SCORE BASED REAL-TIME PERFORMANCE WITH A VIRTUAL VIOLIN

Matthias Demoucron
IRCAM -Centre Georges Pompidou,
Paris, France
demoucron@ircam.fr

Nicolas Rasamimanana
IRCAM -Centre Georges Pompidou,
Paris, France
nicolas.rasamimanana@ircam.fr

ABSTRACT

This paper describes the implementation of a violin physical model tied with the control of music scores to enable the real-time performance of music pieces. The violin model is made of four strings, which allows the performance of double stops, chords and specific resonant effects that can be encountered in violin playing. A graphic tablet is used to control the bowing parameters and to trigger automatically note events contained in a specifically formatted MIDI file. The automatic pitch change helps reducing the violin playing complexity and enables the user to focus on sound shaping and phrasing. The device can be used for pure sound synthesis purposes as well as for experiments related to violinists' sound control. However, the simplified interface for sound and score events is particularly suitable for non violinists wishing to explore expressive capabilities of the instrument and to experience specific features of violin playing.

1. INTRODUCTION

This work aims at developing a simple real-time implementation of a violin model for sound synthesis purposes and musical performance studies. Sound synthesis of sustained instruments like the violin requires a continuous control of the simulated string vibration through three main bowing parameters which are the bow force, the bow velocity, and the bow-bridge distance. The realism of the sound is highly dependent on a realistic control of these parameters and on optimal interaction between the model and the user.

Following Serafin [1], we used a graphic tablet to control the violin model. This interface allows the control of the three main bowing parameters in a rather similar way as in real playing, making the control particularly easy for violinists, and not too difficult to approach for non-experienced users. However, while Serafin mainly focused on control studies and examination of bow strokes, we wanted to allow the performance of music scores in order to facilitate musical applications and music performance related experiments.

Performing music scores required to model the four strings of the violin in order to avoid limitations in the violin repertoire and improve the realism of the performance. Secondly, a simplified implementation was necessary because we wanted to make it accessible even for non violinists. In particular, we decided to make pitch changes automatic so that the user can focus on bowing gestures and on the gesture based shaping of the sound.

In this paper, we first present the physical model that was used for synthesis of the violin sound, and its implementation in Max/MSP environment (Sect. 2). In Sect. 3, we describe how MIDI files were formatted in order to assign pitches to the different strings, and we examine the specific question of chords performance. Sect. 4 shows how the model and the score were controlled by the graphic tablet. Finally, in Sect. 5, we conclude by discussing possible applications for experimental studies, sound synthesis and music pedagogy.

2. SYNTHESIS MODEL OF THE VIOLIN

In this section, the numerical implementation of the string equation for simulation purposes is described. Then, the implementation of a complete violin with four strings is presented and implemented for real-time application in Max/MSP.

2.1. Modal derivation of the string equation

The model is based on the modal decomposition of the string equation, which has already been described by several authors [2, 3]. The displacement $y(x,t)$ of the string at a position x, with a spatial distribution of external forces $F(x,t)$ is written as

$$\rho_L \frac{\partial^2 y(x,t)}{\partial t^2} - T\frac{\partial^2 y(x,t)}{\partial x^2} + EI\frac{\partial^4 y(x,t)}{\partial x^4} = F(x,t) \quad (1)$$

where ρ_L is the linear density of the string, T the tension, E the Young modulus, and $I = \frac{\pi d^4}{64}$ the second moment of area for a circular cross section of the string, with d being the diameter of the string.

For the case of supported ends, the dispersion relation gives

$$\omega_{0n} = \sqrt{\frac{T}{\rho_L}(\frac{n\pi}{L})^2 + \frac{EI}{\rho_L}(\frac{n\pi}{L})^4}$$

where L is the string length, and the solutions $y(x,t)$ can be written with the modal vectors $\phi_n(x)$ as

$$y(x,t) = \sum_{n=1}^{\infty} \phi_n(x) a_n(t), \qquad \phi_n(x) = \sqrt{\frac{2}{L}} \sin\frac{n\pi x}{L} \quad (2)$$

External forces can be expressed on the same base

$$F(x,t) = \sum_{n=1}^{\infty} \phi_n(x) f_n(t) \quad (3)$$

Note that, for the simulations, the summation over n will be truncated to N modes instead of infinity for obvious computing reasons.

Inserting Eq. 2 and Eq. 3 in the string equation (Eq. 1), one find an infinity of second order differential equations depending on time for the modal displacements $a_n(t)$

$$\ddot{a}_n(t) + 2r_n\dot{a}_n(t) + \omega_{0n}^2 a_n(t) = \frac{1}{\rho_L} f_n(t) \quad (4)$$

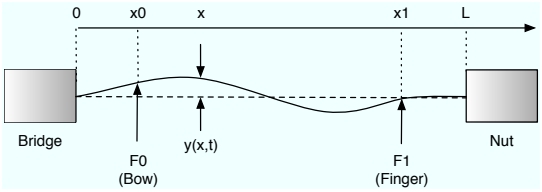

Figure 1: *Schematic representation of the bowed string and notations. The displacement $y(x,t)$ is given by the string equation. External forces F_0 and F_1 represents the action of the bow and the finger, respectively at x_0 and x_1. The two terminations are modelled with supported ends.*

where damping coefficients r_n have been introduced to account for losses.

For the bowed string, external forces are composed of the friction force $F_0(t)$, resulting from the drawing of the bow across the string, and the force $F_1(t)$, representing the action of the finger in order to set the effective vibrating length of the string.

Assuming that each force is applied at a single point of the string, the force distribution can be written as

$$F(x,t) = \delta(x-x_0)F_0(t) + \delta(x-x_1)F_1(t),$$

and the modal components of the force are

$$\begin{aligned} f_n(t) &= \int_{x=0}^{L} \phi_n(x)F(x,t)dx \quad (5)\\ &= \phi_n(x_0)F_0(t) + \phi_n(x_1)F_1(t) \quad (6)\end{aligned}$$

To summarise, the method allows to change the string equation depending on position x and time t into an infinity of equations (Eq. 4) depending only on time, which can be easily implemented with second order filters.

2.2. Numerical implementation

The modal equations with a friction interaction are solved using the same method as [4] , which is summarised in this section. If the forces are assumed to be constant between successive time steps, Eq. 4 can be directly integrated, which provides the expression for the modal component at each time step t_1. Denoting t_0 the previous time step and $dt = t_1 - t_0$, the modal components are

$$a_n(t_1) = a_n^h + X_{3n}\ f_n(t_1) \quad (7)$$

$$\dot{a}_n(t_1) = \dot{a}_n^h + Y_{3n} f_n(t_1) \quad (8)$$

for the displacement and the velocity, respectively, with

$$\begin{aligned} X_{3n} &= \frac{1}{\rho_L \omega_{0n}^2}(1 - (\cos\omega_n dt + \frac{r_n}{\omega_n}\sin\omega_n dt)\exp(-r_n dt))\\ Y_{3n} &= \frac{1}{\rho_L \omega_{0n}^2}(\omega_n + \frac{r_n^2}{\omega_n})\sin\omega_n dt \exp(-r_n dt)\end{aligned}$$

The parameters a_n^h and $\dot{a}_n^h$ are the values that the modal components would take if no force was applied, and

$$\omega_n = \sqrt{\omega_{0n}^2 - r_n^2}$$

With Eq. 8, the velocity at $x_i = x_0$ or x_1 can be written as the velocity v_i^h that the string would take if no force was applied, plus the contribution of external forces F_0 and F_1.

$$\begin{pmatrix} \dot{y}(x_0,t_1) \\ \dot{y}(x_1,t_1) \end{pmatrix} = \begin{pmatrix} v_0^h \\ v_1^h \end{pmatrix} + B \begin{pmatrix} F_0(t_1) \\ F_1(t_1) \end{pmatrix} \quad (9)$$

where the elements B_{ij} of matrix B are

$$B_{ij} = \sum_{n=1}^{N} \phi_n(x_i)\phi_n(x_j)Y_{3n}$$

An expression for the finger force F_1 is deduced from the requirement that the string should not move at the string position x_1. This can be obtained by requiring $y(x_1,t)$ or $\dot{y}(x_1,t)$ to be zero or, alternatively, by modelling the finger as a very stiff spring or a very strong damper. In our case, we choose the later alternative, which can be formalised with

$$F_1(t) = -R\dot{y}(x_1,t)$$

Using this relation in Eq. 9, we obtain for the finger force a solution that depends on the friction force $F_0(t_1)$.

$$\begin{aligned} F_1(t_1) &= \frac{-R}{1+RB_{11}}(v_1^h + B_{01}F_0(t_1))\\ &= C_{11}v_1^h + C_{10}F_0(t_1) \quad (10)\end{aligned}$$

Note that with a high number of modes N, or for a long distance between the finger and the bow (which is usually the case in violin playing), the term in F_0 could even be neglected, which would make straightforward the solution of Eq. 10.

In violin, decay times of the string vibration are normally much longer with an open string than with a fingered one. We can consequently see the interest of using such a model for the finger force: it puts additional losses to string, reproducing the damping effect of the finger.

A similar relation for the friction force is obtained by replacing F_1 with Eq. 10 in Eq. 9

$$\begin{aligned} F_0(t_1) &= \frac{\dot{y}(x_0,t_1) - v_0^h - B_{01}C_{11}v_1^h}{B_{00} + B_{01}C_{10}}\\ &= C_{00}(\dot{y}(x_0,t_1) - v_0^h) + C_{01}v_1^h \quad (11)\end{aligned}$$

The string velocity under the bow $\dot{y}(x_0,t_1)$ is now unknown and must be deduced from specific conditions introduced by the friction interaction. In order to compute the friction force F_0, two cases must be considered: sticking and sliding. When the string is sticking to the bow, the relative velocity between the string and the bow $\Delta v = \dot{y}(x_0,t) - v_b$ is assumed to be zero, then Eq. 11 gives

$$F_0(t_1) = C_{00}(v_b - v_0^h) + C_{01}v_1^h$$

When the bow is sliding under the bow, the friction force is related to Δv with a classical hyperbolic friction model [5, 6] depending on the bow force F_b

$$F_0 = Sgn(v_b)\Big(\mu_d + \frac{\mu_s - \mu_d}{1 + \frac{|\Delta v|}{v_0}}\Big)F_b \quad (12)$$

where μ_s and μ_d are the static and dynamic friction parameters and v_0 describes the slope of the non linearity. Eq. 12 must then

be solved together with Eq. 11 in order to deduce the couple (F_0 - $\dot{y}(x_0)$) to be used in the computation.

The transitions between the two friction states are decided as following. From sticking, the state changes into sliding when the friction force is more than the allowed maximal friction force $\mu_s F_b$. From the sliding state, the state changes into sticking when Eq. 11 and Eq. 12 have no solution.

Once the friction force F_0 is known, the finger force F_1 can be computed with Eq. 10. The modal displacements and velocities are then deduced from Eq. 7 and Eq. 8.

2.3. Modelling a complete violin

Because our goal is to perform typical violin scores with the synthesis model, all the compass of the violin should be accessible to the user, from G3 to notes as high as C8, which requires the possibility of playing on one string or the other. Except for the lowest notes (under the D4 string), each note can be played on different strings, with a low position on the fingerboard, or a higher position, which changes the timbre of the sound.

A first and computationally economical solution is to implement only one string whose parameters will change during the performance when string crossings are required. However, this will drastically restrict the accessible repertoire: typical violin playing often involves more than one string played at the same time in order to perform chords or polyphonic scores (see Fig. 2a). Furthermore, allowing only one string to sound would make the performance less realistic: even in monophonic scores, string resonance resulting from rapid string crossings are often encouraged for musical reasons. Consider for example the musical excerpt in Fig. 2b where the A4 are alternatively played on the open A4 string and on the fingered D4 string in order to create a perpetual resonance behind the melody. When played on the D4 string only, this example losses much of its interest.

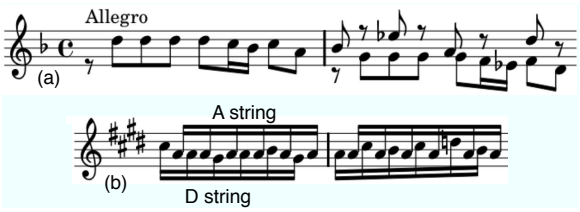

Figure 2: *Typical musical examples requiring more than one string. (a) Polyphonic playing with one leading voice and an accompanying voice (Bach, Fuga from the Sonata I). (b) Rapid string crossings with a repeated open A sounding behind the main theme (Bach, Preludio from the Partita III)*

Consequently, a model including all four strings was implemented. The main string parameters corresponding to each string are summarised in Table 1. Typical data for the tension, length, diameter and density were provided by Pickering [7]. Other parameters like the damping parameters were adjusted in order to get a decay time around 0.4 s for the fundamental and less than 0.05 ms starting from the fifth partial. For all strings, the number N of modes was 15, which we found a good compromise between synthesis quality and computational efficiency, and the computing frequency was set to the audio rate 44100 Hz. Note that other bowed string instruments (viola or cello, for example) could be simulated by simply setting the right parameters for the corresponding strings.

Table 1: *Parameters used for each of the four violin strings. The data mainly come from measurements by Pickering [7]. The last row indicates the fundamental frequency of the string.*

	G string	D string	A string	E string
T (N)	44.6	34.8	50	72.6
L (m)	0.33	0.33	0.33	0.33
d (mm)	0.8	0.8	0.56	0.31
ρ_L (g/m)	2.66	0.92	0.59	0.38
f_0 (Hz)	196	294	441	662

In order to obtain an acceptable violin sound, forces applied by the four strings at the bridge termination were computed and summed up. Then, the resulting total force was convolved with a violin impulse response recorded when striking the edge of the bridge. With this model, some coupling effects of the strings could be introduced as well. When one string is not played, the friction force could be replaced by a driving force given by the bridge forces of all played strings, applied close to the string termination and representing the action of the bridge. This refinement was let to future work.

The bowing parameters are not independent when playing several strings together. When two strings were bowed, the same bow velocity and the same bow-bridge distance was used for the two strings. The case of bow force needs a specific treatment because, when playing double strings, players often press one string stronger than the other in order to highlight one of the voice, for example the leading voice. The bow force should consequently be set for each individual string in order to adjust the relative pressing on each string. Similarly, the performance of chords with three or four notes usually leads to more or less smooth crossings from one string to the other, which involves a time evolution of the pressing on the strings (see Sect. 3). A fine adjustment of the force factor was then used to set the bow force for each string.

Force factors were also used to set the current state of the string: "played" (positive value), when the bow rubs the string, "free" (negative value), when the bow does not touch the string, and "off" (zero), when the string is not allowed to sound at all, which can be interesting in order to stop the resonance, for example.

2.4. Max/MSP implementation

The model described before was implemented as a Max/MSP external object in order to facilitate real-time control. Three DSP inlets are used to input the bowing parameters. String parameters can be set independently for each string by sending the message formatted as "String (string number) (string parameter) (value)". Finger positions x_1 are automatically computed from desired frequencies. Pitches can additionally be changed all together with the message "Freqs" followed by the four new frequencies. Bow force factor can be changed similarly with the message "Forces" followed by the four new values.

The model can normally be used with an Apple PowerPC G4, using around 40 % CPU. However, using the model in combination with a graphic tablet to offer a continuous control (see Sect. 4) was more demanding, and interruptions in the tablet control datastream made the resulting sound not acceptable. All these stream

management troubles are actually solved when using a MacPro Eight Core with 6 Go RAM.

3. FORMATTING MIDI FILES FOR THE CONTROL OF THE MODEL

The real-time performance of music scores is achieved using MIDI files. However a specific formatting had to be carried out in order to manage the attribution of pitches and bow force factors to the different strings. For that purpose, one track was attributed to each string, starting from track 1 for the E5 string, to track 4 for the G3 string. This section describes this formatting and some simple rules in order to facilitate the attribution of the notes to the strings.

3.1. Monophonic phrases

When notes are played one after each other, a specific formatting of the MIDI files is not necessary. A minimal procedure would be to assign each string to different intervals of pitches. For example, G3 to D4 would be assigned to the G3 string, Ds4 to A4, to the D4 string, and so on. However, as discussed in Sect. 2, one could sometimes look for specific effects in which it is necessary to assign a specific string to one note. It is the case, for example, when damped sounds resulting from playing at a high left hand position are looked for, or when successive notes are quickly played on the four strings to obtain resonance effects. In that case, strings must be assigned manually to each note, which can be a strenuous task.

A semi automatic procedure was used in order to facilitate the MIDI formatting. Pitches intervals were attributed to different strings in function of an offset coefficient representing the hand position on the fingerboard. For example, on the G3 string, offset 0 gave G3 to Db4, offset 1 gave Gs3 to D4, offset 2 gave A3 to Eb4, etc. During a preliminary step, MIDI scores had to be annotated in order to describe offsets for each fragment. The attribution to adequate strings was then automatic.

This procedure made the formatting of MIDI files easier in most cases because hand position changes are comparatively less frequent than string crossings, usually. In some very specific cases like Fig. 2b, however, it was not possible to use the procedure and adjustments had to be made manually. Similarly, it was sometimes easier to assign directly a segment to a specific string, for example when phrases are entirely played on one string.

The procedure provided a good alternative between assigning manually the strings and making the procedure entirely automatic without considering musical requirements. It seems difficult to propose a satisfying automatic procedure because many choices require the knowledge of bowing technique, human limitations and musical tastes.

3.2. Chords and double stops

When dealing with chords, two specific problems have to be examined. The first, again, is to assign strings to notes in the MIDI file. The same procedure as before was used, the offset being decided in function of the highest note. Then, lower notes were successively assigned to remaining strings, in pitch decreasing order.

The second problem is that more than two strings can normally not be played all together. Chords are often executed as two successive double stops. The transition from one double stop to the other in the same chord depends on the musical context. Fig. 3 shows the bow motion recording of two successive chords performed on a real violin by a player, measured with an optical motion capture device [8]. The bow inclination allows visualisation of the string being played and illustrates timing of the string crossing (grey areas). The first chord is rather long, with a strong impulse on the lowest double stop G3-D4, followed by a short transition to the double stop B4-F5 which is kept during a long time before B4 disappears. The second chord has a rather similar shape, but the highest double stop is almost not played, the bow reaching very quickly the E string in order to play G5 alone.

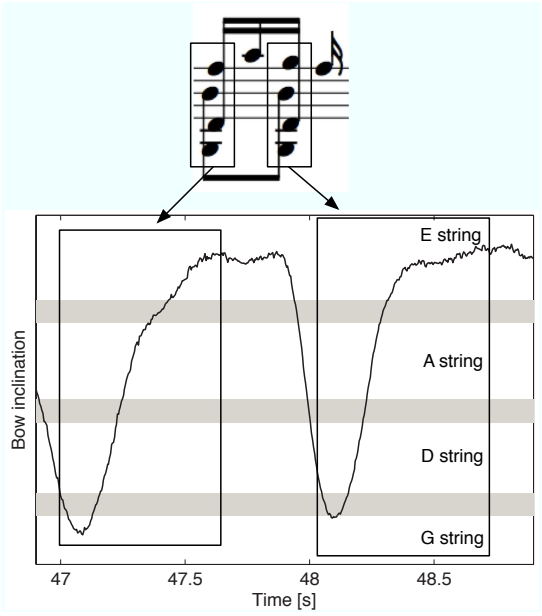

Figure 3: *Examples of chords. The figure exhibits differences in the bow inclination vs time measured during a real perfomance of two successive chords. Area corresponding to the four strings are indicated. Grey area approximately correspond to double stoppings.*

Chords can be played in many different ways and these two examples do not provide a complete taxonomy of multiple strings performance. More than two strings can be played simultaneously, for example by pressing strongly on the bow and playing close to the fingerboard. In contrast, very soft chords are sometimes played without making successive double stops and just making sound the individual notes in a smooth motion of the bow across all the strings. In our case, we did not wish to offer a subtle control on chord performance and we only implemented a standard shape for the chords with two successive double stops whose respective duration can be roughly set in order to allow emphasis on the different parts of the chord. For example, for a three notes chord on the G, D and A strings, the message "Forces 1 1 -1 -1" was first sent to the model, followed quickly (about 100 ms) by "Forces -1 1 1 -1", and finally "Forces -1 -1 1 -1" if the highest note had to be played alone. More elaborate models involving smooth time evolution of the force factors, control of the chord performance, or specific formatting of the chords in the MIDI file, have been experimented but not developed further for the moment.

In order to deduce the direction of the chord (from the lowest strings to the highest, or the contrary) and the succession of messages to be sent, notes were separated into leading voices and

accompanying voices. The channel numbers 1 and 2 were assigned to the leading voice and the accompanying notes, respectively. The leading voice was also the last one which was played at the end of the chord model execution, and consequently entirely controlled by the user.

4. REAL-TIME CONTROL WITH A GRAPHIC TABLET

4.1. Principle

A graphic tablet (Intuos 3 by Wacom) was used to control the violin model. The tablet gives information about the pressing force and the position (x,y) of the pen on the surface. As in [1], bow-bridge distance was given by the position on the shortest axes and bow velocity was computed by differentiation of the position in the other direction (see Fig. 4). In contrast with Serafin [1], additional information provided by the interface, like the pen angle in the two directions were not used.

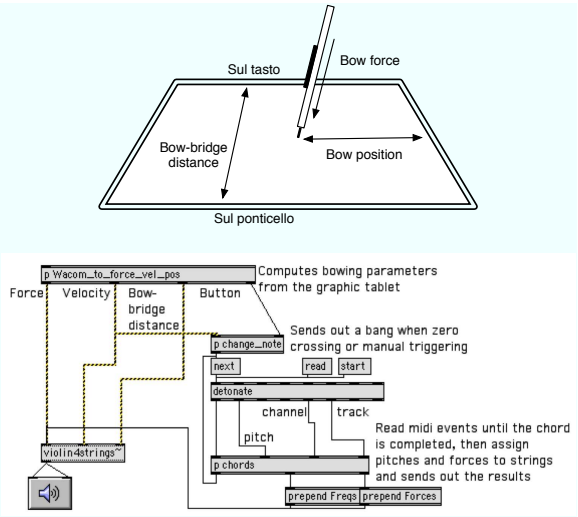

Figure 4: *Top: Schematic representation of the bowing parameters controlled by the graphic tablet. Bottom: Max/MSP patch showing the implementation of the model.*

All control parameters could be scaled in order to match typical ranges of parameters or facilitate the control. For example, bow force was normally scaled between 0 and 2 N, which is the typical range observed in real violin playing [9]. Bow velocity was usually the effective velocity of the pen, but could also be scaled to a greater value in order to compensate for reduced dimensions of the tablet compared to the bow. A low pass filter (cut-off frequency 50 Hz) was applied before sending data to the violin model. The sampling rate for control parameters was about 100 Hz.

4.2. Interest and limitations

Control gesture on a graphic tablet is very similar to the real gesture of the player. The pen is drawn across the surface as does the bow across the string, imitating the control of the velocity, position and force of the bowing gesture. However, several differences were already pointed out by Serafin and can reduce the control possibilities.

In real playing, information about the control are transmitted through different channels including mainly the sound and the tactile feedback (vibrations of the bow felt in the fingers, for example [10]). The effort necessary to move the bow is another important feedback, as well as the reaction of the bow and string when applying bow force. The tablet offers a tactile feedback that corresponds to the friction of the pen on the surface, not directly related to the vibration of the string.

Some features of violin control are highly dependent on the mechanical properties of the bow. Technical gestures often can not be obtained without the help of the bow reaction. For example, it is very difficult to perform the typical bow force lessening encountered in *staccato* playing without the spring-like acting of the bow. Similarly, bouncing bow strokes like *spiccato* require the elasticity of the bow in order to make it jump after contact with the string. This bouncing can be imitated on the rigid tablet by quickly moving down and up the pen, which is a slightly modified gesture compared to the real gesture, as mentioned by Serafin [1]. However, the user can hardly reach the typical contact times measured in real performance of *spiccato*: rebounds on the tablet can hardly last less than 100 ms while typical values in violin playing are around 50 ms [11]. Similarly, repeated, quickly performed, rebounds are also difficult to obtain because they rely upon the equilibrium between the player's action on the stick and bounce mode frequencies of the bow.

Finally, we already mentioned the reduced length of the tablet, compared to the violin bow. The tablet used for performances measured about 20 cm in the long direction. For comparison, usual lengths of the bow are around 65 cm. This difference was a limitation when performing long notes, for example. It was then necessary to make repeated to and fro motions, or to "save bow" more than usual in order to match the expected duration. Alternatively, the length of the tablet could be scaled in the patch in order to provide higher velocities with a reduced motion of the pen.

4.3. Control of note changes

Midi sequences were played event-by-event and triggered by zero crossings in bow velocity. Each time the velocity crossed zero (i.e. at each "bow change"), the next note or chord in the score was sent out in order to set pitches and forces parameters of the violin model. This allowed the user to focus only on the control gesture without having to control pitch changes at the same time, which reduced strongly the complexity of controlling the performance.

In particular, this means that only separate note could be played. We also wanted to offer the possibility of playing *legato*, i.e. with several notes during the same stroke. Events could consequently be triggered manually as well by pushing a button. In addition, another button was used to interrupt automatic pitch changes at zero crossings. This was used, for example, in order to avoid event triggering when more than one length of the tablet was necessary to play long notes.

5. CONCLUDING EXAMPLE AND APPLICATIONS

Fig. 5 illustrates the control parameters of a musical sequence with the implementation described in the previous sections. Bowing parameters are controlled with the graphic tablet and pitches are changed each time the "bow" changes direction. In this case, the "player" was free to adjust his gesture in order to obtain a musically acceptable performance. Bow-bridge distance is slowly vary-

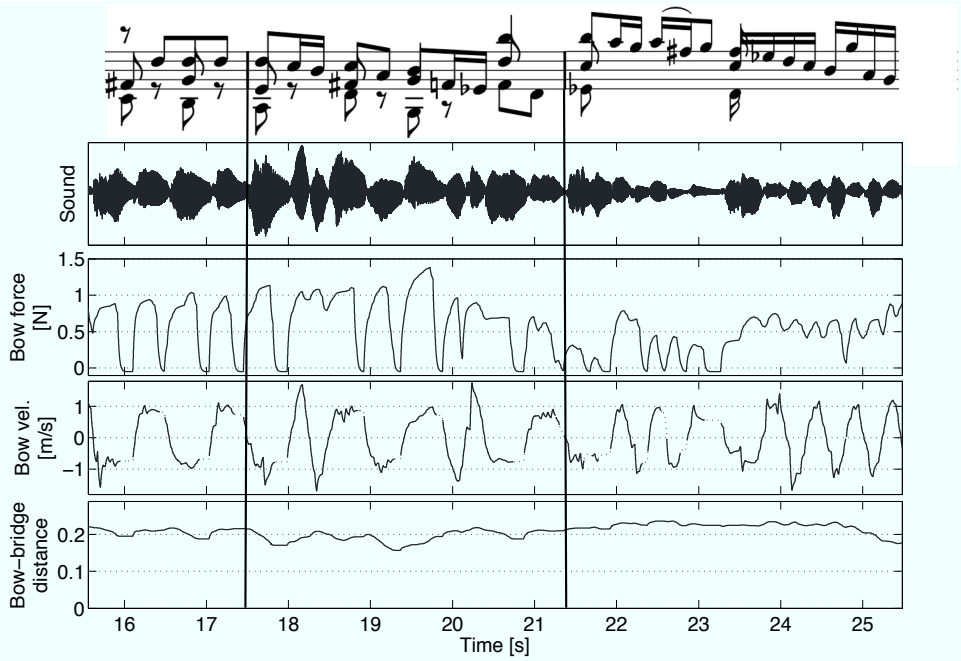

Figure 5: *Example of real time control of the model. The figure shows bowing parameters obtained during the performance of a musical example (Fuga from Sonata I, by Bach). From the top: music score, simulated sound, bow force, bow velocity and bow-bridge distance.*

ing around 0.2, which is a bit more than usual reported values [9]. Bow velocity reaches more than 1 m/s and the bow force shows that notes are clearly separated, with the pen leaving the surface between notes, except for the last few notes which are played with the "bow" in constant contact with the "strings".

We intentionally showed an example that is both technically and musically difficult from a violin playing point of view. Such a score would usually require several years of violin practise in order to manage the double stops and chords with the fuga melody. With the proposed implementation of a virtual violin, this music score can be easily played and controlled by anyone.

Current applications aim at evaluating the device in pedagogical situations. The idea is that, by temporarily easing technical difficulties related to the left hand, string crossings and bow holding, a violin student can focus on building his interpretation of the piece through the control of sound production and musical expression related to bowing gesture. Similar approaches were already experimented in [12]. This is of course not a replacement for the practice of the real instrument and it should be seen as a complementary tool (or musical game) in order to experience some musical issues which normally come later in the education, after years and years of musical frustration. . .

Another application relates to musical performance studies. Used by an expert violinist, the interface is a practical tool for analysing the relation between the control of the virtual violin and musical expressivity. For that purpose, performances using the implementation could be characterised and compared with real recorded performances (gesture and sound) in order to examine the expressive capabilities of the virtual instrument.

6. REFERENCES

[1] S. Serafin, R. Dudas, M. Wanderley, and X. Rodet, "Gestural control of a real-time physical model of a bowed string instrument," in *Proc. ICMC 99: International Computer Music Conference 1999*, Beijing, China, October 1999.

[2] J.D. Morrison and J.M. Adrien, "Mosaic: A framework for modal synthesis," *Computer Music Journal*, pp. 45–56, 1993.

[3] J. Antunes, M. G. Tafasca, and L. L. Henrique, "Simulation of the bowed-string dynamics: part 1 - a nonlinear modal approach," in *Congrès Français d'Acoustique*, Lausanne, Suisse, 2000.

[4] M. Demoucron, *On the control of virtual violins. Physical modelling and control of bowed string instruments*, Ph.D. thesis, Université Paris VI, Royal Institute of Technology (KTH), 2008.

[5] H. Lazarus, *Die Behandlung des selbst erregten Kippschwingungen des gesmchenen Saite mit Hilfe des Laplace Transformation*, Ph.D. thesis, Technical University of Berlin, 1972.

[6] S. Serafin, *The sound of friction: real-time models, playability and musical applications*, Ph.D. thesis, Stanford University, 2004.

[7] N. C. Pickering, "Physical properties of violin strings," *Catgut Acoust. Soc. J.*, vol. 44, 1985.

[8] E. Schoonderwaldt, M. Demoucron, and N. Rasamimanana, "A setup for measurement of bowing parameters in bowed-string instrument performance," in *Acoustics 08 (joint meeting of the EAA and ASA)*, Paris, 2008.

[9] A. Askenfelt, "Measurement of bow motion and bow force in violin playing," *Journal of the Acoustical Society of America*, vol. 80, no. 4, pp. 1007–1015, 1986.

[10] A. Askenfelt and E.V. Jansson, "On vibration sensation and finger touch in stringed instrument playing," *Music Perception*, vol. 9, no. 3, pp. 311–349, 1992.

[11] A. Askenfelt and K. Guettler, "The bouncing bow: An experimental study," *Catgut Acoustical Society (CAS) Journal*, vol. 3, no. 6, pp. 3–8, November 1998.

[12] N. Rasamimanana, F. Guedy, N. Schnell, J-P. Lambert, and F. Bevilacqua, "Three pedagogical scenarios using the sound and gesture lab," in *4th i-Maestro Workshop on Technology Enhanced Music Education*, 2008.