COMPOSING WITH HYPERSCORE: AN INTUITIVE INTERFACE FOR VISUALIZING MUSICAL STRUCTURE

Morwaread Farbood
New York University
Steinhardt School

Henry Kaufman
Harmony Line, Inc.
Cambridge,
Massachusetts

Kevin Jennings
Marino Institute of
Education
Dublin, Ireland

ABSTRACT

Hyperscore is a graphical, computer-assisted composition system that allows users to intuitively visualize and edit musical structures. It maps musical features to graphical elements such as color, shape, and line texture. These graphical elements allow users to control both high and low-level musical features such as pitch, dynamics, melodic contour, and harmonic tension. Hyperscore facilitates composition by providing the user with a visual representation of the large-scale structure of a piece while simplifying the process of integrating diverse rhythmic and melodic material. It is designed primarily for users with limited or no musical training.

1. INTRODUCTION

Hyperscore is a computer-assisted composition system that enables users to compose music graphically. The system breaks down musical structure into two levels: 1) the creation of rhythmic and melodic phrases and 2) the shaping of the large-scale progression of the piece. Graphical user input is in the form of freehand drawing; the lines drawn are interpreted according to shape, color, and position and converted into pitches and rhythmic values. Hyperscore provides a layer of abstraction between these two levels: the melodic, note-level input and large-scale, form-level shaping [5].

It tries to push the concept of "what you see is what you hear" in a score as far as possible. While some musical elements are inputted in piano-roll notation, the score itself is drawn freehand by the user. Thus the interface attempts find a balance between precise control and having a free-flowing and expressive visual interface. Making freehand line drawings into a useful music control system enables users with little no musical experience to explore musical creativity at many levels. The system provides high-level control over the dramatic arc of the piece as a whole as well as the placement of individual melodic and rhythmic elements.

2. PRIOR WORK

In the late 1960's, the advent of the first computer terminals opened up the possibility for graphical interfaces. Max Mathews and L. Rosler began using this new technology to explore graphical applications for music. Utilizing a new computer called the GraphicI, they developed a compositional language that substituted graphical input in the place of tediously punched data cards. The system allowed a musical score to be specified as a group of graphs and provided the means to algorithmically manipulate them by drawing curves on the computer screen [11].

A broad range of graphical computer-assisted composition tools have followed Mathews and Rosler's work. Some systems are suited for professional musicians; these tend to use graphical objects to represent musical functions or tweak parameters. The former category includes PatchWork/OpenMusic [1], a visual, object-oriented programming environment designed by researchers at the Institut de Recherche et Coordination Acoustique/Musique (IRCAM) to encapsulate musical functions in graphical objects that can be dragged, dropped, and interconnected to implement musical algorithms. The latter category includes standard commercial applications such as Digital Performer, Cubase, and Logic. They are in essence multitrack sequencers that use graphical input to manipulate parameters such as volume, timbre, and attack and decay envelops.

Systems such as David Zicarelli's OvalTune, a program in which users create sounds and visual images simultaneously by painting with a mouse, and Cyber-Band [14], developed at IBM, are designed for users who don't necessarily have musical experience. CyberBand is similar to Hyperscore in some ways: it uses riffs, or thematic fragments, as musical building blocks and higher-level modifiers to edit and refine the music. Its interface, however, is fundamentally different from Hyperscore's. It does not use drawing as a method of combining musical material and lacks the visual freedom of Hyperscore's environment.

Iannis Xenakis' UPIC [9] is a system that uses a large, high-resolution graphics tablet for input. Its macrocompositional level lets the user draw a time-frequency score consisting of lines, curves, and points. Even though Xenakis himself used UPIC to compose music, he envisioned musically inexperienced users like children using the system as well. Other applications accessible to musically untrained users include Maxis/Iwai's SimTunes [8], the

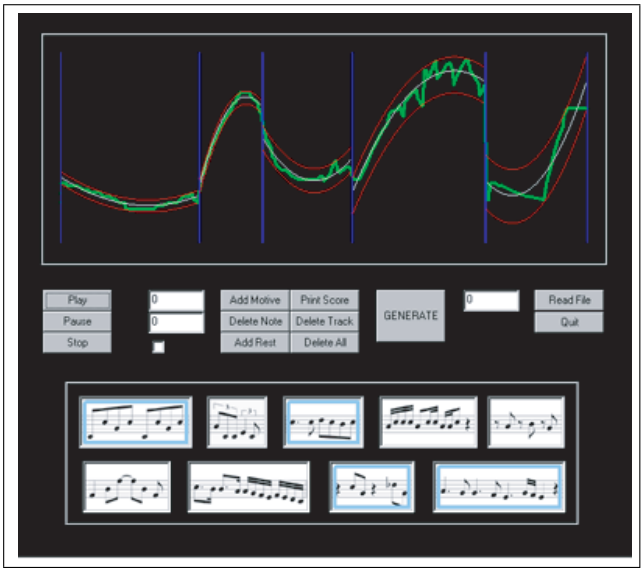

Figure 1. The initial Hyperscore prototype.

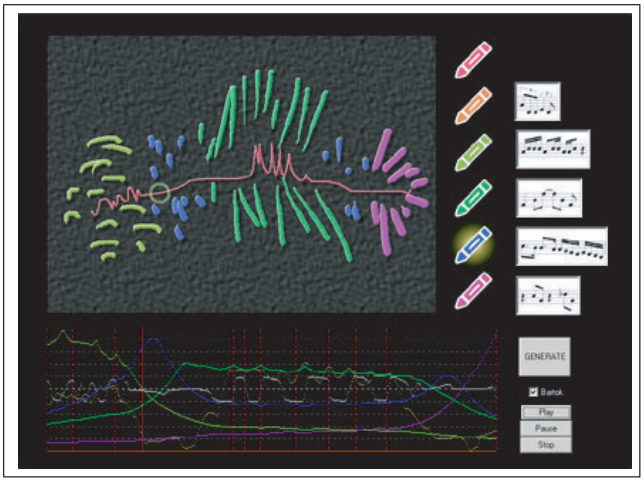

Figure 2. An intermediate version of Hyperscore.

Macintosh program MetaSynth [13], and Morton Subotnik's Music Sketch Pads [12].

Hyperscore is unique from all of these applications for several reasons. While other programs like UPIC might have similar freedom in graphical input, Hyperscore goes a step further from a musical perspective by mapping graphical elements not just to one-dimensional features such as frequency or pitch, but higher-level structures like functional harmony. It does not require users to be proficient either musically or technically—they do not have to play an instrument, or read music, or know how to program. Yet it also allows them to compose music in many different styles without taking away their sense of ownership in the creative process.

3. THE EVOLUTION OF AN INTERFACE

The goals of Hyperscore have evolved considerably since the first prototype was completed in 2000. The earliest version was almost entirely automated [4] [7]. The user drew a "musical tension line" and the program then parsed line and generated a piece of music according to the shape and texture of the line (see Figure 1). The line was divided into parabolic segments and the smoothness of the segments analyzed. If a segment was smooth, the corresponding rhythmic texture in the generated music would be less dense (greater rhythmic activity was equated with more "tension"). The user selected all or a subset of nine pre-composed melodic patterns which were used to generate the piece. The system did not choose which melodic patterns to use deterministically; each time a piece was generated for a specific graph, the results were different. The graph could only influence the texture density produced by the combination of contrapuntal lines.

Over the course of the next few years, the interface went through a series of iterations and the focus shifted to allow more precise user control and less automation. In intermediate versions, users were given the ability to graphically annotate the tension line to indicate at what point in time melodic material would be used by the system (Figure 2). The graphical interface was redesigned [4] and the concept of colored "pens" for drawing lines introduced. Users could indicate where and what kinds of melodic material were used by selecting and drawing with a color that was mapped to the melodic pattern. The line's proximity to the tension curve influenced what motivic material was selected by the generation algorithm. Eventually, as the music generation became more user-driven and deterministic, the tension line ceased to function as such and simply became a time and volume-modulation line.

Another new feature was the automated harmony generator. This algorithm was implemented using hierarchical Markov chains to handle different layers of organization. One set of Markov chains was used to generate a series of higher-level harmonic functions, and another set was used to generate the actual chords. The chord functions were simple, consisting only of three categories: tonic, dominant, subdominant. Chord function transition probabilities were selected based on the time at which the chord occurred and the function of the chord preceding it. The chords themselves were chosen according to time and relative frequency at which the chord would appear regardless of the circumstances (i.e. not dependent at all on the preceding chord). Eventually this feature was abandoned for an improved harmony control that allowed for more user input and more sophisticated chord progressions (see Section 4.2).

4. THE CURRENT SYSTEM

Hyperscore is written in C++ using DirectX and the Win32 API. The current interface consists of an expansive, zoomable canvas where users can create any number of musical fragments and whole pieces. Users can position these musical objects anywhere on the canvas and can view the workspace at any level of zoom for ease of editing.

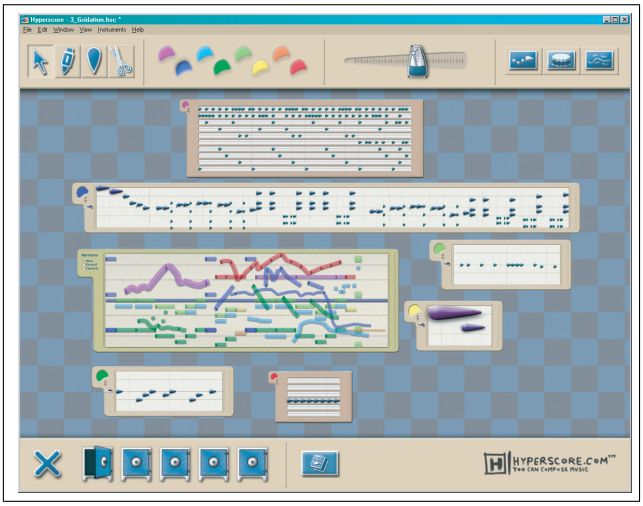

Figure 3. The current version of Hyperscore. All three types of motive windows are present (melodic, polyphonic, and percussion) as well as a sketch window.

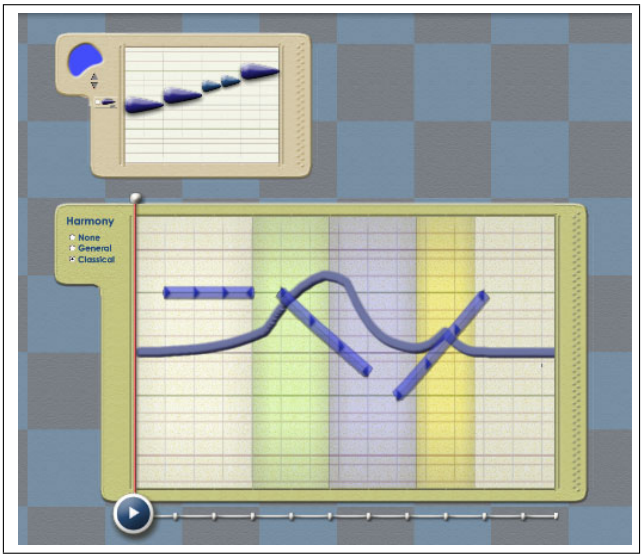

Figure 4. A motive window and a sketch window showing lines drawn in the color (blue) associated with the motive.

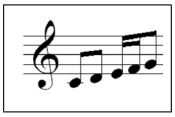

Figure 5. Musical realization of the motive window shown in Fig. 4.

4.1. Interface and interaction

4.1.1. Motives

The basic musical building blocks in Hyperscore are called *motives*. Motives are relatively short musical phrases that can be built up together in a composition. Motives are created in special windows that allow the graphical placement of notes in a piano-roll style notation where time is the X-axis and pitch is the Y-axis (see Figures 3 and 4). Notes can be placed by clicking anywhere in the motive window, and then stretched graphically to change the note duration (using standard "handles" familiar to any GUI for shape manipulation). A user-configurable grid ensures that the note onset and duration remains constrained. Notes can be "multi-selected" and the whole group can be scaled to speed up or slow down the notes. In that case, only the beginning and ending of the whole phrase are constrained to the grid, allowing a more flexible way of editing musical phrases. Individual notes can be snapped to the grid later if desired.

To enforce a melodic conception of a motive, the motive window can dynamically constrain only one note to exist at any given time. As a note is dragged or stretched, other notes that are overlapping with that note are made translucent. Once the editing operation is complete, any translucent notes are deleted. This ensures that the motive is a simple melody with no chords. This feature can be turned off to make more complex, polyphonic motives.

Hyperscore also allows the creation of percussion motives using the general MIDI percussion kit. In the percussion window, up to ten monotonic tracks can be created, each track referring to a particular MIDI percussion instrument note. The collection of tracks forms a kind of sound palette with which to compose rhythms.

4.1.2. Sketching a score

To create compositions with the motivic building blocks, users draw in a sketch window that is the core of the Hyperscore interface. Each melodic and percussion motive window has a color assigned to it that the user can select. When a line of a particular color is drawn in the sketch window, the corresponding motive is sequenced into the composition.

The start and end points of the line determine how many times a motive is repeated. A fixed pixel-to-duration metric calculates the length of time a line plays. If the length of a line does not divide evenly into whole repetitions of a motive, then a fragment of the motive is used for the last iteration. Drawing a straight line makes the motive repeat with the precise melodic intervals of the original material. The vertical position determines how much the motive is transposed up or down. Curves and bends in the line impose a pitch envelope on the motive's repetitions but does not alter the melodic contour to the point that the new material is unrecognizable from the original motive (Figures 4, 5, 6, and 7). The slope of the line does not serve as a literal envelope that directly alters the pitch content; it intelligently transforms the material without altering the directionality of internal intervallic relationships present in the original motive.

Once a line is completed (in a single click and drag gesture), the beginning of the line (defined as the leftmost endpoint) is snapped to the nearest temporal and pitch grid point. Then the end of the line is snapped to the nearest temporal grid point either by extension or clip-

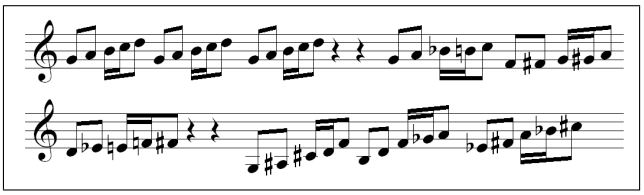

Figure 6. Unharmonized musical realization of the sketch window shown in Figure 4.

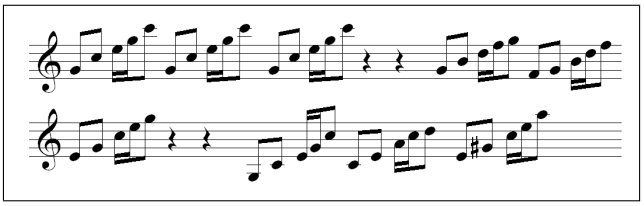

Figure 7. Harmonized musical realization of the sketch window shown in Figure 4. The chord progression generated by the harmony line starts in C major and modulates to A minor.

ping. If a completely free-form line is drawn with overlaps and loops, the line is broken up internally into temporally monotonic sub-segments to facilitate further processing. Though generally this type of line is not very meaningful musically, users expect to "hear a lot of music" if they draw a big and complicated line.

Line editing features include trimming, cutting and pasting, adjusting playback volume, selecting a harmonization mode, and selecting an instrument. The instrument choices include all General MIDI sounds. Hyperscore objects can be saved as MIDI files and in turn can be read into a notation program such as Finale or Sibelius. This makes it possible to go from Hyperscore format to musician-readable format, giving a composer the option of sketching out a composition in Hyperscore and then editing in staff notation.

4.2. Automated harmony

The most significant enabling aspect of Hyperscore's interface is its ability to facilitate the user's exploration of higher-level aspects of composition such as large-scale form and harmony. As discussed before, the original prototype of Hyperscore was based on the idea that an algorithmic composition system could generate music based on a central graph describing changes in musical "tension." Although this idea was phased out to allow for less automation and more user control, the idea of having a tension line merged with the early automated harmony generator into a new idea: a global *harmonic* tension line.

One reason for having a graphical notation system in the form of freehand drawing is to provide the user with an expressive means of shaping musical direction. Drawing a contour is a simple and intuitive way to depict areas of harmonic tension and resolution.

The algorithm for this new "harmony line" was based tangentially on David Cope's Experiments in Musical Intelligence (EMI). EMI takes existing works in a given style, segments them into musical fragments and then reconstitutes them in an intelligent way to form new pieces in the same style. EMI creates a database of musical fragments by analyzing and segmenting examples of a particular category of music (e.g. Chopin Mazurkas). The analysis process uses a classification system of functional identifiers called SPEAC (for Statement, Preparation, Antecedent, and Consequent). Pattern matching is used to determine what recurring signatures should not be segmented; it is important that certain signatures remain intact because they are necessary for the stylistic identity of the music. The segments are then placed in a lexicon according their SPEAC meaning. New music is then generated from the segments by using an augmented transition network. [2]

Cope's idea of classifying functional identifiers influenced the algorithm for interpreting Hyperscore's harmony line. In Hyperscore, users describe harmonic progressions by shaping the harmony line. It is parsed into sections which are then mapped to functional identifiers that resemble SPEAC [7]. Hyperscore's identifiers have been modified from Cope's, and consist of four categories: Statement, Antecedent, Consequent, and Modulation. The harmony line, which runs through the center of each sketch window, can be modified by clicking and dragging. Colored bands appear to indicate the line's parsing (Figure 8). Sections are classified as one of four visual types, each corresponding to a functional identifier:

- **Statement** - flat section, colored white. Musically defined as a statement or prolongation of the tonic.
- **Antecedent** - upward-sloping section, colored green. Musically defined as a combination of chords that need resolution (e.g. dominant chords or combinations of subdominant and dominant chords).
- **Consequent** - downward-sloping section, colored blue. Resolution of preceding Antecedent section. If not preceded by an Antecedent, then restates the tonic.
- **Modulation** - defined by a sharp pointed region or spike, colored yellow. Progression toward a new key.

After the line is parsed, chords are assigned to each section based on its functional identifier, how many beats it spans, and how textured or "bumpy" the section is. The instability of the chords assigned to a section is directly proportional to the amount of texture. The chords chosen are taken from a database that returns either single chords or small progressions based on the selection criteria. The database consists of chord progressions commonly found in Bach chorales.

When the chord progression has been determined for the entire piece, the notes generated from the sketch are altered so they match either the currently assigned chord or a scale tone in the current key (see Figure 7 for a simple example). For minor keys, there are special provisions

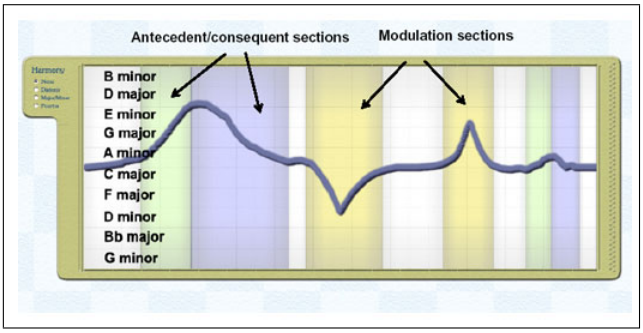

Figure 8. An empty Hyperscore sketch window showing the harmony line. The height or depth of the point indicates the key to which the section modulates (indicated by the text overlay).

for inserting a raised $\hat{6}$ or $\hat{7}$ depending on the chord and context. There are several criteria used in deciding how and in what direction a pitch is altered:

- **Beat** - If a pitch falls on a beat or is longer than a sixteenth note in duration, it is harmonized as a chord tone. If it is short in duration and does not fall on a beat, it is harmonized as a scale tone.
- **Contour** - Notes are moved up or down as minimally as possible while attempting to preserve the contour of the original melodic material. Even if the original pitch is a valid chord tone before being harmonized, it might still be altered if it distorts the overall melodic contour.
- **Voice** - The voice determined to be the bass line does not have the strict melodic contour requirements and could be altered radically to fit not just the nearest chord tone, but the bass note of the chord (root or inversion). This does not apply in the case when there is only a single active line (a solo voice).

Users can choose from different harmony styles including diatonic, major-minor, fourths, and none. "None" indicates that no automatic harmonization is applied. Diatonic mode changes all chromatic pitches into diatonic ones in the current key (defined by the presence of modulation sections in the harmony line). Major-minor is eighteenth-century-style tonal harmony. Fourths mode is based on chords constructed from fourths rather than thirds. Although fourths mode uses the same database as major-minor mode, some of the chord root notes have been altered to fit the functional identifiers more closely. For example, the fourths-mode equivalent to a dominant seventh chord is a chord built on the leading tone, giving it a stronger pull toward the tonic.

Aside from a complete harmonization done with regard to a harmonic progression generated from the harmony line, there is the additional option of selecting any subset of the lines drawn in the sketch window to be unharmonized within the current tonal context. This selection is indicated visually by giving the line color a darker tint. The effect of unharmonizing individual lines does not revert the line to its original chromatic form—it alters all necessary pitches to correspond to scale tones in the current key rather than chord tones.

5. APPLICATIONS

5.1. Educational projects

One of the early applications of Hyperscore was its use as the primary vehicle for composition activities in Tod Machover's Toy Symphony [10], a large MIT Media Laboratory project bringing together children and professional orchestras through the use of technology. The goal of Toy Symphony was to introduce children to creative music-making with specially designed hardware and software. These tools allowed children to perform on stage with musicians as well as compose music that was performed by orchestras. By this point in time, Hyperscore had developed beyond an experimental interface and was sophisticated enough to enable novice users to compose original music of high quality. During the course of the Toy Symphony project (2002-2005), children aged 8 to 15 from all over the world worked with the software to compose pieces for string orchestra [6], some of which were performed in concert by professional orchestras such as the BBC Scottish Symphony and the Deutsches Symphonie-Orchester Berlin.

5.2. User feedback

Hyperscore was initially developed at the MIT Media laboratory from 2000-2004. Following the Toy Symphony project, development continued at Harmony Line, Inc., a company founded to commercialize Hyperscore and reach a wider audience. In order to make the application more accessible to the average user, the interface was overhauled, graphics board compatibility expanded (the GUI uses the 3D acceleration capabilities of the graphics board), and many features added such as "undo/redo" for all editing operations, extensible XML file format, internationalization of GUI components, and many usability improvements over all previous versions.

From January through August 2006, the Hyperscore executable was realeased to the public for free. Included were networking features such as "upload to community website" and "send your music to cell phone as a ringtone." A nominal fee was charged for sending to a phone, but everything else was free. When a composition was uploaded, its Hyperscore source file would be sent to the server along with a MIDI rendering of that song. The server would then take the MIDI file and convert it to an mp3 file that could be played in an online Flash-based music player.

The Hyperscore community parallels other online communities that have formed around commercial programs like Acid and Reason. It now has over 12,000 members, and about 200 of them are passionate users that have been in the community for more than a few months and log in and post comments or music regularly. On the site, people can post their compositions and rate and comment on

other people's music, as well as send anyone's composition to their phone. People can also optionally allow anyone to download their Hyperscore source files.

The community population is skewed toward teenagers, though users in their twenties and thirties are not uncommon. A significant number of users do not know how to read music, but some of the most sophisticated composers have professional music experience of various capacities, and enjoy Hyperscore for its novelty and ease of use.

There have been approximately 6000 songs uploaded since Hyperscore was made available to the public. Some of the compositions are classically oriented (based on self-descriptions), but most are in the Pop/Rock/Hip Hop styles. Many users use the harmony line feature to make their music "sound better," while the more sophisticated users tend to want full tonal control and use the sketch window for sequencing and pitch shifting.

Some very interesting online behavior has been observed that is specific to a community organized around a novel composing tool. More experienced users often serve as guides to the less experienced novices. There are have been some spontaneous behaviors like remix competitions where people share a few motives and then mix them into different compositions in a sketch window. Allowing the sharing of Hyperscore files on the site greatly facilitates this kind of collaboration.

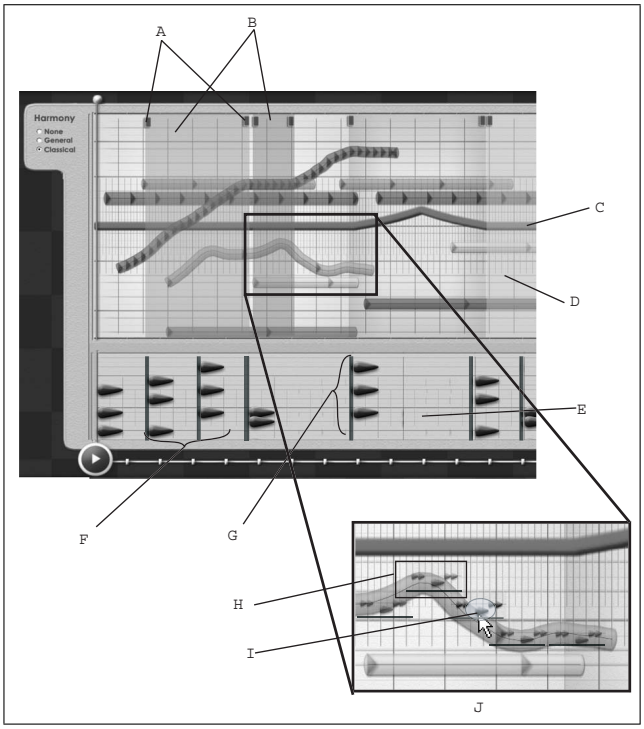

Figure 9. Detailed view of new Hyperscore design. (A) Tabs for moving harmony sections. (B) Colored regions indicating type of harmonic sub-progression. (C) The harmony line. (D) Sketch window canvas. (E) Drop-down harmony grid showing chords generated by the harmony line in piano-roll notation. (F) Notes displayed for each chord change. (G) Tabs to shift chords in time. (H) and (I) Graphics of notes that appear when the zoom level is very high. (J) Close-up of a line in the sketch window at high zoom.

6. CURRENT AND FUTURE DEVELOPMENT

Networked real-time collaboration features are being added and are nearly complete at the current time. Socket-based connections between remote instances of Hyperscore allow several people to work together on a single composition. Several collaboration modes are supported. In server-client mode, all the edit operations in the server Hyperscore instance are replicated on the client machines (that are in read-only mode). This mode is useful for classroom situations and remote demos. Collaborative mode allows each user to modify any item in their instance of Hyperscore and have the edit operations sent to the session server and then broadcast to all the users in the session. The session server arbitrates what is the "correct state" of the file. A chat room feature is also present to allow collaborators to discuss how the composition is progressing.

There are many additional features that should be implemented for Hyperscore to realize its full potential. One major change would be to allow direct editing at the individual note level within the sketch window as opposed to permitting such changes only through altering motives or line-reshaping (Figure 9 inset). There are some user interface design challenges to making note-editing a seamless operation because the notes are algorithmically generated from the combination of motives, sketches, and harmony line. Once a note is manually edited, it should be flagged as such and it should be drawn in a special color. If any of the three source items are then changed, should the edited note remain or should it be overwritten? It generally depends on the type of modification that occurs, and this must be tested.

The harmony line, while powerful and expressive, is too high-level for some users and difficult to control when a specific desired harmonic effect is desired. The user interface for describing the harmonic progression can be improved by generalizing it from the current specific library of harmonic progressions that are generated as a result of the harmony line shape parsing. As shown in Figure 9, a general-purpose harmony tool has been designed to allow people to edit their own chord progressions as they relate to the overall composition. It allows the visualization and editing of the chord progressions and timings that are originally generated from the harmony line. The notes in each chord are displayed in a window below the sketch window. Any note in the chord can be modified, and chord notes can be added or deleted. Also, the chord as a whole can be selected and moved in time (horizontally) to change which section of music in the sketch window it is modifying. It is also desirable to offer other pre-designed harmony modes besides Bach such as those common in jazz, pop, and other genres.

Currently Hyperscore is MIDI-based, but adding full audio capabilities would increase Hyperscore's expressive power tremendously. At the simplest level, audio samples could be used for individual percussion sounds. At the next level of complexity, the audio could be positioned

and lined up with the rest of the MIDI composition and played simultaneously. At the most complex and rich level, audio samples could be used as motives that can then be drawn into a sketch window just like MIDI-based motives. If the audio contains a single voice, it could be analyzed and parsed into discrete pitches that can then be pitch-shifted accordingly and modified to be in a consistent harmonic relationship with the rest of the composition. Once basic audio capabilities are in place, audio effects would be the next natural extension.

Adding external methods of inputting motivic material in both audio and MIDI formats would also be useful. One possibility is a "reverse Hyperscore" process, where the input is a piece of music (in MIDI format, for example) and the output is a Hyperscore rendering. As it basically involves a complex AI problem of inferring motivic structures from an existing piece of music, this would be a far more difficult task than the current graphical-lines-to-music approach. There would need to be some concrete method of breaking down a piece into basic motivic elements, perhaps by doing a statistical analysis of recurring rhythmic, melodic, and harmonic patterns. This process would be greatly assisted by a special type of line (perhaps colored a neutral gray) that would allow the addition of "raw" musical material in a sketch window that is not associated with a motive. After all, while much of music consists of recurring motives, not all of it does.

7. CONCLUSION

Hyperscore facilitates composition through the intelligent mapping of musical features to graphical abstractions, providing a visual analog for what is happening structurally in the music. Users without musical training are able to compose with Hyperscore because it abstracts away complex musical features such as harmony and counterpoint through visualizations of musical structure that are easy to understand and manipulate. The vast musical feature space is reduced and constrained by encouraging conceptualization of a complex musical piece as a composition of short, easily conceived, musical phrases.

Hyperscore successfully makes the process of composing music accessible to a wide audience by removing the requirements to read staff notation and have music theory knowledge, a significant barrier to entry for many potential composers, and by providing visual controls that help users add harmonic progressions to their compositions while retaining the essential "intent" of their melodic contours. Future Hyperscore development will extend the visual metaphor further by allowing more fine-grained control of the musical output to help composers express themselves even more powerfully.

8. REFERENCES

[1] Assayag, G., et al. "Computer Assisted Composition at Ircam: PatchWork and OpenMusic." *Computer Music Jounral*, vol. 23, no. 3, 1999.

[2] Cope, D. *Experiments in Musical Intelligence*. A-R Editions, Madison, Wisconsin, 1996.

[3] Farbood, M. Hyperscore pieces composed by children for the Toy Symphony project. http://www.media.mit.edu/hyperins/HSpieces/.

[4] Farbood, M. *Hyperscore: A New Approach to Interactive, Computer-Generated Music.*, Master's thesis, MIT, 2001.

[5] Farbood, M., Pasztor, E., and Jennings, K. "Hyperscore: A Graphical Sketchpad for Novice Composers." *IEEE Computer Graphics and Applications*, 24(1).

[6] Farbood, M. Hyperscore piece composed by children. http://web.media.mit.edu/~ mary/hyperscore.html.

[7] Farbood, M. *A Quantitative, Parametric Model of Musical Tension.* Ph.D. Thesis. MIT Media Laboratory, 2006.

[8] Iwai, T. http://ns05.iamas.ac.jp/~ iwai/ simtunes, 1996.

[9] Lohner, H. "The UPIC System: A User's Report." *Computer Music Journal*, 10(4):42-49, 1986.

[10] Machover, T. Toy Symphony project website. http://www.toysymphony.net, 2003.

[11] Mathews, M. V., and Rosler, L. "Graphical Language for the Scores of Computer-Generated Sounds." *Perspective of New Music*, 6(2):92-118, 1968.

[12] Subotnik, M. Musical Sketch Pads online activity. http://www.creatingmusic.com/mmm, 1999-2007.

[13] Wenger, E. Metasynth software website. http://www.metasynth.com, 2001-2007.

[14] Wright J., et al. "CyberBand: A 'Hands On' Music Composition Program." In *Proceedings of the 1997 International Computer Music Conference*, Univ. of Thessaloniki.