Algorithms for Computing Geometric Measures of Melodic Similarity

Greg Aloupis,* Thomas Fevens,† Stefan Langerman,‡ Tomomi Matsui,§ Antonio Mesa,¶ Yurai Nuñez,¶ David Rappaport, and Godfried Toussaint***

*School of Computer Science, McGill University
3480 University Street
Montreal, Quebec, Canada H3A 2A7
{athens,godfried}@cs.mcgill.ca
†Department of Computer Science and Software Engineering, Concordia University
1455 de Maisonneuve Boulevard West
Montreal, Quebec, Canada H3G 1M8
fevens@cs.concordia.ca
‡ Chercheur Qualifié du FNRS, Département d'Informatique, Université Libre de Bruxelles
CP212 Boulevard du Triomphe, 1050 Bruxelles, Belgium
Stefan.Langerman@ulb.ac.be
§Department of Mathematical Informatics
Graduate School of Information Science and Technology, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656 Japan
tomomi@misojiro.t.u-tokyo.ac.jp
¶Departamento Ciencias de la Computacion
Facultad de Matematica y Computacion,
Universidad de La Habana
San Lazaro y L, Vedado 10400
Ciudad de La Habana, Cuba
tonymesa@matcom.uh.cu, yurainr@yahoo.com
**School of Computing, Queen's University
Kingston, Ontario, Canada K7L 3N6
daver@cs.queensu.ca

We have all heard numerous melodies, whether they come from commercial jingles, jazz ballads, operatic arias, or any of a variety of different sources. How a human detects similarities in melodies has been studied extensively (Martinez 2001; Hofmann-Engl 2002; Müllensiefen and Frieler 2004). There has also been some effort in modeling melodies so that similarities can be detected algorithmically. Some results in this fascinating study of musical perception and computation can be found in a collection edited by Hewlett and Selfridge-Field (1998).

Similarity measures for melodies find application in content-based retrieval methods for large music databases such as query by humming (QBH) (Ghias et al. 1995; Mo, Han, and Kim 1999) but also in other diverse applications such as helping prove music copyright infringement (Cronin 1998). Previous formal mathematical approaches to rhythmic and melodic similarity, such as the one taken in this article, are based on methods like one-dimensional edit-distance computations (Toussaint 2004), approximate string-matching algorithms (Bainbridge et al. 1999; Lemström 2000), hierarchical correlation functions (Lu, You, and Zhang 2001), two-dimensional augmented suffix trees (Chen et al. 2000), transportation distances (Typke et al. 2003; Lubiw and Tanur 2004), and maximum segment overlap (Ukkonen, Lemström, and Mäkinen 2003).

Computer Music Journal, 30:3, pp. 67–76, Fall 2006

Figure 1. The first two measures of a well-known melody are shown below our representation using an orthogonal polygonal chain.

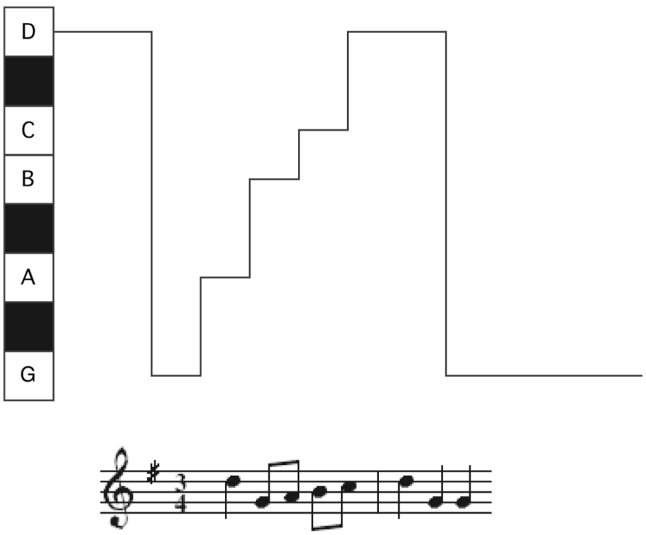
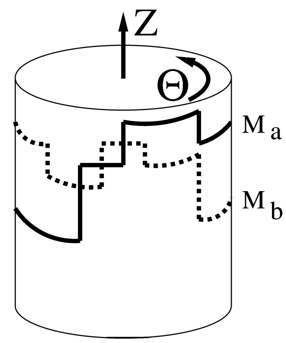


Figure 2. Two orthogonal periodic melodies.



Ó Maidín (1998) proposed a geometric measure of the difference between two melodies M_a and M_b. The melodies are modeled as monotonic pitch-duration rectilinear functions of time, as depicted in Figure 1. This rectilinear representation of a melody is equivalent to the triplet melody representation in Lu, You, and Zhang (2001). Ó Maidín measures the difference between the two melodies by the minimum area between the two polygonal chains, allowing vertical translations. The area between two polygonal chains is found by integrating the absolute value of the vertical L_1 distance between M_a and M_b over the domain Θ. Arkin et al. (1991) show that the minimum integral of any distance L_p ($p \geq 1$) between two orthogonal cyclic chains, allowing translations along Θ and z, is a metric.

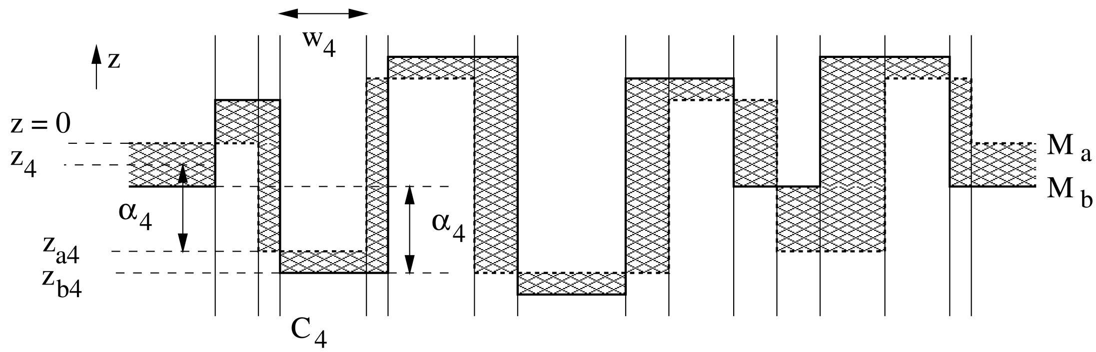
In a more general setting such as music retrieval systems, we might consider matching a short query melody against a larger stored melody. Furthermore, the query can be presented in a different key (transposed in the vertical direction) and in a different tempo (scaled linearly in the horizontal direction). Francu and Nevill-Manning (2000) compute the minimum area between two such chains, taken over all possible transpositions. They do this for a constant number of pitch values and scaling factors, and each chain is divided into m and n equal time steps. They claim (without describing in detail) that their algorithm takes $O(nm)$ time, where n and m are the number of unit time-steps in each query. This time bound can be achieved with a brute-force approach.

In some music domains such as Indian classical music, Balinese gamelan music and African music, the melodies are cyclic, i.e., they repeat over and over. In Indian music the rhythmic cycles (meter) are called *talas* (Morris 1998), and in the music of north Bali they are called *lambatan* (Ornstein 1971). If timbre is added to the talas in the form of drum sounds we obtain what are called *thekas,* which may be considered in effect as cyclic melodies (Clayton 2000). Such cyclic melodies are also a fundamental component of African and Balinese music (Montfort 1985). Although the untrained listener may assume that African drumming is not melodic, this is far from the truth. In the words of A. M. Jones, "the drums are not merely beating time, for each note has to be beaten on its own correct pitch" (Jones 1954). Furthermore, in the Afro-Cuban Batá drumming, where several double-skinned drums are used, the skins are tuned so that the melodies may use tones and even semitones (Nodal 1983). Indeed, African and Afro-Cuban Batá drums produce cyclic melodies. Of course, in much of the Balinese music the cyclic melodies are played on gamelans, in which the metal pieces are even more finely tuned (Carterette and Kendall 1994).

Two such cyclic melodies can be represented by orthogonal polygonal chains on the surface of a cylinder, as shown in Figure 2. This is similar to Thomas Edison's cylinder phonographs, where music is represented by indentations around the body of a tin foil cylinder.

This article is an extension of the material pre-

Figure 3. Contribution of C_4 to area calculation.



sented in Aloupis et al. (2003). We describe two algorithms to find the minimum area between two given orthogonal melodies, M_a and M_b, of size n and m, respectively ($n > m$). Here, n (or m) is the number of non-vertical edges in the polygonal chain representation. The algorithms can be used for cyclic melodies as well as in the context of retrieving short patterns from a database (open planar orthogonal chains). Apart from minor details, there is no difference between the cyclic and open cases. We have chosen to describe the algorithms for the case where the melodies are cyclic. The first algorithm assumes that the Θ direction is fixed, and it runs in $O(n)$ time. The second algorithm finds the minimum area when both the z and Θ relative positions can be varied. We prove that it runs in $O(nm\log n)$ time. In each case, we assume that the edges defining M_a and M_b are given in the order in which they appear in the melodies. Finally, we discuss natural extensions, both for the polygonal description of melodies and for different types of queries.

Minimization with Respect to *z* Direction

In the first algorithm, we will assume that both melodies are fixed in the Θ direction. Without loss of generality, we will assume that melody M_a is fixed in both directions, so all motions are relative to M_b.

To see how the area between the two melodies changes as M_b moves in the z direction, consider a set of lines defined by all vertical edges of the melodies as shown in Figure 3. This set of lines partitions the area between the melodies into rectangles C_i ($i = 1, \ldots, k$), each defined by two vertical lines and two horizontal edges (one from each melody). Note that k is at most $n + m$. The area between M_a and M_b is the sum of the areas of all C_i. If M_b starts completely below M_a and moves in the positive z direction, then for any given C_i the lower horizontal edge (from M_b) will approach the upper fixed horizontal edge, while the area of C_i decreases linearly. This happens until the horizontal edges are coincident and the area of C_i is zero. Then the upper horizontal edge (now from M_b) moves away from the lower fixed horizontal edge, while the area of C_i increases linearly.

We will consider the vertical position of M_b to be the z-coordinate of its first edge. We define $z = 0$ to be the position where this edge overlaps the first edge of M_a. Let $A_i(z)$ denote the area of C_i as a function of z. Define z_i to be the coordinate at which $A_i = 0$. These k positions of M_b where some A_i becomes zero are called *z-events.* The slope of $A_i(z)$ is determined by the length of the horizontal edges of C_i. The total area between M_a and M_b is given by

$$A(z) = \sum_{i=1}^{k} A_i(z).$$

Note that because $A(z)$ is the sum of piecewise-linear convex functions, it too is piecewise-linear and convex. Furthermore, its minimum must occur at a z-event.

The function $A(z)$ is given by

$$A(z) = \sum w_i \mid z_{bi} - z_{ai} \mid ,$$

where z_{bi} is the vertical coordinate of M_b in C_i, z_{ai} corresponds to M_a, and w_i is the weight (width) of C_i, as shown in Figure 3. Let α_i denote the vertical offset of each horizontal edge in M_b from z_{b1}. Thus we have $z_{bi} = z_{b1} + \alpha_i$, and $A(z)$ is now given by

$$A(z) = \sum w_i \mid z_{b1} - (z_{ai} - \alpha_i) \mid .$$

Finally, notice that the term $z_{ai} - \alpha_i$ is equal to z_i. Thus, we obtain

$$A(z) = \sum w_i \mid z_i - z_{b1} \mid .$$

This is a weighted sum of distances from z_{b1} to all the z-events. The minimum is the weighted univariate median of all z_i and can be found in $O(k)$ time (Reiser 1978). This median is the vertical coordinate that z_{b1} must have so that $A(z)$ is minimized. Once this is accomplished, it is straightforward to compute the sum of areas $O(k)$ in time. Recall that k is at most $n + m$; therefore, a minimum of $A(z)$ can be computed in $O(n)$ time.

Minimization with Respect to z and Θ Directions

If no vertical edges among M_a and M_b share the same Θ coordinate, then M_b may be shifted in at least one of the two directions $\pm\Theta$ so that the sum of areas does not increase. This means that to find the global minimum, the only Θ coordinates that need to be considered are those where two vertical edges coincide. Thus, our first algorithm may be applied $O(nm)$ times to find the global minimum in a total of $O(n^2m)$ time. We now propose a different approach to improve this time complexity.

As described in the previous section, for a given Θ, the area minimization resembles the computation of a weighted univariate median. When we shift M_b by $\Delta\Theta$, we are essentially changing the input weights to this median. Some C_i grow in width, some become narrower, and some stay the same width. As we keep shifting, at Θ coordinates where vertical edges coincide, we have the destruction of a C_i and creation of another C_i. An important observation is that all C_i grow (or shrink) at the same rate.

Let us store the z-events and their weights in the leaves of a balanced binary search tree. Each leaf represents one C_i. The leaves are ordered by the value z_i. Each leaf also has a label to distinguish between the three types of C_i: those that are growing, shrinking, or unaffected when M_b is shifted infinitesimally in the positive direction. At every node with subtree T, we store W_T (the sum of weights of all leaves in T) and D (the number of growing leaves minus the number of shrinking leaves in T). The weighted median of all z_i can be calculated by traversing the tree from root to leaf, always choosing the path that balances the total weight on both sides of the path. The time for this is $O(\log k)$.

Suppose that we shift M_b by some offset $\Delta\Theta$, which is small enough such that no vertical edges overlap during the shift. Each w_i belonging to a growing leaf must be increased by $\Delta\Theta$, and each w_i belonging to a shrinking leaf must be decreased by this amount. Instead of actually updating all our inputs, we just maintain a global variable $\Delta\Theta$, representing the total offset in the Θ direction. The total weight of a subtree T is now $W_T + D\Delta\Theta$.

When we shift to a position where two vertical edges share the same Θ coordinate, we potentially eliminate some C_i, create a new C_i, or change type of C_i. The number of such changes is constant for each pair of collinear vertical edges. The weight given to a created leaf must equal $-\Delta\Theta$. Each of these changes involves $O(\log k)$ work to update the information stored in the ancestors of a newly inserted, deleted, or altered leaf. There are $O(nm)$such instances where this must be done and where the median must be recomputed, so the total time to compute all candidate positions of M_b is $O(nm \log n)$.

At every Θ coordinate where we recalculate the median, we also need to calculate the integral of area between the two melodies. For a given median z_*, the area summation for those C_i for which $z_* > z_i$ has the form $\sum w_i(z_* - z_i)$. This can be calculated in $O(\log k)$ time if we know the value of this summation for every subtree. To do this, we store some additional information at every subtree T. Specifically, the area is given by

$$z_*(W_T + D\Delta\Theta) - \sum(w_i z_i) - \Delta\Theta \sum I z_i,$$

where in the second summation, I takes the values (+1, 0, –1) for growing, unchanged, and shrinking leaves, respectively. These two summations are the additional parameters that need to be stored, and they can be updated in $O(\log k)$ time at every critical Θ coordinate. We must also perform a similar $O(\log k)$ time calculation of $\sum w_i(z_i - z_*)$ for all $z_i > z_*$. No additional parameters are needed for this.

Thus, at every critical Θ position, we can calculate the median and integral of area in $O(\log k) = O(\log n)$ time. This implies that a relative placement such that the area between the melodies is minimized can be computed in $O(nm\log n)$ time.

The analysis above can be used to obtain the same result for the problem of matching two planar orthogonal monotonic open chains. Clearly, if we are interested in varying only one direction, an optimal placement can be found in linear time. If the direction of monotonicity is the x-axis, then this problem is more interesting if one of the two chains has a shorter projection onto the x-axis. This "shorter" chain reminds us of a short motif that we might search for in a larger database of music. For this problem, we measure area only within the common domain of the two chains along the x-axis. Naturally, the projection of the shorter chain must be entirely covered by the projection of the longer chain.

Arkin et al. (1991) showed that two polygonal shapes can be compared by parameterizing their boundary lengths and examining their orientation differences. They showed that their measure, which is invariant to scaling, rotation, and translation, can be computed by finding the minimum integral of the vertical distance between two orthogonal chains, which are constructed in a preprocessing step. In fact, some of their techniques are similar to those given in this section. However, they chose to use the L_2 distance (as opposed to the L_1 distance used here), for which the optimal z-position at any Θ can be computed in $O(1)$ time. The complexity of their algorithm is dominated by sorting the $O(nm)$ critical Θ events. They indicated that their algorithm offers no improvement over a $O(n^3)$ time brute-force approach for the L_1 metric.

Extensions

Higher Dimensions

Consider a simple orthogonal open chain that is monotonic with respect to the x-axis. Furthermore, at any particular x-coordinate, suppose that the chain has at most two edges (in the y and/or z directions). This is an extension of the melody representation that we have seen so far. The x-axis still represents time, but now the other axes might represent pitch, loudness, timbre, or chord density. In the plane, the measurement made was an integral of the pitch (height) difference taken over a domain in the x-axis. Here, we still wish to minimize an integral of the distance between two chains over all common x coordinates. Whether this should be computed as a Euclidean distance or perhaps the L_1 distance is debatable; the latter is definitely easier to compute.

Suppose that we allow motions of the chains M_a and M_b only in the y and z directions. Minimizing the sum of pair-wise Euclidean distances is equivalent to the Weber problem, which involves finding a point with minimum sum of distances to points in a given set. It is not possible to find an exact solution to the Weber problem (also known as the generalized Fermat-Torricelli problem; see Groß and Strempel 1998). Using the L_1 metric, we want to minimize the function

$$\sum w_i(|z_{bi} - z_{ai}| + |y_{bi} - y_{ai}|).$$

This can be split into a sum of two terms:

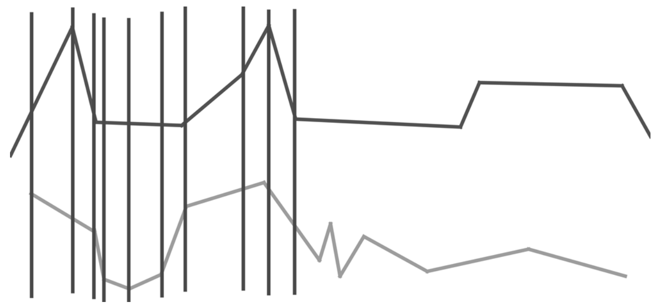
$$\sum w_i |z_{bi} - z_{ai}| + \sum w_i |y_{bi} - y_{ai}|.$$

Thus, we need to perform only two univariate median computations to find the optimal (y, z) placement for a particular relative position of the two chains in the x direction. In d dimensions, we can accomplish this task in $O(dn)$ time. The decoupling of the two coordinates allows us to update each median separately at every critical x coordinate. In three dimensions, there are still $O(nm)$ critical x coordinates and $O(n + m)$ weights/leaves, so the time complexity is the same as for planar chains. If we let n and m be the total number of edges parallel to the x-axis for two chains, then in three dimensions the time complexity becomes $O(nmd\log n)$, using $O(dn)$ space. Note that only these edges are significant in any of the computations we have made so far.

Scaling

Here we consider the effect of scaling planar chains, either in the vertical or horizontal directions. If we shrink the shorter chain horizontally, the domain of

Figure 4. Two monotonic chains and their strips.



the integral becomes smaller, so the total area will tend to zero eventually. How should we deal with this? It seems reasonable to normalize by computing the total area over the domain of the smaller chain. It is equivalent to fix the shorter chain at unit domain length and modify the larger chain instead. Its domain would expand from unit length to some value where its narrowest strip has unit width.

Let an "x-value" be an x-coordinate where there are vertical edges from both chains. For a particular scaling value, we know that the optimal placement of the larger chain occurs when we have an x-value. This follows from the arguments given in the second section of this article. Suppose that somehow we know the optimal scaling factor, and assume that there is only one x-value and that we know which two vertical edges are aligned. Now, we can keep scaling the large chain while using the x-value as an "anchor." One of the two scaling directions will improve the area minimization, at least until we obtain another x-value. Thus, for the scaling method proposed above, the optimal scaling of the larger chain occurs at a position where two or more x-values occur.

This means that we have $O(n^2m^2)$ candidate configurations for the larger chain. Thus a brute-force algorithm to find the optimal configuration (and vertical position) would take $O(n^3m^3)$ time using $O(n)$ space. Our result also applies to vertical scaling. In this case a brute-force algorithm would have a time complexity of $O(n^3m^3 \log n)$, because we would search along Θ for every scaling factor that aligns two pairs of horizontal edges.

Non-Orthogonal Chains

In the preceding sections, it was assumed that a melody can be divided into intervals, and within each interval the pitch (or volume/timbre) remains constant. In a more general setting, these features may vary within each interval. Non-orthogonal chains are relevant in a variety of contexts. In many types of music, we must consider melody in a more general sense than the discrete, static pitches of MIDI or common music notation. This is particularly true for example in Flamenco music and Indian music, in which the expressiveness of the voice plays an important role. A continuous change in pitch also reflects effects such as glissandi in Western classical performance. In such applications, continuous pitch variation is important (Battey 2004). Furthermore, in other applications such as signal-to-score music transcription and pitch tracking in real-time interactive improvisation systems, the input is continuous (Dobrian 2004; Kapanci and Pfeffer 2005).

A further step in this direction is to consider monotonic piecewise linear chains. Consider two such planar chains. Let us divide the plane into strips, just as we had for orthogonal chains. In this case, a vertical boundary is placed at every vertex, as shown in Figure 4.

Within every strip, we have two linear segments. Suppose we vary only the relative pitch of the chains. As one chain is moved down from infinity, the area within a given strip decreases linearly until the two segments touch inside the strip. Then the area decreases quadratically until the midpoints of the segments intersect. Of course, the reverse occurs as we keep moving the chain down. The overall area function of each strip C_i is now a symmetric convex function, which is part linear and part quadratic (around the symmetric point). The total area is a sum of n functions, such as those shown in Figure 5.

The area function is convex and piecewise quadratic with $O(n)$ inflection points. Specifically, an inflection point will exist in the aggregate function only at a coordinate where some individual function changes from linear to quadratic. There are two such points per individual function. Note that the minimum of the aggregate function need not occur at an inflection point, unlike the case of orthogonal chains. Now, it is possible for the minimum to exist

Figure 5. A set of area functions from the C_i *strips.*

Figure 6. The median Q_1 *of function minima.*

Figure 7. The median Q_2 *of left inflection points.*

Figure 8. Q_2 *to the left of* Q_1.

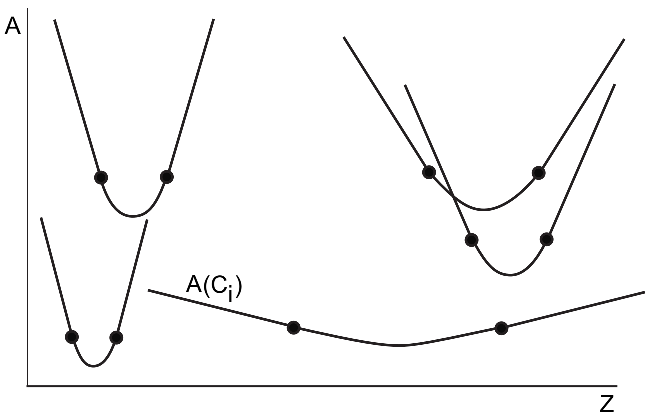


Figure 5

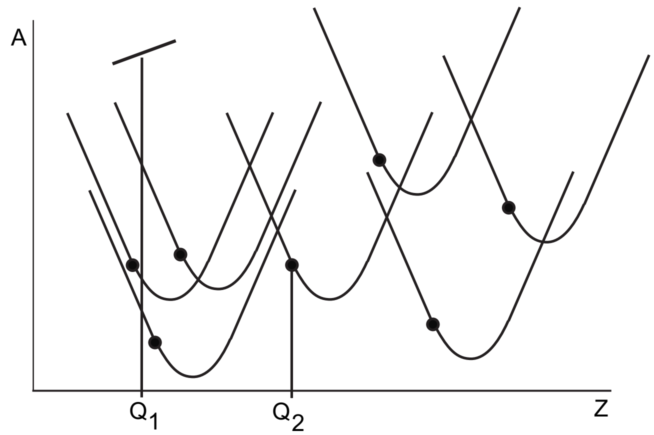


Figure 7

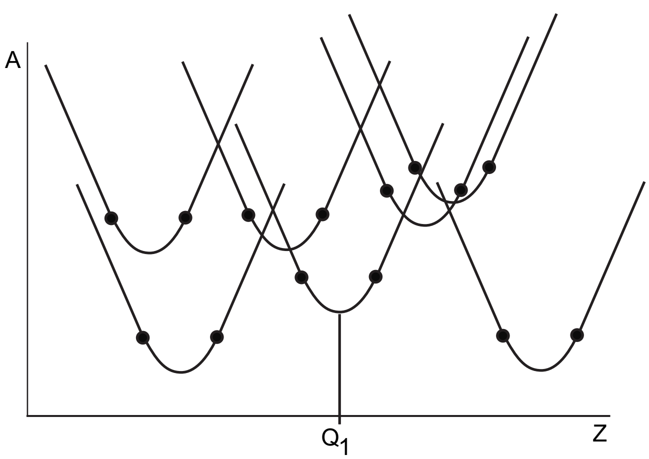


Figure 6

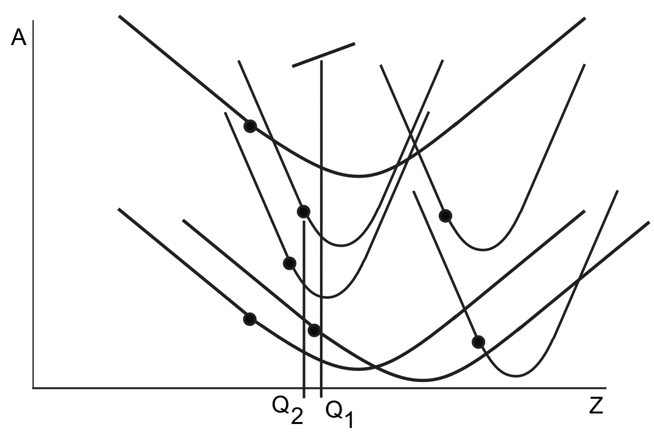


Figure 8

between two consecutive inflection points. This would be the only region between two successive inflection points where the function is not monotonic.

To compute the minimum of the aggregate function, we give the following algorithm:

1. Let R be the set of individual area functions. Let F be a single quadratic term, initialized to zero.
2. Compute Q_1, the median of the x-coordinates of the minima of all functions in R, as shown in Figure 6.
3. Compute the value and gradient of the total area function at Q_1 by querying F and all functions in R. If not at the global minimum, assume without loss of generality that the minimum is to the left of Q_1.
4. For the subset of functions in R whose minima are to the right of Q_1, compute the median Q_2 of their left inflection points. Q_2 splits the subset into the left group and the right group.
5. If $Q_2 \geq Q_1$, as shown in Figure 7, replace all functions in the right group with a single linear term, which is a summation of all individual left-hand linear terms. Update F by adding this term to it. Remove the right group from R.

6. Otherwise, if $Q_2 < Q_1$, as shown in Figure 8, compute the gradient of the total function at Q_2. If the global minimum is to the left of Q_2, follow the instructions of step 5 on the right group. Otherwise, if the minimum is between Q_2 and Q_1, replace all functions in the left group with a single quadratic term, which is a summation of all individual quadratic terms. Then update F and remove the left group from R.
7. Go to step 2.

The algorithm performs $O(|R|)$work during each iteration, and a constant fraction of R is removed each time. The total time is $O(n)$, by a simple geometric series summation, as given in Cormen et al. (2001). Thus, in linear time we can compute the minimum area between two chains monotonic in x, found over all vertical translations.

Updating the aggregate function as we shift one of the chains along the x-axis appears to be non-trivial. It is no longer true that the optimal position must occur when vertices from each chain are aligned vertically. Also, when we make a small shift along the x-axis, not only do the two linear parts of each individual function change slopes, but the center of symmetry of each function may also shift. (Recall that these are functions of the z-coordinate.) These changes depend on the slopes of our chains within each strip and are not difficult to compute on an individual basis. However, understanding their aggregate effect is a different matter. To rephrase, each strip now has three "z-events" instead of one: the two boundaries between linear and quadratic forms, plus the center of symmetry. To make things worse, the z-events change position as a chain is shifted along Θ. Thus, if a tree is used to maintain the median, it will be necessary not only to insert/delete leaves but also to rearrange the order of leaves (to say the least).

Integer Weights/Heights

Now, we discuss the cases where only certain pitches (heights) and/or weights are allowed. If there are $O(1)$ height differences allowed, we can sort all critical points in $O(nm\log n)$ and sweep along each height difference horizontally, updating the area function in $O(1)$ time per critical point (i.e., $O(nm)$ per height difference). As a result, the time complexity is dominated by the sorting step. Even in the simplest case, where we just wish to compute the minimum area while keeping z fixed, we do not know how to avoid sorting all critical positions.

If all weights are equal (i.e., we have evenly spaced sampling of melodies), then each median computation takes $O(m)$ time, and there are $O(n)$ critical positions. Thus, a brute force approach takes $O(nm)$ time. A direct implementation of our tree algorithm would take $O(nm\log n)$ time, because at each of the $O(n)$ critical positions we would have to update all $O(m)$ leaves of our tree. It is possible that this can be greatly improved.

Conclusion

We have given efficient algorithms for computing the minimum area between two polygonal chains, which is a known method of comparing melodies. Other sweep-line algorithms for melodic similarity exist (e.g., Ukkonen, Lemström, and Mäkinen 2003; Lubiw and Tanur 2004); however, ours is designed to handle a continuous spectrum of pitch and time. We do not assume a fixed set of allowed pitches or time differences. On the other hand, we do assume that the input melodies are monophonic. Extending these methods to polyphonic music and arbitrarily complex pitch functions are interesting challenges for future study.

Acknowledgments

We wish to thank all participants of the Second Cuban Workshop on Algorithms and Data Structures, held at the University of Havana, 13–19 April 2003. We also thank Remco Veltkamp and Rainer Typke for bringing Arkin et al. (1991) to our attention. We appreciate the constructive comments by two anonymous referees and by two of the editors, Douglas Keislar and George Tzanetakis.

References

Aloupis, G., et al. 2003. "Computing a Geometric Measure of the Similarity Between Two Melodies." *Proceedings of the 15th Canadian Conference on Computational Geometry.* Halifax, Canada: Dalhousie University, pp. 81–84.

Arkin, E., et al. 1991. "An Efficiently Computable Metric for Comparing Polygonal Shapes." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13(3):209–216.

Bainbridge, D., et al. 1999. "Towards a Digital Library of Popular Music." *Proceedings of the Fourth ACM International Conference on Digital Libraries.* New York: Association for Computing Machinery.

Battey, B. 2004. "Bézier Spline Modeling of Pitch-Continuous Melodic Expression and Ornamentation." *Computer Music Journal* 28(4):25–39.

Carterette, E. C., and R. A. Kendall. 1994. "On the Tuning and Stretched Octave of Javanese Gamelans." *Leonardo Music Journal* 4:59–68.

Chen, A. L. P., et al. 2000. "Query by Music Segments: An Efficient Approach for Song Retrieval." *Proceedings of the IEEE International Conference on Multimedia and EXPO (II).* New York: Institute of Electrical and Electronics Engineers, pp. 873–876.

Clayton, M. 2000. *Time in Indian Music.* New York: Oxford University Press.

Cormen, T., et al. 2001. *Introduction to Algorithms.* New York: McGraw-Hill.

Cronin, C. 1998. "Concepts of Melodic Similarity in Music-Copyright Infringement Suits." In W. B. Hewlett and E. Selfridge-Field, eds. *Melodic Similarity: Concepts, Procedures, and Applications.* Cambridge, Massachusetts: MIT Press, pp. 187–209.

Dobrian, C. 2004. "Strategies for Continuous Pitch and Amplitude Tracking in Real-Time Interactive Improvisation Software." *Proceedings of the 2004 Sound and Music Computing conference (SMC04),* IRCAM, Paris, France.

Francu, C., and C. G. Nevill-Manning. 2000. "Distance Metrics and Indexing Strategies for a Digital Library of Popular Music." *Proceedings of the IEEE International Conference on Multimedia and EXPO (II).*

Ghias, A., et al. 1995. "Query by Humming: Musical Information Retrieval in an Audio Database." *ACM Multimedia* 95:231–236.

Groß, C., and T. K. Strempel. 1998. "On Generalizations of Conics and on a Generalization of the Fermat-Torricelli Problem." *American Mathematical Monthly* 105(8):732–743.

Hewlett, W. B., and E. Selfridge-Field, eds. 1998. *Melodic Similarity: Concepts, Procedures, and Applications.* Cambridge, Massachusetts: MIT Press.

Hofmann-Engl, L. 2002. "Melodic Similarity: A Conceptual Framework." *Proceedings of the 2nd International Conference on Understanding and Creating Music.* Naples, Italy: Seconda Universita degli Studi di Napoli.

Jones, A. M. 1954. "African Rhythm." *Africa: Journal of the International African Institute* 24(1):26–47.

Kapanci, E., and A. Pfeffer. 2005. "Signal-to-Score Music Transcription Using Graphical Models." *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI).* Edinburgh, UK. Available online at http://ijcai05.csd.abdn.ac.uk/.

Lemström, K. 2000. "String Matching Techniques for Music Retrieval." Ph.D. thesis, University of Helsinki.

Lu, L., H. You, and H. J. Zhang. 2001. "A New Approach to Query by Humming in Music Retrieval." *Proceedings of the 2001 International Conference on Multimedia and Expo.* New York: Institute for Electrical and Electronics Engineers, pp. 22–25.

Lubiw, A., and L. Tanur. 2004. "Pattern Matching in Polyphonic Music as a Weighted Geometric Translation Problem." *Proceedings of the 5th International Conference on Music Information Retrieval.* Barcelona, Spain: Universitat Pompeu Fabra, pp. 289–296.

Martinez, I. C. 2001. "Contextual Factors in the Perceptual Similarity of Melodies." *MikroPolyphonie: The Online Contemporary Music Journal.* Available online at http://www.mikropol.net.

Mo, J. S., C. H. Han, and Y. S. Kim. 1999. "A Melody-Based Similarity Computation Algorithm for Musical Information." *Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange.* New York: Institute of Electrical and Electronics Engineers Computer Society, p. 114.

Montfort, M. 1985. *Ancient Traditions, Future Possibilities: Rhythmic Training Through the Traditions of Africa, Bali, and India.* Kentfield, California: Ancient Future Music.

Morris, R. 1998. "Sets, Scales, and Rhythmic Cycles: A Classification of Talas in Indian Music." Paper presented at the 21st Annual Meeting of the Society for Music Theory, Chapel Hill, North Carolina, 3 December.

Müllensiefen, D., and K. Frieler. 2004. "Measuring Melodic Similarity: Human vs. Algorithmic Judgments." Paper presented at the 2004 Conference on Interdisciplinary Musicology, Graz, Austria, 15–18 April.

Nodal, R. 1983. "The Social Evolution of the Afro-Cuban Drum." *The Black Perspective in Music* 11(2):157–177.

Ó Maidín, D. S. 1998. "A Geometrical Algorithm for Melodic Difference." *Computing in Musicology* 11:65–72.

Ornstein, R. 1971. "The Five-tone Gamelan Anklung of North Bali." *Ethnomusicology* 15(1):71–80.

Reiser, A., 1978. "A Linear Selection Algorithm for Sets of Elements with Weights." *Information Processing Letters* 7:159–162.

Toussaint, G. T. 2004. "A Comparison of Rhythmic Similarity Measures." *Proceedings of the 5th International Symposium on Music Information Retrieval.* Barcelona, Spain: Universitat Pompeu Fabra, pp. 242–245.

Typke, R., et al. 2003. "Using Transportation Distances for Measuring Melodic Similarity." *Proceedings of the 4th International Symposium on Music Information Retrieval.* Baltimore, Maryland: Johns Hopkins University, pp. 107–114.

Ukkonen, E., K. Lemström, and V. Mäkinen. 2003. "Geometric Algorithms for Transposition Invariant Content-Based Music Retrieval." *Proceedings of the 4th International Conference on Music Information Retrieval.* Baltimore, Maryland: Johns Hopkins University, pp. 193–199.