Research Plan

Belinda Thom

February 3, 2006

Contents

1 Research Philosophy **1**

2 Overall Research Agenda **2**

3 Research Done Prior to Mudd **3**

4 Current Research **4**
4.1 Database Format 5
4.2 Data Consolidation 5
4.2.1 Optical Music Recognition 6
4.2.2 MIDI Transcription 6
4.2.3 Rhythm Quantization 6
4.3 Audio Transcription 6
4.3.1 Evaluation 8
4.4 Sequence Learning Algorithms 8
4.5 Feature Manipulation 9

5 Research Collaborators **10**
5.1 Ami Radunskaya and Cymra Haskell (Summer 2005-present) . 10
5.2 Bob Keller (Summer 2005-present) 11
5.3 Christopher Raphael (Summer 2005-present) 11

6 Research Students **11**
6.1 Katy Perdue (Summer 2003) 12
6.2 Aja Hammerly (Spring 2004) 12
6.3 Mark Nelson (Summer and Fall, 2003, and Spring, 2004) 13
6.4 Brian Young and Chris Erickson (Summer, 2004) 13
6.5 Chris Erickson (Fall, 2004) 13
6.6 Stephan Jones and Aaron Wolin (Summer 2005) 13
6.7 CS 197 (Fall 2005) 14
6.8 Michael Beyer (3 units, Fall 2005) 14
6.9 John McCullough (3 units, Spring 2006) 14
6.10 National Science Foundation REU (Summer, 2006) 14

1 Research Philosophy

For me, research is inherently hands-on—you learn how to do research by doing it. Because of my subject area, research also comes naturally to me: as a musician, I have a genuine and passionate curiosity about how musical processes might be modeled on the computer. As displayed in Figure 1, the core motivation of my research is physical (hence the spiral): improvising on the violin leads me to reflect on that experience, which results in constructing abstractions that might be used to model aspects of this process, and so on. Computation is an ideal medium because it facilitates transforming vague hunches into viable procedures, producing software that can be exploited to understand musical processes and spark creative new ideas about how to model them.

There are many reasons Harvey Mudd College is an ideal home for my research: technical competence, the humanities emphasis, the institution's value of interdisciplinary scholarship, its amazing undergraduate population, etc. In addition, since my research can be achieved with consistent, incremental effort and doesn't require large amounts of equipment, with patience,[1] HMC's teaching focus provides an ideal environment for me.

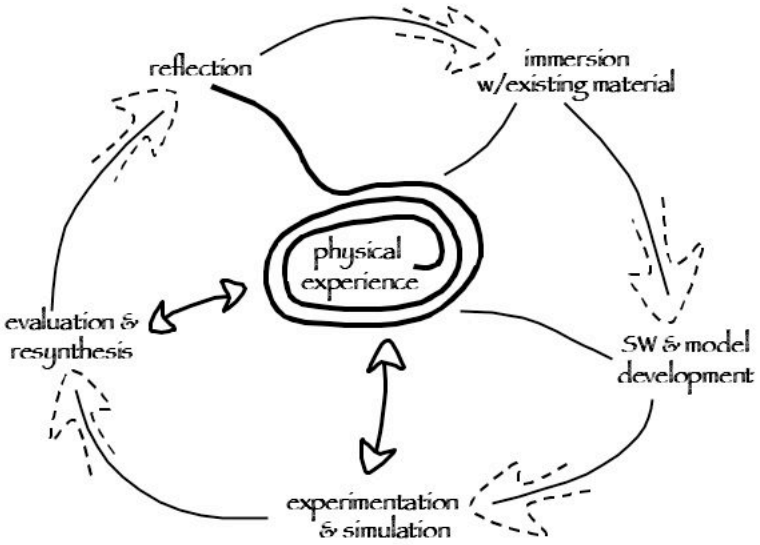
My research does, of course, require some equipment. I have been using my startup funds to build a *Musical Intelligence, Learning, and Optimization (MILO)* research lab. This lab currently resides in the 2nd floor of Sprague Library and houses three computers (2 dual boot PC/Linux boxes and a Mac), a digital keyboard, high-end microphones and mixer, and MIDI[2] recording equipment. In the next year, I will complete this lab's configuration. Additional purchases will include state-of-the-art commercial software (serving both my research needs and as benchmarks) and some MIDI controllers (e.g., a Zeta MIDI violin).

Undergraduate research (Section 6.8) forms an integral part of my current research plan (Section 4). Many Mudders are passionate musicians, and my research topics have generated interest among students. As outlined in my current research section, I have many well-defined, open-ended tasks which can serve as gateways to genuine research experiences in computer music and artificial intelligence. For instance, for many of these tasks, solutions already exist, so an ideal entry point for a student is to learn about and implement an existing technique. A natural next step is to evaluate how well a given technique performs in the improvisational music setting, which is the focus of my research. Invariably, deficiencies will be unearthed, providing an opportunity to improve existing techniques. Even if a student's effort to improve an existing technique doesn't pan out, we have still gained a great deal of insight about using this technique in improvisational settings. In this scenario, such insights often lead to publishable results.

[1] Most progress occurs in the summertime.

[2] MIDI stands for the *Musical Instrument Digital Interface*.

Figure 1: Belinda's Iterative Research Process



2 Overall Research Agenda

My research focuses on teaching the computer how to perceive music, create music, and use these skills to interact with live musicians. The goal is to realize enriched, spontaneous exchanges between improvising musicians and computer agents. I call these agents *Improvisational Music Companions (IMCs)* to remind us of their ultimate goal: becoming first-class musical citizens, companions that we *want* to interact with, as opposed to computer tools we endure because they're what is currently available. This research adopts machine learning technologies because they provide the potential for automatically configuring computational models to explain musical phenomena using data.

Learning algorithms are only as good as the data they receive—as they say: garbage in, garbage out—so a critical aspect of employing machine learning successfully involves finding adequate representations with which to encode musical phenomena in the first place—be they harmonic, melodic, rhythmic, or what have you. Here, computer simulation can be really helpful, for it allows systematic exploration of models trained using different encodings of the same musical dataset. This search is not a *free lunch*, though: automating the search for better representations requires metrics for distinguishing which results are better, yet such metrics likely need good musical representations![3]

[3] A similar search can also be used to explore what types of models are most appropriate. I avoid making this distinction because there is no hard-and-fast line between where represen-

This snarl of interdependence is one reason why Figure 1 contains the arrows it does. Consider, for example, the cycle implied by the dashed arrows. This cycle captures the iterative and incremental approach I take: build a model, play with it, reflect on its deficiencies and merits, and then further tweak it. The hope is that sufficiently reasonable metrics can be developed so that we're able to make some progress in the search for good representations. We then continue the search, honing in on progressively better representations. Ultimately, of course, defining what makes for a good representation *must* be grounded in experience—e.g., is the IMC that results fun to interact with? Thus, evaluation necessarily involves the physical, which the bold double-headed arrows highlight.

It is true that tying evaluation to human impression makes evaluation more difficult. At the same time, by intimately exploring specific, simpler aspects of how humans perceive music,[4] improved understanding about how humans evaluate music more generally will follow.

3 Research Done Prior to Mudd

In my PhD thesis, I developed an improvisational music companion named *BoB (Band-OUT-of-a-Box)* that used machine learning techniques to configure itself to a specific musician's improvisations (Tho03). Once trained, the resulting model was able to perceive short snippets of the musician's transcribed solo—making somewhat musically meaningful distinctions about this snippet's tonal, intervallic, and contour content. With these distinctions, the agent could then respond in kind to what the musician had played.

One of the most impressive aspects of this model was how little data it needed. For instance, the model could be configured with about 100 bars of a solo. In musical domains, this capability is so important, because many aspects of musical structure are highly localized, e.g., a particular harmony is more likely to appear consistent within a given song than across a set of songs (Tho95).

Another impressive aspect was that the learned model behaved reasonably, even though it received no explicit harmonic knowledge. Yet if harmonic knowledge were included, the model could have avoided some of the poor musical choices it did make. Typically, such knowledge is added by hand-coding it into the representation in some way, which necessarily makes the system less flexible. An important aspect of my current research involves figuring out how to incorporate enough of this knowledge into the model without rendering it incapable of adapting appropriately across a variety of improvised environments.

My PhD system was flexible in that it didn't hard-code in specific harmonic or melodic constructs, but that doesn't mean that no musical knowledge was hard-coded. For instance, a musician's solo was transcribed by a human before it was passed into the system. This *transcription process*, whereby a human

tation ends and the model begins.

[4] As outlined in Section 4.3.1.

breaks up an audio signal into a discrete sequence of note events (like the ones you would find in a musical score), implicitly encodes a huge amount of musical sensibility. In addition, this limitation meant that I could not interact with my IMC live! All evaluation of the system's performance was restricted to simulations using pre-transcribed musical input.

Another aspect that was hard-coded into BoB was that transcribed solos were transformed into individual data points on a per-bar basis. While this choice was arbitrary—the model would have accepted melodic snippets of arbitrary length—it presents an example of how a representation can make or break a machine learning algorithm's ability to learn. Suppose the underlying solos were not generated by a process that was bar-aligned. Then, constructing features on a per-bar basis might mix together the very distinctions that the learning algorithm would like to disambiguate.

For this reason, I spent my Fulbright year in collaboration with Karin Höthker and Christian Spevak, exploring algorithms that automatically segment melodies in reasonable ways. Our work shed light on important issues that arise when predicting segmentation boundaries in music using current techniques (STH02). Not surprisingly, our work revealed the difficulty of achieving an ideal solution. Again, there was no free lunch: melodic segmentation becomes easier with the right features, yet these often require better segmentations.

4 Current Research

My current research—constructing viable improvisational music companions (IMCs)—builds on prior work, but I've regrouped my effort somewhat. **My current plan is to construct a melodic improvisation database and to use it systematically to explore what representations and modeling schemes are truly best for IMCs.** I recognize the entirety of this plan, which I outline below, is ambitious and will take years to complete. Fortunately, my students and I will always be able to make incremental, steady progress. When specific machine learning experiments warrant it, we'll enter improvised solos manually, by hand. But usually, our efforts will focus on constructing intelligent music perception algorithms, so that less manual data entry is required. All the while, in the background, machine learning simulations will progress.

Although plenty of improvisations exist, there is no comprehensive repository that catalogs their harmonic and melodic content in a *symbolic*, digitally accessible way.[5] Thus, constructing a reasonably diverse database comprised of improvised solos, along with related context like a tune's head, harmonic progression, and metric information, will be an important contribution. Hosting this database at Mudd will provide increased institutional visibility.

As the database grows, we will begin exploring how to model the data, via systematic machine learning experimentation. In preparation for this en-

[5] With composed music, significant databases do exist and have received some attention in the machine learning community—e.g., the Essen folk data base (Sch95), the chorales of J.S. Bach (DNM98).

deavor, I attended a week-long machine learning summer school held by the University of Chicago in 2005. There, I learned about recent advances in the field and networked with people who have been studying algorithms that could be applied to music.

4.1 Database Format

There are so many possibilities for modeling improvised data—do we look at monophonic or polyphonic solos? do we focus on melody or harmony? do we focus on individual solos, or groups of solos?—that some decisions regarding what information to include must be made at the outset. I'll begin by carrying over *BoB*'s requirement that solos be available in a monophonic, transcribed form. Monophonicity means that the solo was played by a single instrument playing a single melodic line (no chords are present). Transcribing this solo maps it into a musically reasonable sequence of non-overlapping, discrete pitch/duration pairs, which I call a *string of notes*. When harmony is available, it will be similarly encoded as a *string of chords*—i.e., a non-overlapping sequence of chord symbols. The exact syntax I'll use to store improvisations in the database has yet to be determined, but *Guido*'s generality and music-typesetting support make it a likely option (HHRK98).

A compelling justification for storing an improvisation as transcribed note and chord sequences is that this format preserves the information typically found in lead sheets that improvisors use all the time. For live interaction, though, audio transcription is required, which is why I began exploring automated transcription in earnest when I came to Mudd.

4.2 Data Consolidation

A large body of improvised musical data already exists, e.g., audio and MIDI recordings, published transcriptions. A large part of the REU effort this summer (Section 6.10) will involve investigating the feasibility of importing some of this data automatically.

4.2.1 Optical Music Recognition

Since transcription is a thorny issue, one good place to start is with improvisations that have already been transcribed. Ideally, the importation process can be (at least somewhat) automated using off-the-shelf optical music recognition software.

4.2.2 MIDI Transcription

MIDI files might also be an attractive data source option, but it is important to remember that this representation does not necessarily cleanly map into strings of notes and/or chords. If a solo is recorded on its own track, it becomes much more feasible to transcribe it. How difficult this process is depends, in part, on how mechanical the performance was. For instance, if

a MIDI synthesizer performed a score verbatim, the process would be trivial. With human performance, other issues come into play. For instance, even for monophonic melodies played on a digital keyboard, adjacent notes may overlap by milliseconds.

4.2.3 Rhythm Quantization

Provided a MIDI file's pitch and note onset/offset times accurately reflect the actual performance, transcribing MIDI should be easier than dealing with raw audio. The main issue is *rhythm quantization*; i.e., aligning event times to a musically reasonable grid. When tempo is known, the issue simplifies into assigning note durations to appropriate divisions of the metrical beat. The difficulty of this task is also part of why music is beautiful: musicians perform expressively. For instance, I often swing my eighth notes, playing the first much shorter than the second. Benoit Meudic's *casual beat-tracking algorithm* (Meu02) might prove useful when quantizing MIDI. This algorithm also runs in real-time.

Related Undergraduate Research Mark Nelson (Section 6.3) and the upcoming REU (Section 6.10).

4.3 Audio Transcription

Transforming audio into a musically appropriate score becomes easier when scenarios are limited (Ger03). Thus, I am currently only considering specific instruments and playing styles, such as the types I might encounter when jamming with friends in my living room. Instruments include violin, cello, guitar, and flute, and playing styles include bluesy and folksy improvised contexts. It is also important that recordings need not be of professional quality, for I want to be able to use the transcriber in casual settings.

Currently, I'm only considering offline blind-source monophonic audio transcription. Monophonicity should result in a non-overlapping sequence of pitch events, quantized to semi-tone resolution. We'd also like to quantize *where* these pitch events occur in time, a rhythm quantization task.

Blind-source refers to the fact that we don't know what score the musician is playing from.[6] Blind-source transcription would be extremely useful for database construction, because it could convert monophonic recordings into transcribed form. This task could be performed offline—i.e., not in real-time, which simplifies coding considerably. Offline transcription also tends to be more robust, because the entire audio signal can be considered before committing to a solution. Additionally, provided errors are not too plentiful, we can accept less-than-perfect performance in offline settings, because transcriptions can be cleaned-up by hand, if needed. In the long run, however,

[6] Some might argue that improvised music, having been composed on-the-fly, has no score. Regardless, when learning to improvise, musicians frequently transcribe solos from recordings (KJTW06).

I'll need to tackle online transcription, since this capability is required for live interaction with an IMC. My hope is that as I learn with the simpler offline case, I will gain strategies for solving the more difficult online case.

A fundamental task is tracking pitch in the signal. Pitch is a perceptual phenomena, although the algorithms that do pitch tracking typically attempt to estimate the related fundamental frequency (a physical phenomena). Although much research has been performed, pitch tracking can still fail to perform well when a single clean-pitched signal is not provided (Ger03). A intimately related task is detecting onset boundaries, which marks the beginning of notes. One might expect this task to be easier—at least you don't need to classify a note's pitch—but similar performance issues arise here (BDA+04). Moreover, once onsets are determined, rhythm must still be quantized.

Pitch tracking, onset detection, and rhythm quantization are all hard problems in their own right, but when you combine them, things can become surprisingly simpler. For instance, suppose I told you where the onsets and durations of all the notes were. Pitch tracking becomes easier, because one label explains an entire region. With this knowledge, troublesome issues like octave error can virtually disappear. Christopher Raphael's (Section 5.3) graphical modeling techniques for *aligning* audio to a score that is *known in advance* capitalizes on this insight by constructing models that essentially attempt to segment the audio into the number of pitch-regions implied by the score. Simultaneously, his model encourages data to be segmented in such a way that, within a given region, pitch-relevant features look as uniform as possible.

While it is not entirely clear how to extend this approach to the fully-blind case, Christopher and I have begun collaborating on ideas. I will ramp up to this harder problem by first extending his techniques to apply to *partial scores*, where only the score's pitches (as opposed to idealized onset and duration times) will be specified up front. Audio alignment in this partial-score case will be extremely valuable because, for example, it is much easier to figure out what pitches I've played in a given lick than it is to determine where in the audio file those notes start and stop.

Related Undergraduate Research: Katy Perdue (Section 6.1) and John McCullough (Section 6.9). Matt Walsh also built a pitch tracker in CS 182-1,[7] collaborating with Joe Walker (whom Bill Alves and I jointly advised). A group that I advised in E84 in the Spring of 2003 also looked at this problem.

4.3.1 Evaluation

As my students started building audio transcription systems, it became clear that evaluation was a huge issue. Ideally, you'd like to be able to say "this transcriber missed five notes, so its worse than another one that only missed three," but this view is extremely naive. For example, when is a note actually missing? When it is detected 15 milliseconds after the fact? And who determines where this "fact" occurs in the first place? And what if it's only

[7] CS 182-1 is a machine learning seminar I taught in Spring 2005.

13 milliseconds late? Or 19 milliseconds late? Ultimately, human perception *must* be considered. This fact is also receiving attention from others in the field (LG04).

Evaluation becomes even more interesting when you consider that music can be *ambiguous*. For instance, when I play a quick turn on the fiddle, record it, and then play it back, I can't always decide exactly how many notes there are, let alone at what exact times in the playback they occurred, even with careful and disciplined repetitive listening. In a collaboration between Ami Radunskaya, Cymra Haskell, and myself (Section 5.1), we explore this ambiguity. This effort has involved analyzing the same lick, played and recorded multiple times, as well as hand-labeling the same audio recording multiple times. We hope to use this data to construct mathematical models that can be used to assign perceptual fitness values to transcriptions. Ideally, these models would be probabilistic, allowing only those parsings of the signal that are musical reasonable to be assigned high likelihood. To build reliable models, a lot of data must be collected, so extending Raphael's work to handle partial scores will be of great use. There is also a lot of overlap between this work and the work I did while on my Fulbright (STH02).

It is important to point out that our focus on evaluation is folksy, bluesy licks played on instruments like the violin, so these results won't necessarily apply to other settings. However, by understanding perceptual evaluation in this specific situation well, we hope to learn more about human-driven evaluation in general. These metrics will also allow us to quantify how well state-of-the-art technologies like Fiddle (PAZ98) work for our purposes.

4.4 Sequence Learning Algorithms

*Sequential learning algorithms (SLA)*s typically operate on discrete data sequences, for instance, predicting the next character in a DNA sequence or the next word in piece of text, given some prior sequence of characters or words. Most SLAs are probabilistic, which means that we can use them to generate novel sequences that are presumably similar to those sequences that were used to train the model. SLAs can also estimate how likely a sequence appears, a functionality very akin to perception. Given their perceptive and generative capabilities, it is not surprising that SLAs have been applied to music, although mainly in classic contexts (CW95; WP03). As far as I know, Ron Begleiter and Shlomo Dubnov are the only researchers who have applied SLAs to improvised data.

State-of-the-art SLAs tend to be fairly involved, although Ron Begleiter has recently made gaining experience with them easier (BEYY04). That paper serves as both a tutorial and an empirical investigation. The author concludes by comparing various algorithms' performance on textual, musical, and biological data. Particularly useful for my research, the author provides implementations of the SLAs he evaluates, as well as the textual, musical, and biological data he used.

Related Undergraduate Research Michael Beyer (Section 6.8) and Chris Erickson (Section 6.5).

4.5 Feature Manipulation

Melodies naturally lend themselves to different sequential viewpoints (CCI99; CW95). For instance, encoding a melody as a sequence of absolute pitches anchors it to its underlying tonality (which might change over time), whereas an intervallic encoding reflects the distance between adjacent notes (and thus is transposition invariant). Or compare a pitch sequence with a pitch and onset/duration sequence. The latter includes rhythmic information, whereas the former ignores it. Part of what makes a melody work is the interplay between different viewpoints. Ideally, all of these viewpoints would be used, so that as many melodic relations as possible are retained. Unfortunately, enlarging the size of the feature space in this way hastens the curse of dimensionality.

Nevertheless, we would expect an SLA to perform better when its sequences are constructed from an alphabet that contains the most salient information that contributes to its success. For example, consider having to predict the next note given some previous set of notes. This task might become much easier if each note's pitch was first transformed into a new symbol that accounted for its harmonic function. Unfortunately, not much effort has been spent *systematically* comparing different ways to represent and integrate melodic and harmonic content.

Systematic exploration of appropriate features would be *much* easier if we had tools for easily transforming the database's raw data. One idea that I am particularly excited about is extending Impro-Visor (Section 5.2) so that it can be used to rewrite melodic content in new, interesting ways. Because of its flexibility, *Impro-Visor* is an excellent platform for exploring theoretical relationships between harmony and melody. To change the system's notions about musical relationships, one has only to construct a new rule base, which can be imported via a template file.

With minor modifications to *Impro-Visor*, we could essentially reverse its usage. Instead of advising students about what melodic fragments they might use in their solos in particular contexts, the system could analyze existing improvisations according to which knowledge in its template is operating at particular locations. For instance, with this capability, it would be trivial to recode an improvisation in terms of its chord or approach tones. This capability is quite powerful when viewed in terms of my overall research agenda: once sequences have been rewritten, we can then use them to train new SLAs. Properties of these trained models can then be evaluated for their fitness. For example, if our goal were to build a model that captured some melody's inherent internal coherence, entropy would provide a reasonable evaluation metric.

Related Undergraduate Research Aja Hammerly (Section 6.2), Katy Perdue (Section 6.1), and Stephen Jones and Aaron Wolin (Section 6.6).

5 Research Collaborators

The following table summarizes my research collaborators since I've come to Harvey Mudd College. Sections detailing these entries follow this table.

Table 1: Summary of Collaborators in Research.

Collaborator	**Discipline**	**Institution**
Ami Radunskaya	Mathematics	Pomona College
Cymra Haskell	Mathematics	USC
Bob Keller	Computer Science	HMC
Chris Raphael	Computer Science	University of Indiana

o

5.1 Ami Radunskaya and Cymra Haskell (Summer 2005-present)

Ami, Cymra, and I have had a long association playing music together, often in improvised contexts, so this collaboration was a natural one. Ami has been improvising with the computer for years, but typically used fairly coarse signals (e.g., number of notes recently played) when coupling her own performance to the computer. Cymra is active in the local old-time music scene, where expressive performance, simple embellishments, and improvisations to folk melodies are a mainstay. We began thinking about how cool it would be to improvise with each other, using the computer to mediate this experience. For instance, imagine the computer could construct a database of the representative riffs we play while improvising together on our respective instruments (cello, flute, violin). Ideas like these require audio transcription, which motivated the work described in Section 4.3.1.

5.2 Bob Keller (Summer 2005-present)

In the Fall of 2004, Bob Keller approached me about applying for a *Mellon Grant* to create *Impro-Visor*, a jazz education software package that would aid students as they learn to compose solos. Bob asked me to collaborate with him on this grant, in part, because he thought machine learning might enable some aspects of the tool to be automatically configured. Fortunately, the grant we put together was funded.

Initial work has revolved around demonstrating the feasibility of the idea, so machine learning has taken a back seat. In this phase, my role has involved more advising than development, and my main contribution has been to serve as a bridge between Bob's software development and jazz knowledge expertise and current state-of-the-art computer music practice.

The current tool provides a GUI for displaying and editing basic lead-sheet information, as well as an advisor—the *Visor* part—that suggests good note and/or lick choices in specific locations. Because this software derives its advice from a knowledge-base template file, different theoretical frameworks can be easily explored. Currently, this file is hand-coded by a jazz expert (Bob Keller) and considers things like chord tones, relevant scales, motivic cells, and the like. Later, we will incorporate machine learning to configure some aspects of this file.

Impro-Visor is available for download and users have been enthusiastic. This system is detailed in a technical report. We are also exploring what conference venues would be most appropriate for submitting this work.

5.3 Christopher Raphael (Summer 2005-present)

I have been following Christopher's work since 1999 (Rap99). His probabilistic, graphical modeling techniques seem very appropriate for music and apply in an impressive variety of settings (e.g., chord and key labeling (RS03), audio alignment (Rap04)). The main reason I can't use his work off-the-shelf is that his focus is scored music. Recently, Tom Helliwell awarded me some *Mellon* funds to support my career development. I spent part of these funds to visit with Chris in Bloomington for about a week in the Summer of 2005. We brain-stormed about his methodology and how it might be applied to improvised settings. As I make headway on these ideas during my sabbatical, I plan to make a return visit for further collaboration.

6 Research Students

Table 2 summarizes my undergraduate research collaborators since I've come to Harvey Mudd College. Sections that detail most of these entries follow this table. Summer work was done with research assistants (*Beckman* and Mellon funding). Unless otherwise indicated, research occurred as a *Computer Science Research (CS 185/186)* course. For course-based classes with variable units, the number of units for which a student enrolled are shown in parentheses.

6.1 Katy Perdue (Summer 2003)

For the first half of the summer, Katy developed a tool in *Prolog* that we could use to explore various relationships between harmony and melody in jazz. In the second half, she explored the Short Term Fourier Transform (in *Matlab*), focusing on its behavior when applied to pitch tracking on the violin. To conclude this latter work, Katy provided an excellent online overview.

6.2 Aja Hammerly (Spring 2004)

Aja rewrote and significantly extended Katy Perdue's *Prolog* tool. One really cool extension she made enabled her to use the tool to estimate frequency

Table 2: Summary of Undergraduate Research Collaborators.

Summer 2002	**Fall 2002**	**Spring 2003**
in Germany	1st semester at Mudd	E4 project
Summer 2003	**Fall 2003**	**Spring 2004**
Mark Nelson Katy Perdue	Mark Nelson (3)	Mark Nelson (3) Aja Hammerly (3) Josh Kline (1)
Summer 2004	**Fall 2004**	**Spring 2005**
Chris Erickson Brian Young	Chris Erickson (3)	research w/students in CS 182-1[a] [a]CS 182-1 is a machine learning seminar I taught.
Summer 2005	**Fall 2005**	**Spring 2006**
Stephen Jones Aaron Wolin	Michael Beyer (3) CS 197 (3)	John McCullough (3)
Summer 2006	**Fall 2006**	**Spring 2007**
NSF REU		

distributions of pitch class for specific musicians' solos. She then used these distributions to generate melodies, and used these to construct a musical performance for one of Bill Alves's courses. Aja presented her work during our department's Presentation Days. She also wrote a preliminary draft of a comprehensive user manual for the tool she developed.

6.3 Mark Nelson (Summer and Fall, 2003, and Spring, 2004)

In the summer, Mark developed a real-time MIDI rhythm quantizer in *C++* using the *PortMidi* library. As we started using this tool to record solos played on a MIDI keyboard, we became interested in how accurately the system was able to time-stamp incoming MIDI data. This interest led to an extensive exploration of real-time MIDI performance analysis during the fall and spring. Part of this investigation required that we build a piece of electronic hardware, called a *Transcoder*, that had been developed by other computer music researchers. This work resulted in two conference papers in 2004, one accepted to the International Computer Music Conference and the other to the New Interfaces for Musical Expression Conference. We jointly presented these papers in Florida and Japan respectively. Mark also presented various aspects of his work during the department's Presentation Days and during a portion of a CS Colloquia dedicated to student research.

6.4 Brian Young and Chris Erickson (Summer, 2004)

Brian and Chris both developed mixture model toolkits in *Matlab*; Chris focused on gaussian mixtures and Brian on multinomial mixtures. They used their toolkits to explore several cluster validity measures. Brian's explorations involved synthetic data, whereas Chris's involved image data. Their summer experience culminated a poster of their work, which they presented at an HMC student research forum held in the Fall of 2004.

6.5 Chris Erickson (Fall, 2004)

Together, we read Ron Singer's paper on *Probabilistic Suffix Automata (PSA)* (RST96), an SLA that potentially could be very useful in music applications because of its ability to navigate back and forth between a directed graph and a tree-based representation (provided certain assumptions hold). After reading this paper, Chris implemented several of algorithms he had learned about in *Java*.

6.6 Stephan Jones and Aaron Wolin (Summer 2005)

These two students developed a prototype version of *Impro-Visor* (Section 5.2) and were jointly advised by Bob Keller and myself. Student accomplishments included:

1. the design and development of a GUI;
2. the design and development of melodic and chordal representations; and
3. the development of rules capturing various pieces of jazz knowledge.

Development took place in *Java*; the students reworked aspects of *jMusic* to make it usable.

6.7 CS 197 (Fall 2005)

David Buchfuhrer, Julian Mason, and Susanna Ricco approached me about advising them for a 3 unit independent study in machine learning. Things the group did included:

1. reading portions of classic machine learning texts, journal articles, and conference papers;
2. implementing gaussian mixture models in *Matlab* and applying these to an image compression task; and
3. implementing Hidden Markov Models in *Python*.

6.8 Michael Beyer (3 units, Fall 2005)

Michael had taken CS 182-1 from me in the Spring of 2005, where his research project focused on N-grams, a relatively simple SLA. Michael found this work so interesting that he decided to continue in the Fall via CS 185. During that time, Michael read Begleiter's paper (BEYY04) and began playing with the code and data that had been developed therein. Achievements included:

1. integrating Begleiter's *Java* code into *Matlab*;
2. using Toiviainen's *Matlab* MIDI Toolkit to explore Begleiter's music data;
3. applying the *Prediction by Partial Match (PPM)* algorithm to Begleiter's data, *All of Me* (Version 1);
4. using prediction to generate new melodies; and
5. exploring various properties of the learned PPM model.

6.9 John McCullough (3 units, Spring 2006)

John had taken CS 182-1 from me in the Spring of 2005, where his research project focused on pitch tracking using wavelets. I was very impressed with the final paper John produced in that class and would really like to see parts of it published, so I encouraged him to sign up for research with me while I am on sabbatical. The primary difficulty John encountered was described in Section 4.3. By focusing on pitch tracking alone, his problem was more difficult than it might have been had he simultaneously integrated it with onset detection. This Spring, I hope that John can address this issue by adopting some of Chris Raphael's techniques. I would also like to collaborate with him on extending Raphael's algorithm to partial-score settings.

6.10 National Science Foundation REU (Summer, 2006)

I am participating in the CS Department's National Science Foundation REU this summer. This link provides more information.

References

P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler. A tutorial on onset detection in music signals, 2004. submitted for publication.

R. Begleiter, R. El-Yaniv, and G. Yona. On prediction using variable order markov models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004.

E. Cambouropoulos, T. Crawford, and C. S. Iliopoulos. Pattern processing in melodic sequences: Challenges, caveats & prospects. In *Proceedings from the AISB'99 Symposium on Musical Creativity*, 1999.

Darrell Conklin and Ian Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.

C.L. Blake D.J. Newman, S. Hettich and C.J. Merz. UCI repository of machine learning databases, 1998.

David Gerhard. Pitch extraction and fundamental frequency: History and current techniques. Technical Report TR-CS 2003-06, University of Regina, 2003.

Holger H. Hoos, Keith A. Hamel, Kai Renz, and Juergen Kilian. The guido music notation format: A novel approach for adequately representing score-level music. In *Proceedings of the 1998 ICMC*. International Computer Music Association, 1998.

Bob Keller, Stephen Jones, Belinda Thom, and Aaron Wolin. An interactive tool for learning improvisation through composition. Technical Report HMC-CS-2005-02, Harvey Mudd College, 2006.

P. Leveau and R. Gael. Methodology and tools for the evaluation of automatic onset detection algorithms in music. In *ISMIR 2004 Conference Prooceedings*, 2004.

Benoit Meudic. A causal algorithm for beat tracking. In *2nd conference on understanding and creating music*, 2002.

M. Puckette, T. Apel, and D. Zicarelli. Real-time audio analysis tools for pd and msp, 1998.

Christopher Raphael. Automatic segmentation of acoustic musical signals using hidden markov models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 10(4), 1999.

Christopher Raphael. A hybrid graphical model for aligning polyphonic audio with musical scores. In *Proceedings of the ISMIR*, 2004.

C. Raphael and J. Stoddard. Harmonic anlaysis with probabilistic graphical models. In *Proceedings of the ISMIR*, pages 45–52, 2003.

Dana Ron, Yoram Singer, and Naftali Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25(2–3):117–149, 1996.

H. Schaffrath. The essen folksong collection. Technical report, Center for Computer Assisted Research in the Humanities, 1995.

Christian Spevak, Belinda Thom, and Karin Höthker. Evaluating melodic segmentation. In *Proceedings of the 2nd Conference on Music and Artificial Intelligence, ICMAI'02*, Edinburgh, Scotland, 2002.

B. Thom. Predicting chords in jazz: the good, the bad & the ugly. In *Working Notes of the IJCAI Workshop on AI and Music*, Montreal, 1995. International Joint Conferences on Artificial Intelligence.

Belinda Thom. Interactive improvisational music companions. *User Modeling and User-Adapted Interaction, Special Issue on User Modeling and Intelligent Agents*, 13:133–177, 2003.

Garaint Wiggins and Marcus Pearce. An empirical comparison of the performance of ppm variants on a prediction task with monophonic music. In *Proceedings of the AISB'03 Symposium on Creativity in Arts and Science*, 2003.