Vol. 35 No. 9　　情　報　処　理　　Sep. 1994

解　説

音楽情報処理

3. 伴　奏　シ　ス　テ　ム†

堀内靖雄††　橋本周司†††

1. はじめに

伴奏システムとは，コンピュータが人間の独奏者の演奏と協調して伴奏を行うシステムである．従来のコンピュータ音楽の多くはテープに(多重)録音して演奏する形式を取っており，人間の演奏家が一緒に演奏するときには，人間がコンピュータに合わせて演奏しなければならず，演奏家は自由な演奏が行えなかった．この問題を解決するため，コンピュータが人間の演奏に合わせて演奏を行うシステムとして「伴奏システム」が開発された[1),2)]．

2. 伴奏システムの概要

本稿で紹介する伴奏システムとは，独奏・伴奏パートの楽譜があらかじめシステムに与えられており，演奏時に独奏者の演奏を聴きながら，それに合わせて伴奏パートを演奏するシステムである．楽譜をあらかじめ与えないで追従するシステム(即興やリアルタイム作曲を含む)もあるが，紙面の都合上，本稿では割愛する．それらについては文献 3)，文献 4) などを参照していただきたい．

伴奏システムは楽譜情報を用いて独奏者の演奏を追跡し，その楽譜上での位置を認識，それに合わせて伴奏パートの演奏を行う．伴奏システムの処理は大きく 4 段階に分かれる．

1. Preprocessor: 独奏者の演奏を入力とし，ピッチ抽出などの前処理を行う．

2. Matcher: 独奏者の演奏と楽譜とを比較し，それらの間の対応関係を調べる．

3. Accompaniment: 独奏者の演奏に合わせて，伴奏パートの演奏をスケジューリングする．

4. Synthesis: 伴奏パートの演奏を行う．

4. の Synthesis は通常，シンセサイザで演奏を行うだけであるので本稿では省略する．以下 1～3 までの各処理について順に説明する．

3. Preprocessor

独奏者の演奏は MIDI 信号か音響信号として伴奏システムに入力される．入力が MIDI 信号の場合，演奏された音符のピッチは，その時刻とともに正確に認識できるのに対して，入力が音響信号の場合，伴奏システムは独奏者によって演奏された音符のピッチを認識しなければならない．この処理は Preprocessor で行われる．ピッチが認識されれば，それ以後の処理は MIDI 入力の場合と同様に行うことができる．また Preprocessor ではマッチングのための前処理も行うが，これについては **4.2** と **4.3** で説明することとし，本章ではピッチ抽出法についてのみ述べる．

井上らは人間の歌を入力とした伴奏システムを開発している[5),6)]．マイクからの入力信号は A/D 変換され，DSP により歌声のピッチ抽出を行う．ピッチ抽出は，低域通過フィルタにより高調波成分を減衰させた後，適当な閾値以上のピーク間の周期を測定して行う．そのとき 3 周期の平均周波数に最も近い音程を歌声の音程として決定している．また任意の調で曲の途中から歌っても独奏者の位置を検出し，伴奏を行うこともできる．このシステムは独奏者の歌の音程がずれたとき，エフェクタを用いて歌声の音程を補正するように拡張された[7)]．

歌唱の場合，通常，アタック時刻（音符が演奏された時刻）から 50～60 ms 経たないと周波数が安定しないため，その間はピッチ抽出が困難である．そこで竹内らは，アタックの予測時刻とパ

† Computer Accompaniment Systems by Yasuo HORIUCHI (Department of Computer Science, Tokyo Institute of Technology) and Shuji HASHIMOTO (Department of Applied Physics, School of Science and Engineering, Waseda University).
†† 東京工業大学理工学研究科情報工学専攻
††† 早稲田大学理工学部応用物理学科

ワーを用いてアタックの検出を行い，アタックが検出できたときにはその時点で伴奏の調節を行う方法を提案している[8),9)]．ピッチ抽出及びマッチングはその後で行い，アタックの認識などの間違いが見つかったときは修正する．アタックが検出できないときは他の方法と同様にピッチ抽出を行い，それに基づいた伴奏制御を行う．

4. Matcher

Matcher の仕事は Preprocessor から受け取る独奏者の演奏情報を独奏者の楽譜と比較しながら追跡し，独奏者が楽譜のどこを演奏しているかを認識することである．具体的には，音声認識の分野などで用いられている DP マッチング・アルゴリズムをリアルタイムに適用し，演奏と楽譜とのマッチングをとり，その対応関係を調べることにより実現する．このとき伴奏システムは独奏者の演奏を追跡しながら，伴奏パートの演奏を出力しなければならない（独奏者と一緒に演奏しなければならない）ことに注意して欲しい．そのため，伴奏システムは現時点までに入力された情報だけを用いて，独奏者の演奏位置をリアルタイムで認識する必要がある．

Matcher はその入力がモノフォニック（二つ以上の音が同時に演奏されない）かポリフォニック（鍵盤楽器の和音のように，二つ以上の音が同時に演奏される）によって処理が異なるので，それぞれについて述べる．

4.1 Monophonic Matcher

最も簡単で基本的な Matcher は Dannenberg のもの[1)]であると考えられるので，まずそのアルゴリズムを紹介する．

独奏者の楽譜情報と演奏情報はイベント（演奏された音符）のストリームで表現される（**図-1** 参照）．この図は楽譜情報が AGEGABC（これは順にラソミソラシドを表す）のときに，独奏者が AGEDGBC と演奏した状態を表している．このとき，これら二つのイベント・ストリームがどのような対応関係にあるのかを調べる．この対応関係を表すのにベスト・マッチを定義する．これは，二つのストリームの最も長い共有サブ・シーケンスであるとする．この例でのベスト・マッチは図中に示してあり，演奏データから D 音を抜き，楽譜データからA音を抜いたものである．こ

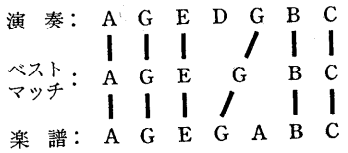


図-1 二つのイベント・ストリームとベスト・マッチ

	演奏: A	G	E	D	G	B	C
楽譜: A	1	1	1	1	1	1	1
G	1	2	2	2	2	2	2
E	1	2	3	3	3	3	3
G	1	2	3	3	4	4	4
A	1	2	3	3	4	4	4
B	1	2	3	3	4	5	5
C	1	2	3	3	4	5	6

図-2 ベスト・マッチを求める行列

こで演奏データから抜いた音は余分な(間違った)音を表し，楽譜から抜いた音は演奏者が演奏しなかった（省略した）音を表す．

ベスト・マッチを求めるのに**図-2** のような行列を考える．行列の行は楽譜イベント，列は演奏イベントを表し，行列の値はその点でのベスト・マッチの値を表している．演奏イベントが検出されるごとに新しい列が計算される．演奏イベント c が検出されたとき，行列の要素 (c, r) (r は楽譜イベント）の値は，もし c と r がマッチしたのなら $(r-1,\ c-1)$ のものよりも 1 大きくなり，そうでないなら $(r-1, c)$ と $(r, c-1)$ の大きいほうを取る．値が前列の最大値よりも大きくなったとき，演奏者の現在の演奏情報として，その楽譜位置を Accompaniment に報告する．図で下線が引いてあるものが楽譜位置を報告したマッチである．

しかし，このように演奏イベントが検出されるごとに 1 列すべて計算するのでは，計算量的に問題がある．演奏は比較的，正確に行われるということを仮定すれば，現在のマッチ付近の小さな窓を設定し，その内部だけを計算すれば良いということになる．窓の中心は，マッチが生じて報告されたら，そのマッチの次の行に動かし，マッチが見つからなかったら，窓を 1 行下に動かす．前と同じ例で大きさ 3 の窓を用いた行列を**図-3** に示す．

このアルゴリズムでは余分な演奏イベントが検

	演奏: A	G	E	D	G	B	C
楽譜: A	1	1					
G	1	2	2				
E	1	2	3	3			
G			3	3	4	4	
A				3	4	4	
B					4	5	5
C							6

図-3 窓を用いたベスト・マッチの計算結果

出され，それらが楽譜を先に飛ばしてマッチするときに問題が生じることがある．たとえば，楽譜がBGEFCBDGとなっていたとき，演奏イベントがFBGと入力されると，その時点では，このアルゴリズムはFを4番目，Bを6番目，Gを8番目の音符とマッチしてしまう．そこで，窓の中心が移動できる量を2行に制限し，楽譜イベントをスキップするマッチにはペナルティを与える．マッチしたイベントのペアには先ほど1ポイント与えたが，楽譜をスキップするときには1ポイント減らす．しかし，そのときベスト・マッチとして複数の候補が考えられることがある．そのような場合には，楽譜の中で一番最初のイベントでマッチが起こっているとして報告する．

初期バージョンは以上のようなマッチングを行っていたが，文献10)では，さらにミスに対して別々のペナルティを与えている．すなわち楽譜中に一致するイベントがないとき c_{extra}，楽譜中のイベントと音程が違うとき c_{wrong}，ある楽譜のイベントが演奏に一致しないとき $c_{missing}$ として，それぞれに適当なコストを与えてベスト・マッチを計算している．

また文献11)では，独奏者の演奏の追跡がある程度不確かなとき，複数のMatcherを作成して追跡し，独奏者が窓の範囲からはみ出さないようにしている．

このようにDannenbergのアルゴリズムはマッチングに独奏者の音程情報のみを用いているのに対して，Vercoeは音程情報に加えて，その立ち上がり時間の情報も用いている[12]．すなわち，マッチングの確からしさは，先のそれぞれのミスに対してのコストと時間的なエラーによるコストとの合計を用いて決定している．時間コストは過去のいくつかのマッチングした点を重み付けして最小

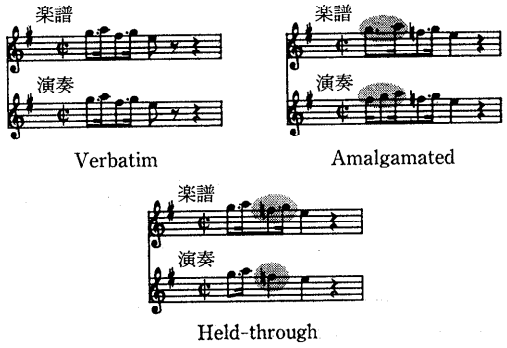


図-4 音楽的パターンを考慮したマッチング（文献13）より）

2乗近似を行い，その得られた直線からの逸脱に応じて決定している．

Bairdらは人間の演奏家はメロディの「モチーフ」を追跡しているという考えに基づき，音符や休符とその長さをユニットとし，それらをまとめたパターン（音楽的モチーフ）ごとのマッチングを再帰的に取る方法を提案した[13]．これは演奏家が犯しやすいミスの種類に基づいてマッチングを行う．それらは楽譜どおりの演奏による Verbatim，演奏家がある音符を間違えて演奏したが，すぐに正しい音符を演奏し直した（余分な音符を演奏してしまった）Amalgamated，楽譜のある音符を演奏せず，前の音符を続けていた Held-through である（**図-4** 参照）．これらに適当なコストを課してマッチングを行っている．

4.2 Polyphonic Matcher

Dannenbergらは文献10）でポリフォニック入力のマッチング方法について述べている．ここでは，その方法として静的マッチングと動的マッチングの二つの考えが示されているが，本稿では静的マッチングについてだけ述べる．動的マッチングについては文献10）を参照していただきたい．

アルゴリズムについて述べる前に，ポリフォニック入力の場合の問題点について述べる．モノフォニック入力の場合，楽譜も演奏も順序付けられたイベントのシーケンスであった．しかしポリフォニック入力の場合は，楽譜は部分的にしか順序付けられておらず，演奏順序は明示されない．すなわち和音ＣＥＧは短い時間内にＧＥＣ，ＧＣＥ，ＣＥＧ，ＣＧＥ，ＥＧＣ，ＥＣＧと演奏される可能性がある．一般には**図-5** の下に示されるように，記号の集合の列として楽譜を記述することができる．ここで同じ集合の記号は丸で囲まれ

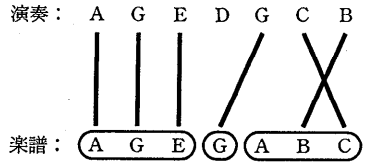


図-5　演奏とポリフォニック楽譜との関係

ている．実際に演奏されるイベントは図の上のようになり，その間の対応が線で示されている．ここで注意しなければならないのは，同じ集合の中にある楽譜記号に線が引かれるときには，その線はクロスしても良いということである．そのため上述のモノフォニック・マッチャーをそのまま適用したのではマッチングは行えない．

この問題は和音など同時に演奏されるべき音符の集合を一つの複合イベントとし，複合イベントどうしをマッチングすることにより，モノフォニック・マッチャーの問題に還元できる．複合イベントとは一つ以上の音符を含むイベント集合である．楽譜を複合イベントに変換するのは簡単である．すなわち楽譜上の縦に並んだ音符を一つの複合イベントとすれば良い．演奏情報を複合イベントにするためには ε テストと呼ばれる処理を Preprocessor で行う．これは ε (Dannenberg らは 100 ms としている) 以内にある演奏イベントを一つの複合イベントにまとめる．

しかし，同時に演奏されるべき和音がアルペジオで演奏されたときのように，和音全体を演奏する時間が ε よりも大きくなることがある．そのとき，ある音符がアルペジオに含まれるのなら，その音符は次の和音の最初の音符よりも前の音符のほうに近くなるということを利用すれば良い．そこで上述の ε テストよりも前に若干ゆるいテストを行う．このテストは，ある音符が次の和音（複合演奏イベント）の予測時刻よりも前の和音（複合イベント）に近いのなら，たとえそれらが ε の間になくても，前の和音と同じ複合イベントにグルーピングするというものである．正確には，あるイベントが入力されたとき，直前の複合イベントからそのイベントまでの時間が次の複合イベントまでの(予測)時間の ε 分数（彼らは 1/4 を用いている）倍よりも小さいのなら，そのイベントは前の複合イベントにグルーピングされる．独奏者の演奏イベントが入力されるごとに，ε 分数テスト・ε テストを行い，それらの音符を複合イベントにグルーピングする．これは決定的に行われ，選択の余地はないので，この方法は静的グルーピングと呼ばれている．

このアルゴリズムでは，独奏者の演奏イベントを処理するごとにベスト・マッチ関数を計算する．すなわち和音の最初の音符が楽譜にヒットするやいなや位置と時刻の情報を得ることができる．その後，同じ複合イベントに属する音符が入力されても，その位置の信頼度を増加するのに用いられるだけである．複合イベント（楽譜複合イベントと現時点での演奏複合イベント）どうしのマッチングは以下の関数を用い，その値が閾値 0.5 以上である場合にそれらの複合イベントがマッチしたと考える．

$$\frac{P_{in\_score}-P_{not\_in\_score}}{P}$$

ここで P_{in_score} は演奏複合イベント中で楽譜複合イベントにあるイベント（正しい音符）数，$P_{not\_in\_score}$ は演奏複合イベント中で楽譜複合イベントにないイベント（余分に演奏された音符）数，P は演奏された全イベント数を表している．すなわち $P_{in\_score}+P_{not\_in\_score}=P$ となる．この関数の意味は，演奏されたイベントが楽譜中に書かれているとき増加し，楽譜中に書かれていないとき減少するということであり，各演奏イベントは，その時点までに演奏されたイベント数に比例して影響を与えることができる．

4.3　トリルとグリッサンド

ポリフォニック・マッチャーの応用として，Dannenberg らは文献11)でトリルとグリッサンドを扱う方法について述べている．トリルとは，ある音符とその二度高い音とを交互に素早く演奏する奏法であるが，このとき独奏者がこれらの音を何回繰り返すのかは楽譜からは分からない．また，グリッサンドは，途中の音も発生しながら，2 音間を素早く移る奏法であるが，途中の音の経過時間は楽譜には指定されておらず，全部の音が発生するとは限らない．そのため，トリルやグリッサンドをイベントのシーケンスで表現することはできない．そこでトリルやグリッサンドも一つの複合イベントに変換するように Preprocessor の処理を拡張する．Preprocessor は次の入力がトリルやグリッサンドであると予測されるときに

は，トリル／グリッサンド・モードへ入る．このとき，トリルやグリッサンドの最初の音符が入力されたら，それを Matcher へ送るが，その後はトリルやグリッサンドの終了条件が満たされるまで入力を無視する．終了を認識したら，通常のモードへと戻る．

4.4 リハーサル

Vercoe は文献 12) でリハーサルの利用を提案している．すなわち独奏者とのリハーサルを何度か行い，そこから独奏者固有の統計情報を抽出する．その情報はリズムの逸脱とテンポの揺らぎであり，リハーサル中の独奏者のイベント・リストのテンポ解析により求める．この解析はリアルタイムで行う必要がないため，未来のイベントも解析に利用でき，より正確な値を得ることができる．マッチングを行うときには，リハーサルから得られた演奏情報（演奏者固有の情報が付加されたもの）と実際の演奏とのマッチングをとるが，そのときサンプルの標準偏差の逆数で重み付けすることにより，その独奏者の癖に合わせた伴奏を行うことができるようになる．

5. Accompaniment

Accompaniment が行う仕事は，Matcher から報告される現在の独奏者の楽譜上の位置情報を用いて伴奏を独奏者に合わせるように伴奏の演奏データをスケジューリングすることである．システムはあらかじめ演奏すべき楽譜を与えられている．問題は，それぞれのイベントをいつ演奏するかを決定することである．

そこで，システム内部に調整可能な時計を実装する．これをコンピュータ内部の実時計と区別するため仮想時計と呼ぶ．仮想時計は楽譜上の時間を意味し，単位は拍である．仮想時間は実時間の関数として以下のように定義される．

$$(R-R_{ref})\times S+V_{ref}$$

ここで R は実時間，R_{ref} は仮想時計が最後にセットされた（実）時間，S は仮想時計の速度，V_{ref} は仮想時計が最後にセットされた仮想時間である．システムは現在の実時間を R_{ref} に割り当て，そのときの仮想時間を V_{ref} に割り当てることによって仮想時計をいつでもセット可能である．また仮想時間の「速度」を定めるために S を任意に更新することができる．これは仮想時間が実時間に対して進む比率であり，演奏テンポを意味する．

Dannenberg の最初のバージョン[1]では，マッチが報告されるごとに仮想時計をリセットして演奏位置のみを変更していたが，その方法では同じ音符が 2 度演奏されたり，ある音符が省略されたりしてしまう．もしマッチが連続しているのなら，伴奏は楽譜位置をジャンプするよりむしろ，演奏テンポを変化させて独奏者に合わせたほうが良い．しかし，たとえば独奏者が長い休符を数え間違えたりして大きな時間変化を行ったとすると，この方法では不自然になってしまう．そのように時間変化がある閾値以上のときは演奏位置のジャンプを行う[10]．マッチが連続していないときは，新しい位置へとジャンプする．また仮想時計の速度 S は，過去のいくつかのマッチ・ポイントから平均して求める．

従来の伴奏システムはコンピュータが人間に追従するものであったが，実際には，人間もコンピュータの演奏に影響を受けており，両者による演奏は互いに影響を及ぼしあっている．井川らは独奏者と伴奏システムとの間に相互作用があるモデルを提案した[14),15]．相互作用モデルにおいては，人間もコンピュータもテンポを決定するための情報は人間のテンポ (ΔT_k) と機械のテンポ (Δt_k) のみである．ただし，人間のテンポ決定時には「曲調（テンポ）を変えたい」という「意志」が働き，これを f_k と表す．このことから人間系のモデルは，

$$\Delta T_{n+1}=\sum_{k=1}^{u_1} a_k\Delta T_{n+1-k}+\sum_{k=1}^{u_2} b_k\Delta t_{n+1-k}+\sum_{k=1}^{u_3} f_{n+1-k}$$

機械系のモデルは，

$$\Delta t_{n+1}=\sum_{k=1}^{v_1} c_k\Delta T_{n+1-k}+\sum_{k=1}^{v_2} d_k\Delta t_{n+1-k}$$

と表す．伴奏システムは，この人間系のモデルに基づいて人間の演奏を予測し，その予測を用いて機械系のモデルに従って伴奏のテンポを制御する．

堀内らは自然なアンサンブル実現のため，システムの自主性を考慮した伴奏システムを提案した[16]．具体的には伴奏システムが独奏者から独立する度合いを表す尺度として独立度を定義し，

独立度に従って伴奏システムの演奏を制御している．独立度は音楽的状況に従って変化し，伴奏システムが合わせるべき状況では独奏者に良く追従し，逆に伴奏システムが独奏者をリードすべき状況では自主的な演奏を行うことができる．このシステムは人間の被検者による聴取実験により，従来のシステムよりも良い評価を受け，また人間の伴奏者にも劣らないという結果が示された[17]．

6. おわりに

楽器入力，特に MIDI 入力のシステムにおいて，伴奏システムは人間の独奏者にかなり良く(大きくずれたりしないという意味で)追従できるようになった．しかし歌唱の追従に関してはまだ改良の余地があるようである．

また，リハーサルも Vercoe によって用いられただけであるが[12]，リハーサルからのさまざまな情報の抽出は重要になるであろう．

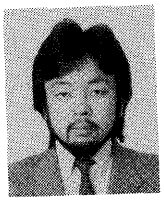
独奏者の身振りや息継ぎなどの情報を用いた伴奏システムはまだ作成されていない．しかしジェスチャを解析して演奏に利用するシステムはすでに存在しているので(文献 18)，9) など)，それらを伴奏システムに適用できれば，さらにその能力を上げることができると考えられる．

人間のように自然な伴奏を行えるようにするため，Dannenberg らは人間の伴奏者のふるまいを心理実験により調査している[19]．このようにして人間の伴奏プロセスが解明されれば，伴奏システムはさらに強力な力をもち，Vercoe の言うように演奏グループのだれかをコンピュータで置き換えても残りのライブ演奏家がその違いに気が付かないくらいに自然な演奏が可能となるであろう．

参考文献

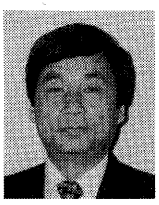
1) Dannenberg, R.B.: An On-Line Algorithm for Real-Time Accompaniment, In *Proc. of ICMC*, pp. 193-198 (1984).
2) Vercoe, B.: THE SYNTHETIC PERFORMER IN THE CONTEXT OF LIVE PERFORMANCE, In *Proc. of ICMC*, pp. 199-200 (1984).
3) Dannenberg, R.B. and Mont-Reynaud, B.: Following an Improvisation in Real-Time, In *Proc. of ICMC*, pp. 241-248 (1987).
4) Driesse, A.: Real-Time Tempo Tracking Using Rules to Analyze Rhythmic Qualities, In *Proc. of ICMC*, pp. 578-581 (1991).
5) 井上 渉，磯貝昌幸，橋本周司，大照 完：歌声のピッチ検出による自動伴奏システム，情報処理学会第 44 回全国大会講演論文集，pp. 1, 387-1, 388 (1992).
6) 井上 渉，橋本周司，大照 完：歌声自動伴奏システム，音楽情報科学研究会夏のシンポジウム '92 Paper Session Proceedings, pp. 79-82 (1992).
7) Inoue, W., Hashimoto, S. and Ohteru, S.: A COMPUTER MUSIC SYSTEM FOR HUMAN SINGING, In *Proc. of ICMC*, pp. 150-153 (1993).
8) 竹内庸博，片寄晴弘，井口征士：Virtual Performer：適応型カラオケシステム，情報処理学会第 46 回全国大会講演論文集，pp. 5, 147-5, 148 (1993).
9) Katayose, H., Kanamori, T., Kamei, K., Nagashima, Y., Sato, K., Inokuchi, S. and Simura, S.: Virtual Performer, In *Proc. of ICMC*, pp. 138-145 (1993).
10) Bloch, J. J. and Dannenberg, R. B.: Real-Time Computer Accompaniment of Keyboard Performances, In *Proc. of ICMC*, pp. 279-289 (1985).
11) Dannenberg, R.B. and Mukaino, H.: New Techniques for Enhanced Quality of Computer Accompaniment, In *Proc. of ICMC*, pp. 243-249 (1988).
12) Vercoe, B. and Puckette, M.: Synthetic Rehearsal: Training the Synthetic Performer, In *Proc. of ICMC*, pp. 275-278 (1985).
13) Baird, B., Blevins, D. and Zahler, N.: Artificial Intelligence and Music: Implementing an Interactive Computer Performer, *Computer Music Journal*, Vol. 17, No. 2, pp. 73-79, 1993 (Summer 1993).
14) 井川孝之，直井邦彰，大照 完，橋本周司：相互作用モデルによる実時間適応自動伴奏とその動作解析，電子情報通信学会 1990 年全国大会講演論文集，pp. 7-216 (1990).
15) 澤田秀之，磯貝昌幸，橋本周司，大照 完：音楽演奏における人間と機械の協調動作について，情報処理学会第 44 回全国大会講演論文集，pp. 1, 389-1, 390 (1992).
16) 堀内靖雄，スパワリークリッサダー，田中穂積：伴奏者の自主性を考慮した自動伴奏システム，情報処理学会第 44 回全国大会講演論文集，pp. 1, 405-1, 406 (1992).
17) Horiuchi, Y. and Tanaka, H.: A Computer Accompaniment System With Independence, In *Proc. of ICMC*, pp. 418-420 (1993).
18) Morita, H., Hashimoto, S. and Ohteru, S.: A Computer Music System that Follows a Human Conductor, *Computer*, Vol. 24, No. 7, pp. 44-53, 1991 (July 1991).
19) Dannenberg, R.B.: Music Understanding By Computer, In *Proceedings of IAKTA/LIST International Workshop on Knowledge Technology in the Arts*, pp. 41-55 (1993).

(平成 6 年 2 月 3 日受付)



堀内 靖雄（正会員）

1967 年生まれ．1990 年東京工業大学工学部情報工学科卒業．1992 年同大学院理工学研究科修士課程修了．現在，同大学院理工学研究科博士課程在学中．音楽情報処理の研究に従事．人工知能学会会員．



橋本 周司（正会員）

1948 年生．1970 年早稲田大学理工学部応用物理学科卒業．1973 年同修士課程修了．1977 年工学博士．東邦大学理学部講師，助教授，早稲田大学助教授を経て，現在，早稲田大学理工学部教授（応用物理学科）．音楽情報処理，神経回路網，画像処理の研究に従事．電子情報通信学会，計測自動制御学会，日本ロボット学会，ICMA（国際コンピュータ音楽協会）各会員．