Available online at www.sciencedirect.com
ScienceDirect
Applied Acoustics 68 (2007) 576–593
applied acoustics
www.elsevier.com/locate/apacoust

Tuning pianos using reinforcement learning

Matthew Millard [a], Hamid R. Tizhoosh [b,*]

[a] *Piano Design Laboratory, Systems Design Engineering, University of Waterloo, 200 University Avenue West, Waterloo, ON, Canada N2L 3G1*
[b] *Pattern Analysis and Machine Intelligence Laboratory, Systems Design Engineering, University of Waterloo, 200 University Avenue West, Waterloo, ON, Canada N2L 3G1*

Received 30 January 2006; received in revised form 16 March 2006; accepted 22 March 2006
Available online 5 June 2006

Abstract

The tuning system of a piano has remained relatively unchanged since the instrument's inception. A piano's tuning system has been designed to be both inexpensive to manufacture and to preserve the tension and thus pitch of each string over long periods of time. This tuning system requires such a high degree of skill to manipulate that only trained professionals are able to tune pianos. This paper presents a novel adjustable impact tuning hammer and a reinforcement learning control system that may allow piano owners to tune their own pianos in the future.

Keywords: Piano tuning; Reinforcement learning; Impact tuning hammer; Automated piano tuning system

1. Introduction

Pianos have remained very difficult instruments to tune despite almost 300 years of technical innovation. Skilled piano tuners have been cherished by piano owners and professional pianists alike for hundreds of years. Innovations in tuning systems have been significant enough to affect the way composers write music. Improvements in piano tuning systems (known as *temperaments*) in the Baroque era gave pianists greater access to the piano's different keys resulting in compositions that were able to make use of the whole keyboard, rather than a few isolated, custom-tuned scales [1].

* Corresponding author. Tel.: +1 519 8884567x6751.
E-mail addresses: mjhmilla@uwaterloo.ca (M. Millard), tizhoosh@uwaterloo.ca (H.R. Tizhoosh).

0003-682X/$ - see front matter
doi:10.1016/j.apacoust.2006.03.006

Practice pianos should be tuned twice per year, and concert pianos before every concert. Presently the only way to reliably tune a piano is by making use of a professional piano tuner. The process of getting a piano tuned can be expensive and inconvenient depending on how often the piano needs to be tuned and on the availability of local professional piano tuners. A semi-automated piano tuning system will be presented in this paper. The novel system allows a non-skilled person to tune any acoustic piano using a minimum of equipment.

The rest of this paper is organized as follows. In Section 2 the current means of tuning will be explored and in Section 3 an overview of our approach to piano tuning will be presented. An investigation into the pitch change vs. impact will be shown in Section 4. Section 5 will explore the techniques that allow the reinforcement learning agent to perform well in this application. Experiments done with the agent will be presented in Section 6 followed by experimental error in Section 7. The paper will end with conclusions and some thoughts on future work in Section 8.

2. State of the art

The piano tuner's task is not an easy one, both from an aural perspective and from a mechanical perspective. From an aural point of view, it is not enough to tune each string on a piano independent of the others. Unlike ideal strings, real strings have a stretched harmonic spectrum that will make a piano sound out of tune if each string is tuned without consideration of the others. Thus, although two pianos can be tuned using the same template, each will then have to be adjusted again to account for each strings inharmonic stretch (known as *inharmonicity*). Choosing appropriate pitches for each string in order to make the piano sound its best is not a trivial task. A number of temperaments have been developed throughout the pianos history and to this day work is being done to improve them and to define new ways of evaluating their performance [2].

The tuning system on a piano consists of a metal pin (a 7 mm diameter pin is typical) driven into a dense block of plywood (known as a pin block) as shown in Fig. 1. This system has been designed to be both cheap to manufacture and to be very stable to prevent

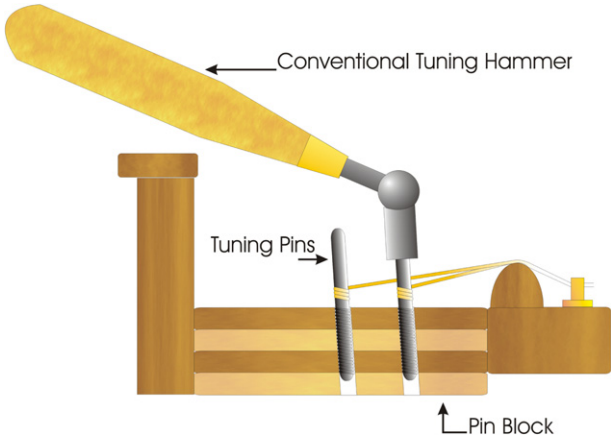

Fig. 1. Tuning pin in the pin block.

the strings from going out of tune. This tuning system design makes a piano tuner's mechanical task of adjusting the pitch of the strings quite daunting. Fine tuning a bass string (e.g. the lowest note, speaking length 2012 mm, 1.7 mm core wire) requires a tuning pin rotation resolution of about 0.16°; a treble string (e.g. the highest note with speaking length 50 mm, 0.8 mm diameter wire) requires a rotation resolution of only about 0.005°. These fine resolutions are very difficult to achieve given the great non-linear frictional forces that hold the pins in place: it takes 17 Nm (150 in-lbs[1]) to turn a typical tuning pin [3]. This task becomes even more difficult when the variability of torques required to turn each pin is taken into consideration: the frictional characteristics vary greatly from piano to piano with pin block design [4], tuning pin design [5] and within a piano over time and across different temperatures. Needless to say, control of these fine resolution rotations is an extremely difficult task to master.

Currently piano tuners address the difficulties of these complex aural and mechanical tasks with a combination of tools, natural ability, training and experience. To aid the piano tuner's aural task a number of companies have designed electronic tuning aids. These aids serve the function of accommodating for a pianos inharmonicity and telling the tuner how flat or sharp a given string is. To aid the piano tuner's mechanical task of finely adjusting pin position the tuner has a choice of two tools: a wrench shown in Fig. 1 and an impact wrench (tuning wrenches are known as hammers by piano tuners). An impact tuning hammer looks very similar to the tuning hammer pictured in Fig. 1 but the handle is weighted and able to rotate through 30°–55° [6] relative to the socket head. Both of these tools require years of experience to develop the tactile skills and strategies necessary to tune any piano [3,7]. These hammers also require the user to exert a great deal of force to use them: as much as 90 N (20 lbs) is needed to operate a tuning hammer, and as much as 1.4 Nm (12 in-lbs) of torque is needed to operate the impact tuning hammer. These hammers are far from ideal as they can cause repetitive strain injuries [7].

There have been many attempts to overcome the difficulty of tuning by altering the tuning mechanism or developing automated tuning systems. Donald Gilmore, an engineer, has made the most notable and recent attempt at developing a self-tuning piano [8]. Gilmore's system heats each piano string (which has been tuned sharp) individually to elongate the string, reducing its tension to bring it from a pre-sharpened pitch into tune. Gilmore's system is currently in development. There have been a few attempts to simply make the tuning mechanism easier to manipulate with more predictable frictional forces; the most recent development making use of tuning pins fitted with lock-nuts [9]. Mason and Hamlin also made use of a linear-screw tuning mechanism (coined the 'screw-stringer') briefly in their pianos over a century ago [10]. All pianos currently available use traditional tuning mechanisms. The automated tuning system presented in this paper is entirely unique because it does not require a conventional piano to be altered in any way.

3. PitchImpact: A next generation tuning system

A number of tools have been developed at the Piano Design Laboratory (University of Waterloo) to make it possible to automate the task of piano tuning. One of the tools is an

[1] Piano tuners and engineers working in the area commonly mix imperial units with metric units depending on the context: tuning pin diameters and string dimensions are usually in mm and tuning pin torques are usually in in-lbs. Imperial units will follow metric units in brackets when typically used.

acoustic spectrum analysis software that allows the tuner to directly accommodate for the string's inharmonic stretch rather than estimating the changes that need to be made as current tools do. The other tool is a new tuning hammer that makes it possible for anyone to make both fine and coarse pin adjustments without years of training and experience. PitchImpact is the complete system with the acoustic and mechanical tools married together with a control system that was implemented using reinforcement learning. The acoustic tool passes the error of the pitch to the reinforcement agent which then determines the appropriate impact setting needed to bring the string into correct tune. This process is repeated until the string is in tune as shown in Fig. 2.

The main topic of this paper is the reinforcement learning system that combines the aural and mechanical tool together to make an automated piano tuning system. For the purposes of this paper it is assumed that the process of choosing the appropriate frequency target given the tuning temperament and the strings inharmonicity has already been completed. The system described in this paper will tune each test string to a predefined frequency.

To understand the problem the reinforcement learning agent faces, it is necessary to understand some of the design details of the new impact hammer shown in Fig. 3. When the tuning hammer was being designed, it was found that an impact machine was the only practical machine that could make both fine and coarse tuning peg adjustments. The new impact hammer allows the user to apply a repeatable impulse torque to the tuning pin. The impulse torque lasts for a very short amount of time (on the order of 10 ms). If the magnitude of the impulse torque is high enough it will exceed the forces holding the pin in place and move the pin, changing the pitch of the string. The combination of high torque levels over short periods of time make it possible to adjust a tuning pin with the required resolution of thousandths of degrees.

To operate the impact hammer the user must wind the top of the hammer to a rotation specified by the reinforcement learning agent. Rotating the top stores energy in the linear torsion spring that connects the top of the hammer to the socket and the tuning peg. When user releases the top it spins until the vertical pin on the top contacts the dial indicator. At

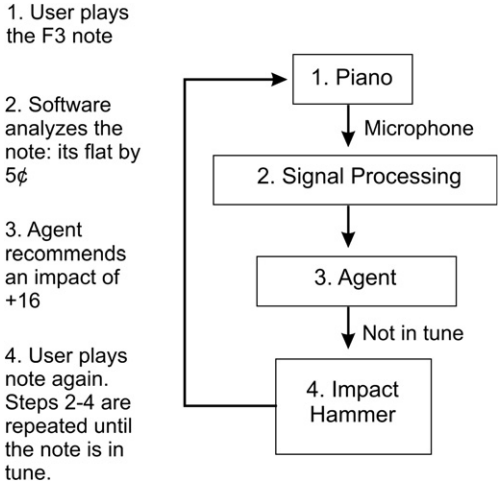

Fig. 2. System diagram.

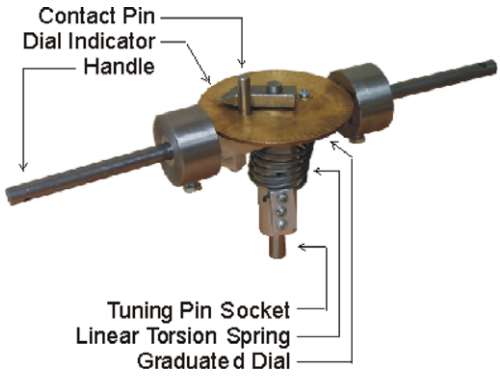

Fig. 3. New repeatable impact tuning hammer.

this moment the top is decelerated very quickly and in doing so applies an impulse torque to the tuning peg.

The peak forces and duration of the impact is controlled by the amount of energy stored in the torsion spring that connects the hammer top the base. Repeatable impacts are made possible with a dial on the top of the hammer that lets the user know how far the top has been rotated and thus how much energy is stored in the spring. The dial has 30 sharpening and 30 flattening impact levels spaced every 5°. These gradations are labelled +6 to +36 in the pitch sharpening direction and −6 to −36 in the pitch flattening direction.

Knowledge of the pins frictional characteristics are needed in order to determine the impact size needed to bring the string into tune. As was previously mentioned, tuning pin friction characteristics vary a lot within even one piano, and even more so when comparing pianos with different pin block designs. These friction properties also vary with time because the wood fibres in contact with the tuning pin wear.

Presently, no piano tuning-pin friction models nor impact models have been developed that might be able to predict how much the pitch of the string will change after a given impact. Thus the dynamics of impact tuning cannot be realistically simulated. In order to solve this highly non-linear time-variant problem, online learning is required. Reinforcement learning was chosen to address this highly dynamic and time variant problem because it is presently the only machine learning technique that can learn *online* without a mathematical model.

Reinforcement learning has the constraint that it must learn long enough to visit all states. Since the agent's environment cannot be simulated, the agent has to train with a real piano. This means that a person has to follow the tuning system's commands and play the string when needed, and operate the impact hammer as instructed. Because these operations require a human and they take time (20 s to play a string and apply an impact) it is impractical to allow the reinforcement agent to train for extended periods of time. Ideally the agent would be able to tune any string on any piano within 20 or so impacts with some a priori knowledge and gradually reach a professional performance level. In our Piano Lab, we can tune the strings from being a semi-tone out of tune to being in tune (within a cent) usually within 6–10 impacts. A conservative estimate of a professional performance was estimated from this result to be 5–8 impacts.

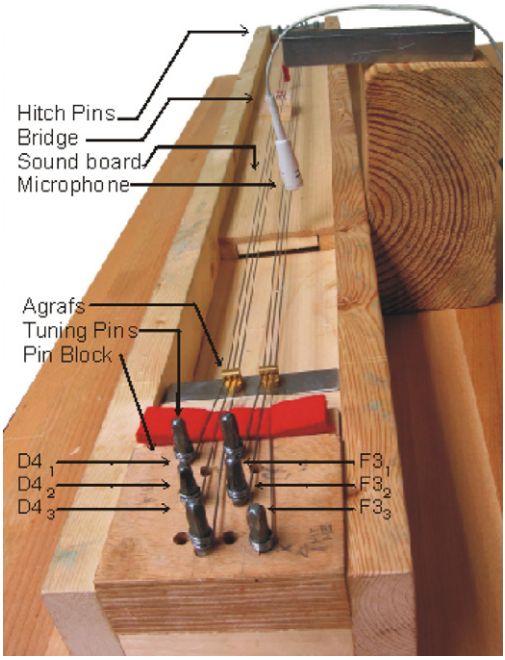

Fig. 4. The test piano *hexachord*.

4. Impact tuning characteristics

A study of the relationship between impact size and pitch change was needed to see if developing a rapidly learning highly flexible reinforcement agent was even possible. A real piano was not available for study purposes and so a test section that would mimic a real piano was constructed. The test section[2] (shown in Fig. 4) has six strings and so was named the hexachord.

The hexachord was built using the same materials, dimensions and techniques that would be used to build a real piano to ensure that its tuning dynamics and sound characteristics would match those of a real piano. The hexachord can play two different notes with its six strings: 3 F3 notes that have a fundamental frequency of 174.61 Hz and 3 D4 notes that have a fundamental frequency of 293.66 Hz. A real piano also has three strings for these two notes.

One key discrepancy between a piano and the hexachord could not be avoided: the hexachord strings had to be plucked by hand. Plucking a string by hand does not produce the same wave pattern in the string and thus pitch as striking it with a felt piano hammer.

Table 1 shows the stick torque of each tuning pin which must be exceeded to raise and lower the pitch on each string. The torques exhibit a high degree of variability from string to string and yet remain in agreement with the tuning torque ranges found in conventional pianos [3].

[2] Prof. Birkett, the head of the Piano Design Lab at the University of Waterloo and a skilled piano restorer built the test section.

Table 1
Sharpening and flattening stick torques

String	Sharpening torque Nm (in-lbs)	Flattening torque Nm (in-lbs)
$F3_1$	16.3 (144)	12.6 (112)
$F3_2$	9.97 (88)	6.11 (54)
$F3_3$	19.5 (172)	16.9 (149)
$D4_1$	13.9 (123)	8.27 (73)
$D4_2$	21.2 (187)	19.0 (168)
$D4_3$	16.4 (145)	9.51 (84)

A plot of pitch change vs. impact for the hexachords 6 strings over the entire range of impact settings available on the hammer is shown in Fig. 5. A number of polynomial curve fits were applied separately to the positive and negative sides of the curves. It was found that quadratic polynomials provided a good fit, accounting for over 90% of the data variation for all curves.

The vast majority of real tuning is on the order of 0.05–0.1% of the string's nominal frequency. Piano tuners commonly refer to a change of $\frac{1}{100}$ of a semi-tone in frequency as 1¢. In the commonly used equal temperament this amounts to a proportional change of $\frac{1}{100}\sqrt[12]{2}$ (approximately 0.0595%) of the nominal frequency. The small change of 1–2¢ typical of piano tuning was only observable over a few of the impact settings. Fig. 5 has relatively large range of pitch changes from −60¢ to 30¢. The quadratic relationship between pitch and impact size found over a large range of pitch changes was not easily observed for fine pitch changes. The same 6 strings were subjected to 12 fine impacts to examine the relationship between fine pitch change and impact setting. The variation in pitch change was too great to extract any relationship from only 12 data points. It was assumed that there was still a quadratic relationship between pitch change and impact size but that it was being overwhelmed by variations in the friction properties between the tuning pin and the pin block.

String $F3_1$ was subjected to 36 equal fine impacts to study the variations in pitch change for a single fine impact. The 36 impacts were done in 6 trials of 6 impacts, each trial began with the string in tune. The average lag one autocorrelation coefficient of the pitch changes

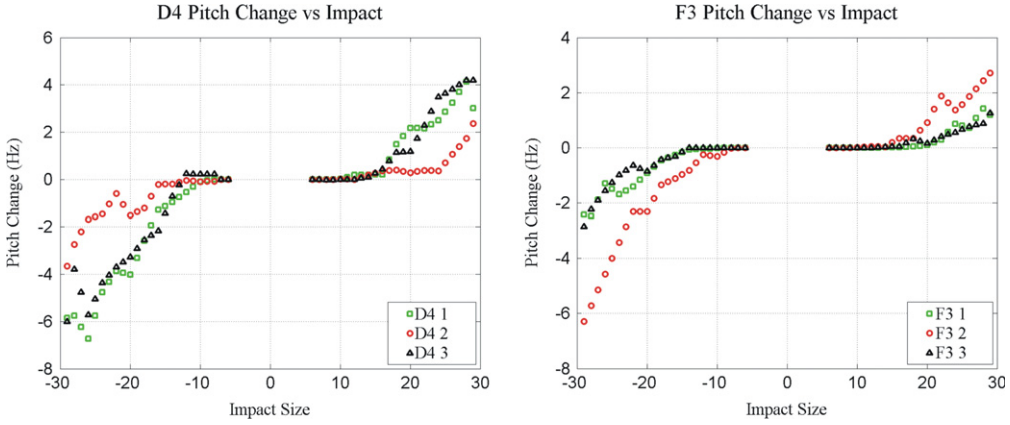

Fig. 5. Change in pitch vs. impact size.

was not statistically significant, indicating that the variations in pitch change (and thus friction properties) are likely random. Parzen windows was used to extract an estimated distribution for pitch change for a fine impact as shown in Fig. 6. It should be noted that on occasion the #16 impact used in the experiment would provide a fine pitch change of 1¢ range (as seen in the first hump), and sometimes change the pitch coarsely by a full 10¢.

These plots provide a number of very useful insights which have corresponding interpretations:

(1) *Pitch change is roughly quadratic with impact size.* The change of pin position ϕ and pitch is directly related to the energy of the impact. The energy of the impact is equal to the potential energy α stored in the torsion spring of stiffness k that is rotated through angle θ. Frictional energy lost while the hammer top rotates is assumed to be negligible.

$$\alpha = \frac{1}{2}k\theta^2 \tag{1}$$

Upon impact it is assumed that the vast majority of the hammer tops kinetic energy is dissipated by rotating the tuning pin by ϕ radians, against an average torque L which consumes β joules of energy.

$$\beta = L\phi \tag{2}$$

The average torque L resisting the pins motion is created by tension T of the string and the force of friction F between the pin and the bin block acting on a tuning pin of radius r.

$$L = (F + T)r \tag{3}$$

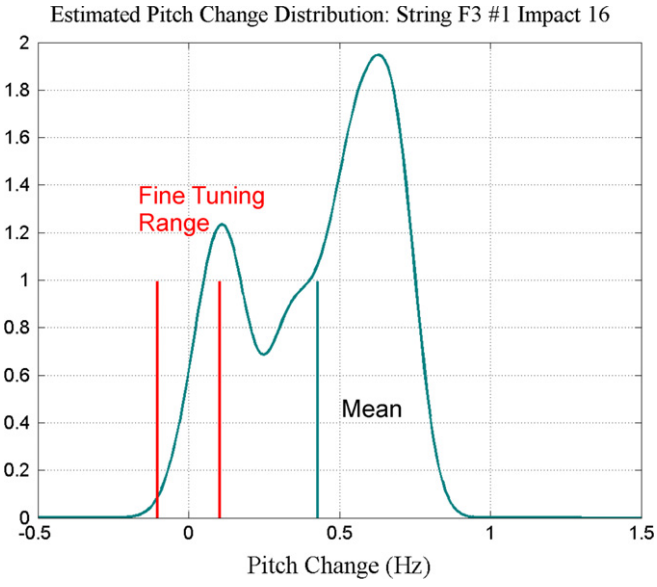

Fig. 6. Estimated pitch change distribution.

Setting Eq. (1) equal to Eq. (2) and substituting for L yields an expression for the rotation ϕ of the tuning pin given the initial rotation of the impact hammer.

$$\phi \approx \frac{1}{2(F+T)r} k\theta^2 \tag{4}$$

A diagram of the impact hammer with the physical variables used in the following derivation can be found in Fig. 7. Over a large range of frequencies the change in pitch of a string is proportional to the root of the strings tension. For fine frequency adjustments typical of piano tuning, the pitch change (whether measured in Hz or cents) $\Delta\mathscr{P}$ is linear with the change in pin position ϕ. Because T and F change with each rotation ϕ so does the relationship between θ and $\Delta\mathscr{P}$.

$$\Delta\mathscr{P} \propto \frac{1}{2(F+T)r} k\theta^2 \tag{5}$$

(2) *Below a certain impact setting $\mathscr{I}_M$, no pitch change occurs*. Below a certain impact size the peak forces generated during the impact do not exceed the force of friction acting on the tuning pin, and the pin does not rotate.
(3) *Fine impacts lead to fine pitch changes that have a great deal of variation*. When the pin is rotating the slight amounts needed to perform fine tuning, it moves between the stick and slip regions of friction which is ill-defined; variations are expected.

Interpretation 3 requires more research to substantiate it. Insights 1 and 2 were instrumental in developing a policy for the reinforcement agent which will be discussed in subsequent sections.

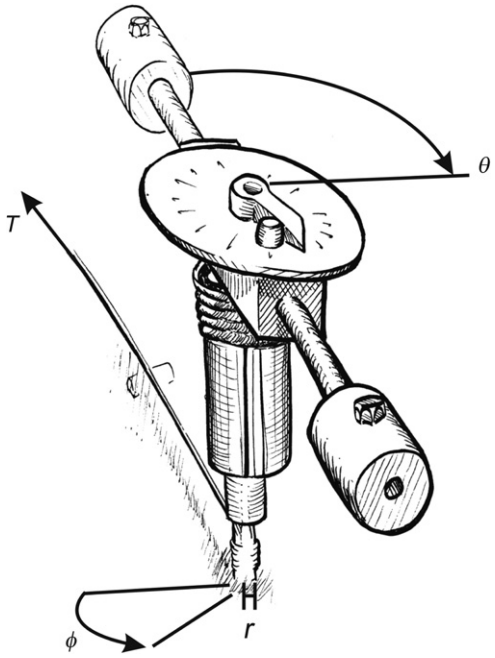

Fig. 7. Diagram of hammer and tuning pin with physical variables used in the derivation of Eq. 5.

5. Reinforcement learning configuration

Reinforcement learning is based on the idea that an agent learns by interacting with its environment [11,12]. It allows software agents to automatically determine the ideal behavior within a specific context that maximizes performance. Several components constitute the general idea behind reinforcement learning. The agent, which is the decision maker of the process, attempts an action that is recognized by the environment. The agent receives reward or punishment from its environment depending on the action taken. The agent also receives information concerning the state of the environment. The agent acquires knowledge of the actions that generate rewards and eventually learns to perform the actions that are the most rewarding in order to meet a certain goal relating to the state of the environment.

Q-Learning is a well-established reinforcement learning algorithm, and in its simplest form, state-action values are updated by the following rule [12]:

$$\mathcal{Q}(s_t, a_t) \leftarrow \mathcal{Q}(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a \mathcal{Q}(s_{t+1}, a) - \mathcal{Q}(s_t, a_t)], \tag{6}$$

where $\mathcal{Q}(s_t, a_t)$ is the learned value function for a given state s_t and action a_t at time t. Eq. 6 also includes parameters such as the learning rate α, the discount factor γ, and the reward value r. The Q-learning algorithm is presented in Table 2 where s is state, a is action, and s' is the next state.

Tuning a piano string by impact fits a typical reinforcement learning very nicely [11]:

- The state of the system is defined by the pitch error, which can be represented in a number of discrete steps.
- Every time an impact is applied to the tuning peg it will cause the pitch error, and thus system state to move from state $\mathscr{I}$ to state $\mathscr{J}$ with a probability $\mathscr{P}_{\mathscr{I}\mathscr{J}}$.
- The current state and previous impact are always known making this process observable.

This process is time-variant because the friction characteristics appear to vary unpredictably with every impact, and do vary as the piano ages.

A reinforcement learning agent was thus developed modelling the impact tuning process as an observable Markov process. As there were considerable challenges to making this system learn quickly in a limited time frame, the model was reduced from a second order Markov model to a first order Markov model. Every single string has its own Markov model of the following design:

Table 2
Q-Learning algorithm [12]

Initialize $\mathcal{Q}(s, a)$ arbitrary
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
  Choose a from s using policy derived from $\mathcal{Q}$ (e.g., ϵ-greedy)
    Take action a, observe r,s'
    $\mathcal{Q}(s, a) \leftarrow \mathcal{Q}(s, a) + \alpha[r + \gamma \max_{a'} \mathcal{Q}(s', a') - \mathcal{Q}(s, a)]$
    $s \leftarrow s'$;
until s is terminal

Table 3
State pitch tolerances

State	Tolerance (¢)	State	Tolerance (¢)
$\mathscr{S}_{-10}$	$-\infty$ to -74	$\mathscr{S}_{10}$	74–∞
$\mathscr{S}_{-9}$	-74 to -58	$\mathscr{S}_{9}$	58–74
$\mathscr{S}_{-8}$	-58 to -42	$\mathscr{S}_{8}$	42–58
$\mathscr{S}_{-7}$	-42 to -24	$\mathscr{S}_{7}$	24–42
$\mathscr{S}_{-6}$	-24 to -14.8	$\mathscr{S}_{6}$	14.8–24
$\mathscr{S}_{-5}$	-14.8 to -11.4	$\mathscr{S}_{5}$	11.4–14.8
$\mathscr{S}_{-4}$	-11.4 to -8.2	$\mathscr{S}_{4}$	8.2–11.4
$\mathscr{S}_{-3}$	-8.2 to -4.8	$\mathscr{S}_{3}$	4.8–8.2
$\mathscr{S}_{-2}$	-4.8 to -2.4	$\mathscr{S}_{2}$	2.4–4.8
$\mathscr{S}_{-1}$	-2.4 to -0.8	$\mathscr{S}_{1}$	0.8–2.4
$\mathscr{S}_{0}$	-0.8 to 0.8		

- The continuous valued pitch error ε_{P} is discretized into 1 of 21 different states $\mathscr{S}_{\mathrm{P}}: \mathscr{S}_{-10}, \ldots, \mathscr{S}_{0}, \mathscr{S}_{1}, \ldots \mathscr{S}_{10}$. The tolerances for each state were derived to provide a conventional fine pitch error resolution, and adequate coarse pitch error resolution for the agent to function. Table 3 shows the pitch ranges for all possible states.
- Every state has a transition matrix $\mathscr{R}_{\mathscr{P}}$ that has all possible impact settings $\mathscr{I}$ as the rows and the next state values $\mathscr{S}_{i+1}$ in the columns. Transition matrix $\mathscr{R}_{\mathscr{P}}$ contains a record of the rewards the agent received when the agent was in state $\mathscr{S}_{\mathscr{I}}$ and chose impact setting $\mathscr{I}$. These rewards are stored in the state that impact setting $\mathscr{I}$ caused.

5.1. Policy development

The simplest action policy involved in bringing a string into tune is a greedy policy[3]: it is most desirable to tune the string as expediently as possible. It was naively assumed that fine and coarse pitch adjustments could be made with equal precision, in which case a greedy policy would perform quite well.

The heart of solving this problem lay in making the agent behave reasonably in an extremely short period of time with very little data. This was accomplished by choosing a policy that exploited the a-priori knowledge gained about this process from earlier testing to best use the rewards the agent gained from its past experience.

5.1.1. Collective learning

An agent that makes no assumptions about the environment must visit every state multiple times before it starts exploiting its knowledge. Every state has 30 different impact settings associated with it and there are 21 different states per string in this system . Assuming that at least 3 visits are necessary, then a traditional agent would have to apply an estimated $21 \times 30 \times 3 = 1890$ impacts to a string before it would behave reasonably. Because this system cannot be simulated and has to be operated by a person, an agent that makes no assumptions about the environment is clearly impractical: if impacts were applied once

[3] A 'policy' refers to a strategy used to make decisions, in this case by the reinforcement learning agent. The term 'greedy' is often used to describe an approach that takes the maximum value ('best' decision) at a given point in time (or any other domain for that matter) without consideration of future consequences of that decision.

every 10 s it would take 5.25 h of labour to train the agent to learn how to tune a single string; it would take in excess of 1155 h for the agent to learn how to tune one piano.

Fortunately Eq. 5 and insight 2 define a rough relationship between pitch change and impact setting. This relationship makes it possible for the agent to make impact predictions within a state even if it has never used that particular impact setting before. If the agent is confronted with a state $\mathscr{S}_{\mathscr{I}}$ that has never been visited, this relationship allows it to use data from another state $\mathscr{S}_{\mathscr{J}}$ to estimate the best impact setting.

Currently all data stored in a state transition matrix is used to predict what the best output would be to bring the string into tune on the next impact. Each piece of data is taken into account with the weighted sum. The weights were derived from two simple assumptions:

(1) Local data is more precise: Past impacts that brought the environment closest to the ideal state will yield better predictions than impacts that put the environment far away from the ideal state.
(2) Heavily rewarded data is more accurate: Impacts that have the highest rewards offer the most accurate sources of information for future output predictions.

Eq. 5, insight 2 and assumptions 1 and 2 were used to derive a policy that could take past rewards in a transition matrix and use it to predict the best impact dial setting $\mathscr{I}$ for state $\mathscr{S}_{\mathrm{P}}$. The ideal change in pitch $\mathscr{P}_{\mathrm{ideal}}$ is the difference between the average pitch of the ideal state $\overline{P(\mathscr{S}_{\mathrm{ideal}})}$ and the current pitch P_i of the string:

$$\Delta\mathscr{P}_{\mathrm{ideal}} = \mathscr{P}_i - \overline{P(\mathscr{S}_{\mathrm{ideal}})} \tag{7}$$

Piano tuners commonly tune a string slightly sharp [3] and then pound the strings key hard to settle the string into the ideal pitch. This technique is used to ensure that the pitch of the string stays stable for as long a time as possible. Thus the ideal state for the string to be in is $\mathscr{S}_1$ because it is slightly sharp compared to $\mathscr{S}_0$ which matches the perfect pitch for the string.

Eqs. (8)–(14) define all of the relationships used to predict the best output in state $\mathscr{S}_i$ to reach state $\mathscr{S}_1$:

Eq. 8 is a measure of the squared pitch difference $\delta_{\mathscr{S}}$ an index in transition matrix $\mathscr{R}_{\mathscr{P}}$ is from the ideal pitch:

$$\delta(\mathscr{S}) = \left(\frac{1}{\overline{P(\mathscr{S}_{\mathrm{ideal}})} - \overline{\mathscr{P}(S)}}\right)^2. \tag{8}$$

Eq. 9 is the total sum of the squared pitch difference $\mathscr{S}_\delta$ for transition matrix $\mathscr{R}_{\mathscr{P}}$:

$$\mathscr{S}_\delta = \sum_{\mathscr{I}=6}^{36} \sum_{\mathscr{S}=-10}^{10} \delta_{I,S} \left\lceil \frac{|\mathscr{R}(\mathscr{I},\mathscr{S})|}{|\mathscr{R}(\mathscr{I},\mathscr{S})+1|} \right\rceil. \tag{9}$$

Eq. 10 is the total reward value $\mathscr{S}_\eta$ a state has been granted:

$$\mathscr{S}_\eta = \sum_{\mathscr{I}=6}^{36} \sum_{S=-10}^{10} \mathscr{R}(\mathscr{I},\mathscr{S}). \tag{10}$$

Eq. 11 is the average state transition $\overline{\mathscr{S}(\mathscr{I})}$ for a given impact $\mathscr{I}$:

$$\overline{\mathscr{S}(I)} = \frac{1}{\sum_{\mathscr{S}=-10}^{10} \mathscr{R}(\mathscr{I},\mathscr{S})} \sum_{\mathscr{S}=-10}^{10} \mathscr{R}(\mathscr{I},\mathscr{S})\mathscr{S}. \tag{11}$$

Eq. 12 is the predicted output for entry $\mathscr{I},\mathscr{S}$ of transition matrix $\mathscr{R}_{\mathscr{P}}$:

$$\mathscr{I}(\mathscr{I},\mathscr{S}) = \frac{\Delta\mathscr{P}_{\text{ideal}}}{\overline{P(S)}} \sqrt{\mathscr{I} - \mathscr{I}_{\mathscr{M}}}. \tag{12}$$

Eq. 13 is the best impact prediction that will bring the string to the ideal pitch for transition matrix $\mathscr{R}_{\mathscr{P}}$:

$$\mathscr{I}_{i+1} = \frac{1}{\mathscr{S}_\eta \mathscr{S}_\delta} \sum_{\mathscr{I}=6}^{36} \sum_{S=-10}^{10} \mathscr{R}(\mathscr{I},\mathscr{S})\delta_S \mathscr{I}(\mathscr{I},\mathscr{S}). \tag{13}$$

It should be noted that impacts that did not cause a state transition are ignored in order to prevent the denominator of Eq. 13 from going to 0. Eq. 14 is the expected standard deviation in pitch change that is associated with impact prediction $\mathscr{I}_{i+1}$.

$$\mathscr{P}_{\text{var}} = \frac{1}{\mathscr{S}_\eta \mathscr{S}_\delta} \sum_{\mathscr{I}=6}^{36} \sum_{S=-10}^{10} |\delta(\mathscr{S})\mathscr{R}(\mathscr{I},\mathscr{S})(\overline{P(\mathscr{S})} - \overline{\mathscr{S}(I)})|. \tag{14}$$

All sums over $\mathscr{I}$ go from 6 to 36 because the sign of the impact setting is a function of the state $\mathscr{S}_{\mathscr{P}}$ only: states $\mathscr{S}_{-10},\ldots,\mathscr{S}_0$ are given + (sharpening) impacts and states $\mathscr{S}_2,\ldots\mathscr{S}_{10}$ are given flattening impacts. State $\mathscr{S}_1$ is the terminating state as previously mentioned. The state transition table $\mathscr{R}_{\text{P}}$ of the current state $\mathscr{S}_{\text{P}}$ is used in concert with Eqs. 13 and 15 to calculate the best output based on the rewards from the transition matrix along with an associated standard deviation. The output space is explored by modifying the predicted output $\mathscr{I}_{i+1}$ randomly by a random amount up to a maximum of one standard deviation of the impact prediction $\mathscr{I}_{\text{var}}$:

$$\mathscr{I}_{\text{var}} = \mathscr{P}_{\text{var}} \frac{\mathscr{I}_{i+1}}{\Delta\mathscr{P}_{\text{ideal}}}. \tag{15}$$

The $\mathscr{I}_{i+1}$ impact prediction

$$\mathscr{I}_{i+1} = \mathscr{I}_{i+1} + \Delta\mathscr{I}, \tag{16}$$

with

$$\Delta\mathscr{I} = \text{sign}(\xi_3 - 0.5)\mathscr{I}_{\text{var}}\xi_2 \frac{\left\lfloor \frac{\xi_1}{\gamma} \right\rfloor}{\frac{\xi_1}{\gamma}}, \tag{17}$$

is modified when a 0 to 1 random number generator returns a value $\xi_i \sim \mathscr{U}(0,1)$ greater than a threshold γ:

$$\gamma = e^{\frac{50+\mathscr{T}}{\mathscr{T}}}. \tag{18}$$

The threshold γ is an exponentially decreasing number defined such that when the number of impacts $\mathscr{T}$ applied to a given string reaches 100 (or roughly 10 tunings) the probability of randomly altering the impact prediction is approximately 10%. It has been assumed that the agent will be able to tune the string at or near a professional's level after 100 impacts. If ξ exceeds γ the impact prediction will be added to a randomly chosen number between 0

and $\mathscr{I}_{\text{var}}$ with a randomly chosen sign. Effort has gone into making controlled explorations in order to search in areas where the ideal impact is expected to be and to prevent wild searches that would slow the agent's current tuning progress.

If the agent arrives at a state $\mathscr{S}_n$ that it has never visited, then it would search out the nearest visited state $\mathscr{S}_{\text{v}}$. The impact setting $\mathscr{I}_{\text{v}}$ that will transition the environment from the nearby state $\mathscr{S}_{\text{v}}$ to $\mathscr{S}_1$ is calculated. Eq. 19 is used to translate the predicted impact value $\mathscr{I}_{\text{v}}$ from the nearby state to the current state:

$$\mathscr{I}_{i+1,n} = \frac{\Delta\mathscr{P}_{\text{ideal,v}}}{\Delta\mathscr{P}_{\text{ideal},n}} \sqrt{\mathscr{I}_{\text{v}} - \mathscr{I}_{\mathscr{M}}}. \tag{19}$$

If the agent arrives at a string that it has never tuned before it uses the pre-initialized values in the transition table that will bring the next state closest to $\mathscr{S}_1$. These pre-initialized values were generated using the data shown in Fig. 5 (page 8). In a real application a general template of the 'typical' tuning pin would be used to initialize every transition matrix. These pre-initialized values have been found to be poor predictors at best. These initialized values are ignored as soon as two impacts have been applied to the string.

5.1.2. Starvation prevention

The policy mentioned in Section 5.1 is incomplete.[4] If Eq. 13 predicts an impact value that is below the ever changing $\mathscr{I}_{\mathscr{M}}$ the state of the string will not change. Entries in the state transition table that do not indicate a state change are ignored because otherwise the denominator of Eq. 12 goes to zero. Thus the transition table remains the same as it was and Eq. 13 will produce the same impact value that is too small to change the environment. Without any other intervention, the random explorations are the only mechanism available to break this cycle. If standard deviation of the impact prediction is low, the agent may never choose an impact value that will change the environment.

This problem was solved using an ad hoc approach. If the previous impact did not change the state of the environment the agent picks the next biggest impact setting. When an impact setting that changes the state of the environment is reached the agent adjusts $\mathscr{I}_{\mathscr{M}}$ to this new impact value. To prevent data from impacts that are much smaller than $\mathscr{I}_{\mathscr{M}}$ from swaying Eq. 12 the agent then cleans all of the irrelevant (empirically determined to be impact values below $\mathscr{I}_{\mathscr{M}} - 2$) data from the state transition table. The value of $\mathscr{I}_{\mathscr{M}}$ is shared amongst all of the states to improve each states (especially the fine tuning states) predictions.

This is certainly not the ideal solution, as there is no corresponding update for $\mathscr{I}_{\mathscr{M}}$ should it be too low: if $\mathscr{I}_{\mathscr{M}}$ is too low then all of the predictions for $\mathscr{I}_{i+1}$ will be too high and $\mathscr{I}_{\mathscr{M}}$ will remain unchanged.

5.1.3. Parzen window rewards

The impact prediction routines described in Section 5.1 will work best with well defined distributions. In particular the estimate for $\mathscr{I}_{\text{var}}$ will not be realistic until a state has been visited a number of times. To help speed up the process rewards, have been given out using a Parzen windows technique [13]:

[4] The term 'starvation' is commonly used to describe a case where the algorithm is stuck and will never arrive at another state.

$$\mathscr{R}(\mathscr{I},\mathscr{S}) = \frac{1}{|\overline{P(\mathscr{S}_i)} - \mathscr{P}| + 1}, \tag{20}$$

$$\mathscr{R}(\mathscr{I},\mathscr{S}-1) = \frac{1}{|\overline{P(\mathscr{S}_{i-1})} - \mathscr{P}|}, \tag{21}$$

$$\mathscr{R}(\mathscr{I},\mathscr{S}+1) = \frac{1}{|\overline{P(\mathscr{S}_{i+1})} - \mathscr{P}|}. \tag{22}$$

Currently the window is only three states in size, and the reward is inversely proportional to the distance in Hz between the average error frequency of the state and the current error frequency ε_P of the string. Data shown in Fig. 5 (page 8) suggested that this window size and shape would realistically interpolate data stored in the transition matrix $\mathscr{R}(\mathscr{I},\mathscr{S})$.

6. Experiments

The system was tested by tuning the hexachord's six strings. Conventional fine and coarse tuning tolerances are 1¢ and 3¢ of the strings nominal value respectively. The F3 and D4 notes were brought to 61¢ and 51¢ Hz flat of their ideal value before the agent was allowed to tune the strings. The agent was not allowed to stop tuning the F2 and D4 strings until they entered state $\mathscr{S}_1$, making them 0.8¢ to 2.4¢ sharp. The notes were severely flattened to see how the agent performed through very coarse to very fine tuning to identify areas for policy improvement.

Table 4 shows the number of impacts needed to tune each string. Each string took between 30 s to just over 3 min to tune with impacts being applied every 10 s. Preliminary results show that this performance is comparable and occasionally exceeds that of a trained amateur piano tuner using the impact hammer, and greatly exceeds the performance of an amateur piano tuner using a traditional tuning hammer. This system has yet to be tested using untrained amateurs in a more realistic tuning setting due to facility and equipment limitations.

Plots of the state transitions of the string with the policy chosen (initialization points, prediction routine or anti-starvation routine) were examined to evaluate the agents performance. An example of such a plot for string $D4_3$ is shown in Fig. 8. This examination revealed that the collective learning routine is being used to make the majority of the

Table 4
Tuning Performance

String	# Impacts required
$F3_1$	10
$F3_2$	8
$F3_3$	3
$D4_1$	14
$D4_2$	19
$D4_3$	10
Agent's policy	**Use (%)**
Pre-initialized values	16
Collective learning	55
Starvation prevention	29

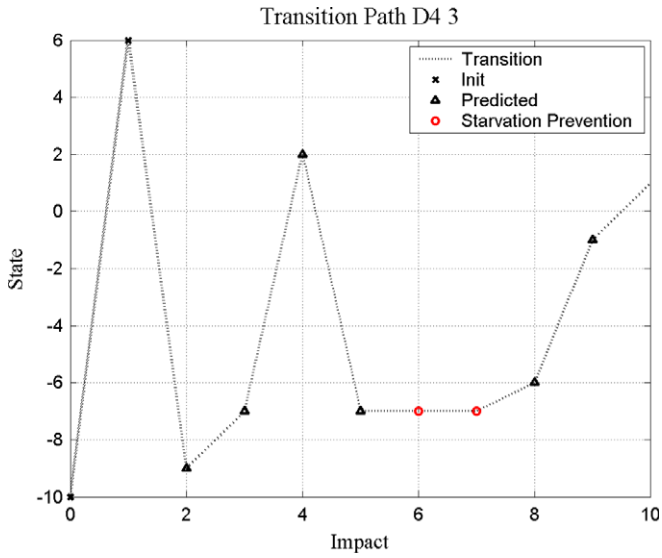

Fig. 8. State transition path for string $D4_3$.

impact predictions and that the performance of the prediction routine improves immediately after the starvation prevention routine has run, and then gradually degrades. The decrease in wild state changes after the starvation prevention routine has run is interpreted as being improved performance.

This improvement can be explained by noting that after the starvation prevention routine has been used the value of $\mathscr{I}_{\mathscr{M}}$ is updated, and is closest to its true value at that instant. The fact that the collective learning predictions improve and then get steadily worse after the starvation prevention routine is used indicates that the agent's performance could be improved if regular, accurate estimates of $\mathscr{I}_{\mathscr{M}}$ could be made. Overall the performance of the system was satisfactory as it was able to tune all of the strings within 20 impacts starting out with a few pre-initialized values in its state transition matrix.

7. Experimental error

During the tuning process it was found that the strings fundamental frequency would actually change suddenly by as much as 1¢. A plot of the beat signal between the string and the ideal note shown in Fig. 9 clearly shows such a sharp transition. These transitions took place only once while the string was sounding at seemingly random time intervals.

These transitions would clearly skew the 'true' frequency of the note. It was assumed that these sudden transitions were anomalies associated with the hexachord as opposed to being typical of pianos in general. This assumption has yet to be confirmed as the frequency stability of piano strings has yet to be studied.

This error likely comes from two sources: the plucking of the string and the stiffness of the hexachords bridge. As was previously mentioned because the string is being plucked by hand its transient dynamics differ from those of a string struck with a felt hammer [14]. In addition, a plucked string may change its three-dimensional modes of vibration faster than one struck with a felt hammer producing the observed slight, rapid changes in frequency.

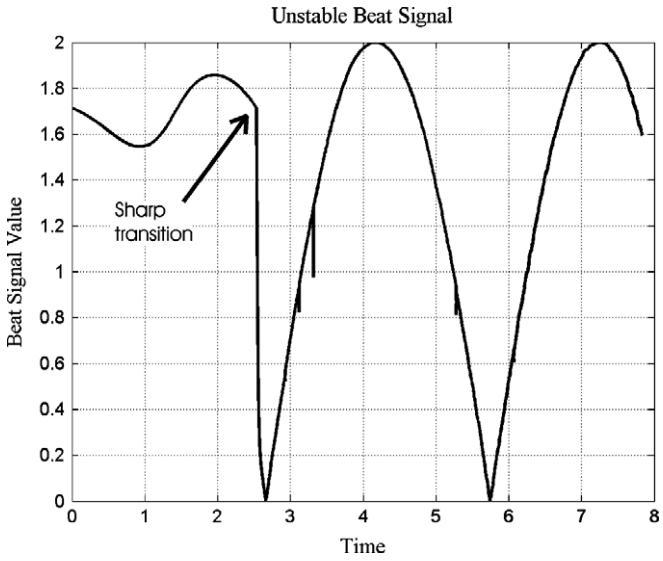

Fig. 9. Sharp transition in frequency.

A second possibility is that the hexachords bridge is not stiff enough and is vibrating slightly with the string, causing the vibration of the string to change in the process. High speed video of the hexachord's and a real piano's bridge has been taken and revealed that a real piano's bridge does not vibrate with the string and that the hexachords does to a small degree. In any event this system would need to be tested on a real piano to give a proper assessment of its performance.

8. Conclusions and future work

PitchImpact is an automated piano tuning system that uses reinforcement learning to control a mechanical impact hammer that adjusts the tuning pegs on a piano. In this paper a brief history and the current state of the art of piano tuning was presented along with a detailed design and performance analysis of PitchImpact. A study of the impact-pitch relationship revealed that coarse pitch change is quadratically related to impact size. A similar relationship in the fine tuning range was not observable due to great variations in pitch change for a given impact. Due to time constraints, a first-order Markov process was used to model this relationship and was coupled with a greedy policy. This system alone may have been able to tune a single string after an incredibly time consuming 1890 impacts. A priori knowledge of a rough relationship between impact setting and pitch change was used to allow the agent to use data from other output values and even other states to make reasonable impact predictions when faced with very sparse amounts of data. An ad hoc strategy was used to overcome a data starvation problem that is inherent with the prediction equations used. A Parzen windows technique was applied to the rewarding process to allow the values in $\mathscr{R}_P$ to form plausible distributions from a few data points, hopefully increasing the convergence rate of the impact prediction routine.

The experimental results of this system indicate that the collective learning policy has allowed the agent to start behaving reasonably with very sparse amounts of data. The

experiments also revealed that the performance of the agent improves after the value of $\mathscr{I}_{\mathscr{M}}$ is updated. Currently $\mathscr{I}_{\mathscr{M}}$ is infrequently updated.

Based on the current results of PitchImpact system, the following are appropriate areas of future study:

First, the current first order Markov model is likely not the optimal model to solve this problem. It was seen that fine tuning adjustments are associated with a great deal of variance. The shortest path to make a fine tuning adjustment may actually involve making two coarse tuning impacts in opposing directions: it may be more difficult to make a precise fine tuning adjustment than it is to make a precise coarse tuning adjustment. If these assertions are true a Markov model that looked forward 2 or more steps ahead would perform much better than the current first order policy.

Second, experiments revealed that updating $\mathscr{I}_{\mathscr{M}}$ more frequently would improve the performance of the system. How this is done will depend on how $\mathscr{I}_{\mathscr{M}}$ affects the pin and pin block friction characteristic. Finding a good estimation $\mathscr{I}_{\mathscr{M}}$ would likely improve the performance of this system appreciably.

Acknowledgements

Professor Stephen Birkett of Systems Design Engineering has been a wealth of obscure information on piano tuning, a wonderful resource and sounding board during the research work in this very new topic. To Alice Malisia, Beth Vary and Melanie Stern, thank you very much for your efforts with the pitch analysis software.

References

[1] Machlis KFJ. The enjoyment of music. 9th ed. W.W. Norton & Company; 2003.
[2] Martin DW, Ward WD. Subjective evaluation of musical scale temperament in pianos. J Acoust Soc Am 1954;26(5):932.
[3] Gravagne N. Tuning pin torque & tuning that too tight pin. Piano Technicians J 1993:24–7.
[4] Larudee P. Tuning pin physics: Part 1. Piano Technicians J 2002:16–9.
[5] Larudee P. Tuning pin physics: Part 2. Piano Technicians J 2002:20–3.
[6] Bowman K. Tuning lever design & maintenance: Part III. Piano Technicians J 2002:18–23.
[7] Levitan D. Tuning technique Part 1: Physical technique. Piano Technicians J 1996:23–6.
[8] Gilmore D. Apparatus and method for self-tuning a piano, US Patent 6,559,369.
[9] Conklin H. Tuning pin for pianos, US Patent 4920847.
[10] Richardson J. Piano forte, US Patent 388720.
[11] Kaelbling AMLP, Littman ML. Reinforcement learning: A survey. Journal of Artifical Intelligence Research 1996:237–85.
[12] Sutton RS, Barto AG. Reinforcement Learning: An Introduction. Cambridge, MA: MIT Press; 1998.
[13] Duda R, Hart P, Stork D. Pattern classification. 2nd ed. New York, NY: Wiley; 2001.
[14] Fletcher N, Rossing T. The physics of musical instruments. 2nd ed. New York, NY: Springer Verlag; 1998.