IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 26, 2019, accepted September 12, 2019, date of publication September 26, 2019, date of current version October 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2944083

Music Visualization Based on Spherical Projection With Adjustable Metrics

OMAR LOPEZ-RINCON AND OLEG STAROSTENKO
Department of Computing, Electronics and Mechatronics, Universidad de las Américas Puebla, Cholula 72810, Mexico
Corresponding author: Oleg Starostenko (oleg.starostenko@udlap.mx)

This work was supported in part by the Universidad de las Américas, Puebla, Mexico.

ABSTRACT Development of techniques for music visualization is important and still open problem in analysis and creation of the quantitative profiles of single or multiple compositions, which could be used as required constraints in music generation or music classification processes. When generating creative data with no objective function, it is hard to select or to find appropriate measurable features. This paper proposes a method to normalize data in MIDI files by 12 dimensional vector descriptors extracted from tonality as well as a novel technique for dimensionality reduction and visualization of extracted music data by 3D projections is discussed. Employing a non-overlapping sliding window through the composition, the harmonic features are found in a music piece. Then a self-similarity matrix is computed using distance metrics to analyze and project the resulting 3D feature vectors. Three dimensional projection creates a quantitative profile of a composition, which correlates the tone similarities along the music piece. The dimensionality reduction is compared with well-known autoencoder. Conducted tests show that our method preserves up to 90% of original data in the projection of reduced dimension. The advantages of the proposed method consist in a novel technique that provides interactive visualization and dynamically adjusts different metrics to observe the behavior of data during music information retrieval and recognition.

INDEX TERMS High dimensional visualization, music composition profile, music data analysis, spherical projection.

I. INTRODUCTION

Music visualization is used to read, write, and edit music compositions in the creation context. This kind of visualization can be limited only by the precision of feature selection and the performance of data extraction tool. Therefore, based on modern techniques of artificial intelligence, data projection and dimensionality reduction used for downgrading the computational resources are needed for appropriate selection and analysis of specific features found in data. The visualization sometimes needs flexibility to be used with different quantitative metrics to compare them between each other, when observing their related behavior either based on perception or interpretation of analyzed data.

Contrary to common dimensionality reduction methods, it is important to control selected redundancy due to the nature of music data properties found in tonality, rhythm, speed, etc. That is the reason, why traditional methods for data projection do not provide techniques to visualize and understand features, which identify genres, style or authors. The study of tonality and rhythmic descriptors can help organizing and finding relevant data behavior. Frequently used patterns in data could be used for music classification and automatic music generation due to its quantitative representation precisely measured by specifying metrics of different elements in music composition.

There exist numerous techniques for unsupervised or semi-supervised machine learning however, they are not practical to study data from music pieces. To make use of supervised learning methods as deep learning, we need big amount of labeled data but, making annotations by hand, it is not so practical. Classification systems like key signature usually are genre bias and thus, it favors one tonal key over another according to the specific style [1]. As result, these systems only work accordingly to a specific genre for which they were designed.

There are different methods for dimensionality reduction and data visualization by multidimensional vectors, which are computed finding redundancy between them.

For this purpose, one of the frequently used techniques is Principal Component Analysis (PCA), which is based on linear transformations and matrix correlations [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Feng Xia.

Other methods provide data projection applying non-linear analysis using neural networks like autoencoders [3]. They employ encoded projections of non-linear related features however, the user can not specify which dimensions, redundancy, or metrics is better to take. In recent updates of the method proposed by Lopez *et al.* [4], there are some approaches to use constraints controlling the latent space represented at the projection of reduced dimensionality. While well-known dimensional reduction techniques as usual are based just on the redundancy of dimensions, the proposed method provides original representation for all possible combinations of tones using dimensionality reduction from required 12 dimensional harmonic descriptors into 3 dimensional vectors [2], [4], [5]. To visualize them a novel projection approach has been developed, which is adjustable to different metrics by changing only one step of the proposed method.

In music information retrieval (MIR) there are different techniques used to study elements of data, for example, to classify music compositions by specific features, which are evaluated by different qualitative and quantitative metrics [5], [6]. Usually, well-known techniques employ limited number of metrics and frequently, they cannot be directly adopted for evaluation of features extracted from data. In order to dispose with multi-faceted study of data properties during music information retrieval, the proposed method can select and fit dynamically multiple metrics providing more accurate data acquisition and processing. We present an example of visualization using two different metrics: Euclidean and Hamiltonian to compare and evaluate their usefulness.

Represented in a MIDI file features such as rhythm or tones can be extracted at a specific moment within a window of time. The proposed approach allows user to test different distance metrics for harmony or rhythm. The distances between vectors, which encode harmony, are used to generate a Self-Similarity Matrix (SSM) and then it is used to project the data in a lower dimensional space to observe the metrics behavior. The projection can also be employed for visualizing results of music data analysis as well as for reducing load processing of generated MIDI file abstractions. The absence of this type of tool and the lack of formal scientific reports applied to field of automatic music generation and edition motivate this research.

The aim of the proposed method is to create a dynamic and adjustable instrument to be used for music information retrieval, which can covers studies in musicology, signal processing, information interpretation and recognition among others. Because of a method extracts quantitative features of music compositions, it can be used for analysis, indexing, classification, recognition and automatic generation of classic or modern music. Recently, this field has a practical need in musical services such as iTunes, Spotify, Pandora, Magenta-Google, etc., [5]–[8].

The rest of this paper is organized as follow: in Section II the related works are explained and the most common methods for data visualization are reviewed. In Section III the proposed methodology is presented with formal description of specific examples to determine the processes and used parameters as well as the obtained experimental results and their evaluation are resumed in Section IV. Finally, Section V presents conclusions and future work of the research progress.

II. RELATED WORKS

One of the goals of dimensionality reduction is the capability to interpret data visually. Another use of projection of data is the elimination of redundancy in information, which helps intelligent systems to learn and find patterns [2]. Some examples of techniques used for reduction of dimensionality are the t-distributed stochastic neighbor embedding (t-SNE) [9], [10], the autoencoder [3], [4] and specific for music analysis the spiral array approach for computational modeling of tonality [11].

The method of t-SNE exploits a clusterization approach in high dimensional data and computes thresholds to determine the closeness of the nodes between each other to classify them as *far* or *close*. The nodes determined as *far* are pushed away in clusters to project the clusterization behavior and authors apply the method of PCA (principal component analysis) to reduce the dimensional redundancy [3]. The approach performs the lower projection by fitting the nodes given in the classification that are *close* within each other to form clusters. However, the principal disadvantage of this method is reducing only redundancy found by PCA and classified by using predefined thresholds. The method is not capable of changing the metrics of closeness other than the one given by the PCA. Thus, it is not directly applicable for music analysis as well.

The spiral array model based on the Tonnetz theory is designed specifically for music visualization [12]. This theory consists in mapping and modeling the structures like chords, keys, and tension between moments of the composition. Each tone (*C, C#, D, D#, E, F, F#, G, G#, A, A#, B*) is represented by a 3 dimensional point inside spiral array. The method splits the composition in overlapped windows of time and averages the values of the assigned points to determine the position inside the spiral. Unfortunately, the approach only considers major and minor key signatures and it uses a fixed window for any composition. It is not entirely adaptable to different keys or changes of speed like beats per minute.

The autoencoder is one of the unsupervised learning techniques on neural networks used to compress data. It contains a hidden layer known as bottleneck, which connects two parts of a neural network called encoder and decoder.

They are layers of neural networks that can be fully-connected layers, convolutional layers, or variations of them. Although it can be used as deep-learning approach for reducing dimensionality, the compressor does not provide better performance than standard compression algorithms like JPEG [13]. The autoencoder is used for specific tasks such as cleansing data for example, denoising or gap filling

in images using its more complex versions [14]. In non-linear methods with no-supervised learning area the variational autoencoder is frequently used, which is a version of the autoencoder, where the bottleneck of the network learns a statistical representation of relations of the data [15]. This method encodes so called latent space, which represents the lower dimensionality projection. The area described by latent space allows to select a point in continuous 2D or 3D space, which is inside a region from cluster of features representing higher dimensionality. The decoder is capable to morph the features between the points of the learned examples.

In the visualization of symbolic music there are other methods based on compression. They are used particularly for reducing repeated figures during visualization of the data extracted from musical piece. One of them is a compression method COSIATEC [16] exploited in pattern recognition. It provides finding shapes formed by onsets of the notes. The patterns can have several sizes of shapes and shapes inside the shapes. The heuristics of the algorithm to choose a size of the preferred shapes is given by prior analysis of points found as roots of the dictionary as well as it selects the ones that give the higher rate of compression.

Another representation based on the Tonnetz theory [12] is the Isochords approach [17], where authors try to represent the dissonance and consonance of the structures of music in two dimensional isometric triangles, which describes relations between triads of notes. Next approach for visualization of raw audio using Self Similarity Matrix (SSM) is introduced by authors in [18]. The visualization shows autocorrelation of segments in time windows manually selected by user. Following this research, we find the approach of the MIDIVIS [19], where authors use colors for establishing the relationships of unigrams into fixed size time windows. The structure of pitch and their length is projected in two dimensions, which can be visualized as patterns in a self-similarity matrix. Since the instruments are dimensional descriptors and user can decide, which instrument to visualize. The mentioned approach is one of the frequently used music visualization tool however, its disadvantages are the fixed size time window and limited number of colors for pattern representation that might lead to overlapping similarities.

The ImproViz technique [20] also uses a fixed width of window, processes six points defined in six beats of the composition and plots a point at every half value of a beat. Then the method draws a connecting line between points to represent a shape. In the harmonic graphing it uses an approach based on the musical genre features, which aimed particularly for jazz. Another interactive tool is a Particle system proposed by Fonteles, to visualize the tones with emitters that are placed in 3D space representing an instrument [21]. So, the volume is represented by size of the particle and pitch by color of the particle. A visualization technique of music features (like tension) is presented by the Tension Ribbons method [22]. In this method the Spiral Array is used to compute different metrics from a music composition. It averages notes found in a fixed window and compares the distances from one window to the next one. This is considered as the tension changed in time and used to find the key of a composition. In addition, in [23] a tonal visualizer has been designed to visualize the possible keys or interactions with different moments of a composition by combining different resolutions of time windows. A visualization of the notes with a color map assigned by the circle of fifths is discussed in [24]. To find the color of a note, it is mapped by a vector in the center of the circle of fifths and then superimposed over the tone represented by vector direction, which represents the hue of a color. At the 3 dimensional projection, the length of vector is proportional to the loudest tone in the given pitch class. When several notes occur at the same time, a color is computed by vector addition. So, the resulting vector direction determines the hue and its length defines the normalized loudness.

In the field of affective computing there exists a quite subjective method of visualizing music compositions by selecting photos inspired by listening to the piece [25]. This is done using previously gathered dataset from existing pictures with their emotional description to train a model and to recognize the emotional class in new pictures by matching features found in music such as loudness and spectrum. However, the approach is based on raw signal processing and cannot be directly applied to visualization of symbolic music compositions. Next interesting technique of visualization of the classical musical tradition reported in [26] consists in processing only the tonal key of different compositions of 33 authors to visualize the development of style and period of the music as well as authors preference, when choosing major or minor tonal key for a composition.

Finally, one representative nonlinear approach that analyzes how music can be viewed by its entropy content is discussed in [27]. Authors propose to extract features found in style, genre and composer encoded by binary symbols inside music files and then they are used as a raw signal on the input. Selected symbols were analyzed to find their entropy and frequency determining their importance to be used as a descriptor of the particular music composition.

In Fig. 1 the classification of relevant linear and non-linear music visualization approaches is resumed with respect to symbolic (MIDI) or raw audio inputs.

The performance of analyzed approaches directly depends on accuracy, complexity and speed of retrieval of multiple features found in the input data. Linear methods are faster than nonlinear as well as processing symbolic data is faster than raw signal data. Recent methods mostly are based on algorithms that rely on neural networks or linear algorithms with specific sequence of steps. None of them is designed to test different metrics to analyze the quality of resulted visualization process.

Unfortunately, most reported approaches do not provide any quantitative computational oriented analysis of the music visualization performance. They cannot be used for comparison of efficiency with the proposed method. Only authors of autoencoder based on t-SNE and Spiral Array try to

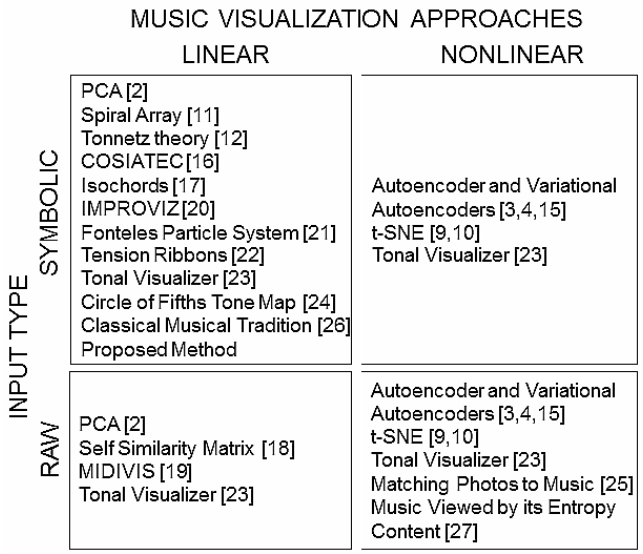


FIGURE 1. Classification of music visualization methods with respect to different input data types.

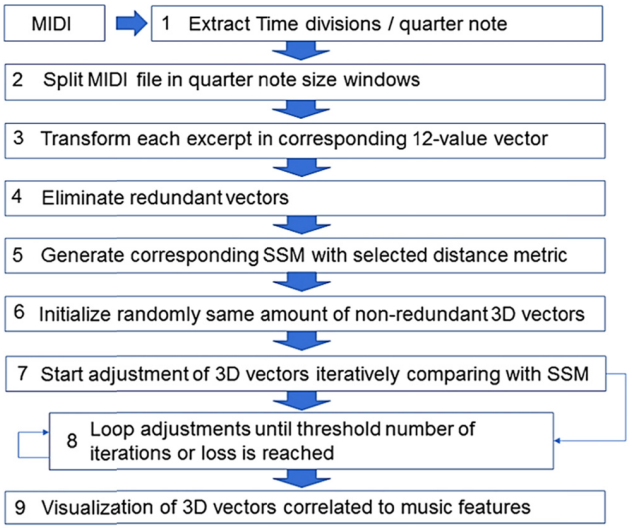


FIGURE 2. The flow chart of the principal steps of the proposed method for dimensionality reduction and projection of extracted data from music compositions.

introduce quantitative evaluation of music visualization process [3], [4], [9], [10], [15]. Because of the autoencoder [3] is the state-of-the-art frequently referenced method therefore, it also will be used for performance analysis of the proposed approach.

III. PROPOSED METHOD

The proposed method for music visualization consists in splitting a MIDI file in quarter note size windows extracting 12 dimensional vectors of music features for next dimensionality reduction and projection of extracted data to 3D space, which serve for analysis of data behavior based on different self-similarity matrixes (SSM). The flow chart of the proposed method is depicted in Fig. 2.

The MIDI file format defines the moments in which the notes of music piece occur. It has two types of sub-structures called *header chunk* and *track chunk*. The standard MIDI file (*SMF*) consists of one header chunk and one or several track chunks as follows [28].

$SMF = <HEADER\_CHUNK> + <TRACK\_CHUNK> + <TRACK\_CHUNK> + \ldots$

Both types of chunks have literal identifying string encoded by ASCII of 4 bytes long. Particularly, the string *MThd* is an identifier for header and *MTrk* is for a track chunk. They indicate that it is in fact a MIDI file. The header chunk is 14 bytes long with the following format.

HEADER_CHUNK =<*MThd* 4 *bytes*> <*Header_length 4bytes*> <*MIDI_format 2bytes*> <*Number_of_tracks* 2 *bytes*> <*Time_divisions/quarter_note* 2 *bytes*>

Four bytes located after string *MThd* represent the length of the rest of MIDI header. The following 2 bytes define MIDI file format labeled as 0, 1 or 2 for single-track file, multiple-track file or multiple song file, respectively. The next 2 bytes are the number of track chunks appended to the header and the last 2 bytes represent the time divisions/quarter note of the MIDI file. The value found in time divisions/quarter note is used as a window size for splitting MIDI file. It is declared only once and the rest of the notes (track events) in the file are related to this value. The track chunk has the following format.

TRACK_CHUNK = <*MTrk* 4 *bytes*> <*Track_length 4bytes*> <*Track_event variable*> <*Track_event variable*>...

A track event consists of *v_time* (a delta time since the last event) and one of three event descriptors: *midi_event* (MIDI channel message such as note-on or note-off), *meta_event* (SMF meta event) and *sysex_event* (SMF system exclusive event). Each event includes the descriptions of the music notes and their properties are used by the proposed method for generation of feature vectors. Particularly, the onset, pitch, and duration of each note are the principal elements to be processed.

The measures and the speed of the music piece are described in relation to the *quarter-note* duration in ticks and along with the event messages in the file. The *crotchet* or quarter-note is found at the beginning of the MIDI file.

This parameter is set only once, so every message of the presented events in the file is related to this length. The size property of ticks of the quarter-note is used in heuristics to compute splitting profile of a song. According to Fig. 2 the entire music piece is subdivided into the measures declared inside the MIDI file structure and then each measure is again split according to size of the crotchet. The composition ends up as several pieces coming from a non-overlapping sliding window equal to the quarter-note size division as instant.

A 12 dimensional vector is filled with the information found in every instant taken by each window. Each vector is correlated to all the possible tones of equal tempered music *[C, C#, D, D#, E, F, F#, G, G#, A, A#, B]* so, a vector represents which notes are found in every slice of

the composition. The vector has a binary representation and if a note is present in the window, then the position that corresponds to a tone has a value of one. For example, if a window has a sequence of *C, E, G,* the resulting vector is defined as *[1,0,0,0,1,0,0,1,0,0,0,0]*.

After slicing the complete song into quarter-notes size windows, the SSM of distances d_i between all the vectors can be generated. Each vector of the composition is compared to the next one assuming distance as selected metrics. Every extracted vector is inserted into a hash table to find only the different combinations of harmonic tones. SSM is filled with each of the distances between tonal vectors, where each row is cross-related to a column with all the other tonal vectors as show in (1)

$$S = \begin{bmatrix} d_0 & \cdots & d_n \\ \vdots & \cdots & \vdots \\ d_n & \cdots & d_n \end{bmatrix}, \tag{1}$$

where S corresponds to the SSM, each element is a distance value d_i, and the n is the total number of slices of the composition.

The distances between vectors are computed among each row vector $\hat{X}$ and column $\hat{Y}$ of the tonal vector matrix found in the slices of a composition. The similarity distances between two binary vectors are computed using the Hamming distance [29] as shown in (2).

$$H = \sum_{k=1}^{n} \left| \hat{X}_k - \hat{Y}_k \right| \tag{2}$$

To reduce dimensionality, we loop through all the vectors and compare the distances to the actual ones in the SSM. In the lower dimension we take $\hat{V}_i$ and compare the distance to $\hat{V}_j$. Then this value is compared to corresponding distance in SSM of the higher dimension according to (3)

$$|d\left(\hat{V}_i, \hat{V}_j\right) - D_{ij}| > t, \tag{3}$$

where $\hat{V}_i$ and $\hat{V}_j$ are the vectors in the lower dimension and D_{ij} are the distances in the SSM in 3D space. If the difference is higher than a threshold t, we adjust the position of the vector according to (4) with a given increment α until the distances will be close to the ones in SSM.

$$\hat{V}_i = \left(\alpha * \hat{V}_i\right) + ((1 - \alpha) * \hat{V}_j) \tag{4}$$

The adjustment $\hat{ad}_{ij}$ of the representing vectors is normalized to the higher magnitude. To normalize the distance, the result of the selected metric is divided by the maximum possible value. For example, if the Euclidean distance is selected and the vector is a binary 12 dimensional set, then it is dived by the square root of 12, which is the maximum possible value. The distance of the representation in the lower dimension is also normalized. The total amount of possible combinations in the harmonic space of 12 tones is equal to $2^{12} = 4096$. When a composition is analyzed, $\hat{ad}_{ij}$ is scaled to the proportion of all combinations by dividing the total amount of vectors found in music composition by *4096*.

To visualize and represent amount of computed vectors in music piece, the adjustment is multiplied by a count of repetitions of the vector divided by the total amount of vectors found in the subdivisions as shown in (5)

$$\hat{ad}_{ij} = \frac{1}{4096} * d\left(\hat{V}_i, \hat{V}_j\right) * \frac{a_i}{V}, \tag{5}$$

where $\hat{V}_i$, $\hat{V}_j$ are two current evaluated vectors, a_i is the total count of the vectors found along the composition, and V is the total amount of found vectors. We can adjust a stop criteria of the dimensionality reduction by setting a specific number of cycles or by using a regression error metrics as the mean square error (MSE) [30] using (6).

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(|d\left(\hat{V}_i, \hat{V}_j\right) - D_{ij}\right)^2 \tag{6}$$

The subtraction between the original SSM and its new version of reduced dimensionality converges over time.

Another possible stop criterion can be formulated from the adjustment of vectors in 3D space, where a small change between the positions of vectors can be counted and used as threshold. The convergence of optimization of the projection as a fitness function is achieved, when the adjustment is decreased under previously set threshold. The size of steps in the increment of α value can create a wobbling motion if the adjustment is too high. When this happens, the optimization starts to jump around the threshold and it could get too many numbers of cycles to converge. This means that adjustment needs to be reduced until changes are smooth enough to get the result of the fitness of the projection and the error of projection is also minimized. The softer stop criterion for defining the required number of cycles is based on computing the adjustment average error (AAE) using (7), where each adjustment a_i in the batch of an epoch is added up and divided by the total amount of epochs (p).

$$AAE = \frac{1}{p} \sum_{i=0}^{n} a_i \tag{7}$$

In many conducted tests both MSE and AAE errors have been computed and compared using empirically defined threshold t set to 0.67, which reached a total count of 300 epochs.

The difference in the behavior of errors is plotted at each epoch using both (6) and (7) in Fig. 3. We can observe how MSE and AAE metrics had different conducts. The decreasing-only behavior is a best fit to set threshold and avoid looping until convergence. At the end of run, when the projection converges, MSE metric still shows the total error missing from the higher dimension.

From the example, where the convergence threshold was set to 0.67, the MSE does not exceed 10% comparing the distances between vectors from the higher and lower dimensions. It means that up to 90% of the original distances of the 12 dimensional tonal vectors are preserved in the 3 dimensional space using MSE in our proposed method.

The obtained results presented in Fig. 3 show the reduction of the AAE with increment of the loops for adjustment of 3D vector positions. Using AAE in our methods it is possible

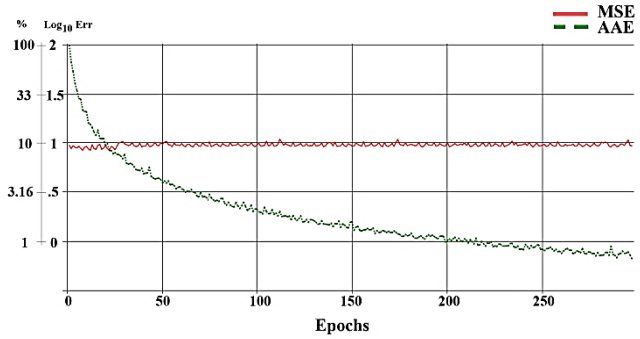


FIGURE 3. The upper line shows the mean square error (MSE) and the lower line represents the average adjustment error (AAE) with the decreasing only behavior. The y axis represents the percentage and logarithmic values of errors and the x axis represents the number or epochs.

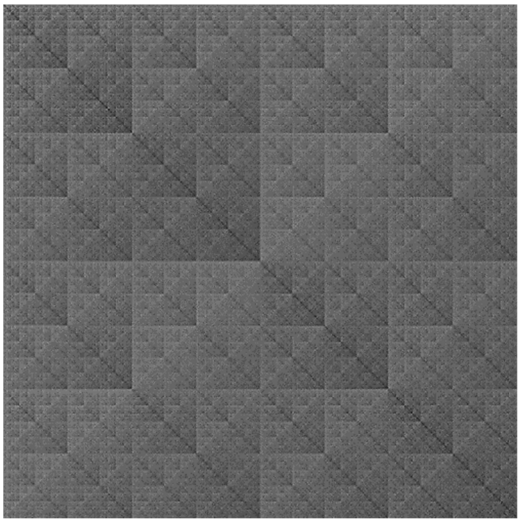


FIGURE 4. The SSM with 4096 combinations of the 12 tones matrix. Each column and each row represent the distance between all of them. It is also a symmetric matrix with a black diagonal, which depicts the distance of each vector to itself with value equal to zero.

to select appropriate threshold depending on the number of epochs. Therefore, AAE is more suitable to use it as a stop criterion of the algorithm for adjustment of vector positions with error less than 10% after 20 iterating cycles.

To increment speed of convergence the iterative step (see Fig. 2) can be optimized by computing the distances of SSM previously. There are only 4096 possible combinations of tones therefore, all distances between those combinations can be previously pre-computed. In the image presented in Fig. 4 all the distances between each of the tonal vectors are normalized from zero to one. They are shown in grey scale, being zero the closest distance represented in black and the bigger distances represented with brighter gray.

The use of this method over all the possible combinations of the 12 possible combined tones tends to produce a sphere (see Fig. 5). After some iterations, when a spherical form is approached, the projection collapses and vectors start to push each other in a loop.

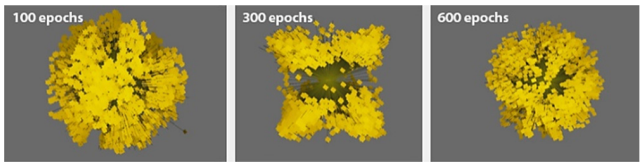


FIGURE 5. Initial 100 epochs (left), 300 epochs (middle), and 600 epochs (right) this numbers were selected to show the loop of the 4096 harmonic vectors and their behavior.

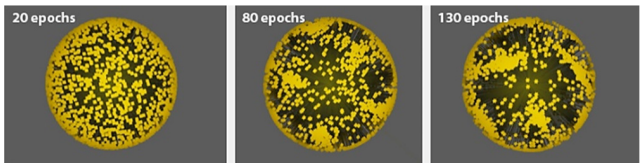


FIGURE 6. SSM with unitary magnitude. Initial 20 epochs (left), 80 epochs (middle), and 130 epochs (right).

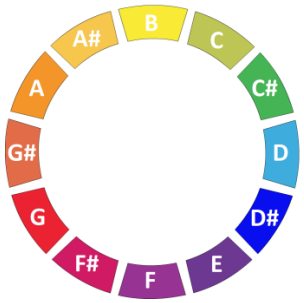
The next step was to set unitary magnitude of each vector for visualization. The result is shown in Fig. 6. Even with uniform scatter vectors no information about the notes or the harmony is visible yet.

For better perception of visualized vectors, the color division mentioned in [24] can be applied, using hue values of HSV (hue, saturation, value) color space in the incremental order. This approach reduces the complexity of techniques used spacing values by seven semitones as in the circle of fifths [31] particularly, adopted in the Spiral Array 3D representation [4]. Combining the same idea of circle of fifths and deep learning process used in *word2vec* method [32], it is possible to encode harmonics and relationships between tonal vectors. The main goal is to find possible candidates to replace existing combinations in the composition. Unfortunately, this approach needs a complex learning process, which could be time consuming.

IV. EXPERIMENTS AND DISCUSSION

This section covers the description of the experiments to evaluate performance of the proposed method. The first set of experiments demonstrates the use of colors on vector projection to visualize harmonic relationships. The second experiment has a purpose to visualize generated clusters of vectors in reduced dimensional space taking into account collisions between them. The last experiment is used for comparison of the projections of all possible harmonic combinations obtained by the proposed method and the autoencoder as a possible reference.

A piano is the musical instrument that has seven octaves and represents the complete set of octaves found in tempered music. The MIDI file also has these octaves and three additional octaves representing every note within them by values between 0 and 127. Whole set of notes correspond to octaves labeled from -1 to 9 [28]. A piano corresponds to a subset of octaves indexed in range from 1 to 7 found in the MIDI values between 21 and 108. Each of these values in MIDI corresponds to a frequency found at each note of



INDEX	RGB			3D		
C = 0	254	242	0	0	-1	0
C# = 1	141	199	1	0.9	-0.4	0
D = 2	0	166	82	0.3	-0.4	0.9
D# = 3	1	168	158	-0.7	-0.4	0.5
E = 4	0	84	165	-0.7	-0.4	-0.5
F = 5	46	49	146	0.3	-0.4	-0.9
F# = 6	102	46	145	0.7	0.4	0.5
G = 7	146	39	143	-0.3	0.4	0.9
G# = 8	238	28	37	-0.9	0.4	0
A = 9	243	101	35	-0.3	0.4	-0.9
A# = 10	255	148	0	0.7	0.4	-0.5
B = 11	254	197	2	0	1	0

FIGURE 7. Hue representation of notes (left), corresponding RGB values with coordinates of 3D positions in an icosphere.

every octave. Tones of equal tempered music lie in range of [8.1757989156 - 15.4338531643] in octave labeled as −1 of the MIDI standard. All the following notes in every octave are a multiple of these initial values. As a reference, the note *A* in octave 4 is called middle *A* and it has the frequency value of 440 Hz as well as number 69 is assigned to it in MIDI standard [33].

To assign a color to each of the twelve notes the consideration is to use a discrete division of color tones as it is shown in Fig. 7. Types of color expressed by hue color space are also split up into 12 incremental subdivisions in the discrete manner and then transformed back to their red, green, and blue values (RGB) assigned to their corresponding index. They were selected in a similar clockwise direction in the same order as the tones found in octaves.

If 12 dimensional vector has more than one value, the resulting color of the vector in three dimensions is computed by averaging tones of the notes found in it. We measure distances between all used 4096 possible combinations of the 12 dimensional vectors and compare them by each other creating the SSM with Euclidean distance as a principal metric.

The first set of experiments shown in Fig. 5 has been repeated to visualize color labeling of the projected 3D vectors. The expectative was to find similar colors placed closer at the same number of epochs as in previous experiments, but visualizing colors that represent tone combinations with the corresponding hue values (see Fig. 8). This experiment runs without applying normalization of magnitude and without detection of vector collisions.

In the second set of experiments we analyzed detection of vector collisions to visualize how many vectors are grouped together. It is an important step of our proposal, because sometimes multiple vectors cannot be seen if they are plotted to the same point in space as well as the clusters are not comparable if they contain one or multiple vectors. This helps to know whether harmonies are representative in the composition or they are outlier, when the density of vectors in cluster is low. Particularly, the normalized distances with an initial state of random positions have been compared with initial positions given by 12 points of an icosphere, which correspond to the 12 notes. The convergence of the projection got faster with the initial state of random positions.

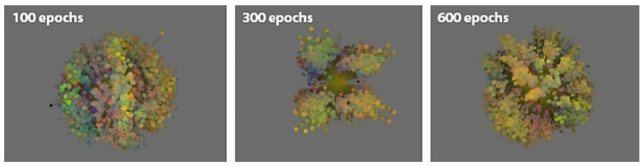


FIGURE 8. SSM with color labeling for Initial 20 epochs (left), 80 epochs (middle), and 130 epochs (right).

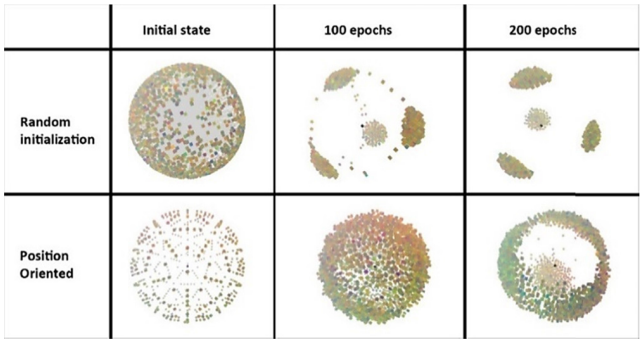


FIGURE 9. Hue colored 12 dimensional vectors of all the possible 4096 combinations. Initial state (left), 100 epochs (middle), and 200 epochs (right) for random and position-oriented initialization.

The projection reached a balanced state in which vectors start a loop. After reaching convergence, there were some vectors changing clusters, but the number of clusters remained the same (see Fig. 9). Before initializing the iterations process and after computing the corresponding to music piece SSM, we tested both starting points for the vectors. In our approach random and fixed initializations have been tested. One starting point was considered as a random position for each of the vectors and another one by setting a fixed position to each vector given by its assigned 3D point of hue value depicted in Fig 7.

The obtained results show that with every random initialization the projection converges faster than with the position-oriented one. With position-oriented initialization at 200 epochs the vectors just started to set the clusters and with the random initialization this was achieved at 100 epochs. By the iteration 200, the random initialization reached a stable projection with four clusters and the fixed initialization of points was still ordering the vectors.

To visualize each of the tonal 3D vectors, 3-orthogonal planes of size *s* are used. Particularly, 3D projection can be used in virtual environments like videogames with associated graphic libraries. The virtual camera is not of fixed-view and it could be adjusted to particular positions for best visualization of the results in different camera angles. It is done by dragging a mouse over a drawing canvas to rotate the resulting projected vectors. Thus, the proposed tool provides multisided visualization instead of static representation of data vectors. In the presented Figs. 8 and 9 as well in the next figures, the camera was adjusted to visualize results by showing the biggest amount of vectors, fixing the silence vector at the center with black color. That provides better visualization

of stems in a composition in relation to themselves and to the silence vector.

The following minimum criterion must be used for collision detection in three-dimensional space particularly, the distance between two vectors cannot be smaller than s. If distance is less than $s/2$, both vectors are separated by $\alpha = s/2$ using (4). For example, if there are two vectors in 12 dimensions described as $v1 = [1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0]$ and $v2 = [0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0]$ with an *Euclidean* distance $d1{=}1.732$ between them so, there are corresponding vectors in 3D space with random initialization, which are depicted as $v1_{3D} = [0.3, 0.5, 0.3]$ and $v2_{3D} = [0.3, 0.7, 0.2]$ with a corresponding *Euclidean* distance of $d2 = 0.223$.

If the size s of each 3 sided plane is equal to $s{=}3$ the value of $d2$ is smaller than size of s/2 so, the distance between two 3D vectors is set by the linear function shown in (4). Since $s{=}3$ then $\alpha = 1.5$ and by applying (4) we get the new values for $v1_{3D} = [0.3, 0.4, 0.35]$ and $v2_{3D} = [0.3, 0.85, 0.125]$ with a new distance $d2 = 0.503$. In result, it is still smaller than the size of s/2, which is the size of 3 orthogonal planes.

This process is applied only once because at the next iteration, when it is evaluated again, both vectors will be updated again generating values $v3 = [0.3, 0.175, 0.4625]$ and $v4{=}[0.3, 1.187, -0.04375]$ with $d2{=}1.132$, which is still smaller than $s/2$ and $d1$. At the following iteration both vectors are updated and their new values will be $v3 = [0.3, -0.331, 0.715]$ and $v4 = [0.3, 1.95, -0.423]$ with $d2{=}2.55$, which is finally bigger than $d1$ and s/2.

To evaluate the proposed method for music visualization based on spherical projection, some famous musical pieces have been processed. First, the method splits the composition into each music *bar/measure* found in the file, and then each measure is subdivided again in non-overlapping window with size of the crotchet declared in the MIDI file. In Fig. 10 we can observe projections of piano sonatas by different classical composers using Euclidean distance to create the corresponding SSM projection. Similarly, Fig. 11 shows some compositions of famous movies soundtracks applying our spherical projection method.

At first sight we can observe the higher complexity in the use of different harmonies expressed by vector sparsity or density at 3D projection. The colors represent tones, similar colors have similar tones. For example, in both Mozart and Chopin sonatas the density is visually higher than the density in *Moonlight Sonata* of Beethoven. This is because in Moonlight Sonata the same combinations of notes are used through the entire music composition.

We can observe a single yellow vector at the bottom of Beethoven projection in Fig. 10, which represents a *B* note colored according to rules presented in Fig. 7. This *B* note is not repeated and is not shared with other notes otherwise, several vectors would be closer to it, but it is isolated. In the projection of Yann Tiersen composition in Fig. 11 we can observe even lower vector density comparing it to density of *Moonlight Sonata* from Fig. 10.

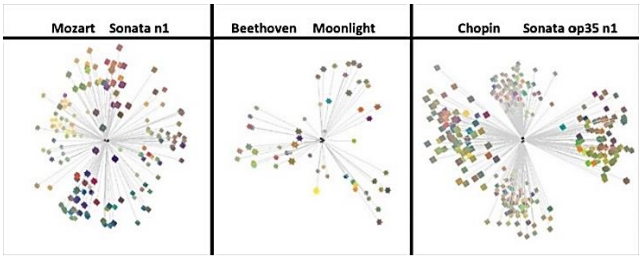


FIGURE 10. A Piano sonata by Mozart, Beethoven, and Chopin, respectively.

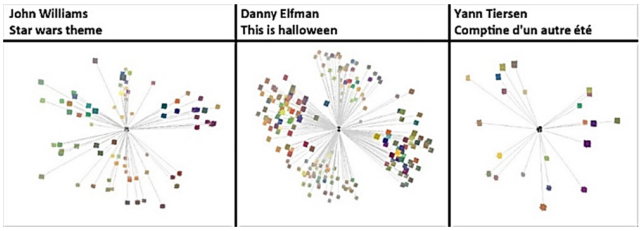


FIGURE 11. A Soundtrack composition by John Williams, Danny Elfman, and Yann Tiersen, respectively.

To compare the efficiency of the proposed method for reduction of dimensionality, the well-known autoencoder is selected for third experiment, because it is the best frequently referenced method. An autoencoder has been used to process 4096 combinations for 12 notes represented by vectors in the 12-dimensional space.

An autoencoder consists of input layer followed by a layer of three neurons as bottleneck *(x, y, z)*. The third layer on its input receives data representing three-dimension vectors and the last layer is used for generation of outputs, which correspond to representation of vectors in 12 dimensions.

An autoencoder uses MSE with a learning rate set to 0.01, which empirically is not too high to keep jumping or too low to stuck in local minima [13]. The loss of the neural network at the implementation of autoencoder keeps decreasing but the adjustment of the deltas stops around the 31,000 epochs. The autoencoder splits the projected vectors at the lower dimensionality shaping a cube (see Fig. 12), which has no relationship with a perception metrics of harmonies in a song.

For comparison of both approaches for dimensionality reduction the MSE and AAE errors are computed according to (6) and (7), respectively. Figure 12, which depicts results of iteration process by the autoencoder, can be visually compared to the results obtained by the proposed approach shown in Fig. 9. Even when the autoencoder reduces dimensionality faster and with less error in reconstruction of the projection, it cannot be tuned to measure harmonic relations by mathematical representations. Our method works with SSM of vectors computed without restrictions to a specific metrics. The SSM works as a bridge for vectors visualization from higher dimensional space into lower dimensional space. It is computed to test metrics and observe their projected behavior.

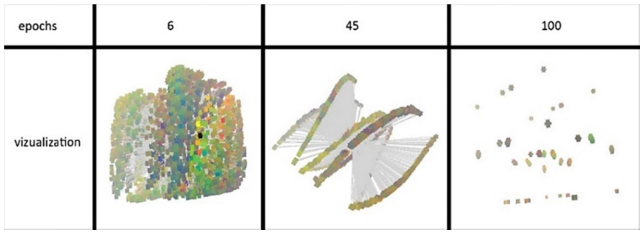


FIGURE 12. Visualization of the projected vectors by autoencoder at different iterations of the process of reducing the 4096 harmonies from the higher dimension.

The epochs are the number of cycles that each of the adjustment methods employ. Every time all the tonal vectors are adjusted, the error fitness process is updated, and an epoch is counted. Despite of the autoencoder has a lower error in the fitness function, it is not adjustable as our approach, which can change the distance function to generate new SSM and produce new projection that can fit the desire relationships. At the projection of the 4096 tonal combinations in 12 dimensions, both errors generated from the autoencoder were MSE $= 3.18\%$ and AAE $= 0.005\%$ with 31,000 epochs.

Our method had only 300 epochs to get AAE $= 0.67$ set by hand as threshold with an MSE $\approx 10\%$. The MSE presented in our method is directly related to the selected distance, unlike the MSE presented in the autoencoder, which is the error obtained by the non-linear compression by the layers of the neural network. This approximates the error of the projection of the fully connected layers, but they are not directly related to the higher dimensionality reduction or music visualization as our proposed method. The absence of correlation of the errors does not allow a direct quantitative comparison.

On the other hand, the clusters generated by the autoencoder could have a difficult interpretation during visualization. It is important to mention that the proposed method has an additional advantage, which consists in very simple process of selection and adjustment of metrics used for analysis of the behavior of dimensionality reduction.

In the next experiment, we have selected Hamming distance metrics to compare it with previously used Euclidean distance. Without any changes of the principal steps of the proposed method depicted in Fig. 2, new self-similarity matrix corresponding to Hamming distance is computed. The obtained results applying Hamming distance metrics are shown in Fig. 13 comparing them to results obtained by exploiting Euclidean distance. The obtained results show that both projections are quite similar and looks like scaled versions of each other (see Fig. 13).

Since colors represent notes, in our approach similar colors represent similar sounds (because the sounds are created by mixing notes), which are clustered in the projections. In Fig. 14 we can observe combinations of the vectors with different density. In jazz composition called *Blue train* by John Coltrane a sparsity of vectors is regular, while a mixture with less combinations is observed in the song *Whiter shade of pale* by Procol Harum. This means that Blue train uses

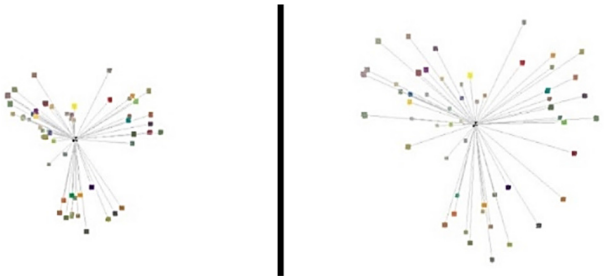


FIGURE 13. Spherical projection of Moonlight Sonata by Beethoven: Euclidean distance took 100103 iterations (left) and Hamming distance took 100201 iterations (right).

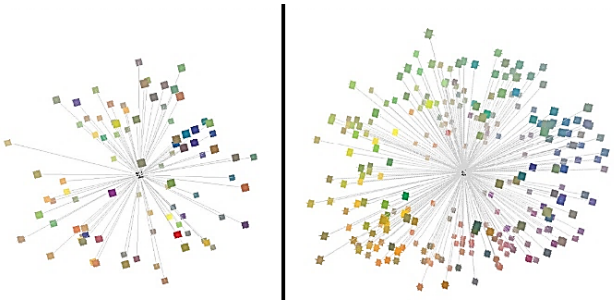


FIGURE 14. Spherical projection of compositions: Whiter shade of pale by Procol Harum (left) and Blue train by John Coltrane (right).

more harmonic combinations of twelve notes through the composition than *Whiter shade of pale*.

The proposed method considers silence and keeps it at the center with a proportional distance to each combination of the notes. This means, it preserves the distances between harmonies related to their distances to silence.

That is not considered in well-known scientific reports.

The precision of the method is defined by MSE and AAE measuring the loss of information during the projection from higher dimension to 3D space. We could manually change the metrics used to generate the SSM and all the procedures of the methods for dimensionality reduction will remain the same. That provides flexibility to explore different relations of harmonics representations in the music compositions.

V. CONCLUSION AND FUTURE WORK

The proposed method determines different kinds of features used for visualizations to choose one that better fit to explore a specific problem. Particularly, it has been exploited for the tonal representation of music compositions in 3D plots. A self-similarity matrix SSM based on the Euclidian and Hamming distances between vectors in 12 dimensional space has been proposed to use. Smooth and decreasing-only behavior of adjustment average error AAE, which relates dimensionality reduction errors with iterations during setting vectors in 3D space, has been adopted as a stop criterion of the algorithm for adjustment of vector positions. It facilitates finding a threshold for projection of reduced spatial dimensionality with error less than 10% after only 20 iterating cycles.

The Hamming distance is quite simple to use for practical implementation due to its binary representation while for visualization the Euclidean distance provides a scaled result proportional to the square root. Both provide the similar spatial but scaled vector magnitude projections. The main difference is the simpler implementation of the Hamming distance and as consequence, the higher speed of dimensionality reduction.

It was discovered that the autoencoder is faster because it needs less epochs to converge, but it reduces dimensionality redundancy using entirely numerical analysis based on non-linear relation inside the neural network. It also has a stop criterion, but the reduction is not based on a specific metrics designed for data analysis or for interpreting the music data. The proposed method can make low dimensional projection and expose visual patterns applying the same process for operating with different parameters used for performance analysis.

Additionally, it considers the silence as a vector and all the relations of the vectors are set around it. No other methods have been found that take into account silence however, it is an important part of the music composition process in general. Particularly, a silence is considered as the current starting point for a music piece and must be counted before sound begins.

The theoretical contributions consist in using the crotchet declared in the MIDI files as a self-reference to set a size of time window to split music compositions. Then the reduction of all possible combinations of harmonies in 12D space is applied to convert them into finite set of 3D vectors, which can be used for analysis and visualization of harmonic features in a file. The practical contributions include an original technique developed for projection of music composition features in 3 dimensional space based on a self-similarity matrix of high dimensional vectors.

As future work, the development of generative model could benefit from the creation of the profiles of songs based on the proposed method for extraction and projection of data with reduced computational cost. A weighted version of the time windows at each moment could be developed taking into consideration the complete sequence of the tones using as metric the cosine distance to generate the SSM. This abstraction can help to precise better the composition profile. The profile could be used to design objective function finding the global optima exploiting meta-heuristics [14] like genetic algorithm or simulated annealing. Both the profile and meta-heuristics can serve to implement more efficient generative model. Also, with this method it is possible to create composer recognition approach based on dimensionality reduction of harmonics. We already have some promising results with this new approach.

REFERENCES

[1] F. Korzeniowski and G. Widmer, "Genre-agnostic key classification with convolutional neural networks," in *Proc. 19th Int. Soc. Music Inf. Retr. Conf.*, Paris, France, Aug. 2018, pp. 1–7. [Online]. Available: https://arxiv.org/abs/1808.05340

[2] S. Kaski and J. Peltonen, "Dimensionality reduction for data visualization [applications corner]," *IEEE Signal Process. Mag.*, vol. 28, no. 2, pp. 100–104, Mar. 2011.

[3] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[4] R. Lopez, J. Regier, M. I. Jordan, and N. Yosef, "Information constraints on auto-encoding variational Bayes," in *Proc. Int. Conf. Neural Inf. Process. Sys. (NIPS)*, Montreal, QC, Canada, Dec. 2018, pp. 1–16. [Online]. Available: https://arxiv.org/abs/1805.08672

[5] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proc. 14th Python Sci. Conf.*, Austin, TX, USA, 2015, pp. 18–24.

[6] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "madmom: A new python audio and music signal processing library," in *Proc. ACM Int. Conf. Multimedia*, New York, NY, USA, Oct. 2016, pp. 1–4. [Online]. Available: https://arxiv.org/abs/1605.07008

[7] R. Valle, *Visual Display and Retrieval of Music Information*. Berkeley, CA, USA: Univ. California, 2018. [Online]. Available: https://arxiv.org/abs/1807.10204

[8] Google AI. (Aug. 2019). *Magenta—Open Source Research Project*. [Online]. Available: https://ai.google/research/teams/brain/magenta/

[9] L. van der Maaten, "Accelerating t-SNE using tree-based algorithms," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3221–3245, Oct. 2014.

[10] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, "MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer," in *Proc. ISMIR Int. Soc. Music Infor. Retr. Conf.*, Paris, France, Sep. 2018, pp. 1–8. [Online]. Available: https://arxiv.org/abs/1809.07600

[11] E. Chew, "Real-time pitch spelling," in *Mathematical and Computational Modeling of Tonality*, vol. 204. Boston, NY, USA: Springer, 2014, ch. 8, pp. 133–148. [Online]. Available: https://www.springer.com/gp/book/9781461494744

[12] R. Cohn, "Introduction to neo-riemannian theory: A survey and a historical perspective," *J. Music Theory*, vol. 42, no. 2, pp. 167–180, 1998.

[13] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *Proc. Int. Conf. Lear. Repres. (ICLR)*, Toulon, France, Apr. 2017, pp. 1–19. [Online]. Available: https://arxiv.org/abs/1703.00395

[14] Y. Liu, J. Pan, and Z. Su, "Deep blind image inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 1–9. [Online]. Available: https://arxiv.org/abs/1712.09078

[15] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Repres. (ICLR)*, Scottsdale, AZ, USA, Dec. 2013, pp. 1–14. [Online]. Available: https://arxiv.org/abs/1312.6114

[16] D. Meredith, "Compression-based geometric pattern discovery in music," in *Proc. 4th Int. Workshop Cogn. Inf. Process. (CIP)*, Copenhagen, Denmark, May 2014, pp. 1–6. doi: 10.1109/CIP.2014.6844503.

[17] T. Bergstrom, K. Karahalios, and J. C. Hart, "Isochords: Visualizing structure in music," in *Proc. Graph. Interface*, Montreal, QC, Canada, 2007, pp. 297–304.

[18] J. Foote and M. Cooper, "Visualizing musical structure and rhythm via self-similarity," in *Proc. Int. Comp. Music Conf.*, La Habana, Cuba, 2001, pp. 4–8.

[19] J. Wolkowicz, S. Brooks, and V. Keselj, "MIDIVIS: Visualizing music pieces structure via similarity matrices," in *Proc. Int. Comp. Music Conf.*, Montreal, QC, Canada, 2009, pp. 53–56.

[20] J. Snydal and M. Hearst, "ImproViz: Visual explorations of jazz improvisations," in *Proc. Extended Abstr. Hum. Factors Comput. Syst. (CHI)*, Portland, OR, USA, 2005, pp. 1805–1808.

[21] J. H. Fonteles, M. A. F. Rodrigues, and V. E. D. Basso, "Creating and evaluating a particle system for music visualization," *J. Vis. Lang. Comp.*, vol. 24, no. 6, pp. 472–482, Dec. 2013.

[22] D. Herremans and E. Chew, "Tension ribbons: Quantifying and visualising tonal tension," in *Proc. Int. Conf. Tech. Music Notations Repres.*, Cambridge, U.K., 2016, pp. 1–6.

[23] C. S. Sapp, "Harmonic visualizations of tonal music," in *Proc. Int. Comp. Music Conf.*, La Habana, Cuba, 2001, pp. 8–18.

[24] P. Ciuha, B. Klemenc, and F. Solina, "Visualization of concurrent tones in music with colours," in *Proc. Int. Conf. Multimedia*, Firenze, Italy, 2010, pp. 1677–1680.

[25] C.-H. Chen, M.-F. Weng, S.-K. Jeng, and Y.-Y. Chuang, "Emotion-based music visualization using photos," in *Advances in Multimedia Modeling*, vol. 4903, S. Satoh, F. Nack, and M. Etoh, Eds. Berlin, Germany: Springer, 2008, pp. 358–368.
[26] Y.-H. He, *A Visualization of the Classical Musical Tradition*. London, U.K.: Univ. London, 2017, pp. 1–9. [Online]. Available: https://arxiv.org/abs/1709.04038
[27] G. Febres and K. Jaffe, "Music viewed by its entropy content: A novel window for comparative analysis," *PLoS ONE*, vol. 12, no. 10, Jul. 2017, Art. no. e0185757. doi: 10.1371/journal.pone.0185757.
[28] D. Back. (Mar. 2019). *Standard MIDI-File Format Spec. 1.1*. [Online]. Available: http://www.music.mcgill.ca/~ich/classes/mumt306/StandardMIDIfileformat.html
[29] H. S. Warren, *Hacker's Delight*, 2nd ed. Upper Saddle River, NJ, USA: Addison-Wesley, 2013.
[30] X. R. Li and Z. Zhao, "Measures of performance for evaluation of estimators and filters," *Proc. SPIE*, vol. 4473, Nov. 2001, pp. 1–12.
[31] J. Clough and G. Myerson, "Musical scales and the generalized circle of fifths," *Amer. Math. Monthly*, vol. 93, no. 9, pp. 695–701, Nov. 1986.
[32] C.-H. Chuan, K. Agres, and D. Herremans, "From context to concept: Exploring semantic relationships in music with Word2Vec," in *Neural Computing and Applications*. London, U.K.: Springer, 2018, pp. 1–14. doi: 10.1007/s00521-018-3923-1.
[33] F. Stolzenburg, "Harmony perception by periodicity detection," *J. Math. Music*, vol. 9, no. 3, pp. 215–238, Sep. 2015.
[34] L. Blier, P. Wolinski, and Y. Ollivier, "Learning with random learning rates," in *Proc. Int. Conf. Lear. Repres.*, New Orleans, LA, USA, May 2019. [Online]. Available: https://arxiv.org/abs/1810.01322
[35] B. Ghojogh, S. Sharifian, and H. Mohammadzade, "Tree-based optimization: A meta-algorithm for metaheuristic optimization," *Pattern Anal. Appl.*, vol. 19, no. 1, pp. 1–10, Sep. 2018.

OMAR LOPEZ-RINCON was born in Tampico, Tamaulipas, Mexico, in 1979. He received the B.S. and M.S. degrees in computer science from the Universidad de las Américas Puebla, Mexico, in 2005 and 2015, respectively, where he is currently pursuing the Ph.D. degree in intelligent systems.

He is the author of more than ten research articles in several refereed journals, books, and conference proceedings. His research interests include computer vision, signal processing, generative models, video games, pattern recognition, agents, deep learning, and music composition with artificial intelligence.

OLEG STAROSTENKO was born in Lviv, Ukraine, in 1959. He received the B.S. and M.S. degrees in Computer science from the Lviv State University, Ukraine, in 1982, and the Ph.D. degree in mathematics and physics from the University Autonoma, Mexico, in 1996.

Since 1996, he has been a full-time Professor with the Department of Computing, Electronics, and Mechatronics, Universidad de las Américas Puebla, Mexico. He has authored more than 200 research articles in several refereed journals, books, and conference proceedings. His current research interests include access, retrieval, transmitting, and processing of multimedia information in distributed environments.

Dr. Starostenko belongs to the Mexican National System of Researchers (Level I), since 2000.