音楽処理の2つの試み

白川　貴浩*　　冨塚　英省**　　五十嵐　滋***
Takahiro Shirakawa、Hidemi Tomitsuka、Shigeru Igarashi
*筑波大学工学研究科　　**筑波大学理工学研究科　　***筑波大学電子・情報工学系

1　はじめに

本研究室では、ピアノおよびシンセサイザの自動演奏を中心とする、総合音楽情報システムPSYCHEを開発している。このシステムでは、楽器演奏の技術を計算機や機械にまかせ、個人の思うままの演奏が実現できることや、音楽の科学的な研究を、計算機の助けを得て行える環境を整えることを目標としている。

ここでは、PSYCHEにおいて現在開発中のソフトウェアである、ミュージック・コンパイラと対話型音楽情報処理システムについて述べる。

2　PSYCHEの概要

PSYCHEの名称は、Program SYstem-Conducted Harmony and Expression（表情・調和を指揮するプログラムシステム）に由来する。

PSYCHEのハードウェア構成は現在、図1のようになっている。楽器としては、ドライブユニット付グランドピアノPL-10が1台、ディジタルシンセサイザDX7が3台、DX9が1台あり、これらの楽器を制御するためのパーソナル・コンピュータとしてPU-1、TESMIC8000、PC-9801VM2がある。また、ソフトウェアの

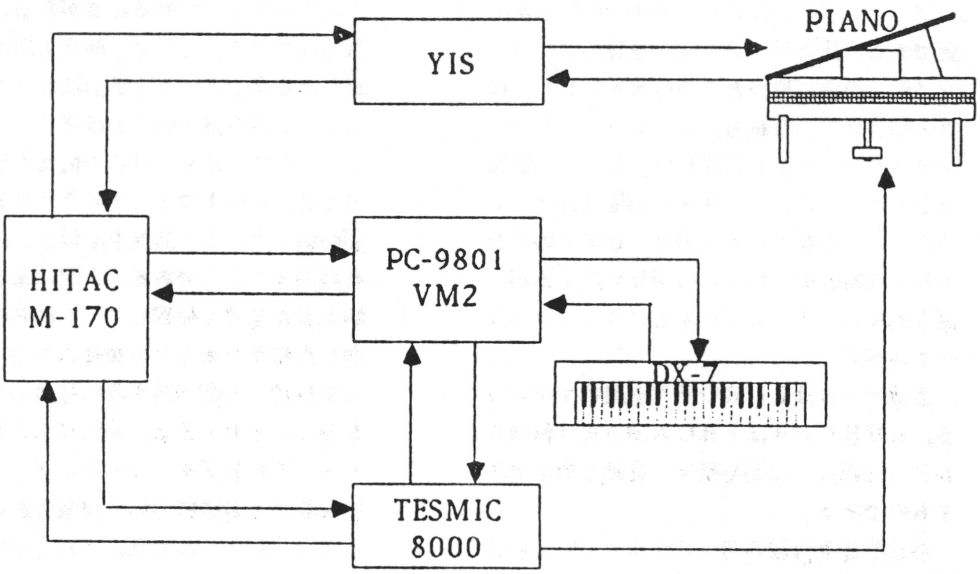


図1　PSYCHEの構成

開発に、ＰＣ－９８０１と汎用大型計算機ＨＩＴＡＣ　Ｍ－１７０が使われている。以下で述べるミュージック・コンパイラはＭ－１７０上で、対話型音楽情報処理システムはＰＣ－９８０１上で開発されている。

ＰＳＹＣＨＥの基本的な機能としては、

① パーソナル・コンピュータで楽器を制御することにより、自動演奏を行う
② 人間が楽器を演奏した演奏データを記録する
③ 汎用音楽記述言語ＥＵＲＯＰＡ［1］で楽譜データを記述し、計算機に入力する
④ ミュージック・コンパイラにより、楽譜データから演奏データを作成する
⑤ 演奏データを編集する

の５つがある。

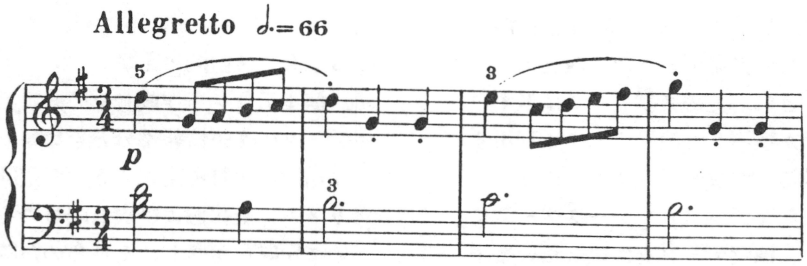
演奏データは楽器制御用のデータで、ピアノを演奏させることを基本にしている。ピアノを制御して１つの音を鳴らすためには、打鍵する鍵盤の位置（音の高さ）、打鍵の強さ（音量）、打鍵時間（発音のタイミング）、打鍵の長さ（音の長さ）、の４つの情報が必要である。演奏データでは、鍵盤のアクション（動く鍵盤の位置とそのＯＮ・ＯＦＦ、ＯＮの場合はその音量も）とアクション間のインターバル（４ｍｓ単位）で、これらの情報を表している。シンセサイザの場合には、アフタータッチなどがあるので、ピアノの場合よりも楽器制御のための情報量が増えるが、基本的には、インターバルとアクションですべて表せる。

楽譜データはＥＵＲＯＰＡを用いて記述する。ＥＵＲＯＰＡはＩＲＣＡＭ方式やＭＭＬなどと同様に、音高を英字、音価を数字で表すものである。

対話型音楽情報処理システムで用いているデータ・フォーマットとして、Ｕｎｉデータがある。これは、演奏用の情報や楽譜情報など様々な情報を記述できるようにした、中間データである。テキスト・エディタで扱えるように、１行が演奏における１音に対応するようになっている。１行の内はフィールド名と値との組が順不同で並んでおり、ファイル全体では可変構造体の配列と考えることができる。

3　ミュージック・コンパイラにおける試み
―人間が演奏した演奏データの利用―

ミュージック・コンパイラはＥＵＲＯＰＡで記述された楽譜データから演奏データを作成するソフトウェアである。楽譜データを入力とした場合の計算機による自動演奏は、機械的で平坦であるという指摘が、以前からなされている。その原因として、楽譜に書かれている情報だけで演奏を作り出そうとしていることが挙げられる。人間の演奏者が演奏を行う場合には、曲想などの楽譜には明示されていない情報を付加して演奏している。したがって、計算機で楽譜データから演奏データを作成する場合にも、なんらかの方法でこれらの情報を補ってやる必要がある。これらの情報を補う方法として、演奏方法を細かく指定した楽譜を入力とする方法などがあるが、ここでは計算機への入力は楽譜情報だけに限定し、他の方法でこれらの情報を補うことを考える。その１つとして考えられるのが、人間の演奏者による演奏を記録した演奏データを利用することである。人間による演奏は、演奏者が楽譜を解釈して、これらの情報を付加した結果であるとみなせるので、これを利用すれば、人間が演奏時に付加している情報を補うことができる。人間による演奏を解析して、その結果をミュージック・コンパイラに反映する方法については報告がなされているが［2］、ここでは、それとは異なる、人間が演奏した演奏データを利用する方法について述べる。




```
!r 3/4
!t allegretto
!c f#
!P1.1v1 piano
!P1.1v1 d+4 g8 a b c+ d+.4 g. g.
!P1.1v1 leg.  !P2.1v1 eoleg.
!P1.1v2- [g/b/d+]2 a4 b2.
!P3.1v1+ e4 c8 d e f g.4 g-. g-.
!P3.1v1 leg.  !P4.1v1 eoleg.
!P3.1v2 c2. b-
```

Hexadecimal Data	Total Time	Timer	Action
00			
05 97 00 90 4A 34 05	0:00.604	00.604	D 4 52
05 01 00 90 37 20 05	0:00.608	00.004	G 2 32
05 01 00 90 3B 34 05	0:00.612	00.004	B 2 52
05 01 00 90 3E 28 05	0:00.616	00.004	D 3 40
05 98 00 90 43 34 05	0:01.224	00.608	G 3 52
05 0B 00 90 4A 00 05	0:01.268	00.044	D 4 off
05 32 00 90 45 38 05	0:01.468	00.200	A 3 56
05 11 00 90 43 00 05	0:01.536	00.068	G 3 off
05 37 00 90 47 3C 05	0:01.756	00.220	B 3 60
05 01 00 90 39 38 05	0:01.760	00.004	A 2 56
05 04 00 90 3E 00 05	0:01.776	00.016	D 3 off
05 02 00 90 45 00 05	0:01.784	00.008	A 3 off
05 01 00 90 3B 00 05	0:01.788	00.004	B 2 off
05 02 00 90 37 00 05	0:01.796	00.008	G 2 off
05 37 00 90 48 38 05	0:02.016	00.220	C 4 56
05 0B 00 90 47 00 05	0:02.060	00.044	B 3 off
05 41 00 90 4A 40 05	0:02.320	00.260	D 4 64
05 04 00 90 3B 3C 05	0:02.336	00.016	B 2 60
05 02 00 90 48 00 05	0:02.344	00.008	C 4 off
05 08 00 90 39 00 05	0:02.376	00.032	A 2 off
05 80 00 90 43 38 05	0:02.888	00.512	G 3 56
05 06 00 90 4A 00 05	0:02.912	00.024	D 4 off
05 69 00 90 43 00 05	0:03.332	00.420	G 3 off
05 20 00 90 43 3C 05	0:03.460	00.128	G 3 60

図2　EUROPAと演奏データの例

３．１　基本的な考え方

楽譜上で似ているものは、同じように演奏される。そこで、楽譜データから演奏データを作成するときに、その楽譜データに似ている楽譜を人間が演奏した演奏データを利用することができる。つまり、楽譜とその曲を人間が演奏した演奏データとをデータベースに何曲か用意し、演奏データを作成したい楽譜データと最もよく似た楽譜をデータベースから探す。そして、それに対応する演奏データを修正することによって、目的の演奏データを作成しようというものである。楽譜が似ているかどうかの比較は、１小節ごとに行うことにした。楽譜データから演奏データを作成する手順は次のようになっている。

① 人間の演奏を記録した演奏データに楽譜情報を付加したもの（以下、参照データと略す）を用意する。

② 楽譜の似ている、似ていないを判定するために、楽譜間に距離（実際は対称ではないので擬距離であるが、以下、距離と略す）を定義し、２つの小節間の距離を計算する距離評価関数を作る。

③ 演奏データを作成したい曲の楽譜データと参照データとを、各声部（右手と左手）１小節ごとに距離評価関数により比較し、最も似ているものを選ぶ。

④ 各小節について得られた参照データを演奏したい楽譜データに整合するように修正し、それをつなげることによって目的の演奏データを得る。

楽譜の比較を１小節単位にした理由としては、

① 楽譜に明示された情報である。

② 強拍弱拍の繰り返しの１周期である

③ 固定長である

などがある。また、右手と左手は、通常別な譜表に書かれるし、旋律を考える場合には分離したほうが扱いやすいので、それぞれ分けて扱うことにした。

３．２　楽譜間の距離

演奏データを作成したい楽譜データと参照データの楽譜情報部分とを比較するために、楽譜間の距離を定義する。楽譜間の距離は、リズム、旋律および表情記号のそれぞれについて距離を定義し、それらの距離の関数として定義した。現在のところ、これらの重み付きの和である。

リズムは、まず、２つの楽譜で小節内の拍数で数えて同じ位置にある音符の関係を、

① 楽譜データと参照データの両方に音符があり、音符の長さも等しい。

② 楽譜データと参照データの両方に音符があるが、その長さは異なる。

③ 楽譜データのほうには音符があるが、参照データには音符がない。

④ 参照データのほうには音符があるが、楽譜データには音符がない。

の４つに分類する。そして、それぞれに重み（Ｗ１からＷ４）を付けて、それを加えたものをリズムの距離とした。これらの重みは、現在、Ｗ１＝０、Ｗ２＝１０、Ｗ３＝１、Ｗ４＝１００になってる。楽譜データから演奏データを作成するときに使う参照データを選ぶのが目的なので、この値は、参照データの演奏データ部分で、それに使えない情報や欠落している情報がどれくらいあるかを考慮して決めた。つまり、①は、まったく同じである。②は打鍵の長さが異なる。③は情報は何も欠落していないが、目的の演奏データを作成するときには不要のデータである。④は打鍵のタイミング、打鍵の強さ、打鍵の長さの３つの情報が欠落している。距離の定義が対称でなく、擬距離になっているのはこの部分である。ここでＷ３＝Ｗ４であれば対称になる。

旋律は、音符と音符の間で、音程がどれだ

け動いたかを調べ、その動き方の違いを比較する。音程の動きを調べる点として、同じ位置に音符がある点（リズムの分類で①、②に該当するもの）を選ぶ。その隣り合った２点の音符が、半音階でどれくらい音程が異なるかを楽譜データと参照データでそれぞれ計算し、その差の絶対値を加えたものを旋律の距離とする。今のところ、音程の動きを比較しているだけで、和音については考慮していない。和音の場合はその最高音を代表として計算する。

表情記号は、まず、表情記号を複数の音符にわたって影響するタイプ１（クレッシェンド、リタルダンド、スラーなど）と、１つの音符にしか影響しないタイプ２（スタッカート、アクセント、トリルなど）の２つに分ける。タイプ１のものは表情記号の異なっている区間の音符としての長さを、タイプ２のものは表情記号の異なっている音符の数を、それぞれ距離を決める要素とする。つまり、タイプ１のものは、楽譜データと参照データのどちらか一方にだけ表情記号が付いていて、もう一方には付いていない区間の長さに重みをかけ、それらを加えたものをタイプ１についての距離とする。この場合、その区間に含まれる音符の数は、距離に関係しない。タイプ２のものは、楽譜データでも参照データでも、表情記号の付いている音符に対して、もう一方のデータのその音符と同じ位置に音符がないもの（リズムの分類で③か④）と、同じ位置に音符があっても、その音符に同じ表情記号がついていないものの数を数え、その数に重みをかけたものをタイプ２についての距離とする。この場合、表情記号の付いている音符の長さと距離とは無関係である。そして、タイプ１とタイプ２の値の和を表情記号の距離とする。

３．３　演奏データの修正

距離評価関数を用いることによって、楽譜データの各小節について、距離が最も短い参照データがデータベースから選ばれた。しかし、そのままでは目的の演奏データにはならないので、それを楽譜データに整合するように修正する必要がある。

修正を行うのは、

① 音高の不一致
② リズムの不一致
③ 表情記号の不一致
④ テンポの整合をとる

の４つについてである。

音高の修正は、音の高さ、つまり打鍵する鍵盤の位置を楽譜データの音の高さに変更する。

リズムについては、音符の長さが一致していないもの（リズムの分類で②）と、音符が欠落しているもの（リズムの分類で④）が修正の対象となる。音符の長さが一致していないものは、演奏データの項目のうち打鍵の長さを修正する必要があり、音符が欠落しているものは、打鍵タイミング、打鍵の強さ、打鍵の長さの３つの値を得る必要がある。修正の方法は、まず、修正が必要な音符と同じ長さの音符をデータベース中の参照データから探す。その音が１つしかなければ、それが修正用のデータとなるが、その音符が複数ある場合には、その音符の前後に部分的距離評価関数を適用し、部分的に最も似ているものを選ぶ。そして、修正が必要な音符、または欠落している音符に対応する演奏データの該当する項目を、選ばれた音のその項目の値で置き変える。

表情記号の修正は、足りない表情記号を補う場合と、余分な表情記号を取り去る場合との２通りがある。足りない表情記号を補う場合は、リズムの修正の場合と同様に、データベース中から同じ表情記号を探す。同じ表情

記号が付いているものの中から、タイプ1ならば、表情記号がかかっている区間の長さが同じものを、タイプ2なら表情記号が付いている音符の長さが同じものを探す。そして、それが複数あれば、それに部分的距離評価関数を適用し、修正用のデータを選ぶ。余分な表情記号を取り去る場合には、タイプ1ならその表情記号が付いていない区間で長さの同じものを、タイプ2ならその表情記号が付いていない音で音符の長さの同じものを、それぞれデータベース中から探す。その後は、足りないものを補う場合と同じである。ここで修正の対象となるのは、演奏データの項目のうち、表情記号が主に影響を与える項目についてである。例えば、クレッシェンドなら打鍵の強さであり、スタッカートなら打鍵の長さである。表情記号が他の項目にも影響を与えることは十分考えられるが、ここでは考慮しないことにした。

リズムと表情記号の修正で、修正用のデータがみつからない場合には、前後の音符の演奏データなどを参考にして、計算によって求める。

テンポの整合は、声部間、または前後の小節でテンポが不自然にならないようにするために行うものである。もともと異なった曲の小節をつなげるので、そのままではテンポが異なっており、なんらかの方法で調整をしてやる必要がある。前後の小節のテンポの整合は、参照データのもとの曲の全体のテンポの平均値と、楽譜データで指定されているテンポを基準とする。つまり、参照データのその小節のテンポが曲全体のテンポの平均値からずれている分だけ、楽譜データの指定のテンポから変動するようにした。また、声部間の整合では右手を基準とすることにし、右手に左手を合わせるようにした。

3．4　展望

以上で述べた方法で、人間の演奏データを利用して楽譜データから演奏データを作成する場合の利点としては、細かいアーティキュレーションなどは演奏者の特徴が忠実に、しかも簡単に実現されること、データベースを入れかえれば異なった演奏が得られること、などがある。

問題点としては、まず、1小節を単位として比較し、これをつなげて曲としているために、曲想のような比較的大きな特徴を扱いにくいことがある。これを解決するためには、比較を行う長さの単位をもっと長くするか、小節ごとの処理とは別に、曲のもっと大きな構造を扱う処理を行う必要がある。この比較の長さの単位、および方法については、さらに検討の必要があると思われる。

また、各小節について似ている参照データを探すときは、基本的には、参照データのすべての小節について距離の計算を行うので、参照データの量が増えれば、計算時間もそれに比例して増えるの。そこで、その場合には高速化を考える必要がある。解決方法としては、ソフトウェア的には、例えば、リズムパターンごとに参照データの各小節を分類し、まず、リズムについて距離を計算し、ある程度ふるいにかけてから小節どうしの距離の計算を行う、などがある。また、ハードウェア的には、参照データと楽譜データの距離を計算する部分は並列化することができるので、プロセッサを複数にする方法が考えられる。

現在のところ、演奏データに楽譜情報を付加して参照データを作成するのが、わりと手間のかかる作業である。演奏データから楽譜情報を得る自動採譜プログラムを開発中であるので、これや楽譜情報の編集を行うエディタを使えば、より簡単に参照データを増やすことができるようになり、より良い演奏が得られるようになるであろう。

4　対話型音楽情報処理システム
における試み
―視覚化に向かって―

人間の観賞に耐え得る計算機による自動演奏の実現のためには、音楽の構造をふまえて強弱・速度を制御し、適度な表情付けが行われなければならない。しかし、表情付けの定式化は極めて困難であり、計算機科学的なアプローチにおいても人間の演奏の解析や表情付けの試行錯誤的な実験が必要となる。　本研究室では、『楽譜データ入力→演奏データ作成→楽器制御』なる型の自動演奏システムの研究を進め一応の成果を得た。しかし、そのままでは芸術的な演奏を行うために楽譜データの中で奏法を細かく指定しなければならず非常に労力を要するという問題があった。さらに、楽譜データ入力に手間がかかること、楽譜データから演奏データの変換時間が長いこと、演奏データの編集ができないこと等の短所があるので、それらを解消し、試行錯誤的な実験をより容易にするために、次の３項目を目的とした対話型音楽情報処理システムをパーソナルコンピュータＰＣ－９８０１ＶＭ２上で製作した。

ⓐ演奏データを扱えること
ⓑ変更した演奏データをただちに演奏できること
ⓒ確立された音楽情報処理の知識を蓄積できること

４．１　演奏データ

本システム扱う演奏データは、Ｕｎｉデータと呼ばれる形式のデータ上に展開されている。Ｕｎｉデータ形式の演奏データ例を右図３に示す。各行は可変構造体であり、ファイル全体はその可変構造体の配列と考えることができる。^や%はフィールド名とフィールド値の区切り記号である。右図の例では、Ｉ，Ｋ，Ｖ，Ｌがフィールド名であり、十進で表された各数がそれらに対応するフィールド値である。演奏データは一音一行でＵｎｉデータ形式に展開されている。演奏に必要十分な情報は各音の起点時刻・高さ・強さ・長さの４つであり、演奏データは各行にそれらに対応する４つのフィールド（例においてはＩ，Ｋ，Ｖ，Ｌ）を持つ。また、小節線の位置や音名、声部　などの情報も必要に応じて付け加えていくことができる。このデータ形式は各行が可変構造体なので、必要が生じた時にいつでもどんな情報でも付け加えることができるというところに特徴がある。

```
^I%151^K%74^V%52^L%166^W%1
^I%1^K%55^V%32^L%295^W%2
^I%1^K%59^V%52^L%292^W%2
^I%1^K%62^V%40^L%289^W%2
^I%152^K%67^V%52^L%78^W%1
^I%61^K%69^V%56^L%78^W%1
^I%72^K%57^V%56^L%153^W%2
^I%0^K%71^V%60^L%74^W%1
^I%63^K%72^V%56^L%82^W%1
^I%76^K%74^V%64^L%32^W%1
^I%4^K%59^V%60^L%441^W%2
^I%138^K%67^V%56^L%31^W%1
^I%143^K%67^V%60^L%38^W%1
^I%141^K%76^V%60^L%143^W%1
^I%4^K%60^V%60^L%415^W%2
^I%133^K%72^V%52^L%80^W%1
^I%67^K%74^V%60^L%83^W%1
^I%72^K%76^V%60^L%75^W%1
^I%66^K%78^V%52^L%92^W%1
^I%68^K%59^V%60^L%419^W%2
^I%6^K%79^V%64^L%37^W%1
^I%128^K%67^V%60^L%33^W%1
^I%139^K%67^V%64^L%27^W%1
^I%140^K%57^V%56^L%424^W%2
^I%1^K%72^V%60^L%141^W%1
^I%136^K%74^V%60^L%78^W%1
^I%69^K%72^V%60^L%76^W%1
^I%67^K%71^V%56^L%76^W%1
^I%66^K%69^V%48^L%78^W%1
```

図３　　Ｕｎｉデータ例
（バッハ　メヌエットより）

4．2　概要

現在のところ、演奏データの編集を視覚的に行うための視覚エディタ、ＭＩＤＩ規格による楽器制御機能、範囲指定による整数の検索機能を備えたテキストエディタ、マクロ実行部という４つの部分から成り立っている。

視覚エディタは、数値の羅列であるＵｎｉデータを人間にとって見易い形にし、編集し易くすることを目的として製作したエディタである（図４）。横軸に時刻、縦軸に音の高さをとり、演奏データの各音を”音線”つまり音の長さを表す線で表示し、同時にその音の強さも画面の上方または下方に棒グラフで表示するようにした。これを用いて、各音の高さ・長さ・強さ・声部などの変更、音や小節線の挿入・削除・移動を行うことができる。

楽器制御は楽器－マイコン間のＭＩＤＩ規格に基づく通信により行われる。すなわち、本システムはこれによりＭＩＤＩ楽器からの演奏データ入力及びＭＩＤＩ楽器の自動演奏を実現している。

テキストエディタは文字列の検索機能が優れているＱＥＤを拡張したもので、テキストファイル上に展開した演奏データを編集・合成することを目的としたエディタである。

マクロ機能は、特に演奏データの編集・解析・合成などの処理を記述し実行する目的で製作され、Ｕｎｉデータ形式のファイル上に書き出してある値を変数として扱えるものである。これはエディタのマクロというよりも手続き型プログラミング言語に近い言語仕様となっている。しかし、設計思想及びその実現方法に多少問題があり現在検討中である。

簡潔に述べると、本システムは、視覚エディタを中心として、演奏データに対してミスタッチの修正や表情付け等の処理を施し、音楽性の高い演奏データを作成することを容易にするシステムであるといえる。

システム構成は、図５のようになっている。テキストエディタからは視覚エディタ、楽器制御部、マクロ実行部を呼び出すことができ、視覚エディタからはテキストエディタ、楽器制御部を呼び出すことができる（自動採譜システムについては後述）。すなわち、１つ１つの音や小節線について行う細かい編集は視覚エディタを用い、マクロ作成やファイル操作等の処理はテキストエディタを用いる、というように作業の内容によってそれらを使い分けられるようになっている。

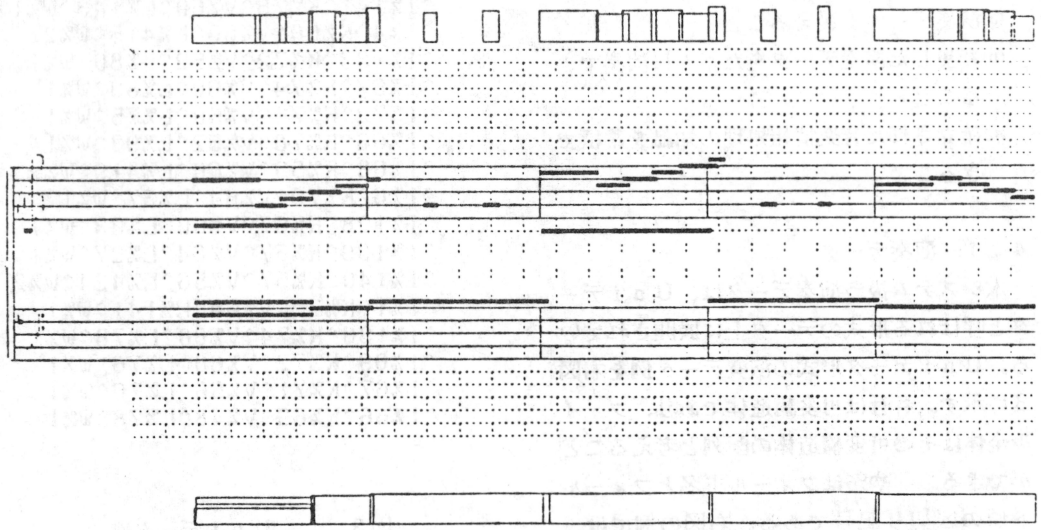


図４　　視覚エディタの出力画面

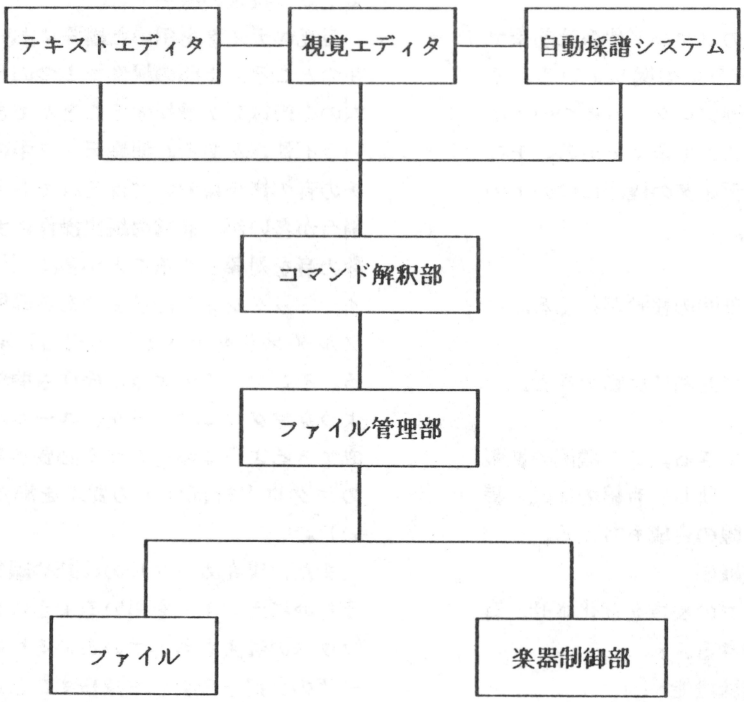


図5　　対話型音楽情報処理システム構成図

４．３　視覚エディタの仕様

本項では視覚エディタの画面表示の特徴を楽譜との違いを述べながら挙げていく。

（五線）上下２段の五線を表示し、８８鍵のピアノの鍵盤の音高が表示できるようになっている。声部には関係なく高音は上段の、低音は下段の五線上に表示される。五線は、緑色の実線で表示している。

（音線）音の長さを五線上に線の長さで表している。声部の違いや、白鍵にあたる音と黒鍵にあたる音の違いは音線の色によって区別している。

第１声部の白鍵にあたる音・・・・・・・・紫色
第１声部の黒鍵にあたる音・・・・・・・・赤色
第２声部の白鍵にあたる音・・・・・・・・水色
第２声部の黒鍵にあたる音・・・・・・・・青色

ここで、休符は音線とその次の音線との間の音線のない部分の長さで表される。

（ボリュウム棒グラフ）棒グラフの書かれる場所は画面の上下で、第１声部の場合は対応する音線の真上、第２声部の場合はその真下に表示される。棒グラフの長さは音量を表し、色は対応する音線の色と同じである。

（時間目盛り）本エディタの横軸は時間を表すが、そのレンジは１ドットあたり 8 msec にとってある。１画面の時間幅は、8 msec× 556 dots ＝ 4448 msec＝ 4.448 sec である。

（カーソル）本エディタでは、カーソルを画面上で動かして、音線、小節線、ボリュウム棒グラフの画面編集や画面のスクロールを行う。形はX印形で、色は通常白、編集モード中では黄色である。

４．４　機能

テキストエディタのコマンド体系は前項で述べたとおり、ＱＥＤに準拠している。また、ＭＩＤＩ楽器制御プログラムについては概要の部分で述べたことで全てである。そこで、ここでは視覚エディタの機能についてのみ述べることにする。

①音線移動機能
音高間、左右、声部間の移動が行える。

②小節線移動機能
小節線を時間軸上で左右に移動できる。

③音線伸縮機能
音線の長さを伸縮できる。この機能の拡張として音線の削除、新しい音線の作成、横に並んだ２本の音線の合成も行える。

④ボリュウム調整機能
ボリュウム棒グラフの長さを変化させ、音量の上げ下げができる。

⑤小節線挿入・削除機能
表示画面を見やすくする目的でこの機能を導入した。

本視覚エディタは、カーソルを画面上で動かすことによって音線、小節線、ボリュウム棒グラフの画面編集を行う。操作手順は次の通りである。

１）カーソルを画面上の編集対象の位置へもっていく。
２）編集モードに入りカーソルを動かして処理を行う。
３）編集が終了したらモードを出る。

このとき、編集したい対象が複数あればそれぞれに対して１）から３）を繰り返す（編集モードに入っているときはカーソルが黄色になり、ユーザにすぐわかるようになっている）。

また、カーソルは片手で簡単に操作できるように、ほとんどの操作をテンキー上で行えるようにしたところにも特徴がある。

４．５　将来の展望

視覚エディタを用いた編集においては、現在のところ、１回の操作で１つの音または１本の小節線しか操作することができない、という不便さがある。演奏データ中のミスタッチの音の修正においてはそれでも十分である場合が多いが、通常の編集操作においては複数の音を対象とすることが多く（例えば、ダイナミックレンジの変更やある範囲の音にリタルダンドをかける等の場合）不自由である。そこで、そのような操作も簡単に行えるようなマクロコマンドを、ユーザが自由に作成できるような環境を作る必要がある（現在のマクロ実行部はその要求を満たしていない）。

また、現在カーソルの移動や編集操作を行うためにテンキーを用いるようにしているがマウスの導入も考えている。それによってユーザの負担を多少とも減らすことができるであろう。

テキストエディタを用いた編集においては図３のようなＵｎｉデータ形式の演奏データファイルを直接操作して編集を行う。しかし、数値というものはやはり人間にとっては分りにくいものであるので、テキストエディタを用いてＵｎｉデータを直接編集するのは面倒であり困難でもある。そこで、将来的には演奏データの編集は全て視覚エディタ上で行えるようにする予定である。

現在製作中の自動採譜システムも本システムに組み込まれる予定である。自動採譜システムとは、演奏データから楽譜データを得る（演奏データを小節分けしたり、各音を音符に直したりすること）システムであり、このシステムによって音楽鍵盤による楽譜データ入力が可能になる。ここで、そのようにして得られた楽譜データから楽譜画面を出力するプログラムも現在製作中であることを付け加えておく。

４．６　まとめ

対話型音楽情報処理システムの目標としていることは、次のようなことが可能な環境を提供することである。

①演奏データをユーザの好みの表情付けがなされたものに変えられること

②音楽演奏の解析・自動化に関する研究を進められること

③高度に知的な情報処理を実現できること

すなわち、それぞれのユーザが何度も試行錯誤を繰返し、良い表情付けがなされた演奏データを作るにはどうしたらよいかを経験的に知ることができ、そのような演奏データを容易に作ることができれば、目標は達せられたといえるであろう。

５　おわりに

以上、現在、ＰＳＹＣＨＥで開発中の２つのソフトウェア、ミュージック・コンパイラと対話型音楽情報処理システムについて述べてきた。ＰＳＹＣＨＥでは、この他に、ＬＰやＣＤなどの音声信号から、楽器制御用の演奏データを得るソフトウェアの研究などもなされている。今後、機能をさらに充実させ、より高度な自動演奏が実現できるシステムをめざしている。

参考文献

［１］笹川、三好、五十嵐：汎用音楽記述言語の設計・試作、第２５プログラミング・シンポジウム報告集、ｐｐ．１６－２５（１９８４）

［２］小川、三好、五十嵐：ピアノ演奏のコンピュータ分析とその自動演奏への応用、第２７プログラミング・シンポジウム報告集、ｐｐ．１５９－１６８（１９８６）