

音樂內容查詢不匹配問題與檢索模式之研究

Vocabulary Mismatch Problems and Query Models in Content-Based Music Retrieval

曾元顯 輔仁大學 圖書資訊學系

資訊傳播與圖書館學, 第 6 卷, 第 4 期, 2000 年 6 月, 頁 35-48

摘要：

音樂資料過去是以書目資料的形式提供查詢、取得與利用，然而此種檢索的彈性相當有限。音樂內容查詢即在允許由記憶不全或表達不清的音樂片段進行資料查詢。然而音樂內容查詢會面臨查詢不匹配、查詢表達困難以及結果瀏覽檢視與篩選等問題。本研究將就這些問題，進行音樂資料特性分析，從而提出適當的查詢模式與使用介面、開發音樂內容特徵的分析與擷取技術、發展快速有效的索引與檢索模式，以解決音樂查詢不匹配問題，進而建立同時允許書目性資料查詢以及內容查詢的音樂檢索系統，並評估其成效。目前本研究已發展出一套架構在 WWW 上的系統，提供友善的檢索方法及便利的結果檢視與瀏覽。經由特殊的編碼方法與索引技術，系統容許任意音高的查詢及查詢條件中多音、少音、音高不準的隨機錯誤。而系統回應的畫面中，有查詢條件的試聽按鈕、自動提示的候選查詢條件、以及依近似程度排序的檢索結果。此系統的特點，是其關鍵旋律擷取的設計，除便利並加速瀏覽篩選的查詢過程外，也作為查詢提示、相關回饋的功能，進而有效提昇檢索的成效。

Abstract:

The problem of music retrieval based on surrogates of music, such as titles or composers has been solved by existing bibliographic retrieval systems. In contrast, retrieval of a piece of music based on the music contents, especially based on an incomplete, imperfect recall, has not yet fully explored. There are several problems specific to content-based music retrieval, such as various types of query mismatch, difficulty in query formulation, result inspection, and browsing. This research is aimed at proposing solutions to these problems and developing a pragmatic system for both bibliographic and content-based retrieval. Currently a retrieval system has been developed on the Web and provides flexible user interface for query formulation and result browsing. With the proposed pitch encoding and n-note indexing methods, allowance for query in any key and approximate matching in sub-linear time is achieved. The most distinct feature of this system is the key melody extraction function for query suggestion and effective retrieval. Evaluation of the retrieval effectiveness of the proposed methods is also provided.

關鍵詞：音樂內容查詢 (music content-based retrieval)、哼唱查詢 (query by sing)、關鍵旋律擷取 (key melody extraction)、成效評估 (effectiveness evaluation)

一、前言

音樂資料過去是以書目資料的形式提供查詢、取得與利用。圖書館或資料提供者將音樂資料組織、分析，依作曲者、音樂形式等項目分門別類，進而建立音樂書目資料庫。使用者依照這些項目雖可檢索到資料，然而此種檢索的彈性仍然有限。例如，我們常常僅記得音樂的片段旋律，卻不記得曲名或作曲家，此時便無法以這些書目資料查得音樂原件（即 CD、錄音帶、錄影帶、樂譜、文件等音樂載體）。又如音樂創作者一有靈感，腦海中響出一段旋律，想要查詢過去是否有近似的音樂創作時，也無法以書目性資料庫就音樂內容進行查詢。

電腦網路的發達，使得各種媒體資訊的出版、傳播、與取得過程更加便利。越來越多的資訊出現在網路上，使用者可免費或付費利用。網路資源便於取用 (accessibility) 的特性，不僅讓使用者克服了時間、空間的障礙找到資料，取得全文文件，同時也刺激了資訊提供者加速現有資料的數位化，以提供網路化的資訊服務。

如今音樂資料也有相當多的數位化檔案出現在網路上，供人下載利用。例如，以 MIDI (Musical Instruments Digital Interface) 形式記載相當於完整樂譜內容的數位檔案，即如雨後春筍般在網站上出現。將音樂資料以 MIDI 數位形式編輯製作的族群，有的是音樂的愛好者，將原有譜曲以自己喜好的方式詮釋或配樂後，送上網路與人分享；有的是音樂的創作者，利用網路無遠弗界、時空無礙的特性，達到出版、散佈其創作的目的。因此，在數位音樂資料越來越豐富的時代，提供有別於書目性資料，而以內容為主 (content-based) 的查詢方式，變得越來越重要。

然而，要建立一個實用性高並容許以內容查詢的音樂檢索系統需要克服數項問題。首先，一首曲子可以用任何一個音高起唱，而還是同樣的曲子。這情形就好像是在 KTV 裡唱歌唱不上去，就調整伴奏曲的 key 一樣。因此，音樂檢索系統必須能允許使用者以任何音高查詢，並能比對到正確的曲調，亦即達到「音調無關」(key-independent) 的查詢。其二，使用者查詢時可能無法正確記憶曲調而輸入不完整的片段，甚至不完全正確的曲調。此種情形，在傳統資訊檢索領域稱為「字彙不匹配問題」(vocabulary mismatch problem)。亦即，對同樣的概念，使用者所下的檢索詞與系統所記錄的索引詞不同，而造成檢索失敗的情形。因此，「近似比對」在音樂檢索系統裡特別重要，以便允許使用者走音、多個音或少個音等隨機錯誤的查詢狀況。此外，傳統資訊檢索領域裡常用來解決字彙不匹配的方法，如相關詞提示、相關回饋等也都對音樂檢索有所幫助。其三，音樂檢索還有其他特殊的字彙不匹配問題，如震音、碎音、琶音等裝飾音或其他指法技巧 [1]。這些音容易被使用者誤聽或記錯，而將數個短音輸入成一個長音，或將一個長音輸入成數個短音。這些情況容易造成連續錯誤，而比前一種隨機錯誤造成的字彙不匹配問題更嚴重。除此之外，音樂內容的查詢條件並不容易表達，尤其對音樂訓練較少的使用者尤然。因此音樂檢索系統需更為注重任何有助於輸

入、瀏覽、篩選等功能的設計，以及友善便利的使用環境。

本研究將針對上述問題，分析音樂資料的特性，發展適當的查詢模式與使用介面，開發音樂資料內容特徵的分析與擷取技術，發展快速有效的索引與檢索模式，解決音樂查詢不匹配問題，進而建立同時允許書目性資料查詢以及內容查詢的音樂檢索系統，並評估其成效。由於未來音樂數位資料的累積將越來越快，本研究的成果將能夠讓音樂數位資料的分析、擷取、索引、檢索與取用更具自動化，加快音樂數位化的進程，從而發展更多數位音樂的利用方式與應用。

音樂資料內容的分析處理，在過去的資訊檢索研究中較少被觸及，音樂資料獨特的性質，對資訊檢索領域而言，乃新的問題與挑戰。因此，本研究所提的內容具有相當的學術研究價值。又由於發展並建構這樣的檢索系統，可實際應用於現今的網路環境，因此本研究也具備了實用價值。過去三、四年來，歐、美、日等國紛紛進行圖書資料的數位化研究工作，以便利人民利用資訊，進而提昇國家整體的競爭力。本研究所提內容，亦是數位化圖書館所需的核心技術之一，對便利民眾利用資訊、提昇音樂的教育、研究與創作環境，將有所助益。

二、過去相關研究之概況

目前國內外雖有音樂檢索的研究，但針對上述各項音樂檢索課題，提出完整解決方案者並不多見。Pfeiffer 等人 [2] 發展出一套工具可以從訊號中擷取基音 (fundamental frequency) 作為特徵，並以相關運算 (correlation) 作為「範例查詢」(query by example) 的比對方式，找出廣告記錄中某一廣告音樂的發生次數。Wold 等人 [3] 則從音樂訊號中，計算出響度、音高、亮度、頻寬、和諧度等作為內容特徵，對聲音作分類，以進行相似度查詢。然而這些音響特徵的語意程度太低，一般人並不熟悉，因而無法善加利用進行音樂檢索。Foote [4] 也從音樂訊號中擷取類似的特徵，他用一種樹狀量化結構來降低近似比對的計算量。以上這些方法多少都以「範例查詢」的方式，避開音樂檢索查詢不易下達的問題。

因此，國際上另一個研究方向，是追求更直接的查詢方式，例如以「彈奏來查詢」(query by playing)、「以哼唱來查詢」(query by singing) 等。Hawley [5] 發展過一種系統，可以允許使用者從 MIDI 合成器的鍵盤上直接輸入一串音符，但其查詢方法僅允許從樂曲開頭處比對。Ghias 等人 [6] 則發展能從麥克風直接哼唱的系統。此系統能將麥克風的音訊輸入，轉換成字串，並以「升音、降音、平音」三個符號將輸入編碼成旋律輪廓 (melody contour)，以允許任何音調的查詢。此旋律輪廓再以動態近似字串比對演算法與 183 首曲子比較。其系統能容許多音、少音、音符位置對調等情況。但由於以相當少的符號對音樂作編碼，造成樂曲間的詫異性變小，影響大量資料時的檢索效果，且其動態近似字串比對相當費時，是此系統的兩個主要缺點。McNab [7] 也發展出一個允許以哼唱輸入的類似系統。此系統架構在 Web 環境下，提供多種比對方式，如提供旋律及節奏的近似查詢。不過其利用動態規劃製作的近似比對，查詢 9400 首樂曲需時 21 秒。未

來他們還規劃擷取主題或關鍵旋律，以加快比對速度。

國內與本研究相關的研究有清華大學音樂資料庫方面的數項研究。其中一項也發展了以哼唱作為查詢輸入的技術，但目前其類似 Hamming 距離的比對方式，所提供的查詢彈性較小 [8]。另外數項，則運用和旋、節奏、旋律等音樂特性作為內容特徵與查詢條件，所採用或發展的技術包括 1-D 串列、PAT-tree 等 [9-11]。不過這些研究中，在查詢提示、相關回饋等提昇檢索成效的技術上，運用得較少。

我們目前則發展出一套系統，架構在 WWW 上，提供友善的檢索方法及便利的結果檢視與瀏覽。使用者可輸入簡譜做查詢、或利用網路上的音樂編輯軟體以哼唱方式輸入查詢、甚至不需輸入任何音樂資料而選用系統提示的關鍵旋律進行「範例查詢」。透過特殊的編碼方式與索引技術，系統容許任意音高的查詢及查詢條件中多音、少音、音高不準的隨機錯誤。目前的測試顯示，1941 首音樂的近似比對只需約一兩秒時間。而系統回應的畫面中，有查詢條件的試聽按鈕、自動提示的候選查詢條件、以及依近似程度排序的檢索結果。點選其中的試聽按鈕會播放以查詢條件即時作成的音樂供使用者試聽確認。若此次查詢得不到正確結果，則可點選候查詢條件進一步檢索。此外每一筆結果，都可任意點選以進行「範例查詢」。最重要的是，此系統有關鍵旋律快速擷取的設計，除便利並加速瀏覽篩選的查詢過程外，也作為查詢提示、相關回饋的功能，進而有效提昇檢索的成效。就上述的分析瞭解，此系統的近似比對能力、檢索速度、以及所能提供的查詢彈性與檢索功能，都有異於上述其他系統的表現。

三、音樂內容查詢之問題與解決方法

如前所述，音樂內容檢索的主要困難在於查詢不匹配 (query mismatch) 查詢表達困難以及結果瀏覽檢視與篩選的問題。在查詢不匹配的問題方面，有音調不匹配 (key mismatch) 音高錯誤 (pitch error)，多個音、少個音 (note insertion, or note deletion) 以及碎音或合併音 (fragmentation, or consolidation) 等問題。例如，若以簡譜 1, 2, 3, ... 代表 Do, Re, Mi, ...，則「祝你生日快樂」這首曲子的前幾個音符 (note) 可以表達成：「1 1 2 1 4 3」，也可以用較高的音調唱成「5 5 6 5 1+ 7」，其中「1+」代表高八度的 Do。顯然以這兩種旋律字串去做比對時，會完全沒有交集，而造成比對錯誤的情形。這種情形，即為音調不匹配 (key mismatch) 的情況。又如，筆者本身曾一度將上述曲調表達成：「1 1 2 1 5 3」，一直到為了確認無誤，彈奏了鋼琴以後，才發現連自己也都會表達錯誤，將原來 4 (Fa) 的音，唱成 5 (So) 的音。此乃音高錯誤 (pitch error) 問題。此外，在進行一些檢索試驗時，筆者將「鬥牛士之歌 (Toreador Song)」的一段旋律表達成：「5 6 5 3 3 3 2 3 4 3」，請筆者的妻子來檢索時，她卻表達成：「5 6 5 3 3 4 3 2 3 4 3」，兩者之間差了一個短促的 4 (Fa) 的音。此乃兩者的認知或記憶不同，而造成的多音或少音的現象。最後，所謂碎音 (fragmentation)，乃指數個音合

起來的拍子 (duration) 與某個單音的拍子相同，而此數個音的平均音高與此單一音的音高近似 [1]。至於合併音 (consolidation) 則與碎音的情況相反。例如「布穀鳥」的旋律，可能原為：「3 1 3 1 3 5 5 3 1 3 2」，但演奏者可能將其活潑化，而表達成「33 1 33 1 33 5 5 3 1 3 2」，其中加底線相連的音，代表相連的音合計的拍子 (duration) 與原來一個音的拍子一樣長，但由於以連續的兩個或多個短音代替原來的一個音，會造成樂曲變得輕快的感覺。這就是碎音的效果。顯然以此兩旋律字串直接做比對時，兩者的字串長度不同，也容易造成比對上的錯誤。

以上這些音樂查詢不匹配 (mismatch) 的問題，有些源於檢索者對樂曲的記憶錯誤 (無法完全正確的回憶) 有些源於閱聽音樂時的認知 (perception) 錯誤 (無法完全正確的記憶) 有些源於表達查詢條件時的錯誤 (受限於音樂的訓練與素養，無法完全正確表達心中的樂音) 甚至有些是檢索者與樂曲的演奏者之間，對原創曲譜或對原作曲者的詮釋不同，而造成的錯誤。

針對上述問題，我們採用一種「音調無關」(key-invariant) 的編碼方式將音樂旋律編碼，使其允許任意音調的查詢，以解決音調不匹配的問題。目前的設計是利用 Pn, Mn, R 等符號來編碼，其中 P 代表目前的音比前一個音高、M 代表低、R 代表相同的音高、而 n 代表上升或下降半音 (semi-tone) 的個數。例如「小蜜蜂」一曲，其前面數個音為「Sol Mi Mi Fa Re Re」，以標準音名表示為「G4 E4 E4 F4 D4 D4」(其中的 4 代表第四個八度音)，編碼以後為「M3 R P1 M3 R」。如此，使用者不管以什麼調唱小蜜蜂時，如「Re Si Si Do La La」(「D4 B3 B3 C4 A3 A3」)，都會被編成相同的字串，而達到「音調無關」情況。然而當查詢輸入因為音高不準、或多音、少音的現象而發生 m 個隨機錯誤時，即原字串有 m 個隨機錯誤時，我們可以證明，編碼後的字串最多會有 2m 個錯誤，其錯誤率為 2。編碼後的字串錯誤率變高 2 倍，這是此種編碼方式需要付出的代價。因此，我們在做旋律字串的比對時，就需要容錯能力更強的檢索或近似比對模式。

受閱聽者的聽力、音樂表達能力、以及音樂資料本身被詮釋方式的影響，音高不準、多個音、少個音的不匹配 (mismatch) 現象將相當普遍，因此近似比對 (approximate match) 是相當重要的檢索需求。過去的研究大多利用動態規畫 (dynamic programming) 技術來作音樂旋律的比對。動態規劃的比對可顯示兩字串之間的「編輯距離」(edit distance)。所謂編輯距離就是將一個字串利用「插入」、「刪除」與「代換」的動作轉成另一個字串「所需最少步驟的個數」(或是「所需最少的計算成本」)。例如：「cause」與「causal」的編輯距離為 2，因為把 cause 的 e 代換成 a，再插入 l 於最後面就變成 causal 了。動態規劃的演算法可表達如下 [12]：假設有兩字串 A 與 B，長度各為 n 與 m。將兩字串從頭比對起，則比對到 A 的第 i 個字 (以 A[i] 表示) 與 B 的第 j 個字 (以 B[j] 表示) 的編輯距離為

$$d[i, j] = \min(\begin{array}{l} d[i-1, j] + w(A[i], 0), \\ d[i-1, j-1] + w(A[i], B[j]), \\ d[i, j-1] + w(0, B[j]) \end{array})$$

其中 $\min(X, Y, Z)$ 表示取 X, Y, Z 三個數中最小的值，而初始值為

$$d[0, 0] = 0$$

$$d[i, 0] = d[i-1, 0] + w(A[i], 0), 1 \leq i \leq n$$

$$d[0, j] = d[0, j-1] + w(0, B[j]), 1 \leq j \leq m$$

另外，函數 $w(X, Y)$ 的意義為

$w(A[i], B[j])$ ：表示將 $A[i]$ 代換成 $B[j]$ 的計算成本

$w(A[i], 0)$ ：表示插入 $A[i]$ 的計算成本

$w(0, B[j])$ ：表示刪除 $B[j]$ 的計算成本

我們以 $A=adec$ ， $B=adecdecf$ 為例，且假設代換、插入、與刪除的計算成本都為 1，則 $d[i, j]$ 可以表示在底下矩陣的第 i 列的第 j 行：

A \ B	a	d	e	c	d	e	c	f
a	0	1	2	3	4	5	6	7
d	1	0	1	2	3	4	5	6
e	2	1	0	1	2	3	4	5
c	3	2	1	0	1	2	3	4

由於我們可以在上面矩陣的最後一列的第四行發現一個編輯距離為 0 的結果，所以我們可以知道 A 是 B 的一個子字串 (sub-string)，且其在 B 中的位置是從第 1 個字到第四個字。然而此種方法的計算複雜度為 $O(Kmn)$ ，其中 K 為資料庫中所有旋律的總數，且當子字串 A 在 B 的中間部分時，則無法如前述方法看出 A 為 B 的子字串。例如 $A=adec$ ， $B=acdadecf$ 時，則其距離矩陣為：

A \ B	a	c	d	a	d	e	c	f
a	0	1	2	3	4	5	6	7
d	1	1	1	2	3	4	5	6
e	2	2	2	2	3	3	4	5
c	3	2	3	3	3	4	3	4

最後一列沒有一行的值為 0。此種現象主要是 A 與 B 的長度不同，而且此動態規劃的比對方式僅適合兩個字串從頭比對起，而無法從任意段落比對起。

改善此種缺點的方法可將上述的比對方式作些許的修改，亦即比對時每次都限定在相同的長度段落來比對即可，本文中稱此種修改過的動態規劃比對方式為視窗動態規劃法 (Windowed Dynamic Programming)，因為我們以相同的長度作為的視窗，在此視窗內還是運用原來的動態規劃比對法。以上述的 A 與 B 字串為例，此修改過的比對方式得到的距離矩陣其最後一列的結果如下：

	B	a	c	d	a	d	e	c	f
A									
a									
d									
e									
c				3	2	2	0	2	

從最後一列的第 7 行的值為 0 可知，A 為 B 的子字串，且 A 是 B 的第 4 個到第 7 個字的子字串。由於此種方式每次都要比對資料庫中全部的樂曲，因此較難預先做好任何的資料處理，且其計算複雜度更高，為 $O(Kmn^2)$ ，因此查詢時的反應速度會隨資料量的增加而變慢，以致到無法接受的地步。

在本研究中，我們主要是利用 n-note 的索引方式（類似 n-gram 的作法 [13]），以及反向索引檔法（inverted file）與雜湊函數法（hash function），來同時解決近似比對以及查詢效率的問題。我們的 n-note 索引方式，就是在音樂資料庫的旋律字串中，對長度為 n 的任意旋律片段及其長度為 m 的子片段（其中 $1 \leq m \leq n$ ），都將其索引起來。檢索者以某個自訂的旋律字串查詢時，也對其做相同的處理，然後再將這些子片段與音樂資料庫中的子片段做比對。比對時以下的公式計算資料旋律 d_i 與查詢旋律 q_j 之間的近似程度：

$$Sim(d_i, q_j) = \frac{\sum_{k=1}^T d_{i,k} q_{j,k}}{\sum_{k=1}^T q_{j,k}}$$

其中 T 代表所有 m-note 的總數； $d_{i,k}$ 是資料旋律 d_i 第 k 個 m-note 的權重，其值不是 1 就是 0，代表子片段 $d_{i,k}$ 在資料旋律 d_i 當中有出現或是沒出現；而 $q_{j,k}$ 是查詢旋律 q_j 第 k 個 m-note 的權重，其值為 $tf(2(m-1)+1)$ ，其中 tf 為詞頻，代表其在查詢旋律中出現的次數，換言之，查詢旋律中較長的子片段、或出現較多次的子片段會有較大的權重。上述相似度計算公式的特色之一，就是當查詢旋律是某個資料旋律的子字串時，那麼該相似度會有最大的值（maximum value）；否則，其值會隨相似的程度而遞減。如此，我們可以得到近似字串比對的效果。當然，由於這些子片段的總數量可能非常大，例如，在總共 1183 首音樂旋律字串當中，以 3-note 的規格來索引時，其子片段（即所有的 m-note， $1 \leq m \leq 3$ ）數量即高達 63213 個（即 $T=63213$ ），在計算相似度時，會影響查詢的反應速度。所以我們採用反向索引檔法（inverted file）與雜湊函數法（hash function）的技術，預先做好計算，來提升檢索比對時的速度。過去我們的研究顯示，類似的作法可有效對付大量的資料。如檢索 35 萬筆書目資料時，61% 的檢索可在 1 秒內完成，79% 的檢索可在 2 秒內完成，87% 的檢索可在 3 秒內完成 [14]。在上述的方法中，近似比對的容錯能力與 n-note 的參數 n 關係密切。底下，我們將透過評估實驗，探討各種參數對檢索成效的影響，以決定最佳的參數設計。

至於大量連續音錯誤的問題，本研究中分析探究各種發生的狀況，透過查詢提示（query suggestion）、查詢擴展（query expansion）、相關回饋（relevance

feedback) 等資訊檢索技術來處理。目前我們已分析歸納出一些案例，以候選查詢 (query candidates) 的形式自動提示使用者作進一步的查詢。初步試驗顯示此方式確實可以處理目前我們發現的大量連續音錯誤的問題。未來將繼續根據音樂理論分析探究各種狀況，提出相關問題的解決之道。

除此之外，此系統還具備一套關鍵旋律自動擷取的功能，用以擷取重要的音樂特徵。由於重複性 (repetition) 是音樂創作的規則之一 [15]，且重複片段經常是樂曲中重要的部份，是作曲者要強調的部分，也是閱聽者較容易記住的部份，使用者檢索時，也較容易以此片段進行查詢，因此是相當值得擷取的重要特徵。因為此種內容特徵具備代表性，可視為樂曲的「關鍵旋律」(key melody) 或「主題旋律」(theme)。與文字檢索對比，關鍵旋律具有「摘要」或「關鍵詞」的效果，在互動式的檢索環境下，可加快瀏覽、篩選的過程，亦可用在查詢提示、查詢擴展、相關回饋等功能上，進一步提昇檢索成效。過去有擷取重複片段做為重要音樂內容特徵的研究 [16]，其計算複雜度為 $O(n^2)$ ，其中 n 為音樂資料的長度。我們之前亦開發了擷取重複字串作為關鍵詞的技術，可直接用在音樂資料上，作為關鍵旋律自動擷取的方法 [17-18]。其計算複雜度為 $O(mn)$ ，其中 n 如前述、而 m 為最長關鍵旋律的長度。初步的實驗顯示，流行樂曲的 m 大約為 n 的 20%，而古典音樂的 m 大約為 n 的 14%。此外，值得一提的是，在擷取重複字串的研究方面，已有快速的方法可在線性的時間內 ($O(n)$) 擷取出非重疊

(non-overlapping) 的重複字串。至於可重疊的重複字串方面 (例如：字串 ABCABCBCA，其中 ABCBCA 以重疊的方式出現了兩次 加底線的一次，斜體字的一次)，目前還沒有線性時間的擷取演算法發展出來，甚至學者們還懷疑是不是有線性時間的演算法存在 [19]。而我們的方法是可以擷取重疊的重複字串的演算法，所以其 $O(mn)$ 的計算複雜度算是相當快速，甚至可能是最佳 (optimal) 的演算法。

關鍵旋律擷取出來後，可組成一個關鍵旋律資料庫。使用者在查詢音樂檔案資料庫前，先查詢此關鍵旋律資料庫，如此，即使在較低的比對分數下，也可以得到正確的結果，而沒有太多不相關資料的干擾。這是因為關鍵旋律的資料量較少、較獨特，彼此間的差異較大，因此可以容許查詢條件更大的誤差，而仍可得到結果。一旦正確的關鍵旋律找到後，依此找其對應的音樂檔案將不成問題，因為關鍵旋律是原音樂檔案的片段，完全沒有誤差，所以不會產生查詢不匹配的問題。其整體的效果是，查詢反應時間與檢索失敗的情況降低，而近似比對的容錯能力更強。目前我們初步的試驗，驗證了此項觀點 [20]。

關鍵旋律除了作為一個容錯能力更強的資料庫外，也可用階層式的架構做分類處理，配合資訊視覺化 (information visualization) 的技術，以二維或三維的圖形介面提供導覽 (navigation) 與篩選 (selection)，甚至因此可以做到不必輸入任何音樂資料，即可進行查詢的使用介面，這對音樂背景較弱的使用者將是極大的查詢利器。未來我們將探討此種資訊視覺化的技術，以應用在音樂檢索系統中。

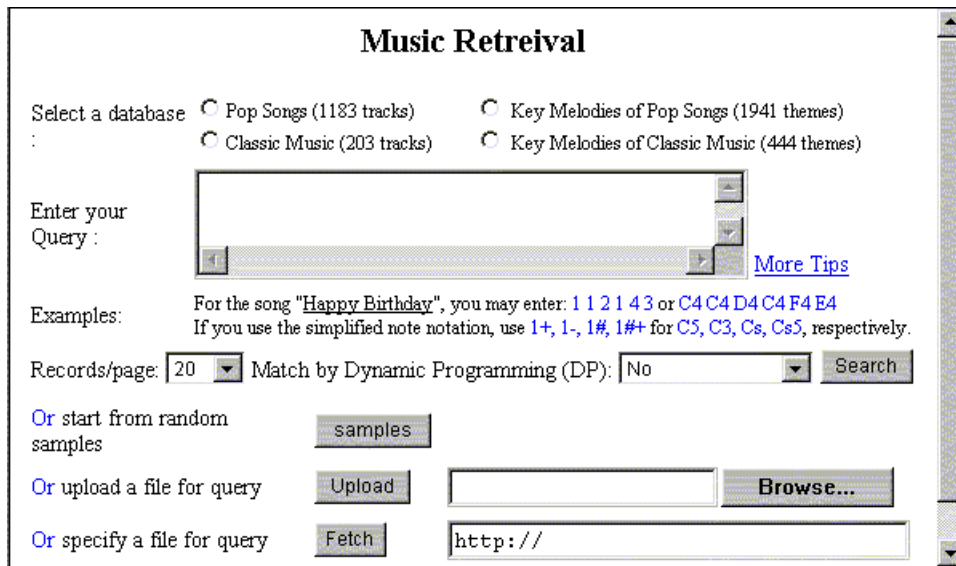
四、系統展示

拜網路發達之賜，目前已從好幾個音樂網站蒐集到一些數位音樂檔案，這些檔案都是 MIDI 格式。每一個 MIDI 檔案都記錄了一首曲子裡每一個樂器的樂譜，所以從 MIDI 檔案可以擷取出音樂的旋律 (melody)、節奏 (rhythm) 與和弦 (chord)。由於旋律構成了音樂的主要內容，所以目前我們僅擷取出旋律，以供使用者做音樂內容的查詢。有時候 MIDI 檔案的製作者也會將一些書目性的文字資料，例如音樂曲名、作曲者、使用到的樂器名稱、甚至歌詞都存放在 MIDI 檔中，如果有這些文字資料，我們也會將其擷取出來，讓使用者可以查詢。

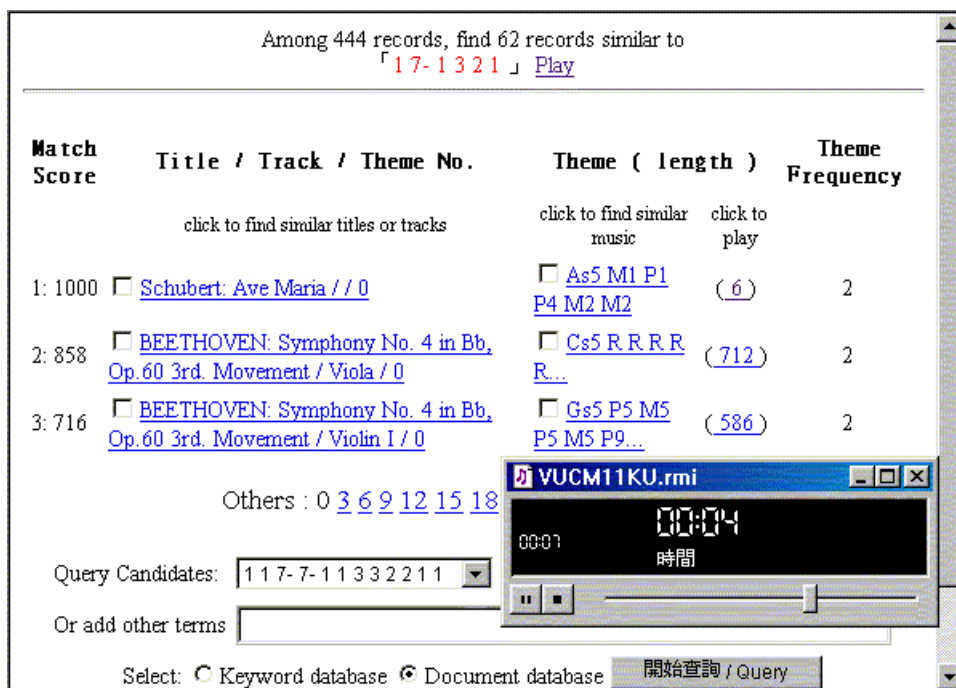
根據前面的描述，我們實做出一個系統，可提供實際的檢索與測試 (其網址在 <http://www.lins.fju.edu.tw/~tseng/musicir/>)。如圖一所示，此檢索系統裡有二個音樂檔案資料庫、二個關鍵旋律資料庫：

- 一、「流行音樂資料庫」，目前約 1000 首各式樂器組成的 135 首中文歌曲；
- 二、「流行音樂關鍵旋律資料庫」，乃擷取流行音樂的關鍵旋律而來，約 1900 首主題旋律；
- 三、「古典音樂資料庫」，目前約 200 首各式樂器組成的 31 首古典音樂；
- 四、「古典音樂關鍵旋律資料庫」，擷取古典音樂的關鍵旋律而來，約 400 首主題旋律。

使用者可用簡譜的方式輸入旋律。如前述的範例「小蜜蜂」，可輸入「5 3 3 4 2 2」查詢，其中數字 1 到 7 代表 Do 到 Si 七個音，使用者可用 1-、1+、1#、1+#，來對 Do 表示低八度音、高八度音、升半音、以及高八度音並升半音。若不熟音律，使用者也可以透過一些網路上的音樂軟體 (如：AKoff Music Composer [21])，以哼唱的方式錄製 MIDI 檔案，再傳送上來查詢，或直接給定網路上的某個音樂檔案位址，以「範例查詢」的方式進行。再不然，可以點選「samples」按鈕，由系統隨機提示一些關鍵旋律，再進行「範例查詢」。此外，除了旋律的查詢外，使用者也可在同樣的輸入框裡輸入完整或部分的曲名、作曲者、樂器名稱或歌詞，系統可以自動辨識查詢的字串是旋律還是文字，做適當的近似查詢比對，提供甚至比一般的音樂書目資料庫更好的功能。因此，這是一個既可用曲目資料查詢，也可同時用樂曲旋律來查詢的系統。



圖一：音樂內容檢索系統查詢介面



圖二：查詢結果範例

使用者的查詢結果如圖二所示。點選其中的「play」超連結，可讓使用者傾聽自己所下的查詢旋律，以確定所給的音調、音高是否與心中預期相同。候選查詢（query candidates）則是根據前述的音樂特性分析，由系統自動擴展（expand）查詢條件而給的候選條件，若使用者無法以現有查詢條件找到正確資料，則可點選候選條件。這對連續音不匹配的查詢非常有效。此外，每一檢索結果均按近似程度排序（近似程度表現在 Match Score 的欄位），而且均可點選、聆聽、篩選，以進一步查詢近似的旋律。至於圖二中的樂曲名稱與作曲家名稱部份，有的是從

音樂檔案內容或檔名擷取出來（如果檔案製作者有加入文字描述的話），有的是由 Robot 網頁擷取軟體的輔助以半自動的方式建立。因此系統亦可用中、英文字串查詢曲名、作曲家或樂器名稱等，而同時具備書目性資料庫與內容查詢資料庫的特性。未來將利用 Robot 技術，在網路中蒐集更多音樂檔案，持續擴展系統的資料與檢索功能。

五、檢索模式成效之評估與比較

雖然此系統提供友善的介面與彈性的查詢方式，然而系統的檢索成效仍舊是使用者最在意的部分。因此，我們透過成效評估實驗，來檢驗上述各種查詢不匹配的解決方案，看看他們在實際的情況下是否能發揮效果。

由於音樂資料與文件資料的特性不同，相同主題（subject）的音樂可能有完全不同的創作表達方式，因此音樂內容檢索的成效無法用文字檢索的標準：查全率（recall rate）與查準率（precision rate）來衡量。取而代之的，是讓使用者按照自己對音樂的回憶，給定一段旋律，然後檢驗系統能否很快的找到該首曲子。

針對音調不匹配（key mismatch）的問題，我們可比較經過編碼的旋律，其被檢索的成效是否優於沒有經過編碼的旋律。而不同的索引與檢索模式，可以比較對音高錯誤（pitch error）、多音（note insertion）、少音（note deletion）等隨機錯誤的容錯能力。至於碎音（fragmentation）、合併音（consolidation）或其他音樂特性對檢索成效的影響，則可分析使用者的查詢與資料庫的旋律，來瞭解查詢不匹配的情況。最後，對於關鍵旋律的擷取與運用是否能夠提升檢索成效，則可比較有沒有運用到關鍵旋律來試驗。因此，我們一共有八種檢索模式要做比較。如表一所示：檢索模式 1 與 2 在比較不做旋律編碼、且以 2-note 方式索引時，有沒有運用關鍵旋律，對檢索成效的影響。如前所述，系統自動擷取出的關鍵旋律，由於資料量較少、較獨特，彼此間的差異較大，可能可以容許查詢條件更大的誤差。因此關鍵旋律的運用為兩階段的查詢，第一階段先查關鍵旋律資料庫，第二階段則點選正確或近似的關鍵旋律以找到其對應的音樂檔案。如果沒有運用關鍵旋律，則表示使用者做單一階段的查詢：即一開始就選擇原來的音樂資料庫，直接進行查詢。如同檢索模式 1 與 2 一樣，3 與 4、5 與 6、以及 7 與 8 都在相同的情況下比較關鍵旋律的運用，是否會有比較好的檢索結果。至於檢索模式 1 與 3、2 與 4 則在相同情況下比較「音調無關」（key-invariant）的旋律編碼對音調不匹配的查詢是否有效。而檢索模式 3 與 5、4 與 6 在比較 n-note 索引法其參數 n 對檢索成效的影響。最後，檢索模式 3 與 7、4 與 8、5 與 7 以及 6 與 8 在比較 n-note 索引法與視窗動態規劃法（Windowed Dynamic Programming，WDP）到底那種方法有較佳的容錯比對能力。

檢索模式 (mode)	資料庫	旋律編碼	索引方法
1	Key melody	No	2-note
2	Original	No	2-note
3	Key melody	Yes	2-note
4	Original	Yes	2-note
5	Key melody	Yes	3-note
6	Original	Yes	3-note
7	Key melody	Yes	No, match by WDP
8	Original	Yes	No, match by WDP

圖一：八種檢索模式的規格

我們邀請了八位受試者參與此項實驗。這八位受試者中，有三位過去有學過數年的鋼琴，一位有學過一年的吉他，其他四位則僅有中小學音樂課程的基本訓練。實驗進行時，每位受試者都給一份資料庫中的樂曲名單，此名單上列了 30 首古典音樂與 135 首中文流行音樂。受試者從中挑選自己喜歡的樂曲，依照他們自己的記憶給定一段旋律作為查詢條件。如果他們不能立刻回憶起其中的任何片段，實驗中則允許他們試聽其中一小段隨意挑選的音樂片段幫助他們回憶，然後根據他們的認知進行查詢。由於查詢結果依近似程度由大到小排序，實驗中建議受試者檢視其中的前 100 筆結果。如果前 100 筆結果找不到他們想要找的樂曲，這次查詢則視為失敗。我們就分析比對失敗的因素，並將原因告訴受試者，受試者再據以修正下另一次的查詢。例如：在檢索模式 1 中，如果受試者無法掌握正確的音調，我們就告訴受試者在資料庫中該首曲子起始的調子。如此反覆進行，一直到前 100 筆中出現正確的查詢結果。這樣的分析與修正，可以讓我們在實驗的過程中順便分析出更多的查詢不匹配問題，使系統將來可以做出更好的自動查詢提示。在受試者實際試驗前，他們可以用兩首曲子做練習，以熟悉整個實驗的流程與細節。

表二列出了八位受試者自訂的 20 筆查詢。針對每一首查詢旋律，受試者都被要求做出兩個查詢字串：一是用在檢索模式 1 與 2 上，以絕對音高的標準音名查詢；另一個是用在檢索模式 3-8 上，以簡譜的形式進行查詢。表二的第二欄顯示：受試者是第一次查詢就找到資料（type 的內容為 I，表示 Initial query 之意）或是經過至少一次的修改才找到資料（type 的內容為 M，表示 modified query 之意）。表二的第三欄是查詢字串，第四欄是查詢字串的長度。

這 20 首查詢當中，其中前七首為古典音樂的查詢，其他為中文流行音樂的查詢。以古典音樂為例，被查詢的樂曲依序為比才的鬥牛士之歌（Toreador Song by Bizet）貝多芬的第四鋼琴奏鳴曲（Piano Sonata No. 4 by Beethoven）舒伯特的聖母瑪莉亞（Ave Maria by Schubert）格雷哥의早晨（Morning Mood by Grieg）以及維瓦第的四季第一樂章（The Four Seasons 1st movement by Vivaldi）共五首。這是因為有兩位受試者選擇了鬥牛士之歌做查詢，分別列在第 1 及第 2 筆查詢；另外有兩位受試者選擇了貝多芬的第四鋼琴奏鳴曲做查詢，分別列在第 3 及第 4

筆查詢；其他三首古典音樂的查詢則列在第 5 到第 7 筆。讀者可稍微注意一下，第 1 及第 2 筆查詢是針對相同的樂曲、相同的片段進行的，但不同的受試者不同的音樂訓練下了不同的查詢字串。另外，第 3 及第 4 筆查詢也是針對相同的樂曲相同的片段，但查詢的字串長度稍有不同。

song	type	Query	length
1	I	G5 A5 G5 E5 E5 E5 D5 E5 F5 E5	10
	I	5 6 5 3 3 3 2 3 4 3	
2	M	C6 C6 D6 D6 C6 C6 A5 A5 A5 A5 G5 A5 B5 A5	15
	I	1+ 1+ 2+ 2+ 1+ 1+ 6 6 6 6 6 5 6 7 6	
3	I	G5 E5 C5 E3 G5 C6	6
	I	5 3 1 3 5 1+	
4	M	As5 G5 D5 G5 As5 C6 E6 D6 C6 E6 As5	12
	I	5 3 1 3 5 1+ 3+ 2+ 7 1+ 3+ 5	
5	M	As5 A5 As5 D6 C6 As5	6
	I	1 7- 1 3 2 1	
6	M	B4 Gs4 Fs4 D4 Fs4 Gs4	6
	I	5 3 2 1 2 3	
7	M	Gs6 Gs6 Gs6 Fs6 E6 B6	6
	I	3 3 3 2 1 5	
8	M	E5 D5 C5 C5 D5 E5 G5 A5 B5 C6 C6 A5 G5 E5 G5	15
	M	3 3 2 2 1 1 1 1 2 2 3 3 5 5 6 6 7 7 1+ 1+ 1+ 1+ 6 6 5 5 3 3 5 5	30
9	M	C6 C6 C6 C6 A5 G5 A5 F5 F5 G5 A5 D5	12
	M	5 5 5 5 5 5 5 5 3 3 2 2 3 3 1 1 1 1 2 2 3 3 6- 6-	24
10	M	Fs6 A6 B6 B6 A6 A6 Fs6 E6 D6 E6 Fs6 E6 D6 B5	14
	M	3 3 5 5 6 6 6 6 5 5 5 5 3 3 2 2 1 1 2 2 3 3 2 2 1 1 6- 6-	28
11	M	E5 G5 E5 E5 D5 E5 E5	7
	M	3 3 5 5 3 3 3 3 2 2 3 3 3 3	14
12	I	C6 D6 E6 G6 G6 G6 E6 D6 D6 D6 D6 E6 D6	13
	I	1 2 3 5 5 5 3 2 2 2 2 3 2	
13	M	G5 D6 D6 E6 C6	5
	I	5 2 2 3 1	
14	M	A5 E5 G5 G5 E5 D5 C5 D5 E5 A5	10
	I	6 3 5 5 3 2 1 2 3 6	
15	M	C5 C5 C5 C5 C5 D5 C5 D5 D5 D5 D5 G5 A5	14
	I	1 1 1 1 1 2 1 2 2 2 2 2 5 6	
16	M	C5 C5 C5 A5 G5 D5 D5 D5 E5 D5 C5	11
	I	1 1 1 6 5 2 2 2 3 2 1	
17	M	C7 D7 E7 E7 E7 E7 E7 E7 E7 D7 C7 D7 E7	13
	I	1 2 3 3 3 3 3 3 3 2 1 2 3	
18	M	Fs5 Gs5 B5 Cs6 Cs6 B5 B5 B5	8
	I	5 6 1+ 2+ 2+ 1+ 1+ 1+	
19	M	C5 C5 D5 D5 E5 E5 F5 F5 A4 A4	10
	M	1 1 2 2 3 3 4 4 6- 6-	
20	M	G5 D6 D6 E6 C6 G5 D6 D6 E6 C6 E6 E6 B5 D6	14
	I	1 5 5 6 4 1 5 5 6 4 6 6 3 5	

表二：八位受試者自訂的 20 筆查詢字串

查詢的結果列在表三，欄位中的數字代表出現正確檢索結果的位置，所以數字越小，表示越早可以發現正確的結果。例如第 1 筆查詢，用第一種檢索模式可在第 3 筆結果找到正確的樂曲，用第 2 種模式則要在第 20 筆結果才能找到正確的樂曲。由於一共有 $8 \times 20 = 160$ 項數字，為方便比較，我們依據表三另外做出表四的數據。在表四中，第 i 列第 j 行的數據，代表第 i 種檢索模式比起第 j 種模式在 20 次的查詢中成效較好的次數。所以如果此數據超過一半，即 10 次，表示第 i 種檢索模式比第 j 種檢索模式有較優異的表現。但是由於我們不計數兩種檢索模式表現一樣好的次數，所以事實上如果第 i 列第 j 行的數據大於第 j 列第 i

行的數據，那麼我們已經可以說第 i 種模式比第 j 種模式表現較優異了。表四中我們特別用粗體字來表示此種狀況。另外，表四的最後一欄顯示每一種檢索模式，在 20 次的查詢中表現最佳的次數，亦即表三中數字最低或與別的模式同樣有最低數字的次數。

Song mode	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	3	2	15	7	1	3	3	12	7	1	10	7	1	1	1	77	6	35	52	6
2	20	4	6	4	3	13	12	14	21	5	96	27	36	22	5	189	27	44	119	40
3	26	12	3	1	1	38	13	156	102	85	26	63	43	1	11	85	2	7	49	3
4	88	63	9	4	3	6	13	245	256	78	421	232	53	8	7	356	4	8	18	53
5	26	4	1	1	1	38	9	5	13	8	3	24	19	1	23	63	1	1	8	1
6	87	46	2	1	1	4	8	16	24	12	79	72	30	1	54	222	2	1	3	2
7	39	5	1	1	1	34	1	1	43	1	1	47	4	1	3	106	8	1	254	1
8	61	12	1	1	1	2	36	1	42	1	8	126	17	1	1	269	24	1	149	1

表三：分別用八種檢索模式進行 20 筆查詢的結果

mode	1	2	3	4	5	6	7	8	20 次查詢中表現最佳的次數
1		18	12	15	9	12	10	9	12
2	2		10	13	4	8	6	7	6
3	6	10		15	1	6	4	6	8
4	5	5	4		2	1	3	3	2
5	9	15	14	18		13	7	9	12
6	6	12	10	19	3		4	6	8
7	7	14	13	17	7	12		8	14
8	7	13	10	17	5	10	4		13

表四：第 i 種模式（第 i 列）比第 j 種模式（第 j 行）查詢成效較好的次數

從表四的結果中，我們可以看出（請讀者比較檢索模式 1 與 2、3 與 4、5 與 6、7 與 8），關鍵旋律的運用，不管是哪一種檢索模式，都比直接查詢音樂資料庫的效果好，其提升檢索成效的效果相當明顯。另外 n -note 索引法的 n 越大，則檢索成效越好（比較 3 與 5、4 與 6）。這是因為我們運用到更多的旋律子片段，其區分音樂旋律的效果就越好。此外，以絕對音高的標準音名來查詢未經編碼的旋律比用簡譜查詢編碼過的旋律其效果較好（比較表四中的 1 與 3、2 與 4），不過如果我們回去看表二的資料，即可發現這種情形只對音樂素養非常良好的使用者有用：在表二中，第一次就能用絕對音高作對查詢的只有三次（在第 1、3、12 筆查詢），其他十七次都需要告知樂曲的起始音高受試者才有機會找到正確的樂曲。在動態規劃比對法方面，比起最佳的 n -note 索引法（即檢索模式 5），此方法的檢索效果稍好。然而，每做一次動態規劃比對都要花費數百秒的時間，而利用前面提過的 n -note 索引法、反向索引檔、及雜湊函數技術，則每次查詢只需花費 1 至 2 秒。最後，由碎音或合併音引起的大量連續錯誤的不匹配問題，都無法由上述的檢索模式解決。從表二的第 8、9、10、11、19 筆查詢字串可看出，受試者都需要經過修正，才能克服此種查詢問題。因此，系統自動提示候選查詢

條件的功能，對此種查詢不匹配的問題相當重要。

六、結論

由於電腦網路的快速發展與普及，使用者除透過網路檢索書目性音樂資料外，對以音樂內容來查詢資料，以及直接取用全文檔案的需求越來越強。這幾年來，網路上越來越多的網站，已逐步提供音樂全文數位檔案，提供閱聽者利用。然而由於音樂內容檢索的研究與系統仍相當罕見，這些數位檔案仍難以供閱聽者善加運用，而無法達到較高程度的應用。本研究認為音樂數位檔案運用的第一步，是將音樂檔案做剖析，擷取重要內容特徵，然後予以自動編碼、索引，透過先進的使用者互動模式與資訊檢索技術，提供檢索利用。這些過程，均可透過機器設備與軟體的處理，達到高度的自動化作業。此模式不僅讓使用者可以作書目性資料的查詢，也可以作較為特定或深入的檢索，進而取用到完整的全文資料。讓資料提供單位，如視聽中心、圖書館等，可以大量節省人力與成本，縮短數位化與網路化的時程。對便利民眾利用資訊、提昇音樂教育、研究與創作環境，有相當大的幫助。

本研究對音樂文件所做的成果，也可以適用於其他類型的資料，如音訊或視訊資料庫（audio or video databases）。例如將訊號作適當的量化處理降低資料的變異性後，關鍵旋律擷取找尋重複性片段的演算法也可適用在一般的訊號上。這些音訊或視訊資料庫，或者儲存較不具音樂性的聲音檔案、或者儲存音樂的直接錄音檔案，雖不含語意層次較高的音樂樂譜的特徵，仍可擷取其重複片段，運用本研究所述的各種方式，而得以進一步處理與利用。因此，本研究對多媒體資料庫的處理與檢索，也將有所貢獻。

誌謝

本研究蒙國科會專題研究計畫補助，計畫編號：NSC 89-2213-E-030-002-，僅此誌謝。

參考文獻

1. Rodger J. McNab, Lloyd A. Smith, Ian H. Witten, Clare L. Henderson and Sally Jo Cunningham, "Towards the Digital Music Library: Tune Retrieval from Acoustic Input," In Proceedings of the ACM Digital Libraries 1996, pp. 11-18.
2. Silvia Pfeiffer, Stephan Fischer and Wolfgang Effelsberg, "Automatic Audio Content Analysis," In Proceedings of the Fourth ACM International Multimedia Conference, 1996, pp. 21-30.
3. Wold, E., Blum, T., Keislar, D., and Wheaton, J., "Content-based Classification,

- Search, and Retrieval of Audio,” IEEE Multimedia, Vol.3, No.3, 1996, pp. 27-36.
4. Jonathan T. Foote, “Content-Based Retrieval of Music and Audio,” Multimedia Storage and Archiving Systems II, Proceedings of SPIE, Vo. 3229, 1997, pp. 138-147. [Http://svr-www.emg.cam.ac.uk/jtf/papers/spie97-abs.html](http://svr-www.emg.cam.ac.uk/jtf/papers/spie97-abs.html)
 5. Hawley, M., “The Personal Orchestra,” Computing Systems, Vol. 3, No. 2, 1990, pp.289-329.
 6. Ghias, A., Logan, H., chamberlin, D., and Smith, B. C., “Query by Humming: Musical Information Retrieval in an Audio Database,” In Proceedings of Third ACM International Conference on Multimedia, 1995, pp. 231-236.
 7. Rodger J. McNab, Lloyd A. Smith, David Bainbridge and Ian H. Witten, “The New Zealand Digital Library MELody inDEX,” D-Lib Magazine, May 1997. [Http://www.dlib.org/dlib/may97/meldex/05witten.html](http://www.dlib.org/dlib/may97/meldex/05witten.html)
 8. Benjamin Chen and J.-S. Roger Jang, “Query by Singing,” In Proceedings of the 11st Conference on Computer Vision, Graphics, and Image Processing, 1998, pp. 529-536.
 9. Chen, J. C. C. and A. L. P. Chen, “Query by Rhythm: An Approach for song Retrieval in Music Databases,” Proceedings of IEEE International Workshop on Research Issues in Data Engineering, 1998, pp. 139-146.
 10. Chou, T. C., A. L. P. Chen, and C. C. Liu, “Music Databases: Indexing Techniques and Implementation,” International Workshop on Multimedia Data Base Management Systems, 1996.
 11. Liu, C. C., A. J. L. Hsu, and A. L. P. Chen, “1-D List: An Approximate String Matching Algorithm for Content-Based Music Data Retrieval,” submitted for publication.
 12. David Sankoff and J. B. Kruskal (ed.) *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Reading, MA: Addison-Wesley, 1983.
 13. Harding, W. B. Croft, and C. Weir, "Probabilistic Retrieval of OCR Degraded Text Using N-Grams," 1995, in *Research and Advanced Technology for Digital Libraries*, Carol Peters and Costantino Thanos, Editors, 1997. pp. 345-359.
 14. Yuen-Hsien Tseng and Yu-I Lin "Evaluation of Fuzzy Search, Term Suggestion, and Term Relevance Feedback in an OPAC System," *Bulletin of the Library Association of China*, No. 61, 1998, pp.103-126.
 15. Jones, G. T., *Music Theory*, Harper & Row, Publishers, New York, 1974.
 16. Jia-Lien Hsu, Chih-Chin Liu, and Arbee L. Chen, “Efficient Repeating Pattern Finding in Music Databases,” Proceedings of ACM Seventh International Conference on Information and Knowledge Management.
 17. Yuen-Hsien Tseng, "Multilingual Keyword Extraction for Term Suggestion,"

- Proceedings of 21st International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '98, Aug. 24-28, Australia, 1998, pp.377-378.
18. Yuen-Hsien Tseng, "Fast Keyword Extraction of Chinese Documents in a Web Environment," Information Retrieval Workshop for Asia Languages - 1997, Oct. 8-9, Japan, pp. 81-87.
 19. Tim Crawford, Costas S. Iliopoulos, and Rajeev Raman, "String-Matching Techniques for Musical Similarity and Melodic Recognition," Chapter 3, pp. 73-100, in *Melodic Similarity: Concepts, Procedures, and Applications*, Computing in Musicology 11, Edited by Walter B. Hewlett and Eleanor Selfridge-Field, The MIT Press, 1998.
 20. Yuen-Hsien Tseng, "Content-Based Retrieval for Music Collections," Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval-SIGIR '99, Aug. 15-19, Berkeley, U.S.A., 1999, pp.176-182.
 21. "AKoff Music Composer", <http://www.akoff.com/>