Capturing Violin Performance Gestures with a Single Camera System

Kosmas Kritsis

MASTER THESIS UPF / 2016

Master in Sound and Music Computing

Master thesis supervisor:

Dr. Rafael Ramirez-Melendez

Dr. Alfonso Pérez Carrillo

Dr. Zacharias Vamvakousis

Department of Information and Communication Technologies

Universitat Pompeu Fabra, Barcelona

Capturing Violin Performance Gestures with a Single Camera System

By **Kosmas Kritsis**

MASTER THESIS UPF / 2016
Master in Sound and Music Computing

“Any who may wish to profit himself alone from the knowledge given him, rather than serve others through the knowledge he has gained from learning, is betraying knowledge and rendering it worthless.”

Haile Selassie I

Αφιερωμένο στους γονείς μου Ευάγγελο και Ζαφειρούλα, στις αδερφές μου Άννα και Φωτεινή καθώς και στα δίδυμα αγγελούδια.

Acknowledgements

First of all I would like to express my gratitude to Dr. Xavier Serra, for the given opportunity to actively participate within the department as an M.Sc. student and obtain valuable knowledge concerning the sound and music technology field of research. Also, I would like to gratefully thank my professor, Dr. Rafael Ramirez-Melendez for his constant support though-out my studies. Moreover, I would like to express my deep appreciation to my supervisors Dr. Alfonso Perez Carrillo and Dr. Zacharias Vamvakousis, for their confidence regarding my skills, guidance, support, understanding and patience during our collaboration. Special thanks go also to my advisor, collaborator and best friend Georgios Z. Papadopoulos, who is currently an Associate Professor at the Telecom Bretagne, Rennes, France. Last but not least, I would like to deeply express my wholehearted love to my parents and sisters for encouraging and believing in me, because nothing would be possible without their support and help.

Abstract

In this work we present a novel approach in capturing violin performance gestures by employing both direct and indirect acquisition methods based on low-cost equipment that is available with any modern personal device. It is a multidisciplinary study that covers several research fields, including audio signal processing, machine learning, motion capture and computer vision. According to the bibliography, the majority of the proposals employs a direct way for the acquisition of musical gestures, by measuring the physical variables with the support of sensors which are placed on the instrument or on the performer. However, these systems invoke some kind of intrusiveness that affects the performance procedure. An alternative approach is to apply indirect acquisition from the analysis of the audio signal. The main difficulty of this method is to develop robust detection algorithms and provide accurate measurements similar to the sensors. Therefore, our goal was to implement a hybrid audio-informed system that utilizes the built-in web camera and microphone of a laptop, in order to provide qualitative feedback to the performer. This was achieved by developing an algorithm that employs video frame analysis, augmented reality and audio signal processing methods. After computing the various features from the video and audio domains, we unified the retrieved information into a single dataset in order to apply feature selection and machine learning techniques for investigating the regression prediction between the audio descriptors and the bowing controls, as they were computed from our analysis algorithm. The results are promising, since they present high correlation rates between the Bow Inclination parameter and the audio features, with maximum accuracy of 97%.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

During a musical performance, the artist is required to transform a musical pattern or score into a sequence of instrumental gestures in order to control the instrument and thus producing the expected sonic result. In this sense, the musical idea can be transformed between different representations including the musical score, the gestures and the sound domain. Moreover, the interaction between the musician and his instrument gets much more complex when performing excitation-continuous musical instruments such as the bowed-string instruments, which are considered of being between the most expressive musical instruments. Tracking and acquiring violin instrumental gestures parameters, by means of physical movements that are directly involved in the sound production procedure, has receive a lot of research attention, with multiple applications in performance transcription, performance modeling, mapping performance models with gestural patterns, sound synthesis, and pedagogy.

1.1 Motivation

A novice violin player needs to develop a greate variety of cognitive and physical skills specified by reading music notation (i.e. the score), counting notes and have rests in order to follow the rhythm, learning how to place correctly his fingers on the fingerboard as well as listening to the sound outcome for playing in right tuning. Furthermore, in order to reach professional performance skills in playing a string instrument such as the violin, a musician has to develop precise control of complex arm movements, as well as great postural awareness.

The bowing action in bowed-string instruments, is a complex motor skill that requires the coordination of various number of degrees of freedom (DOF) between the shoulder, elbow, wrist and hand. The most difficult part of playing a string instrument, lies in the sound generation process, which happens due to the frictional interaction between the bow and the string. Therefore, learning how to play the violin is a difficult and long process that demands effective teaching reinforced through excessive practice and training.

Traditionally, the learning of how to perform the violin requires to observe and imitate the actions of the teacher while listening to his verbal feedback. Sometimes a mirror is used so that the students can watch their own bowing actions and postures. However, learning by observation and imitation is challenging for novice players since most of the times they do not know which are the important parameters to focus and how to translate what they see into their own movements. Moreover, this master-apprentice model often restricts the student to have short sessions with his teacher and long periods of self-study that unfortunately results to high abandonment rates.

However, various scientist have approach this problem in order to develop interactive systems that are able to assist the players and promote self-learning through augmented feedback. The most important part of these systems is the technology that they employ in order to capture and validate the performance gestures. The majority employs a direct way for the acquisition of musical gestures, by measuring the physical variables with the support of sensors which are placed on the instrument or on the performer. Usually, the direct measurement involves the use of expensive sensors with some degree of intrusiveness and generally consists of complex setups. An alternative approach is by applying indirect acquisition from the analysis of the audio signal. Indirect acquisition is not an intrusive method and has numerous advantages such as its simplicity, low-cost setup in addition to the possibility of examining old audio recordings. Nevertheless, the main difficulty is to develop robust detection algorithms and provide accurate measurements similar to the sensors. Only the last years, there are some studies that employ both methods in order to minimize the overall cost while increasing the accuracy of their systems.

1.2 Objectives

As an aftermath, this Master's Thesis will try to approximate the efficiency of a low-cost gesture recognition system that utilizes in real-time both built-in cameras and microphones, which nowadays are available with many personal devices such as the laptops and smartphones. Additionally, we will investigate the potential relations amongst the variety of the available low-level audio features and their gestural representation. Hence, the goals of this work can be specified to the following:

- Record real violin performances by means of multimodal data.

- Implementation of a video processing algorithm for tracking different bowing parameters.
- Identify relevant audio features that can be related and predict the various bowing gestures.
- Implementation of an audio-informed tracking system for bowing gestures.

1.3 Thesis structure

After this brief introduction, Chapter 2 focuses on presenting the related state of the art in the given research field, along with general information that may help for the understanding of this project. Next, Chapter 3 describes the methods that were employed for recording real violin performances as test data, along with the algorithms that were developed for capturing the bowing parameters, in both audio and video recordings. Chapter 4 presents and explains the evaluation method along with a discussion on the retrieved results. Final comes Chapter 5, which exposes the reproducibility aspects and the contributions of this project and lists possible future work that could be developed by means of extending the system, while improving its results and general performance.

Chapter 2

STATE OF THE ART

Initial efforts on experimenting with digital interactive systems within music and arts, are dated to more than fifty years old. For instance, in the piece entitled "Variations V", Cage and Cunningham developed a system where dancers are capable to control the soundscape creation with their movements, hence establishing a new relationship between gestures and music performance [Miller, 2001]. However, the interactive digital technology gained interest from the research community during the 1990s due to the emergence of the different sensor technologies and software that were developed for tracking human and object motion. This field of research and technology is known as Motion Capture (MoCap), which can be further employed in various applications, including Human-Computer Interaction (HCI) systems, animation creation, biomedical analysis, sports performance analysis and as already mentioned in music and arts performance.

Furthermore, researchers and developers have implemented special software and programming libraries for motion tracking, such as EyesWeb [Camurri et al., 2007], Vicon [Ltd, 2016], MoCap Toolbox [Burger and Toiviainen, 2013], EyeCon [Wechsler et al., 2004], The Gesture Recognition Toolkit [Gillian and Paradiso, 2014] and openCV [Bradski, 2000]. However, some of them are commercial software that require the users to purchase licenses, thus introducing further cost considerations. On the other hand, open source software can provide a cost-free solution, that enable the users to proceed with an experimental approach on the utilization of technology.

In a general view, there is a diversity on the available approaches for developing MoCap application that can be classified either as marker-based or marker-free systems [Bregler, 2007]. For instance, in a marker-based tracking system, we record the reflections of markers that are placed on the subject of interest, while in a marker-free capture system we may record multiple natural points for replicating the three-dimensional (3D) space.

However, we can argue that the aforementioned classification is not efficient for describing the details and requirements of MoCap utilization in artistic applications, since technical constrains influence the expression and freedom of movements during performance. In order to provide meaningful systems we should analyze the various gestural notations, with respect to the different scientific disciplines that converge towards gestural research, including computer science, engineering, music, dance and cognitive science.

In this sense, this Chapter provides a musical approach on the meaning of gestures. Next, it outlines the main categories of MoCap technology as employed in numerous artistic-centered projects, followed by a brief description of the bowing gestures and the state of the art on the possible acquisition techniques for these gestures.

2.1 Gestures in music

Globally, gestures derive as intrinsic information that is further perceived and transmitted in different levels, such as visual and auditory queues in conjunction with internal cognitive processes. For instance, the visual information are transmitted by the movements of a musician, along with the audible information that is produced whilst performing his instrument. Therefore according to their physical nature, the musical gestures, can be broadly defined either as composition gestures (i.e. not physical) or performance gestures (i.e. physical) [Metois, 1996]. Moreover, the composition gestures can be thought as the musical information that is notated on the musical score, such as the notes, chords, dynamics and gradual changes etc. On the other hand, the performance gestures refer to movements that are conducted by the performer during the transformation of the notations into musical sound.

In this sense, we can further specify the performance gestures, as being either ancillary or instrumental gestures [Wanderley and Depalle, 2004]. We refer to instrumental gestures to those that take part in the sound production procedure, such as the fingerings on the strings and the bow acceleration in the context of a violin performance. The ancillary gestures are introduced by the musician, by means of extra body movements that are not directly connected with the sound production, rather than associating with the performance in a way to express and transmit emotional information, which can also modulate slightly some sound features.

Furthermore, the instrumental gestures can be categorized into more detailed groups according to their function on the instrumental object that they are applied [Cadoz, 1988], which are the excitation, modulation and selection gestures as described below:

- Excitation gestures are those that emit the energy which is found in the

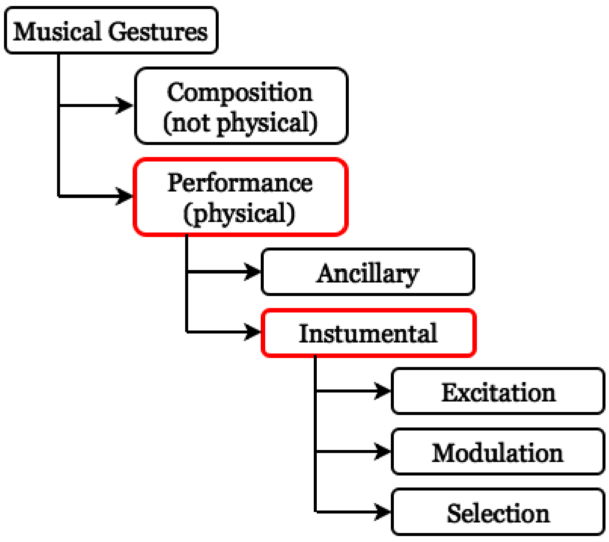


Figure 2.1: Musical gestures classification [Cadoz, 1988].

sound outcome, such as the blowing force in wind instruments.

- Modulation gestures refer to those that modulate the sound properties of the instrument without transmitting energy on the sonic outcome and can be further divided into parametric and structural. Parametric modulation gestures are those that affect constantly an instrumental parameter, similar to the bow pressure variations during a violin performance. On the other hand, structural modulation gestures are those that modify the structure of the instrument, such as opening or closing the keys of a saxophone during performance.

- Selection gestures refer to those that require an instant choice between various but equivalent forms, that happen during a musical performance, such as in bow-based instruments, where the choice of the bow direction does not affect the energy outcome neither applies any instrumental modification.

However, in real performance conditions, it is hard to distinguish the different gestural mechanisms since they happen in parallel. For instance excitation and modulation gestures can be found during a vibrato in a flute performance. A more diverse example is the complex functionality during a violin performance, where the musician emits energy with the movement of his bow (i.e. excitation gesture), while modulating its speed (i.e. modulation gesture) and deciding its direction (i.e. selection gesture). The aforementioned classification is illustrated in Figure 2.1.

2.2 MoCap for music applications

Consequently, we can classify the variety of gesture technology for motion capture according to the medium that is employed for transmitting the information, by means of body, space and time sensing [Bevilacqua et al., 2011].

2.2.1 Body-centered

It is obvious that the physical movements are the most common medium for interfacing with digital systems. However, additional body-centered signals may be employed for interaction, such as the physiological signals.

Physiological signals

There is a great variety of possible physiological parameters that can be utilized in interactive systems. Most of the sensing technology for capturing this kind of signals have been developed for health monitoring and general biofeedback applications, such as the electromyography (EMG) and the electroencephalography (EEG) methods. Furthermore, EEG is used to monitor the electrical activity of the brain neurons by placing multiple electrodes along the scalp. For instance, musical interactive systems based on the brain waves activity have been developed for real-time emotive control in music performance [Giraldo and Ramirez, 2013] as well as treating depression [Ramirez et al., 2015].

Similarly, the EMG allow us to monitor the skeleton-muscle motions in form of electrical activity, by applying surface skin conductive sensors on various parts of the body, according to the area of interest. A literature review of the artistic interactive systems based on EMG bio-signals tend to fall into three main fields, including performance skill analysis, training method evaluations and bio-feedback for pedagogy and injury remediation [Visentin and Shan, 2011]. However, these types of sensing technologies delicate to pre-movements of muscle tension, which further introduce noisy recordings, thus requiring some sort of filtering in order to retrieve meaningful measurements.

It is important to mention that the property of the skin to be electrically conductive allow us to develop numerous touch sensitive applications and monitor additional bio-signals such as the heart beat. For instance, researchers have experimented with the heart rate bio-feedback for proposing training methods in the context of emotional management and anxiety recovery during performance [Thurber et al., 2010].

Movements and postures

There are numerous technological methods that allow us to capture gestures and movements, since most of the interactive systems require some kind of body-centric action. One solution can be provided by the wearable sensors that carry accelerometers and gyroscopes which can monitor the pitch, yaw and roll rotations as well as the acceleration of the object that carries these devices. Nowadays, the production cost of these sensors has decreased dramatically, thus allowing the industry to integrate them in multiple electronic devices, such as the smartphones and other gaming interfaces. Furthermore, flex sensors can be employed in order to monitor the curvature measurement, such as the bending of the fingers in different postures [Mitchell, 2011] or the elasticity of object, similar to the WaveSaw [Dahl et al., 2007] and Music Ball projects [Jensenius and Voldsund, 2012].

Another efficient approach on capturing gestures and movements remotely is to employ video cameras, in conjunction with special software, as those mentioned on the beginning of this Chapter, that allow us to apply computer vision technics so as to filter the input recordings and track the subject of interest. For instance, EyesWeb [Camurri et al., 2007] is capable of recognizing the body skeleton from single camera captions and providing details about the direction, stability and quantity of the subjected motion.

However in simple single camera installations the motion of the subject and its position are always relative to a two-dimensional (2D) representation that is also limited to the visual range of the camera, thus introducing further complexities for extracting 3D information. In order to overcome such problems, one solution is to install multiple cameras that usually associate with reflective markers or special LEDs in order to digitally reconstruct the 3D captions. Nevertheless, these systems are expensive to acquire, which by its side introduces further cost considerations.

A more cost-efficient option can be found within the modern multimedia cameras such as the Microsoft Kinect and Leap Motion, which both utilize infrared (IR) technology in order to recognize the 3D space, according to the orientation of specifically projected IR patterns. For instance, Hantraku and Kaczmarek have experimented with Leap Motion for sound synthesis as well as an assistive performance tool for modulating effects in real-time [Hantrakul and Kaczmarek, 2014]. Furthermore, Kinect has been used in research projects in the context of musical interfaces, similar to the Air Violin project [Fan and Essl, 2013], as well as for gesture analysis in music conducting [Sarasúa, 2013] and music performance [Hadjakos, 2012].

2.2.2 Space-centered

There are multiple research proposals that follow a space-centric approach for interfacing. Such applications usually require the user to manipulate either physical or virtual space features, in order to actively participate in the interaction process. Usually, the events are triggered according to the movements and presence of objects in predefined spatially distributed zones. However, through the literature we are able to distinguish two broad approaches on the utilization of space, which can be either physical or virtual.

For instance, an early experimentation with virtual space interactions can be considered the piece ”Various V” where the dancers were able to interact with Theremin antennas that were placed along the stage [Miller, 2001]. Modern proposals on the subject have experimented with virtual augmented reality, where users can react with virtual audio objects such as the AHNE interface [Niinimäki and Tahiroglu, 2012], as well as for promoting collaborative creativity, similarly to the Music Room project [Morreale et al., 2014].

Nevertheless, researches on musical interactive systems within the physical space, include studies for tangible music tables such as the Reactable [Jordà et al., 2007], which can be played by manipulating a set of objects that are distributed on top of an interactive table surface. Furthermore, each object has a specific function on the generation, modification and control of the output sound, in addition to the projected visual feedback that further illustrates the sound path onto the table surface. Generally, motion is naturally related with these interactions by means of measuring its orientation in particular spatial zones, thus referring to absolute locations, and not to the body itself as described in the section above.

2.2.3 Time-centered

Even if it is not clear how time is used as a medium for interactions, we can argue that temporal features are able to be utilized for triggering events, in a coherent manner to the space-centered interactions. In this sense, interactions can take place in predefined time limits or on fixed moments, similar to the spatial zones. Therefore this approach requires a temporal analysis of the sensing information, since the interactions are driven by instant events, sequences and synchronization mechanisms, according to specific gestures and audiovisual processes.

An example of research with time-centric interactions is the EyeHarp [Vamvakousis and Ramirez, 2012], where the performer is able to triggered musical events over a continuous real time tracking of his gaze. Another paradigm that falls in this type of interfacing, is the Virtual Conductor project, which is a software that emulates an orchestra conductor, able to lead and interact with musicians while they simultaneously perform a musical work [Reidsma et al., 2008].

2.3 Instrumental gesture acquisition classification

In general, the acquisition of instrumental gestures can be approached from two different domains, which are the domain of physical actions and the domain of audio analysis. Moreover, the very small details of the physical actions during an instrumental performance are hard to be perceived by a listener only from the produced sound, thus emerging the need to develop and apply methods that directly attend the actual instrumental control signals. However, the acquisition of the instrumental gestures on the physical domain may arise some limitations on the permmited level of intrusiveness by the measuring setup.

Therefore, the knowledge of the underlying physical phenomena that take an active role on the modulation of the perceptual properties of the instrumental sonic result, allow us to employ indirect models which are induced by the physical actions throughout a musical performance. According to the study of Wanderlay and Depalle entitled "*Gestural control of sound synthesis*" [Wanderley and Depalle, 2004], such methods belong to the indirect acquisition domain and can be considered as an alternative or even complementary approach to the general acquisition problem.

2.3.1 Indirect acquisition - audio analysis

Globally, the indirect acquisition of instrumental gestures has received much attention due to its interdisciplinary background. The given simplicity on recording the sonic environment with microphones and the previous research that already has been conducted by the community on the different audio analysis techniques, along with the knowledge on the acoustical attributes of the various musical instruments, enables the scientists to experiment with the acoustic signal itself in order to extract the involved instrumental gestures.

For instance there are various articles that approach the acquisition of the guitar plucking point, either from the spectral domain [Traube and Smith, 2001, Traube et al., 2003] or from the time domain [Penttinen and Välimäki, 2004]. Examples of studies on wind instruments investigate the estimation of the fingerings on the flute [Kereliuk et al., 2007] and the saxophone performance [Smyth and Wang, 2014], as well as the reed pulses of the clarinet mouthpiece [Smyth and Abel, 2012] and the lip-valve system relation of the trombone [Smyth and Scott, 2011].

Generally, the domain of audio processing for retrieving instrumental gestures is proven to be effective in cases where the analysis and modeling of the perceptual mechanism is of higher importance. However, these methods may fall upon inconvenient surjections, since their description is based only on the sonic result of the physical actions which are performed by the musician. In other words, indirect acquisition may provide incomplete representation of the instrumental gestures that affect the timber of the outcome sound, since several control parameters may have a unique point in the instrument's timber space [Maestre, 2009], in addition to mutual influences of noisy attributes that are also present in the resulting sound, such as the effect of the room acoustics and possible performer movements [Wanderley, 2001].

2.3.2 Direct acquisition - physical actions

In theory, the direct acquisition of musical gestures provides the best solution since the recorded signals are uniquely coupled with discrete gestures. Furthermore, the physical actions are measured by utilizing one or more sensors, which usually are of different type and technology, such to meet the needs of the various capturing systems and tracking scenarios. The signals from these sensors are able to isolate and acquire raw information of the basic physical features of gestures such as pressure, linear or angular displacement and acceleration. Typically, each sensor is employed in order to capture a specific physical variable of the gesture, thus not requiring any preprocessing step for gestural induction.

However, due to the independence of the captured variables, direct acquisition techniques may underestimate the interdependencies that may exist amongst them [Wanderley, 2001]. Moreover, the high difficulty of gathering information without intruding to the instrument or the performer, makes this method a less explored field, with exception the bowed-string instrumental family, which has been researched by numerous scientists and proven to be a very successful capturing technique, with respect to the characteristics of the bowing-based interaction between the performer and the musical instrument. Examples that fall into this acquisition methodology are presented and commented in the following section, after a brief introduction to the basics of the bowing gestures in violin performance.

2.4 Bowing gestures in violin performance

Bowed-string instruments are considered being highly musical expressive, similar to the human voice. Even though, the space of the control parameters is limited, it can still provide adequate freedom to the instrument player for constantly modifying in detail the timber of the sonic result. Most part of the performance expressiveness is carried by the musician through his navigation style between the notes in the control parameter space [Schoonderwaldt, 2009].

The violin, as a bowed-string musical instrument, is usually played by sliding the bow on one of the strings, known as bowing performance. Furthermore, the

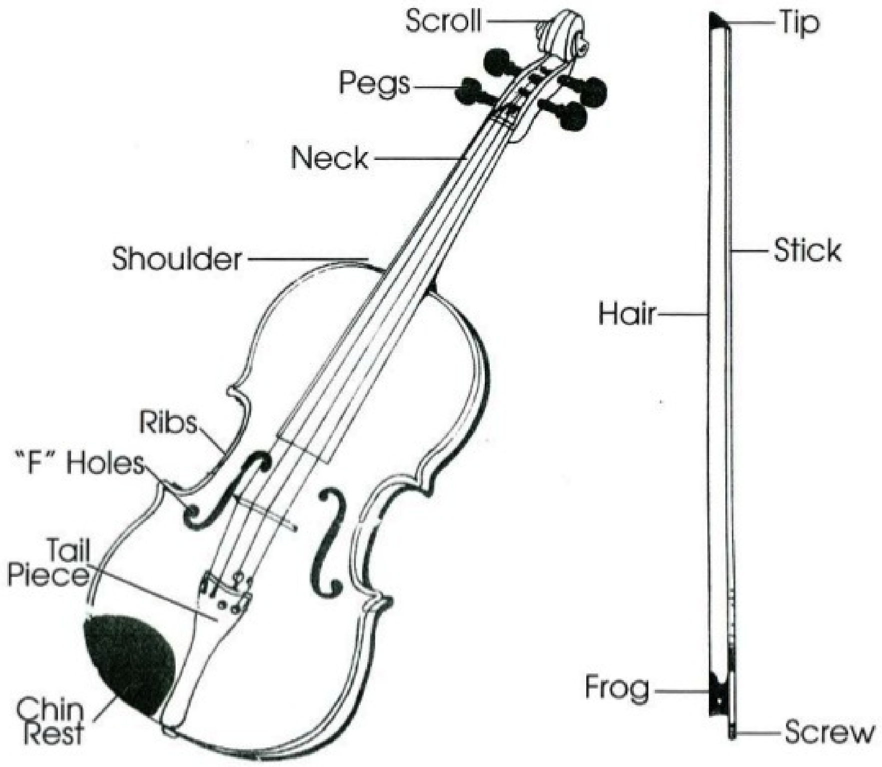


Figure 2.2: Parts of violin and the bow.

violin consists of 4 strings, usually tuned at 660 Hz (E), 440 Hz (A), 293 Hz (D), and 195 Hz (G). Usually, the violin musicians enumerate the strings in ascending order, starting from the highest to lowest frequency, however in this document they are addressed by their note name, in order to avoid any confusions and misconceptions.

As it is illustrated in Figure 2.2, the strings start at the peg box, where each string is inserted into a hole in one of the pegs respectively. The peg is a cylindric piece able to turn and roll the string so as to modify its tension and thus tuning it. Next the strings pass over the nut, which provides four grooves so as to ensure that the strings are equally spaced, then continue over the fingerboard up to the bridge and reach their ending point on the tailpiece, that usually contains one or more fine tuners. However it is important to mention that the bridge is one of the most critical parts on the violin, since it is the component that is resonating by the strings' vibrations and is responsible for transmitting the movement down to the top plate and the rest of the violin body.

On the other hand, the bow is a stick that it is usually made of wood or even carbon fiber. The stick needs to be light, supple and bendy in order to be able to support the tightening and loosening of the bow hair, which are attached as a ribbon between the tip and the frog of the bow. Usually, the hair are made of either a synthetic material or horse hair. The musician can pull the ribbon and increase its tension by moving the frog downwards along the stick, with the help of a screw. If the screw on the end of the frog is completely unscrewed then the frog can be detached from the bow.

As with the majority of the classical musical instruments, both hands are required in order to perform the violin. By giving a non-functional perspective to the previously mentioned classification of instrumental gestures, violin performance involves left-hand and right-hand instrumental gestures. Typically, the left hand is in charge of controlling the fingerings and the length of the played string, while the right hand holds the bow and transfers the excitation energy to the strings, by modeling the desired envelope and modulating its timber space. The interaction between the bow hairs, the string and the fingerings, creates the characteristic vibration of the bowed-string, which by its side is resonating the violin body through the bridge end that is in contact with the string. Finally, the structure of the resonating violin body amplifies the string vibration and gives the sonic result that is transferred through the surrounding air molecules to the listener's ears.

Consequently, the playing techniques that are related with the performance expressiveness in classical violin are conducted by the right hand instrumental gestures that are not directly involved on the selection process of the played string [Maestre, 2009]. Generally, these right hand gestures that are directly affecting the bow-string interaction, are know as *bowing controls* or *bowing parameters*, and they are employed continuously during a violin performance in order to control the timbre attributes of the final sound.

2.4.1 Violin bowing parameters

The performance of bowing control parameters is carefully controlled by the musician so as to achieve the expected acoustical character of the notes and phrases while his physical actions are constrained by the bow-string system. In other words, the violin performer is required to continuously synchronize various bowing parameters, where some of them may be in conflict with each other as a result of physical, bio-mechanical (i.e. the players build and level of performance skills), or musical (i.e. the score) limitations. However, according to Schoonderwaldt's doctoral study [Schoonderwaldt, 2009], there are three basic bowing parameters participating in the violin performance, which are the bowing velocity, the bow-bridge distance and the bow force, in addition to four "secondary" attributes, including the bow position, tilt, skewness and inclination. The aforementioned bowing parameters are illustrated in Figure 2.3 and are briefly described as follows.

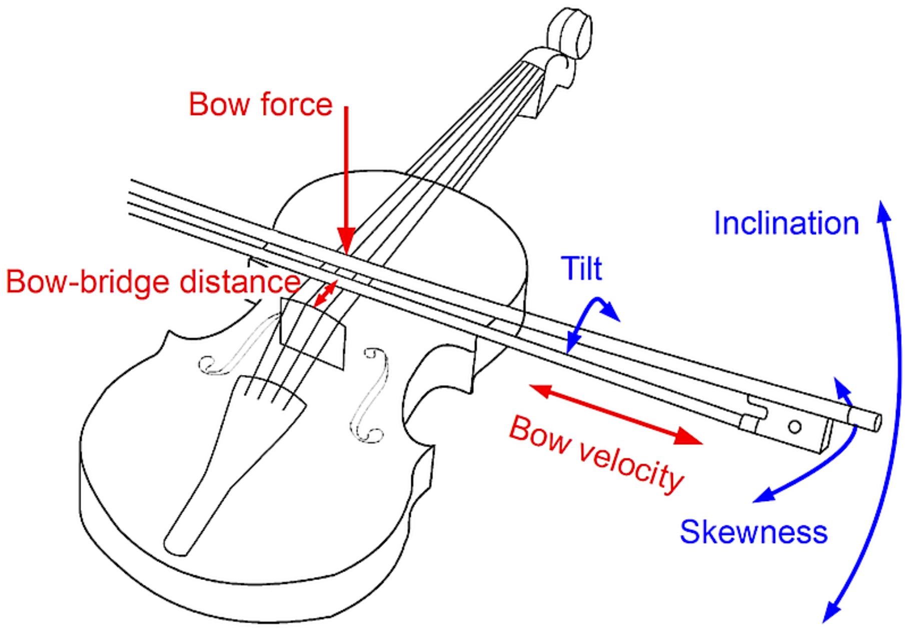


Figure 2.3: Violin bowing parameters. The red color represents the basic parameters in addition to the secondary parameters that are marked with blue color [Schoonderwaldt, 2016].

- **Bow velocity:** Represents the velocity of the bow as imposed by the player's right hand, while he is holding the bow at the frog. The instant velocity on the contact point with the string is not exactly the same due to small modulations in the bow hair and bending vibrations of the stick. Bow velocity sets the string amplitude along with the bow-bridge distance.

- **Bow-bridge distance:** Is specified as the distance along the string, between the contact point of the bow and the bridge. The bow-bridge distance in conjunction with the bow velocity shape the string amplitude, and it is often modulated as a mean for controlling the high-frequency content or "brightness" of the resulted sound.

- **Bow force:** The force with which the bow hair is pressed against the string at the contact point. The bow force parameter determines the timbre of the tone by controlling the high-frequency content in the string spectrum, accordingly to the two previously mentioned parameters. Moreover, in tones of normal quality the bow force needs to stay within a certain allowed range. The upper and lower limits for this range in bow force range increase with increasing bow velocity and decreasing bow-bridge distance.

- **Bow position:** Is the measurement that specifies the distance between the contact point of the bow hair on the string and the frog. The bow position alternates between ?at the tip? and ?at the frog?. The bow position does not influence the string vibrations by itself, but has a profound influence on how the player organizes the bowing movement. The given length of the bow hair is one of the most important limitations in violin performance.

- **Bow tilt:** As it is illustrated in Figure 2.3, the bow tilt is the rotation of the bow around the length axis. The bow is often tilted during performance in order to reduce the number of bow hairs that are in contact with the string. In classical violin playing, the bow is tilted with the stick towards the fingerboard. Altering the tilt angle helps the performer to modulate both, the width of the hair ribbon and the pressing force that is applied on the string.

- **Skewness:** Is the angle that is formed between the length axis of the bow and an hypothetical line parallel to the bridge. Skewness is an indicator that is used for specifying the deviations in the so called "straight bowing".

- **Inclination:** Is the pivoting angle of the bow, vertically to the strings. The inclination is a parameter mainly employed in order to select the played string.

2.4.2 Acquisition of bowing parameters in violin performance

Initial efforts on the realization and analysis of the bowing gesturers during violin performance approach the subject from a pedagogical perspective. Moreover, there is an extensive literature that focuses on performance training and education techniques, while they are providing qualitative description of the possible bowing patterns in classic music [Jackson et al., 1987, Fischer, 1997, Galamian and Thomas, 2013].

However, there are also some early studies that approach the problem by analyzing information captured from real violin performances. At the late 50's of the last century, Hodgson made the first attempt on capturing and visualizing the trajectories of the bow and the bowing arm by using the so called "cyclegraphs" [Hodgson, 1958]. This technique was developed by the manufacturing industries in order to analyze the time efficiency of their employees. Nevertheless, he took advantage of this technology and by attaching small electrical bulbs on the bow and on the performer's arm, he was able to record the light motion on a still-film plate and thus retrieving visualizations of the general bowing patters.

Almost three decades later, Askenfelt investigated the basic aspects of the bow motion, by using a modified bow that was equipped with custom-made sensors for retrieving calibrated measurements of most bowing parameters, except the variety of the bow angles [Askenfelt, 1989]. Additionally, he achieved to measure the typical ranges of the bowing parameters as well as the basic bowing styles including, detache, crescendo-diminuendo, sforzando and spiccato. The results of his study show that the coordination of the bowing parameters is the most interesting aspect during violin performance.

After ten years, a different approach was studied by Paradiso and Gershenfeld, who measured the bow displacement by means of oscillators which were driving the sensing antennas (i.e. electric field sensing). In a first prototype that was carried by a cello, a resistive strip was attached to the bow and it was driven by a mounted antenna behind the bridge, thus resulting to a wired violin-bow system. The second prototype was improved, since it was the first attempt for developing a wireless acquisition system for tracking the bowing parameters. It was implemented for the violin and the antenna worked as the receiver, while two oscillators that were placed in the bow, worked as drivers. The bow pressure was measured by using a force-sensitive resistor below the point where the forefinger is placed [Paradiso and Gershenfeld, 1997].

In the more recent years, the research community has developed less intrusive systems for tracking the violin bowing parameter. For instance, in the studies published by Young, he measured the bow downward and lateral pressure with foil strain gages, while the bow-bridge distance was measured in a similar way to the previously presented method of Paradiso and Gershenfeld. Furthermore, the strain gages were permanently mounted around the middle point of the bow stick and the sensing data were collected by a remote computer via a wireless transmitter, which was mounted at the frog [Young, 2002, Young, 2007]. However this setup resulted being of considerable intrusiveness to the performer thus affecting his movements and his overall performance.

An interesting approach for developing a less intrusive system, was developed by Goudeseune, who employed a commercial Electromagnetic Field (EMF) device for tracking some low-level movements and later, he was mapping them to specific digital synthesis parameters which were controlled in real-time during a violin performance scenario. However, his procedure on extracting the movements was not representing the real bowing gestures, since he was simply using the speeds and rotations of the sensors that were placed on the violin and the bow without considering the relevant instrumental gestures and bowing parameters [Goudeseune, 2001].

Another interesting study that was conducted by Rasamimanana et al. [Rasamimanana et al., 2005], describes a wireless acquisition system that measures the acceleration of the bow by attaching accelerometers on the bow, in addition to some force sensitive resistors (FSRs) which are used in order to obtain the strain of the bow hair by means of bow pressure. The advantage of this system was that it could be easily installed to any type of bow. On the other hand, its significant drawback was that the sensing data required heavy post-processing so as to obtain the representative gestural information, since it was able to acquire only the acceleration measurement. However, an attempt to overcome this limitation was conducted by Schoonderwaldt et al., who employed video cameras in conjunction with the measurements given by the accelerometers in order to model the different bow velocity profiles [Schoonderwaldt et al., 2006].

Furthermore, the accuracy and robustness of measuring the bow force was studied and improved by numerous scientific contributions, by using strain gages that were attached to the frog-end of the bow hair, hence achieving to measure the hair ribbon deflections [Demoucron and Causse, 2007, Demoucron et al., 2009, Guaus et al., 2007, Guaus et al., 2009].

Considering the advancements of commercial EMF systems, there was another important research that was conducted by scientist members of our department (MTG-UPF), where they developed an accurate and transformable MoCap system for measuring the various bowing control parameters [Maestre et al., 2007]. More details of the aforementioned EMF system are presented in the next Chapter, since it was employed in this thesis for retrieving the ground-truth measurements of the bowing controls.

Another example of research that employ MoCap technology in combination with attached sensors for tracking the bow force and other bowing controls, was developed again by Schoonderwaldt, and it was based on an expensive commercial camera-based motion capture system that required a more complex calibration system with difficult post-processing procedures [Schoonderwaldt and Demoucron, 2009].

Also, the overall research in capturing bowing parameters led to commercial products like the K-Bow [McMillen, 2016], which consists of an augmented bow and a removable emitter that is attached to the underside of the fingerboard. Furthermore, the K-Bow is able to measure the bow position, acceleration, force and tilt for controlling sound processing algorithms in real time.

However, there are also scientific contributions that approach the acquisition problem from the spectral domain. For instance, Perez et al. have published a series of articles where they study various indirect acquisition methods for tracking violin controls from the audio signal by applying machine learning algorithms on empirical data that were previously collected with a highly accurate sensing system. The learning consists of training statistical models with a database that includes audio spectral features and instrumental controls, such as bow tilt, bow force, bow velocity, bow-bridge distance as well as the played string [Perez-Carrillo and Wanderley, 2012, Perez-Carrillo and Wanderley, 2015].

Chapter 3

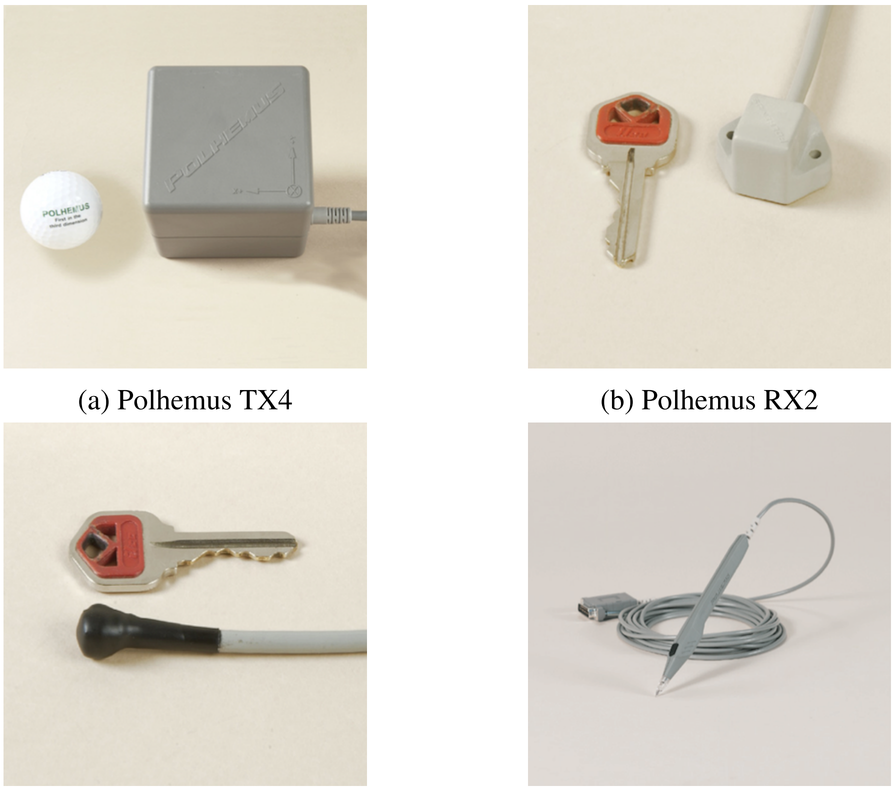
MATERIALS AND METHODS

This Chapter contains all the appropriate information regarding the various techniques that were employed for acquiring the multimodal data, along with the recording set up and the audiovisual analysis algorithms that were developed for this Thesis study. Moreover it describes the details of the Polhemus[1] EMF sensing system and its application on the acquisition of the bowing gestures, as proposed by Maestre et al. [Maestre et al., 2007], in addition to the low-cost camera and microphone system as they were utilized in this project. Next, the analysis algorithms that were developed for the video and audio data are presented along with the applied post-processing for removing possible acquisition noise before the transition to the evaluation phase.

3.1 Data collection and recording set-up

As an initial step in our study, we emphasized on collecting violin performance information from diverse domains. Hence, we decided to select the laptop's built-in web camera and microphone as low-cost equipment for recording the real violin performances. Next, we had to select a reliable sensing system, able to provide accurate measurements from the physical movements of the violin player and later use this information as ground-truth. Consequently, we employed a commercial EMF tracking system, which was previously studied by research members of our department (MTG-UPF) with successful results on measuring the different bowing parameters. In addition to the laptops built-in microphone's recordings, we captured the audio of the violin with a bridge pickup in order to capture a cleaner audio signal. Finally, all these multimodal data required preprocessing procedures so as to get synchronized before applying the various analysis algorithms in the different representation domains.

[1]http://polhemus.com/



(a) Polhemus TX4 (b) Polhemus RX2

(c) Polhemus RX1-D (d) Polhemus ST8

Figure 3.1: The main accessories of the Polhemus system as employed in this study.

3.1.1 Polhemus EMF motion tracking system

For the acquisition of the related bowing parameters we employed the EMF Polhemus Liberty commercial device, which is able of delivering real 6 DOF sensing, based on alternative current (AC) electromagnetic technology [Polhemus, 2016]. Furthermore, it consists of a source transmitter unit along with a set of wired sensors that work as receivers with sizes around 0.5cm and weights down to 6gr. Each sensor provides 3 DOF for translating its position from the source and 3 DOF for its orientation by means of yaw, pitch and roll. In our case, we use the TX4 source model (see Figure 3.1a), which has a sampling rate of 240Hz, latency less than 0.4ms, static accuracies of 0.75mm for the translation and 0.15 degrees for the orientation within a range of 2m.

As it is depicted in Figure 3.2, the sensor model RX2 (see Figure 3.1b) is attached to the violin body and is marked as S1, while the RX1-D sensor model (see Figure 3.1c) that is named as S2, is attached close to the center of gravity of

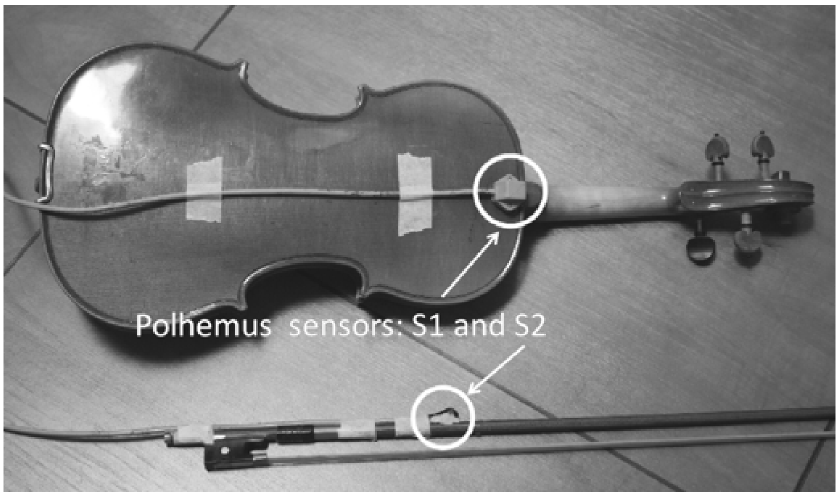
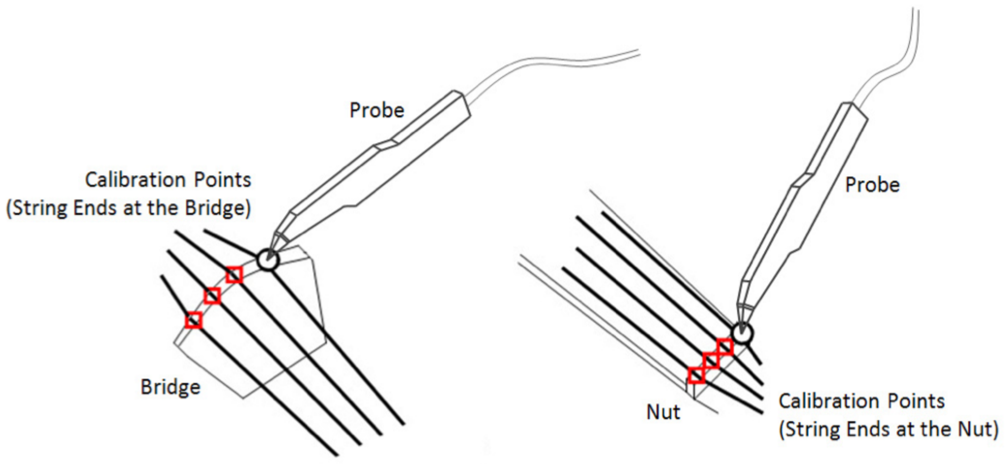


Figure 3.2: Detail on the positioning of the two 6DOF Polhemus sensors [Maestre, 2009].

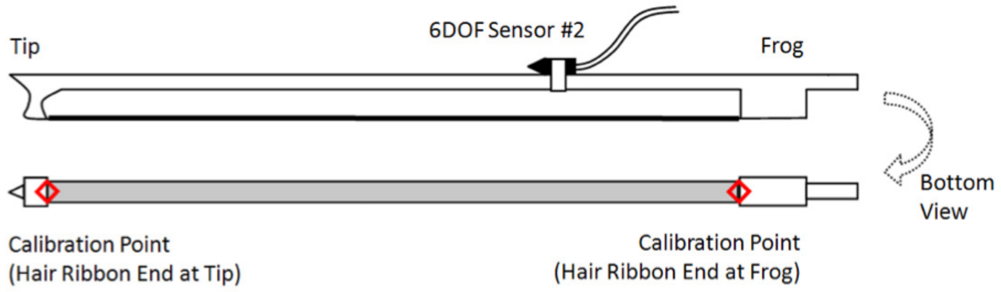
the bow. This model was selected in order to minimize the intrusiveness on the bow while its balanced point remains unaffected. From the 6DOF data stream of the sensor that is attached on the body, we can extract the position of the ends of each string, which can be estimated for any position or orientation of the violin. Analogously, the ends of the hair ribbon can be accurately estimated from the 6DOF data streams of the sensor that is attached on the bow. The ends of the strings and the hair ribbon, along with the reference positions of the two sensors, are able to obtain the various bowing parameters, including the bow transversal position, bow velocity, bow-bridge distance and bow tilt. However in order to obtain meaningful results we have to calibrate the system.

The calibration process consists of two steps that are quit similar. The sensor S1 that is attached on the violin provides the mechanism for estimating the eight ending positions of the four strings, between the four slopes at the bridge and the other four at the nut (see Figure 3.3a), relatively to the coordinate system defined by the sensor's six DOF for its positioning and orientation. With the second step we obtain the ending positions of the hair ribbon, between the tip and the frog, relatively to the coordinate system defined by the translation and rotation of the S2 sensor that is attached on the bow (see Figure 3.3b).

In order to calibrate the system, Polhemus provides the stylus sensor ST8, as it is depicted in Figure 3.1d. The stylus sensor is required for the calibration and is used similarly to a marker for annotating the locations of the various points of interest, so that they can be expressed in their respective coordinate systems, while the violin and bow sensors remain still. Generally, the three employed sensors are able to support accurate positioning and rotation measurements with regards to the



(a) Schematic view of the strings calibration step.



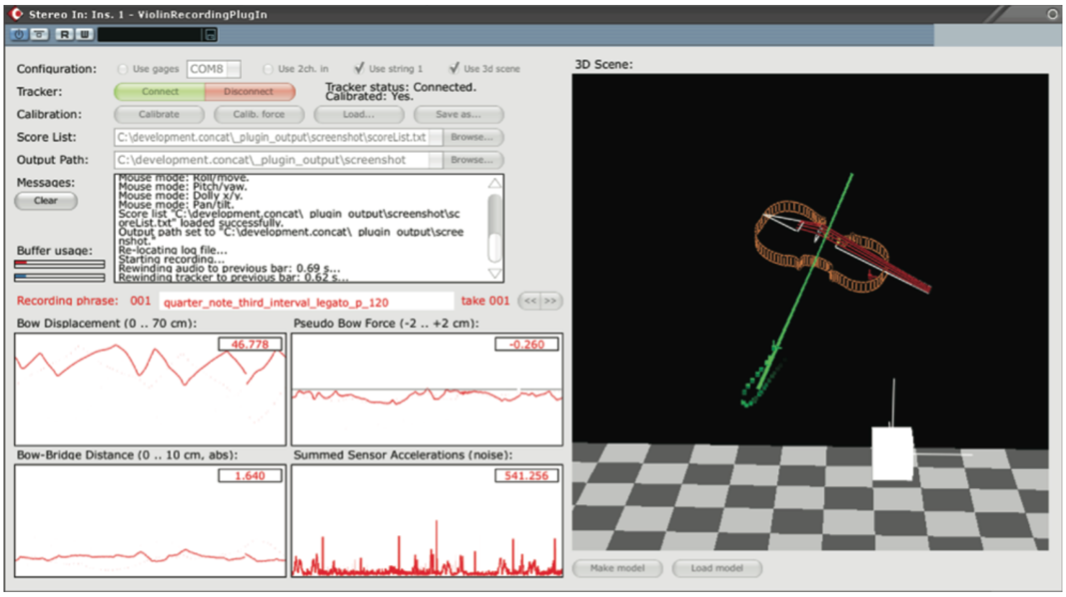
(b) Schematic view of the hair ribbon calibration step.

Figure 3.3: Detail view of the calibration points of interest on the strings and the bow hair ribbon.[Maestre, 2009].

common reference point defined by the coordinates of the Polhemus TX4 source transmitter. The translation and orientation of S1 define the violin coordinate system in which the coordinates of the string ends remain unchanged while the violin is in motion. In a similar manner, the translation and orientation of S2 measures the coordinates of the two extremes of the bow hair, since they move and rotate jointly to the bow motion.

After the calibration process, the sensors remain attached to the bow and violin body, so that the strings and the hair ribbon can be tracked during real performance. Once we obtain the coordinates of the individual points in their respective vectorial spaces through calibration, we are able to detect their translation and rotation in the reference vectorial space (i.e. the physical coordinate system), at any time by monitoring the relevant sensing data from the corresponding sensors.

Synchronization of the various sensing data is integrated via a VST plug-in



Configuration: COM8 ✓ Use string 1 ✓ Use 3d scene

Tracker status: Connected.
Calibrated: Yes.

Score List: C:\development.concat_plugin_output\screenshot\scoreList.txt

Output Path: C:\development.concat_plugin_output\screenshot

```
Mouse mode: Roll/move.
Mouse mode: Pitch/yaw.
Mouse mode: Dolly x/y.
Mouse mode: Pan/tilt.
Score list "C:\development.concat\_plugin_output\screenshot\sc
oreList.txt" loaded successfully.
Output path set to "C:\development.concat\_plugin_output\scree
nshot."
Re-locating log file...
Starting recording...
Rewinding audio to previous bar: 0.69 s...
Rewinding tracker to previous bar: 0.62 s...
```

Recording phrase: 001 quarter_note_third_interval_legato_p_120 take 001

Bow Displacement: 46.778 · Pseudo Bow Force: -0.260 · Bow-Bridge Distance: 1.640 · Summed Sensor Accelerations: 541.256

Figure 3.4: Screenshot of the VST while providing real-time 3D representation of the violin and bow positions, in addition to some visualizations of the bowing parameter.

that is used within the Steinberg's Cubase [2] commercial digital audio workstation (DAW) software. Furthermore, the plug-in is able to manage and correct the latency between the different data streams arriving from the bridge pickup and the EMF motion tracker. Additionally, it contains specific working modes for carrying out the calibration process, while it visualizes in teal-time the 3D representation of the position of the violin and bow, as well as some graphs of the bowing parameters which are useful for giving visual feedback to the performer. An example of the VST's graphical user interface (GUI) is presented in Figure 3.4.

3.1.2 Low-cost camera system

According to our goals, we have selected to build our system over low-cost multimedia equipment that is available on most personal electronic devices, such as smartphones, tablets and laptops. Consequently, we utilized the built-in FaceTime HD camera of an early 2012 MacBook Pro (see Figure 3.7a), formerly known as *iSight* camera, in order to capture real violin performances. This web camera is capable of providing an image resolution of 1280x720 pixels with a sampling frequency of 30 frames per second (fps), thus allowing us to record and analyze high quality videos. Additionally, all the video recordings were captured by utiliz-

[2]http://www.steinberg.net/en/home.html

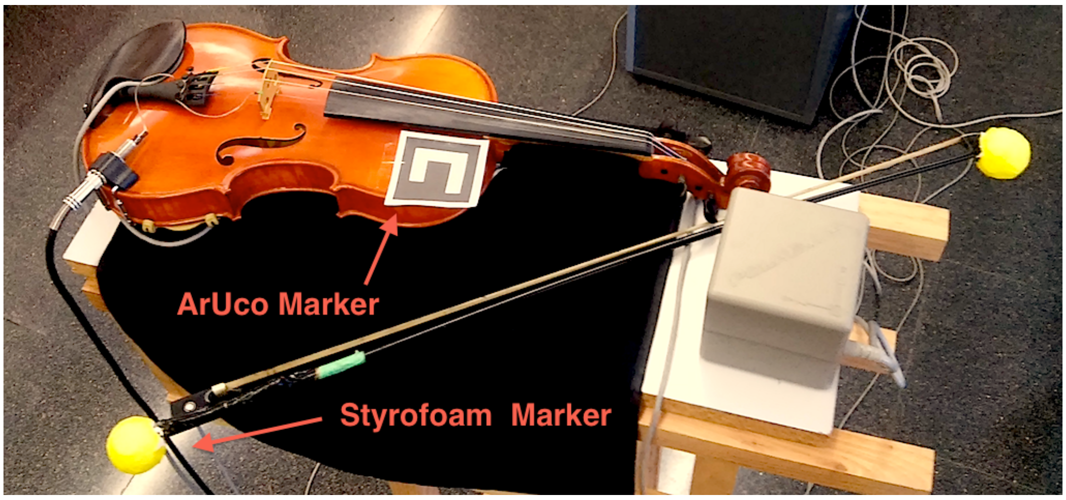
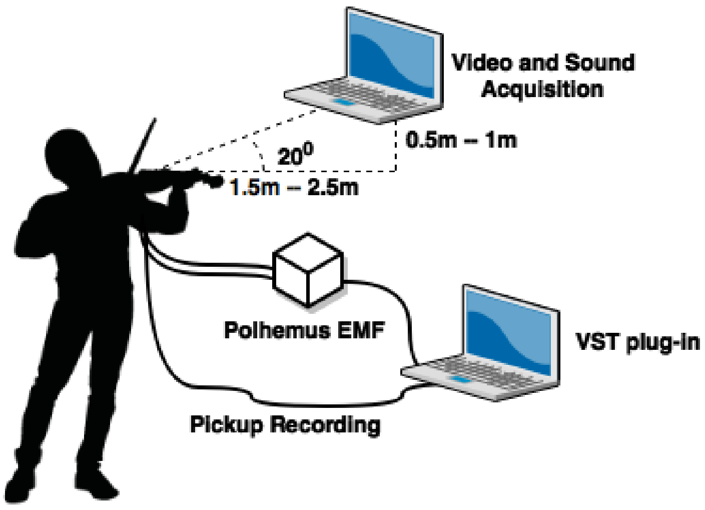


Figure 3.5: Detail on the positioning of the styrofoam color markers for detecting the bow, in addition to the ArUco marker for estimating the pose of the violin body with respect to the point of view of the camera.

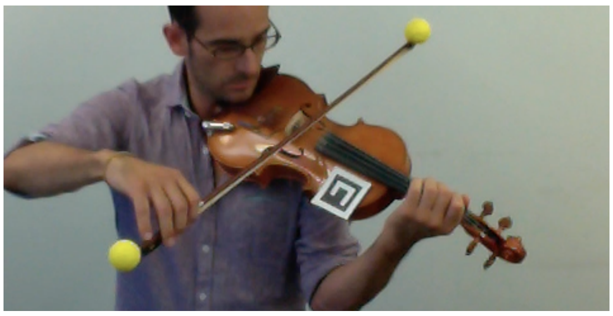
ing Apple's QuickTime Player version 10.4, with maximum quality settings and stored in *.mov file format.

In order to proceed with the video recordings, we have selected to employ color markers and the ArUco fiducial markers [Garrido-Jurado et al., 2014], as a low-cost alternative to the Polhemus system. Furthermore the colored markers are custom made out of styrofoam so as to be light enough and to avoid adding extra weight on the bow. As it is illustrated in Figure 3.5, they are placed on the tip and the screw points of the bow, while having a spheric shape with a yellow color so they can be easily tracked from any point of view and lighting conditions. The ArUco fiducial marker is used for estimating the camera pose with respect to the marker. In other words, it allows us to detect the translation and orientation of the fiducial marker according to the point of view of the camera, thus making it an optimum solution for Augmented Reality (AR) applications. Regarding our application setup, we employ a single fiducial marker that is placed at the bottom right of the top plate of the violin body, under the fingerboard where the neck starts, as it is depicted in Figure 3.5. We have selected this position to minimize the possibility of sight overlapping by the performer's hand and bowing movements while recording the video.

Consequently, the recording setup consisted of two laptops. One was dedicated to record the Polhemus EMF system along with the audio that was coming from the violin pick up. The second laptop was placed between 1.5m and 2.5m away, with a height of 0.5m to 1m from the the violin player's shoulder, as it is presented in Figure 3.6a. Moreover, the performer needed to face the camera, while



(a) Recording set-up general scheme.



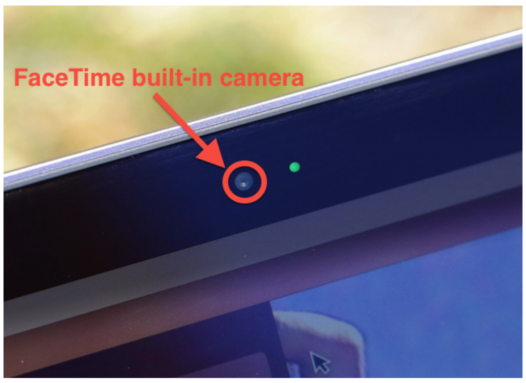
(b) Example of the point of view from the web camera.

Figure 3.6: Recording setup scheme and an example on the perspective from the web camera.

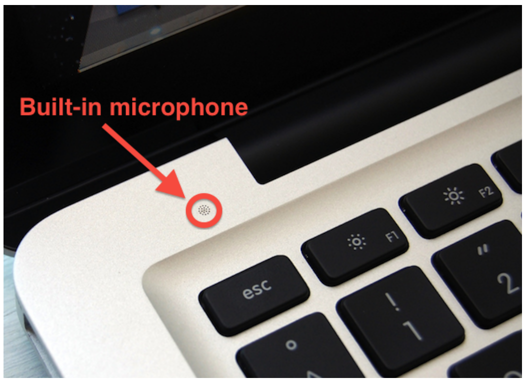
holding the violin's fingerboard in a parallel position with the camera, similarly to Figure 3.6b.

3.1.3 Audio acquisition

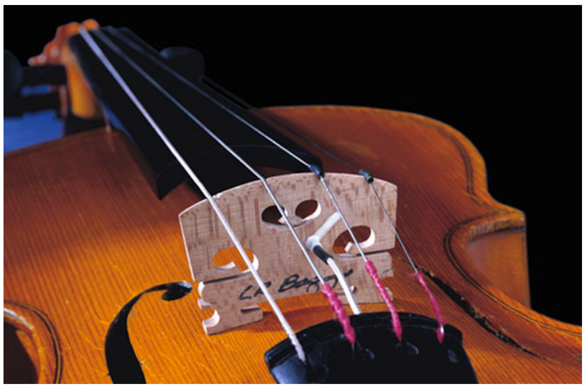
At first sight, the estimation of bowing control signals does not appear to be directly linked with the audio signal that is produced by the violin. However, by synchronizing the recorded audio with acquired bowing parameters, it is possible to retrieve important information concerning the effect of the various bowing controls on the resulted sound. Hence, by analyzing the timber related features of the recorded audio content, we will investigate their relevance to the acquired bowing parameters so as to result in an audio-informed model.



(a) MacBook Pro FaceTime HD camera.



(b) MacBook Pro built-in microphone.



(c) L.R. Baggs acoustic violin pickup.

Figure 3.7: Detail of the positioning of the built-in FaceTime HD camera and microphone of an early 2012 MacBook Pro, as well as a closeup picture of the employed L.R. Baggs bridge pickup.

Regarding our recording setup we employed two types of audio microphones, including the built-in microphone of the same laptop that was used for the video captures (see Figure 3.7b), in addition to the L.R. Baggs acoustic violin bridge pickup[3] that is illustrated in Figure 3.7c. The built-in microphone was selected as a low-cost alternative to the bridge pick-up which is capable of recording much cleaner sound, by means of maximum sensitivity. The pickup has a piezoelectric transducer as an integral part of the bridge that captures the instrument's inherent dynamics and resonances. Furthermore, its selective admittance pattern is aligned so that the signal is generated while the bridge moves in a left-right motion, in response to the string vibrations. Additionally, the pickup rejects any impulse directed to the plane of the bridge by minimizing possible undesired signals from the finger squeaks, body movements or background noise.

[3]http://www.lrbaggs.com/pickups/violin-pickup

3.1.4 Data preprocessing

In total we recorded two series of real violin performances. The first attempt took place in one of the offices within our department, where we utilized only the low-cost camera and microphone system. There the player was asked to play all the notes of the four strings in three different bowing speeds. On the second attempt we employed also the Polhemus EMF system in addition to the low-cost system, and the recording took place in the demonstration room of our department. There we asked from the performer to play in four different bowing speeds all the notes of the four strings as well as a short improvisation part. However, during the performances the player was having rests between the different patterns, which resulted in recoding irrelevant movements as well as audio signals that required further processing in order to clear and synchronize our multimodal recordings before proceeding to the analysis phase.

As it is already mentioned, the multimodal data were captured with two laptops. One laptop was utilizing the VST software which is capable of synchronizing the EMF tracking data with the audio signal from the bridge pickup. The other laptop was recording the performances with the QuickTime software while the camera was focusing to keep within an optical range the various markers that were placed on the bow and the violin. Then, the video recordings were divided in numerous parts according to the current played string by utilizing the Apple's iMovie software, version 10.1. With this application we were able to edit simultaneously the video images with the recorded audio signal from the built-in microphone.

3.2 Video analysis

Even thought there are many available software and programming libraries for motion tracking, such as EyesWeb [Camurri et al., 2007], Vicon [Ltd, 2016], MoCap Toolbox [Burger and Toiviainen, 2013], EyeCon [Wechsler et al., 2004] and The Gesture Recognition Toolkit [Gillian and Paradiso, 2014] we have selected to employ the OpenCV [Bradski, 2000] library, since it comes with an open-source license and it is well maintained with a lot of documentation.

Moreover, OpenCV[4] is a computer vision and machine learning software library that provides C, C++, Python and Java interfaces for coding, while supporting cross-platform applications amongst Windows, Linux, Mac OS X, iOS and Android operating systems (OSs). It has more than 2500 core functions which are written in optimized C and C++, in order to take advantage of the multi-core processing. Additionally, it allows the programmer to enable the OpenCL technol-

[4]http://opencv.org/

ogy that uses the hardware acceleration of the underlying heterogeneous compute platform, thus improving the overall performance in real-time applications.

Similarly, the ArUco library was developed on top of the OpenCV and is considered as an integral part of the OpenCV package, hence inheriting its core functions and capabilities. Furthermore, it contains functions for generating and detecting various fiducial marker dictionaries, including AprilTag, ArToolKit, ARTAG and ArUco. However, the pose estimation function supports only the ArUco markers.

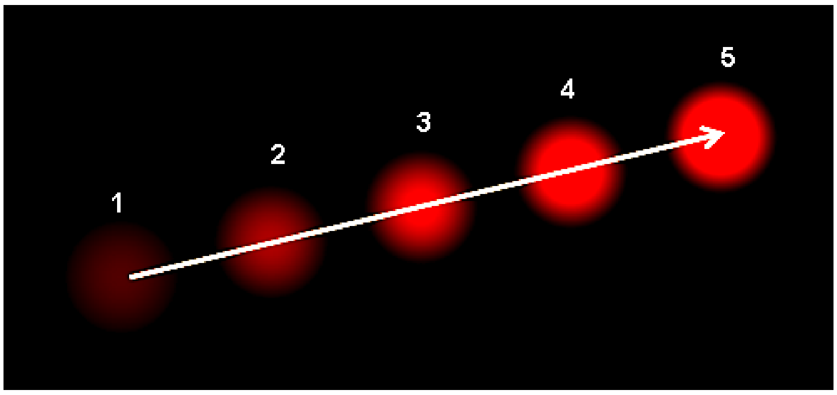
Therefore, the video analysis phase contains two main parts. The first step is dedicated into tracking the color markers that represent the two extremes of the bow, while the second is focused on detecting the ArUco marker that is attached on the violin body and estimating its pose with respect to the camera orientation. All of the algorithms that we developed for this phase are written in C++ with Apple's Xcode integrated development environment (IDE), version 7.2.1.

3.2.1 Color tracking

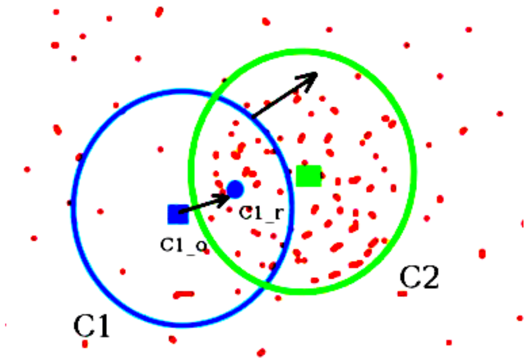
Up to date, there are numerous studies that contribute in the computer vision research field of object detection and scene segmentation. However, it still remains a challenging task due to the great diversity of possible backgrounds and shapes of the tracking objects. The most convenient way to detect and segment an object from an image is by applying color filtering. In order to achieve meaningful results with this method, the object of interest should have a significant color difference from the background.

As a result, we needed to simplify our problem since we were constrained by the low-cost requirement of our system design. Hence, we selected to employ the yellow styrofoam balls in order to achieve maximum color difference. Furthermore, this material is light and easy to handle while having a rough surface that minimizes the light reflections from the different lighting conditions. Additionally, its spheric shape maximizes the visual contact within the camera range, during the performance of the various bowing gestures. Considering our study, we investigated the performance of three different approaches for the color tracking, which are the Meanshift, Camshift and a modified version of the optical flow algorithm.

The Meanshift algorithm works by starting with a set of pixels within the limits of a window that is moving while searching the image area for the maximum pixel density, as it is illustrated in Figure 3.8b. Moreover, the initial window that is presented by a blue circle with the name *C1*, has an original center which is marked with a blue rectangle and is named *C1_o*. However, if we compute the density centroid of the points inside the given window we will get the point that is marked as *C1_r*. This means that the window should move till its center matches



(a) Example of the optical flow algorithm between five consecutive frames.



(b) Example of the Meanshift algorithm.

Figure 3.8: Examples on the Meanshift and the optical flow motion tracking algorithms.

the previously calculated density centroid. This iterative process continues till the center of the window and its density centroid falls on the same location, so as to obtain a window with maximum pixel distribution which is presented by a green circle and marked as *C2*. However, this method has the disadvantage of a static window size that results to many errors.

As an improvement, the OpenCV research community developed the Continuously Adaptive Meanshift algorithm, know as Camshift. This method starts by applying Meanshift. Once it converges, it updates the size of the window and calculates the orientation of the best fitting ellipse. Next it applies the Meanshift with the new scaled window and its previous window location. This process iterates until it reaches a predefined accuracy, in a similar manner to Meanshift.

After implementing both of the aforementioned algorithms, our results show very low accuracy in tracking the movements of the two color markers. More specifically, the centroid of the window was unstable in nearly every frame of our video recordings, thus making it impossible to track the bow stick and its movements. Therefore, we developed the third algorithm that is based on the optical flow approach. The basic concept behind this method is the detection of the pattern of the apparent motion of an object within a visual scene between two consecutive frames. This motion can be specified by two-dimensional vectors, where each vector represents the displacement of the points of interest, from the first frame to the second. Considering the example that is presented in Figure 3.8a, the arrow shows the displacement vector between five consecutive frames.

Our algorithm takes advantage of the main idea behind the optical flow, by means of tracking the displacement of the contours of the two color markers, between the consecutive frames of the recorded videos. As it is illustrated in Figure 3.9a, our implementations starts by receiving an incoming video frame that is converted into the HSV color space, which stands for *hue*, *saturation*, *value*, and is the most common cylindrical-coordinate representation of the RGB color model. This color space allow us to isolate a specific color within different saturation and brightness ranges, thus enabling robust color filtering. Next, the filtered image that contains the yellow color contours is converted into grey-scale, so as to apply the appropriate morphology filtering that is useful for clearing possible background noise and isolating the contours of the two markers.

Up to this point we are able to calculate the size of the bow by measuring the maximum distance in pixels, between the two centroids of the contours of the detected markers, with respect to the real size of the bow stick. This step is where we obtain the bow calibration data, which further allow us to estimate real-world measurements from their pixel representations. Consequently, as a final task we apply the optical flow method along with single view geometry principles in order to calculate the velocity vectors, velocity magnitude and acceleration of the two markers, in addition to the bow inclination according to the camera orientation. The mathematical formulas that we employed for calculating these parameters are listed as follows:

- **Bow length:** Is calculated by applying the euclidean distance between the coordinates of the two points in the center of the detected marker contours. As $A(x, y)$ is considered the attached marker on the tip of the bow, while $B(x, y)$ is the other marker that is placed on the frog screw. This formula is calculated in pixels.

$$L = \sqrt{(A_x - B_x)^2 + (A_y - B_y)^2} \tag{3.1}$$

- **Velocity vector:** In order to calculate this vector, it requires to calculate the coordinates of the individual marker contours at least in two consecutive video frames. Negative X vector values specify upward directions, while positive vectors have a downward direction. This equation is applied separately for the A and the B markers. The velocity vector is calculated in pixels.

$$V_t = (X_t - X_{t-1}, Y_t - Y_{t-1}) \tag{3.2}$$

- **Velocity magnitude or instant velocity:** Similarly to the velocity vector, we need at least two consecutive video frames for calculating the instant velocity of the A and the B markers. It is a function of time that specifies the rate of pixel displacement. In order to translate it to the physical magnitude we need to multiply by the number that is given from the ratio between the maximum bow length and the current bow length in pixels.

$$v_t = \sqrt{\frac{\partial x^2 + \partial y^2}{\partial t}} \times \frac{max(bowlength)}{bowlength} \tag{3.3}$$

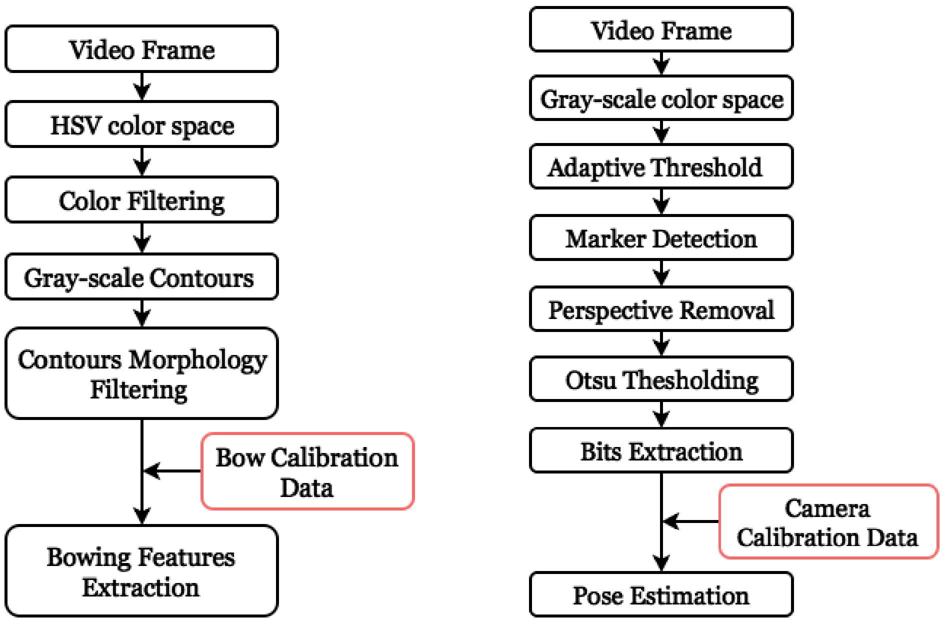
- **Acceleration:** Is the rate of change of the velocity of the A and the B markers as a time function between two continuous video frames.

$$\alpha = \frac{\partial v}{\partial t} \tag{3.4}$$

- **Inclination:** This parameter is calculated according to the pythagorean theorem that is applied to the imaginary right triangle, that is shaped between the current two dimensional projection of the three dimensional bow and its angle with respect to the camera orientation. More specifically, we consider the current bow length as the adjacent side and the maximum bow length as the hypotenuse. Hence, we are able to estimate the angle between these two sides with the following equation.

$$Inclination = \cos^{-1}\left(\frac{bowlength}{max(bowlength)}\right) \tag{3.5}$$

The aforementioned parameters are computed for every incoming frame of the video sequence and their results are saved as plane text in a *comma-separated values* (CSV) file format. Each line of the file consists of eight fields that represent the various calculations of a single frame, including the Current Bow-length, Velocity Vector A, Velocity Vector B, Velocity Magnitude A, Velocity Magnitude B, Acceleration A, Acceleration B and Inclination. As A and B we refer to the color markers that are placed on the tip and the frog screw respectively.



(a) Color tracking algorithm. (b) ArUco tracking algorithm.

Figure 3.9: The schemes of the developed algorithms for tracking the movement of the color markers in addition to the detection of the ArUco marker.

3.2.2 ArUco AR marker tracking

Generally, the ArUco markers are binary square fiducial markers that can be used for camera pose estimation and AR applications. The software library includes methods for the detection of this type of markers as well as the appropriate tools for executing pose estimation and camera calibration. The main advantage of this library is that the marker detection algorithm is simple to use, robust and fast.

However, in order to perform pose estimation we need first to calculate the calibration parameters of the employed camera. These parameters allows us to determine how a 3D point within the visual range of the camera is projected on the camera sensor. Furthermore, the camera parameters can be divided into intrinsic and extrinsic parameters as it is depicted in Figure 3.10. As intrinsic parameters are specified the following.

- The coordinates in pixels of the focal length in the two axes: $f(x, y)$
- The coordinates in pixels of the optical center of the camera sensor in the two axes: $c(x, y)$
- The radial and tangential camera distortion coefficients: k_1, k_2, k_3, p_1, p_2

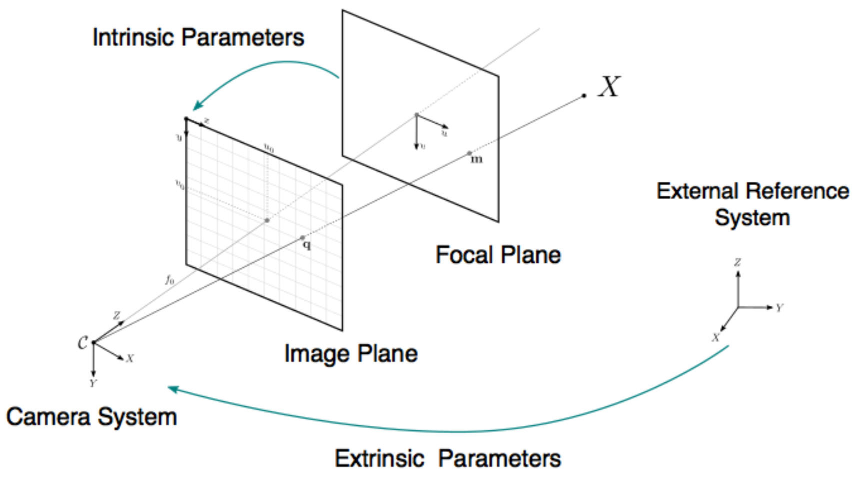


Figure 3.10: The intrinsic and extrinsic projection parameters. Intrinsic parameters allows to model the optic component considering distortions introduced by the lens in the image. Extrinsic parameters represent the camera position and orientation with respect to an external reference system.

In an ideal camera with no distortion coefficients we could estimate the projection of a 3D point (X, Y, Z) with the formulas below:

$$x' = \left(\frac{Xf_x}{Z}\right) + c_x \tag{3.6}$$

and

$$y' = \left(\frac{Yf_y}{Z}\right) + c_y \tag{3.7}$$

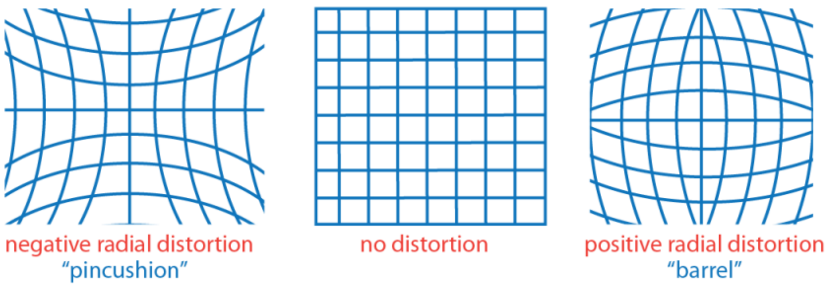
Nevertheless, the camera lenses normally distorts the scene while projecting the points far from their real center on the camera sensor, by means of radial and tangential distortion. The positive radial distortion promotes the so called *barrel* effect as depicted in Figure 3.11a, and can be calculated with the following formulas:

$$x_{distorted} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{3.8}$$

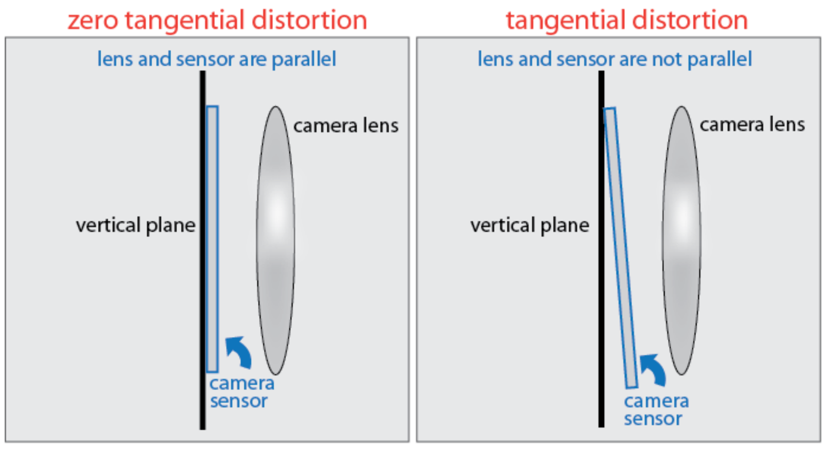
and

$$y_{distorted} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{3.9}$$

with $r^2 = x^2 + y^2$ and x, y as the undistorted pixel coordinates. Typically, two coefficients are sufficient for the calibration. In cases of severe distortion, such as in wide-angle lenses, we can add the k_3 coefficient in our equation. On the other hand, tangential distortion occurs when the lens and the image plane are not



(a) Radial distortion.



(b) Tangential distortion.

Figure 3.11: Examples of the radial and the tangential camera distortions [Mathworks, 2016].

perfectly parallel, similarly to Figure 3.11b. The tangential distortion coefficients model can be calculated as:

$$x_{distorted} = x + \left[2p_1xy + p_2(r^2 + 2x^2)\right] \tag{3.10}$$

and

$$y_{distorted} = y + \left[p_1(r^2 + 2y^2) + 2p_2xy\right] \tag{3.11}$$

with $r^2 = x^2 + y^2$ and x, y as the undistorted pixel coordinates.

As a consequence, if we want to know a pixel's projection, then we must take into account the distortion coefficients. However, the above formulas assumes that we know the 3D location of the point of interest with respect to the camera reference system.

In order to compute the projection of a point with respect to an external reference system, we must then employ the extrinsic parameters (see Figure 3.10). Basically, these parameters represent the 3D rotation and translation vectors which are required for translating the camera reference system to the external one. The vectors are specified as:

$$R_{vec} = (R_x, R_y, R_z) \tag{3.12}$$

and

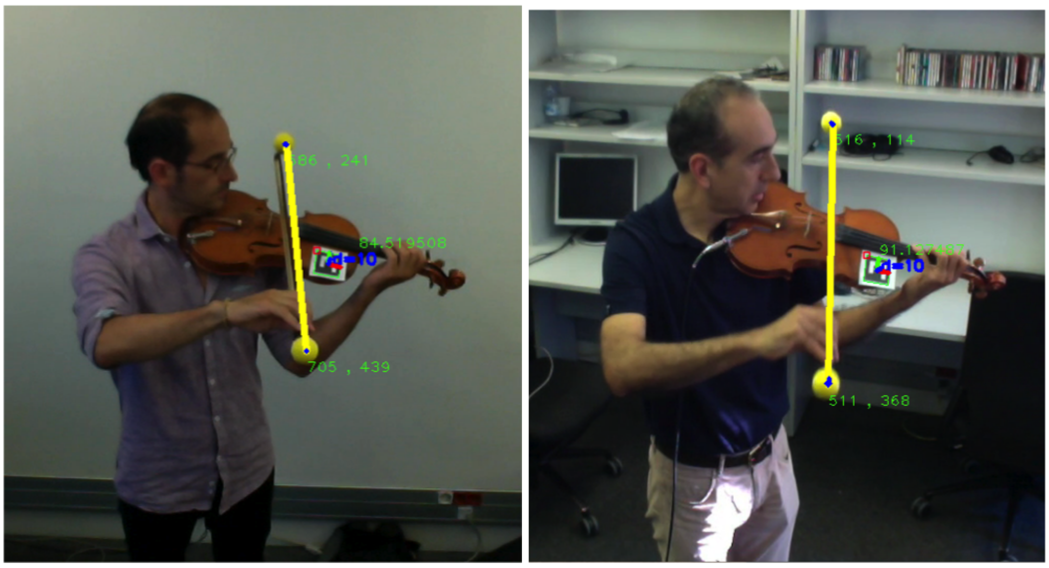
$$T_{vec} = (T_x, T_y, T_z) \tag{3.13}$$

Regarding our application, these parameters allow us to estimate the camera pose with respect to the center of the fiducial marker, as the 3D transformation from the marker coordinate system to the camera coordinate system. For retrieving the intrinsic and extrinsic parameters of our camera, we recorded a video with multiple perspectives of an asymmetrical circle pattern that was later fed to the OpenCV calibration algorithm. As a result, the algorithm returns an *.xml file that contains all the required camera calibration parameters. Once we obtain the intrinsic camera parameters we are able to use them whenever it is required.

After computing the calibration features, we continue with the detection process of the ArUco marker. As it presented in Figure 3.9b, our algorithm starts by receiving an incoming video frame that is converted into grayscale color space in order to apply an adaptive threshold for obtaining the various line contours, including the real marker's contours but also many other undesired borders. Hence, the marker detection step aims on filtering out all the unwanted borders, by removing the contours that do not approximate to a square shape in addition to those that consist of few points. Furthermore, the contours that are too close to each other are also discarded, since the adaptive threshold normally detects both the internal and external contours of the marker border and up to this stage, we are interested in keeping only the external border.

Later to the candidate detection, it is necessary to determine if they are actually markers by analyzing their inner codification. Firstly, perspective transformation is applied by using homography so as to obtain the frontal view of the marker in its canonical form. Then, the canonical image is thresholded with the Otsu's algorithm in order to identify the internal marker code by separating the white and the black bits. Moreover, the maker image is divided in a 6x6 grid, where the internal 5x5 cells contains the code information, while the rest correspond to the external black border. Then the bits are analyzed to determine if the marker belongs to a valid marker dictionary and if necessary error correction techniques are employed, by means of corner refinement with sub-pixel interpolation. Finally, by utilizing the *.xml file that contains the camera parameters from the calibration step, we are able to proceed with the computation of the extrinsic parameters for estimating the marker's pose with respect to the camera orientation.

Even though, the detection algorithms for the ArUco and color markers are described separately, their implementation is combined in a single application. For



(a) Low brightness. (b) High brightness

Figure 3.12: Examples of the video analysis algorithms for the detection of the ArUco and color markers, in different lighting conditions.

instance, Figure 3.12 presents the outcome of our algorithm in different lighting conditions.

3.3 Audio analysis

For analyzing the recorded audio signals we needed to apply music information retrieval (MIR) methods so as to acquire meaningful results. Numerous software libraries have been proposed over the last years to facilitate research and applications development in MIR. Most of the libraries concentrate on extracting low-level features from the audio signal, in addition to higher level features such as tonal and rhythmic descriptors. Some examples of the available MIR libraries include the Marsyas framework [Tzanetakis and Lemstrom, 2007] that is based on C++, MIRToolbox [Lartillot and Toiviainen, 2007] that contains a set of Matlab methods, SCMIR[Collins, 2011] which utilizes the SuperCollider programming language and the Python-based Librosa [McFee et al., 2015]. In our project we employed the Essentia library,[Bogdanov et al., 2013], since it comes with an open-source license and it is well maintained with a online documentation.

Essentia is a C++ library, which was developed within our department and it is used for audio analysis and audio-based MIR applications. It consists of an extensive collection of reusable algorithms, including audio input/output function-

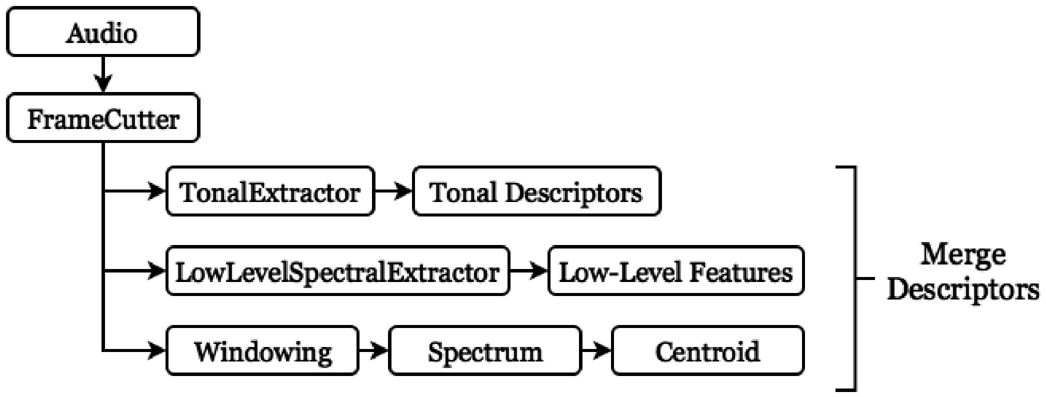


Figure 3.13: .

alities, standard digital signal processing methods, statistical characterization of data, as well as a large set of spectral, temporal, tonal and high-level music descriptors. However, Essentia is not considered a framework, but rather a collection of algorithms that is wrapped in a programming library. Furthermore, it focuses on the robustness, performance and optimality of the provided algorithms, as well as its simplicity in building custom audio processing methods. The flow of the analysis is decided and implemented by the user, while Essentia is responsible for the implementation details of the employed algorithms.

The library is also wrapped in Python and includes a number of predefined extractors for the available music descriptors as binary executables, which further promotes fast prototyping and rapid implementation of research experiments. However, they should be considered as examples and not as the only correct way of utilizing the Essentia functionalities. Additionally, there are various third-party extensions of Essentia that enable its utilization within the frameworks of PureData, Max/MSP, openFrameworks, and Matlab. Furthermore, it can be be cross-compiled to JavaScript for developing web-based applications, as well as a Vamp plug-in within Sonic Visualiser so as to visualize the outcomes of the provided algorithms. Hence, the library is considered to be cross-platform, since it supports Linux, Mac OS X, Windows, iOS and Android OSs.

Consequently, our audio analysis algorithm was implemented in Python due to its syntax simplicity during the programming process, compered to C++. Furthermore we employed the Jupyther Notebook[5] web application for the Python development, since it supports live coding with direct feedback. As it is illustrated in Figure 3.13 our algorithm, starts by loading a given audio file with the MonoLoader function that downmixes the the stereo signal to a mono channel, while resampling it to a given sample rate, with a default frequency of 44.1 KHz.

[5]http://jupyter.org/index.html

Then, we apply the FrameCutter method in order to slice the input signal to the appropriate number of frames so as to retrieve the same analysis resolution with the video frame rate. Therefore, we choose a window size of 2942 samples with 50% overlap, in order to match the 30 fps frequency of the video recordings.

Next we follow three different approaches for computing the various descriptors with respect to their abstraction level. For calculating the tonal features we employed the predefined TonalExtractor method that computes a collection of tonal-related features, including chords changes rate, chords histogram, chords key, chords number rate, chords progression, chords scale, chords strength, Harmonic Pitch Class Profile (HPCP), key scale and key strength. However, for our study we take into account only the HPCP and the key tonal features. In a similar sense, we computed most of the low-level descriptors by utilizing the predefined LowLevelSpectralExtractor algorithm which calculates a set of low-level audio features, such as barkbands, kurtosis, skewness, spread, high frequency content (HFC), Mel-frequency cepstral coefficients (MFCCs), pitch, pitch instantaneous confidence, pitch salience, silence rate, complexity, crest, decrease, energy, low energy band (20 - 150 HZ), middle-low energy band (150 - 800 Hz), middle-high energy band (800 - 4000 Hz), high energy band (4000 - 20000 Hz), flatness, flux, spectral root mean square (RMS), roll-off, strong-peak, zero crossing rate, inharmonicity, tristimulus and odd to even harmonic energy ratio. The third approach calculates the spectral centroid by applying a hanning window on the sliced frames, then it computes the spectrum magnitude of the window and finally its centroid.

As a last step of the audio analysis phase, the extracted descriptors of the aforementioned algorithms are merged and saved as plane text in a CSV file format. Each line of the outcome file represents each individual frame and consists of numerous fields equal to the total amount of the previously computed tonal and low-level features.

3.4 Post-processing and dataset creation

Before proceeding to the evaluation phase, we first need to concatenate the results from the video and audio analysis into one data file. Furthermore, it is critical to ensure that there are no noisy or null records within our dataset, since they affect directly the quality and accuracy of the evaluation results. Therefore, we employed the Unix bash shell for developing small scripts and executing the appropriate commands for filtering the two CSV files and merging them into one dataset. We chose to use the bash shell since it allow us to execute the required filtering conveniently with a single line of code.

Chapter 4

EVALUATION AND RESULTS

In this Chapter, we will describe all the methods that were involved in the evaluation procedure of our study, in addition to the acquired results. More specifically, we present the details of the machine learning methods that were applied on our dataset, for investigating the possible relations between the audio descriptors and the bowing features, as computed by our low-cost tracking system. Next, we list the retrieved regression accuracies from various machine learning algorithms and discuss on the given results.

4.1 Evaluation

After the creation of our final dataset that contains the filtered acquisition information from the video and sound domains, we were interested into researching the possible connections amongst the low-level and tonal descriptors with the bowing parameters as they were computed by our video analysis algorithm. For this purpose, we employed the Weka Data Mining[1] software [Hall et al., 2009], which is an open source application specific for machine learning tasks.

Furthermore, Weka provides a collection of algorithms for data pre-processing, classification, regression, clustering, association rules, visualization and predictive modeling, which are written in Java programming language. Therefore, by taking advantage of the Java Virtual Machine (JVM) and its portability property, Weka is capable of running on almost any modern computer platform. Additionally, it comes with a GUI that enables direct application of the available algorithms on a given dataset, which further facilitates the overall working process. However, Weka can be also imported as an external library, so as to enable the developers to access its algorithms as call functions within a custom Java program. The imported dataset is required to be contained in a single file, where each data record

[1]http://www.cs.waikato.ac.nz/ml/weka/

is described by a fixed number of attributes that are either numeric or nominal.

Regarding our study, the evaluation process consists of two steps. Firstly we applied a feature selection method in order to minimize the dimensionality of our research problem and possible over-fitting. Then we performed a regression analysis on the selected features by utilizing four different machine learning algorithms, so as to estimate the relationships between the audio and bowing gestural features.

4.1.1 Feature selection

Generally, *feature selection* or *attribute selection* is the process of automatically searching for the best subset of attributes in a given dataset. The concept of "best subset" is relative to the subjective problem that we try to solve, but typically reflects to higher accuracy. The hypothesis behind the feature selection technique is that usually the dataset contains many attributes which are either redundant or irrelevant, hence they can be removed without causing information loss. However, it is important to mention that redundant and irrelevant features are two distinct states, since one relevant feature may be redundant in the presence of another relevant feature with which it is strongly correlated.

Therefore, the goal is to navigate through the feature space and locate the best or a good enough combination of features that improves the performance of our predictive model, over selecting the total amount of attributes. By applying feature selection to our dataset we avoid over-fitting due to the existence of less redundant features. Additionally, the absence of noisy attributes results to more meaningful models with higher accuracy confidence. Furthermore, by minimizing the amount of data we also reduce the training time of our predictive model. In Weka, the attribute selection process is separated in two parts. Firstly we need to specify the attribute evaluator algorithm, which is responsible for assessing the various attributes subsets. On the second step it is required to select the search method that is in charge of searching the feature space for possible subsets according to the results of the subset evaluation method.

In this sense, for our dataset we applied three times the feature selection procedure, since we were interested to investigate the best attributes subset for three different classes, which represent the two velocities of the two color markers in addition to the bow inclination. Furthermore, we selected the *CfsSubsetEval* algorithm as attribute evaluator that assesses the worth of a subset of features by considering the individual predictive ability of each feature, along with the degree of redundancy between them. In other words, good subsets are considered those that are highly correlated with the class while having low intercorrelation. Additionally, we enabled the *locallyPredictive* property, which iteratively adds the attributes with the highest class correlation, as long as there is not already another attribute in the subset that has a higher correlation compered to the attribute in question. As for the search method, we utilized the *BestFirst* algorithm, which searches forward the space of attributes subsets by starting with an empty set. The algorithm is also equipped with a backtracking mechanism by setting the number of consecutive non-improving nodes that are allowed before terminating the search. The quality of the attribute subset is determined by a process of 10-fold cross-validation, where the original dataset is randomly partitioned into 10 equal sized subsamples. A random subsample is selected as the validation data for testing the model, and the remaining 9 are used as training data.

By applying the aforementioned attribute selection process for the Velocity A and Velocity B as target classes, we retrieved the same 7 features, including the MFCC1, MFCC5, MFCC7, HFC, spectral crest, flux, and strong-peak out of the total 37 low-level descriptors that were contained in our dataset. In a similar manner, we acquired 10 features for the Bow Inclination as the predicted class, specified by the MFCC2, MFCC4, MFCC6, MFCC11, MFCC12, barkbands spread, pitch, pitch instantaneous confidence, spectral crest and Inharmonicity low-level audio descriptors.

4.1.2 Regression analysis with machine learning

The regression analysis is a famous statistical process for estimating the relationships between variables. This method is widely used for solving prediction problems, thus including many techniques for modeling and analyzing multiple variables. The main target of regression is to investigate the relationships between a dependent attribute, which is the target class, and many independent variables, which are also known as predictors. More specifically, regression analysis aids on the understanding of how the typical value of the dependent feature changes, when there is a variance in one independent variable, while the rest independent variables are kept with fixed values. Therefore regression analysis can be performed over machine learning algorithms, due to the substantial overlap on their research interest.

Considering our study, we employed four different machine learning approaches for computing the regression correlations amongst the audio descriptors and the bowing features, including *Decision Trees* (DTs), *k-Nearest Neighbor* (kNN), *Support Vector Machines* (SVMs) and *Multilayer Perceptron* that are also known as Neural Networks (NNs). These algorithms are briefly described as follows, in terms of how they demonstrate along with their key parameters and their implementation details within Weka.

Decision Trees

Decision Trees that are also known as Classification and Regression Trees (CART), are methods for constructing prediction models from data. The models are computed by recursively partitioning the data space and fitting a smaller prediction model within each partition. Consequently, the partitioning process can be represented graphically as a decision tree that starts from the root class and moves down to the attribute leaves until a prediction is achieved. The process of creating a decision tree works by greedily selecting the best split-point in order to make predictions and iterate until the tree reaches a fixed depth. More specifically, the regression trees start from a dependent variables that take continuous or discrete ordered values, while the prediction error is typically measured by the squared difference between the observed and the predicted values. After the tree construction, we can prune its leaves in order to improve and generalize the model's performance to new data.

For our study we selected to work with the *M5Rules* algorithm, that generates a decision list for regression problems by using the divide-and-conquer approach. This algorithm in each iteration builds a model tree and makes the "best" leaf into a rule. Furthermore we applied this method by keeping the random parameter values in addition to a 10-fold cross-validation for evaluating the model's performance.

k-Nearest Neighbor

The k-Nearest Neighbors algorithm works by storing the entire training dataset and querying it to locate the k most similar training patterns when making a prediction. Therefore, there is no model other than the raw training dataset and the only computation that is performed is the querying of the training data whenever a prediction is requested. It is a simple algorithm that under account only the distance between the data instances for providing meaningful predictions. As such, it often achieves high performance with very good accuracy rates. In the case of predicting regression problems, the kNN computes the mean of the k most similar instances in the training dataset. Moreover, the size of the neighborhood is specified by the k parameter. For instance, when it set to 1, the predictions are made using the single most similar training instance to a given new pattern for which a prediction is requested. Common values for k are 3, 7, 11 and 21 with respect to the dataset size.

In Weka the kNN technique is implemented with the *IBk* algorithm, which stands for Instance Based k. This method can automatically discover a good value for the k, by using cross-validation inside the algorithm as specified in the *cross-Validate* parameter. Another important setting is the selected distance measure, which can be configured in the *nearestNeighbourSearchAlgorithm* field that further controls the way in which the training data is stored and searched. The default method is the *LinearNNSearch*. Furthermore, we can specify the distance function that is used by the search algorithm in order to calculate the distance between the data instances. By default, Weka uses the euclidean distance for computing this measurement. In this sense, we employed the IBk algorithm with default settings and selected 10-fold cross-vilidation for our training phase.

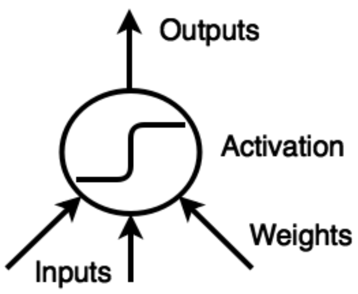
SVMs

The SVMs were initially developed for solving binary classification problems, although extensions to the method have been made for supporting multi-class classification and regression problems. The adjustment of the SVM to regression analysis is also known as Support Vector Regression (SVR). SVMs were implemented for working with numerical input variables, while providing a normalization function in addition to a conversion mechanism that translates the nominal values to numerical.

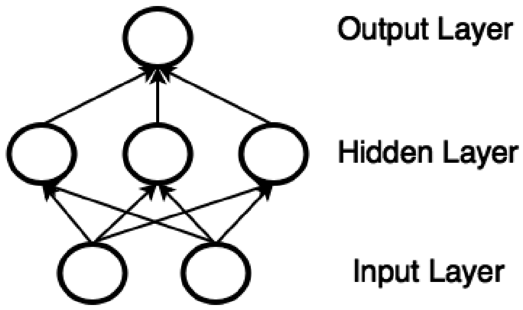
The basic idea behind SVM is to find a line that separates in the best possible way the training data into different classes, while SVR works by finding a line of best fit that minimizes the error of the cost function. This is done by following an optimization procedure that only considers those data instances in the training set that are closest to the line with the minimum cost. These instances are called support vectors and give their name to the main technique. However, in real problems it is difficult to draw a line that best fits the data. Therefore a margin is added around the line in order to relax the constraint that further allows some bad predictions to be tolerated. In this sense the generalization error of the classifier is minimized while providing better overall results. Furthermore, few datasets can be fit with just a straight line. Sometimes a line with curves or even polygonal regions need to be marked out. This is achieved by projecting the data into a higher dimensional space in order to draw the lines and make the appropriate predictions, according to the flexibility of the employed kernel. In our study we utilized the *SMOreg* algorithm as implemented in Weka, by keeping the random parameter values in addition to a 10-fold cross-validation for evaluating the model's performance.

Multilayer Perceptron

The Multilayer Perceptron or Neural Network algorithm is a complex method for predictive modeling due to its greate diversity of the configuration parameters that can only be tuned effectively through intuition and many iterations. Moreover, it is a method that was inspired by the model of biological neural networks in



(a) Structure of a neuron.



(b) Example of a neural network.

Figure 4.1: Illustrations of the neuron structure in addition to an examples of the different layers involved in a neural network.

the brain, where small processing units called neurons are organized into layers that if configured well are capable of approximating any prediction function. In classification we are interested in approximating the underlying function that best discriminates the various classes, while in regression problems we are interested in approximating a function that best fits the real value output. In other words, a neural network has the ability to learn the data representation from the training set and how to best relate it to the output variable which is subjected for prediction.

As it is illustrated in Figure 4.1a, the building block of a neural network are the artificial neurons or simply nodes. They are small computational units that receive weighted input signals and produce an output signal according to an activation function. Furthermore, the input weights are very much like the coefficients of a regression equation, in addition to a bias which can be thought as an input that always has the value 1. For instance, a neuron may have two inputs but it requires three weights, one for each individual input and one for the bias. Typically, the weights are randomly initialized with small values, between the range of 0 to 0.3, however more complex initialization patterns can be used. On the other hand, by applying large weights we increase the complexity and fragility of the network. Hence, it is better to apply regularization techniques and keep weights within a small range. After the weighting process, the neuron receives the inputs and applies an activation function on their sum. This function is a simple mapping of the sum of the weighted inputs to the output of the neuron. It is called an activation function because it specifies a threshold at which the neuron is activated in order to reinforce the output signal. A famous activation functions is the sigmoid which outputs a value between 0 and 1 with an s-shaped distribution, in addition to the hyperbolic tangent function that outputs the same distribution within the range -1 to +1.

However, the neural networks consist of multiple nodes which are arranged in rows that are called layers. The overall architecture of neurons in the network is known as network topology, and consists of three types of layers including the input, the hidden and the output layers, as it is depicted in Figure 4.1b. The first layer that receives the input from the given dataset is the input or visible layer. Typically, a neural network has a visible layer with one neuron per input value or attribute of the dataset. However, these neurons do not match to the above description, since they simply pass the incoming value to the hidden layer. They are called hidden because they are not directly exposed to the user. The simplest network structure contains a single neuron in the hidden layer that directly outputs the value. The final hidden layer is called the output layer and it is responsible for outputting a value or vector of values, according to the problem requirements.

Weka implements NNs with the *MultilayerPerceptron* algorithm that allows the user to manually specify the structure of the employed neural network. However, by keeping the default settings, Weka will automatically design the network and train it with a single hidden layer network. The nodes in this network are all sigmoid, except in the case where the class is numeric and the output nodes become unthresholded linear units. Additionally, it is equipped with a configurable learning process by setting a learning rate that specifies the updating amount of the weights. The learning process can be further tuned with a decay attribute, which causes the learning rate to decrease over time. More specifically, it divides the starting learning rate with the epoch number so as to determine what learning rate should be applied. Hence it performs more learning at the beginning of training and less at the end. This method helps to stop the network from diverging from the target output, as well as it improves the general performance. For the evaluation task of our study we kept the random parameter values in addition to a 10-fold cross-validation training procedure.

4.2 Results

After applying the attribute selection process, we demonstrated the four machine learning methods for investigating the regression correlations between the bowing features, as computed from our video analysis algorithms, and the audio descriptors, which were extracted from the built-in microphone recordings with the Essentia algorithms. Since our study is focused on the analysis of the low-cost system, we did not take under account the tracking information of the EMF system and the recorded signal of the bridge pickup.

More specifically, we applied the same machine learning procedure for predicting the three video features, including the VelocityA, VelocityB and Inclination bowing parameters. Additionally, we researched the regression accuracies over the complete dataset, in addition to its reduced version, as retrieved from the feature selection method with respect to the different target classes. The retrieved accuracies are presented in Tables 4.1, 4.2 and 4.3.

Generally, we can observe that VelocityA provides the lowest accuracy rates, even when it is compered with the accuracies of VelocityB. However this may happen due to the smaller distance of the lower marker to the source of the bow motion, which is the hand of the performer. Also, we can see that by selecting all the audio features for estimating the regression correlations results to overfitting, with a clear example the performance of the IBk algorithm for predicting the VelocityA. There the retrieved accuracy of the complete feature space is almost double from the reduced one. Additionaly, the same regression method with prediction class the VelocityB has 15% to 18%lower performance compered to the other machine learning algorithms. Paradoxically, when the IBk was employed for estimating the Bow Inclination feature, resulted to the best accuracies of 95% and 97% (see Table 4.3). In particular, the Bow Inclination feature is highly correlated with the audio descriptors, which is reasonable, since this feature is directly related to the current sting played that further affects directly the values of the low-level audio descriptors.

Velocity A		
Machine Learning Algorithm	All features	Selected Attributes
M5 Rules	0.52	0.50
IBk	0.67	0.34
SMOreg	0.53	0.48
Multilayer Perceprton	0.43	0.52

Table 4.1: Table of regression accuracies of the various machine learning algorithms for predicting the VelocityA bow feature.

Velocity B		
Machine Learning Algorithm	All features	Selected Attributes
M5 Rules	0.62	0.60
IBk	0.45	0.42
SMOreg	0.62	0.57
Multilayer Perceprton	0.54	0.58

Table 4.2: Table of regression accuracies of the various machine learning algorithms for predicting the VelocityB bow feature.

Bow Inclination		
Machine Learning Algorithm	All features	Selected Attributes
M5 Rules	0.94	0.93
IBk	0.97	0.95
SMOreg	0.79	0.79
Multilayer Perceprton	0.89	0.87

Table 4.3: Table of regression accuracies of the various machine learning algorithms for predicting the Bow Inclination feature.

Chapter 5

CONCLUSIONS AND DISCUSSION

With this work we present a novel approach in capturing violin performance gestures by employing both direct and indirect acquisition methods based on a low-cost system. According to the bibliography, most of the available proposals fall into the direct acquisition technique, which requires either to develop custom electronic devices or commercial MoCap systems that usually increase the overall cost. Moreover, the freedom of movements is crucial for musical expression and these systems invoke some kind of intrusiveness that may have an effect on the overall performance abilities of the artist. However, there are studies that address the acquisition problem by utilizing indirect methods for gestural estimation from the audio signal, rather than the physical domain. Also, hybrid systems have been developed, but still they entail complex set-ups.

Therefore, our goal was to develop a hybrid system over low-cost equipment that is available with the modern personal devices. We employed the built-in camera and microphone of a laptop for developing our system algorithms that are based on video frame analysis, augmented reality and audio signal processing methods. For the video analysis we employed color markers for recognizing the bow, as well as an AR marker for estimating the violin pose with respect to the camera coordinate system. Next we extracted the various tonal and low-level spectral audio descriptors of the built-in microphone's input signal. Then we unified the retrieved data from the two analysis processes in a single filtered dataset in order to proceed with the evaluation. Following, we applied attribute selection techniques and machine learning methods for regression in order to investigate the possible relations amongst the audio features and the bowing controls, as they were computed from our algorithms.

The results are promising, since they show high correlation between the Bow Inclination parameter and the audio features, with maximum accuracy of 97%.

Additionally, the video analysis algorithm performed well in different lighting conditions, in addition to its ability to run in real-time. Furthermore, the utilization of styrofoam color markers and a simple printed AR marker are almost transparent to the player during a real violin performance without affecting his freedom of movements. Finally we can say that we opened the door for further research in the gesture recognition field of study, since we achieved intuitive direct acquisition of complex 3D movements from a single camera perspective with simple tools and methods.

5.1 Reproducibility

Throughout the designing, development and evaluation phases of this project, we aimed on utilizing open source software and freewares in order to promote the reproducibility aspect of our study. Additionally, we achieved to develop a platform-independent application by taking advantage of the programming power of the C++ language in conjunction with the cross-platform implementations of the OpenCV and Essentia software libraries. Also, the code that we developed is open to the public since it hosted on GitHub[1] as an online repository with GNUv3 General Public license. The link to the repository is specified by the following address:

`https://github.com/kosmasK/ViolinBowTracking.git`

5.2 Possible extensions

Due to the novelty of our system, there are many improvements that can be addressed as future work. Regarding the already implemented video analysis algorithm, it can be enriched with extra computational formulas for estimating additional bowing parameters, such as the bow skewness and distance from bridge. Furthermore, we plan to develop a GUI that would enable the user to intuitively calibrate and configure the system, as well as monitoring his performance through real-time visualizations. Considering the multimodal data that we recorded with the EMF system and the bridge pickup, can be further used as ground truth with which we can revaluate the overall performance of our system. Moreover, we plan to share online the multimodal recordings over the RepoVizz[2] web service, as a reproducibility extension.

[1]https://github.com/

[2]http://repovizz.upf.edu/repo/Home

Bibliography

[Askenfelt, 1989] Askenfelt, A. (1989). Measurement of the bowing parameters in violin playing. ii: Bow–bridge distance, dynamic range, and limits of bow force. *The Journal of the Acoustical Society of America*, 86(2):503–516.

[Bevilacqua et al., 2011] Bevilacqua, F., Schnell, N., Fdili Alaoui, S., Klein, G., and Noeth, S. (2011). Gesture capture: Paradigms in interactive music/dance systems. *Emerging Bodies: The Performance of Worldmaking in Dance and Choreography*, 183:183–193.

[Bogdanov et al., 2013] Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J., and Serra, X. (2013). Essentia: an open-source library for sound and music analysis. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 855–858. ACM.

[Bradski, 2000] Bradski, G. (2000). opencv library. *Dr. Dobb's Journal of Software Tools*.

[Bregler, 2007] Bregler, C. (2007). Motion capture technology for entertainment [in the spotlight]. *IEEE Signal Processing Magazine*, 24(6):160–158.

[Burger and Toiviainen, 2013] Burger, B. and Toiviainen, P. (2013). MoCap Toolbox – A Matlab toolbox for computational analysis of movement data. In Bresin, R., editor, *Proceedings of the 10th Sound and Music Computing Conference*, pages 172–178, Stockholm, Sweden. KTH Royal Institute of Technology.

[Cadoz, 1988] Cadoz, C. (1988). Instrumental gesture and musical composition. In *ICMC 1988-International Computer Music Conference*, pages 1–12.

[Camurri et al., 2007] Camurri, A., Coletta, P., Varni, G., and Ghisio, S. (2007). Developing multimodal interactive systems with eyesweb xmi. In *Proceedings of the 2007 Conference on New Interfaces for Musical Expression (NIME07)*, pages 302–305. ACM.

[Collins, 2011] Collins, N. (2011). Scmir: A supercollider music information retrieval library. In *Proceedings of the International Computer Music Conference*, pages 499–502.

[Dahl et al., 2007] Dahl, L., Whetsell, N., and Van Stoecker, J. (2007). The wavesaw: a flexible instrument for direct timbral manipulation. In *Proceedings of the 7th international conference on New Interfaces for Musical Expression*, pages 270–272. ACM.

[Demoucron et al., 2009] Demoucron, M., Askenfelt, A., and Caussé, R. (2009). Measuring bow force in bowed string performance: Theory and implementation of a bow force sensor. *Acta Acustica united with Acustica*, 95(4):718–732.

[Demoucron and Causse, 2007] Demoucron, M. and Causse, R. (2007). Sound synthesis of bowed string instruments using a gesture based control of a physical model. In *International Symposium on Musical Acoustics*, pages 1–1.

[Fan and Essl, 2013] Fan, X. and Essl, G. (2013). Air violin: A body-centric style musical instrument. *Ann Arbor*, 1001:48109–2121.

[Fischer, 1997] Fischer, S. (1997). *Basics: 300 exercises and practice routines for the violin*, volume 7440. Edition Peters.

[Galamian and Thomas, 2013] Galamian, I. and Thomas, S. (2013). *Principles of violin playing and teaching*. Courier Corporation.

[Garrido-Jurado et al., 2014] Garrido-Jurado, S., noz Salinas, R. M., Madrid-Cuevas, F., and Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280 – 2292.

[Gillian and Paradiso, 2014] Gillian, N. and Paradiso, J. A. (2014). The gesture recognition toolkit. *Journal of Machine Learning Research*, 15:3483–3487.

[Giraldo and Ramirez, 2013] Giraldo, S. and Ramirez, R. (2013). Brain-activity-driven real-time music emotive control. In *Proceedings of the 3rd International Conference on Music & Emotion (ICME3), Jyväskylä, Finland, 11th-15th June 2013. Geoff Luck & Olivier Brabant (Eds.). ISBN 978-951-39-5250-1*. University of Jyväskylä, Department of Music.

[Goudeseune, 2001] Goudeseune, C. (2001). *Composing with parameters for synthetic instruments*. PhD thesis, University of Illinois at Urbana-Champaign.

[Guaus et al., 2009] Guaus, E., Blaauw, M., Bonada, J., and Maestre, E. (2009). Pérez,?a calibration method for accurately measuring bow force in real violin performance,? In *Proc. 2009 Int. Comput. Music Conf.*

[Guaus et al., 2007] Guaus, E., Bonada, J., Perez, A., Maestre, E., and Blaauw, M. (2007). Measuring the bow pressing force in a real violin performance. In *Proceedings of International Symposium on Musical Acoustics*.

[Hadjakos, 2012] Hadjakos, A. (2012). Pianist motion capture with the kinect depth camera. In *Proceedings of the International Conference on Sound and Music Computing, Copenhagen, Denmark*.

[Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

[Hantrakul and Kaczmarek, 2014] Hantrakul, L. and Kaczmarek, K. (2014). *Implementations of the Leap Motion in sound synthesis, effects modulation and assistive performance tools*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library.

[Hodgson, 1958] Hodgson, P. (1958). *Motion study and violin bowing*. American String Teachers Associations.

[Jackson et al., 1987] Jackson, B. G., Berman, J., and Sarch, K. (1987). Dictionary of bowing terms for string instruments.

[Jensenius and Voldsund, 2012] Jensenius, A. R. and Voldsund, A. (2012). The music ball project: Concept, design, development, performance. In *Proceedings of the international conference on New Interfaces for Musical Expression*, pages 300–303.

[Jordà et al., 2007] Jordà, S., Geiger, G., Alonso, M., and Kaltenbrunner, M. (2007). The reactable: Exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*, TEI '07, pages 139–146, New York, NY, USA. ACM.

[Kereliuk et al., 2007] Kereliuk, C., Scherrer, B., Verfaille, V., Depalle, P., and Wanderley, M. M. (2007). Indirect acquisition of fingerings of harmonic notes on the flute. In *Proceedings of the International Computer Music Conference, ICMC ?07, Copenhagen, Denmark*, volume 1, pages 263–266.

[Lartillot and Toiviainen, 2007] Lartillot, O. and Toiviainen, P. (2007). A matlab toolbox for musical feature extraction from audio. In *International Conference on Digital Audio Effects*, pages 237–244.

[Ltd, 2016] Ltd, V. M. S. (2016). Vicon – motion capture systems.

[Maestre, 2009] Maestre, E. (2009). Modeling instrumental gestures: an analysis/synthesis framework for violin bowing. *Department of Information and Communication Technologies*.

[Maestre et al., 2007] Maestre, E., Bonada, J., Blaauw, M., Perez, A., and Guaus, E. (2007). Acquisition of violin instrumental gestures using a commercial emf tracking device. In *Proceedings of the 2007 International Computer Music Conference (ICMC07)*, volume 1, pages 386–393.

[Mathworks, 2016] Mathworks (2016). What is camera calibration.

[McFee et al., 2015] McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th Python in Science Conference*.

[McMillen, 2016] McMillen, K. K. (2016). K-bow.

[Metois, 1996] Metois, E. (1996). *Musical Sound Information: Musical Gesture and Embedding Synthesis (Psymbesis)*. PhD thesis, Thesis for MIT Media Laboratory.

[Miller, 2001] Miller, L. (2001). Cage, cunningham, and collaborators: The odyssey of variations v. *The Musical Quarterly*, 85(3):545–567.

[Mitchell, 2011] Mitchell, T. J. (2011). Soundgrasp: A gestural interface for the performance of live music. In *Proceedings of the international conference on New Interfaces for Musical Expression*.

[Morreale et al., 2014] Morreale, F., De Angeli, A., Masu, R., Rota, P., and Conci, N. (2014). Collaborative creativity: The music room. *Personal and Ubiquitous Computing*, 18(5):1187–1199.

[Niinimäki and Tahiroglu, 2012] Niinimäki, M. and Tahiroglu, K. (2012). Ahne: a novel interface for spatial interaction. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, pages 1031–1034. ACM.

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

[Penttinen and Välimäki, 2004] Penttinen, H. and Välimäki, V. (2004). A time-domain approach to estimating the plucking point of guitar tones obtained with an under-saddle pickup. *Applied Acoustics*, 65(12):1207–1220.

[Perez-Carrillo and Wanderley, 2012] Perez-Carrillo, A. and Wanderley, M. M. (2012). Learning and extraction of violin instrumental controls from audio signal. In *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, pages 25–30. ACM.

[Perez-Carrillo and Wanderley, 2015] Perez-Carrillo, A. and Wanderley, M. M. (2015). Indirect acquisition of violin instrumental controls from audio signal with hidden markov models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(5):932–940.

[Polhemus, 2016] Polhemus (2016). Polhemus liberty.

[Ramirez et al., 2015] Ramirez, R., Palencia-Lefler, M., Giraldo, S., and Vamvakousis, Z. (2015). Musical neurofeedback for treating depression in elderly people. *Frontiers in neuroscience*, 9.

[Rasamimanana et al., 2005] Rasamimanana, N. H., Fléty, E., and Bevilacqua, F. (2005). Gesture analysis of violin bow strokes. In *International Gesture Workshop*, pages 145–155. Springer.

[Reidsma et al., 2008] Reidsma, D., Nijholt, A., and Bos, P. (2008). Temporal interaction between an artificial orchestra conductor and human musicians. *Computers in Entertainment (CIE)*, 6(4):53.

[Sarasúa, 2013] Sarasúa, Á. (2013). Context-aware gesture recognition in classical music conducting. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 1059–1062. ACM.

[Schoonderwaldt, 2009] Schoonderwaldt, E. (2009). Mechanics and acoustics of violin bowing. *Unpublished doctoral dissertation). Stockholm: KTH.*

[Schoonderwaldt, 2016] Schoonderwaldt, E. (2016). Bowing parameters - bowing3d.

[Schoonderwaldt and Demoucron, 2009] Schoonderwaldt, E. and Demoucron, M. (2009). Extraction of bowing parameters from violin performance combining motion capture and sensors. *The Journal of the Acoustical Society of America*, 126(5):2695–2708.

[Schoonderwaldt et al., 2006] Schoonderwaldt, E., Rasamimanana, N., and Bevilacqua, F. (2006). Combining accelerometer and video camera: Reconstruction of bow velocity profiles. In *Proceedings of the 2006 conference on New interfaces for musical expression*, pages 200–203. IRCAM?Centre Pompidou.

[Smyth and Abel, 2012] Smyth, T. and Abel, J. S. (2012). Toward an estimation of the clarinet reed pulse from instrument performance. *The Journal of the Acoustical Society of America*, 131(6):4799–4810.

[Smyth and Scott, 2011] Smyth, T. and Scott, F. S. (2011). Trombone synthesis by model and measurement. *EURASIP Journal on Advances in Signal Processing*, 2011(1):1–13.

[Smyth and Wang, 2014] Smyth, T. and Wang, C.-i. (2014). *Toward Real-Time Estimation of Tonehole Configuration*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library.

[Thurber et al., 2010] Thurber, M. R., Bodenhamer-Davis, E., Johnson, M., Chesky, K., and Chandler, C. K. (2010). Effects of heart rate variability coherence biofeedback training and emotional management techniques to decrease music performance anxiety. *Biofeedback*, 38(1):28–40.

[Traube et al., 2003] Traube, C., Depalle, P., and Wanderley, M. (2003). Indirect acquisition of instrumental gesture based on signal, physical and perceptual information. In *Proceedings of the 2003 conference on New interfaces for musical expression*, pages 42–47. National University of Singapore.

[Traube and Smith, 2001] Traube, C. and Smith, J. O. (2001). Extracting the fingering and the plucking points on a guitar string from a recording. In *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pages 7–10. Citeseer.

[Tzanetakis and Lemstrom, 2007] Tzanetakis, G. and Lemstrom, K. (2007). Marsyas-0.2: a case study in implementing music information retrieval systems. *Intelligent Music Information Systems. IGI Global*, 14.

[Vamvakousis and Ramirez, 2012] Vamvakousis, Z. and Ramirez, R. (2012). Temporal control in the eyeharp gaze-controlled musical interface. In *New Interfaces for Musical Expression (NIME) 2012*, Ann Arbor - Michigan.

[Visentin and Shan, 2011] Visentin, P. and Shan, G. (2011). Applications of emg pertaining to music performance-a review. *Arts BioMechanics*, 1(1):15.

[Wanderley, 2001] Wanderley, M. M. (2001). Gestural control of music. In *International Workshop Human Supervision and Control in Engineering and Music*, pages 632–644.

[Wanderley and Depalle, 2004] Wanderley, M. M. and Depalle, P. (2004). Gestural control of sound synthesis. *Proceedings of the IEEE*, 92(4):632–644.

[Wechsler et al., 2004] Wechsler, R., Weiß, F., and Dowling, P. (2004). EyeCon: A motion sensing tool for creating interactive dance, music and video projections. In *Proceedings of the Society for the Study of Artificial Intelligence and the Simulation of Behaviour*.

[Young, 2002] Young, D. (2002). The hyperbow: a precision violin interface. In *International Computer Music Conference, Göteborg*.

[Young, 2007] Young, D. (2007). *A methodology for investigation of bowed string performance through measurement of violin bowing technique*. PhD thesis, Citeseer.