Control Improvisation With Application to Music

Alexandre Donzé\textsuperscript{1}, Sophie Libkind\textsuperscript{2}, Sanjit Seshia\textsuperscript{1}, and David Wessel\textsuperscript{1}

University of California, Berkeley

Swarthmore College

September 29, 2013
Control Improvisation Problem

Generic Goal:

- Generate control inputs in a changing environment
- “improvised” $\Rightarrow$ randomized
- Soft constraint: stay “close” to some nominal controller
- Hard constraints: always satisfy some specifications

In the musical context:

- Generate random melodies in a given harmonic context
- Generated melodies that “sound similar” to a reference melody
Control Improvisation Problem

Generic Goal:

- Generate control inputs in a changing environment
- “improvised” $\implies$ randomized
- Soft constraint: stay “close” to some nominal controller
- Hard constraints: always satisfy some specifications

In the musical context:

- Generate random melodies in a given harmonic context
- Generated melodies that “sound similar” to a reference melody
Control Improvisation Problem

Formally, given

- A specification FSM $A^s$,
- A reference accepting word $w_{\text{ref}}$,
- A similarity (or “creativity”) measure $d$ between words,
- A target distance (or “creativity”) $d_r$ from $w_{\text{ref}}$
- Any $\epsilon, \delta > 0$

Find a FSA $A^c$ such that

$$w \in \mathcal{L}(A^s | A^c) \implies Pr(|d(w, w_{\text{ref}}) - d_r| < \delta) > 1 - \epsilon$$

The target creativity controls how far the improvisation should be from $w_{\text{ref}}$
Solving the Control Improvisation Problem

General Idea

- Use learning to create a generator automaton $A_g$ from $w_{ref}$
- Use supervisory control $A_{sc}$ of the generator to enforce the specification, so that $A_c = A_g || A_{sc}$
- Tune probabilistic transitions to meet the creativity criterion
Factor Oracle-based improvisations

1. **Input**: a training melody $w$, represented as a finite sequence of notes, e.g., $\text{abbbaa}$

2. **Learning**: construct a compact structure (factor oracle) representing sub-sequences of $w$, e.g.,

3. **Generation**: assign probabilities to transitions of the factor oracle and loop on repeated suffixes
Factor Oracle-based improvisations

1. **Input**: a training melody $w$, represented as a finite sequence of notes, e.g., abbbaab

2. **Learning**: construct a compact structure (factor oracle) representing sub-sequences of $w$, e.g.,

3. **Generation**: assign probabilities to transitions of the factor oracle and loop on repeated suffixes
Factor Oracle-based improvisations

1. **Input:** a training melody $w$, represented as a finite sequence of notes, e.g., abbbbaab

2. **Learning:** construct a compact structure (factor oracle) representing sub-sequences of $w$, e.g.,

3. **Generation:** assign probabilities to transitions of the factor oracle and loop on repeated suffixes
Properties of Factor Oracles [Assayag and Dubnov, 2004]

A FO reproduces factors of the original string with high probabilities. With low probabilities, it branches away, generating creativity.

Advantage
Unsupervised learning: no structure is assumed on the input

Disadvantage
Branching likely breaks the underlying structure of the original melody and creates violations of musical specifications
Musical Specifications

It Don’t Mean a Thing by Duke Ellington

- **Rhythm specifications**: Jazz melodies can be decomposed into licks separated by rests. Licks can be forced to begin on certain beats.

- **Pitch specifications**: Notes that sound “weird” with a chord cannot begin a lick and must approach a “good” sounding note.

In practice, we decouple generation of note durations (rhythm) and pitches...
Musical Specifications

It Don’t Mean a Thing by Duke Ellington

- **Rhythm specifications:** Jazz melodies can be decomposed into licks separated by rests. Licks can be forced to begin on certain beats.

- **Pitch specifications:** Notes that sound “weird” with a chord cannot begin a lick and must approach a “good” sounding note.

In practice, we decouple generation of note durations (rhythm) and pitches...
Musical Specifications

It Don’t Mean a Thing by Duke Ellington

- **Rhythm specifications**: Jazz melodies can be decomposed into licks separated by rests. Licks can be forced to begin on certain beats.

- **Pitch specifications**: Notes that sound “weird” with a chord cannot begin a lick and must approach a “good” sounding note.

In practice, we decouple generation of note durations (rhythm) and pitches.
Musical Specifications

It Don’t Mean a Thing by Duke Ellington

- **Rhythm specifications**: Jazz melodies can be decomposed into licks separated by rests. Licks can be forced to begin on certain beats.

- **Pitch specifications**: Notes that sound “weird” with a chord cannot begin a lick and must approach a “good” sounding note.

In practice, we decouple generation of note durations (rhythm) and pitches.
Improviser

Input training: chords + melody

E7  A7  etc

Rhythm Improviser

Rhythm Specifications

{\{, \gamma, \ddot{\gamma}, \dddot{\gamma}\}\}}

Current Beat

Generate notes durations or rests

Pitch Improviser

Pitch Specifications

A7

Current Chord

Generate notes pitches

Generated improvisation

E7  A7  etc
Improviser

Input training: chords + melody

E7  A7  etc

E7  A7  etc

Rhythm Improviser

Rhythm Specifications

Current Beat

Generate notes durations or rests

Pitch Improviser

Pitch Factor Oracle

Current Chord

Generate notes pitches

Generated improvisation

UC Berkeley

Control Improvisation With Application to Music

Sophie Libkind
Improviser

Input training: chords + melody

E7

ˇĹĹ

ˇ

ˇ

ˇŤŤŤŤ

ˇ

ˇšššš

ˇ

ˇÚÚ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ

ˇ

ˇÚÚ

ˇ

ˇĹĹ

ˇ

ˇšššš

ˇ
Improviser

Rhythm Improviser

Rhythm Specifications

Rhythm Factor Oracle

Current Beat

Generate notes durations or rests

Pitch Improviser

Pitch Factor Oracle

Current Chord

Generate notes pitches

Generated improvisation

Input training: chords + melody

E7

A7

etc

UC Berkeley

Control Improvisation With Application to Music

Sophie Libkind
Measuring Creativity

Two sources of divergence:

- Replication probability (probability of not branching in the FO)
- To avoid deadlocks (no note proposed by the generator satisfy specs), we allow to pick notes not proposed by the FO

To measure divergence, we use distances based on compression algorithms.
Compression and Complexity

\( K(y) \) is the **Kolmogorov complexity** of an object \( y \) is the length of the shortest compressed code to which it can be losslessly reduced.

We define a measure of creativity which is a variant of the Normalized Compression Distance:

**Creativity Divergence (CD)**

Approximates amount of information in \( y \) not in \( x \).

\[
CD(x, y) = \frac{K(y|x)}{K(y)}.
\]

\( K(y) \approx C(y) \) where

\[
C(y) = \text{length of } \text{compress}(y)
\]

for a compression algorithm \( \text{compress} \).
**Compression and Complexity**

\( K(y) \) is the **Kolmogorov complexity** of an object \( y \) is the length of the shortest compressed code to which it can be losslessly reduced.

We define a measure of creativity which is a variant of the Normalized Compression Distance:

**Creativity Divergence (CD)**

Approximates amount of information in \( y \) not in \( x \).

\[
CD(x, y) = \frac{K(y|x)}{K(y)}.
\]

\( K(y) \approx C(y) \) where

\[
C(y) = \text{length of } \text{compress}(y)
\]

for a compression algorithm \( \text{compress} \).
Compression and Complexity

$K(y)$ is the Kolmogorov complexity of an object $y$ is the length of the shortest compressed code to which it can be losslessly reduced.

We define a measure of creativity which is a variant of the Normalized Compression Distance:

**Creativity Divergence (CD)**

Approximates amount of information in $y$ not in $x$.

$$CD(x, y) = \frac{K(y|x)}{K(y)}.$$  

$K(y) \approx C(y)$ where

$$C(y) = \text{length of } \text{compress}(y)$$

for a compression algorithm $\text{compress}$. 
Evaluating Improvisations

Average creativity with respect to rhythm of 100 improvisations generated by supervise factor oracle.
Controlling Similarity via Compression-based Improvisation

Why?

- Dictionary based compression relies on pattern finding/matching.
- We can control creativity by controlling the difference between $C(\text{improv})$ and $C(\text{orig} + \text{improv})$.

Idea

Given an original training melody $\text{orig}$ and its compressed form we can synthesize an improvisation $\text{improv}$ such that

- $\text{improv}$ meets specifications
- $CD(\text{orig}, \text{improv})$ is in a specified range
Controlling Similarity via Compression-based Improvisation

Why?

▶ Dictionary based compression relies on pattern finding/matching.
▶ We can control creativity by controlling the difference between $C(\text{improv})$ and $C(\text{orig} + \text{improv})$.

Idea

Given an original training melody $\text{orig}$ and its compressed form we can synthesize an improvisation $\text{improv}$ such that

▶ $\text{improv}$ meets specifications
▶ $CD(\text{orig}, \text{improv})$ is in a specified range
Conclusion and Future Work

Summary

- We implemented a supervised musical improviser based on Factor oracles
- We proposed a divergence “creativity” measure
- We derived a new improvisation scheme controlling creativity divergence

Future work

- Further experiments with listeners exposed to our creativity measure
- Real-time (current work with Max/MSP and Ptolemy via OSC)
- Application of control improvisation to other domains
Conclusion and Future Work

Summary

▶ We implemented a supervised musical improviser based on Factor oracles
▶ We proposed a divergence “creativity” measure
▶ We derived a new improvisation scheme controlling creativity divergence

Future work

▶ Further experiments with listeners exposed to our creativity measure
▶ Real-time (current work with Max/MSP and Ptolemy via OSC)
▶ Application of control improvisation to other domains