

# ESP: A Driving Interface for Expression Synthesis

Elaine Chew, Alexandre François, Jie Liu, Aaron Yang  
University of Southern California Viterbi School of Engineering  
Integrated Media Systems Center, Los Angeles CA 90089-0193  
{echew, afrancoi, jieliu, ayang}@usc.edu

## ABSTRACT

In the Expression Synthesis Project (ESP), we propose a driving interface for expression synthesis. ESP aims to provide a compelling metaphor for expressive performance so as to make high-level expressive decisions accessible to non-experts. In ESP, the user drives a car on a virtual road that represents the music with its twists and turns; and makes decisions on how to traverse each part of the road. The driver's decisions affect in real-time the rendering of the piece. The pedals and wheel provide a tactile interface for controlling the car dynamics and musical expression, while the display portrays a first person view of the road and dashboard from the driver's seat. This game-like interface allows non-experts to create expressive renderings of existing music without having to master an instrument, and allows expert musicians to experiment with expressive choice without having to first master the notes of the piece. The prototype system has been tested and refined in numerous demonstrations. This paper presents the concepts underlying the ESP system and the architectural design and implementation of a prototype.

## Keywords

Music expression synthesis system, driving interface.

## 1. INTRODUCTION

The goal of the Expression Synthesis Project (ESP) is to create a driving (pedals, wheel and display) interface for generating expressive performances interactively in real time from expressionless music files. The premise of ESP is that driving can serve as an effective metaphor for expressive music performance.

Almost anyone can drive a car, but not everyone can play an instrument. By using the familiar driving interface, we not only make ESP immediately accessible to a large population of users, but also minimize the cognitive overhead in learning to use a new control device. Expressive performance is the goal, but often also the final frontier, in music education. The degree of control required to generate expressive performance typically requires many years of practice. The proposed interface will allow non-experts access to high-level expressive decisions without having to master an instrument; it will also allow expert musicians (including composers) to experiment with expressive choice without having to first master the notes in a piece of music or the instrument. In the ESP system, the driving wheel, pedals and display offer an attractive interface that is easy to use and understand. Creating

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Nime'05*, May 26-28, 2005, Vancouver, BC, Canada.

Copyright remains with the author(s).

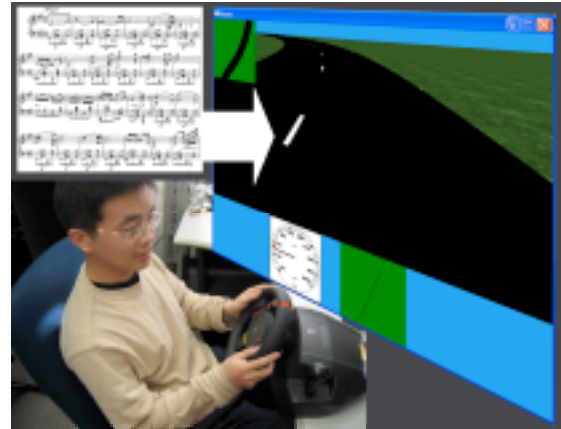


Figure 1. ESP driving interface for expression control.

a performance becomes a journey through a terrain in a driving game that even children can play and appreciate.

In ESP, the score is mapped to a road in a virtual world and the user controls the way in which this road is traversed, which corresponds directly to the tempo and loudness of the music as it unfolds over time. Figure 1 shows our present prototype of the ESP system. The turns in the road correspond to musical structure and guide the user in his or her expressive choice.

### 1.1 Motivation for the Driving Metaphor

Compelling metaphors are the keys to intimate control of music through computers and other interfaces [17]. Anecdotal evidence suggests that high-level musical performance is, in many ways, very much like driving a car. A musical score can be conceived of as the map to a road. For the musician, each performance of the piece is an experience of a particular traversal of this road. This experience is shaped by the performer's choices during that particular journey. These choices include how long to dwell on certain details, which ones to gloss over and which ones to emphasize. Continuing with our analogy, the subject of these choices correspond to the speed and energy with which one passes through each segment of the road, each turn and every bump. The use of a car interface to control musical expression by navigating through a 3D virtual environment representing the content of a piece is hence a reasonable choice.

In ESP, the user experiences and manipulates the music through an interactive, game-like interface in a 3-dimensional virtual environment. The driving interface has been adopted in other real-time interactive music systems, such as *Amplitude* by Harmonixmusic [9] and the *Harmonic Driving* [14] in Tod Marchover's Brain Opera; these applications focus on generating music rather than expressive renderings of a pre-existing piece. In ESP, the car on the road marks the present; and acts as a metaphor for the reading cursor of a music player that drives the aural rendition of the piece. As the car

progresses along the road, the dynamics and articulation of the music rendered are linked to its acceleration and tempi respectively, in real-time. The visual display gives a first person view of the road, a speedometer that shows the tempo in beats per minute, and a map of the bird's eye view of the road.

By using a driving interface, we capitalize on the natural analogies between automotive control and smooth musical timing. Cambouropoulos et al [3] showed that humans prefer expressive performances that maintain smooth tempo transitions. As anyone who has driven (or ridden in) a car can attest, humans also prefer smooth transitions in the handling of an automotive. In the ESP system, the car is subject to the laws of physics in the real world.

The defining feature of the ESP virtual environment is a road that represents the music, with curves that correspond to its twists and turns. Musical expression is most effective when it serves to highlight and reinforce musical structures. In ESP, this link between performance decision and musical structure is established through the embedding of musical and physical cues in the terrain to be traversed.

## 2. RELATED WORK

The history of creating expressive performances from existing music using machinery can be traced back to the invention of the player piano in the late 1800's [5]. Many systems have been created since to provide more natural (and less laborious) intermediaries for the control of tempo and loudness in the rendering of composed music.

More recently, conducting systems are a popular choice for the control of tempo and loudness. One of the earliest conducting systems is Max Mathews' *Radio Baton* [1]. The player moves the batons above a planar receiver, which tracks the 3D coordinates of the baton and then determines the conducting gestures. *You're the Conductor* [11] uses an infrared baton whose signal is detected by a chip inside a camera. Two computers process the signal to calculate the desired tempo and loudness, and then control the time stretching and the playing of audio and video. Another recent system, described in [12], uses computer vision techniques to track the position of the baton, and the tempo of the music output is adjusted in real time. Marrin's [15] 2000 digital baton experiments showed that a gestural interface was problematic for accurate extraction of beats and discomfort with the interface design can lead to unnatural gestures.

In contrast to the gesture-controlled conducting systems, Haruhiro and Keita [13] showed a tapping-control system that allows the player to use beat tapping, recorded by a sensor, to conduct the performance. The use of both gestural and tapping input for tempo control puts the onus of maintaining flow and smoothness, important criteria for expressive performance, on the user. In ESP, we propose to mitigate this difficulty for non-experts by using an interface that is naturally biased towards smooth tempo transitions.

ESP and the systems described above focus on real-time human control of music to create expressive performances. Other systems exist that generate expressive performances without real-time human intervention, by learning from and emulating existing performances [2], or by rule-based and piecewise manipulation of local expressive parameters [16].

In the remainder of this paper, we present our current ESP prototype, its architectural design, and its mapping of input information to music parameters. Section 3 describes the ESP

system; Section 3.1 the car control and dynamics; Section 3.2 the aural and visual rendering. Finally, we present some future plans for the ESP system in the concluding Section 4.

## 3. THE ESP SYSTEM PROTOTYPE

In this section, we describe our solutions to the two main challenges of creating the ESP system: its design and implementation using an architecture model that supports concurrent and parallel processing of generic data streams, and the effective mapping of the input data to car dynamics, and to music rendering parameters that sound good.

### 3.1 Architectural Design

ESP is a complex real-time interactive virtual environment. As such, a challenging aspect of the ESP system is the simultaneous processing of multiple data streams in real-time, such as graphics, MIDI, music computation, car dynamics, and driving controls. Perhaps even more challenging is the organization of, and interaction between the software components carrying out these tasks. We designed the system using the Software Architecture for Immersipresence (SAI) framework [6]. The resulting architecture is modular and extensible, which benefits future development and research.

The ESP design is implemented in C++ using the Modular Flow Scheduling Middleware (MFSM) [8], an open source architectural middleware that provides code support for SAI's architectural abstractions. MFSM also offers a number of useful open source functional modules. Using some of these modules significantly reduced the coding effort and allowed us to focus on ESP-specific aspects during the development of the ESP prototype.

Figure 2 shows a conceptual level system diagram. The ESP graph is expressed in the SAI notation. SAI distinguishes between persistent and volatile data. Persistent data is held in repositories called *sources* (red disks), while volatile data flows down *streams* (thin blue arrows) in the form of *pulses*. Pulses travel down streams through processing elements called *cells* (green squares). The ordering of the cells along a stream captures their dependency relations. Each cell is connected to exactly one source, while many cells may be connected to a same source. All the cells connected to a given source have concurrent shared access to the data stored by the source. When a pulse reaches a cell, it is used as input data to a process that may result in the generation of new volatile data that is added to the pulse. The process may also result in the update of some of persistent data held by the cell's source, also used as input to the cell's process. When the process is completed, the pulse continues its journey downstream. The dynamic evolution of a stream may be thought of as a cascade of processes triggered by pulses. Cells are specialized to implement required functionalities. The *Pulsar* is a special cell type that generates empty pulses at specific time intervals.

The ESP system design employs SAI architectural patterns that are typical of interactive systems [7]. A central repository holds persistent data structures that are accessed independently by cells placed on independent streams. These data structures, are maintained during the whole life of the application, and include a 3D model of the virtual environment, a dynamic model of the car, and a MIDI representation of the input music piece (list of MIDI events).

Each stream embodies a specific aspect of the system, namely dynamic evolution of the car model, car control through the user input device, aural rendering, and visual rendering. A

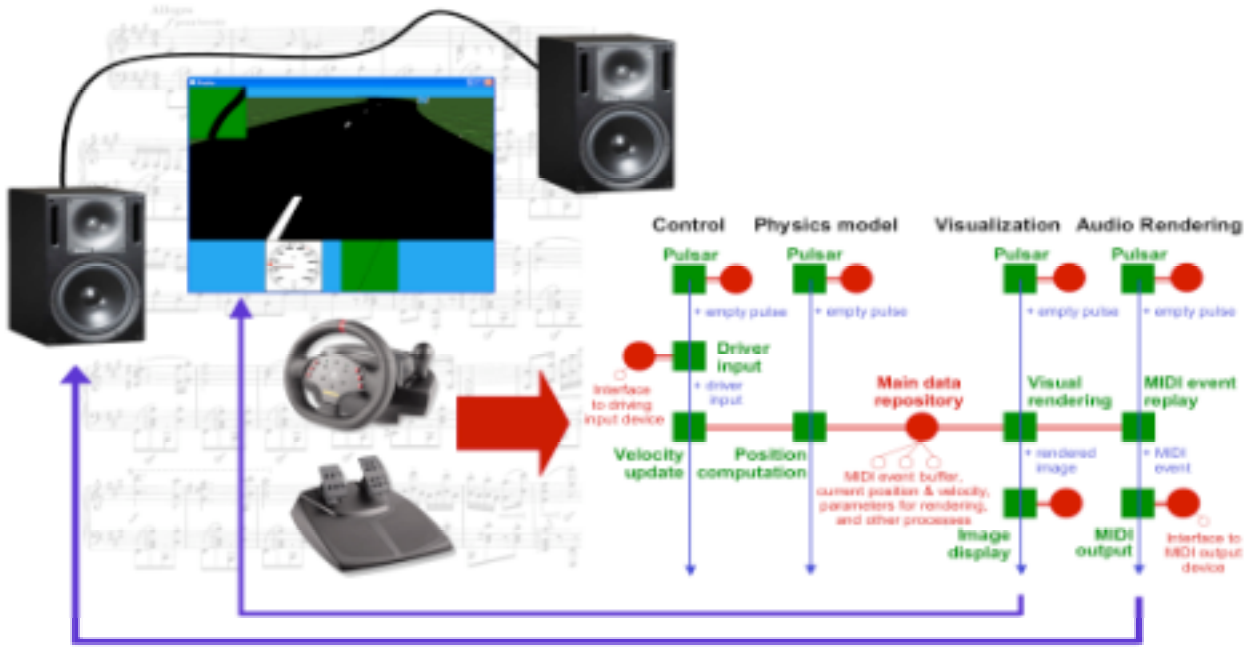


Figure 2. ESP system diagram.

separate Pulsar regulates each stream so that its behavior is independent of the others. In particular, each stream can operate at a separate frequency adapted to the specific requirements of its function. The specifics of the function of each module are described in the following sections.

### 3.2 Mapping Car Control Input to Tempo

The current prototype uses a Logitech MOMO Racing Force Steering Wheel with realistic gas and brake pedals. This section details the mapping of raw input parameter values to the meaningful modifications of the car dynamics model and associated musical tempo. We use a simple car dynamics model [10] to simulate the physics of actual car motion.

The user input via the game device follows an established SAI pattern [6] that had been implemented and was readily available for reuse in the ESP prototype. As shown in the ESP system diagram (Figure 2), the Driver input cell polls the device at regular intervals (set by a Pulsar) and generates a volatile data structure containing all the parameters describing the current state of the device. The Velocity update cell uses these values to affect the dynamic model of the car. This model is used concurrently by the Position computation cell, which applies dynamics equations to compute the car position at regular time intervals from its current position and the current values of its dynamic parameters.

The driving interface has two pedals used for the control of the music. Depressing the right pedal increases the tempo of the music; stepping on the left pedal slows down the music. The forward force is assigned to be a linear function of the pedal value. Given a forward force  $F$ , the equation for acceleration as a function of force can be represented as follows:

$$a = \alpha_1 F - \alpha_2 v^2 + K_1(\alpha, \dot{\alpha}),$$

where  $\alpha_1$  and  $\alpha_2$  are constants, and  $K_1$  is a function of the angle of the wheel,  $\alpha$ , and of the angular velocity,  $\dot{\alpha}$ . Simulated values show that, for fixed  $(\alpha, \dot{\alpha})$ , the curve of acceleration as a function of force typically takes a sigmoid shape.

Using the steering wheel, a user can navigate the turns and thus control the car's direction and the view of the road. We discovered through experimentation that a linear map between the wheel angle and car angle changes leads to an uncontrollable driving situation. To create a more controllable wheel interface, we chose a non-linear (sigmoid) mapping between the wheel's angle to the turning of the car.

As in a real car, the angle of the turn affects the forward acceleration. When one considers a fixed forward force and angular velocity, the equation for acceleration as a function of the angle of the turn  $\alpha$  becomes the following

$$a = -\alpha_1 \alpha - \alpha_2 v^2 + K_2(F, \dot{\alpha}),$$

where  $\alpha_1$  and  $\alpha_2$  are constants, and  $K_2$  is a function of the forward force,  $F$ , and of the angular velocity,  $\dot{\alpha}$ .

The car velocity directly sets the tempo of the music. The car's velocity can be readily obtained from the acceleration. This velocity is directly proportional to the music's tempo

$$T_t = T_{t-1} + a dt.$$

The initial tempo is the default given by the score-generated MIDI file. Similarly, we can readily obtain the score position as a function of the tempo

$$D_t = D_{t-1} + T_t dt.$$

An independent stream (labeled *Audio Rendering* in Figure 2) handles the generation of output "expressive" MIDI events according to the car's position along the road. The score-generated MIDI information is stored as an ordered list in the Main data repository. A Pulsar generates pulses at regular intervals. Upon receiving such a pulse, the MIDI event replay cell determines whether an output event must be generated (from the original list) according to the current car position on the road. The output MIDI events can then be interpreted and rendered by a MIDI synthesizer. We reused some existing MIDI manipulation code and developed some other code specifically for this project. Note that the pulsing frequency of the Pulsar sets the temporal resolution with which output events are generated, which is directly related to the accuracy

resolution of the system. A high pulsing frequency results in greater accuracy, but increases the computation load on the system. We found that values between 30 Hz and 100 Hz were appropriate for our tests.

### 3.3 Mapping Acceleration to Loudness

We explain the mapping of acceleration to loudness in this section. Informal observation of Dixon et al's [4] visualization of expert performances in tempo-loudness space reveal that the rate of tempo change appears to be directly correlated with changes in loudness. Hence, we base the loudness of the music on the car's acceleration. In the current ESP prototype, there exists a linear map between acceleration and dynamics, with the loudness (as given by the velocity of each onset) increasing when the user accelerates the car, and decreasing when the car decelerates. If  $V_0$  is the original onset velocity, then the new onset velocity,  $V_1$ , is given by

$$V_1 = V_0 + a \square,$$

where  $\square$  is a scaling factor which makes the variation in loudness reasonable. The velocity of each output MIDI event is calculated from the original event velocity and the current car acceleration according to this formula.

### 3.4 Mapping Car Position to Visual Display

The user experiences the virtual environment both aurally and visually from a first person perspective while controlling the car. As the car progresses along the road, changes to the tempo and dynamics of the music rendered are linked to the driver's actions in real-time. Visual experience of the virtual environment is achieved by synthesizing a pictorial representation of the current state of the environment.

An independent stream (labeled *Visualization* in Figure 2) handles the production of such images. A Pulsar triggers empty pulses at regular intervals. Upon receiving a pulse, the Visual rendering cell creates a synthetic composite image that is placed on the stream, and that the Image display cell will then present the image in a window on the screen. This is another established SAI pattern, implemented in an open source "white-box" module [8] that uses the OpenGL library for 3D rendering. The open source module was specialized to the specifics of this project. The pulsing frequency on the stream sets the refresh rate of the display on the screen. We typically run our rendering stream at frequencies of at least 15 Hz so as to provide a smooth visual experience.

## 4. CONCLUSIONS

ESP provides a novel and compelling driving metaphor for expressive control that allows a user access to high-level interpretative decisions in musical performance. We have presented the concepts underlying the ESP system and described the design and implementation of its prototype. Initial tests and demonstrations using Brahms' *Hungarian Rhapsody No. 5* have been enthusiastically received by technologists and other visitors from industry and academia, and by engineering and music university students.

By recording the output of the system, we plan to study the relations between expressive decisions and musical structure. Just as the road can be designed to bias the player towards expressiveness where expressiveness is due, one user's expressive rendering of a piece can be used to create the road to guide another player, or to create an automatically generated performance via an automatic drive option. Future work will be

directed at the developing of automatic methods for analyzing where expressions should be created.

We also plan to study other mapping strategies between the car controls and dynamics, and tempo and loudness, so as to best connect the user's intent to the resulting expressive rendering of the piece. Future versions of the ESP prototype will explore other mappings and modifications to the interface.

## 5. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 0347988. The work made use of the Integrated Media Systems Center Shared Facilities supported by NSF under Cooperative Agreement No. EEC-9529152. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

## 6. REFERENCES

- [1] Boulanger, R., and Mathews, M. The 1997 Mathews Radio-Baton & Improvisation Modes. In *Proc. of the 1997 Intl Comp Mus Conference (ICMC' 97)* (Thessaloniki, Greece).
- [2] Bresin, R., Artificial Neural Networks Based Models for Automatic Performance of Musical Scores. *J. of New Mus Res* 27:3 (1998), 239-270.
- [3] Cambouropoulos, E., Dixon, S., Goebel, W. and Widmer, G. Human Preferences for Tempo Smoothness. In *Proc of the VIII Brazilian Symposium on Comp Mus* (Fortaleza, Brazil, 2001).
- [4] Dixon, S., Goebel, W. and Widmer, G. The PERFORMANCE WORM: Real-Time Visualization of Expression Based on Langner's Tempo-Loudness Animation. In *Proc of the Intl Comp Mus Conf (ICMC 2002)* (Göteborg, Sweden, Sep 16-21, 2002).
- [5] Edwards, B., Player Piano History and Evolution. [www.perffessorbill.com/ragtime5.shtml](http://www.perffessorbill.com/ragtime5.shtml)
- [6] François, A. R. J. A Hybrid Architectural Style for Distributed Parallel Processing of Generic Data Streams. In *Proc of the Intl Conf on Software Eng (ICSE '04)* (Edinburgh, Scotland, May 2004). pp. 367-376.
- [7] François, A. R. J. Components for Immersion. In *Proceedings of the IEEE Intl Conf on Multimedia & Expo (ICME '02)* (Lausanne, Switzerland, August 2002).
- [8] François A. R. J. Modular Flow Scheduling Middleware (MFSM), 2001. [mfsm.sourceforge.net](http://mfsm.sourceforge.net)
- [9] Harmonixmusic. [www.harmonixmusic.com](http://www.harmonixmusic.com)
- [10] Hatwal, H. and Mikulcik, E. C. Some Inverse Solutions to an Automobile Path-Tracking Problems with Input Control of Steering and Brakes, *Vehicle System Dynamics* 15 (1986), 61-71.
- [11] Lee, E., Nakra, M.T., and Borchers, J. You're The Conductor: A Realistic Interactive Conducting System for Children. In *Proc of the 2004 Conf on New Instruments for Musical Expression (NIME'2004)* (Hamamatsu, Japan, June 3-5, 2004).
- [12] Murphy, D., Andersen, T.H., Jensen, K. Conducting Audio Files via Computer Vision. In *Proc of Gesture Workshop* (Genova, Italy, April 15-17 2003).
- [13] Katayose, H., and Keita, O. Using an Expressive Performance Template in a Music Conducting Interface. In *Proc of the 2004 Conf on New Instruments for Musical Expression (NIME'2004)* (Hamamatsu, Japan, June 3-5, 2004).
- [14] Machover, Tod, Harmonic Driving. [brainop.media.mit.edu/text-site/onsite/harmony.html](http://brainop.media.mit.edu/text-site/onsite/harmony.html)
- [15] Marrin, T. N. *Inside the Conductors Jacket: Analysis, Interpretation and Musical Synthesis of Expressive Gesture*. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, 2000.
- [16] Sundberg, J., Friberg, A., and Bresin, R. Attempts to Reproduce a Pianist's Expressive Timing with Director Musices Performance Rules. *J. of New Mus Res* 32, 3 (2003), 317-325.
- [17] Wessel, D. and Wright, M. Problems and Prospects for Intimate Musical Control of Computers. *Comp Mus J.* 26, 3 (2001), 11-22.