Beatrix: The Amorphous Drum Ensemble v3

Jonathan Bachrach
MIT Amorphous Computing Seminar

Fall, 2002

Abstract

We present a model of an African drum ensemble. We develop a model of polyrhythms, their creation, and their goodness. We assume that drummers are distributed and autonomous and must coordinate their rhythms and timing amongst themselves. We introduce a new amorphous clustering algorithm and show that distributed temporal synchronization is possible under certain restrictions. We develop a suite of visualizations that provide key insights into the structure of polyrhythms. Finally, we present interactive mechanisms that allow users to participate in and investigate polyrhythm evolution.

1 Introduction

The goal of this project is to develop an understanding of the nature of African drum ensembles and to create an engaging interactive art installation modeling a drum ensemble. In an African drum ensemble, polyrhythms are created in a coordinated yet decentralized fashion. As an art installation, Beatrix permits users to interact through the introduction of their own beats, voices, and tempo changes.

Beatrix is built as a matrix of 8-24 self-powered speakers mounted on a white panel with visualizations superimposed on top with a projector. The ensemble is composed of drummers performing solos or accompaniment. The accompaniment parts are simple rhythms composed of limited voices that together fuse into a parsimonious yet complex interlocking rhythm. Appropriate accompaniments are determined based on other accompaniments. Soloists perform highly syncopated rhythms that cut across this accompaniment foundation. Soloists are also able to change tempo, change rhythms, and crescendo with agreed on patterns. Finally, sensor input controls tempo and allow the introduction of parts and voices.

Beatrix is a music producing machine and because music is a subjective experience, it is challenging to measure its success. Nevertheless, we measure success on a number of more objective axes. Although we focus on typical drum ensemble plausible scenarios, we also consider both larger ensembles and

```
1+++2+++3+++4+++
1234567890123456
DUUUDUUUDUUUDUUU
```

Figure 1: One measure of basic 4/4 time divided into 16 secondary pulses, with 4 downbeats (D) and 12 possible upbeats (U).

```
1++2++3++4++
123456789012
DUUDUUDUUDUU
```

Figure 2: One measure of basic 12/8 time divided into 12 secondary pulses, with 4 downbeats (D) and 8 possible upbeats (U).

algorithms that are more optimal, scalable, and generally more amenable to computer implementation.

2 Rhythm Basics

Beatrix is based on the principles of African polyrhythms [5]. In this section, we describe rhythm basics. In its most basic form, rhythm is time structuring. We start by introducing temporal structures of a single rhythm.

We assume that time is divided up into a sequence of *measures* with a finite period and resulting tempo. Measures are then further evenly divided up into a series of *primary pulses*. Beats played on primary pulses are called *downbeats*. For the purposes of this presentation we assume that there are four primary pulses per measure, with the first beat, called the *one*, occurring at the beginning. Additionally, there is a secondary structure, whereby finer grain *secondary pulses* subdivide the primary divisions. Beats played on non-primary secondary pulses are called *upbeats*. Two popular secondary pulse schemes are three secondary pulses per primary pulse and four secondary pulses per primary pulse. These correspond respectively to 12/8 (shown in Figure 1) and 4/4 (shown in Figure 2) time in Western music. We focus on 4/4 time (i.e., 16 secondary pulses per measure) in this presentation.

Now that we have an evenly space temporal grid, we can describe rhythm patterns as the subset of the pulses played. Played pulses are called *beats*. One representation of a pattern is a bit vector of length equal to the number of secondary pulses in one measure. Ones in a bit vector pattern are interpreted

```
1001001000101000
```

Figure 3: The beatbox (bitvector) notation of a basic 4/4 clave part.

Drum/Tone	Character
Bass	B
Tone	T
Slap	S
Doundoun	D
Sangban	S
Kenkeni	K

Figure 4: African drum notation font.

```
C--C--C---C-C---
```

Figure 5: The alphabetic beatbox notation of a basic 4/4 clave part.

as played beats while zeroes are interpreted as rests. This is equivalent to *beatbox* notation shown in Figure 3, which is a popular drum notation originally popularized by drum machines. Although the beatbox notation is linear, we assume that patterns are played in a loop and thus the one is both the beginning and end of a pattern.

Now rhythms would be pretty boring without a variety of drum sounds. We call these drum sounds *tones*. In this treatment (and without loss of generality), we assume that each drummer plays a single drum which produces a single tone. At each time, a drummer is playing a particular pattern with a certain tone.

Drummers typically utilize a phonetic scheme for communicating rhythms and can often be seen speaking these phonetic representations under their breath during solos [1]. An extensive font of African tones is available from [3] and Figure 4 shows a typical mapping. These tone letters can now be used instead of 1's and 0's in the bitvector representation as shown in Figure 5. This allows for a very condensed representation of tone and time.

The fundamental aspect of African rhythm is the notion of a *polyrhythm*, which is the composition of multiple drum patterns. *Cross rhythm* is another name for polyrhythm, emphasizing the concept of the interaction between multiple contrasting drum patterns. The character of cross rhythms can be elucidated by examining their interplay. Beats that coincide create a sense of calm and order, whereas disagreeing beats engender a sense of motion or disorder. In traditional African polyrhythms, the "one" is the focus of coincident beats. Finally, patterns fuse more completely when they are played with more similar tones and stay separate when played with less similar tones. Spatialization can also affect the separation of patterns.

[1] Phonetic representations are typically optimized to allow efficient articulation and some even also represent the hand usage.

3 Assumptions

Beatrix makes a number of assumptions both to ensure realism and pragmatism. The assumptions meant to parallel real world ensemble qualities are as follows. Drummers are placed in two-dimensional space, have a finite dimension, and radius of communication. Neighbors are those drummers located within a radius of communication. We assume that drummers are autonomous agents, computing independently, but are loosely coupled by the perception of the beats produced by their local neighbors. A beat produced by a drummer is broadcast to all of its neighbors much as sound travels through air [2]. We assume that a small percentage beats might not be heard and that our algorithms should be robust in the face of these failures.

The assumptions that are meant to simplify beatrix and that may not be realistic are as follows. We assume but do not rely too heavily on the fact that drummers have unique ids. We could use random ids without loss of generality. For the purposes of this paper, we assume that drummers are fixed in space, approximately evenly spaced, and that the spatial coordinates of a few drummers are known. We assume that the one pulse is locally broadcast at the beginning of every period. The problem of inferring the phase of a drummer from merely the recently heard beats is beyond the scope of this paper. Beats are communicated instantaneously and without attenuation. Although sound travels fast enough to produce effectively zero delay, we do believe that there are large enough and probabilistic enough delays in the perception and motor control systems of drummers to justify factoring delays into temporal algorithms. We choose not to for this paper. Finally, we do not address the selective attention of drummers nor other human cognitive artifacts.

4 Basic Amorphous Computations

Beatrix requires a basic set of organizational structures and distributed computations in order to coordinate certain music behaviors such as soloing and part specialization. In this section, we describe the basic structures and computations that will form building blocks for higher level behaviors. While these algorithms have nice computational properties, some of them might not be plausible for human drummers in African ensembles.

4.1 Flooding

It is often necessary to get messages out to all drummers. A simple flooding procedure can achieve this. A message is sent from a particular originating drummer to its neighbors. When a drummer receives a new message it records it and sends it out to its neighbors. Already heard messages are ignored.

[2] except that sound is attenuated with distance.

4.2 Election

In certain situations, it is important to elect k leaders. We can choose leaders by using their unique id's as follows [6] [3]. Each drummer maintains a list of their top k ids initialized to its id. During each round drummers merge max ids heard in the last round and then broadcast these lists of ids if their max list changed. This procedure stop when there are no more changes in lists. At this point all lists will converge to the max k ids. The leaders are then the drummers with ids in their lists.

4.3 Gradients

Gradients are a biologically inspired mechanism for estimating minimum relative measures through local communication [7] [9] [10]. For example, relative drummer distance to soloist could be estimated by maintaining minimum communication hop counts to the soloist. In more detail, a seed drummer initiates a gradient by sending its neighbors a message with a zero count. When a drummer receives a message with a count less than previously heard, then it remembers it and sends a message along to its neighbors with count incremented by one. This procedure completes when no more updates occur. Hop-counts can then be interpreted as an estimated distance by multiplying by the radio range. Smoothing hop-counts with neighbors can increase the resolution.

4.4 Aggregation

Often it is necessary to aggregate information across drummers. One way to achieve this is to construct a spanning tree such that children communicate their values to their parents and values are aggregated on their way up towards the root of the spanning tree. The root will contain the total aggregated value at the end of the aggregation process.

A spanning tree can be constructed starting from the root [6]. A parent recruits children by broadcasting a recruit message including the parent's id. Unrecruited drummers hearing the recruit message record the parent message and then recruit their own children recursively. The process completes when all drummers (except the root) have parents. A minimum spanning tree can be constructing by choosing nearest parents using gradients.

5 Temporal Structure

In order for a drum ensemble to play coherent polyrhythms, their primary pulses must coincide. In order for this to happen, drummers' phases and periods must be equal. We can identify a couple dimensions that control the level of complexity of this problem. The first is whether there are none, one or multiple true time keeping drummers. The second is whether drummers can

[3] Nagpal and Coore [8] present an algorithm not requiring unique id's.

hear all the time keepers or not. We consider these various scenarios but in the end develop an algorithm able to temporally coordinate in real-time using distributed local neighborhood coordination. More realistic models might have to consider probabilistic delays introduced by perceptual and motor systems. More robust algorithms require more structure and control and as such are less realistic as models of natural drumming situations where time synchronization is fallible.

5.1 Phase Estimation

The first problem in temporal coordination of drummers is the problem of estimating a common phase. To simplify matters in this subsection, we assume that all drummers play at the same tempo.

5.1.1 Direct Phase Estimation

The simplest mechanism for phase locking with a single true and globally audible time keeping drummer is as follows. When each of the drummers hear a time keeper's one pulse they adjust their phases forward or backward in proportion to the discrepancy with the time keepers phase. The update equations is:

$$\phi_j(t+1) \quad = \quad \phi_j(t) + \alpha(\phi_i(t) - \phi_j(t)) \tag{1}$$

where $\phi_j(t)$ is the phase of drummer j at time t, drummer i is a neighboring drummer time keeper, and $0 < \alpha << 1$. We can increase the speed with which a drummer tracks phase changes and the extent to which tracking is sensitive to noise by increasing α. Lower α makes drummers track slower but also makes them less sensitive to fluctuations in phase noise.

5.1.2 Mirollo and Strogatz

Mirollo and Strogatz [13] present a model of synchronization of biological oscillators. They assume that the oscillators' frequencies are very close and that each oscillator can sense the firings of all the other oscillators. Upon hearing the firing of another oscillator, an oscillator advances its phase as a nonlinear function of its current phase, phase advancement increasing with phase. They are able to prove convergence for a whole population of oscillators starting with random phases. Unfortunately, its not obvious how to extend their model to the case of local sensing and unequal frequencies.

5.1.3 Synchronous Fireflies

Buck and Buck [2] present a model of the synchronization of fireflies. They assume that firefly firing frequencies are very close and that each firefly can sense the firings of fireflies with a strength inversely in proportion to the square of the distance. Fireflies maintain an excitation level which is incremented at

regular intervals. When the excitation exceeds a threshold the firefly fires and resets its excitation to zero. At the same time, if the accumulated received light from neighboring fireflies exceeds a trigger value, then a given firefly resets its excitation to zero (without firing). However, if a firefly is close enough to its firing threshold, then it proceeds along undisturbed. While a nice model of fireflies, it's not clear how to extend this to unequal frequencies, not to mention how to prove convergence.

5.2 Period Estimation

Unfortunately, it is not enough to just coordinate drummers' phases. Drummers must also coordinate their tempo as we can not assume that drummers will all play at the same tempo.

5.2.1 Direct Period Estimation

A simple mechanism for coordinating neighboring periods is for a drummer to attempt to directly estimate a time keeper's period and to adjust its period to minimize the discrepancy. Drummers can estimate a period at the onset of a one by subtracting the current time from the time of the previous one. The estimation procedure is as follows:

$$\hat{\gamma}_i(t) = \tau_i(t) - \tau_i(t-1) \quad (2)$$

$$\gamma_j(t+1) = \gamma_j(t) + \alpha(\hat{\gamma}_i(t) - \gamma_j(t)) \quad (3)$$

where $\tau_i(t)$ is the time of drummer i's one, $\gamma_j(t)$ is the period of drummer j at measure t, $\hat{\gamma}_i(t)$ is the current estimate of drummer i's period, and $0 < \alpha << 1$. As in direct phase estimation, adjustments in α tradeoff tracking speed for insensitivity for target noise.

5.2.2 Proportion Estimation

If all drummers can hear each other and know the total number of drummers, then one can imagine drummers estimating the average tempo by measuring the proportion of ones heard in a given period [4]. If the number of ones is less than the total number of drummers, then the period could be increased, and if greater, then decreased. Although not perhaps the most realistic model, we have shown experimentally that it will in fact find a common tempo.

5.3 Phase and Tempo Estimation

We can combine phase and frequency estimation to form a comprehensive temporal synchronization algorithm. In particular, we can combine direct phase and frequency estimation. Each time a drummer hears a neighbor it adjusts its phase and frequency to better correspond to its neighbors. Preliminary results

suggest that even for small neighborhood and a large number of drummers, this mechanism converges quickly.

Another way to achieve the same effect is to use a simple form of phase lock looping [16] (PLL) which uses errors in phase to drive corrections in frequency:

$$\tau_j(t+1) \quad = \quad \tau_j(t) + \alpha(\phi_i(t) - \phi_j(t)) \tag{4}$$

This is a mechanism for adjusting phase and frequency together by only modulating the frequency. The lowest error solution will be the one where the phase and frequencies agree. Convergence can be proven for the two player case, but not for the n player version as we desire.

5.4 Temporal Synchronization Success

Beatrix plays as an ensemble coordinating phase and tempo. One major piece of work is to develop algorithms that can permit distributed tempo tracking and phase locking. The easiest version of this involves the global communication of each drummer's downbeats. The hardest version involves local communication of the beats themselves without obvious downbeats. A simple measure of success of phase locking and tempo tracking is the average square error across the ensemble. A more advanced measure of success for tempo tracking is the maximum sustainable change in tempo per measure. We have described a simple and intuitive algorithm for phase and tempo synchronization which tracks quickly and accurately in simulation for plausible ensembles. More work is needed to properly test this algorithm and to improve the realism of the simulation.

6 Specialization

Clearly, multiple parts are necessary for the creation of compelling polyrhythms. Each part involves different patterns and tones. Often parts will each be played by multiple drummers with slight variance in time and tone giving a part more depth and power. As in an orchestra, parts are more effectively played in spatially colocated groups. In this section, we investigate algorithms for part assignment.

6.1 Small Ensemble Specialization

In the simplest configuration, drummers are preassigned roles and particular drums with their specific tones. A drum leader will then assign patterns to each of the drummers. This seems quite reasonable for small ensembles but unfortunately does not scale so well.

6.2 K-Means Clustering

The most popular algorithm for dividing up a group of examples into k clusters is called *k-means clustering*. The algorithm starts by assigning best guess to k

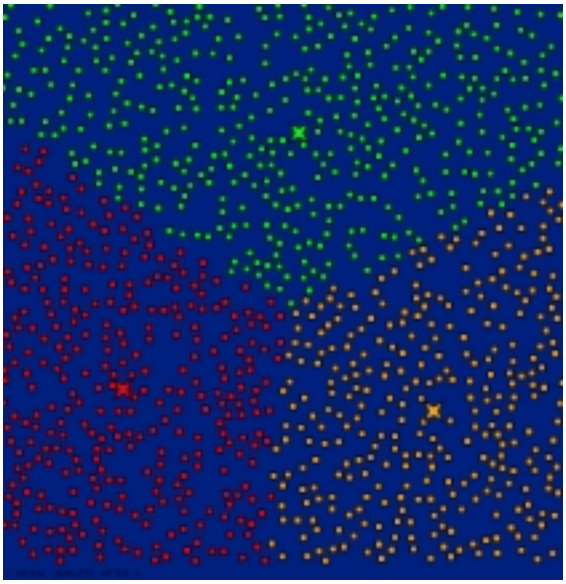

Figure 6: k-means clustering after 6 iterations with 3 clusters and 1024 drummers.

means. Until there is no more change in the means, we use the estimated means to classify the examples into k clusters and then replace each mean i with the mean of all examples from cluster i.

The way that this algorithm could roughly apply to ensemble specialization is to map examples to drummers' spatial locations, using these locations to determine both cluster membership and means. In particular, a drummer's cluster membership is calculated as the cluster which has the closest mean and new means are calculated as the center of mass of a cluster's members. Figure 6 shows the result of running k-means clustering on 1024 drummers with 3 clusters.

6.3 Amorphous K-Means Clustering

Running k-means clustering on an amorphous computer presents unique challenges. Fortunately, most of the building blocks are already available. The main challenges are to calculate the center of mass of clusters and to calculate minimum mean distances. In order to simplify the algorithm, we assume that we assign mean drummers, that is drummers that are at the center of mass of their clusters.

Cluster membership can be determined by choosing a cluster with a minimum estimated distance to its mean. Distance to mean drummers can be estimated using smoothed minimum hop-counts calculated using gradients. Center of mass can be calculated using aggregation of cluster member coordinates and

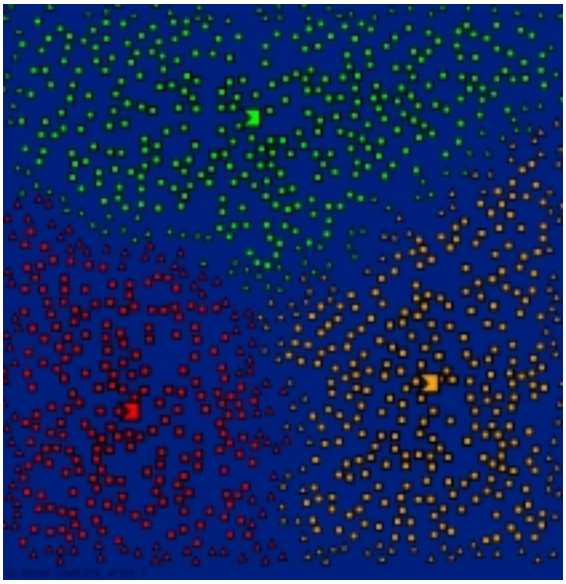

Figure 7: Amorphous k-means clustering after 3 iterations with 3 clusters and 1024 drummers. Triangles are drummers on cluster edges. Shape line width is proportional to hop-count distance.

cluster membership count. The center of mass then becomes the average x and y coordinates of cluster members.

A center of mass algorithm more in the spirit of amorphous computing can be based around gradient propagation. The edge of a cluster is detected by drummers noticing that neighbors are in other clusters. The edge is then seeded and gradients propagate inwards towards the center of a cluster where their values are maximal. Local maxima can then be detected by noticing that a drummer's gradient value is higher than its neighbors. The mean for a given cluster is chosen as the highest local maxima or randomly if there are multiple global maxima. Choose more optimal local maxima is a topic for future research. Figure 7 shows the result of using this amorphous style k-means clustering algorithm on 1024 drummers and 3 clusters.

7 Composition and Improvisation

In this section we examine in more depth the principles of polyrhythm composition and propose goodness measures and introduce likely compositional operators.

```
B---B---B---B---
C--C--C---C-C---
```

Figure 8: The basic 4/4 clave part contrasted against the 4/4 beat pattern.

7.1 Goodness Measures

We start by examining the attributes of good drum patterns. Though somewhat subjective, we articulate these features in hopes of formulating a goodness measure so as to better understand why successful rhythms work and unsuccessful rhythms fail and to potentially identify new rhythms. We back our choices by testing these attributes on both successful and unsuccessful patterns. We will use the popular 4/4 clave pattern shown in Figure 8 as a running example.

7.1.1 Monorhythms

We start with a few basic principles of creating single patterns in isolation. We suggest that successful single patterns can form the root of a whole family of rhythms, where a family is a set of variations or improvisations on the root pattern. By discovering new successful root patterns we can hope to also discover a whole new family.

We restrict our patterns to bit-vectors of a particular length, and for the purposes of this discussion, we consider only bit-vectors of length 16, corresponding to 4/4 rhythms. Some of our principles are described in terms of a generative model, that is, in terms of how to produce conforming patterns. Other principles suggest numeric goodness metrics that can favor certain patterns over others.

Conjecture 1 *A good pattern is rooted in the one, that is, the one is played.*

This is necessary to provide a moment of recurring tranquility and to aid in temporal synchronization. Secondary patterns can successfully start on an upbeat but primary patterns must be rooted.

Conjecture 2 *Patterns are aligned with an underlying pulse structure* [4].

Conjecture 3 *Interesting root patterns have a medium density of beats.*

Patterns with too low or high a density (number) of beats have too little structure and create too little interesting contrast. Furthermore, high density patterns are less appropriate for improvisations.

As an experiment consider the case of 4/4 patterns, where we have 16 possible beats and as a rule of thumb consider the best patterns to have between 4 (0.25) and 8 (0.5) beats (density). We only consider only rotational unique

[4] as a simplification, we do not consider swing in the playing of patterns

2/3 Pattern	Spread	Symmetry	4 Pattern	Spread	Symmetry
SS------	0.70	0.50	SSSS----	0.58	0.00
S-S-----	0.90	0.50	SSS-S---	0.75	0.50
S--S----	0.98	0.50	SS-SS---	0.75	0.50
S---S---	1.00	1.00	S-SSS---	0.75	0.50
SSS-----	0.61	0.25	SS--SS--	0.79	1.00
SS-S----	0.78	0.25	SSS--S--	0.79	0.50
S-SS----	0.78	0.25	S-SS-S--	0.90	0.00
SS--S---	0.84	0.75	S-S-SS--	0.90	0.50
S--SS---	0.84	0.75	SS-S-S--	0.90	0.50
S-S-S---	0.94	0.75	S-S-S-S-	1.00	1.00
S-S--S--	0.98	0.25			

Figure 9: The sparse 8-bit patterns with 2 to 4 beats and their spread and symmetry values.

patterns, that is, we choose only one pattern per set of equivalent pattern under rotation. Out of 65536 possible 16-bit patterns, we have 38506 unique patterns and 2418 sparse patterns. Out of 256 8-bit patterns, there are 154 unique patterns and 21 sparse patterns with 2 to 4 beats as shown in Figure 9. These numbers are small enough to enumerate and discover new interesting patterns.

Conjecture 4 *Compelling patterns have a good spread of beats across a pattern.*

Good spread is also justified by the need for interesting structure. Spread can be measured with the following equation:

$$S(p) \quad = \quad \sum_{b_1, b_2 \in Nbrs(p)} \log D(b_1, b_2) / (|Ones(p)| * \log(N/|Ones(p)|)) \qquad (5)$$

where p is a bit-vector, $Ones(p)$ is the set of ones in p, $Nbrs(p)$ is the set of neighboring ones, $D(b_1, b_2)$ is the interbeat distance, $|Ones(p)|$ is the number of ones in the vector, and N is the size of the bit-vector. A perfect spread is normalized to be one using the maximum spread for the given number of beats as the denominator of Equation 5. Figure 9 shows the spread values for the sparse 8-bit patterns.

Beyond good patterns having a solid structural basis, they also have the qualities of movement and surprise and contrasting moments of order and disorder.

Conjecture 5 *Good patterns exhibit a certain amount of asymmetry.*

Asymmetry creates contrast between parts of itself and other patterns. The asymmetry of a particular pattern can be measured as the deviation from itself when broken up into equal parts. One way to calculate symmetry is to count the number of beats in agreement starting at each phase and normalized by the

maximum possible agreements. Asymmetry could then just be the negative of the symmetry. Figure 9 shows the resulting symmetry values.

Conjecture 6 *Interesting patterns often utilize syncopation to create expectation and surprise.*

We have yet to devise a good measure of syncopation other than an obvious variant of the proposed symmetry metric. We can see that spread and asymmetry/syncopation are somewhat at odds. Interesting patterns (e.g., `S-S--S--`) strike a balance between the two properties.

The clave pattern from Figure 8 exhibits all of these basic qualities. In particular, it starts with an even pattern and then changes gears to realign itself with the pulse beat pattern.

Goodness metrics were developed that we think highly correlate with success. For example, a pattern goodness metric has been invented and in the future will be used to exhaustively test all 16 bit drum patterns to both confirm the recognition of the goodness of standard patterns and to discover new patterns with high goodness. We plan to use linear regression to find a weighting of the goodness features that most faithfully predicts success.

7.1.2 Polyrhythms

Now we turn to the question of how polyrhythms combine to create successful polyrhythms. Clearly, if all drummers played the same patterns, the resulting composite polyrhythm would not be much more interesting than if one drummer were to play it. So patterns must complement each other, that is, patterns should have beats that are not in common. Furthermore, common beats would more likely be on primary pulses, although patterns can have no beats in common.

In the future, we will be investigating measures of goodness for polyrhythms. One promising direction is to measure the mutual information between cross rhythm patterns [11].

7.2 Compositional Operators

We have described desirable basic qualities of patterns but we have not necessarily invented mechanisms to create them. One obvious technique is to copy popular patterns. This is not the final answer but a good point of departure [5]. We would hope to create new patterns that are both interesting and novel in their own right, but also best reflect the actively played set of (or library of root) patterns.

A popular and intuitive popular approach is to model the creation of patterns as a form of evolution [12] [14], whereby new patterns are created based on or as a function of actively played patterns. New patterns could then be evaluated using goodness measures described in Section 7.1.1. Others [15] have

[5] we would like to use our measures of goodness developed here to choose these root patterns

Period	Pattern
8	B-------B-------
7	B------B------B-
6	B-----B-----B---
5	B----B----B----B
4	B---B---B---B---
3	B--B--B--B--B--B
2	B-B-B-B-B-B-B-B-

Figure 10: The set of basis patterns with beats occurring at constant frequencies.

examined the use of genetic based operators for generating improvisational music. Our approach differs by grounding the genetic operators in plausible drum improvisation techniques instead of merely genetic operations.

We can actually create all of the standard rhythms using a set of basis patterns and combinator operators. The set of basis patterns are the patterns created by placing beats at evenly spaced intervals. Figure 10 shows a number of basis patterns for successive periods from 2 to 8 secondary pulses. The most compelling basis patterns are those of periods 2, 3 and 4. We can see that the clave pattern (shown in Figure 8) is actually a crossover of period 3 and 2 basis patterns.

New patterns can be created by manipulating prior patterns. Patterns can be rotated to created a new variant patterns. Rotation is an important source of novel patterns and in fact is often used with classic clave patterns to create a family of patterns. Patterns can be imitated with mutation, where there are certain probabilities of adding or dropping beats.

Patterns can be combined in a few different ways. First, two patterns can be crossed over, with the first part of the pattern coming from the first part of one pattern and the last part coming from the last part of the other pattern. Second, variety can be introduced by merging multiple patterns creating a new composite pattern.

7.3 Compositional Control

Now that we have introduced measures of rhythmic goodness and have described a number of compositional operators, we turn to the question of compositional control. We first describe the current composition control algorithm in beatrix and then consider more compelling designs.

The current version of Beatrix is a simple improvisational rule-base system. Each drummer plays a pattern for a random amount of time from 3 to 16 measures. When it is time for a drummer to improvise, it randomly chooses between four different operators: *template*, *mimicry*, *syncopation*, and *rotation*. Template chooses patterns from a set of classic patterns and applies a random amount of mutation. Template introduces root patterns into the rule-base and are chosen to balance the distribution of tones. Drum tones are divided into four

categories: *low*, *mid*, *high*, and *bell*. Patterns are chosen evenly between these categories. The remaining operators base their improvisation on patterns being played by themselves or neighbors. Mimicry copies an existing pattern and again applies random mutation. Syncopation adds intervening beats to existing rhythms. Rotation creates new patterns through small amounts of rotation. These four operators produce both a feeling of progression and allow a large amount of novelty.

Despite such a simple control design, members of the ensemble form distinct roles and play parts which together are more than the sum of the parts. Each part is unique and simple enough to be uninteresting on its own but together produce a captivating ever changing rhythm. Part determination takes place in a distributed fashion.

There are many opportunities for improvement. Although crude versions of call and response behaviors occur, we would like to build this more fundamentally into our control regime. The use of dynamic range can be a very powerful compositional technique. For example, we would like to have periods of coordinated inactivity followed by the incremental addition of drummers. We would like more informed decisions as to the introduction of beats and tones. Mutation can be directed towards creating new patterns that introduce syncopation or that more generally increase mutual information [11]. Finally, we would like to add more improvisational operators. Two likely candidates are crossover and merging.

7.4 Musical Success

Beatrix produces music that is captivating and enjoyable while at the same time ever changing. The ultimate measure of success would be whether this system could be sold in quantity, commissioned for art exhibitions, and would be actively listened to. Short of this, a survey could be conducted or master drummers consulted. An informal but true measure of success could be whether people would actually spontaneously dance to the produced rhythms. Currently, we have yet to conduct such tests.

8 Visualization

In this section, we introduce visualization techniques. Visualization is important for understanding the mechanisms of polyrhythms, developing alternative representations, and for creating audio/visual beauty. We start by describing important features and then go on to present a few visualization strategies.

8.1 Single Beat Representations

At the lowest level, for each drummer it is important to visualize the pattern and tone being played and the state of playing it. If a beat is being played at a particular moment, it is useful to have a visual indication of it. Also, it is

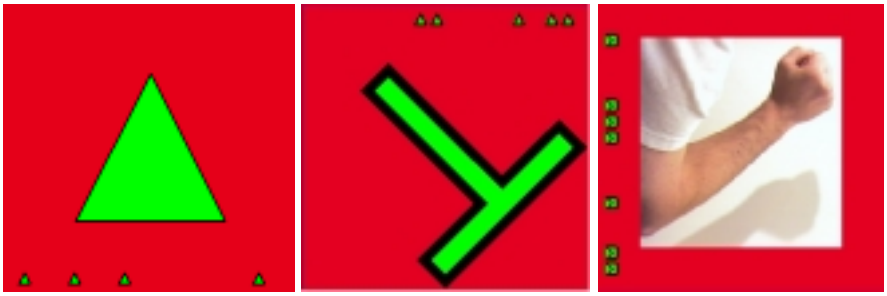

Figure 11: Single beat visualizations.

Figure 12: Linear pattern visualization.

informative to see the history of the recently played beats. We have investigated the use of size, color, and filmstrips (shown in Figure 11) to indicate the onset and history of beats.

It is also important to portray the tones. We have experimented with the representation of tones as colors, approximately mapping the sound frequency onto the color spectrum. For example, low frequency sounds would have cold colors such as blue and green, while high frequency sounds would have hot colors such as orange and red. Another equally compelling tone visualization scheme is through an alphabetic representation also shown in Figure 11.

Finally, it is important to show the phase of the drummer. The most obvious depiction is animated rotation. This works best with shape beat representations such as letters or squares.

8.2 Single Pattern Representations

The most straightforward visual representation of a pattern is the beatbox shown in Figure 12, that is, a linear pattern layout with beat shapes corresponding to played beats. The problem with this depiction is that the beginning and end of a pattern are disjoint even though a pattern is played cyclically. A radial representation is much more natural, although a linear representation is still useful in certain situations. The easiest realization of a radial representation is by just laying out the linear visualization along a circle.

8.3 Ensemble and Polyrhythmic Representations

At the higher level it is interesting to see the relationships between patterns and drummers. For certain tasks (such as temporal synchronization) it is important

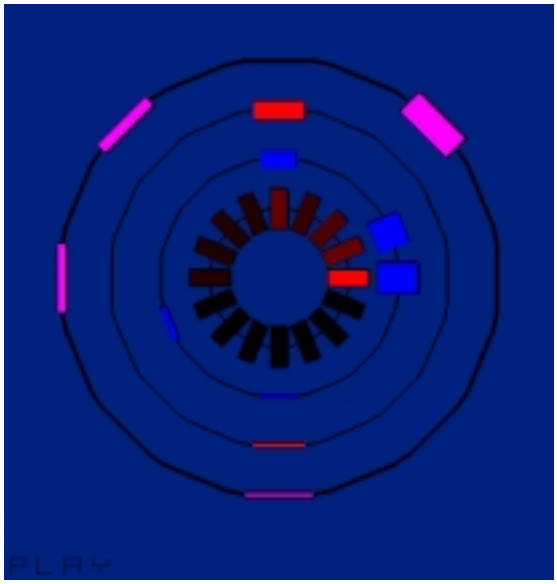

Figure 13: Radial visualization of drum ensemble.

to show the spatial layout and neighborhood relationship of the drummers. From there, it is interesting to see the relationship between drummers' playing state such as phase and tempo. During improvisation it is important to understand how and from where patterns emerged. Here the linear beatbox visualization is useful because two patterns can be lined up side by side in order to understand their relationship.

8.3.1 Radial Representation

The radial representation is also useful for depicting the relationship between patterns and the state of play. In this representation, there is a concentric ring for each drummer which is divided evenly into contiguous segments corresponding to each of the secondary pulses. A segment expands to full size when it is being played and fades back to zero afterwards. The center ring is used as a metronome, showing the pulses as they are crossed.

8.3.2 Grid Representation

The grid representation shown in Figure 14 is good at showing spatial structure and improvisational movements during play. Each of the drummers is placed in a square at a grid location, where there is an beat icon. The beat icon peaks in size when a corresponding beat is played and shrinks with time. The age of a pattern is represented by a square's background color, with bright red meaning brand new, bright blue meaning at least ten measures old, and interpolated

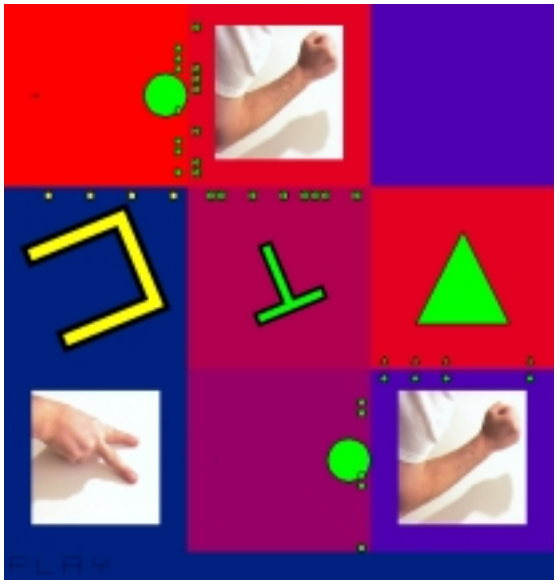

Figure 14: Grid visualization of drum ensemble.

colors between red and blue corresponding to less than ten measures old. At the introduction of a new pattern, the source and destination patterns are brought next to each other in order to show both an improvisation's origin but also the operation as a function of its origin.

8.3.3 Spatial Representation

The spatial representation shown in Figure 15 is good for showing more liberal spatial layouts. Drummers are randomly placed in a rectangle such that they do not overlap in space. A drummer's neighborhood is controlled by a radius parameter, which allows a drummer to either hear just its neighbors all the way up to hearing all the other drummers. In this configuration, drummer state is represented merely with a beat icon.

9 Interactivity

Beatrix allows the introduction of a certain amount of interactive control. Users can control the tempo, introduce patterns, and record new tones.

9.1 Computer Keyboard and Mouse

The simplest input device is the computer keyboard. Beatrix allows users to control many aspects through keyboard and mouse input. For example, the

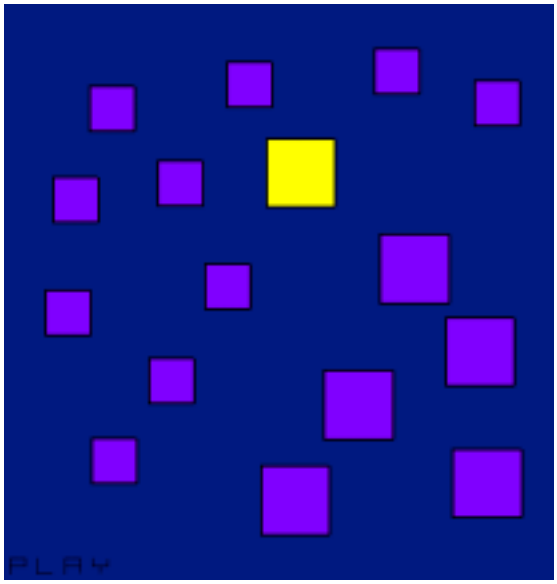

Figure 15: Spatial visualization of drum ensemble.

mouse can be used to selectively control the drummers' audio levels by adjusting each volume inversely with the distance from the mouse allowing users to focus in on particular drummers and their parts.

9.2 Midi Devices

Midi [1] input is very good for time critical input. Midi devices can be connected to Beatrix allowing the creation of patterns in real-time. Midi input can be used to control the overall tempo by measuring inter note onset intervals. We have also begun investigating using drum pattern recognition to control different aspects of Beatrix and the use of Beatrix as a semiautomatic music controller.

9.3 Voice Input

Beatrix also records digital audio input coming in through a computer microphone. This input can be recorded and used as new drum tones.

9.4 Interactivity Success

Beatrix creates an amusing interactive experience for the users. The minimal measure of success is that users feel that they have a real affect on the rhythms. The next measure is that users have fun interacting with beatrix. Ultimately beatrix works if users feel that they can introduce compelling rhythms and

control the rhythmic creation without prior drumming experience. Preliminary results are that users do feel that Beatrix is responding to them and that it is entertaining and compelling.

10 Future Work

In this paper, we introduced a very early version of Beatrix. We have only scratched the surface on the characterization of drum pattern goodness. We hope to push this much further in the future hopefully identifying new families of rhythms. Much future work remains including the incorporation of real microphones and modular visualization. Ideally there would be n Beatrix modules independently capable of producing sound and visualizations and listening to each other. Among other things, this would force more realism into the temporal synchronization algorithms.

11 Conclusions

We have presented a model of an African drum ensemble. We have developed a model of polyrhythms, their creation, and their goodness. We have introduced a new amorphous clustering algorithm and shown that distributed temporal synchronization is possible under certain restrictions. We have developed a suite of visualizations that provide key insights into the structure of polyrhythms. Finally, we have invented interactive mechanisms that allow users to participate in and investigate polyrhythm evolution.

References

[1] Midi Manufactures Association. The complete midi 1.0 detailed specification, 1995.

[2] J. Buck and E. Buck. Synchronous fireflies. *Scientific American*, pages 74–85, 1976.

[3] Lennart Hallstrom. African drum notation, 2002. www.djembe.net/djembe-e.shtml.

[4] Attila Kondacs, Oct 2002. Personal communication.

[5] C.K. Ladzekpo. Foundation course in african dance drumming. *Web Course*, 1995. http://www.cnmat.berkeley.edu/ ladzekpo/Foundation.html.

[6] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, Wonderland, 1996.

[7] R. Nagpal. Organizing a global coordinate system from local information on an amorphous computer. AI Memo 1666, MIT, 1999.

[8] R. Nagpal and D. Coore. An algorithm for group formation in an amorphous computer. In *Proceedings of the 10th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS'98)*, October 1998.

[9] R. Nagpal, H. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *The 2nd International Workshop on Information Processing in Sensor Networks*, 2003. Under Submission.

[10] C. Nusslein-Volhard. Gradients that organize embryo development. *Scientific American*, August 1996.

[11] Rebecca Reich. Simulation of a drum circle, 2002. http://web.media.mit.edu/ rreich/nmmproject.html.

[12] Karl Sims. Artificial evolution for computer graphics. In *Proceedings of SIGGRAPH*, pages 319–328, 1991.

[13] S. Strogatz and I. Stewart. Coupled oscillators and biological synchronization. *Scientific American*, pages 68–75, 1993.

[14] S. Todd and W. Lathum. *Evolutionary Art and Computers*. Academic Press, 1992.

[15] N. Tokui and H. Iba. Music composition with interactive evolutionary computation. In *Proceedings of the Third International Conference on Generative Art*, 2000.

[16] Wong and Lok. Theory of digital communications: Synchronization. http://www.wireless.ece.ufl.edu/twong/Notes/Comm/ch3.pdf.