that should not be corrected, the free path represents the case where the singer do not have the intention to play any specific note. Each part must be compared to the desirable pitch curve, which is different for each region. For example for the staircase part, the desired signal is a staircase. For the vibratory part the ideal pitch would be the same vibration but well centered. And for the third part, the original signal would be the ideal pitch, rather than a correction we want to preserve it. These assumptions are illustrated on figure 6, the calculation of MSE is done point by point. The mean over each region is reported in Table 1. As it is shown and mentioned before, DPW perform better correction of vibratos while preserving the free path of the note, and ATA is better for the staircase part while losing more of the vibrato and free path parts.

Table 1: *MSE and MAE between input and corrected f_0 for the different regions*

	MSE		MAE	
Region	DPW	ATA	DPW	ATA
1	0.0146	0.0146	0.0747	0.0914
2	0.0415	0.0642	0.1304	0.2103
3	0.0539	0.0280	0.2015	0.1463

Please note that all the comparison are focused on the pitch correction curves. For DPW we use the pitch correction method that is different than the full algorithm audio. The implementation of the vocoder, described in section 4, is a complex process and the vocoder we have use in making the audio DPW tracks is not as advanced as the vocoder of ATA. As a result, some imprecision may be present in the generated f_0 paths for the DPW audio examples. Despite these limitations, it is worth highlighting the valuable insights gained from this comparison, which shed light on the respective strengths and weaknesses of each method.

4. IMPLEMENTATION OF AN OFF-LINE AUDIO PITCH CORRECTION

This section talks about the off-line implementation of DPW. DPW works in an analogous way to Cantor Digitalis. However, instead of an incoming f_0 given by a table, we use an f_0 value obtained from a pitch tracker on a pre-recorded vocal audio track. The general structure for a autotune system is conformed by: a pitch tracker, a pitch correction algorithm, and a pitch warping algorithm (vocoder). DPW can follow a similar approach using a pitch tracker to acquire f_0.

4.1. Development of the off-line retuner

We developed a methodology for off-line vocal retuning using the DPW method; this process requires obtaining F0 data and a

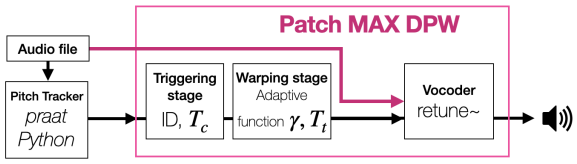

Figure 15: *Configuration of the offline retuner*

transparent vocoder as shown in 15. For pitch tracking, we utilized Praat[8] (software to analyze audio prosody), the *To PitchTier* method allows us to obtain F0 curves for the original audios within a Praat file sampled at Praat time intervals. We did a Python code (with package *wave*) to extract the file's relevant data and to create arrays for time and f_0 information; the arrays were re-sampled at the original audio files sampling rate. The *pathlib* package was employed to process multiple sound library files simultaneously, resulting in a library of the original audios and the f_0 files. The Max/MSP environment was used to process the f_0 information (on Semi tones and Hz) and write retuned audio files using the different vocoders (retune∼, freqshift∼, pitchshift∼, supervp∼, etc). Our goal was to identify the most transparent vocoder that generated a voice signal closest to the input F0, using the original f_0 data, the *retune∼* object was selected as the most transparent modification for the entire library; this ensured that the vocoder avoided introducing sound artifacts that could affect the perception of quality and retuning. However, the overall quality of the presented vocoder, retune∼, is not as precise and good as the ATA vocoder. Therefore, the resulting audio tracks using retune∼ may not be as good as those using the ATA vocoder. Therefore, we dispose of an alternative option, with an wrapper of the *World* [16]vocoder, provided by the research engineers of Lutherie-Acoustique-Musique Group, the audio obtained with *World* is done through a non-real-time transposition through python. The resulting audio has a better quality than the MAX implementation. The sound library for DPW correction using both vocoders can listen on the soundcloud playlist noted in section 3.2.

5. CONCLUSIONS

Through our research, we studied DPW algorithm for audio pitch correction. It is possible to control and trigger a pitch correction thanks to three degrees of freedom that preserves low-amplitude vibratos and ornaments in the neighborhood of the target note. We have also shown how the pitch correction methods are composed of two stages (triggering and warping), and how the modification of the control parameters can lead to equivalent configurations for different systems. We have identified a scenario where ATA and DPW exhibit similarity: extreme correction. Moreover, we have identified three types of correction: staircases, vibratos, and free paths, and have illustrated that DPW performs better for vibratos and free paths, while also being adequate for staircase correction. DPW also exhibits less trade-off between its parameters compared to ATA.

In addition, we have developed an audio application that includes the DPW method. Compared to ATA, its control parameters allow for a smooth pitch trajectory transition towards the nearest

[8] https://www.fon.hum.uva.nl/praat/

notes on a defined scale, minimizing distortion of melodic ornaments between the notes. However, it is important to note that the vocoder used in our application (retune∼) may not provide the same level of quality, precision and accuracy as the ATA vocoder.

We plan to undertake a comprehensive perceptual evaluation of the two systems in a formal setting. This evaluation aims to assess the perceptual salience of the pitch effects introduced by the DPW method, as well as their potential musical relevance.

6. ACKNOWLEDGMENTS

This research was funded through ANR National Research Agency projects: Analysis and Transformation of Singing Style (ANR-19-CE38-0001) and Gepeto: GEsture and PEdagogy of inTOnation (ANR-19-CE28-0018)

7. REFERENCES

[1] C. Vincent, *La voix chanteé*, chapter De l'antipop à l'Autotune, pp. 123–142, N. Henrich & De Boeck Solal, 2013.

[2] P. Boulez and A. Gerzso, "Computers in music," *Scientific Amer.*, vol. 258, no. 4, pp. 44–51, April 1998.

[3] A. Wilson and B. Fazenda, "Perception and Evaluation of Audio Quality in Music Production," in *Proc. of the 16th Int. Conf. on Digit. Audio Effects*, Maynooth, Ireland, September 2013, pp. 68–77.

[4] J.M Chowning, "Digital sound synthesis, acoustics and perception: A rich intersection," in *Proc. of the Int. Conf. on Digit. Audio Effects*, Verona, Italy, December 2000, pp. 1–6.

[5] H. Hildebrand, "Pitch detection and intonation correction apparatus and method," Auburn Audio technologies, Auburn, AL, USA Patent US5973252A, G10H-007/00, Oct. 14, 1992, pp 10-18.

[6] L. Haken, "Position correction for an electronic musical instrument," Champaign, IL, US Patent 76191562009, Int GIOH 1/22, Apr. 19, 2007, pp 6–12.

[7] L. Haken, E. Tellman, and P.Wolfe, "An indiscrete music keyboard," *Comput. Music J.*, vol. 22, no. 1, pp. 30–48, Spring 1992.

[8] L. Feugère, C. d'Alessandro, B. Doval, and O. Perrotin, "Cantor digitalis: chironomic parametric synthesis of singing," *EURASIP J. on Audio, Speech, and Music Process.*, vol. 2, pp. 98 1–19, December 2017.

[9] Sebastian Rosenzweig, Simon Schwär, Jonathan Driedger, Meinard MüllerA. Wilson, and B. Fazenda, "Adaptive pitch-shifting with applications to intonation adjustment in a cappella recordings," in *Proc. of the 24th Int. Conf. on Digit. Audio Effects*, Vienne, Austria, September 2021, pp. 121–128.

[10] N. Orio, N. Schnell, and M.M. Wanderley, "Input devices for musical expression: Borrowing tools from hci," in *Proc. of the Int. Conf. on New Interfaces for Musical Expression*, Seattle, USA, April 2001, pp. 62–76.

[11] O. Perrotin and C. D'alessandro, "Target acquisition vs. expressive motion: Dynamic pitch warping for intonation correction," *ACM Transactions on Computer-Human Interaction*, vol. 23, no. 3, pp. 17 1–21, June 2016.

[12] S. Rossignol, P. Depalle, J. Soumagne, X. Rodet, and J.-L. Collette, "Vibrato: Detection, estimation, extraction, modification," in *Proc. of the Int. Conf. on Digit. Audio Effects*, Verona, Italy, December 1999, pp. 1–6.

[13] R. Lamb and A.N. Robertson, "Seaboard: a new piano keyboard-related interface combining discrete and continuous control," in *Proc. of the Int. Conf. on Digit. Audio Effects*, Oslo, Norway, June 2011, pp. 503–506.

[14] A.P. McPherson, A. Gierakowski, and A.M. Stark, "The space between the notes: Adding expressive pitch control to the piano keyboard," in *Proc. of the SIGCHI Conf. on Human Factors in Comput. Syst.*, New York, NY, USA, June 2013, p. 2195–2204.

[15] O. Perrotin and C. d'Alessandro, "Quel ajustement de hauteur mélodique pour les instruments de musique numériques?," in *Journées d'Informatique Musicale (JIM 2015)*, Montréal, Canada, May 2015, pp. 186–189.

[16] K. Ozawa M. Morise, F. Yokomori, "World: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 7, pp. 1877–1884, July 2016.

[17] V. Verfaille, U. Zölzer, and D. Arfib, "Adaptive digital audio effects (a-dafx): a new class of sound transformations," *IEEE Transactions on Speech and Audio Processing 14 (5)*, pp. 1817– 1831, 2006.

[18] N. Zacharov, *Sensory Evaluation of Sound*, CRC Press, Boca Raton, FL, USA, first edition, 2019, pp. 60–99, 107-134.

[19] J. Chowning, *Music, Cognition, and Computerized Sound*, MIT Press, Cambridge, Massachusetts, London, England, 1999, pp. 261–276.

[20] S. Bernsee and D. Gökdag, "Methods for extending frequency transforms to resolve features in the spatio-temporal domain," Zynaptiq GmbH , Hannover, Germany, USA Patent 11079418 B2, Aug. 23 , 2018, pp 26–41.

[21] Cycling '74. CA, USA, "Max online documentation," Accessed: 19.09.2022. [Online]. Available: https://docs.cycling74.com/.

[22] Ircam. Paris, France, "Supervp for max max reference pages 11/2012," [Online]. Available: https://forum.ircam.fr/media/uploads/software/SuperVP%20for%20Max/supervp-for-max.pdf, published Nov-2012, pp 1–14.