# AUTOMATIC TRANSCRIPTION OF GERMAN LUTE TABLATURES

## AN ARTIFICIAL INTELLIGENCE APPLICATION

*Hélène Charnassé & Bernard Stepien*

**Abstract**

Before the 18th century, a great deal of instrumental music was commonly notated in tablature. This is an alphanumeric notation that serves one main purpose: helping the specific instrumentalist to perform a piece by indicating finger positions on his instrument. This, however, is achieved at a cost, the loss of the musical structure of the piece. The ERATTO team has developed integrated software to handle all aspects of tablature transcriptions into modern notation. This includes data capture and editing, translation, restructuring and generation of graphics. The first part of this paper describes the musicological background and the history of the project. The second part describes various problems encountered in the computer implementation, and discusses alternative ways to solve the problem using artificial intelligence techniques.

*Computer Representations and Models in Music*
*ISBN 0-12-473545-2*

# PART I: MUSICOLOGICAL BACKGROUND

## THE TABLATURE NOTATION CONCEPT

From the 15th century to the middle of the 18th century, the repertoire of instrumental music for lute, guitar, harp, viols and organ in some countries was notated with alphanumeric characters. This peculiar notation is called a tablature. It is a practical way to show finger positions on an instrument, each position depicted by a different code. As a result, only a few interpreters or musicologists can have access to this music. A non-specialist has to transcribe the notation. Hundreds of collections are waiting to come out of the dark through transcription and publication by musicologists. In 1973, Hélène Charnassé brought about the creation of a research team, ERATTO (C.N.R.S.), whose task is to perform automated transcriptions of German lute tablatures.

In Germany, during a large part of the 16th century, authors[1] used for the lute a specific notation different from Italian or French lute tablature. Each character represents a position of a finger on a string/fret intersection (see Figure I.1). The notation is composed of only two elements: the alphanumeric symbols and necessary musical symbols. The first characteristic of this notation is to be practical for the player. However, in most cases, it is abstract in its presentation: it is not intended to show visually the musical structure of the piece through the typographical layout of characters. In Figure I.2, one sees that German lute tablature uses virtual horizontal lines to help the lutenist to read the piece. Each virtual line theoretically corresponds to a voice. However, due to undocumented reasons, the position of a character in a virtual line corresponding to a given voice is not always respected, and sometimes even completely ignored.

The second characteristic of this notation is that it is not intended to show the exact duration of each sound. The rhythmic symbol on the top of the vertical shows only the local duration as a whole. If a vertical contains three characters, three strings must be plucked, but the rhythmic value indicates only the duration of the shortest sound. Other sounds can have a longer duration; for instance to sustain a melodic line. This sustaining is rarely indicated in tablatures.

[1]Throughout this chapter, the term "author" is used for the originator of the tablature, which, as explained below, would generally have been an arrangement for lute of music composed by someone else.



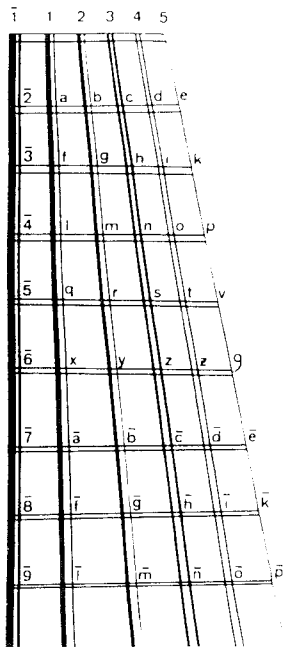

**Figure I.1.** Correspondence of tablature characters to positions on the neck of a lute

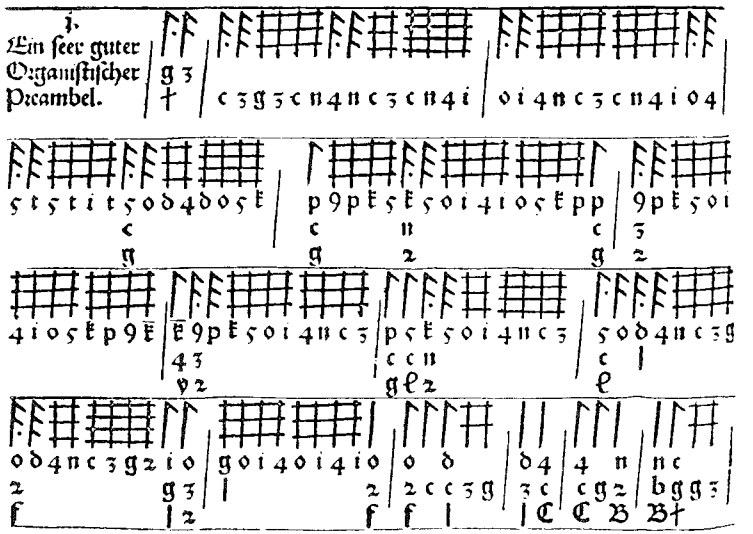

**Figure I.2.** German lute tablature by Hans Newsidler. *Der ander Theil des Lautenbuchs*, Nuremberg, 1536, Aiij v.

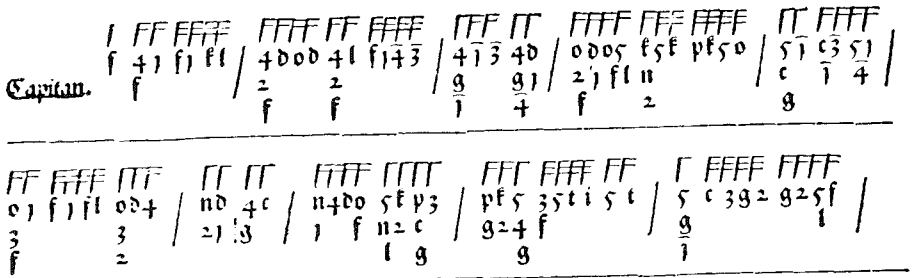


**Figure I.3.** Abstract tablature by Hans Gerle. *Tabulatur auff die Laudten*, Nuremberg, 1533, f G v.

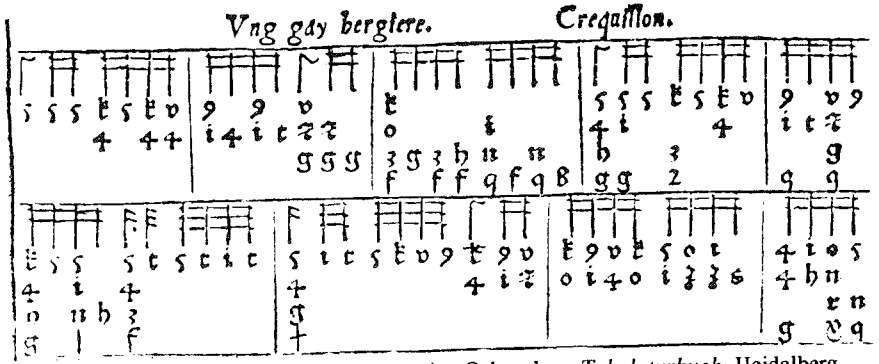


**Figure I.4.** Structured tablature by Sebastian Ochsenkun. *Tabulaturbuch*, Heidelberg, 1558, f 86 v.

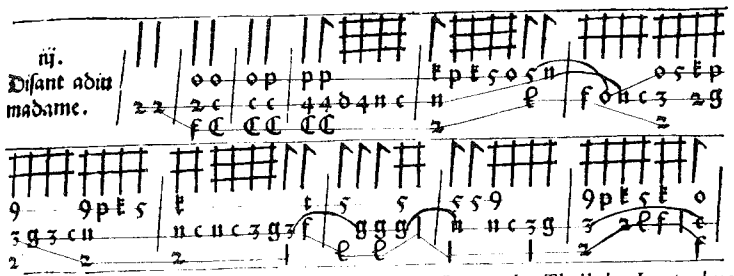


**Figure I.5.** Mixed tablature by Hans Newsidler. *Der ander Theil des Lautenbuchs*, Nuremberg, 1536, f Biij.

The third characteristic is that the set of characters used is not constant. Some of them vary from one author to another, and sometimes from one collection to another of the same author. Those characters do not correspond to given sounds. The corresponding scale is determined by the tuning of the lute, most often in G or A during this period, depending on the altered notes found in the piece concerned.



There are three types of layouts in use in Germany during the 16th century:

1. The abstract tablature, used by Hans Judenkünig and Hans Gerle, which regroups the characters toward the top or the bottom of the tablature (Figure I.3).
2. The structured tablature, used by Sebastian Ochsenkun. In most cases, the characters are positioned according to the voices to which they belong (Figure I.4).
3. The mixed tablature, used by Hans Newsidler, which uses both abstract and structured approaches (Figure I.5).

The interpreter is consequently faced with a restructuring problem. However, despite a superficial inconsistency, there appears to be a very strict internal logic that was well known by 16th century musicians. This logic has partly disappeared with the loss of tablature usage and our goal is to rediscover it. The work has been accomplished in two steps using two radically different computer technologies.

## HISTORY OF THE PROJECT

### A raw transcription

The ERATTO team started to experiment using only the "explicit" elements found in the tablature: i.e. those which are directly translatable. It published "raw" transcriptions by automating the trivial aspects of the problem such as character translation and graphics. The restructuring problem could be addressed only once we had some kind of readable music that was easier to work with. The first experiment was performed using the abstract tablature of Hans Gerle namely the *Tabulatur auff die Laudten*, published in Nuremberg in 1533 (Figure I.6). This work was performed on a mainframe computer using a Benson plotter.

The only elements found in this transcription are:

— the heights of the sounds, on two staves
— the identification of altered notes (accidentals)
— the stacking of notes for chords.

For the tablatures themselves, we cannot think that the lutenist was playing the piece as a melodic line punctuated with chords. The musical style used in Germany at that time was the polyphonic style based on counterpoint rules with superimposition of "voices".

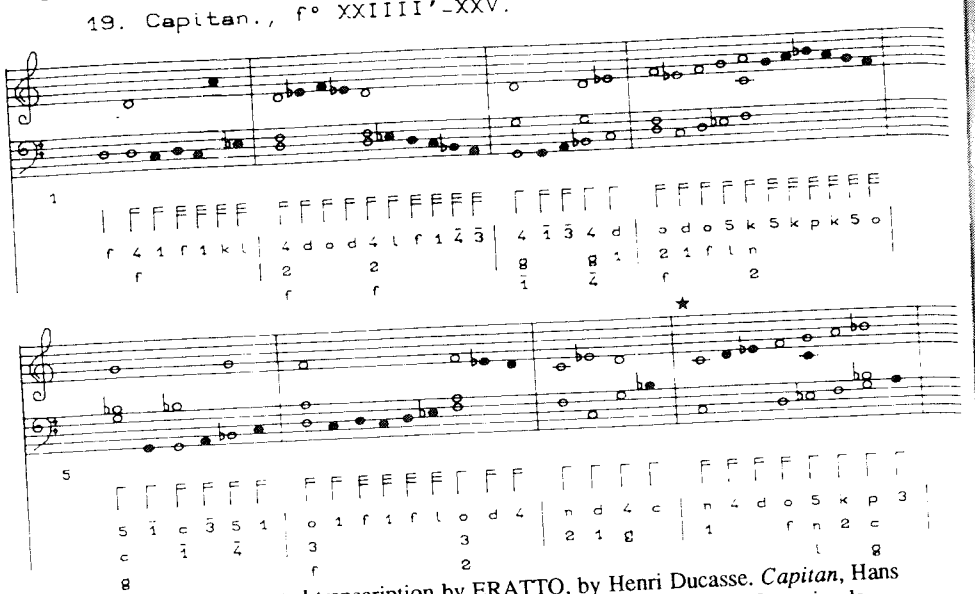


**Figure I.6.** Automated transcription by ERATTO, by Henri Ducasse. *Capitan*, Hans Gerle, *Tabulatur auff die Laudten*, Nuremberg, 1533, f G v. Société francaise de musicologie, 1975, II, p.34.

Consequently, the notation must have contained "implicit" interpretation rules that were as prescriptive as the explicit data in the tablature. Effectively, a close look at the results of the raw transcription shows an underlying organisation of this piece in three separate voices. The continuity of melodic lines can be detected visually. It was tempting to attempt a fully automated structured transcription. This required an extensive musicological analysis to enable the computer implementation.

At this point, there was a seven year break, employed to perform musicological analysis. There were two questions to be answered: first to determine the "implicit" data which allowed the lutenist to reconstruct the good polyphonic structure, second to find what kind of logic was used by the author in his notation. After two or three years of research, we understood that the only answer was the musical logic of that time. The lutenist must have known the essential rules of counterpoint and applied them. As those rules are well known today, it was possible to determine new transcription algorithms. The restructuring work was achieved at the beginning of 1986 with the cooperation of Bernard Stepien for the computer implementation.



## Towards structured transcriptions

This step was performed using a micro computer to take advantage of instant graphics and interaction capabilities. For this part of the work, we had to address three kinds of problems.

**Basic translation problems.** A German lute tablature does not contain such things as accidentals, consequently when translating tablature characters we need a way to determine the correct note and any associated accidental. This approach is totally independent of the character sets used and uses strictly numeric comparisons (see below).

**Rhythmic symbol interpretation problems.** The true duration of a note has to be more specifically calculated. A rhythmic symbol is not determining by itself: a note could sound longer than indicated. Physical properties of vibrating strings and the instrument concerned must be used in conjunction with non-physical properties of a musical nature such as compositional methods used by the author. The calculation of the true duration of a note has to incorporate four elements:

— the number of parts in a polyphony
— the determination of the melodic line corresponding to a part
— the sustaining necessary to maintain the continuity of this line
— the analysis of other voices, to determine the musical validity of sustained notes.

**Polyphonic interpretation problems.** The problems in dealing with the restructuring of the polyphony vary from one author to another. For chords, the characters are stacked up according to the height of the notes; isolated characters are usually pushed up or down along the virtual lines. Most German lute tablatures we are working on are instrumental versions of vocal pieces, written in counterpoint, and the tablature usually follows the vocal model. According to the custom of that time, the lutenist was supposed to follow the same vocal model, but using a notation that was introducing a kind of fuzziness.

Those vocal pieces are very often preserved. As they are notated in proportional notation, they give the melodic line of each voice, and we can easily reconstruct the polyphonic structure. These models are of great help for analysis, creation of algorithms and verification of results.

However, the lute tablature is an instrumental setting, which also has its own rules. Other problems have to be solved whenever the author modifies

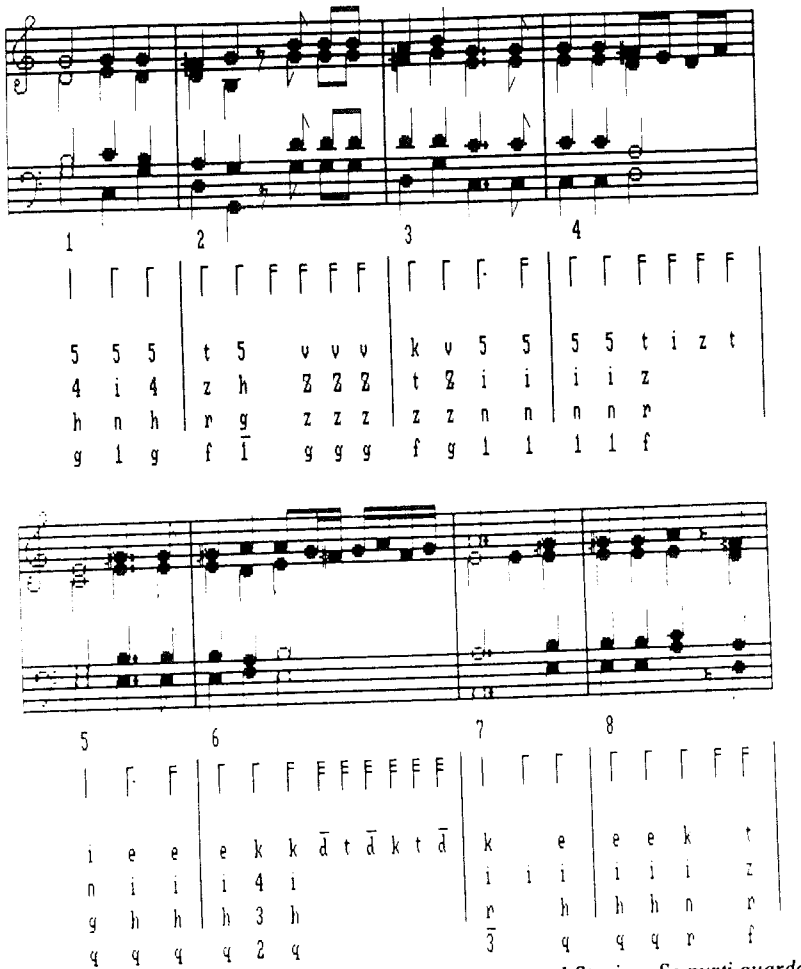

**Figure I.7.** Automated restructured transcription by Bernard Stepien. *Se purti guardo*, Sebastian Ochsenkun, *Tabulaturbuch*, Heidelberg, 1558, f 86 v.

the model, for instance when he suppresses one of the voices or when he merges two voices in a long ornamental melodic line.

The best way to have a chance of discovering the restructuring rule was to study pieces that have already been structured by the author himself in the 16th century. This is the case for the tablature of Sebastian Ochsenkun (Heidelberg, 1558). In this work, the author most often shows the polyphonic structure of the pieces by assigning each character to the virtual



line corresponding to a voice. In such a case, the transcription is easy to perform, only note durations and graphical aspects have to be derived (Figure I.7).

## FINDING A RESTRUCTURING METHOD

In order to discover structuring rules we have made unstructured versions of the tablatures of Ochsenkun by pushing all characters toward the top of the tablature like Hans Gerle. The original version was used as a bench mark against which our trials and errors could be compared and conclusions could be drawn.

Since this music was originally intended for a vocal ensemble, we derived a concept of "chord saturation". A saturated chord is a chord were all voices participate. The first attempt consisted in assuming that a saturated chord is a clear indication of the range of each voice. We consequently must find out the continuity of a melodic line between two saturated chords.

The first rough method was to conceive a system that recognises the continuity of a melodic line. This method generated some interesting results with little effort. Errors were analysed and new rules were created to cope with contexts or exceptions. The comparison of the artificially restructured version with the original bench mark version showed that this kind of algorithm produces fairly good results with sections of the pieces written in "free" counterpoint (Figure I.8). Sometimes even, the results of the computation turned out to be more logical than the version given by the author.

However, this type of method fails in cases of voice crossing that occur when an imitation of a theme is performed. The analysis of vocal models shows that the authors used two kinds of logic:

— the "free" counterpoint style of composition
— the imitation style, where a melodic and rhythmic pattern is reproduced in different voices with or without variations.

These two different styles of writing implied a variety of algorithms to perform the automated restructuring. We developed the methodology using the mixed tablatures from Hans Newsidler as a testing ground.

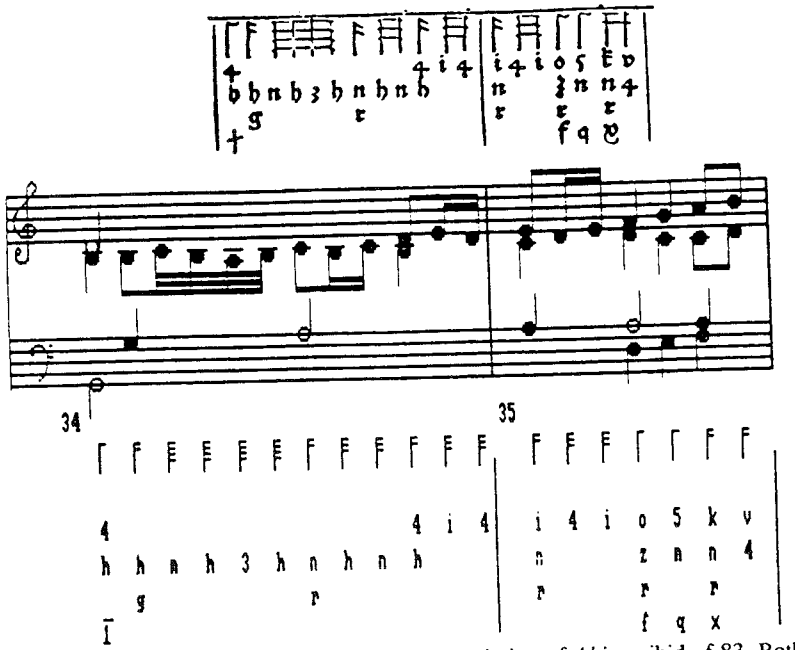

**Figure I.8.** Original tablature vs. automated transcription of *Ahime*, ibid., f 83. Both versions agree.

## The general algorithm for counterpoint development

The musical analysis of the vocal model and its adaptation in tablature notation shows that the performer has to use specific rules well known by musicians of that time. First, the lutenist has to reconstruct a polyphonic piece. The number of notes in saturated chords indicate the number of voices. Second, the lutenist has to apply the basic rules of the counterpoint to reconstruct the continuity of melodic lines.

Besides these rules, there are other implicit rules that are common sense for a trained lutenist. I came across these obvious rules in transcribing manually many tablatures. For instance, Antonio de Cabezon, in his keyboard tablature, fails to specify the correct octave level for high pitch symbols in continuous melodic lines. Continuity is a sufficient criterion for the player. Another example: in Ochsenkun tablatures, a bass symbol is sometimes pushed up on another melodic line when there is no doubt about its real position. Another observation: a short dotted rhythmic value implies that the complementary note belongs to the same voice.



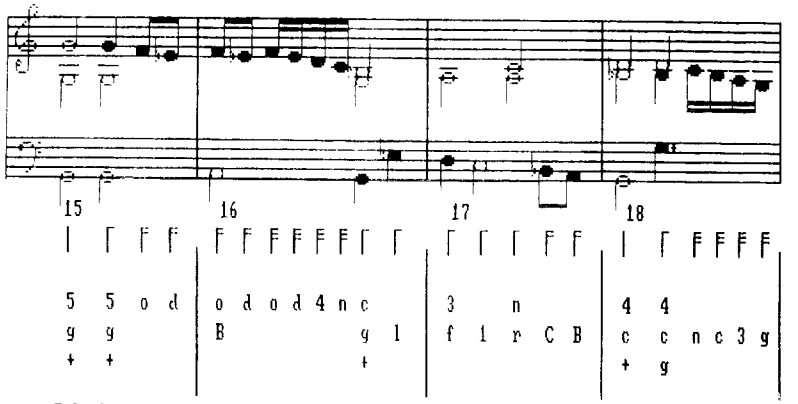

**Figure I.9.** Automated restructuring of a fragment of counterpoint. Mixed tablature, *La plus te plus*, Hans Newsidler. *Der ander Theil des Lautenbuchs*, Nuremberg, 1536, f Biij.

These musical rules have been implemented in a "rule based" expert system (see below). Preliminary results showed that the assumptions made for Ochsenkun were still valid for Hans Newsidler. The main difference consisted in the priority of indicators to decide voice assignment (see Figure I.9).

## The processing of imitations

Initially, imitations looked like a major hurdle for automated transcription. For melodic lines, continuity is no longer the essential rule: voices sometimes cross each other. Since a tablature was often itself a transcription of a well known vocal piece, the lutenist was generally very familiar with the themes. The only solution appeared to be to feed the computer with the original themes, which would again require a long manual task to find original vocal documents. We came quickly to the conclusion that this was highly impractical. The solution came from traditional similarity finding algorithms; this is where computer science is of definite help.

Figure I.10 shows the three stages one can go through when dealing with imitations. The top example shows the problems encountered in maintaining the stacking order of characters in the tablature according to note height in the face of an imitation where parts cross. The middle example shows a nice contrapuntal continuity possibility, but this turns out to be wrong. The bottom part shows the imitation clearly separated by the computer. Each algorithm that we use is tested on a significant sample of pieces depicting various cases of writing styles. The results of the processing are compared to the solutions

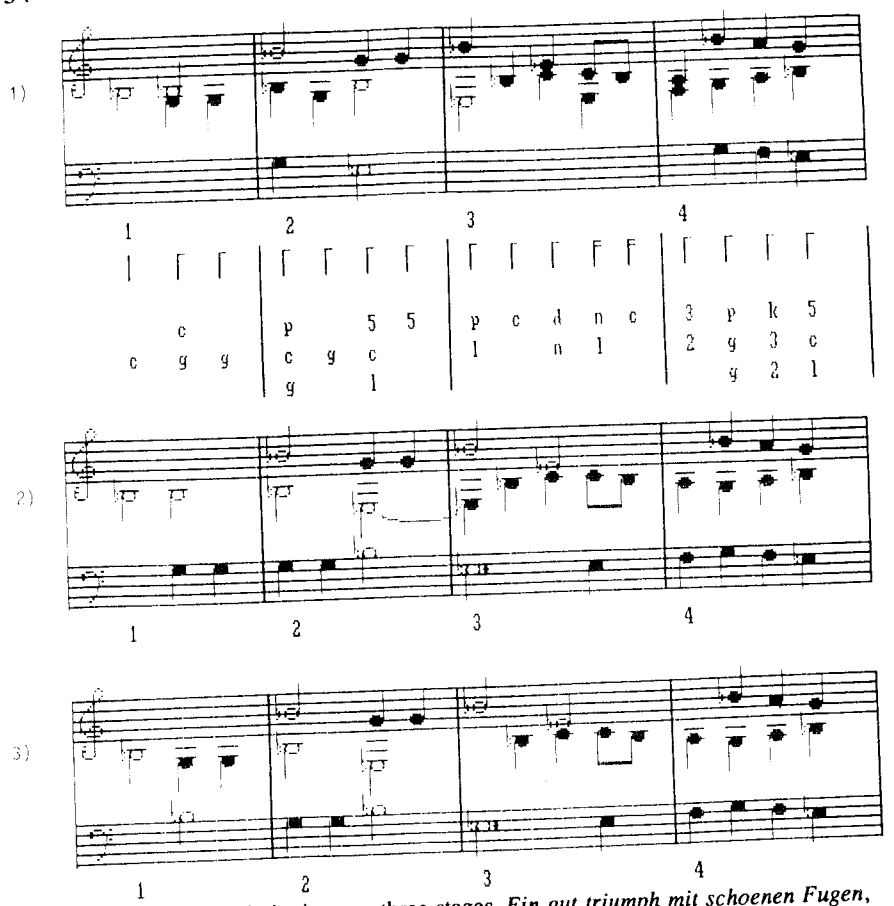

**Figure I.10.** Handling imitations — three stages. *Ein gut triumph mit schoenen Fugen*, ibid., f Diij v.

given by the vocal model. Algorithms are then screened, some are discarded, others are either kept or further refined. As the processing of new sections of the repertoire proceeds, new peculiarities are discovered and translated into new computation rules.

Presently, we are beginning to analyse the collection: *Musica and Tabulatur* (Nuremberg, 1552) which contains many Italian lute dances transcribed in German lute tablature by Hans Gerle. It opens the way to the transcription of the new style of music of Italian and French lute tablatures.



# PART II: COMPUTER IMPLEMENTATION TOPICS

## SYSTEM OVERVIEW

We will briefly describe the general structure of the system and some basic concepts that were used to increase efficiency.

### Summary of requirements

As we have seen above, a lute tablature is composed of two main pieces of information:

— an alphanumeric character that indicates a finger location on the lute

— the minimal rhythmic value or duration of a vertical.

There are major elements missing in a tablature to obtain readable music with a simple translation of characters: the exact duration, the graphical aspect and any alteration of a note must be computed once we know the exact location of that note in the polyphony, which most of the time is another key piece of missing information in tablatures.

There are consequently five main tasks to accomplish in this project:

— translate a tablature character

— establish the alteration of the notes (flat or sharp)

— establish the polyphonic structure

— derive the duration of the notes

— determine the graphical aspect of the notes

These tasks are performed in sequence, one depending on the previous one.

### Data structure

The character sets, or layout of characters on string/fret intersections, vary considerably from one author to another and one country to another. Consequently, if we want to design a system that is general enough to handle that variety, it is essential to work on a standard representation of a tablature. In addition to this problem, lutes can be tuned differently. Consequently, we use an internal representation, totally independent of tablature, that uses a continuous value system for each note or rhythmic value instead of tablature characters. The string on which a note is played is also of prime importance to determine duration or voice assignments and also to be able to translate the

music back into original tablature notation or into more accepted tablature systems such as Italian or French systems.

### Processing modules

**Character set definition module.** Before translating a tablature we need to know what note each character represents and how the lute is tuned.

**Data capture and editing**. Since a tablature looks like an ordinary text composed mainly of characters, the most obvious way to handle data encoding and modification is through a sort of word processing system. Optical reading has been considered but has been assigned a lower priority for the moment. The word processing option requires very little immediate time investment.

The user can see on the screen exactly what he sees on the manuscript. He can browse through his tablature using the arrows and the page up and down keys along with a variety of function keys, and perform editing functions such as adding or deleting bars or verticals. The screen is laid out like a template and only the zones defined by the template are accessible. This is the only visible difference with respect to a normal word processor. The invisible part is of course the converter into the internal representation described above. The tablature editor is an independent program called S.I.T. (*Saisie Interactive de Tablature*).

**Restructuring of the polyphony module.** This is the module that required the most research. It works on the internal representation of the music, and is described at length below.

**Duration establishment module.** Assuming that the exact location of a note in the polyphony is known, the exact duration of that note can easily be calculated by accumulating the minimum vertical durations notated in the tablature from the point at which that note begins until one of the following conditions occurs:

— another note has been played on the same string
— a new note appears in the same voice regardless of which string it was played on
— a new note conflicts harmonically with the current note.

This shows the reason for keeping the string information in our data structure, and also the importance of knowing the exact structure of the music.



**Constructing the graphic proportional notation.** Once the conversion of tablature notation has been performed, the ultimate goal of this activity is to display or print the music. Here again, graphic components of a note such as stem orientation, stave location, decomposition of durations into admissible rhythmic structures, etc. are missing in a tablature and have to be logically derived. All of this can be determined from the three components: voice assignment, height, and duration of a note. This module performs this process and displays the results. Display is crude at this time mostly because we are waiting to integrate our results in one of the off-the-shelf high quality music printing packages that are widely available. At this stage we prefer to devote our energy to topics that have not been addressed by the scientific community.

## THE USE OF ARTIFICIAL INTELLIGENCE IN THE POLYPHONIC RESTRUCTURING PROGRAM

### The restructuring concept

Imagine three conversations going on simultaneously at different rates. If one writes a transcript of those conversations showing their relative positions in time, one might obtain something like:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1: | the | camel | | has | four | legs | and | only | | one | hump |
| 2: | looking | | from | a | tree | the | | legs | look | nice | |
| 3: | once | upon | | a | time | there | was | a | | nice | king |

An unstructured lute tablature would represent such a conversation the following way:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| the | camel | from | a | has | four | the | legs | and | only | nice | one | hump |
| looking | upon | | | tree | time | | there | legs | look | nice | king | |
| once | | | | a | | | | was | a | | | |

The challenge of transcribing lute tablatures is indeed to restore the original three conversations from the scrambled version. As one can see, this is not simple, even for our English text where each word has a very specific meaning. One can imagine the problem in music where each note does not have an obvious independent meaning, but is usually very dependent upon context.

What are the strategies available for such a task? In the case of our English scrambled conversation we may use all kinds of common sense rules drawn

mostly from grammar and context. For instance, the fragment "and only nice one hump" shows an adjective before a number which is grammatically wrong, so we could deduce that "nice" must belong to another person's text and consequently we can restore that fragment to: "and only one hump". This in turn makes the end of the conversation look like:

and only one hump
legs look nice king
was a nice

Here again there are two obvious incompatibilities:

— the fragment "legs look nice" cannot be followed by a noun.

— the fragment "was a nice" sounds incomplete.

Consequently, since "king" is superfluous in the second person's conversation and there is a missing word in the third person's conversation, it seems obvious to move the word king to the third person's conversation. That portion of the conversations was relatively easily restored.

In music there are similar "rules" that can be used. However, they are not as definite as in a human language. While a word in English can, most of the time, belong to only one kind of grammatical category, in music the same note can mean very different things depending on its context. To make things even worse, when two voices use the same note in synchronisation the latter is usually notated only once in a lute tablature. Consequently, one very often has to find out if the next note to be examined could belong to two different voices.

## A one-pass restructuring exploration

If we look at a modern representation of tablature music, the first property that strikes us is the presence of ornaments that link vertical clusters of notes as bridges link islands.

**The principle of continuity.** In German lute tablature music, melodic lines are often composed of small intervals. More specifically, this music is based on a vocal concept where traditional voices like bass, tenor, alto and soprano are set at given ranges and rarely trespass each other's boundaries. Often in a chord where there are as many notes as the maximum number of voices (we call this a saturated chord), each note determines the range of each corresponding voice, and any note in an ornament that is within range of that chordal note is believed to belong to the same voice.



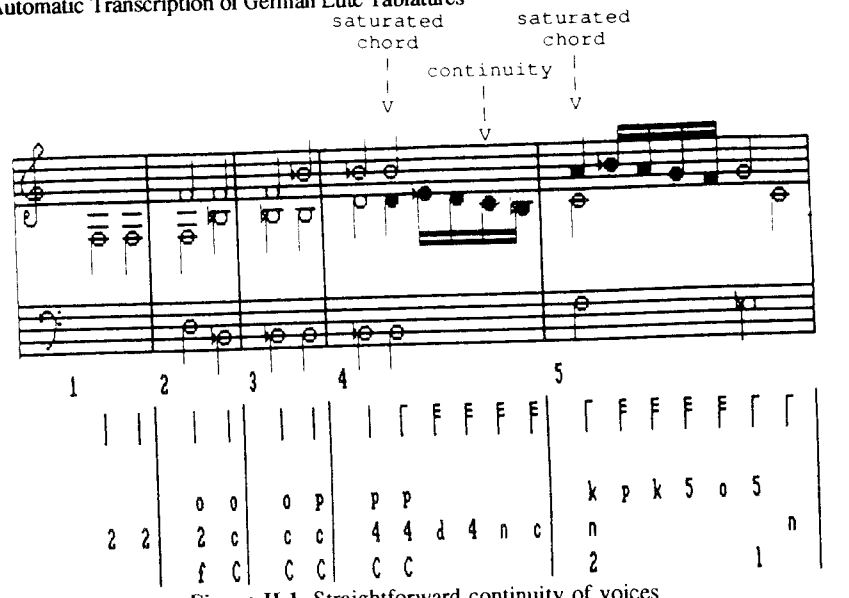


**Figure II.1.** Straightforward continuity of voices

In a one-pass concept we start scanning the notes from one end of the saturated chord's frame and we assign each note to a voice according to the value of the minimum interval with the neighbouring notes. Once a note has been assigned we move on to the next note and we use the same process (this is similar to a contagion process, hence if an error occurs there is also a contagion of error), until all notes that lie between two saturated chords (framing saturated chords) have been processed. The music is thus processed in chunks delimited by saturated chords. The process is clear in Figure II.1 at bar 4 where there is no doubt that the ornament E♭, D, C, B♭ can be assigned to the second voice. The saturated chords at bar 4 and 5 that frame the ornament unambiguously indicate the second voice for that ornament.

**Continuity disruption.** Unfortunately not all cases are that easy. Notes that do not belong to ornaments are usually scattered in time. In Figure II.2 we see a middle D that is repeated from bar 27 to bar 29. In this case the note D also belongs to the framing saturated chords. This led us to establish a "last assigned note" concept (i.e. the last note assigned to a particular voice).

Disruption can also occur because an interval with the neighbouring or last assigned note is too large to enable a safe decision. We determined empirically that in those cases when a contiguous note shows a large interval it realigns with one of the distant saturated chord's notes. In Figure II.3 we

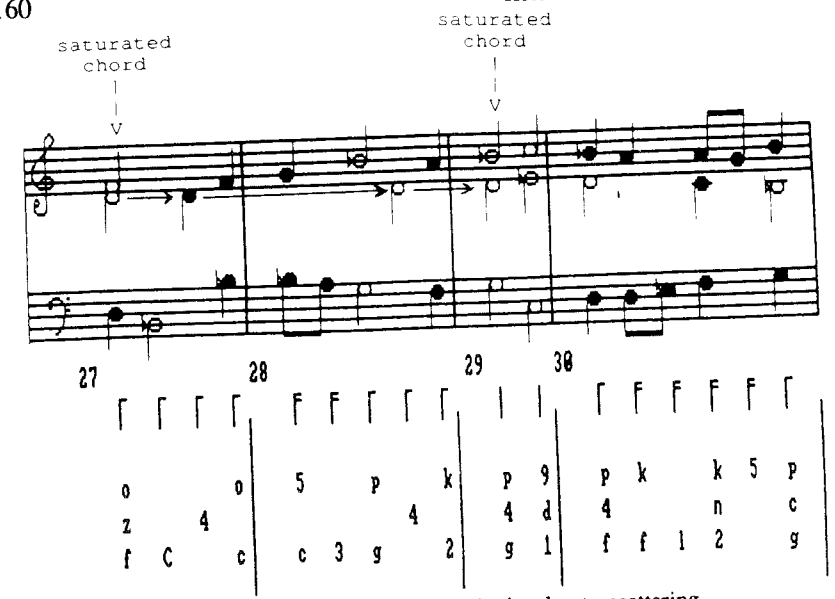


**Figure II.2.** Disruption of continuity due to scattering

see clearly in bar 8 that the middle B♭ next to the F is too far away from the F (we are working backwards here, see below) but fits nicely with the B♭ found in the framing saturated chords at bar 7 and at the end of bar 9.

**Ambiguity.** When the interval from a note to two reference notes (contiguous or last-assigned) is equal or very close, no clear decision can be

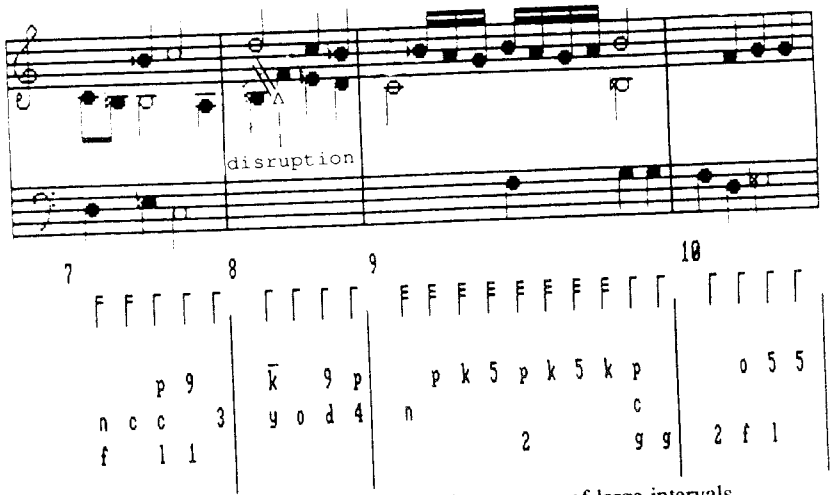


**Figure II.3.** Disruption caused by the presence of large intervals



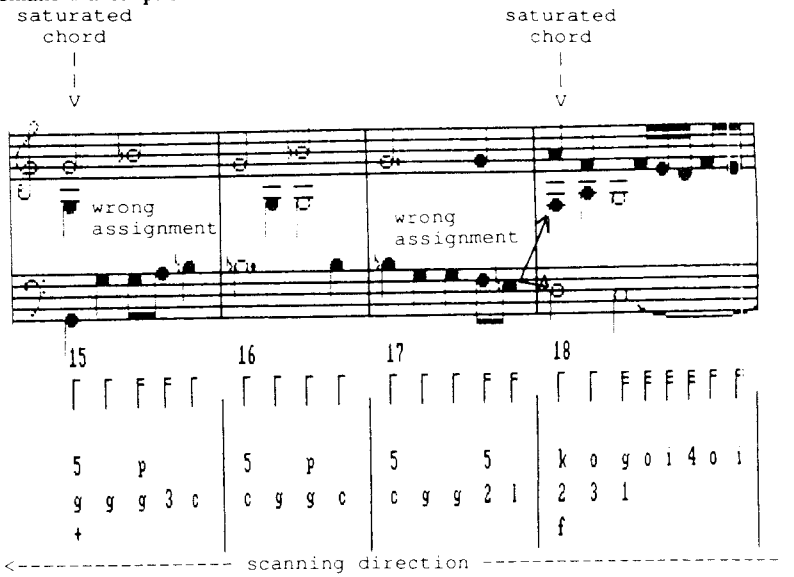


**Figure II.4.** Ambiguity due to small difference in intervals

made. In a one-pass system it is essential to resolve an ambiguity immediately because no further chaining can be performed if there is a missing link. In cases of ambiguity we usually look at the framing saturated chords to find a possible solution.

However, ambiguity could occur with any of the four indicators we have:

— the smallest interval to a contiguous note
— the smallest interval to a last assigned note
— the smallest interval to a note of the first framing saturated chord
— the smallest interval to a note of the last framing saturated chord.

Ambiguity is handled like disruption of continuity by looking at alternative indicators until one satisfies the chaining tolerance. If all fail, we start to loosen up the threshold values until one indicator is satisfied. In Figure II.4 we observe a very small difference in intervals between the E♭ at the end of bar 17 and either the F or D of the first vertical of bar 18. In a one-pass scanning this throws out the entire phrase that is linked to the E♭ when scanning from right to left (see rationale below).

**Disruption caused by voice crossing.** Voice crossing is mainly due to the presence of repetition of melodic themes. This is called an imitation and is

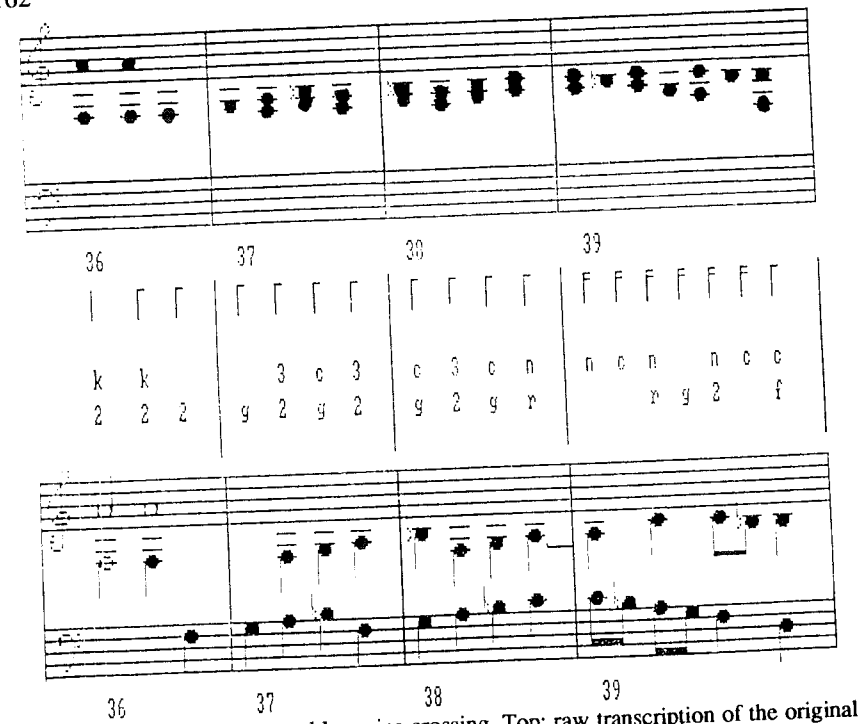


**Figure II.5.** Disruption caused by voice crossing. Top: raw transcription of the original version proposed by the author (the easy-to-play thirds show the practicality of tablature notation). Bottom: automatic theme finder detects two interwoven imitations. From *Ein gut triumph mit schoenen Fugen*, Hans Newsidler, 1536

very characteristic of ancient music. Figure II.5 shows a perfect case of voice crossing. The top part shows the results obtained by using the continuity principle, while the bottom part shows the themes clearly separated.

The first attempt to solve this problem consisted of feeding the computer with a copy of the theme. Using a finite automaton, it was possible to find the imitations, freeze them and complete the continuity process around them. This was consistent with the belief that musicians in those times knew the themes of popular songs and consequently could themselves separate them from the confusing tablature. However this approach is cumbersome from an automation point of view, and we quickly designed an automatic "theme finder" which is an application of the classic similarities search problem.

**Residual cases.** Even in a one-pass approach, when we cannot find a solution for a note belonging to an unsaturated chord, one can sometimes decide the voice assignment residually. In Figure II.6 the lower F in bar 15



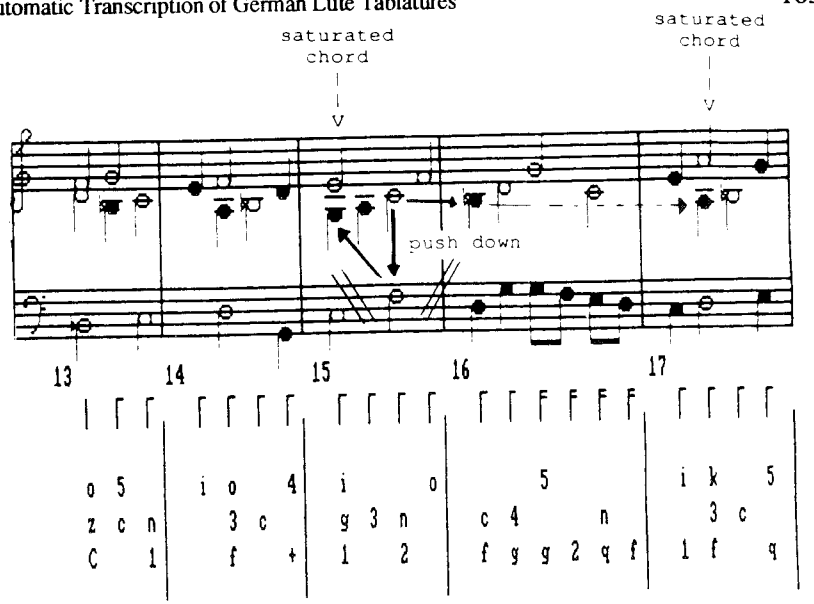


**Figure II.6.** Residual assignment of a note

shows a disruption in continuity with its neighbours. The C above the F seems to have no problems in being assigned to the middle voice. Consequently, the only logical position for the F is the third voice. The entire Figure II.6, however, shows the treacherous seas one is sailing when dealing with tablature music restructuring.

**Coping with variety.** The great variety of problems encountered rapidly led us to allow some parametrisation in various threshold values for continuity decisions. Priorities among indicators depending on various contexts such as rhythmic patterns, etc. were also implemented.

One of these was the scanning direction. Traditionally, musicologists perform this task manually by scanning from left to right as the music is actually written. We discovered empirically that in the case of a one-pass approach we encountered fewer problems when scanning from right to left. Although this seems very strange, there is some kind of underlying logic to it. This music is heavily embellished, and ornaments usually lead into a chord or sustained note. Consequently, as an ornament approaches its resolution chord, it seems to narrow down its range. In Figure II.7 the top part using right to left scanning looks more logical than the bottom part which uses the more natural left to right scanning.

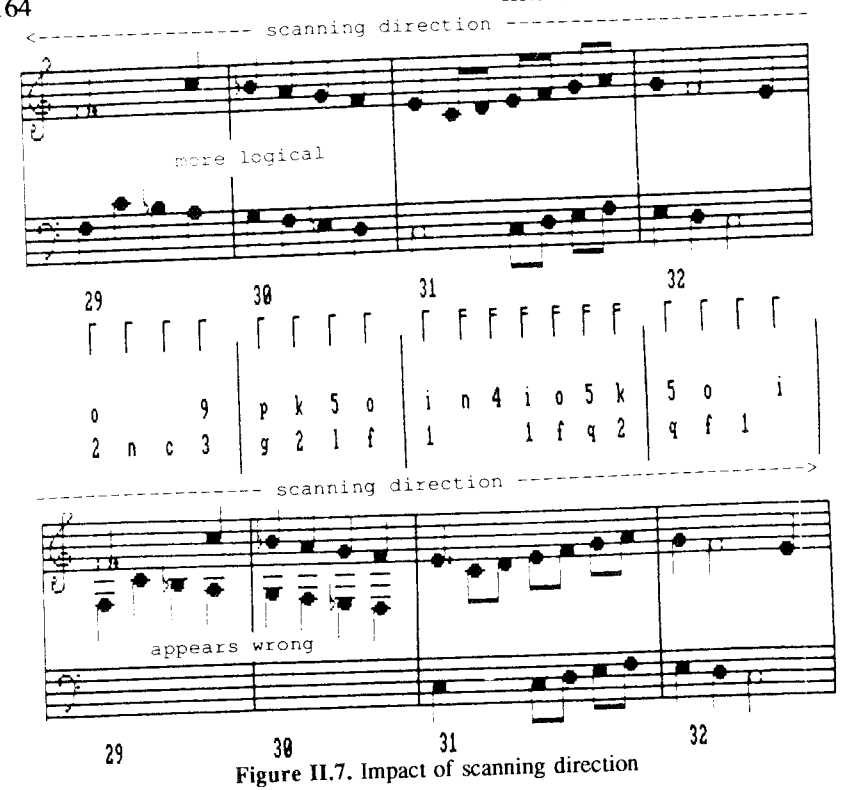


**Figure II.7.** Impact of scanning direction

**Conclusion on the one-pass approach.** The one-pass approach was the result of a step-by-step exploration of this music. Initially, very few rules were known or formalised. This approach enabled us to detect the various topics that had to be addressed. We were able to scan 170 pieces of an average length of 30 bars in a very short time compared to a manual process. Also, a great advantage of automation is the ability to rerun the process changing methodology at very little cost. (For example, a method of solving some of the problems seems to be to vary the scanning direction depending on the results.) One cannot even dream about this in a manual set up.

The one-pass approach generates good results in up to 80% of cases. The remaining 20% were mostly cases where a more complex approach of the residual derivation method could be of help. This, however, could not be performed in a one-pass concept. This is where more sophisticated artificial intelligence methods had to be used.



## Fixed rules inference system in Prolog

Two main problems had to be addressed:

— separating themes from "free" counterpoint

— resolving complex disruptions in continuity.

**Finding theme imitations by using Prolog.** Prolog is an interesting computer language based on a simple principle: success or failure. Failure is not final in Prolog: the system will look for alternative solutions by looking for alternative rules if a given rule is not satisfied, or for alternative data structures (called "facts") in its internal database. This feature is called backtracking.

Assuming we have a Prolog database containing notes described using the data structure `note(X,Y)` where `X` represents the pitch of a note and `Y` its location, we can easily find any occurrences or recurrences of any particular note. The following two segments of Prolog program code would represent a possible implementation (text between `/*` and `*/` is comment):

```
/* scan each note */
  findthemes:-
  /* find the next note from the data base */
       note(N,Location),
       findsimilar(N,Location),
       fail.
  findthemes.

/* for each note, find all recurrences */
  findsimilar(N,Oldlocation):-
  /* find the next occurrence of a note */
       note(N,Location),
  /* make sure it is not the same occurrence */
       Location <> Oldlocation,
       compare_remaining_notes(Oldlocation,Location),
       fail.
  findsimilar(_,_).
```

The predicate `findthemes` will merely scan the database containing notes. The value of each note will be returned in variable `N`. This value is passed to another predicate `findsimilar` that will find all recurrences of the given note. This would possibly generate the starting point of an imitation. Once a new starting point is available, the remaining notes of the two occurrences of a phrase are compared to verify that we have an imitation.

This is a well known technique of which implementation in Prolog is very simple. However, it can result in a lengthy search if unrestricted.

Furthermore, it was found that when run unrestricted, the search can find irrelevant similarities, so this simple principle was developed through rather complex research to determine the appropriate cutoff values to confine the search to imitations rather than general similarities. For example, an imitation is usually overlapping, so if a similar theme is encountered too far away, the process will stop because the new case is not part of an imitation but is more likely to be a favourite melodic fragment. Another restriction concerns the length of the repetitive pattern found. A short pattern does not qualify as a theme, since a theme is usually composed of at least four notes when using long durations and at least six for short durations.

Figure II.8 shows a good variety of imitations. The theme finder detects them and no continuity analysis needs to be performed for them. This is achieved by taking advantage of another feature of the Prolog internal database: the ability to assign temporary attributes to a note. If a note has been determined to belong to a theme imitation, we can "freeze" it to prevent its use in continuity analysis by merely asserting a fact (i.e. adding a new fact to the database) such as `frozen(Note,Location)`.

Besides its use in detecting crossing voices due to overlapping imitations, the imitation finder also detects immediate repetitions of patterns. In a repetition of a basic pattern, a one-pass scanning could reveal a disruption of continuity, and looking at alternative indicators might actually destroy the pattern. The theme finder finds the repetition, and linkage to the framing chords is performed for only one of the patterns, the other or others being assigned identically.

**Resolving ambiguities using Prolog.** We still use the same knowledge base as before (interval between notes and saturated chord indicators), but we separate information gathering from the inference process. Notes are still scanned to compute intervals with contiguous notes to detect potential continuities, but no decision about voice assignment is made immediately. Chains of logical continuous lines are formed using threshold values, but at this point we decide only that a chain of notes should belong to the same voice without trying to find out to which voice. Once all notes have been scanned and assigned to independent chains, we try to fit the chains together comparing their endpoints to each other or to the notes of the framing saturated chords.

We use production rules to take decisions on voice assignment for an entire chain of notes. For instance, we check if both ends of a chain agree



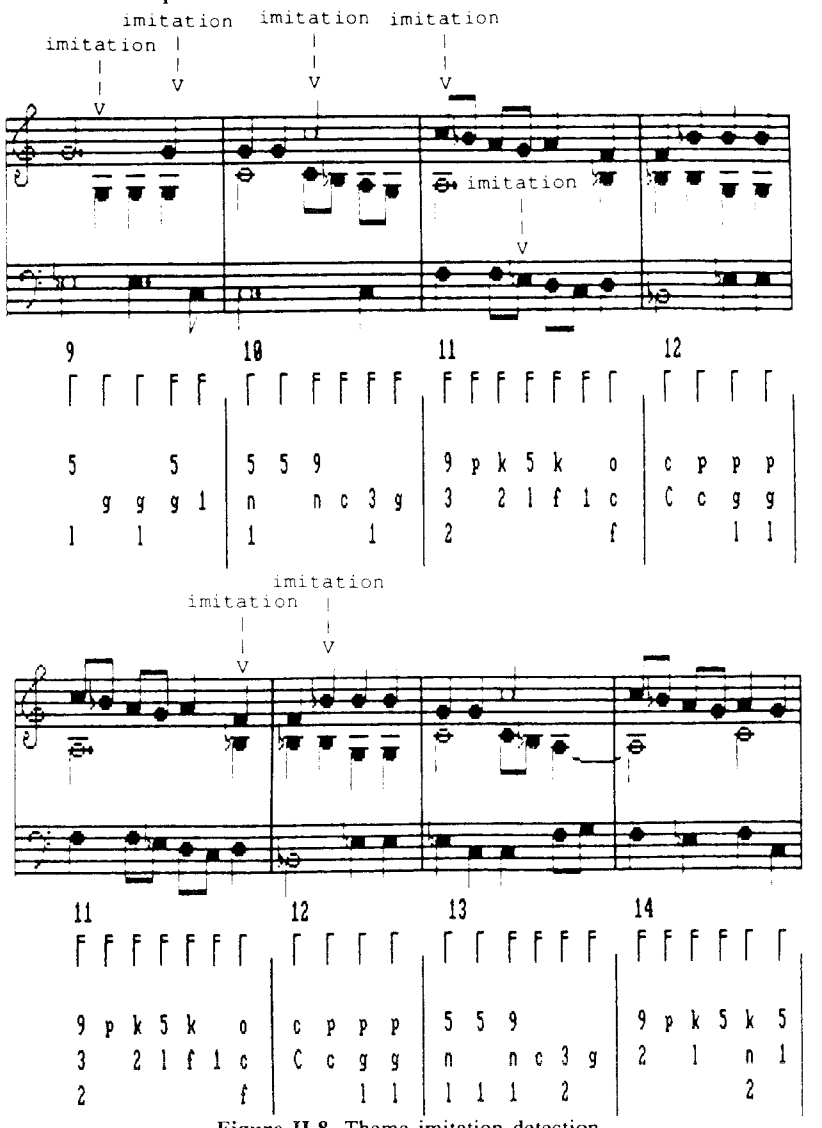


Figure II.8. Theme imitation detection

with respect to voice assignment. If they do not agree, we further delay the voice assignment and we look at another chain. This will result in all unambiguous cases being resolved first. The remaining cases will be derived residually by considering their relative position with respect to resolved cases. Consequently, we use an iteration process that sifts through any remaining

unassigned chains to resolve them. The production rules used are determined by an ongoing process of trial and error, where usually an error leads to the generation of a new production rule.

Here again Prolog's internal database is of great help since there are no complex pointer systems to maintain as in traditional programming languages. Instead facts are asserted or retracted to assign a property to, or remove a property from, a chain of notes. Given a representation of a note `note(Height,Location)` (the note fact is actually more complex than this) we can relate this to a chain by asserting `chain(Height, Location,Chain_number)` and the chain to a voice by asserting `voice(Chain_number,Voice_number)`.

In Figure II.9 the top part represents results achieved with the one-pass algorithm. The melodic fragment F E♭ D has been assigned to the wrong voice due to the presence of an ambiguity, resolved on the basis of the proximity of the note D to the D in the first framing saturated chord. With the new method, resulting in the bottom part of Figure II.9, the scanner determines six consistent chains using a threshold value of one tone maximum. The fitting starts with chains 1, 4 and 6 being unambiguously connected. At this point both end points of this meta-chain can be unambiguously connected to the notes belonging to the second voice of the saturated chords of bar 36 and 38. Chains 2 and 5 can be fitted together, and since its notes lie above chain 6, this implies that this chain belongs to the first voice. Chain 3 remains isolated and is then compared to the saturated chords for resolution.

**Conclusion on the second method.** The new method using Prolog and artificial intelligence concepts produced far better results than the one-pass approach implemented in a traditional computer language. However, further research is required to find solutions using knowledge other than continuity, such as harmonic and contrapuntal rules for instrumental music. Also, at present the entire system relies too heavily on the "saturated chords" as indicators of correct musical structure. During the development of the second method we discovered that solutions to restructuring problems can be found outside the natural frame of saturated chords. Separating ornaments from melodic lines is one of the possible avenues.

The very interesting point we made through this study was to go beyond traditional pattern recognition theory, where one has to find a clearly defined object that is hidden among a collection of objects. In music, most of the time we do not know what to look for because we are dealing with the creativity of



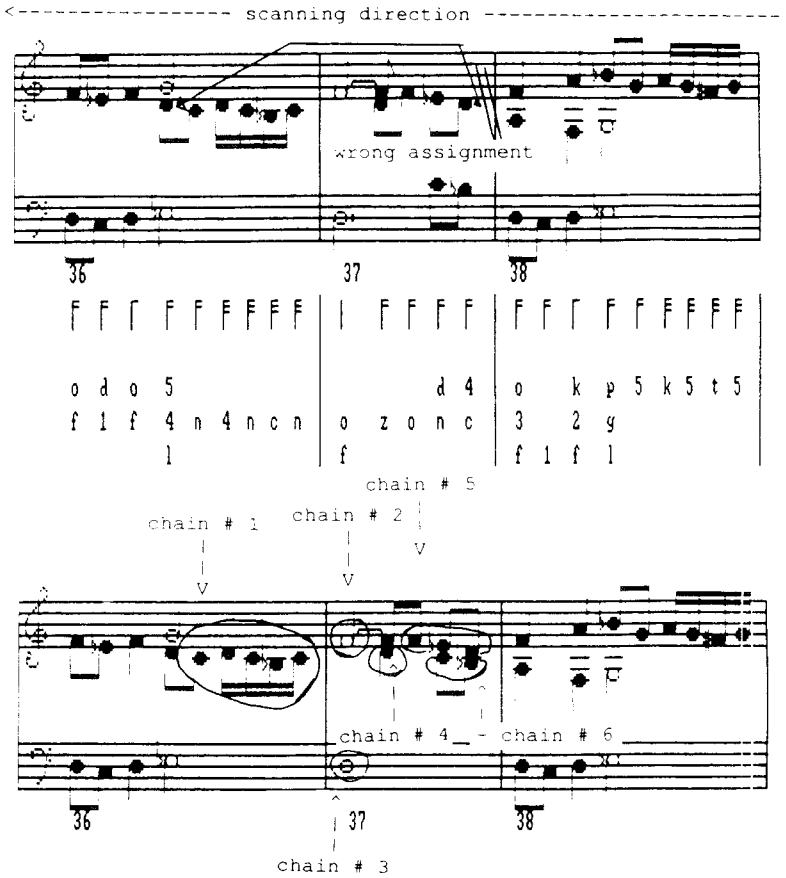


**Figure II.9.** Comparison of one-pass (top) and chain-fitting (bottom) processes

the composer. In music analysis, the task consists in finding the internal logics that a composer has used to create new "objects". Our closing remark is that if so far music is benefiting from other sciences including computer science, this seems to be the time where music analysis is producing benefits for other non musical sciences such as cognitive science.

## Future work

Although it uses artificial intelligence techniques, the current system has one major disadvantage: it is hard coded. This prevents generalisation. For example, it is likely that the techniques used here would not work for Japanese Koto tablatures. The next step would be to create a knowledge-

based system where a musicologist could teach the system how to behave with Japanese Kotos. This involves describing the knowledge and defining dynamically the production rules to be applied to the knowledge without having to program. The general philosophy in this methodology could be used for various other applications in restitution and publication of ancient documents.

## ASSOCIATED LITERATURE

Binkley, Th.E. Electronic Processing of Musical Materials. In Harald Heckmann (Ed.) *Electronische Datenverarbeitung in der Musikwissenschaft*. Regensburg: Bosse, 1967, 1–20.

Charnassé, H. La transcription automatique des tablatures, un aspect de la recherche méthodologique. In D. Heartz & B. Wade (Eds.) *Report of the Twelfth Congress of the International Musicological Society*, Kassel: Bärenreiter, 1981, 330–336.

Charnassé, H. Musicologie et informatique. L'analyse d'une notation instrumentale: la tablature de luth allemande. Report of the *Congrès International Informatique et Sciences Humaines*, Liege: LASLA, 1983, 177–194.

Charnassé, H. *Informatique et Musique, Session Musicologique de l'International Computer Music Conference* (IRCAM, Paris, 1984). Ivry-sur-Seine: ELMERATTO, 1988.

Charnassé, H. & Dubon, H. L'emploi de l'informatique pour la transcription et l'édition des textes musicaux. Report of the conference *Methodes Quantitatives et Informatiques dans l'Etude des Textes* (Nice, 1986). Genève-Paris: Slatkine-Champion, 1986, 187–194.

Charnassé, H. & Ducasse, H. (Eds.) Hans Gerle, *Tabulatur auff die Laudten*, Nuremberg, 1533. Automatic transcription by the ERATTO/CNRS team. Paris: Société Francaise de Musicologie, 1975–1978 (5 vols.).

Charnassé H. and Ducasse H. (Eds.) Hans Gerle, *Tabulatur auff die Laudten*, Nuremberg, 1552. Automatic transcription by the ERATTO/CNRS team. Ivry-sur-Seine: ELMERATTO, 1978 (4 vols.).

Charnassé, H. & Stepien, B. Automatic Transcription of Sixteenth Century Musical Notations. *Computers and the Humanities*, 1986, *20*, 179–190.

Hultberg, W.E. Development of Computer Programs Designed to Transcribe Tablature to Standard Notation. In H.B. Lincoln (Ed.) *The Computer and Music*. Ithaca: Cornell University Press, 1970 (2nd. ed. 1972), 288–292.

Hultberg, W.E. Computerized Tablature Transcription Processes. In D. Heartz & B. Wade (Eds.) *Report of the Twelfth Congress of the International Musicological Society*, Kassel: Bärenreiter, 1981, 336–339.

Hultberg, W.E. Computer-Based Processes for Tablature Transcription: Input Language Applications and Development; Analytical Aspects. In C.Roads (Ed.) *Proceedings of the 1978 International Computer Music Conference*, Computer Music Journal, 1979, 689–719.

Hultberg, W.E. Computer-Based Processes for Tablature Transcription. *RCA Review*, University of Texas, San Antonio, January 1979.

# COMPUTATIONAL THEORIES OF MUSIC

## THEORETICAL AND APPLICATIVE ISSUES

*Lelio Camilleri*

### Abstract

This paper considers approaches to the representation of musical knowledge which share the computational paradigm but have different objectives. The discussion focusses on studies concerning music analysis, music theory and musical cognition, because I believe the computational paradigm allows these to be viewed as aspects of an overall theory. The studies are reviewed in terms of their contributions to the development of a computational theory of musical cognition, and also of their applicative power to produce analytical tools. Attention is devoted to different levels of description of musical behaviour, and to the use of the computer metaphor to explain processes underlying the perception and cognition of music.

## INTRODUCTION

Computer modelling in music can be seen as a new approach that can unify various methodological aspects which have arisen through the conjunction of music theory with other disciplines in science and the humanities. I would like to stress the significance of new methodologies, because their existence implies a rethinking and indeed a redefinition of music theory, perhaps not entirely new, but certainly more explicit than before. Since 1950, new

*Computer Representations and Models in Music*
*ISBN 0-12-473545-2*