hpDJ: An automated DJ with floorshow feedback

Dave Cliff
Digital Media Systems Laboratory
HP Laboratories Bristol
HPL-2005-88
May 20, 2005*

This report describes extensions to the hpDJ automated dance-music disk-jockey system, that allow it to monitor the responses of a crowd of listeners and react to their changing responses. Although the initial intention was simply to allow the crowd responses to guide the selection of the next pre-recorded music song to play, the nature of the feedback is such that it can be used to help compose new variations on existing tunes, or potentially entirely new music too.

* Internal Accession Date Only
Approved for External Publication
© Copyright 2005 Hewlett-Packard Development Company, L.P.

Chapter N

hpDJ: An automated DJ with floorshow feedback

Dave Cliff

This is the floorshow the last ideal
It's populist got mass appeal
The old religion redefined
For the facile futile totally blind.

Mundane by day inane at night
Pagan playing in the flashing light
In the violet hour to the violent sound
Going round and around and around and around and around

The Sisters of Mercy, *Floorshow.* (A. Eldritch, 1982).

1 Introduction

Many radio stations and nightclubs employ Disk-Jockeys (DJs) to provide a continuous uninterrupted stream or "mix" of dance music, built from a sequence of individual song-tracks. In the last decade, commercial pre-recorded compilation audio CDs of DJ mixes have become a significant growth market. DJs exercise skill in deciding an appropriate sequence of tracks and in mixing 'seamlessly' from one track to the next. Online access to large-scale archives of digitized music via automated music information retrieval systems offers users the possibility of discovering many songs that they like, but the majority of consumers are unlikely to want to learn the DJ skills of sequencing and mixing, and even if they had such skills, they may not have the time to devote to the mixing task. This chapter starts with a description of *hpDJ*, an automatic DJ system in which compilations of dance-

music can be sequenced and seamlessly mixed by computer, with minimal user involvement. The user may specify a selection of tracks, and may give a qualitative indication of the type of mix required. The resultant mix can be presented as a continuous single digital audio file, whether for burning to CD, or for play-out from a personal playback device such as an iPod, or for play-out to rooms full of dancers in a nightclub. Results from an early version of this system have been tested on an audience of patrons in a London nightclub, with very favourable results. Subsequent to that experiment, we designed technologies that allow the hpDJ system to monitor the responses of crowds of dancers (or listeners), so that hpDJ can dynamically react to those responses from the crowd. The initial intention was that hpDJ would monitor the crowd's reaction to the song-track currently being played, and use that response to guide its selection of subsequent song-tracks in the mix. In that version, it was assumed that all the song-tracks existed in some archive or library of pre-recorded files. However, once reliable crowd-monitoring technology is available, it becomes possible to use the crowd-response data to dynamically "remix" existing song-tracks (i.e, alter the track in some way, tailoring it to the response of the crowd) and even to dynamically "compose" *new* song-tracks suited to that crowd. Thus, the music played by hpDJ to any particular crowd of listeners on any particular night becomes a direct function of that particular crowd's particular responses on that particular night. On a different night, the same crowd of people might react in a different way, thereby leading hpDJ to create different music. Thus, the music composed and played by hpDJ could be viewed as an emergent property of the dynamic interaction between the computer system and the crowd, and the crowd could then be viewed as having collectively collaborated on composing the music that was played on that night, but the act of collaboration is also one of *consumption*: it's the crowd's appreciation of currently-playing music that leads hpDJ to create the next piece of music. This *en masse* collective composition raises some interesting legal issues regarding the ownership of the composition (i.e.: who, exactly, is the author of the work?), but revenue-generating businesses can nevertheless plausibly be built from such technologies.

2 Background: What a DJ Does

What will happen when the major problems in music information retrieval are solved? Imagine if they were solved now, so the 1,000,000-plus songs held by sites such as Mp3.com or Napster.com could be automatically ranked in order of similarity to your entire record collection, or maybe your current favorite five songs. The resultant ranking would be a personalized music recommendation service based not on the purchasing patterns of strangers, but on your personal taste in music. This could be a good way of finding new music to listen to.

Say that such a recommendation service came up with a bunch of songs. How would you want them presented to you? Maybe streamed over the web as a "virtual radio" channel, or maybe burnt onto a CD, or possibly downloaded to an mp3 player such as an Apple *iPod*. However, many young(ish) people listening to radio, or dancing to CDs, want their songs to have been 'mixed' by a disk-jockey (DJ). The job of a DJ isn't simply just playing a bunch of records. There's art and skill in deciding the order of the records, and in mixing between successive records.

For these reasons, many radio stations and nightclubs employ DJs to provide a continuous stream or "mix" of music, built from a sequence of individual song-tracks. Moreover, sales of commercial compilation CDs of DJ mixes (a type of CD unknown until 1992: Brewster & Broughton 1999, p.368) have boomed in recent years, constituting a major sector of chart CD sales (in the UK at least). The London *Ministry of Sound* nightclub was estimated to have income from sales of its compilation CDs (produced by its own *Sound of Ministry* independent record label*)* in excess of £20m for the year 1999 (Kershaw 2000, p.60), although there are reports of sales having subsequently slowed in this sector. The shelf-life of a typical DJ compilation CD is short (often no more than 6 months), but in that time it may sell 500,000 copies (Kershaw 2000, p.60).

In recent years, DJ's have become a new breed of music performer (Haslam, 2001). Top DJs are international stars, earning millions of dollars. According to Kershaw (2000), the fee a top DJ receives for producing a compilation CD (a task that may take little more than a couple of hours) may be up to £50,000. Kids who want to be cool want to be DJs: sales of DJ equipment now exceed sales of guitars in the UK. Nevertheless, working as a DJ requires skill at two levels: the macro-level of *sequencing* and the micro-level of *mixing*.

Sequencing (also sometimes referred to as *programming*) involves deciding an appropriate ordering of tracks. While this is manifestly dependent on the DJ's personal taste in music, there is an element to sequencing that is somewhat more mechanistic. In many instances, the music's tempo (traditionally measured in units of beats-per-minute or "bpm") will be systematically and smoothly varied over the duration of the DJ's playing session (which typically lasts anything from 30-40 minutes to 5 or 6 hours). The tempo is dynamically varied to follow some trajectory, in a manner analogous to the distinct movements that constitute a symphony in classical music. In a nightclub, there will be definite periods of "warm-up" (when the tempo of the tracks rises over time – encouraging the clientele onto the dance-floor), plateaus (keeping the dancers dancing) and peaks (aimed at driving the dancers into a brief frenzy, after which they need to buy another drink). Toward the end of a DJ session, there may be a period where the tempo is progressively reduced (the "come-down" or "chilling it out"), to start to encourage people to think about leaving, or about buying another drink. Commercial DJ-mixed compilation CDs almost always follow some such trajectory – sometimes split across multiple disks.

The micro-level of mixing 'seamlessly' from one track to the next depends on artful “cross-fading”: fading down the volume of the outgoing track while simultaneously bringing up the volume of the incoming track. DJs typically employ multi-channel audio mixers with at least two input channels (each of which is usually stereo), and most often the cross-fade is effected by moving a linear slider across from its extreme left position (where the output of the mixer is 100% of the signal from input channel A; and 0% of the signal from input-channel B) through its mid-point (50% A and 50% B); to its extreme right position (0% A, 100% B). Thus, for some duration during a cross-fade, both tracks will be audible simultaneously: this works best if the two tracks are playing at the same tempo and in perfect synchrony (that is, in more technical language, with zero phase difference between the major rhythmic elements of the two tracks). Getting the two tracks to play at the same tempo and in synchrony is a process known as “beat-matching”, which allows one track to be faded into the next without any discernable alteration in the underlying rhythmic beat. Figure 1 shows the effects on the output mix of a poorly executed cross-fade with no beat-matching, while Figure 2 shows the results of a well-executed cross-fade.

Hence, seamless mixing often requires dynamic alteration in the pitch, tempo, and phase of the two tracks being mixed between. Alterations in pitch and tempo are achieved by reducing or increasing the playback speed of a track, while phase differences are rectified by very briefly slowing or pausing the playback of one of the tracks. Sometimes it is not possible to beat-match two tracks because even when their tempo is identical and there is no phase difference, their interaction sounds bad. In such cases the DJ may choose to cross-fade at a point where the beat is absent in one of the tracks – that is, during a so-called “breakdown” in the beat of the track, or alternatively to apply filters to either or both of the two audio signals in the cross-fade, boosting energy in some frequency ranges and/or reducing energy in others. This latter technique is known colloquially as “EQ’ing” (from *frequency EQualization*). An example of EQing in a cross-fade might be to cut (or “kill”) all high-frequency energy in the new (incoming) track, and then during the cross-fade the DJ might progressively filter out (i.e. reduce from full to zero) the bass frequencies in the old (outgoing) track, while progressively filtering in (i.e. bringing up from zero to full) the bass frequencies of the incoming track. Thus, for some part of the cross-fade, the audience will hear the low frequency components (e.g. bass drum, bass synth) of the incoming track, but with the high-frequency components (e.g. snare drum, hand-claps, voice) of the outgoing track still dominant in the mix. At some appropriate point soon after the cross-fade, the DJ would then bring back the high-frequency components of the new track by cutting out the high-frequency filter. Basic DJ audio mixers typically offer rotary control knobs for two or three limited-bandwidth frequency filters (e.g. “high” and “low”; or “high” and “mid” and “low); more sophisticated mixers also offer two or three corresponding “kill

switches" which each cut their specified frequency range to zero "instantly" (i.e., at the push of the button) rather than requiring the DJ to twist a knob.

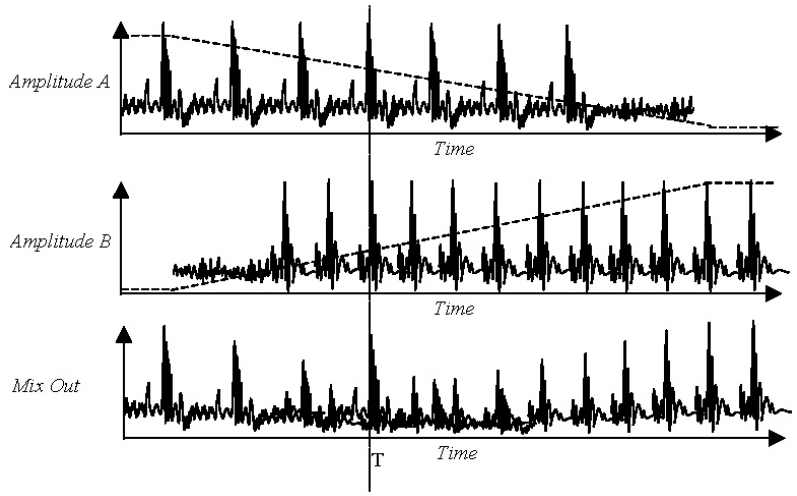

Figure 1. Cross-fading done badly*. The upper two graphs show illustrative amplitude-time plots of the audio in two songs being cross-faded:* A *is the outgoing track and* B *(with a faster tempo) is the incoming track. If the amplitude is signal strength following low-pass filtering, then the pronounced peaks are likely to represent the songs' underlying beat (i.e., the bass drum). The dashed diagonal lines show the relative volumes of tracks* A *and* B *during the cross-fade: note that the sum of the two volumes is constant. Note also that the beats in* A *and* B *are only coincident at time* T *(indicated by the vertical dotted line). The bottom graph shows the resulting mixed output. Because the beats in* A *and* B *are not coincident elsewhere, there is a noticeable drop in the amplitude of the beats in the mix. Also, around time* T *the beats in the two tracks combine to give a brief section in the output mix where there is an audible beat-pattern that is quasi-periodic and that has approximately twice the tempo of* A *and* B.

The audio-source hardware used by DJs usually consists of two or more playback devices, or "decks". Each deck typically provides a stereo input to an audio mixer that allows cross-fading between two or more of its inputs. For historical reasons, the most popular music playback technology is still analog 12-inch vinyl disks rather than digital Compact Discs (CDs), although the market penetration of CDs does appear to be increasing rapidly. DJ decks differ from domestic hi-fi machines in several important respects. Both for vinyl turntables and for digital CD

players, DJ versions of these devices will have smoothly-variable controls that can alter playback speed: typically by up to plus or minus around 10% of the normal speed. This allows the DJ to beat-match the tracks being played from the two devices. On analog vinyl turntables, alterations in playback speed will affect both the tempo and the pitch of the recording being played. The same is true of lower-cost CD decks, while more expensive CD decks use digital signal processing (DSP) techniques to allow pitch and tempo to be varied independently. Alteration in phase is achieved on a CD deck via a jog-wheel controller, while on a vinyl turntable the DJ's fingers are used to either push the vinyl disk forwards slightly to give a momentary increase in playback speed, or to "brake" the disk, momentarily slowing it down.

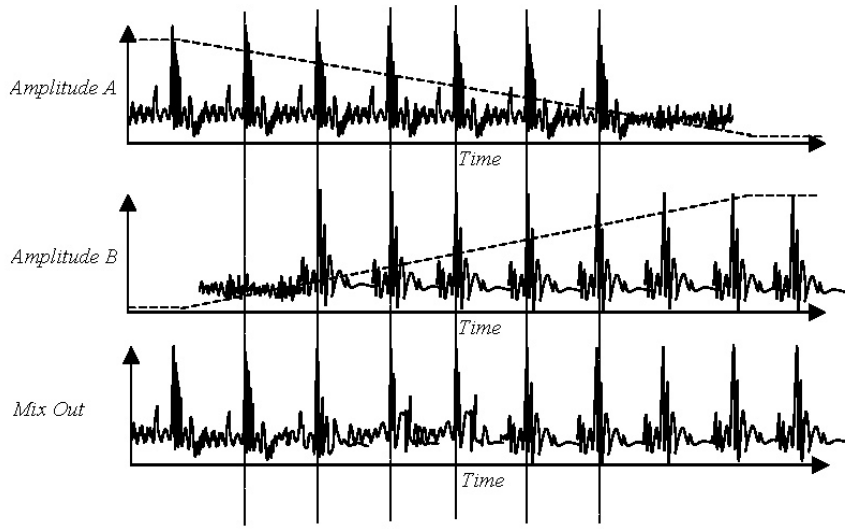

Figure 2. Cross-fading done well*. As in Figure 1, the upper two graphs show impressionistic amplitude-time plots of the audio in two songs being cross-faded where* A *is the outgoing track and* B *is the incoming track. But here track* B *has been time-stretched (e.g. by slowing its playback speed) so that its tempo matches that of track* A*, and the two tracks are synchronized such that there is no phase difference between their beat patterns. The two tracks beat-match for 6 beats (indicated by the vertical dotted lines – note that in practice the beat-matching would last for many more beats). In consequence, the mix output shows no discernable drop in amplitude, and shows a constant beat tempo as the two tracks are cross-faded. Once the cross-fade is complete, the playback speed of track* B *may be gradually increased (reducing the time-stretch back to zero).*

The original hpDJ computer system automates these DJ tasks, and was initially designed to be used as a component of a user interface in commercial music information retrieval systems or digital entertainment centers. hpDJ starts with some method for specifying a collection of song-tracks; those songs are then automatically sequenced to follow some tempo trajectory; and they are then seamlessly mixed, without any need for further human intervention and without any need for human preprocessing of the tracks.

The original (and so far only) version of hpDJ operates best on "dance" styles of music, where the rhythmic element of the music is very regular and pronounced. In the sublime poetry of English Law, these styles are defined in Clause 58 (1) (b) of the *1994 Criminal Justice Act* as "…sounds wholly or predominantly characterized by the emission of a succession of repetitive beats." Such music styles include those popularly known as "disco", "electronica", "house", "garage", "techno", "hip-hop", "drum n bass", and "trance". These styles are the mainstay of many nightclubs and of dance-oriented radio stations, and they regularly constitute the majority of the songs in the national top-twenty charts of many countries. In fact, the styles known as "house", "techno" and "trance" typically have the most regular beat-patterns of all, and are also very popular, and so hpDJ was first constructed to work with those. We see no reasons in principle why hpDJ could not be extended to work with the more complex beat patterns of other genres, although we have not yet pursued this in any depth.

Many prominent dance-music DJs also have a hand in producing new recordings, and in the latter part of this chapter we describe how hpDJ echoes this,. It is this extensions of hpDJ that allows crowds of listeners to collaborate (via their interactions with hpDJ) on the creation of new song-tracks. And that's the reason why the rather unwieldy phrase "song-track" has been used throughout this chapter to denote a single complete recording, or "song", even though in many cases the song will contain no vocal element, and even though most people commonly use the word "track" to mean "song" (as in: "how many tracks are there on your new CD?"). The usual colloquial usage of the word "track" as a synonym for "song" is avoided here because we need to reserve the word "track" for the specific context of creating *multi-track recordings* of songs. That is, in the recording of a song, multiple separate audio tracks are mixed down to create the stereo audio data. For example there might be one track each for the drums, for the bass guitar or synthesizer, for the rhythm guitar/synthesizer, for a lead guitar/synth, for lead vocals, for backing vocals, for the piano, and for the brass or horns section; so eight separate tracks (each of which might itself be a stereo pair) are mixed down to create the final audio recording. In the vast majority of dance music, each track within a song involves a pattern of repetitions of short sequences of music, perhaps only one or two bars long. Frequently, these sequences are not played by musicians (or by programmed synthesizers but rather they are actually digital audio samples, played in repetitive

loops, and many dance-music producers use multitrack nonlinear arrangement and editing systems to compose their songs: popular products include Sony's *Acid* (Sony, 2005); Cakewalk *Sonar* (Cakewalk, 2005); Digidesign's *ProTools* (Digidesign, 2005); Steinberg's *Cubase* (Steinberg, 2005); and Apple's *Logic Audio* (Apple, 2005).

Those DJs with an involvement in music production often start out by *remixing* existing songs. This involves being given access to the original multi-track source material and altering some or all of the tracks in the song. So, for example, a new bass line and an altered vocal could be recorded and these could replace the original bass and vocal tracks in the song, leaving all the other component tracks in their original form, such that the remixed version of the song is clearly a revised version of the original. However, more extreme remixes show ever greater departures from the initial song, and in some cases the remixed version of a song is barely recognizable as having any resemblance to its source.

Relevant prior work is reviewed in Section 3. Section 4 then describes hpDJ in detail. In Section 5 we give results from a test of hpDJ in a London nightclub, and in Section 6 we discuss the extensions to the system that allow it to monitor the audience's reaction to the music as it is playing, and to use this crowd-feedback data to alter the selection of tracks being played and also to dynamically remix and compose new tracks – something we discuss in Section 7.

3 Related Work

The European patent application entitled *Automatically performed crossover between two consecutively played back sets of audio data* (L'Hopital, 1999) claims the invention of a solution to the problem of automating what DJs do, but has the following disadvantages:

- It requires pre-specified "begin" (end-of-fade-in) and "end" (start-of-fade-out) cue-markers to be added to each track's audio data. It gives no indication of any automatic method for doing this, and so the only reasonable interpretation is that skilled human operators are employed to decide on these begin and end points for each and every track.

- Each track has only one "begin" and one "end" marker, whereas in most situations the end of fade-in and the start of fade-out for any one track will depend on the circumstances of its usage (i.e., the particular sequence it is being used in, and its location within that sequence).

- In the third claim of L'Hopital's patent, varying the speed of playback over the "begin" or "end" periods of a track is claimed as an aspect of the invention. Yet

no method or apparatus is specified or claimed for dealing with the nontrivial effects that variations in playback speed routinely have on the pitch, tempo, and phase of the tracks being mixed between.

- It says nothing about ordering of tracks within an extended sequence of tracks (i.e., more than two) and the temporal evolution (trajectory) of music tempo that skilled DJ's devise in such extended sequences.

All of these deficiencies are remedied in hpDJ.

A commercial product called *Databeat DJ Master* is marketed by Sound Management Services Ltd of Newbury, UK, to bars and pubs (see Databeat, 2005). At the time of writing, *Databeat* is installed in over 1000 sites around the world, with remote updating of each installation from the *Databeat* archive. All music in the *Databeat* system is catalogued by human operatives who record production data (such as year of release) along with data used by their proprietary mixing software. This mixing data includes the start-chord, end-chord, track tempo (bpm), and the location of (human-placed) “begin” and “end” cue-points similar to those involved in the method claimed by L'Hopital. Thus, unlike hpDJ, the *Databeat* system is not fully automatic in that it requires human operatives to generate the cataloging meta-data. Details of how the human-generated meta-data is employed by the *Databeat* system are not available.

With the rise in popularity of DJing as a pastime for young people, a number of software vendors have started to offer "virtual DJ" systems that give a software simulation of the physical hardware used by a DJ. In most cases, the software amounts to a graphical user interface (GUI) showing two simulated decks and a simulated mixer. The user selects digital audio files to be "played" by the two decks and has the capability to allow the user to beat-match by altering the playback speeds of the tracks and also by altering their relative phase by "jogging" the tracks slightly forwards or backwards in time. However, when using such software, just as when working with real physical decks and mixers, all the DJing skills in producing the mix have to come from the human operator: in this respect the computer is entirely passive. Thus no such virtual DJ software packages are comparable to hpDJ, because they do not automate the tasks performed by the human DJ.

One notable product that goes beyond this is *MixMeister*, produced by MixMeister Technology of Seattle, Washington, USA (Mixmeister, 2005); a company founded in May 2000. MixMeister works in a similar fashion to hpDJ. It allows a user to define a "playlist", i.e. a set of song-tracks to be mixed, and it then analyses those songs to determine their tempo, and can perform beat-analysis to allow automated "snap-to-beat" positioning of one track relative to another in time, thereby giving a form of beat-matching. MixMeister has an attractive and well-designed GUI, allowing the user to vary the arrangement and settings of the mix, in

a manner similar to the professional multi-track music-production software systems described in Section 2.

This similarity with music-production software is revealing. *MixMeister* offers a GUI onto a set of tools that allow a user to produce a DJ-style continuous mix from a pre-existing playlist of songs, and thus MixMeister *assists* rather than *replaces* the human user in the process of creating the mix. At the time of writing, MixMeister still requires the human user to select the ordering of the tracks in the playlist (the nearest to automated sequencing it can offer is to sort the tracks by tempo, either into an ascending list or a descending list), while hpDJ has much more sophisticated sequencing capabilities. There are also more detailed points of difference, such as the fact that MixMeister's automated beat-matching works only on 8-bar overlapped sections of music (shorter or longer cross-fades require user intervention).

4 hpDJ Version 1: hands-free automatic DJing

Starting with access to a collection of songs stored as digital audio files (in any format – mp3, wav, etc), the operation of the first version of hpDJ can be summarized as follows.

It takes as input a list of desired tracks (which may have been specified by the user, or may come from another source such as an automatic recommendation service, or a random picker). This list of *n* tracks is referred to here as the *set*.

The first stage involves determining a sequence for the set, where the degree of user involvement in the sequencing process is variable from fully user-specified to fully automatic. The digital audio tracks do not require any pre-processing to locate fade begin and end points, because these points are calculated dynamically for each sequence and indeed the fade-in and fade-out points for any one track are likely to vary from sequence to sequence. We use pre-established digital signal processing (DSP) algorithms to automatically vary the pitch and tempo (i.e., the playback speed) of tracks as appropriate to the particular sequence, and the process then automatically sets the relative phase of successive tracks with high precision, to ensure seamless beat-matched mixing. The resultant continuous large file of digital audio can be produced as output for subsequent recording (e.g. burning onto CD) or play-out (e.g. over audio broadcast or narrowcast systems, or over a nightclub public-address sound system). Additional data, such as the time-points at which one track transitions to another, may also be recorded by the system (e.g. so as to provide a table of contents for a CD to be written with time indices for each track). Individual steps in the process are described below. Further details are available elsewhere (Cliff, 2002, 2003a, 2003b, 2003c). The process is described here as a linear sequence of steps, but in Section 4.4 we discuss nonlinear versions.

4.1 Track Mapping

Beat-detection techniques similar to those developed by other authors (e.g. Yamada et al, 1997; Scheirer 1998) are used to determine a *tempo-map* for each of the tracks to go into the mix. The tempo-map is an indication of the bpm measured at intervals across the duration of the track. Figure 3 shows a schematic illustration of the beats in a sample of music and the corresponding tempo-map.

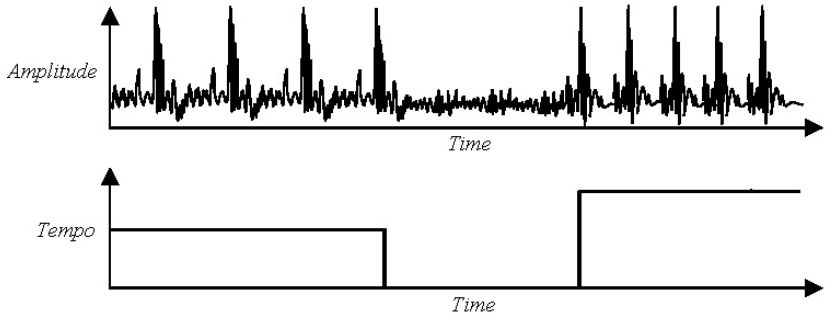

Figure 3: Tempo-map. *The upper graph shows a schematic amplitude-time plot for a section of a song where a tempo change occurs following a "breakdown". The lower graph schematically illustrates a corresponding tempo-map showing the initial lower tempo, followed by the breakdown (zero tempo), followed by the subsequent return of the beat at a higher tempo.*

Similarly well-established DSP techniques can be used to determine maps of amplitude and possibly also pitch/key for each track. These maps are dependent only on the original recorded version of the track, and so could be saved for the next time the track is used, or could all be computed in advance for each track in the music collection.

4.2 Trajectory Specification

The sequence of tracks can be fully and explicitly specified by the user, or sequencing can be completely automated, or it can be partially automated with some guidance from the user. This guidance can take the form of the user specifying a qualitative tempo trajectory (QTT) and optionally also by specifying some ordering constraints (e.g. “don’t play Track A before Track B”). A QTT is a specification of how the tempo should vary over the duration of the mix, expressed in relative, rather than absolute, terms. This allows the same QTT to be used when compiling separate

mixes of different durations, or of different tempo-ranges. For instance, a simple “warm-up” QTT would show a monotonic increase in tempo from a minimum value at the start of the mix to a maximum value at the end of the mix. A graphic representation of this would be to plot a straight upward-sloping line on a graph of tempo over time: example QTTs are illustrated in Figure 4. Significantly, the duration of the mix is not explicitly specified, so the same QTT could be used for a mix lasting thirty minutes, and for one lasting three hours. Similarly, the bpm values of the minimum and maximum tempos in the mix are also unstated, thereby allowing the same QTT to be used for mixing both a compilation where all tracks have tempos in the range 100-120bpm, and for one where the set's tempo range is 125-145bpm.

The QTT for a mix might be directly specified by the user, or chosen by the user from a set of pre-specified QTTs, or randomly chosen by the system from that set of pre-specified QTTs. The user may also specify a maximum time duration for the mix (e.g., in preparing a mix to be burnt to a standard-format CD, the duration should be no longer than 74 minutes).

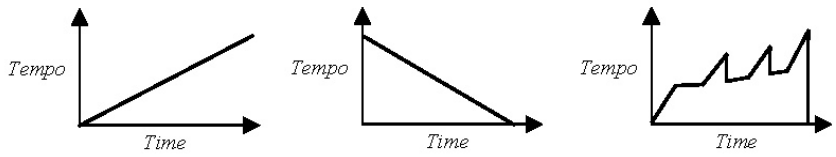

Figure 4: Qualitative Tempo Trajectories (QTTs). *The left-hand graph shows a QTT for a "warm-up" set. The center graph shows a QTT for a "come-down" set. The right-hand graph shows a QTT suitable for a protracted set on radio or in a nightclub: after the initial warm up comes a plateau that is followed by a sequence of three peaks of successively higher maximal tempo, with the set ending immediately after the fastest song.*

4.3 Sequencing

The QTT imposes constraints on the sequence of the tracks, constituting a partial ordering. For example, the (qualitative) point in the mix where the lowest tempo is specified on the QTT indicates the approximate location of the slowest track in the set; and the point where the highest tempo appears in the QTT indicates the approximate location of the fastest track in the set. Turning these approximate indications into a concrete sequence is a straightforward procedure.

The QTT is discretized by dividing it into *n* sections. The tempos of these QTT sections are then ranked in order from highest to lowest. The tracks in the set are also sorted in order of their overall native tempo, from highest to lowest (a track's overall tempo is taken as the average of the nonzero tempos recorded over track's tempo-map, when the track is played at its "native" speed). These two ordered lists are then used to determine the sequence, with the highest-tempo track being assigned to the highest-tempo QTT section, the second-highest-tempo track being assigned to the second-highest tempo QTT section, and so on, as illustrated in Figure 5.

Elementary constraint-satisfaction techniques can be used to check for violations of any of the user-supplied ordering constraints and to take appropriate action when violations are detected. The end result is a list of the tracks in the set, in the order they are to appear in the mix: this list is the sequence for the mix.

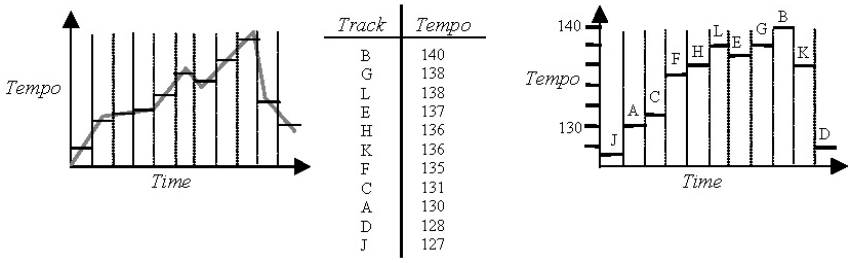

Figure 5: Sequencing. *Left: the QTT is discretized by dividing it into* n *slots (*n*=11 here). Center: the* n *tracks are ranked by tempo. Right: the highest-tempo-ranked track is assigned to the highest-tempo QTT slot; the second-highest-tempo-ranked track assigned to the second-highest-tempo QTT slot, and so on until the lowest-tempo-ranked track is assigned to the lowest-tempo QTT slot. The final sequence of tracks in this example is thus* J-A-C-F-H-L-E-G-B-K-D

4.4 Overlapping

In order for the mix to be "seamless", there should be no "dead-spots" between tracks. While the avoidance of absolute silences is trivial, it is insufficient because many dance-music tracks have long (and relatively boring) "intro" (start) and "outro" (end) sections, where often the main melody or vocal content is absent, with only the rhythmic component of the song being present. Few listeners would want to hear the outro of one track playing to its very end, followed by the intro of the next track played from its very beginning. Indeed, the intention of the music producers is that these intro/outro sections are to be played while cross-fading from/to the

outgoing/incoming track in the mix. Thus, the tracks in the mix have to be overlapped.

Determining the degree of overlap between tracks depends on whether the user has specified a maximum duration for the mix. If no duration has been specified, an initial arrangement of overlaps can be set by making each track overlap with the next by some pre-specified amount – either a fixed number of seconds, or a number of seconds that is a fixed proportion of that track's duration. If a mix-duration of d seconds has been specified, and the total combined length of the n tracks in the mix is l seconds, then the initial arrangement of overlaps can be set by overlapping each pair of tracks by $(l-d)/(n-1)$ seconds. Note that this assumes that $d<l$ (if $d>=l$ then the duration set by the user is irrelevant, and the overlap is set as if no duration was specified).

Once the initial overlaps have been determined, a number of fine-tuning heuristics can be automatically applied. For example, if on examining the tempo-maps for two tracks in the areas where they are currently overlapped shows that the planned overlap occurs near to a position where either track shows a beat "breakdown", the overlap point may be moved to allow the cross-fade to occur during the breakdown. Also, if a maximum duration has been specified for the mix, moves that lengthen the mix-duration are forbidden. A number of other overlap-moving heuristics have been developed. Once any such moves have been executed, the tempo-maps for the tracks are combined to create an overall tempo-map for the entire mix.

4.5 Time-stretching and Beat-Matching

Comparison of the tempo maps for overlapped tracks may reveal areas in the mix where those overlapped tracks have different tempos. In such cases, one or both of the tracks are time-stretched so that the tempos of the tracks in the overlapped portion are near-identical. For example, in a three-track set where the track tempos are 100, 110, and 120 bpm respectively, the first and third tracks could be left in their native states while the second track could be time-stretched so that its tempo is 100bpm for its intro overlap period (when it is cross-faded in over the 100bpm first track). Then, in the main portion of the second track a "gliding" time-stretch could be used that takes the tempo from 100bpm, through the track's native tempo of 110bpm, and up to 120bpm. Then in the second track's outro-overlap section, a constant time-stretch (strictly, a time-compression) could be applied to give a fixed 120bpm tempo while it is cross-faded out under the incoming 120bpm third track. Note that strict equality of tempo is desirable but often unachievable because of imprecision in the tempo-detection process. Alternatively, the same three racks could be mixed by performing gliding stretches to all three tracks so that their tempo

ranges are 100-105, 105-115, and 115-120 respectively. Choices between such alternative but functionally equivalent uses of time-stretching may be made by the user, or may be left to hpDJ.

Once the time-stretching has been applied to bring all the tempos into line, simple beat-detection algorithms can be re-applied to identify the positions of the beats in the tracks and to align overlapping tracks such that there is zero (or minimal) phase difference between them. This involves moving the tracks in the sequence by small amounts of time – typically less than half a second.

4.6 Cross-Fading

Finally, the volumes of the tracks are altered in the overlap areas, in a manner analogous to the cross-fading volume alterations a DJ performs. In the simplest case, linear amplitude decay/increase modulates the outgoing/incoming track, but other curves for these amplitude envelopes are possible.

While a simple "blind" strategy of reducing the volume of the outgoing track while increasing the volume of the incoming track will give acceptable results most of the time, such an approach has an implicit assumption that the amplitude of each track's recording is constant during the cross-fade. In some instances, the music producer will have recorded the music with a fade-in at the start of the "intro" section or fade-out at the end of the "outro" section, and these systematic variations in intrinsic amplitude need to be detected and compensated for. How hpDJ does this is described in (Cliff, 2003c).

However, as was discussed in Section 2, most hardware DJ mixers are built not only with a linear-travel potentiometer for the cross-fader control, but also a small number of rotary potentiometers affecting the frequency equalization or "EQ" for different frequency ranges on each input channel. Each of these rotary controls can be set to cut or boost signal components for that channel within the specified frequency ranges (in much the same way as a linear-travel potentiometer does on a domestic hi-fi *graphic equalizer*). Often the DJ will use these controls in situations where there is a perceived "clash" between the musical components of two tracks being cross-faded. For example the bass guitar component of the incoming track may clash with the bass of the outgoing track, in which case the DJ might choose to reduce or eliminate ("kill") the bass frequencies of one of the tracks during the overlapped period when the cross-fade occurs. A typical arrangement of EQ controls might be a "bass" control for low frequencies up to around 250Hz, a "mid" control for perhaps 0.25-5kHz, and a "high" control for frequencies over 5kHz.

Although uncommon on DJ mixer devices, professional recording-studio mixing desks (which might have 16, 24, or 48 input channels in comparison to the 2, 3, or 4

of a DJ mixer) will often have more sophisticated “swept” EQ controls. Swept EQ controls typically have one rotary controller for the degree of cut or boost, and another rotary that controls the center frequency of the filter. Typically, the number of sweepable EQ controls is fixed to a small number (one or two) and is identical for all input channels; often only the mid-range EQ controllers are sweepable in this manner.

However, because hpDJ operates in the pure software realm of digital signal processing (DSP), it is possible to create as many sweepable band-pass/cut filters as is desired for any particular cross-fade from one track to another. As with traditional hardware mixers, each DSP filter can have variables that control the degree of attenuation or boost, and its center-frequency. In addition to this, the shape of the DSP filter’s transfer function (e.g. the nature and rate of the fall-off or boost) and its bandwidth can also be under automatic control. Recording studios do have filters with these added controls, but such filters (known as a *Parametric EQ*) are too expensive to be built into each channel of professional mixing desks on a many-per-channel basis.

Thus, it becomes possible to specify hpDJ so that it analyses the audio frequency-time spectrogram for the incoming and outgoing tracks in the cross-fade, and uses a number of heuristics to determine how many DSP Parametric EQ filters are necessary and what their settings should be. This can be used to, for instance, selectively suppress the frequencies for a synthesizer melody-line in one track, attempting to make that melody “disappear” while keeping the bass-guitar and percussion elements in place during the cross-fade. By employing simple heuristics for detecting when one component of one track “clashes” with another component of the other track, such aesthetically unpleasant clashes (which may remain despite perfect beat-matching) could be automatically eliminated by hpDJ. Details of this sophisticated cross-fading technique are given in (Cliff, 2003c).

4.7 Nonlinearities

Although the process as described above is linear, starting with a list of tracks to go in the set and progressing through the stages described in Sections 4.1 to 4.5, there are obvious ways in which the process could be altered to be nonlinear or iterative. A nonlinear process could be invoked if the user specifies only a small number of tracks relative to the time limit on the mix and also requests that the unspecified time is filled with tracks chosen by the system. In this case, it may be more appropriate to introduce "wild-card" (unassigned) tracks when sequencing and overlapping, and to then select appropriate songs from some song database to instantiate these wild-cards after sequencing is complete. Deferring automatic selection of songs in this way allows the system's choice of songs to be constrained

by the tempo and/or pitch of the surrounding tracks. In particular, the deferred instantiation of wild-card tracks can be used to bridge over major tempo transitions in the sequence. For example, if the user's specifications and choices result in an incoming 140bpm song having to be mixed into an outgoing 100bpm song with beat-matching during the cross-fade, both tracks would require unacceptably high alterations in their playback speeds. Slowing the fast track by 14% (i.e., setting its playback speed to 86%) reduces its tempo to 120bpm, and speeding the slow track by an extra 20% increases its tempo to 120bpm also, but the songs are likely to sound unappealingly different from the familiar original versions at these playback speeds. In such cases it may be better to add a small number of "wildcard" tracks between the two user-specified tracks: these could be chosen from a song database on the basis of their tempo. In the example just given, if three wildcard tracks are introduced, constrained to have tempos of around 130, 120, and 110bpm, then the tempo changes between successive tracks in the final mix would all require less extreme (and hence more tolerable) alterations in playback speed.

5 The London Nightclub Test

An early version of the hpDJ system was tested in an experiment organized in a London nightclub called *UnderSolo*. For a detailed journalistic report on this experiment, see Graham-Rowe (2000). An invited audience of 72 people, including a number of professional DJs, were asked to listen to two 30-minute sessions of music apparently played by Jesse Rose, a professional DJ and sometime resident DJ at London's *Ministry of Sound* nightclub. The audience was told that one session would be Mr Rose playing live, and the other session would be Mr Rose miming while the crowd heard output from the hpDJ system, and that at the end of the second session the audience would each be asked to decide whether they had heard *live-followed-by-mime* or *mime-followed-by-live*. The same set of five songs were used in each session, but Mr Rose chose the sequence for his session while hpDJ automatically determined its sequence for the mimed session. From the audience's viewpoint, Mr Rose was visible only from the shoulders upward, so it was not possible for the audience to use visual cues to determine whether he was miming or not. Before the audience arrived at the club, Mr Rose was given some time to familiarize himself with the songs and to rehearse his miming. The experiment took place from approximately 8:30pm to 9:30pm on a Tuesday night, with a full bar service available from 6pm (when the audience were allowed into the club) onwards. Thus, as far as was possible, the experiment replicated one of the intended environments into which hpDJ could be deployed.

At the end of the second session, the audience was asked to cast their votes. The result was that 45 people (62.5%) correctly identified that they heard *mime-*

followed-by-live, while 27 people (37.5%) incorrectly voted for the other ordering. Now if hpDJ was truly terrible, presumably 100% of the audience would have made the correct choice and 0% would have made the incorrect choice; and if hpDJ was exactly as good as the human professional, then the audience's best response would be to guess, implying that 50% would be correct and 50% incorrect. Under this reasoning, the worst that the hpDJ could score is 0% incorrect votes and the best is 50%, so the actual score of 37.5% can be expressed as three-quarters (37.5/50=75%) of the best possible score. Although manifestly an *n=1* data-point, the results from this experiment are nevertheless very encouraging indeed.

6 hpDJ Version 2: Direct Crowd Feedback

> *"I can't see that the 'whites-of-their-eyes' relationship between clubbers and DJs is going to be affected in any way by this. ... [DJing] is all about spontaneity, none of which can be supplied by anything other than the human real deal."*
>
> Judge Jules (a top British dance-music DJ) commenting on the *New Scientist* hpDJ nightclub test, on Britain's BBC Radio One *Newsbeat* news programme, 5 January 2001.

In the nightclub test described in the previous section, the hpDJ output was based purely on its analysis of the tempo of the songs and its choice of a QTT: if the audience didn't like the music or the mix, hpDJ had no way of knowing. Thus, one function performed by a human DJ that should also be built into hpDJ is the ability to "read" the audience's reactions to the music as it is played and to alter the subsequent selection of music accordingly.

In response to this perceived lack, we have designed technology that passively monitors the responses of an audience in a suitably pre-wired and instrumented nightclub (Cliff & Wilkinson, 2004). We use the word "passive" here to denote the fact that the audience do not need to actively participate in the monitoring: their presence in the venue is all that is required of them. Multi-modal sensor technologies such as under-floor pressure sensors, laser break-beams, video surveillance (both in visual and infra-red bands), and so on, are used to detect patterns of activity in the bar or nightclub and to infer from this the crowd's reaction to the music being played. A simple set of rules then determines whether the tempo of the music being played should be increased, decreased, or remain unchanged. In effect, the hpDJ commits to an initial QTT but that QTT may then be dynamically altered on the basis of crowd responses, and new song-tracks that fit with the emerging QTT are selected from a database of songs in the nonlinear fashion described in Section 4.6.

Although such passive monitoring is readily achievable using off-the-shelf technologies, it is typically very expensive. One issue is that it is not sufficient to monitor only the dance-floor: knowing that there are twenty people dancing on the dance-floor does not tell you much. Knowing there are twenty people dancing and two hundred people standing around in the bar area tells you that the music is not very popular, and it's probably time to change the tunes; knowing that there are twenty people dancing and the rest of the club is empty tells you that you're doing as well as could be expected. So, you have to monitor pretty much the whole nightclub. Because the costs of adding, calibrating, and maintaining this passive sensor technology to a nightclub are likely to be somewhere between "high" and "prohibitive", and because each such club requires an installation-specific design, we have also designed alternative solutions that achieve the same result but with much more portability and/or less cost. Our first alternative is a highly portable and personal technology that actively monitors the responses of individual members of the audience, using Bluetooth wireless communications links to read a combined sensor/feedback device worn as a wristwatch-sized personal appliance (Cliff & Wilkinson, 2004). The appliance could report on its approximate location using well-established techniques (e.g. triangulation) and it could also contain accelerometers (to detect movement of the arm when dancing); thermometers and galvanic skin resistance sensors (to report on the temperature and perspiration levels of the wearer), and possibly also could monitor the wearer's heart-rates using technologies commonplace from the wristwatch wearable heart-rate monitors currently sold in sports shops. Although this solution is much cheaper and less installation-specific than installing passive sensors throughout a nightclub, providing one such appliance per user in a large-capacity nightclub it is still likely to be too costly for many applications (unless the technology becomes so wildly popular that economies of scale drive the cost per unit down to affordable levels).

For this reason, in a third attempt at allowing users to give feedback to hpDJ, we hit upon the idea of a simple wristwatch transmitter device, with two big buttons (Cliff & Wilkinson, 2004). Let's say that one button is green and has a simple drawing of a smiley face on it, and the other is red with a drawing of a sad face on it. Each member of the crowd in the nightclub wears one such watch. When one of the buttons is pressed, it sends a "vote" to hpDJ over a Bluetooth wireless link. When listeners are enjoying the music, they can signal their pleasure to hpDJ by pressing the green button: the more they press the button (e.g. the longer they hold it down, or the more frequently they hit it), the more they signal to hpDJ that they are enjoying the current song or mix. Conversely, the more they press the red button, the more they signal their lack of enjoyment of the current song or mix. Such a "voting watch" would require comparatively little in the way of internal electronics and so could be produced much more cheaply than the other means of monitoring crowd feedback. As with the initial passive crowd monitoring system, the original intended

use for the voting watch was that responses gathered from the crowd were used to dynamically alter the QTT, and that those dynamic alterations in the QTT would affect what music was selected, so that to fit the current desired tempo. That is, initially the only motivation was to monitor crowd responses in order to guide the selection of songs to add to the mix in the immediate future of the hpDJ "performance".

However, it rapidly became clear that the feedback signal from the crowd is a source of information that could be put to much more use than merely deciding whether to alter the tempo of the music being selected for the mix. And this is true , however that crowd feedback is gathered: it could be gathered from a simpler but less user-friendly source, such as having "voting terminals" positioned around the room, with hardwired rather than wireless connectivity to the hpDJ server; or it could even come from a geographically dispersed "crowd", such as the listeners to an "internet radio" broadcast, voting via their home PCs.

Specifically, having developed such crowd-feedback technology, it becomes possible for the audience to play an active role in the dynamic on-the-fly *composition* of the music they are listening to, thereby dispensing not only with human DJs but potentially also with human recording artists too. That is a development discussed next.

7 The Crowd as a Mass Collaborative Composer

Feedback received from the crowd via the monitoring technologies introduced in the previous section gives hpDJ a means not only of helping to decide what song to play next, but also of estimating the crowd's view of the merits of each song. For the sake of this discussion, let's assume that the feedback data, however it is gathered, is boiled down into one rating-value or "score" from the crowd for each song, and let's say that the score is a percentage so that a song rated at 10% is pretty unpopular while a song with a 90% score is really very popular. And remember that here the notions of "popular" and "unpopular" are relative to the particular crowd that is being monitored or doing the voting or otherwise providing the feedback – a different crowd, or even the same crowd on a different night, might give different scores to the songs.

Now it happens that such single-value feedback scores are commonly found in a popular class of automated optimization systems that draw inspiration from Darwinian evolution via random variation and natural selection – so-called "evolutionary computation" techniques, the most widely practiced of which is a specific approach known as a *genetic algorithm* (see e.g. Goldberg, 1989; Mitchell, 1996). Given the availability of crowd feedback scores, it becomes possible to explore the use of genetic algorithms in automatically designing (i.e. authoring) new

songs, in an attempt to "optimize" those songs (i.e., to create songs that yield high scores from the crowd). To explain how to do this, it is necessary to give a brief general overview of how a genetic algorithm works, before talking about the specifics of how to apply the genetic algorithm in the hpDJ context of using crowd feedback in automatically creating new remixes of existing songs, or indeed in automatically creating entirely new songs.

A genetic algorithm (GA) operates on a bunch of candidate solutions to some problem, referred to as a *population* of *individuals*. For the sake of this discussion, each individual is just a string of values – numbers and/or letters -- and that string of values is referred to as the individual's *genes*. In a GA it is also necessary to have some method of testing an individual, to assign that individual a score known as its *fitness*. To start with, we create an initial population by randomly generating each individual – that is, by randomly choosing values for each gene in each individual. All these randomly-generated individuals can be tested, and assigned a fitness value. Because the individuals in the initial population are all randomly generated, they will all typically score very low fitness values (that is, they are all rubbish), but across the entire population there should be some variation in the scores (that is, some are less rubbish than others). Then, we select individuals for *breeding*, such that the higher an individual's fitness, the more likely it is to be selected. In the breeding process, the genes from two selected "parent" individuals are mixed up to create one or more "child" individuals – in a manner inspired by sexual reproduction in plants and animals. Additional random changes (*mutations*) may also be introduced to the child genomes, to introduce additional variation in the GA's gene-pool. This breeding process continues until we have sufficiently many "children" to replace the "parent" population. At that point, the parent population is thrown away, and the children are then all tested to give them their fitness scores which can then be used to determine which of them will be selected for breeding. This sequence of test-breed-replace is referred to as one *generation*, and (so long as it is set up correctly) a GA will show improvement in fitness scores over a number of successive generations. A common intriguing aspect of GAs is that the final population will show a set of individuals with high fitness scores (i.e., good solutions to the "problem"), yet these solution have not been designed by a human designer and so may possibly show unexpected but attractive "design features", which might be attributed to creative flair if they had been thought up by a human designer or creator.

So, we can consider each song in some collection as an *individual*, and the crowd feedback scores can clearly be used as the *fitness* for each song, but what about the *genes* of a song?

Recall that, as was discussed in Section 2, most dance-music songs are created via multi-track recording techniques, and that the individual tracks on each song are typically some small number of musical phrases or samples, repeated in some

appropriate pattern. For example, the bass track for a song might be composed of a patterned placement/repetition of two distinct phrases, *A* and *B*, and let's say they're each four bars long; then if the placement pattern for these two phrases in the bass track is this *AAABAAABAAABAAABABBBAAAA*, we have 24 placements each of four bars, so a 96-bar track in total.

Now a minimally-different remix of that example track could be generated by replacing one of the two bass phrases with a new phrase, which we'll call *C*. If we choose to replace phrase *A* with phrase *C* then we'd get a new bass track of: *CCCBCCCBCCCBCCCBCBBBCCCC*. But in the language of GAs, we could consider this as a bass-track *mutation* of Gene *A* to Gene *C*. Similarly, we could note that the original bass-track placement pattern has its own internal repetitions: the pattern *AAAB* is repeated four times at the start of the track. So, we might also consider these four-phrase chunks as "genes" in the specification of a placement pattern: if we allow "*X*" to represent the pattern *AAAB*, "*Y*" to represent *ABBB*, and "*Z*" to represent "*BBBB*" then the initial bass-track placement pattern could be written more concisely as *XXXXYZ*, and possible mutations of this pattern include *XXXXZZ*, *XXXXYY*, *XXXYYZ*, and so on.

So the genes for any one track within a song would consist of an encoding of the placement-pattern (e.g. *XXXXYZ*) and a set of mappings from placement-pattern encodings to actual phrase-placement sequences (e.g. *"X=AAAB"*) and a set of specific phrases or samples that are substituted into the phrase-placement sequences. All of these could be subject to mutations, as just described, and also to so-called *crossover* or recombination, which is the GA version of sexual mixing of genes. For example, if one of the parents has a placement pattern *XXXXYYYYZZZZ* and the other parent has placement pattern *yyyyxxzzxxyy* then possible children resulting from the breeding of these two parents could include *yyyyYYYYZZxy* and *XXyyxxYYZZZy*. Of course, all of the discussion so far has been in terms just of one track within the multi-track recording. The genes for each track could be kept separate and considered as different *chromosomes* for the individual song, or the genes could be all strung together into one long gene-sequence for the song; in practice there's not much difference. Note also that this does not require all songs to have the same number of tracks, or for all songs to have music playing in all tracks at all times, as some of the samples or phrases in the gene-pool could represent so many bars of silence, thereby allowing specific tracks to be muted for all or part of a song.

So, we have here a sketch of how to encode a multi-track specification of a song (represented by a set of samples/phrases and a set of placement patterns for those phrases) as the genes of the individuals in a GA; and with the crowd-monitoring technologies we have a means of evaluating the fitness of each song. One important point of departure from the sketch of the GA that was provided above is that, for the hpDJ system to stand a decent chance of generating acceptable or interesting new

remixes and compositions, it is important *not* to start with an initial population that is generated at random. The music resulting from randomly generating songs according to the scheme laid out here would almost definitely be judged by the audience to be really very poor indeed: rubbish, in fact. The point that some of the songs sound less rubbish than others will just not compensate for the fact that, actually, *all* the songs in an initial randomly-generated population will sound like rubbish. If the audience has any sense, they will probably leave the nightclub rather than attempt to dance or otherwise respond to a set of randomly-generated songs. So, the trick is to seed the initial population not with random songs, but instead with an archive of songs that have been written by skilled musicians. The likelihood then is that "mutants" of the original songs really are like minor remixes, and that some of the "child" songs show characteristics of their mixed parentage, representing a "fusion" of different styles of composer/composition within the genre. Thus, the "composition" of new work by hpDJ is not an *ab initio* process, but rather one of successive tinkering with existing forms and of opportunistic plagiarism of ideas from different pre-existing sources. It mirrors a process that is clearly observable in the high-turnover world of the human-composed dance-music industry, where every now and again one innovative producer releases a new song with a particular sample or sound our compositional feature which make that song distinctive (and popular) and which is then quickly copied by a number of other composers, rapidly being replicated in the songs released over subsequent weeks, until it is judged passé or otherwise part of the norm, thereby motivating a search for a new sound or sample or style. So, in essence, the mechanism proposed here for hpDJ just echoes the process that is already evident in the real world.

Of course, there are some legal and commercial considerations. We need to make sure we can do this without breaking any laws or infringing any copyright, and we would like (in principle at least) to be able to actually make money out of hpDJ – if nothing else, it would be good to recoup the costs of building an hpDJ system.

For revenue, the most obvious potential source of income is the people in the crowd doing the dancing and interacting with hpDJ. If a bank of CD-writing hardware is installed somewhere in the nightclub, the punters can be offered copies of that night's music for sale as they leave the club. The fact that many of them will be leaving in an intoxicated state will, presumably, increase the likelihood of purchases being made. The sales pitch is part an appeal to impulse-purchase, and part an appeal to sentimentality: as the music made each night is a (hopefully unique) function of that particular crowd's responses on that particular night, the CDs can be sold to the departing clubbers on the promise that if they don't buy the CD tonight then they will never have the chance to listen to *exactly* that mix of music ever again.

The legalities of getting hpDJ to generate its own remixes and compositions is straightforward enough: the authors of the original songs that seed the initial

population need only to sign over appropriate rights (presumably in return for appropriate compensation). So long as appropriate copyright clearance is given for compositional use of all the constituent samples in the "gene pool", the compositions made up of those samples will not be violating the copyright of the owners of the original samples. For example a successful company called Zero-G (Zero-G, 2005) has for many years sold CDs of original "copyright cleared" samples, where the copyright in the individual samples rests with Zero-G, but the samples are licensed to the user in such a way that, so long as the samples are not re-sold as-is (i.e., so long as they are actually used by being combined with other samples or recordings in the final song), the composer of the song owns the copyright on that song.

There is an old saying in the music industry: "where there's a hit, there's a writ". Now although it is really extremely unlikely that an hpDJ composition would ever climb to the top of the charts, it is worth pointing out here, for sake of completeness, that the authorship (and hence ownership) of the music on the CDs sold to the crowd as they leave the club is just a little bit murky. In principle, it could be argued that *all* of the people in the club whose activity was monitored in any way by the hpDJ system are partial authors of the music. So if a copy of the music they helped to make ever made it to Number One, they would each be due a share of the royalties (or could each have grounds to sue a plagiarist). Such is the future.

8 Conclusion

The hpDJ system described here goes some way towards replacing the tasks performed by human DJs. It has potential use as a component in the user-interface to audio-based consumer digital entertainment systems, converting the audio data stored on such systems from a set of songs into a continuous seamless mix. Such mixes are suitable for play-out over streaming media (e.g., in personalized internet radio), or for writing to an appropriate recording medium (such as CD, the hard disk of an iPod, or a flash ROM card) for subsequent playback, or for playing to crowds of dancers in real nightclubs. Results from the nightclub experiment are promising, and our subsequent development of monitoring technology allows crowd feedback to influence hpDJ's choices of songs, making it even more human-like. The use of human-inspired heuristics in dynamically selecting customized DSP filters for the cross-fade has the potential to allow hpDJ to perform cross-fades in ways that would be virtually impossible for a human DJ playing live. While there is a growing market for software products that give a "virtual" version of traditional human-DJ hardware, and while *MixMeister* provides a pleasant interface to a set of software tools that allow an unskilled human to create professional-quality continuous mixes, hpDJ as described here is as far as we know the first and only system that aims to

totally automate the tasks performed by a human nightclub DJ, including dynamically reacting to the responses from the crowd in real-time. Although we have yet to test Version 2 in a real nightclub, it is clear that the prospect of crowd monitoring opens up new possibilities for the computer-assisted composition of music. But, whereas most computer-aided music composition systems assume a single human author working with the machine, the vision in hpDJ is that the author is an entire crowd of participants, collaborating indirectly, giving feedback as they consume the music. That feedback being generated either actively by the members of the crowd hitting the buttons on their voting watches; or passively by them merely dancing and having a good time, while the computer watches them.

Acknowledgements

Thanks to my HP Labs colleagues Heppie Freeburn and Lakhdip Nagra, who built the first version of hpDJ from my patent specification. Thanks to Duncan Graham-Rowe of *New Scientist* for organising the nightclub test, to Jesse Rose for DJing (and miming!) and to the people there who gave up their time to listen and to comment, especially Zero 7, Coldcut, and DJ Food. Thanks also to Tim Wilkinson of HP Labs for his bluetooth expertise without which the crowd-monitoring stuff would not have been invented. Also, if Judge Jules had been a little more positive about hpDJ Version 1, then we would probably not have been prompted to invent Version 2. So, sincere thanks to Judge Jules for his critique. Ball in your court, Jules.

References

Apple, 2005: website at www.apple.com/logic

Brewster, B., & F. Broughton, 1999: *Last Night a DJ Saved My Life*. London: Hodder Headline.

F. Broughton, & Brewster, B., 2003: *How to DJ Right: The Art and Science of Playing Records*. London: Grove Press.

Cakewalk, 2005: website at www.cakewalk.com

Cliff, D. 2002: *Automated Compilation of Songs*. US patent number 6,344,607; issued on 05 February 2002; assignee: Hewlett-Packard.

Cliff, D., 2003a: *Automated Compilation of Music*. US patent number 6,543,700; issued on 18 March 2003; assignee: Hewlett-Packard.

Cliff, D., 2003b: *Method and Apparatus for Composing a Song.* UK patent number 2,379,076; issued on 17 Sep 2003; assignee: Hewlett-Packard.

Cliff, D., 2003c. *Automated Mixing of Musical Tracks.* UK patent number GB 2,378,626; issued on 19 Nov 2003; assignee: Hewlett-Packard.

Cliff, D., & Wilkinson, T. 2004: *Portable Apparatus Monitoring Reaction of a User to a Performance.* UK patent number GB 2,379,987; issued on 21 April 2004; assignee: Hewlett-Packard.

Databeat, 2005: website at www.databeat.com

Digidesign, 2005: website at www.protools.com

Goldberg, D.E., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley.

Graham-Rowe, D., 2000: "Hang the DJ". *New Scientist* 23rd/30th December 2000, pp.56—59.

Haslam, D., 2001: *Adventures on the Wheels of Steel: The Rise of the Superstar DJs.* London: Fourth Estate.

Kershaw, M., 2000: Compilation Wars. *Mixmag* 104, pp.58—64, January 2000.

L'Hopital, C., 1999: *Automatically performed crossover between two consecutively played back sets of audio data.* European patent application EP0932157A, July 1999.

Mitchell, M., 1996: *An Introduction to Genetic Algorithms.* MIT Press.

Mixmeister, 2005: website at www.mixmeister.com

MP3, 2005: website at www.mp3.com

Napster, 2005: website at www.napster.com

Scheirer, E. D., 1998: "Tempo and Beat Analysis of Acoustic Musical Signals", *Journal of the Acoustic Society of America.* 103(1):588-601, January 1998.

Sony, 2005: website at: mediasoftware.sonypictures.com/products/acidfamily.asp

Steinberg, 2005: website at www.steinberg.de

Yamada, Y., T. Kimura, T. Funada, & G. Inoshita., 1997: *Apparatus for detecting the number of beats.* US Patent 5,614,687, March 1997.

Zero-G, 2005: website at www.zero-g.co.uk