ABSTRACT

This paper explores the concepts of virtuosity and flow in computer music, by looking at the technological, interactive and social factors in soundtracking; a text-based, computer keyboard-manipulated notation for real-time computer-aided music composition. The role of virtuosity in both the personal user experience and the wider demoscene hacker-artist subculture are discussed. Comparisons are made to mainstream music interaction paradigms, such as performance capture and sequencing, where support for virtuosity is present in MIDI devices and episodes of live performance recording, but otherwise impeded by mouse-driven interfaces designed around visual metaphors for novice use, rather than the development of practised skill. Discussions and observations are supported by initial findings from a large-scale, 2-year user study of over 1,000 tracker and sequencer users.

1. INTRODUCTION

We define “virtuosity” as the enabling factor of fluency in a domain (i.e. music composition) through mastery of a device or system [1]. This paper explores the concept in the context of the users, interfaces, and technology of soundtracking, a computer-aided composition technique that stands apart from mainstream digital music – in terms of user and listener demographic, technology and interaction style.

This paper begins with a brief description of flow and tracking technology, before discussing the role of virtuosity in both the tracker user interface (highlighting appropriate contrasts with popular sequencing programs) and the demoscene, a computer art subculture that specializes in realtime, non-interactive audio-visual presentations designed to demonstrate coding and artistic skill (detailed in Section 3.2). To illustrate and explore both phenomena, we present initial findings from a large-scale, 2-year user study of over 1,000 tracker and sequencer users. Using real-time logs of interaction and surveys, the experiment has allowed us to empirically study the nature and development of virtuosity in computer music. Analysis also reveals indications of specific components of flow, within the interaction. Video footage of one expert tracker user was also recorded, to augment the interaction log and allow a closer examination of a composer “in flow”.

2. FLOW IN MUSIC NOTATION USE

We are particularly interested in what motivates end-users to interact with systems, to keep using them for an extended period of time, and to develop mastery with their tools. To this end, we apply Csikszentmihalyi’s theory of flow [4], which describes a trance-like mental state of total focus and concentration on an activity, enabled by a balance of challenge and ability.

Too little challenge and the individual becomes bored, too much and they become anxious. A novice can achieve flow in simple tasks, but gradually develops the ability to tackle greater challenges, towards complete mastery of the domain. At this ultimate point of development, macro-flow, the individual feels in complete control of the tools, the task and themselves, enabling action-awareness merging, where the individual becomes totally absorbed in the activity and oblivious to the outside world. Table 1 summarises the nine characteristics commonly found in descriptions of flow experiences.

<table>
<thead>
<tr>
<th>Characteristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Clear goals and intentions.</td>
</tr>
<tr>
<td>2. Direct and immediate feedback.</td>
</tr>
<tr>
<td>4. Focus and concentration.</td>
</tr>
<tr>
<td>5. Intrinsically rewarding activity.</td>
</tr>
<tr>
<td>6. Distorted sense of time.</td>
</tr>
<tr>
<td>7. Sense of personal control / autonomy.</td>
</tr>
</tbody>
</table>

Table 1. Characteristics of flow experience [4].

Flow has been observed in many activities, including music [5], video games and programming [1]. Tracking, as an intersection of these three domains, thus seems a likely candidate for support of such experiences, which represent an intrinsically-motivated path towards the development of virtuosity [4].

3. SOUNDTRACKING

This section begins with a technical description of the notation and interface, followed by an historical account of technology and social context, notably within the user community and demoscene subculture, where the virtuosic coding talents of practitioners are recognised and valued as much as the artistic content.
3.1. Technical overview

Primarily using text to represent notational elements, a soundtracker (or simply, “tracker”) allows the user to create patterns of note data (often 4 bars) comprising a short passage of music. The music is realised in real-time, traditionally through an integrated sample engine and user-supplied set of samples. These patterns – resembling a spreadsheet in appearance, and analogous to a step-sequencer or player piano in function – are then arranged in a specific order to produce a song. The saved file (or module) stores the song together with all the notes, samples and instrument settings.

![Figure 1](image1.png)

**Figure 1.** A comparison of tracker and score notations.

Tracker programs are primarily controlled using the computer’s QWERTY keyboard, used for nearly all tasks, including the entry and editing of musical data, as well as management and navigation within the program. The text representation allows many parameters to be simply typed, but pitches are entered using a virtual piano (Figure 2). Keyboard shortcuts and macros accelerate almost every aspect of the program, replacing the mouse’s typical role in block selection and cursor navigation, with the provision of rich-cursor movement control. This allows the user to stay at the keyboard without incurring the time penalty or interruption of homing between input devices.

![Figure 2](image2.png)

**Figure 2.** Pitch entry in tracker software.

The majority of interaction takes place in the pattern editor (Figure 3), which also provides a central visual focus for the user. Other screens offer control of song, sample and instrument settings using more traditional methods, such as buttons, sliders and text boxes, but typically play a peripheral role after the initial set-up is complete. However, even these other screens maintain fixed layouts and keyboard focus, permitting the learning of screen configuration (as spatial schemata [13]), for fast visual inspection, navigation and editing.

![Figure 3](image3.png)

**Figure 3.** The Pattern Editor, in Impulse Tracker 2.

Playback is also controlled using the keyboard, providing dedicated keys for playing the current pattern or the whole song, from either the beginning, or from the editing cursor. This last feature enables a tight coupling between editing and auditioning music, which we have shown, in other work, to enable a fast feedback cycle, supporting the embodied learning of music composition through the notation [8], where the notation is quickly resolved into the musical sounds it represents, aiding experimentation and learning in the program. [1]

---

1. Unlike MIDI, the notation represents relative changes in musical parameters over time, in a mode similar to score notation.
Summarising the tracker user experience, Computer Music magazine highlighted the virtuosity supported by use of the keyboard, observing, “The art of tracking has often been likened to a sort of musical touch-typing”. [9]

In a manner similar to expert programming editors such as Emacs2, trackers avoid visual metaphor and traditional music notation abstractions, focusing on a concise textual representation of musical ‘source code’, for fast editing, real-time interpretation and playback by a synthesizer (typically integrated, but also via MIDI). Our hypothesis is that a user’s mastery of the computer keyboard, in controlling a tracker program, supports a level of virtuosity and flow that is currently lacking in aspects of the user experiences of other mainstream computer music methods, like sequencing.

3.2. Social and cultural context

While the development of virtuosity is intrinsically rewarding, allowing it to be studied in the context of flow [4]; the social, extrinsic rewards of exceptional skill have also had a significant influence in the design and use of trackers. This section presents an historical summary of the social background behind tracking, and the role of virtuosity in the demoscene subculture.

Emerging in the late 80’s, the first trackers were based on technologies developed to provide background music in computer games [3]. They ran on home computers with limited processing power, storage space, graphics and audio hardware.3 The crude simplicity of the tracker’s text-based interface and sound engine reflected not only its legacy as a programmer tool, but the limitations of the underlying hardware.

Consequently, the musical capabilities and appeal of early trackers were limited: the early tracker MOD format supported 4 monophonic channels, hosting one of up to 15 instruments (mono, 8-bit, PCM samples). Sample compression, panning, and software mixing were beyond the hardware.4

However, for more technically-minded users, such as young hackers and video gamers, these limitations simply presented a challenge, where individuals competed against one another to defy the apparent limitations of the format. Tricks such as the polyphonic samples (e.g. recorded major or minor chords) and virtual polyphony (interwoven monophonic pitches or samples to create the perceptual illusion of polyphony or multi-timbrality), enabled users to create rich musical soundscapes, and highly complex pieces in almost any musical style – not just dance, but electronica, rock, jazz, blues, and even orchestral. The quality of the music improved, if not the sound quality – and user-defined samples offered significantly more sonic creativity and control than General MIDI sound sets of the time.

Ultimately a commercial failure, tracker programs became the care of these expert users, who continued development of the technology, as part of a subculture known as the demoscene [15]. This community of coders, artists and musicians began with young hackers, who would reverse engineer and “crack” games to remove copy protection or change playing conditions. To flaunt their coding ability, hackers teamed together in crews, adding splash screens with credits, greetings or messages to friends or rival crews (see Figure 4). Over time, these intros became a prominent showcase for coding talent, exhibiting increasingly complex visual art, animations, and music, ultimately eclipsing and displacing the original hacking activities.

Practitioners and crews met at demo parties, partly to socialise, play games, and swap coding tips, but mostly to exhibit and compete against each other with their latest audio-visual demos and music. Works were judged not just on their artistic quality, but by the technical virtuosity demonstrated by authors. Small gatherings that began in hackers’ basements have since grown to prestigious events, and tightly invigilated competitions, attended by thousands (see Figure 5).

2 According to Raymond [11]; hackers, in the era when Emacs was developed, coined the term “hack-mode” to describe “a Zen-like state of total focus” – a description closely mirroring observations made by flow research in other fields, such as music and sports [4].

3 For example, the popular Commodore Amiga, the original platform of choice for the tracker musician, was a 16-bit computer, with 4 channel, 8-bit, 20kHz stereo sound, typically booting programs from a 720KB floppy disk into 512kB of RAM.

4 Nonetheless, the programs represent one of the first examples of low-latency software synthesis, foreshadowing the digital desktop studio.
Between parties and before the advent of the World Wide Web, crews would exchange, publish, share and review programs, demos and music over bulletin board systems (BBS), and even via floppy-disk compilations and newsletters, called disk mags, mailed by post. The tiny file sizes of tracker music (often around 100kB) enabled their transfer over slow modem connections, yet still contained all the music and sample data necessary for playback on almost any computer with a soundcard, without the need for specialist MIDI or audio hardware. Moreover, music was shared in a completely open format that allowed any listener to load, edit, change the music or re-use the samples in their own work. This opportunity represented a valuable learning resource for the novice, who could develop knowledge of the program by observing its use by others, and tinkering with the music to learn the workings of the notation.

By the late 90’s, the scene had moved to the IBM PC and DOS, and new programs and formats began to support up to 64 polyphonic channels of CD quality audio. However, at the same time, the original, increasingly obsolete MOD format survived. Although modern DSP, resampling, and upsampling marginally improved the playback quality, the format was retained specifically for its limitations and its capacity to test the ingenuity of composers, whose endeavours to defy listeners’ expectations and garner the respect of their peers continued.

Similar trends are found elsewhere in the demoscene [15]. DirectX and video hardware provided similar leaps in graphics capabilities, in demos, but competition categories formed, with explicit limits on the size of demos, affording entrees no more than a 4kB (or 64kB) footprint5 for their executable – which had to contain the entire code and content for the presentation of all visual and audio content. Exploiting complex procedural generation algorithms to mathematically create intricate textures, shapes, visual effects, sounds and music, winning entrees nonetheless deliver intricate, high quality, high resolution audio-visual spectacles, often several minutes in duration, comparable to other productions measured in megabytes or gigabytes.

In the 20 years of tracking history, though hardware capabilities and user interface design have moved on, the basic design of tracker software and interfaces has changed little - in appearance, function, or use. Little effort has been made to make the programs easier-to-use, and obsolete file formats and limitations from DOS and Amiga programs are still supported and celebrated. Instead, the appeal of these programs and limitations are that they specifically provide a challenge to the user, that mastering them is rewarded both intrinsically and extrinsically, in the user experience and community, respectively. Consequently, the designers of new musical interfaces need to consider more than the simple sonic capabilities or usability factors of their offerings.

3.3. Comparisons with sequencing

The modern sequencer, or digital audio workstation (DAW), offers a direct “recording” input method, where audio signals or MIDI messages from an external instrument are captured in realtime, after a record button is pressed. Using this tape recorder metaphor and playing a physical instrument or device allows musicians to apply the virtuosity and access the flow experiences observed in traditional virtuoso musicianship (e.g. [5]). Once recorded, any further edits or revisions are affected by pointer-based manipulation on the computer – to an extent that, even with MIDI note data, the tendency is often to erase, rewind and re-record instead.

The sequencer equips the desktop computer with recording capabilities formerly found in the electronic studio – with many different windows and tools to define the structure of a piece of music before it is rendered to audio. These representations typically include, but are not limited to: an arrange window, mixer, score, piano roll, data list, software synthesizers, audio editor and resource pools, some of which are pictured in Figure 6.

Figure 6. Apple Logic, under Mac OS X.

Most of these notations are presented via onscreen visual metaphors – analogies to methods and devices that pre-date digital music, from the piano keyboard and score, down to literal depictions of mixing desk potentiometers, faders and even dangling wires [6]. However, when representations are so literally based on physical apparatus, a gulf opens up between the user’s concept of music and what is easily encapsulated in the notation. Leman argues that such layers of abstraction lead to “indirect involvement” in the music process [8]. The overly metaphorical correspondence to physical music equipment also means that interacting through generic devices, like the mouse, become cumbersome.

In earlier work [1], we explored the difference between direct manipulation and programming-like approaches in computer music, represented by sequencer and tracker programs respectively. In these contexts, sequencers favour continuous representations of musical elements,
using visual metaphors to physical devices in the electronic studio designed for live use; whilst trackers focus on the editing of a notation, describing a script of future events, more like the source code of a program, or even the musical score.

We found that, outside of realtime recording, the feedback from sequencer favours visual, rather than audio, modalities, leading to reduced “liveness” [14], a measure of the availability of access to the end product (i.e. music). Though trackers could not support the level of liveness inherent in a ‘live’ musical performance, fast editing and audio feedback represented an improvement over the mouse-driven interaction that follows episodes of recording in the sequencer. These issues concerning feedback are further explored using the study described in the next section, in the context of virtuosity and flow.

4. ANALYSING THE USER EXPERIENCE

To explore computer music user experiences, we created iMPULS (Internet music program user logging system), a system to record data on user interaction in a given music program, running in the user’s native music environment, and deliver the data back to our lab6. The system was deployed with an established tracker, reViSiT [12] (see Figure 7), which runs as a VSTi plugin to DAW software7, allowing us to observe user interaction in both the tracker plugin and host sequencer. Users registered online, filling in an initial questionnaire detailing their musical and technical background, and are emailed an activation code to unlock the program. Using this code, we are able to identify individual users and their logs, and observe their development over time.

The system records a variety of data during interaction, allowing us to look at the use of input devices such as the keyboard and mouse, as well as windows and specific uses of the program such as pitch usage and playback control. Each event in the log is time-stamped to allow fine timing and performance analysis of interaction. To protect privacy, a user’s music data is not collected, but the system extracts basic statistics about song content, such as the total number of instruments used, average density and coverage of patterns, usage frequency of specific pitches or effects.

2351 participants signed up for the experiment, launched in December 2008. Of those, we have successfully received logs from 1125, yielding 13,142 logs to study – containing over 5800 hours of interaction. Participation varies from brief experiments with the software, to prolonged, daily use.

To analyse the logs, iMPULSIVE (iMPULS Interactive Visualisation Environment) was developed to download, manage, filter, process and visualise the experiment and its data, and export it to formats for use by other programs (e.g. Excel, R, SPSS, MATLAB). The program is pictured and briefly described in Figure 8.

The initial questionnaire surveyed participants’ attitudes to various interaction modes and styles, establishing a baseline of their existing experience of, and expertise with, various music technologies. Analyses were then able to correlate their responses with their interaction logs, looking for trends and phenomena that emerged as a result of experience, or might indicate periods of flow.

A second online questionnaire was issued at the end of the 2-year study, to gauge their impression of tracking and the reViSiT software, as well as any changes in their attitudes to computer music interaction or levels of expertise. The survey also probed attitudes to the user experiences of both reViSiT and one other sequencer of the user’s choice (e.g. their host program), based on established psychometric tests of the nine components of

6 See http://experiment.nashnet.co.uk for the study’s online presence.
7 See http://revisit.nashnet.co.uk for details of the reViSiT software.
flow, extended to incorporate various dimensions in the Cognitive Dimensions of Notations framework [7] (an approach for evaluating interfaces and notations), in an effort to correlate interface characteristics with support for flow, while also allowing relative comparisons between trackers and sequencers.

Our analysis of both the questionnaire and interaction data continues, but some interesting findings have already been made, and are discussed in the next section.

5. INITIAL FINDINGS & DISCUSSION

Keyboard interaction lies at the core of the tracker user experience. Thus, one of our major focuses was to explore how keyboard use might facilitate flow—looking at the intervals between key presses, as well as the distribution and sequences of key combinations used. Figure 9 shows the distribution of keystroke intervals, for four groups with differing levels of prior tracker experience. As often seen in studies of action intervals in user interaction, the distribution for each group is in the general form of a power law distribution.

For the least experienced group, our data is too sparse to make a reliable estimate of the distribution. These 161 newcomers to tracking (20% of the total participants) provided only around 10,000 key presses (1% of the total). Thus, we focus on the three groups with greater experience, which we expect to show evidence for increasing levels of flow and virtuosity.

The height of the peaks and proportion of lower keystroke intervals (more rapid keystrokes) increase with experience, matching the usual expectation of faster interaction with practice. However, the mode of the inter-key interval distribution becomes slower (the peak moves from 140ms to 155ms) in the top group. Furthermore, the top two groups also show an increase, relative to a simple power law, in the slower range of 200ms to 500ms. More skilled users appear often to work fractionally slower, rather than at the higher speeds that a power law of practice [10] alone would predict.

To shed light on this surprising pattern, we recorded video footage of a professional composer (see Figure 8), who had been using the reVIsIT software for several years. On viewing 5 hours of footage from a single working session, he was surprised not only by the length of time he had been working, but to see how “obsessed” he was, comparing his typing to “speedcubing” (competitive Rubix™ Cube solving). Despite a rapid work-rate, he had felt unhurried, commenting:

*I’m never conscious of those kazillions of keyclicks [...] It’s also as if it’s very long/boring. I was almost afraid that this vid’ showed tracking is *not* fast, but alas, when in the first 18 minutes I have a full orchestra/bigband… I guess it’s still radically fast.*
His distorted perception of time and lack of self-consciousness, confirmed by our analysis of the video, indicate that he was able to maintain the flow state for almost the entire duration – breaking only for meals or to audition his music at length. Moreover, rather than isolated bursts of top speed interaction; a steady yet still rapid rhythm was evident – contributing to a calmer and more controlled interaction experience.

To pace interaction in this way, mastery of the tool requires a user to plan actions several steps ahead. This requires a program where actions and feedback are not just direct and immediate, but predictable and learnable.

This aspect of virtuosity and the fluidity of the user experience is defeated by dynamic, moveable screen layouts and the constant visual attention demanded by the mouse pointer. Within the multiple-window context of the DAW, users spend significant time managing the arrangement of windows, rather than interacting with the music itself. Window management accounts for a quarter (24.8%±SE•0.08) of all keyboard and mouse input in the programs we studied, but only 3% (±SE•0.06) in the tracker plugin. Furthermore, although we saw some increase in the speed of DAW operation among experienced users (in part due to a shift from mouse to keyboard shortcuts), the overhead for window management remains, providing a frequent distraction. Since windows are always moving and being hidden, it represents a valuable target for design improvement.

These findings mirror those in a recent video study of a composer at his computer [2], which observed similar inefficiencies in the sequencer interface, where the need for frequent “housekeeping” like the windowing tasks we noted earlier, got in the way of writing music.

In a music program, timely audio feedback is perhaps even more critical than visual. Analysing keystroke categories showed that users with different amounts of tracker experience differ in their use of playback. Novices tend to audition their music from the beginning of the piece or phrase, while more experienced users audition shorter passages at or around the editing cursor, developing a rapid edit-audition cycle, with edit commands interwoven between playback of a single tracker row, the beat (4 rows) or the bar (16 rows).

Analysing the lengths of playback episodes, it is evident that trackers support a tighter edit-audition cycle. Whilst greater experience led to fast feedback cycles in both programs, the overall distribution of playback episode lengths shows how sequencer playback is heavily quantised to musical bars, as illustrated by the spikes, in Figure 10. The mode of tracker episodes is a duration of 0.5s (1 beat at 120bpm). By comparison, sequencers show a tendency towards whole bars and longer phrases (2s, 4s, and 8s; or 1, 2 and 4 bars at 120bpm, 4/4).

This disparity is explained by the provision of cursors in the UI, which indicate where the focus of work is. Both programs maintain a playback cursor, but while the keyboard focus of tracker programs provides a constant edit focus, no fixed focus or context is supported by the sequencer’s mouse pointer, which can affect any moment or property in the music, across any window.

Auditioning recent sequencer edits thus requires a more involved reconfiguration of the playback cursor, which is then typically snapped to the nearest bar or beat.

The overall tendency for experts to focus on finer details and shorter sections would initially seem to contradict other findings that observe expert composers’ greater ability to focus on the “big picture” [2]. However, our finding can be interpreted as a reiteration of their improved ability to retain the big picture in mind, without recourse to auditioning longer musical passages.

Collins [2] saw a contradiction between his subject’s desire to work fast (“achieving as much as possible”) and frequent “pre-occupation with small-scale actions”. From a perspective of flow, there is no contradiction; while the composer’s net productivity may drop in these moments, their energy, concentration, and interaction rate remain high, correlating with our earlier theories that fast feedback cycles, afforded by simpler, more iterative edits, are more likely to lead to enjoyable flow experiences and perceptions of greater ‘liveness’ [1].

The resultant barrage of sound in these editing sessions seems discordant to observers, as the disjointed playback jumps randomly and fleetingly between short excerpts of the music. However, our video subject remains unfazed, maintaining concentration and focus on the interaction, thus exhibiting a further indication of flow, the loss of self-consciousness.

Action-awareness merging is also evident. To an extent, this is already evident in the trance-like interaction shown in the video, but it is also suggested by anomalies in the log data of other users. In Figure 11, we see that keyboard interaction rates are influenced by the tempo of the music being worked on, even when not playing. During cursor navigation, we see peaks in the distribution of intervals between keystrokes not only at default key-repeat and rapid manual input rates, but also a further peak at 0.5s, corresponding to the default musical tempo of 120 beats per minute. As expected, more experienced users have a tendency to use a wider...
variety of tempos, which are similarly reflected in the data. The habit of listeners taping the beat of music has been widely studied, but here this latent tendency is merging with program interaction.

![Figure 11. Intervals between cursoring (no playback).](image)

Early analysis of responses from the second, end-of-experiment questionnaire correlated with our earlier findings. The psychometric tests of flow components indicate that expertise and experience have a significant, positive effect on the conditions required for flow, in trackers and sequencers ($p < .05$; Figure 12). Moreover, the metrics also indicate that, among more experienced users of each program, the tracker architecture appears significantly more conducive to flow experiences, compared to sequencers and DAWs ($p < .05$).

![Figure 12. Flow metric vs. user experience.](image)

6. CONCLUSIONS

When considered as a visual notation, it is clear that although trackers lack the appealing visual metaphor and surface ‘musician-friendliness’ of many sequencers or DAWs, their use of a concise, flexible and learnable visual formalism coupled with the ready-to-hand availability of the end product (here, sound) supports an engaging user experience, that supports flow, virtuosity, and a see/hear-understand learning cycle [8]. In the scenarios studied here, flow ultimately results from higher-level mastery of the musical task, built on rapid feedback cycles and low-level motor learning of manual (e.g. keyboard) skills. Some theoretical aspects of this work were discussed in [1], which we plan to expand upon in future publications, towards a modelling framework for music systems that integrates the factors of virtuosity, feedback, flow and liveness discussed here.

7. ACKNOWLEDGEMENTS

Our research was part-funded by the Harold Hyam Wingate Foundation. We’d also like to express our sincere gratitude to the participants in our study and the wider reViSiT and tracking user communities whose openness and support made our research possible. Special thanks are also due to composer, Maarten van Strien, for providing a window into his studio; and to the guys at Yamaha, Steinberg, and Cakewalk, for their input, support and assistance during the experiment.

8. REFERENCES