AUGMENTING SOUND MOSAICING WITH DESCRIPTOR-DRIVEN TRANSFORMATION

Graham Coleman, Esteban Maestre, and Jordi Bonada, *

Music Technology Group,
Universitat Pompeu Fabra
Barcelona, Spain
firstname.lastname@upf.edu

ABSTRACT

We propose a strategy for integrating descriptor-driven transformation into mosaicing sound synthesis, in which samples are selected by taking into account potential distances in the transformed space. Target descriptors consisting of chroma, mel-spaced filter banks, and energy are modeled with respect to windowed bandlimited resampling and mel-spaced filters, and later corrected with gain. These transformations, however simple, allow some adaptation of textural sound material to musical contexts.

1. INTRODUCTION

New descriptors for describing sonic and music content are continually developed. There is a clear trend and demand for using these descriptors as controls for sound synthesis and transformation. For example, their use allows example-based processing, in which controls are derived directly from an example target.

Mosaicing is a type of example-based synthesis in which samples can be selected and assembled to reconstruct that target. In some cases, we would use mosiacing to transfer a sonic texture using the target as a kind of structure. However, if a texture has a small coverage of the overall descriptor space (kind of the idea of texture) then most matches with the target descriptors will be bad. This is especially true for relatively small source databases.

Descriptor-Driven transformation, in which input samples are transformed with respect to target descriptors, offers a possible strategy for overcoming this limitation. By expanding the source database into a space of potential transformed samples, we aim to allow better matches in the final mosaic.

1.1. Mosaicing

Many forms of mosaicing can be thought of as *matching* processes, in which each *unit* (sample) of a segmented target sequence is paired with a corresponding source unit, creating a sequence of retrieved units. In some systems the units are small uniform-length segments corresponding to frames, while other systems use higher-level divisions such as units segmented by score alignment, by onset detection, or by beat detection.

We call those *criteria* that help us decide between potential retrieved sequences. Criteria are commonly expressed as cost functions or probabilities. Perhaps the primal criteria are *target criteria*, which measure the quality of the match between paired target and source units. Systems that perform *basic* matching based on target criteria only include Soundmosaic [1], MATConcat [2], and that of Jehan [3].

Perhaps the first system to address *unit selection* by descriptor in musical sound synthesis would be an early version of Caterpillar [4]. It adapted the Viterbi algorithm for minimizing path cost from Concatenative Speech synthesis, where criteria are limited to target criteria and *concatenation costs* (*local* differences between source units in sequence). Further development [5] brought enhancements such as the inclusion of a variety of descriptors and specific modifications for musical speech synthesis.

* We thank Sašo Muševič for insightful comments on this work.

Zils and Pachet [6] propose *global* criteria on the retrieved sequences that seem to be incompatible with Viterbi path search. They introduce a modified formulation of mosaicing as a soft[1] Constraint Satisfaction Problem, and apply a local search method known as Adaptive Search [7], also later adopted in Caterpillar.

Hoffman et al [8] propose one of the first mosaicing systems to consider the *superposition* in time (mixing) of units. In our matching analogy, superposition means that each target unit may be matched with several source units with different weights or gains. Criteria are expressed in their system by a temporal Bayes net (an example of criteria expressed as probabilities). Gibbs sampling then provides an estimate of the best solution.

In this work, we implement several modes that can be seen as simplifications of the archetypal mosaicing system types above: a *basic* mode with only target criteria, a *sequence* mode that adds local sequence constraints, and a simplified *superposition* mode. All modes match and transform units based on perceptually motivated descriptor domains.

1.2. Descriptor-Driven Transformation

As in mosaicing, descriptor-driven transformation is also based on target criteria; but in this case, *input* units are transformed to resemble more the target. Most systems are composed of some fixed *process* controlled by parameters (possibly time-varying) that are chosen according to the inputs and target descriptors. Descriptor-driven transformations can thus be characterized by what kind of process they use, and how parameters are chosen.

With some processes, parameters can be chosen that correspond directly to target descriptors. This will be referred to as *direct modification*. The alternative is *parameter search* in which a range of parameters are evaluated against criteria. Search can be further characterized by whether information about the parameter space comes from predictive models of the process (model search) or black-box Synthesis-Analysis (SA) queries, or both.

One framework allowing for direct modification transformations is Analysis-Synthesis, in which one estimates parameters of the input for a synthesis process, modifies some of these parameters based on the target, and then resynthesizes the sound using the modified parameters. Many classic transformation techniques follow this approach, such as pitch and time stretch modifications by the phase vocoder [9], harmonic models such as SMS [10], and other source-filter based modifications [11].

Park et al. propose a large battery of transformations designated by the authors as Feature Modulation Synthesis [12]. These transformations are designed to modulate specific descriptors and use mostly a direct modification approach. By contrast, recent works based on model search include that of Coleman and Bonada [13], which uses spectral moments to drive resampling and filtering, and that of Caetano and Rodet [14], which interpolates spectral envelopes in a descriptor space of spectral moments in the interest of perceptual smoothness.

[1] In a standard CSP, all constraints must be met for a valid solution. In *soft* CSP, each constraint has an attached cost, and in the referenced work, this cost is further distributed among retrieved units.

The following works in descriptor-driven synthesis illustrate design alternatives. One recent system based on model search is that of Mintz [15], in which MPEG-7 instrumental descriptors are used to drive a harmonic+residual synthesis process. Linear programming is used for parameter selection. By contrast, systems proposed by Hoffman [16] and Yee-King and Roth [17] primarily use SA queries for measuring distance and genetic algorithms as the search method.

2. SYSTEM OVERVIEW

We outline a mosaicing system under development that is based on augmenting sample retrieval with model-based transformation. Sequences of target units that correspond to uniform-length frames are matched by choosing source units and transformation parameters that adapt them to target contexts.

The target and source units are uniformly segmented and analyzed from a target file and from one or more source files that define the source database or corpus. They are then described by descriptor vector sequences $b_{1..K}$ (target) and $s_{1..S}$ (source).

The system functionality is divided into *modes* with differing criteria and methods; such division can be seen as the result of mutual incompatibility between certain criteria and methods. In the basic and sequence modes, each target unit is matched with a single source unit n_k and a transformation parameter vector p_k. By contrast, in the superposition mode, multiple transformed source units (a weighted set of associated source indices and parameter vectors) are used to reconstruct a single target unit. In this case, we write the m^{th} element of the set of M_k matched with the k^{th} target unit as the tuple $u_k^m = (w_k^m, n_k^m, p_k^m)$ of weights w, source indices n, and parameter vectors p. For each mode, we attempt to minimize a sum of cost functions relevant to that particular mode, described in Section 2.3.

In general, we use models $\hat{t}(x, p)$ to predict descriptors of transformed units rather using the transformations $t(x, p)$ to test potential parameters. We do this to reduce the computational cost of queries in search, in exchange for some deviation of predicted descriptors from the true ones that we refer to as *model error*.

A process overview is given in Figure 1. Next, we will explain the descriptors used for control, the transformations available, and the criteria that allow us to choose between sequences. We will reserve detailed discussion of the models, a key aspect of our approach, for the following section.

2.1. Descriptors

The choice of descriptors (along with other aspects of criteria) will influence the resemblance of target to output in descriptor-driven systems. We have chosen a compact set which describe the following with regard to short time segments: the tonality, the timbre, and the loudness. (Short-time, hence any rhythmic properties must be described as sequences of these static descriptors.) In addition, source file and frame indices are included as *auxiliary* descriptors for retrieval and to support the contiguity criteria.

We measure the tonality of units with a *chroma* vector $\overrightarrow{ch}$, which is computed by assigning energy from the power spectrum to 36 chroma bins(3 per semitone) in a limited frequency range. In this application, chroma is treated as a relative distribution over energy, for which we divide the value in each bin by their sum.

We measure the shoft-time timbre with *mel-spaced filter banks* $\overrightarrow{fb}$, often used as a front-end to MFCCs. They are computed by assigning energy from the power spectrum to 40 mel-spaced bins of equal mel-width. As with chroma, the filter bank can be expressed as a weighted sum of spectral energy, which we treat equally as a normalized distribution.

We measure loudness with *energy* e, a low-level descriptor relevant in the time and spectral domains.

Chroma, timbre, and energy each exhibit *approximate linearity*, that is, the descriptors of a sum of units tend toward the sum of descriptors. This assumption is important for the superposition mode, and is validated alongside the models in Sec. 3.3.

2.2. Unit Transformations

A small initial set of three transformations is currently supported, based on the available models. *Bandlimited resampling* is a basic transformation that stretches or shrinks a signal in time by a factor L, and stretches or shrinks the spectrum by the inverse of that factor. In the chroma domain, it has an effect of tonally *transposing* by $\text{tp} = -\log_2 L$ octaves, to be detailed in Section 3.1.

That bandlimited resampling is a *length-changing* transformation complicates prediction. To limit the temporal scope of the composited unit, we window both before and after resampling.

Overlapping *mel-spaced triangular filters* modulated by a vector of gains $\overrightarrow{g}$ complement the timbre representation by allowing near-direct modulation of the timbre. However, due to limited resolution of the mel-spaced filter banks, only smooth filters are permitted, so costs are imposed according to a measure of smoothness (see Section 3.2 on filter smoothness).

Gain is used to directly adjust the energy of segments. Since chroma and mel-spaced filter banks are expressed as distributions over energy, they are invariant to gain.

2.3. Criteria

The criteria in this system are each expressed by cost functions that are weighted and summed for the total cost. Different criteria are available according to mode.

We classify criteria into *target criteria*, which concern the relationship between matched units, and *sequence criteria*, which concern relationships in the retrieved sequence and parameter sequence. Furthermore we designate *local* those that concern adjacent or small groups of adjacent units.

Distances Distances are functions $d(x_1, x_2)$ that dictate unit similarity in descriptor space for target and continuity criteria. In general we are free to use any kind of function or vector norm. However, the superposition mode is currently limited by the regularized least-squares method to using the l_2 norm on chroma and timbre. In all modes, we can weight the relative importance of chroma, timbre, and energy with weights λ_{ch}, λ_{fb}, and λ_e.

Target distance for basic and sequence modes In the absence of superposition we simply use the distance between corresponding target and retrieved units:

$$\text{TD}(n_{1..K}, p_{1..K}) = \sum_k^B d(t(s_{n_k}, p_k), b_k) \tag{1}$$

Target distance for superposition Recall that the superposition mode matches multiple source units (and multiple parameters and associated weights) to each target unit. We therefore must measure how well the combination, in this case the weighted sum, matches the target unit. The adapted cost function is as follows:

$$\text{TDSup}(u_{1..K}) = \sum_k^K d\left(\sum_m^{M_k} w_k^m t(s_{n_k^m}, p_k^m), b_k\right) \tag{2}$$

Note we are summing the descriptors of superposed units, which amounts to an assumption of linearity.

Retrieval and parameter criteria We may prefer *a priori* certain subsets of source units or of the parameter space. For example, using time-varying weights on source files such as in Loopmash [18] allow transitioning between different material. In terms of parameters, we may wish to avoid parameters with high model error (e.g. deriving cost from incurred model error). For these reasons we impose unit costs UC and parameter costs PC.

The parameter costs currently imposed for the system are a cost related to the tonal transposition: $\text{PC}_{tp} = \lambda_{tp} \left| \frac{tp}{tp_{\max}} \right|$, and a cost related to the filter roughness: $\text{PC}_{fg} = \lambda_f \frac{|D\overrightarrow{g}_{db}|_2^2}{R_{\max}}$, where D is the local difference between adjacent filter gains in $_{db}$ and $R_{\max}$ is the maximum roughness in a list of cantidates (Section 4).

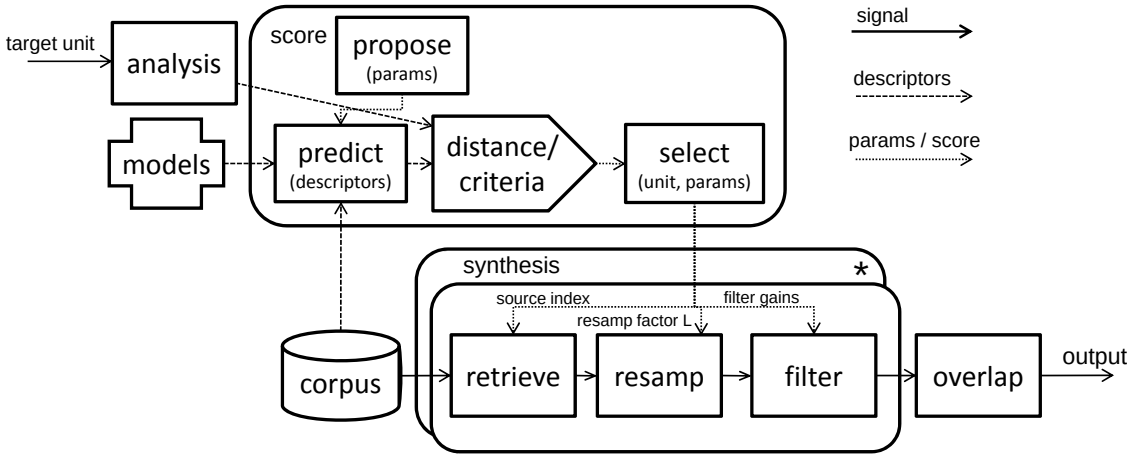

Figure 1: An visual overview of the system. An analysis frontend extracts target descriptors, then used to select units and transformation parameters according to distance from the target and other criteria. The results are then synthesized by retrieving the selected units, transforming them with the selected parameters, and overlapped using synthesis windows. * multiple synthesis blocks denote separate retrieval and transformation for superposed units.

Local sequence criteria We refer to local changes in retrieved descriptors or parameters as *continuity*, and the retrieval system to choose subsequences of the source sequences as *contiguity*. We choose a contiguity cost function CgC that allows (with a light penalty) stepping a few units forward, while heavily penalizing *jumps* regardless of the size of the jump. Likewise, for subsequences of contiguous units made of steps only, we impose a similar *parameter continuity* cost PCn on changes in transposition tp.

Sparsity in superposition To better preserve source qualities, one may prefer a small set of matched elements. Set size can be measured in a vector function of w known as sparsity. Perhaps the primal sparsity measure is known as l_0 ($|w|_0$), the number of nonzero elements in w. Due to computational intractability of the optimization procedure, it is often substituted by the l_1 norm, ie $|w|_1 = \sum_i |w_i|$ the absolute sum of all of the weights.

Total costs for the modes can be summarized as follows:

$$\text{TCBas} = \text{TD} + \text{UC} + \text{PC}$$
$$\text{TCSeq} = \text{TD} + \text{UC} + \text{PC} + \text{CgC} + \text{PCn}$$
$$\text{TCSup} = \text{TDSup} + \text{UC} + \text{PC} + \sum \lambda |w_k|_1.$$

3. TRANSFORMATION MODELS

A model is a function $\hat{t} : X \times P \rightarrow Y$ that predicts output descriptors from Y using input descriptors from X and parameters from P for the transformation t. To fully predict target distance, we must predict all target descriptors with non-zero weight in d.

3.1. Resampling

Bandlimited resampling linearly scales locations in frequency in cases where the output frequency is below f_N, the Nyquist frequency. That is, a sinusoid previously at frequency f will be relocated to $\frac{f}{L}$ in the output. To predict spectral descriptors under resampling, we map this scale relationship to descriptor domains.

Chroma The usual mapping from a frequency f to K chroma bins can be described as: $k_f = \text{mod}(K \log_2 \frac{f}{f_0}, K)$ given a reference frequency f_0. By substituting the scaled frequency $\frac{f}{L}$ from resampling in chroma bin assignment, we see that energy remaining in the chroma region is shifted by $-K \log_2 L$ chroma bins. This can be stated by the circular shift function:

$$\hat{t}(\overrightarrow{ch}, L) = \text{shift}_\circ(\overrightarrow{ch}, -K \log_2 L) \quad (3)$$

Mel-spaced filter bank Using our mapping to mel bands we can linearly interpolate (denoted with square brackets) at the center frequencies of the filters f_i to get a prediction.

$$\hat{t}_{f_i}(\overrightarrow{fb}, L) = \overrightarrow{fb}\left[m\left(\frac{f}{L}\right)\right] \quad (4)$$

Energy Energy changes under the windowed resampling vary per unit according to both the spectral energy distribution and the temporal energy distribution. Rather than trying to account for the temporal and spectral effects separately, we find it simpler to use a piecewise linear regression based on SA queries. That is, in the Analysis phase we estimate 3 energy ratios: SAR_0, the double windowed unit with no resampling, SAR_{-1}, the same unit downsampled by an octave, and SAR_1, the same unit upsampled by an octave (all in relation to the energy $e(x) = \|w_a \cdot x\|^2$):

$$\text{SAR}_0(x) = \|w_a \cdot_c (w_s \cdot x)\|^2 / e(x) \quad (5)$$
$$\text{SAR}_{-1}(x) = \|w_a \cdot_c \text{resamp}\,(w_s \cdot x, 0.5)\|^2 / e(x) \quad (6)$$
$$\text{SAR}_1(x) = \|w_a \cdot_c \text{resamp}(w_s \cdot x, 2)\|^2 / e(x) \quad (7)$$

where w_a and w_s are analysis and synthesis windows, $\|x\|^2$ denotes temporal energy of x, $w \cdot x$ denotes pointwise multiplication (windowing), and $w \cdot_c x$ denotes centered pointwise multiplication where x is cut or zero-padded symmetrically to match the length of w. To predict the energy with an arbitrary resampling factor, we linearly interpolate between known samples $\{\text{SAR}_{-1,0,1}\}$:

$$\hat{t}(e, \text{SAR}, L) = e \cdot \text{SAR}\,[-\log_2 L] \quad (8)$$

This approach logically extends to piecewise linear interpolation of larger grids of SA queries. One difference between this model and the vector models for chroma and timbre is that this model does not generalize over units—each unit can be thought of as having its own model. In practice we can think of the queries as additional descriptors. Since a small fixed number of queries can be performed for each unit at analysis time, SA queries are unnecessary during search time.

3.2. Filters

The timbre representation consists of filter banks of spectral energy, the filters apply gain directly to these, so the i^{th} band can be predicted as:

$$\hat{t}(fb_i, g_i) = fb_i \sqrt{g_i} \quad (9)$$

The accuracy of this prediction can be related to the smoothness or roughness of the filter and is treated below. The effect of filtering on the chroma is not modeled, but the effect in terms of additional error is measured empirically.

3.3. Accuracy of Models

For a rough comparison of the accuracy of the various models, we test them against SA queries on a small database of at least 1000

	Chroma	Timbre	Energy
Resampling (avg)	0.123	0.362	0.155
src1 (electronic)	0.087	0.310	0.112
src2 (vocal formants)	0.106	0.206	0.123
src3 (singing voice)	0.115	0.409	0.140
src4 (commercial pop)	0.143	0.436	0.122
src5 (impact sounds)	0.177	0.468	0.327
src6 (orchestra)	0.087	0.353	0.113
Filtering	0.228	0.206	5.882
Sum	0.106	0.263	0.189

Table 1: Accuracy of resampling models (l_1, l_1, db) compared with the error measured when assuming linearity of descriptors.

units from various sources.

To compare the vector descriptors of chroma and timbre, a scaled l_1 vector norm is used: $d(v,z) = 0.5\sum_i |v_i - z_i|$ where the constant scales the maximum distance over distributions to unity (when two vectors have completely nonoverlapping support). To compare the energy, we take the absolute value in decibels between the true value and the prediction: $d(e_t, e_p) = \left|10\log_{10}\frac{e_p}{e_t}\right|$ Table 1 shows the accurary of the predictors for resampling along with some of the other operations.

4. UNIT AND PARAMETER SELECTION

For all modes, we predict a matrix under a grid of resampling parameters and use the predicted descriptors in the unit and parameter selection, procedures for which are sketched below:

Basic In the basic mode, we loop over target units and match source units to them independent of sequence.

When filtering is enabled, we can generate smooth filter parameters by considering a tradeoff between deviation (in decibels) from the ideal filter $f_{\text{ideal}} = \sqrt{\frac{b_{\text{fb}}}{s_{\text{fb}}}}$ and the smoothness of the filter. We use a technique known as quadradic smoothing in which the quantity $|f_{\text{ideal, db}} - \overrightarrow{g}_{db}|_2^2 + \lambda_f |D\overrightarrow{g}_{db}|_2^2$ is minimized by solving a linear system (p312 Boyd [19]): $\overrightarrow{g}_{db} = (I + \lambda_f D^T D)^{-1} f_{\text{ideal, db}}$, giving us the smoothest and closest filter to the ideal. Using both the target distance and the parameter cost based on filter roughness we can determine the best candidates based on potential filtering.

Sequence One mode gives an approximate solution for the *target distance* criteria, a *contiguity criteria* that allows the retrieved unit to skip forward according to the original source without a high cost, and a *continuity* criteria on the transformation parameters.

We build a non-optimal relaxation of the original problem, by first programming a path (Viterbi) through retrieved unit space using the distance for the *ideal* transposition, then using that path to find a path through transformation space.

Superposition For creating a mosaic with superposed source units but no sequence criteria, we can solve a regularized least-squares problem as in *basis pursuit* (Chen et al [20]): $\min_w ||Aw - b||_2^2 + \lambda_{sp}||w||_1$ using the l1_ls [21] package. We use the predicted descriptor matrix as the problem matrix A, and weight them by their limits. Next, we can solve the problem with a given sparsity parameter λ_{sp}, which gives a set of gains on the units.

5. CONCLUSION

We have prepared sound examples to demonstrate the algorithms that can be found at:
`http://www.dtic.upf.edu/~gcoleman/mosaic10/`.

First, objective (descriptors) and subjective (listening tests) are needed to validate the work. Second, finding ways to avoid exhaustive evaluation or choosing subsets of the prediction matrix (in superposition mode) could likely speed up the computation. Finally, a way to combine sequence constraints with superposition mosaicing could possibly better preserve timbre characteristics of source material while matching a polyphonic target.

6. REFERENCES

[1] S. Hazel, "Soundmosaic," Available at http://awesame.org /soundmosaic/, accessed Feb. 03, 2010.

[2] B. L Sturm, "MATConcat: an application for exploring concatenative sound synthesis using MATLAB," in *Proceedings of DAFx04*, Naples, Italy, Oct. 5–8 2004.

[3] T. Jehan, "Event-synchronous music analysis/synthesis," in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-04)*, Naples, Italy, Oct. 5–8 2004.

[4] D. Schwarz, "A system for data-driven concatenative sound synthesis," in *Proceedings of Digital Audio Effects (DAFx 2000)*, Verona, Italy, Dec. 7–9 2000.

[5] D. Schwarz, *Data-Driven Concatenative Sound Synthesis*, Ph.D. thesis, Ircam-Centre Pompidou, 2004.

[6] A. Zils and F. Pachet, "Musical mosaicing," in *Proceedings of DAFx01*, Limerick, Ireland, Dec. 6–8 2001.

[7] P. Codognet and D. Diaz, "Yet another local search method for constraint solving," in *AAAI Fall Symposium on Using Uncertainty within Computation*, North Falmouth, Massachusetts, Nov. 2-4 2001.

[8] M. Hoffman, P. Cook, and D. Blei, "Bayesian spectral matching: Turning Young MC into MC Hammer via MCMC sampling," in *Proceedings of ICMC 2009*, Montreal, Canada, Aug. 16–21 2009.

[9] M. Dolson, "The phase vocoder: A tutorial," *Computer Music Journal*, vol. 10, no. 4, pp. 14–27, 1986.

[10] X. Serra, *A System for Sound Analysis/Transformation/Synthesis based on a Deterministic plus Stochastic Decomposition*, Ph.D. thesis, Stanford University, 1989.

[11] V. Verfaille and P. Depalle, "Adaptive effects based on STFT, using a Source-Filter model," *Proceedings of DAFx'04*, pp. 296–301, 2004.

[12] T. H Park and Z. Li, "Not just prettier: FMS marches on," in *Proceedings of ICMC2009*, Montreal, Canada, Aug. 16–21 2009.

[13] G. Coleman and J. Bonada, "Sound transformation by descriptor using an analytic domain," in *Proceedings of DAFx 2008*, Espoo, Finland, Sept. 1–4 2008.

[14] M. Caetano and X. Rodet, "Evolutionary spectral envelope morphing by spectral shape descriptors," in *Proceedings of ICMC2009*, Montreal, Canada, Aug. 16–21 2009.

[15] D. Mintz, "Toward timbral synthesis: a new method for synthesizing sound based on timbre description schemes," M.S. thesis, University of California, Santa Barbara, 2007.

[16] M. Hoffman and P. Cook, "The Featsynth framework for feature-based synthesis: Design and applications," in *Proc. ICMC-07*, Copenhagen, Denmark, Aug. 27–31 2007.

[17] M. Yee-King and M. Roth, "Synthbot: An unsupervised software synthesizer programmer," in *Proceedings of ICMC-08*, Belfast, N. Ireland, Aug. 24–29 2008.

[18] Steinberg, "Loopmash (Cubase)," Commercial software, Jan. 2009.

[19] S. P Boyd and L. Vandenberghe, *Convex optimization*, Cambridge Univ Pr, 2004, Available at http://www.stanford.edu/~boyd/cvxbook/.

[20] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[21] K. Koh, S. Kim, and S. Boyd, "l1_ls: Simple matlab solver for l1-regularized least squares problems," Available at http://www.stanford.edu/~boyd/l1_ls/, 2008.