Low-Latency Audio Pitch Tracking: a Multi-Modal Sensor-Assisted Approach

Laurel S. Pardue , Dongjuan Nian, Chris Harte, Andrew P. McPherson
Queen Mary University of London
Mile End Road, E1 4NS
London, UK
laurel.pardue@eecs.qmul.ac.uk, ee08b060@gmail.com, christopher.harte@eecs.qmul.ac.uk,
andrewm@eecs.qmul.ac.uk

ABSTRACT

This paper presents a multi-modal approach to musical instrument pitch tracking combining audio and position sensor data. Finger location on a violin fingerboard is measured using resistive sensors, allowing rapid detection of approximate pitch. The initial pitch estimate is then used to restrict the search space of an audio pitch tracking algorithm. Most audio pitch tracking algorithms face a fundamental tradeoff between accuracy and latency, with longer analysis windows producing better pitch estimates at the cost of noticeable lag in a live performance environment. Conversely, sensor-only strategies struggle to achieve the fine pitch accuracy a human listener would expect. While this paper is violin centric, it demonstrates a more general concept for augmented instruments that by combining the two differing approaches, high accuracy and low latency pitch can be simultaneiously achieved.

Keywords

pitch tracking, position sensing, audio analysis, multi-modal, violin, fingerboard

1. INTRODUCTION

Monophonic pitch tracking is sometimes considered a solved problem in audio analysis, but existing approaches do not always meet the stringent accuracy and timing demands of live instrumental performance. Musicians can detect reaction latencies of 20-30ms in musical instruments [17, 1], and latency under 10ms is an accepted target for interactive audio systems [9]. Expert live performance also demands an extremely low detection error rate.

Most audio pitch tracking algorithms are based on windowing the signal, whether the analysis is performed by FFT or directly in the time domain (as in autocorrelation-based approaches). But a commonly-used window size of 4096 samples at 44.1kHz requires 93ms to fill; even a 512-sample window, short enough to degrade the performance of most algorithms, lasts over 11ms. Where multiple consecutive frames are compared to improve robustness, this will multiply the total latency. In many algorithms, minimum window size is inversely related to the lowest frequency to be detected, reflecting inherent tradeoffs of time and frequency resolution in short-time spectral analysis.

NIME'14, June 30 – July 03, 2014, Goldsmiths, University of London, UK.

Sensors measuring the performer's actions can instead be used to detect pitch [12], but the mechanics of most instruments makes these solutions incomplete. On string instruments, finger position and string tuning (which can vary during a performance) must be known; on winds, both fingering and embouchure affect the pitch. Thus, regardless of the accuracy of the sensors, audible pitch errors are likely.

We propose a hybrid approach which uses sensors to produce an initial pitch estimate. The estimate restricts the search space of a low-latency audio analysis algorithm whose accuracy would otherwise be unacceptably low. Because audio and sensors tend to produce different types of errors (harmonics being common in audio, slight mistuning with sensors), the combination can be both fast and accurate. This paper demonstrates the approach using position sensing on a violin fingerboard, with sensor-assisted versions of two commonly-used audio analysis algorithms.

2. PRIOR WORK

2.1 Augmented Violin

Augmented stringed instruments have a long history from Max Mathews' early electric violin [18] in the 1970s, to Young's "Hyper-Cello" [22] in the 1990s, to Overholt's "Overtone Violin" [20] and "Overtone Fiddle" [21] and Jensenius's violin augmented with video motion tracking [13]. Poepel [24] provides a review of work up to 2006.

While many augmented instruments aim to enable completely new performance techniques and gestures, this paper focuses on measurement of traditional technique. Traditional technique analysis focuses in two areas: left-hand finger placement (connected to pitch) and right-hand bow tracking. Bow tracking has been well-explored [2, 26, 16, 27, 23], though it is by no means a solved problem.

Fingerboard tracking is in certain ways an easier problem as physical contact location between the finger and the instrument can be measured. Freed has examined use of commercial position and force sensors [8, 10] for finger position tracking in a traditional context, while Grosshauser augmented a traditional violin with both capactive and pressure based position sensors [11, 12]. Ajay Kapur's "E-Sitar" [14] is a fretted instrument which uses a different fingerboard measurment method, linking the metalic frets with resistors and measuring string-fret contact by electrifying the strings. Although this is not a technique available to fretless instruments the E-Sitar is notable for using audio analysis to refine estimated pitch, as sitar pitch is determined by both the frets and the bending of the strings.

2.2 Monophonic Pitch Tracking

In-depth reviews of monophonic pitch tracking are presented in [3] and [6]. There are three primary means of pitch detection: spectral methods, temporal methods, and combina-

tions of the two. Spectrally-based methods work by short-time Fourier analysis of the audio signal, looking for a dominant frequency and its harmonics. The dominant frequency can be identified in several ways, including autocorrelation within the spectral domain [15], cepstral analysis [19], and constant-Q methods [4].

Temporal methods are typically based on autocorrelation. By cross-correlating a windowed signal with itself, peaks occur in the autocorrelation function at harmonics of the fundamental frequency within that window [25]. Ideally, the largest peak is the target frequency; however, resonances and variations in the signal result in non-trivial error rates. Since basic autocorrelation weights results equally no matter the lag, it often leads to the frequency estimate being too low. An alternative is to reduce the weight at greater lags $\tau = F_{SR}/F_P$, by effectively reducing the correlation window using the equation:

$$r_t(\tau) = \sum_{j=t+1}^{t+W-\tau} x_j x_{j+\tau} \quad (1)$$

While Equation 1 reduces the likelihood of underestimating the fundamental frequency, it does this at the cost of increasing the likelihood of overestimating the fundamental as shorter periods are weighted more heavily. This paper uses the intentionally "biased" equation, Equation 1, as one of the two test algorithms for pitch estimation.

$$d_t(\tau) = \sum_{j=1}^{W} (x_j - x_{j+\tau}^2) \quad (2)$$

A third variation of temporal-based autocorrelation is to use a normalized difference function. With Kawahara and Cheveigné's Yin [7], using a difference function, Equation 2, reduces the influence of amplitude changes on the autocorrelation function, Equation 1, to achieve a dramatic decrease in errors. Yin has been proven highly robust and reliable and has become the dominant means for pitch detection. It is also the second test algorithm used.

3. LOW-LATENCY PITCH TRACKING

3.1 Capturing Finger Placement

We propose using high-speed position sensors to assist audio pitch-tracking. Our fingerboard hardware is designed to improve upon existing methodologies by being non-intrusive, non-destructive, and retaining seamless fingerboard feel and appearance.

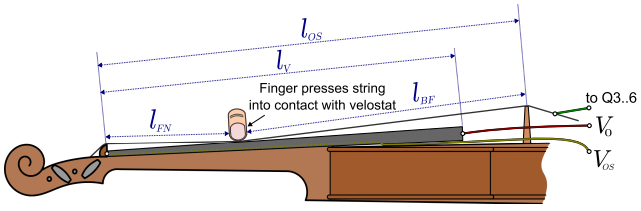

Figure 1: Fingerboard sensor configuration. The fingerboard is covered with a layer of resistive velostat. Contact between a string and the fingerboard produces an electrical connection.

A typical means for making a linear potentiometer strip is to place a conductive layer over a resistive strip with an air-gap separating the two. When pressed, the conductive material contacts the resistive strips at the point of pressure, forming a voltage divider. Commercial linear potentiometers have a built-in air gap which requires minimal but still noticeable height, and with that, noticeable edges. The height and edges alter the feel of the instrument, making the sensor non-ideal for use on traditional instruments.

We instead apply a bare resistive strip directly to the fingerboard, making use of the conductivity of metal strings and the normal air gap between the string and the fingerboard (Figure 1). A single thin layer of velostat (carbon-infused polymer) is glued to the fingerboard with connections hidden under the bridge end of the fingerboard. Along with its resistive properties, the velostat has a smooth finish similar to the fingerboard. Each violin string is electrified by running the ball end through an electrically connected solder tab. We removed the fine tuner from the violin E-string as it was made of non-conductive plastic. We successfully used both generic inexpensive strings and high-end professional strings. With the exception of added wires from the fingerboard, the feel of the violin is only minimally changed.

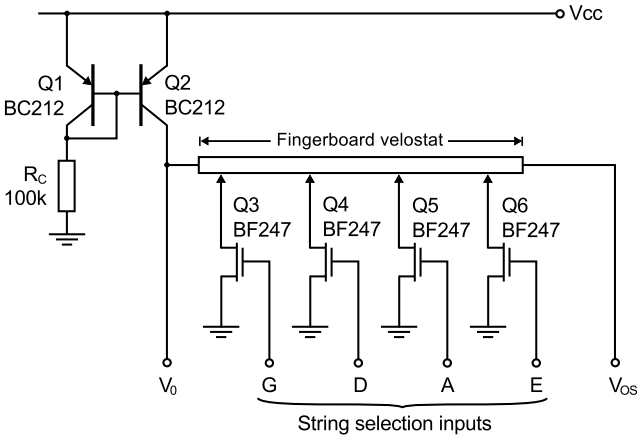

Figure 2: Circuit consists of a current mirror on one end of the velostat, and a MOSFET on each string which is used to switch between strings.

As shown in Figure 2, a current source (here, a current mirror based on a BC212 PNP transistor) provides a constant current to the variable fingerboard-string resistance, producing an output voltage which is linearly related to resistance: $V_o = IR_C$. The resistor R_C should be chosen to be greater than the maximum resistance of the velostat strip in order to detect across the full length of the strip. Our strips had a typical resistance of around $80k\Omega$ so we chose $R_C = 100k\Omega$.

Expected pitch based on finger position is determined by considering the ratio of change between the open string length (l_{OS}) land the new string length measured from the bridge(l_{BF}):

$$f_{\text{played}} = f_{\text{string}} \frac{l_{OS}}{l_{BF}} \quad (3)$$

To determine the new string length we subtract the distance of the finger to the nut (l_{FN}) from the overall string length. We find the finger position from the measured voltage output. As V_o must be normalized against the full velostat length (l_V) along the fingerboard, with V_{MS} as the maximum measured voltage, the equation for determining finger position from the nut becomes:

$$l_{BF} = l_{OS} - l_{FN} \quad (4)$$

$$l_{FN} = l_V(1 - \frac{V_o}{V_{MS}}) \quad (5)$$

Further this gives an overall function for voltage to pitch of:

$$f_{\text{played}} = \frac{f_{\text{string}} \times l_{OS}}{l_{OS} - l_V(1 - \frac{V_o}{V_{MS}})}) \quad (6)$$

An Atmel AVR32UC3C board is used to sample the voltage at each string and calculate a normalized position for the finger closest to the bridge. A sample rate of 300kHz is possible with each sample being the average of 4 ADC measurements. Position accuracy using non-conductive material to press the string was found to be roughly 3mm. Due to the high resistance of the velostat, the conductive properties of the human hand introduces significant noise, halving accuracy with single string contact to around 6mm. Using a lower-resistance material would reduce the noise from human interaction substantially, but as we have quite high noise tolerance for left hand input, the velostat is acceptable. For simplicity, in this paper, we assume single string contact only since we are restricting ourselves to monophonic tracking.

Obviously, with accuracy only within 6mm, finger position alone can not be used to determine the exact pitch. This accuracy is sufficient to estimate the target pitch within a whole tone. For the lower octave on each string, the distance between semitones is between 10-17mm. Even if it were perfectly accurate in determining finger position, there would still be issues of string tuning and slight differences between the point of pressure and the effective vibrating end of the string. In order to find the actual pitch being performed, we proceed to the audio domain.

3.2 Assisted Pitch Tracking

Audio pitch tracking, though fairly accurate especially at long window sizes, suffers two related problems in real-time low-latency scenarios. The first is harmonic errors, the tendency of many algorithms to identify the harmonic of a frequency rather than the fundamental. The second is the fundamental limitation of needing one or more periods of a wave to fit in the window, placing a lower bound on window size (and hence latency).

For instance, the low G on a violin (192Hz) takes 5.2ms for a complete wavelength, and on a cello, the low C (65Hz) takes 15.3ms for a complete wavelength. But with sensor-assisted pitch tracking, we begin with a rough estimate of pitch based on finger position. This allows us to use even the most basic algorithm with shorter window settings to achieve a more accurate result.

Finger placement nearly eliminates harmonic errors. Moreover, where the sensors indicate a fundamental frequency below what could be detected with a given window size, we can instead search for a harmonic of that frequency using a shorter window, allowing us to estimate the fundamental in shorter than one period.

3.2.1 Biased Search Restricted Autocorrelation

We used two audio methods for sensor-assisted pitch estimation: biased autocorrelation with quadratic interpolation, and a variation on Yin [7]. Biased autocorrelation is based on Equation 1. This equation is commonly used in signal processing, as it weights events with less lag more than higher-lag events, creating a decay envelope that can be seen in the top of Figure 3. Typically, estimated pitch period is determined by picking the highest peak in the autocorrelation function. However, this method is prone to error, especially at smaller window lengths when there are fewer accumulated wavelengths or notes below the minimum detection frequency. For instance, testing against a sample violin recording, using basic biased autocorrelation with a window size of W = 2048 samples had a 4.8% error rate1, but a 2048- sample window incurs an unacceptably high 46ms latency. Reducing the window to 512 samples, the error rate rose to 21.1%. At 256 samples, corresponding to an inherent latency of 5.8ms at a 44.1 kHz sample rate, the error rate grew to 41.1

We use the sensor-based pitch estimate (Equation 6) to restrict the range on which we search for the maximum of Equation 1. For autocorrelation-based pitch detection, errors tend to occur at harmonics of the correct pitch. While octave errors are most common, fifths are sometimes also found. With this in mind, we started by restricting the autocorrelation search space to be within a just-intonation major third (±25%) of the touch sensor estimate. Subsequent trials found that for small windows, where pitch estimates were less robust, it was useful to use a whole-tone search window (±12.5%), as in all cases, the hardware pitch estimate was typically within 8% of the correct pitch. However, for window sizes above 512 samples, at least two full periods fit within the window for all notes within the violin's frequency range. The increased robustness of the pitch estimate allowed relaxing reliance on the hardware sensors enabling an increased search range of a fourth (±33%).

If there is no position estimate from the fingerboard, we assume the audio must be produced by an open string and restrict our search to a set of narrow frequencies around each string.

If the window length W is too short to effectively evaluate a low frequency, rather than search for the fundamental frequency, we search the area around the expected second harmonic as demonstrated in the second example of Figure 3. The second harmonic will sometimes be a minimum instead of a maximum of the autocorrelation function, since the lag corresponding to the second harmonic also represents a 180^0 phase shift of the fundamental. We thus search for either a minimum or maximum in this case, choosing the stronger of the two responses.

Lastly, the autocorrelation function can only be evaluated at integer multiples τ of the sample interval, even though the actual period of the signal may lie between those multiples. In an effort to estimate the theoretical maxima, we use parabolic interpolation based on the peak value and the two surrounding points [7]. If the optimal lag τ is on the edge of the window and the correlation function is monotonic, we do not perform interpolation.

3.2.2 Restricted Search Yin

We have also implemented a frequency restricted search version of Yin, as Yin typically yields much better results than biased autocorrelation. Yin improves on simple autocorrelation in five stages. The first is the use of the Equation 2 from Section 2.2, a modified autocorrelation based on difference in signals rather than the raw signal. This method increases resistance to errors due to amplitude change. Second, Yin uses a cumulative mean normalized difference function to reduce "too high errors", by weighting a result on its difference from a running cumulative average.

We incorporate the sensor data restriction in the third stage of Yin. Since Yin starts from a difference-based autocorrelation variation, it looks for a minimum instead of a maximum and defines an arbitrary threshold which any result must be below. Yin selects the minimum from the first contiguous set of values under the chosen threshold, or the overall minimum if nothing is below the target threshold.

Rather than using an arbitrary (though effective) threshold, we replace the threshold search by the same frequency restriction technique used in biased autocorrelation, restricting the search for the minimum to be within a given range

of the estimate from the fingerboard sensor. The Yin search uses the same search intervals as the assisted auto-correlation- a just whole tone for $W < 512$ and a just fourth otherwise. Similarly, we apply the same concept of looking for second harmonics when the expected period is otherwise too long for reliable estimation, and we finish with parabolic interpolation. When the sample window is small and the frequency is too low for even a second harmonic Yin estimate, we use the raw sensor pitch estimate.

3.2.3 Implementation

Finger position estimates are received by USB and forwarded by OSC to a VST audio plugin which performs the pitch tracking. The plugin was implemented using JUCE[1] and hosted in the REAPER environment. Block size for audio analysis is selectable, and four pitch estimation techniques can be chosen: biased autocorrelation and Yin, with or without sensor data.

3.2.4 Example

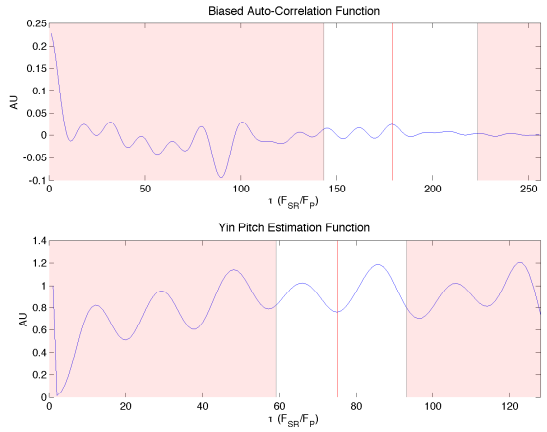

Figure 3: Examples of biased autocorrelation (top) and Yin difference function with cumulative mean (bottom). Both examples, taken at different points in time, use $W = 256$ and F_{SR} of 44.1kHz.

Figure 3 demonstrates the advantages and mechanisms of sensor assistance. While the algorithms would normally search across the whole function, the fingerboard sensor informs us that the pitch must lie in the non-shaded area. In the autocorrelation example (top), the hand labeled target pitch is 246.94Hz, as marked by the red line. The g-string gives us a normalized voltage reading of 0.778 which, using Equation 6, provides a hardware pitch estimate of 243.12Hz. The major third either side of the hardware estimate defines the autocorrelation search area of 194Hz-304Hz, corresponding to a τ between 145 to 227. The algorithm then searches that *tau* range to find the maxima, resulting in a pitch estimate of 247.23Hz.

In the Yin example (Figure 3 bottom), the hand labeled target pitch is 293.66Hz. This frequency is actually below Yin's range when using a 256 sample window at 44.1kHz. However, the finger sensor input tells us the expected frequency is 296.61Hz so we define a minima search around the second harmonic, 593.22Hz, or $\tau = 74.34$. We find the minima at $\tau = 75$ which then undergoes quadratic interpolation and is doubled to provide a pitch final pitch estimate of 294.99Hz. Both examples in Figure 3 are instances where the pitch estimate would be incorrect without the assistance of fingerboard hardware.

[1]http://www.juce.com

4. TESTING AND RESULTS

4.1 Test Setup

Tests of low latency pitch detection were done by recording a performance and the corresponding data feed. In order to directly explore differences in performance for variable window size, results were collected post-performance using the original audio and data for repeat processing. The audio and data samples were synchronized by recording a low sample rate version of the audio as part of the sensor feed and then matching onsets in the low sample rate audio with the regular 44.1kHz sample rate audio. The session was then played back with live analysis of the audio informed by the sensor data measured at the corresponding time when originally recording. The method is equally suited to real-time use in live performance, though latency from gathering the sensor data over USB should be considered and the audio delayed to compensate if needed.

Three sessions of 2-3 minutes made up of 40, 137, 202 and notes were evaluated. Two of the segments were recorded at 44.1kHz, and one at 48kHz. The segments consisted of scales and arpeggios spanning all notes in G major in first position range of the violin (G3-B5). There was a weighting towards lower notes on the D and G string (under 440Hz). As the performance was to be hand labeled using standard pitches, the violinist was asked to avoid vibrato and to try to minimize holding multiple fingers down on different strings since the current system only presently supports monophonic performance. They were otherwise free to play normally.

Pitch estimates were collected for window sizes of 128, 256, 512, 1024, and 2048 samples. In each case, the hop size was set to one quarter of the window, with the exception of the 2048 sample window which, for on-line computational reasons, had a hop size of one half the window. For $W < 512$, the algorithm used a search window of a (just intonation) whole-tone around the fingered expected pitch and a semitone around expected open strings. Otherwise, the search region was within a perfect fourth. Results were filtered to exclude periods of silence (defined as < -48db) and the result from each hop compared against two sets of labels. The first were hand labels of the expected pitch, and the second set was from a 2048 sample window, 64 sample hop-size, Yin-FFT analysis using the Aubio pitch detection plugin by Bossier and Cannam in Sonic Visualiser [5].

The reason for the two comparison sets is that the audio is a human performance on a violin. Hand labels matched intended pitch which may differ from actual pitch. This may be because of performer error or instrument tuning. For instance, in one session, the entire violin was out of tune by 40 cents. As performer pitch error would influence scoring of pitch estimates, a Yin estimate at a high window size was also used as Yin is widely accepted as a solution to (high-latency) monophonic pitch tracking. The Yin estimate will itself have errors as Yin is not 100% accurate– for instance, note changes typically result in momentary loss of valid estimate – however, when stable, the Yin label is expected to be more accurate than the hand label.

Comparisions were made within 5 (0.28%), 10 (0.57%), 30 (1.75%), 50 (2.93%), and 100 (5.95%) cents of the central pitch. Some of these are quite tight tolerances but were chosen based on psycho-acoustic tolerances. Pitch differences within 5 cents are largely indistinguishable, 10 cents is tolerable, an estimate within 50 cents should round to the correct MIDI pitch, while 100 cents is the nearest equally

tempered semitone.

4.2 Results

Results for window size, W, of 128, 256, and 512 are given in Table 1. These window sizes convert to window lengths of 2.9 ms, 5.8 ms, 11.6 ms at 44.1kHz, and 2.7 ms, 5.3 ms, 10.7 ms at 48kHz respectively. Window lengths of 1024 and 2048 samples, 23.2 ms and 46.4 ms, were calculated for reference and discussion, but are not considered fast enough for low-latency use.

	128 Sample Window							
	Hand Labeled				Yin Labeled- 2048 buffer			
	Accuracy within # cents				Accuracy within # cents			
Alg.	10	30	50	100	10	30	50	100
AC	.145	.227	.234	.236	.089	.202	.231	.233
AC+S	**.268**	**.483**	**.559**	**.668**	**.197**	**.462**	**.556**	**.664**
Yin	.103	.171	.178	.180	.066	.159	.181	.185
Yin+S	.095	.227	.317	.527	.073	.204	.304	.541
	256 Sample Window							
	Hand Labeled				Yin Labeled- 2048 buffer			
	Accuracy within # cents				Accuracy within # cents			
Alg.	10	30	50	100	10	30	50	100
AC	.502	.505	.511	.511	.242	.463	.490	.495
AC+S	**.537**	**.769**	**.801**	**.819**	**.458**	**.744**	**.783**	**.803**
Yin	.344	.528	.552	.558	.263	.502	.534	.543
Yin+S	.336	.552	.611	.687	.270	.534	.598	.677
	512 Sample Window							
	Hand Labeled				Yin Labeled- 2048 buffer			
	Accuracy within # cents				Accuracy within# cents			
Alg.	10	30	50	100	10	30	50	100
AC	.507	.720	.736	.741	.441	.711	.736	.39
AC+S	**.660**	**.930**	**.954**	**.962**	.618	.914	.941	.946
Yin	.656	**.930**	.952	.959	**.632**	**.920**	**.948**	**.952**
Yin+S	.649	.923	.949	.957	.629	.908	.937	.940

Table 1: Pitch detection accuracy for 3 window sizes of 128, 256, and 512 samples. Comparisons use both hand-labeled performance data and a 2048 sample Yin-FFT pitch detection analysis.

It is clear that for shorter windows under 512 samples, sensor-assisted pitch estimates outperform traditional "biased" autocorrelation and Yin. Within 100 cents using a 128 sample window, assisted autocorrelation and assisted Yin triple the accuracy of traditional methods. With a 256 sample window, assisted pitch detection still significantly out performs existing means– assisted Yin offering a 25% improvement, and assisted autocorrelation offering a 48% improvement over standard Yin (100-cent accuracy, Yin labels). Increasing the window size to 512 samples, the advantage of sensor assistance is reduced and accuracy differences become insignificant.

At window sizes above 512 samples, assisted pitch-tracking tends to slightly out-perform Yin when evaluated using hand labels: .956 (AC+S) vs .941 (Yin) accuracy at 50 cents, and .952 (AC+S) vs. .928 (Yin) accuracy at 50 cents with a 2048 sample window. This also hold true using Yin-FFT annotations: .939 (AC+S) vs. .931 (Yin) within 50 cents with a 1024 window, and .926 (AC+S) vs .901 (Yin) within 50 cents with a 2048 sample window. It is possible that comparing against Yin annotations will give Yin results an artificial advantage, since both the algorithm under test and the annotations may produce the same errors.

An additional advantage of sensor assisted pitch detection for some contexts is that the error range is much more limited. As visible in Figure 4, for fingered notes, error with assisted autocorrelation is limited to within a minor third with the only large error occuring during an open g-string. In contrast, the raw Yin estimate fluxuates dramatically across more frequencies.

5. DISCUSSION

While the 66.8% hit rate using a 128 sample window is too low for practical use, using assisted autocorrelation with a 256 sample window is promising. The advantage the sensor

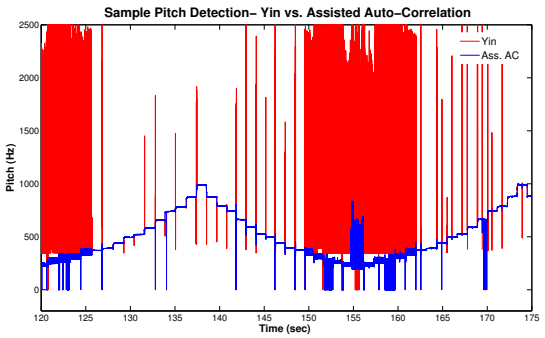

Figure 4: Pitch estimates derived using Yin and Assisted autocorrelation with a 256 sample window and major third search area at 44.1kHz for an audio segment consisting of scales. The sensor assisted results have significantly less variation in error.

information gives the algorithm over an uninformed estimate can be made clear by considering Figure 4. First, the search limits provided by the sensors help screen out noise during a pitch change so that the new pitch is typically found faster. As visible in Figure 4, the loss of pitch estimate during transitions is easily visible with Yin, but far less so with assisted autocorrelation. This additional improvement in accuracy when finding a new pitch directly enhances the low-latency performance.

Next, a 256 sample window at 44.1kHz corresponds to a 5.8 ms window so that no frequency under 172 Hz will complete a wavelength within the window. Additionally performance will improve with multiple wavelengths. Yin has a minimum detection period of $\frac{1}{2}\tau_{max}$ where τ_{max} is limited to the length of the window. Hence, with a 5.8ms window, Yin will fail to detect frequencies under 344Hz, a result clearly illustrated in Figure 4. However, sensor assisted autocorrelation is able to reasonably estimate the correct pitch, is never further off than a minor third, and rarely significantly underestimates the frequency.

One issue with the present sensor arrangement is that on the violin, the finger can stop the string without being pressed firmly into the fingerboard, so a fingered note will not always produce an electrical connection. This is particularly problematic on the thin E-string, which can make a groove in the finger and not contact the fingerboard. An example of failed electrical contact affecting pitch estimates can be seen on the right side of Figure 4, around 170 seconds. Identifying and removing instances where the string unexpectedly lost electrical contact with the fingerboard improves estimation accuracy roughly 1.5-5.0% over present accuracy for both modes of assisted prediction. This improvement is enough that if the contact error can be eliminated, assisted pitch detection is expected to outperform Yin for all window sizes.

Beyond refining sensing arrangements, a major opportunity for improving results is to fully calibrate the hardware sensor for correct pitch. At low window sizes, accurate pitch estimate turned out to be heavily reliant on hardware estimate accuracy, but we did not fully exploit hardware sensor accuracy and stability. For this paper, we used open string voltage as V_{MS} and did not diligently tune the hardware beyond ensuring that fingering near the nut would be minimally detectable. Of the three test sets analyzed, only one had an average hardware pitch estimate error within 100 cents of labeled pitch, thus enabling us to experiment with a semi-tone search window. With a 256 window size and the semi-tone search window, this set was 91% accurate within

100 cents using assisted autocorrelation and 89% accurate using assisted Yin. 3-4% of the error was due to contact error with much of the remaining error occuring during open g-strings. Presumably, if we had calibrated hardware estimates to be closer to expected pitch, we would see higher accuracy at low-latencies.

An additional hardware challenge is that velostat is temperature sensitive so its resistance is not stable. This was dealt with in the short term by adding a potentiometer to control the current supply so we could vary the drive current and more intentionally calibrate the fingerboard sensor. We also found that charge appeared to build in the velostat when left on for a long time. This further altered conductivity but could be countered by turning off power for a while. Both these issues could use more robust solutions.

6. CONCLUSIONS

Combining audio analysis with sensor input allows us to significantly improve pitch-tracking results. In isolation, the audio pitch tracking algorithm and the sensor data both exhibit significant amounts of error, especially at small audio window sizes. However, because the errors are of different types, combining the two produces improvements over the current state-of-the-art in pitch tracking, allowing accurate detection at low latencies.

Improved low-latency pitch-tracking has applications to a wide variety of live electroacoustic performance situations, such as corrective learning strategies fixing pitch, live synchrony with performance derived visuals, and enable non-predictive score following or auto accompaniment. Although the sensor methods presented in this paper are both violin and pitch specific, the utilization of both hardware sensor and audio analysis need not be. As evidenced by the already well utilized "E-Sitar" [14], a similiar strategy of sensor and audio analysis could be employed for a multitude of augmented instruments.

7. ACKNOWLEDGMENTS

Thanks to Ho Huen, Emmanouil Bentos, Nan-wei Gong, Bogdan Vera, Victor Zappi, Siddarth Sigtia and C4DM.

8. REFERENCES

[1] B. D. Adelstein, D. R. Begault, M. R. Anderson, and E. M. Wenzel. Sensitivity to haptic-audio asynchrony. In *Proc. Multimodal Interfaces*, pages 73–76, 2003.

[2] A. Askenfelt. Measurement of bow motion and bow force in violin playing. *J. Acoustical Society of America*, 80:1007, 1986.

[3] E. Benetos. *Automatic transcription of polyphonic music exploiting temporal evolution*. PhD thesis, 2012.

[4] J. C. Brown. Musical fundamental frequency tracking using a pattern recognition method. *J. Acoustical Society of America*, 92:1394, 1992.

[5] C. Cannam, C. Landone, M. B. Sandler, and J. P. Bello. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *ISMIR*, pages 324–327, 2006.

[6] A. de Cheveigné and H. Kawahara. Comparative evaluation of f0 estimation algorithms. In *INTERSPEECH*, pages 2451–2454, 2001.

[7] A. De Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *J. Acoustical Society of America*, 111:1917, 2002.

[8] A. Freed. Novel and forgotten current-steering techniques for resistive multitouch, duotouch, and polytouch position sensing with pressure. In *Proc. NIME*, 2009.

[9] A. Freed, A. Chaudhary, and B. Davila. Operating systems latency measurement and analysis for sound synthesis and processing applications. In *Proc. ICMC*, pages 479–81, 1997.

[10] A. Freed, F.-M. Uitti, S. Mansfield, and J. MacCallum. "old" is the new "new": a fingerboard case study in recrudescence as a nime development strategy. In *Proc. NIME*, 2013.

[11] T. Grosshauser, S. Feese, and G. Tröster. Capacitive left hand finger and bow sensors for synchronization and rhythmical regularity analysis in string ensembles. In *Stockholm Music Acoustics Conference*, 2013.

[12] T. Grosshauser and G. Tröster. Further finger position and pressure sensing techniques for stringed and keyboard instruments. In *Proc. NIME*, 2013.

[13] A. Jensenius and V. Johnson. Performing the electric violin in a sonic space. *Computer Music J.*, 36(4), 2012.

[14] A. Kapur, A. J. Lazier, P. Davidson, R. S. Wilson, and P. R. Cook. The electronic sitar controller. In *Proc. NIME*, 2004.

[15] M. Lahat, R. Niederjohn, and D. Krubsack. A spectral autocorrelation method for measurement of the fundamental frequency of noise-corrupted speech. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(6):741–750, 1987.

[16] E. Maestre, J. Bonada, M. Blaauw, A. Perez, and E. Guaus. Acquisition of violin instrumental gestures using a commercial EMF tracking device. In *Proc. ICMC*, volume 1, pages 386–393, 2007.

[17] T. Mäki-Patola and P. Hämäläinen. Latency tolerance for gesture controlled continuous sound instrument without tactile feedback. In *Proc. ICMC*, 2004.

[18] M. V. Mathews and J. Kohut. Electronic simulation of violin resonances. *J. Acoustical Society of America*, 53:1620, 1973.

[19] A. M. Noll. Cepstrum pitch determination. *J. Acoustical Society of America*, 41:293, 1967.

[20] D. Overholt. The overtone violin: A new computer music instrument. In *Proc. ICMC*, pages 604–607, 2005.

[21] D. Overholt. The overtone fiddle: an actuated acoustic instrument. In *Proc. NIME*, 2011.

[22] J. Paradiso and N. Gershenfeld. Musical applications of electric field sensing. *Computer Music J.*, 21(2):69–89, 1997.

[23] L. S. Pardue and A. P. McPherson. Near-field optical reflective sensing for bow tracking. In *Proc. NIME*, 2013.

[24] C. Poepel and D. Overholt. Recent developments in violin-related digital musical instruments: where are we and where are we going? In *Proc. NIME*, 2006.

[25] L. Rabiner. On the use of autocorrelation analysis for pitch detection. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 25(1):24–33, 1977.

[26] N. Rasamimanana, E. Fléty, and F. Bevilacqua. Gesture analysis of violin bow strokes. *Gesture in Human-Computer Interaction and Simulation*, pages 145–155, 2006.

[27] E. Schoonderwaldt and M. Demoucron. Extraction of bowing parameters from violin performance combining motion capture and sensors. *J. Acoustical Society of America*, 126:2695, 2009.