Physical Modeling meets Machine Learning: Teaching Bow Control to a Virtual Violinist

Graham Percival*, Nicholas Bailey*, George Tzanetakis†

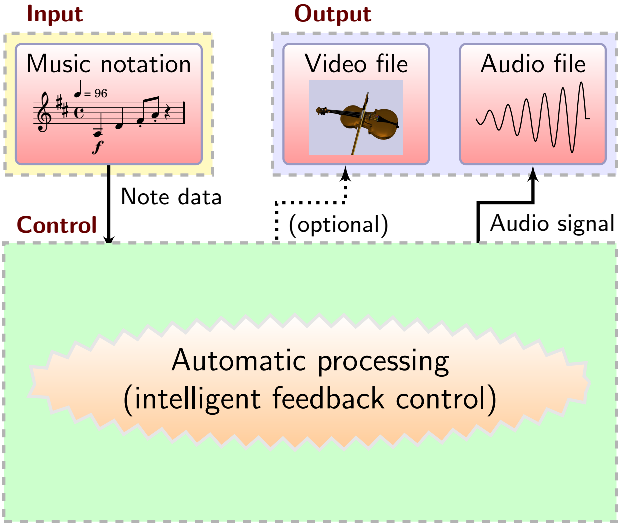
* School of Engineering, University of Glasgow, UK
† Department of Computer Science, University of Victoria, Canada

http://percival-music.ca/vivi.html

Teaching Bow Control to a Virtual Violinist

1. Introduction
   - Music performance with *Vivi, the Virtual Violinist*
2. Performing on a virtual instrument
   - Generating sound: Physical modeling of a violin
   - Pedagogical inspiration for physical parameters
3. Intelligent control loop
   - Intelligent feedback control of bow force
   - Automatically determining other parameters
4. Making music
   - Musical performance style
   - Conclusion and future work

Music performance with *Vivi, the Virtual Violinist*



Input

Music notation

Output

Video file

Audio file

Note data

(optional)

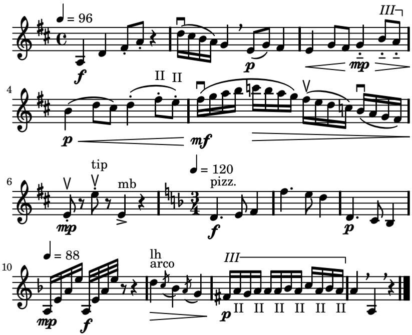
Audio signal

Control

Automatic processing
(intelligent feedback control)

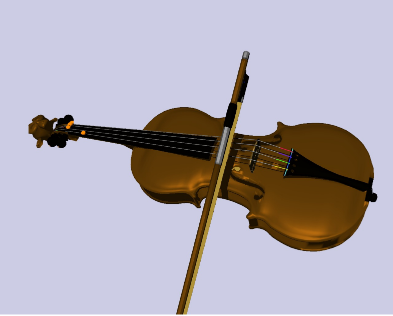
Music example: “black-box testing”

Input



(pdf produced with GNU LilyPond, MusicXML input also possible)

Output



Video: black-box.mpeg

Generating sound: Physical modeling of a violin

- No recordings of violin performance; we use physics [1]
  - Wave equation for a stiff string with modal dampening

$$\rho_{\mathrm{L}} \frac{\partial^2 y(x,t)}{\partial t^2} - T \frac{\partial^2 y(x,t)}{\partial x^2} + EI \frac{\partial^4 y(x,t)}{\partial x^4} + R_L(\omega) \frac{\partial y(x,t)}{\partial t} = F(x,t)$$

[1] M. Demoucron, "On the control of virtual violins: Physical modelling and control of bowed string instruments," Ph.D. dissertation, IRCAM, Paris, 2008

Generating sound: Physical modeling of a violin

- No recordings of violin performance; we use physics [1]
  - Wave equation for a stiff string with modal dampening

$$\rho_{\mathrm{L}} \frac{\partial^2 y(x,t)}{\partial t^2} - T \frac{\partial^2 y(x,t)}{\partial x^2} + EI \frac{\partial^4 y(x,t)}{\partial x^4} + R_L(\omega) \frac{\partial y(x,t)}{\partial t} = F(x,t)$$

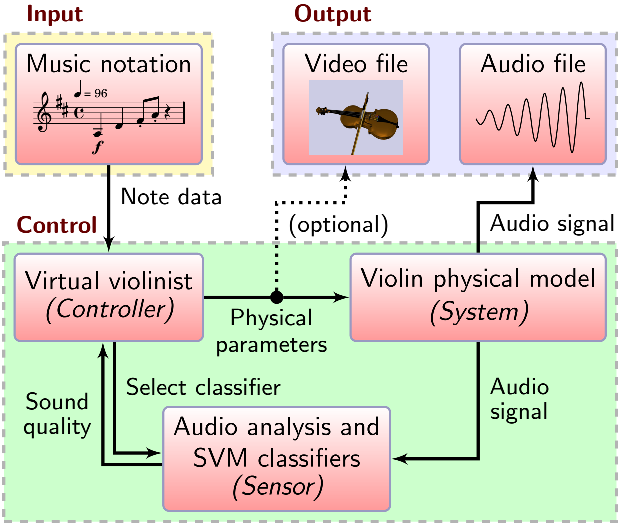
- Implemented as a C++ library, published under GNU GPLv3+

Input parameters

- Violin string number s
- Left-hand finger position x_1
- Bow-bridge distance x_0, velocity v_b, force F_{b}

[1] M. Demoucron, "On the control of virtual violins: Physical modelling and control of bowed string instruments," Ph.D. dissertation, IRCAM, Paris, 2008

Music performance with *Vivi, the Virtual Violinist*



Pedagogical inspiration for physical parameters

- Pedagogical inspiration
  - Suzuki violin book 1
  - Treat *Vivi* like a beginning student

Pedagogical inspiration for physical parameters

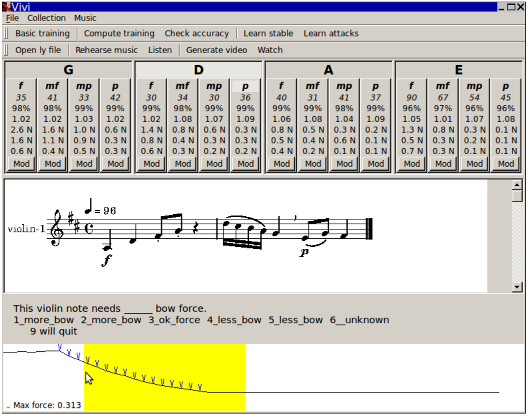
- Pedagogical inspiration
  - Suzuki violin book 1
  - Treat *Vivi* like a beginning student
- Most parameters can come from sheet music and pedagogy
  - String s, finger x_1: printed note
  - Bow-bridge distance x_0: dynamic "bow lanes" or "Kreisler Highway"
  - Bow velocity v_b: teacher saying "use half bow" and giving tempo

Pedagogical inspiration for physical parameters

- Pedagogical inspiration
  - Suzuki violin book 1
  - Treat *Vivi* like a beginning student
- Most parameters can come from sheet music and pedagogy
  - String s, finger x_1: printed note
  - Bow-bridge distance x_0: dynamic “bow lanes” or “Kreisler Highway”
  - Bow velocity v_b: teacher saying “use half bow” and giving tempo
- Bow force F_b from SVM classifiers
  1. not audible: needs a lot more bow force (example)
  2. “whispy”: needs a little more bow force (example)
  3. acceptable: no change (example)
  4. “harsh”: needs less bow force (example)
  5. not recognizable: needs much less bow force (example)

Interactive training

- Basic training: only 32 files (bad example)
- After interactive training: 203 files (good example)
- ≈ 4 hours to be fully trained (including calculations)

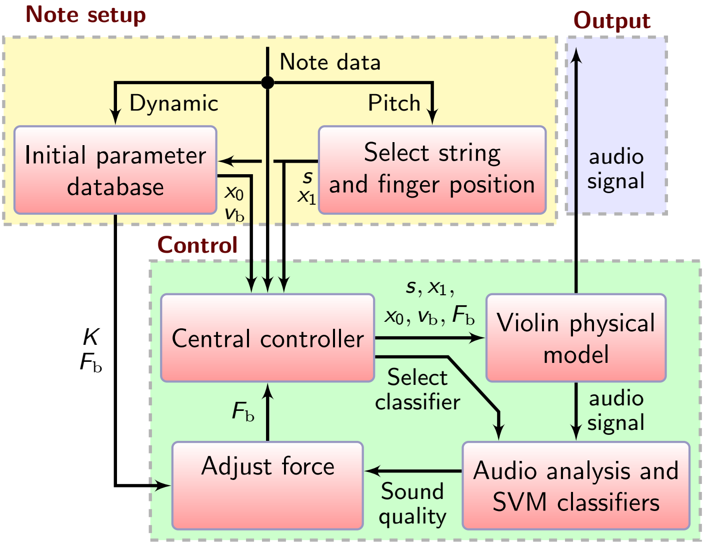


G				D				A				E			
f	*mf*	*mp*	*p*	*f*	*mf*	*mp*	*p*	*f*	*mf*	*mp*	*p*	*f*	*mf*	*mp*	*p*
35	*41*	*33*	*42*	*30*	*34*	*30*	*36*	*40*	*31*	*41*	*37*	*90*	*67*	*54*	*45*
98%	98%	99%	99%	99%	98%	99%	99%	99%	99%	98%	99%	96%	97%	96%	96%
1.02	1.02	1.03	1.02	1.02	1.08	1.07	1.09	1.06	1.08	1.04	1.09	1.05	1.01	1.07	1.08
2.6 N	1.6 N	1.0 N	0.6 N	1.4 N	0.8 N	0.6 N	0.3 N	0.8 N	0.5 N	0.3 N	0.2 N	1.3 N	0.8 N	0.3 N	0.1 N
1.6 N	1.1 N	0.9 N	0.3 N	0.8 N	0.4 N	0.3 N	0.3 N	0.5 N	0.4 N	0.6 N	0.1 N	0.5 N	0.3 N	0.2 N	0.1 N
0.6 N	0.4 N	0.5 N	0.3 N	0.6 N	0.3 N	0.2 N	0.2 N	0.4 N	0.2 N	0.1 N	0.1 N	0.7 N	0.3 N	0.1 N	0.1 N
Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod	Mod

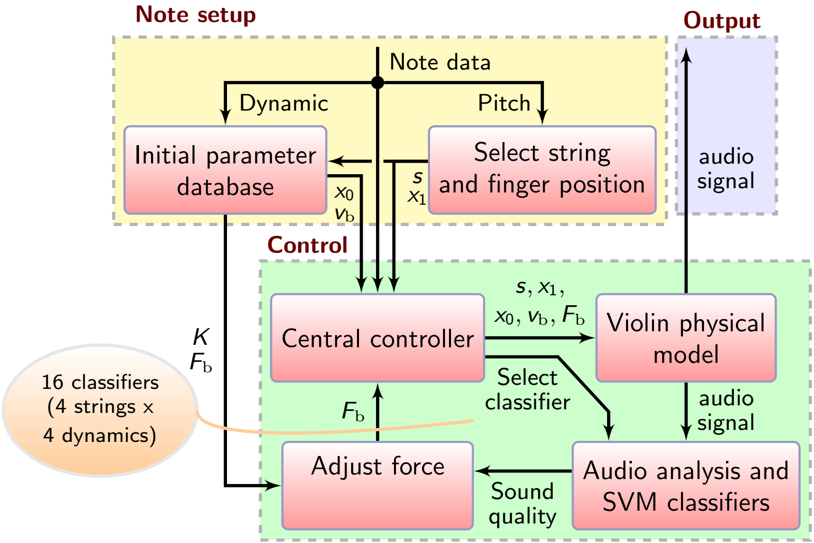
This violin note needs _____ bow force.
1_more_bow 2_more_bow 3_ok_force 4_less_bow 5_less_bow 6__unknown
9 will quit

Max force: 0.313

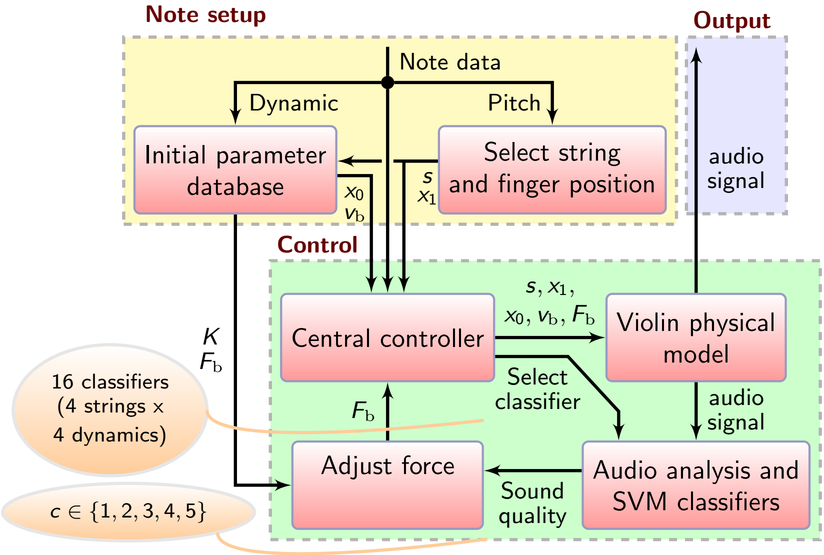
Intelligent feedback control of bow force



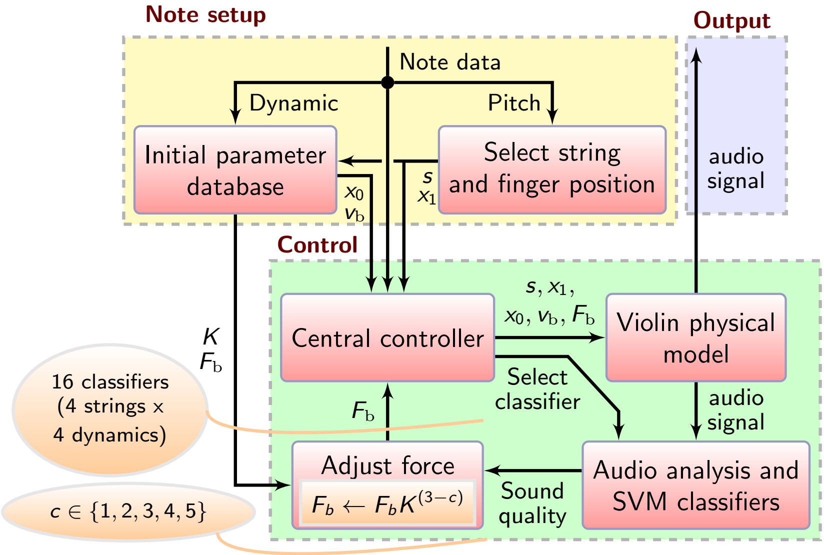
Intelligent feedback control of bow force



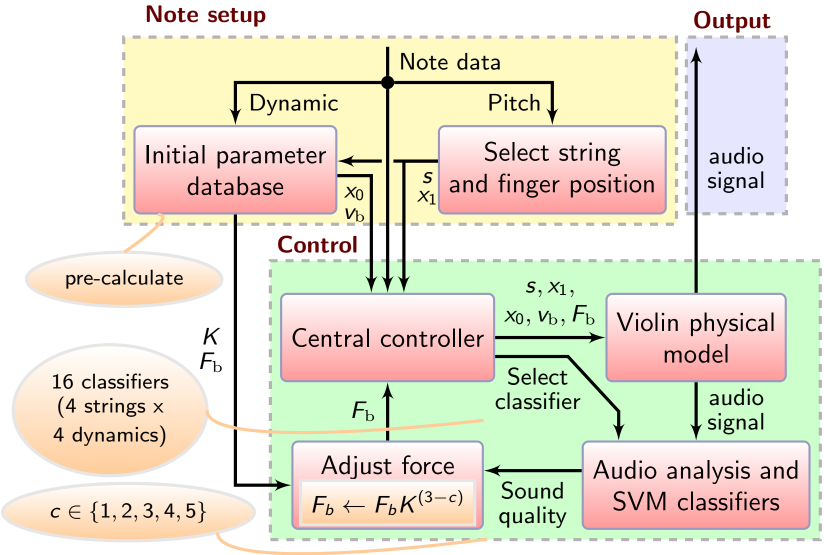
Intelligent feedback control of bow force



Intelligent feedback control of bow force


Note setup
Output
Note data
Dynamic
Pitch
Initial parameter database
Select string and finger position
x_0
v_b
s
x_1
audio signal
Control
Central controller
$s, x_1,$
$x_0, v_\mathrm{b}, F_\mathrm{b}$
Violin physical model
K
F_b
Select classifier
F_b
audio signal
16 classifiers (4 strings x 4 dynamics)
Adjust force
$F_b \leftarrow F_b K^{(3-c)}$
Sound quality
Audio analysis and SVM classifiers
$c \in \{1, 2, 3, 4, 5\}$

Intelligent feedback control of bow force



Automatically determining K

Cost of a candidate K

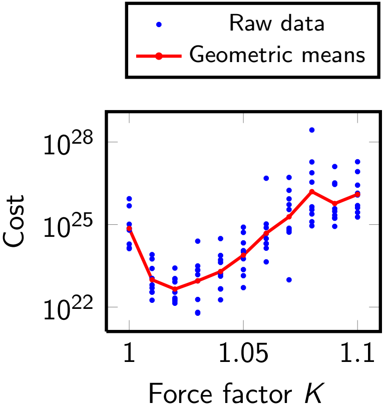
1. Play a simple musical pattern
2. Get list C of judgements c
3. Split C into sublists A_i based on c changing from below to above 3 (and vice versa)
4. Calculate
$$\text{cost} = \prod_{i}^{|A|} \sum_{c \in A_i} (3 - c)^2$$
5. Repeat 12 times and find the inter-quartile geometric mean

Automatically determining *K*

Cost of a candidate *K*

1. Play a simple musical pattern
2. Get list C of judgements c
3. Split C into sublists A_i based on c changing from below to above 3 (and vice versa)
4. Calculate
$$\mathrm{cost} = \prod_{i}^{|A|} \sum_{c \in A_i} (3 - c)^2$$
5. Repeat 12 times and find the inter-quartile geometric mean

Sample force factors of the D string played *mf*



Automatically determining initial bow force F_b

Cost of a candidate initial F_b

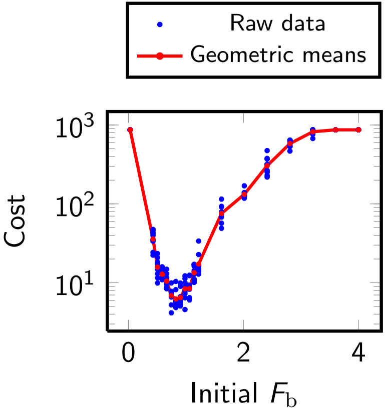
1. Play a simple musical pattern
2. Get list C of judgements c
3. Split of C into list A (note attack): attack is over when
$$0.5 > \frac{1}{N} \sum_{c \in L_N} (3 - c)^2$$
($L_N =$ previous N values of C)
4. Calculate
$$\mathrm{cost} = \sum_{c \in A} (3 - c)^2$$
5. Repeat 4 times and find the inter-quartile geometric mean

Automatically determining initial bow force F_b

Cost of a candidate initial F_b

1. Play a simple musical pattern
2. Get list C of judgements c
3. Split of C into list A (note attack): attack is over when
$$0.5 > \frac{1}{N} \sum_{c \in L_N} (3 - c)^2$$
($L_N =$ previous N values of C)
4. Calculate
$$\mathrm{cost} = \sum_{c \in A} (3 - c)^2$$
5. Repeat 4 times and find the inter-quartile geometric mean

Sample initial forces of the D string played *mf*, open string



Musical performance style

- Some notation is unambiguous
  - Slur = don't change bow direction

Musical performance style

- Some notation is unambiguous
  - Slur = don't change bow direction
- Some notation is open to stylistic interpretation
  - Staccato = shorter duration; but by how much?
  - For now, we use hard-coded values.
    "Make it sound like a book 1 Suzuki student."

Musical performance style

- Some notation is unambiguous
  - Slur = don't change bow direction
- Some notation is open to stylistic interpretation
  - Staccato = shorter duration; but by how much?
  - For now, we use hard-coded values.
    "Make it sound like a book 1 Suzuki student."
- Short-term: let humans specify/adjust stylistic interpretation
  - Most beginning music students (age 4–8 years) simply follow instructions from their teachers
  - Humans should give high-level judgements ($\approx$ 1–10 Hz); computers should do low-level processing (control parameters at $\approx$ 172 Hz)

Conclusion and future work

- *Vivi* performs music with a similar skill level to Suzuki violin students with 1 year of experience
    - Intelligent feedback control
    - SVM training requires $\approx$ 4 hours of feedback
    - Other parameters are determined from the SVM classifiers

Conclusion and future work

- *Vivi* performs music with a similar skill level to Suzuki violin students with 1 year of experience
  - Intelligent feedback control
  - SVM training requires ≈ 4 hours of feedback
  - Other parameters are determined from the SVM classifiers
- Open source, published under the GNU GPLv3+:
  - Artifastring ("artificial fast string"): violin physical modeling library, C++ with SWIG bindings
  - *Vivi, the Virtual Violinist*: sheet music → control loop → audio and video files, python and C++
  - `http://percival-music.ca/vivi.html`

Conclusion and future work

- *Vivi* performs music with a similar skill level to Suzuki violin students with 1 year of experience
  - Intelligent feedback control
  - SVM training requires ≈ 4 hours of feedback
  - Other parameters are determined from the SVM classifiers
- Open source, published under the GNU GPLv3+:
  - Artifastring ("artificial fast string"): violin physical modeling library, C++ with SWIG bindings
  - *Vivi, the Virtual Violinist*: sheet music → control loop → audio and video files, python and C++
  - `http://percival-music.ca/vivi.html`
- Future work: apply these techniques to "continuous excitation" instruments in STK (Synthesis ToolKit in C++)
  - Clarinet, saxophone, flute, brass