Vol.2018-MUS-118 No.11
Vol.2018-SLP-120 No.11
2018/2/20

自動運転車のためのリアルタイム作曲システムに向けて

長嶋洋一[†1]

概要：世界中で開発が進められている自動運転車は安全走行のため多種の搭載センサを外界に向けており、GPSを含むこのセンシング情報を生かして周囲の状況に反応する「リアルタイム作曲システム」として、運転の必要がない全ての搭乗者に快適で音楽著作権の不要なBGMをライヴ生成する自動作曲手法が求められている。英国Forkswagenが2013年に発表した"Play the Road"は運転手目線のシステムでありやや目的が異なるが、その音楽スタイルの検討は重要な参考資料となった。ライブ生成の要素を除外するとしても、古今東西の音楽データを収集し深層学習させたAIが良質な音楽を自動生成するためには、教師データとして膨大な音楽嗜好感性ビッグデータを適用すればよいが、いまだに成功例が報告されていない。筆者は2006年にIPA「未踏」に採択され発表した"FMC3(Free Music Clip for Creative Common)"において、音楽的ヒューリスティクスを重視した考え方を提案したが、豊田中央研究所に共同研究を依頼され開発した2016年版の車載用リアルタイムBGM生成システムにおいて、この原則を発展させた。自動運転車が生成するセンサ情報と音楽音響的情報との関係を整理した上で、本稿ではこのリアルタイムBGM生成システムの試作に関して、(1)BGM進行のための音楽的な基本原理、(2)ループの繰り返しとリズム/ビートのスタイル、(3)センサ/マッピング/音楽生成のブロック分割、(4)音楽要素パラメータへの確率統計的な重み付け、(5)調性とコードとスケールの構成、(6)試作と実験の模様、などについて詳細に報告する。

Towards a "realtime musical composing" system for autonomous vehicle

YOICHI NAGASHIMA[†1]

1. はじめに

コンピュータの歴史とほぼ同じ長さを持つ**コンピュータ音楽**の歴史は、コンピュータによる**自動作曲**というテーマを含めて始まった。単なるサイコロ(ランダム)による音楽生成では面白くないので、音楽理論的ルールを考慮して統計確率処理し、過去の名作曲家の音楽作品データベースを基に特徴抽出した感性情報処理フィルタを施して音楽的個性を反映した学習を行う、というようなアイデアは既に数十年の歴史がある。本稿ではまず次節でこれらの古典をサーベイしつつ、国内の学会で延々と頻発する音楽的でない不毛/皮相的な自動作曲システム研究に警鐘を鳴らす。

レコードが消滅しCDが衰退しダウンロード聴取よりもストリーミング音楽を聞き流す時代となり、職業音楽家(作曲家/演奏家)は絶滅危惧業種に挙げられるようになった。聞き流すだけの低価格/無料のBGMであれば、古今東西の音楽データを収集し深層学習させたGoogleあたりのAIが良質な音楽を自動生成してくれる筈だが、囲碁や将棋と違って明確なルールのない音楽領域では教師データとなる膨大な嗜好感性ビッグデータが分散するのか、いまだに成功例は報告されていない。

ライブComputer Musicの世界ではここ30年ほど、音楽演奏の場で音楽情報をリアルタイム生成する「**アルゴリズム作曲**」というスタイルの自動作曲が探求されてきた。センサや新楽器[1]のライブ情報を音楽情報に**マッピング**したり、センサや外部環境のライブ情報によって音楽音響をリアルタイムに変容させたり、さらに過激にはリアルタイム作曲のプログラム自身を音楽ライブ情報によって書き換え/上書きしていく(自爆テロのような)アプローチもある。

21世紀になって世界中で開発が進められている**自動運転車**は多種の搭載センサを外界に向けており、GPSを含むこのセンシング情報を生かして周囲の状況に反応する「リアルタイム作曲システム」として、運転の必要がない全ての搭乗者に快適で音楽著作権が不要なBGMをライヴ生成する自動作曲手法が求められるようになった。英国Forkswagenが2013年に発表した"Play the Road"[2]はその先駆けとして運転状況(加速減速/ステアリング操作)に応じて音楽演奏情報を変化させたが、どちらかといえば運転手目線のシステムでありやや目的が異なる。

筆者は2006年にIPA**未踏**に採択され発表した"**FMC3**(Free Music Clip for Creative Common)"[3]において、**音楽的ヒューリスティクス**を重視した考え方の自動作曲システムを提案した。これは当時ネットで流行したFlashムービー等コンテンツのBGMを厖大な組み合わせから自動生成するもので、4小節ループ音楽のコード進行を駆動する音楽的ルールを基礎として、53種類のコード進行から選び刻々とランダム転調するアルゴリズムで標準MIDIファイル形式の音楽演奏データを自動生成した。

理由/原理が分からなくても結果が良好であれば構わないという昨今のAIブームに背を向けて、筆者はこの"FMC3"の音楽的ヒューリスティクスを重視した考え方を、**豊田中央研究所**に共同研究を依頼され開発した2016年版の車載用リアルタイムBGM生成システムにおいて発展させた。自動運

[†1] 静岡文化芸術大学
Shizuoka University of Art and Culture

Vol.2018-MUS-118 No.11
Vol.2018-SLP-120 No.11
2018/2/20

転車が生成するセンサ情報と音楽音響的情報との関係の全体については2018年3月の電子情報通信学会大会[4]で報告の予定なので、本稿ではこのリアルタイムBGM生成システムの試作に関して、BGM進行のための音楽的な基本原理、ループの繰り返しとリズム/ビートのスタイル、センサ/マッピング/音楽生成のブロック分割、音楽要素パラメータへの確率統計的な重み付け、調性とコードとスケールの構成、試作と実験の模様、などについて報告する。

2. 「自動作曲」のサーベイ

ここでは筆者の手元にある「コンピュータと音楽」関係の文献のうち1972年から2001年までに国内出版された9冊(図1)などから、関連する事項を簡単に紹介する。筆者が前節で「国内の学会で延々と頻発する音楽的でない不毛/皮相的な自動作曲システム研究」と呼んだのは、サーベイの体をとっているものの直近5〜10年程度の学会発表(それらも同様にほぼサーベイレス)をReferencesに並べるだけで、中身は実質的にははるか遠い過去に行われてきた研究と同じ(というより安直な分だけより低レベルな)システムを厚顔無恥に発表しているという、音楽の歴史に対してあまりに無頓着な研究室(指導教官)の姿勢に対してである。

図 1 サーベイ参考文献
Figure 1 Surveil References.

2.1. コンピューターと音楽

1972年の「**コンピューターと音楽**」[5]では、その後の文献で扱いが漸減している「作曲」という章が全21章のうちの4章を占めていた。20世紀に出現したコンピュータと対峙する音楽家の哲学的な考察は圧巻であり、第2章「音楽的想念からコンピューターへ　そしてふたたび音楽的想念へ」、第3章「コンピューター作曲の倫理学と美学」は、21世紀の今でも作曲を志す者にとって必読である。第4章「コンピューターで作曲された音楽---歴史的な展望」は世界各地の拠点で創造されてきた事例の厖大なサーベイとなっており、第5章「MUSPEC」というアルゴリズム作曲のために構築提案されたプログラム(1966)は、とても50年前とは思えない音楽的/網羅的に精緻に設計されたものだった。これと比べたら赤面するしかない稚拙な設計のシステム発表に、ここ30年で何度、触れてきたことだろう。

2.2. コンピュータと音楽

情報処理学会の正式な研究会である音楽情報科学研究会(SIGMUS)へと移行した1993年より以前、任意団体「音楽情報科学研究会(JMACS)」は8年間で計41回の研究会を開催してきた[6]。その中心メンバーが共立出版「bit別冊　**コンピュータと音楽**」[7]を出したのは1987年であり、この中では「シミュレーションとしての作曲　クセナキスの数学的技法を中心に」(神前尚生)という詳細なサーベイ、さらに「音楽のおもちゃプログラム」(坪井邦明)ではそれぞれ10行ほどのBASICプログラムによって「擬似わらべ唄」「擬似民謡」「1/fゆらぎ音楽」「古典西洋音楽風」などの生成アルゴリズムが紹介された。

2.3. 計算機と音楽

その翌年の1988年には情報処理学会誌のほぼ全てを「**計算機と音楽**」という特集記事が占めたが、解説記事「計算機による作曲と編曲」の中に、現在にまで通じる悲劇の萌芽を発見した。『筆者の研究室で得られたこの方法の自動作曲による演歌の旋律は、演歌の素人には、既存の演歌に聴こえた』という工学領域の記事における言述が、音楽においてどれだけ不毛なものであるのかに気付かないで研究を進めること自体、筆者が垣間見るところ「悲劇」なのである。この特集記事の最後を飾る「座談会」には、冨田勲・伊勢正三などの有名人も登場し、この時期の研究会にはプロミュージシャンも来ていた良き時代であった。

2.4. これがコンピュータミュージックだ!

その翌年の1989年には、コンピュータ科学の普及に乗った雑誌「Computer　Today」誌(2003年に休刊という名の廃刊)を「**これがコンピュータミュージックだ!**」[9]という特集が占めた。岩竹徹・上原和夫・湯浅譲二・藤枝守・小谷義行など錚々たる執筆者が並び、「MITのハイパーインストルメント」「ハミングされたメロディの自動編曲」「ゲームミュージック」など当時の話題を網羅した素晴らしい特集だった。しかし「作曲アルゴリズムProfile」(1984年頃)を見ると、自動作曲のアプローチは結局のところきわめて古典的であり、この世界の奥深さ(筆者もまったく手を出さず)を再確認できた。

2.5. 音楽情報処理の技術的基盤

SIGMUSのスタートした1993年は、それまで北米と欧州で開催されてきた国際会議ICMCが初めてそれ以外に出たという記念すべきICMC1993(早稲田大)の年であり、ほぼ全員がICMC実行委員会メンバーとなった国内の音楽情報科学研究者が「**音楽情報処理の技術的基盤**」[10]という科研費報告書をまとめた(筆者も執筆に参加)。この中で「コンピュータ音楽の制作環境」(嶋津武仁)は「歴史的概要」として海外の事例を詳細にサーベイするとともに「わが国における展開」として「日本の電子音楽」「日本のコンピュータ音楽」をコンパクトながら総括してICMC1993に繋いでいて、この分野のサーベイとしては必須の重みがある。

2.6. 音楽情報処理

ICMC1993の成功を受けて、情報処理学会誌では1994年に「**音楽情報処理**」[11]という(小)特集が出た(筆者も執筆に参加)。感性情報処理、音源分離、自動伴奏、などのホットな話題の記事が集まったが、ここでは「音楽認知への計算的アプローチとその課題」(平賀譲)に注目したい。著者が冒頭に「作曲・演奏といった生成面は扱わない」と述べているものの、ここで整理総括している議論(とその限界)は、アルゴリズム作曲システムを構築する研究者にとって必見の視点であり、ぜひともこの領域をサーベイしてから

取り組んでいただきたいのである。作曲という脳内プロセスが過去の事例を参照しつつ創作(つまり脳内にはComposerだけでなくPlayerとListenerもいるのが作曲家)試行錯誤する以上、音楽認知こそ「良い音楽の作曲」への前線基地であると思う。

2.7. コンピュータと音楽の世界

任意団体の音楽情報科学研究会が出したbit別冊から10年後の続編、を合言葉に若手音楽情報科学研究者が集って執筆(出版は1年遅れの1998年)したのが、その後しばらくの間は国内のこの分野でバイブルとされてきた分厚い「bit別冊 **コンピュータと音楽の世界** 基礎からフロンティアまで」[12](図2)であり、筆者は付録CDROM編集・記事4編執筆・ホテル缶詰の最終校正を担当した。しかしこの電話帳のような文献には「自動作曲」が見当たらない。1990年代に入ってコンピュータの性能が飛躍的に向上したものの、その機能はコンピュータ音楽における「リアルタイム性」の追求に向かい、音楽演奏情報や音楽音響信号をリアルタイム生成するようなアルゴリズムが走るシステム(MaxやSuperColliderやProcessing等のプラットフォーム)が普及することで、古典的な音楽情報科学がテーマとした(非実時間的)自動作曲という意味ではほぼ消滅したからだろう。しかし音楽的/情報科学的な課題が解決された訳ではなく、根本的課題は基本的に残ったままなのである。

図 2 コンピュータと音楽の世界
Figure 2 Computer & Music.

2.8. コンピュータサウンドの世界

筆者は上記bit別冊の執筆とほぼ同時期にCQ出版から「**コンピュータサウンドの世界**」[13][14](図3)を出し、その続編として上級編「**作るサウンドエレクトロニクス**」[15]も執筆しWeb発表した。これらは音源/システムの部分に重点があるものの、コンピュータが強力になった時代を反映して、「第5章 コンピュータミュージック」「第6章 センサとメディアアート」の部分では、期せずしてリアルタイム作曲/アルゴリズム作曲について言及していた。自分がComputer Musicの作曲家として活動していたのだから当然であるが、根底の目線は作曲にあったのである。

図 3 コンピュータサウンドの世界
Figure 3 Computer & Sound.

2.9. コンピュータ音楽 歴史・テクノロジー・アート

音楽情報科学分野の研究者であれば絶対に必読となった新バイブルは、電話帳2冊分を超える1054ページの「**コンピュータ音楽** 歴史・テクノロジー・アート」[16]である。MITで長らくComputer Music Journal編集長を務めたCurtis Roads氏の厖大な集大成の編訳に世界各地の研究者がチャレンジしたが、オリジナルの英語以外でこの文献を完全に翻訳出版したのは日本チームだけなのだという。「第18章 **アルゴリズム作曲システム**」「第19章 **アルゴリズム作曲システムの表現と技法**」で計90ページにもなり、この領域の完全な歴史的サーベイからほぼ最近に繋がる先端技法までが、まさに網羅的に紹介解説されている。つい先日の某学会でも、ある若手研究者のショボい自動作曲システムの発表に触れてこの文献を紹介したところだが、彼はおおいに刺激を受けて猛勉強しているという。このような文献を入手できるというのは羨ましい時代でもある。

2.10. 音楽情報科学の世界

筆者は河合楽器を退社独立してフリーだった時代、電波新聞社「マイコン」別冊「Computer Music Magazine」誌に「**音楽情報科学の世界**」として、以下のような内容で12ヶ月にわたる連載記事を執筆した(1996-1997)[17]。

1. **イントロダクション**
2. **インターネットとコンピュータ・ミュージック**
3. **コンピュータに音楽を演奏させる**
4. **コンピュータに音楽を聴かせる**
5. **電子楽器とコンピュータ・ミュージック**
6. **コンピュータとセッション演奏する**
7. **コンピュータに音楽を創造させる**
8. **コンピュータに歌わせる**
9. **マルチメディアとコンピュータ・ミュージック**
10. **人間とコンピュータ・ミュージック**
11. **コンピュータ・ミュージックの先端状況**
12. **これからのコンピュータ・ミュージック**

図は無いものの(この雑誌も1999年に廃刊)文字原稿は[17]にあるので参照可能である。この中の「**コンピュータに音楽を創造させる**」というのが自動作曲と関連した部分である。**ニューラルネット**や**カオス**にハマッていた筆者が20年以上も昔に書いた原稿ではあるが、いまだこの内容は現役であるのが、このテーマの奥深さを物語っている。

3. 音楽的な予備知識

本稿は次節から「**自動運転車のためのリアルタイム自動作曲システム**」について報告するが、その内容を完全にでなくても理解するためには、**最低限の音楽的予備知識**が必要である。幸いにも情報処理学会研究会予稿は電子化により**ページ数の上限が無くなった**ので、過去の原稿では割愛せざるを得なかった基本的な部分についてここで整理しておく。これは今後「国内の学会で延々と頻発する音楽的でない不毛な音楽情報科学研究」の発表(冒頭で「私は音楽については素人ですが‥」などと言い訳しつつ発表する音楽情報科学研究がどれほど空虚なものかを自覚すべきであろう)をなるべく未然に防ぎたい、という筆者の切実な希望が動機である。この領域で研究する者であれば本節の各項目ぐらいは知っている、という最低線まで勉強した上で本命の研究に取り組んでほしい。

3.1. MIDI

コンピュータ音楽においてはさまざまな階層の音楽情報表現形態があるが、本稿で次節から紹介する自動作曲においては**MIDI**情報[18]を採用した。1983年という遠い昔に規定されたMIDIがいまだ現役というのはある意味で標準化の歴史における驚異だが、過去にICMC/NIMEなどの場で色々な「**拡張MIDI**」が提案されてもまったく普及しないのも事実である。詳細は筆者の解説[18]やWikipediaを参照されたい。MIDIの基本概念である音楽**演奏**情報、すなわち基本的にはノートイベント(鍵盤のONとOFF)と相互の相対時間(**デルタタイム**)のシーケンスで音楽(楽曲)情報を記述する(サウンドであれば1分間の無音は約10MBだがMIDIなら約10bytesで、MIDI情報から楽音[音響]に変換する**音源**が別途に必要)というコンセプトの理解が重要である。

3.2. Max

MIDIシーケンス(標準MIDIファイル)として**再生**される音楽情報の作曲であれば各種の「打ち込み」音楽ソフト(DTM)で十分であるが、リアルタイムにMIDI音楽情報を生成するアルゴリズム作曲となると、**Max**のようなプラットフォームがほぼ必須となる。思えば、筆者が1989年に作曲家・中村滋延氏の自宅スタジオでIRCAMのXavier Chabot氏から「来年発表するけど」と言いながら見せてもらったフロッピー1枚に入るMacintoshのソフトが、このMaxであった[19]。そこから約30年、現在は**Max7**となっているが、筆者は「bit別冊**コンピュータと音楽の世界**」[12]では「**アルゴリズム作曲**」[20]としてMaxを紹介し、もう25年以上も音楽情報科学の相棒として活用している[21]。過去の"**FMC3**"[3]もMaxで開発し、もちろん本研究でも活用している。豊田中央研究所の研究者は他の処理系を使っていてMaxを知らなかったが、筆者との共同研究で**強力なプロトタイピングプラットフォームとしてのMax**を知ったことも大きな収穫だったそうである。

3.3. 音律

本研究においては**12等分平均律の音体系に限定**している。‥こう書かれてピンと来る読者は次項にスキップしても結構である。音律については筆者が1993年から公開している、完全な書き下ろしの解説[22]を参照されたい。本研究の音楽的基礎となっている**和音の協和**については、各種の**音律**の理論的基礎となっている**協和感覚**とは別次元である、という前提の理解が重要である。

例えばバッハの目指した「**調性の性格**」(Well-Tempered音律では長3和音それぞれの協和度[緊張度]が異なることを活用して作曲)というような視点は本研究においては捨象されている。つまり、例えば調的中心(Tonal Center)がC(ハ長調)であるように調律されたKirnberger音律やWerckmeister音律では、ハ長調の主要和音は12等分平均律よりもはるかに純正律に近い良好な響きを持ち、これが転調を繰り返してハ長調から遠く離れていくと次第に主要和音の協和度が低下して、耳のいい人であれば最初の純正な響きに比べて濁りが増えてくるために、どことなく落ちつかない気分あるいは緊張感の高まりを感じる‥というような音楽美学的な議論は本研究においては触れない。

3.4. 楽典(音程と協和、調とコード)

本稿はアルゴリズム作曲について音楽的原理から解説する必要があるので、理解するためには**最低限の音楽的知識**が必須となる。世の中には各種のレベルの**楽典**が存在しており、筆者も希望学生に対して行う「**音楽理論特訓集中講座**」[23]の冒頭では、楽譜の読み方(コード解説の理解のためには楽譜が読める必要あり)から始まってコードネームの定義などを叩き込んだ上で、本研究にも繋がる**コード進行の原理/理論**を解説している。書店には各種の楽典/書籍があり、ネット上にも音楽愛好家の多種の**楽典サイト**がある(ただし盲目的な過信は禁物である)。

注意点として、楽典など音楽理論には時代/レベル/地域(国)による「**方言**」のような差異が多く、一見すると相互に矛盾した記述が多数存在する、という事実も理解すべきである。例えば入門的西洋音楽では不協和音とされるコードが、Jazz/Popsの音楽理論では気持ちいい/カッコいい響き、と解説されるのは何の矛盾もない。ポリフォニー音楽において同時に鳴る音程はオクターブと純正完全5度/完全4度のみ許された中世ヨーロッパでは純正長3度音程は不協和(邪悪な音程)だったのが、イギリスを皮切りにホモフォニックに純正長3和音を**協和**として活用したところから古典派・バロック・ルネサンス音楽の時代が花開いた‥というような音楽美学的な議論も本稿においては深入りしない。

本稿は全体としては音律の世界の主役「**純正協和**」は対象外となるので、まずここで**音程**と**協和**について簡単に補足する。**音程/協和**に関する考察において、全ての音はオクターブ単位で上下させても音楽的な意味が変わらない(mod 12)ので、適宜、1オクターブ内に収まるように上下移動させる。また「足しあわせて1オクターブとなる」2種類の音程は**転回音程**といい、和声的な意味は相補的(等価)である。**音程の協和度**とは、初級の楽典にもあるように、振動数比が簡単な整数比になるほど協和するというギリシャ時代(ピタゴラス学派)からの物理学的真理を基礎としている。もっとも協和する音程は、**完全0度**(同じ高さ,1:1)と**完全8度**(オクターブ,1:2)である。

その次に協和するのは**完全5度**(2:3)と**完全4度**(3:4)であり、この2つは**転回音程**として相補的(完全5度上は完全4度下と等価)なので別種の音程として扱わず完全5度のみを代表させる。完全5度は12等分平均律(700セント)と純正音程(2:3,701.955セント)との誤差がきわめて小さいことから音楽理論において最も重要な役割りを持つので特別にp5(perfect 5th)と呼ばれる。基準音から連続してp5上行を繰り返し適宜1オクターブ内に収まるように上下させると、p5を生成元として平均律12音からなる加法群を生成し**5度円**が出来る(円を逆回りすればp5下行、あるいは完全4度上行)。**半音**(100セント)は平均律の定義から平均律12音の加法群を生成するもう一つの生成元であるが、等価な転回音程(完全4度/長7度)を除くと加法群を生成できる音程は他に存在しない事からもp5の重要性を確認できる。

次に協和する純正音程は、長3和音(major)と短3和音(minor)に登場する**長3度**(4:5)と**短3度**(5:6)の音程である(長3度の転回音程は短6度、短3度の転回音程は長6度であ

り、相補的/等価なので以降は省略)。オクターブやp5が堅牢な枠組みとして**無色**なのに対して、これらの音程は**音楽的な色合い**を持つ。純正な長3和音(major)は4:5:6という振動数比、純正な短3和音(minor)は10:12:15という振動数比である。この1度と5度のp5が上述のように平均律でもほぼ純正音程に近いのに対して、平均律長3度(半音4つ,400セント)と純正長3度(4:5,386.31セント)との差は**ヴォルフ**(唸り)を生むほど離れている。生まれた時から電子楽器の正確な12等分平均律の和音に包まれる現代人は、この**平均律長3度の唸り**をある種のコーラス効果のような**美**として無意識に享受して育つので、まったく唸らない純正な長3和音の響きを聞くと空虚で面白くない印象を持つ。つまり**人類の和声協和感覚は退化してきている**(北欧の少年少女合唱団がノンビブラートの純正なハーモニーでメシアンあたりを演奏するのを、YouTubeのスピーカ再生でなく実際に空気振動の体感音響として教会やホールで聴いてみれば間違いなく人生観が変わる、と筆者は約束する)。

振動数比を通分すると、純正長3和音は20:25:30、純正短3和音は20:24:30となって、長3度(半音4つ)と短3度(半音3つ)との隔たり、すなわち純正な半音(半音階的小半音,**小クロマ**)の振動数比は24:25 = 96:100(4%)となる。半音2つ分が**全音**(200セント)であり倍の8%となるが、振動数比が簡単な整数比となる[22]という要請からは2種類の全音として、**大全音**(9:10,10%)と**小全音**(8:9,11.1%)が規定されている。電子音響音楽を作曲公演してきた筆者の経験則としては音響データの再生速度を10%上げたり下げたりすれば「ほぼ全音」と聴取できる。これらの**半音**や**全音**、あるいは転回音程である**短7度**(半音10個)や**長7度**(半音11個)は、ごく初歩的/古典的な和声理論においては不協和音程であるとされたが、現代のJazz/Popsあたりの和声においては**良い不協和**(**テンション**)として音楽的な位置付けが大きく向上している。

ここまで、転回音程の小さな方で代表させると「半音100cent・全音200cent・短3度300cent・長3度400cent・完全4度500cent」と登場して最後に残った音程はあと1つ、**半音6つ**(**3全音**)の**増4度/減5度**音程(600cent)である。楽譜にすると見た目は4度ないし5度であるが、音程を数えてみると半音6個分、あるいは全音3個分(→ここから**トライトーン**と呼ばれる)、さらには短3度を2個分(→これがdimコードを生む)というこの音程は、鳴らして聞いてみれば判るがいつの時代にも**不協和音程**である。イロモノ/嫌われ者/脇役のようなこの音程が、本稿の第6節では**影の主役**として重要な意味を持つことは興味深い。

さて、本研究においては3和音をベースとする古典的な西洋音楽の枠組みでなく、基本的に4和音を主役とするJazz/Popsの音楽理論を基礎とする[23]ことに注意されたい。すなわち、例えば調的中心(Tonal Center)がC(ハ長調)である場合、**全音階**(ダイアトニックスケール)Diatonic Scaleの各音は「C,D,E,F,G,A,B」となり、それぞれを基音rootとするダイアトニックコードDiatonic Chord(4和音)は「CM7,Dm7,Em7,FM7,G7,Am7,Bm7b5」である。このうち基音と5度音の音程がp5でなくトライトーンである7の和音(Bm7b5)だけを不協和音として除外した6種の(主要)和音はそれぞれ、トニック(CM7,Am7)、ドミナント(Em7,G7)、サブドミナント(Dm7,FM7)の3種類の**機能**Functionを持つ(これ以上の音楽理論的考察は第6節)。

12音から特別な関係(下から半音単位で2,2,1,2,2,2,1という間隔で並ぶ7音)で選ばれたのが「C,D,E,F,G,A,B」(Key=C)というDiatonic Scaleであるが、これはどの**調**においても**5度円**において**隣接する7音**となっている。ちなみに5度円において**隣接する5音**を取り出すとPentatonic Scale(**5音音階**)になっていて、黒鍵だけで弾く「猫踏んじゃった」のように、どのようにデタラメに弾いても音楽的に聞きやすい妥当なメロディー(音組織)となり、鳴らす順序やリズムを工夫すると、それだけで簡単に演歌に聞こえたり民族音楽に聞こえたりする。

12等分平均律においてはどこに移調/転調しても音楽的には変わらないので、混乱を避けるために本稿では全て調的中心をC(ハ長調)としている。刻々と転調を繰り返した”FMC3”[3]との大きな違いとして、逆に、本システムのBGM生成では「**一切転調しない**」という新たな発想に到達したのが一つの「売り」である。

4. 自動運転車とセンサ

4.1. 自動運転の定義

自動運転車といっても世界的に色々なタイプがあるようだが、Wikipediaによれば、日本政府や米国運輸省道路交通安全局(NHTSA)では自動化のレベルを以下のように定義しているという。ちなみに筆者は今でも意固地にマニュアル大衆車に乗り続けているが、13年間乗ったDemioから2017年1月に同型Demioの新車に買い換えたところ、オートマでないのにアイドリングストップする事に驚いたり、カーナビも自動ブレーキもないのに勝手に近接レーダが人やクルマの近接に際して警告音を出す**レベル0**(警告だけで何もしない)仕様になっている事に驚いた。

レベル0 前方衝突警告まで

レベル1(運転支援) 安全運転支援システム(自動ブレーキ)

レベル2(部分自動運転) アダプティブクルーズコントロール(ステアリングアシスト)

レベル3(条件付自動運転) システムが要請したときはドライバーが対応

レベル4(高度自動運転) 特定の状況下(高速道路上)、加速/操舵/制動をシステムが行う

レベル5(完全自動運転) 無人運転

上の自動運転の定義を念頭に、図4のような色々な自動運転車のイメージ写真/図を眺めてみると、これまでもGPSカーナビやドライブレコーダーカメラ等のセンサ技術が集結していた自動車が、駆動の心臓部をエンジンからモーターに移行していくとともに、完全に**電子機器**に進化しつつある時代を実感できる。末端まで含めると重厚長大な巨大産業であった自動車産業は、3Dプリンタ技術でマンションを建てたりオリジナルCPU/OSを実現したり世界中にドローンを供給する某隣国の興隆しつつある巨大電子産業に飲み込まれてしまうのかどうか、目が離せないところである。

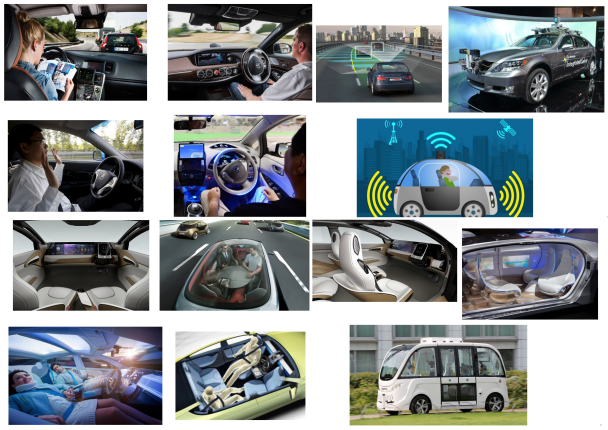

図 4 自動運転車のいろいろ
Figure 4 Autonomous Vehicles.

4.2. サウンドによるクルマ酔い低減の研究

本研究において、リアルタイムBGM作曲のパラメータとして「**車速**」「**ステアリング操作**」などの**運転情報**を利用しているが、これらのセンシングについては2010-2011年にパイオニアから依頼された受託研究「音響機器の人体に与える影響の研究開発」において筆者も実際に開発した。詳細は学会発表[24-26]に譲るが、瞑目しイアホンを装着した複数の被験者を乗せた自動車を多数のカーブが連続する**阿弥陀籤状**の道路で運転し、カーブに伴い身体にかかる左右方向の加速度と、聴取する音楽の**音像(PANPOT)**移動との関係について、手元のジョイスティックで**好き嫌い**をリアルタイム判定してもらうという実験を行った。

図 5 車速とステアリングのセンシング
Figure 5 Sensing of speed and steering.

当初の実験では車内の加速度センサ出力に対応して音像を移動させたが、「**ちょっと後に来る加速度の予感**」という先行情報生成のために車速とステアリング操作をセンシングする必要が生じて、図5のように車内ハーネスから手作業で分岐させた信号線によって車速センサ情報を取得し、さらにステアリングの軸にポテンションメータの回転軸を密着させて直接にハンドル回転をセンシングしてGainerに取り込みMaxに供給した。現代のカーエレクトロニクスは多数のセンサ情報や制御情報を全て**CAN**というネットワークで統制しているので、上記のような手法でなく、本研究においてはトヨタ実験車の**CAN**から取得したセンサデータをOSC/USBシリアルによってMaxシステムに取り込んだ。

4.3. センサ情報と自動作曲

最近の自動車では一般的な運転席からの視界のカメラ情報・CANデータ(運転状況センサ)・GPS情報などに加えて、衝突防止/近接物体検出などの目的で多種のレーダセンサが搭載されており、漠然としたカメラ画像による画像認識と異なり、クルマに直接に近接する物体や状況(運転者や同乗者が車窓の変化として感じる刹那的な状況変化)を得ることが出来る。そこで自動運転車のリアルタイム作曲システムとしては、**連続的に自動作曲演奏を続けるBGM**とともに、このレーダセンシング情報に対応した**刹那的なサウンド生成**を行って、BGMシステムのサウンドとともに運転者/同乗者にトータルのサウンド(音楽)を提供することが望まれる。この後者の刹那的に生成されるサウンドには、楽音と効果音という大きく2種類のタイプがある。

楽音とは音楽的枠組みに乗っているサウンドの事で、具体的には個々の音(note)がピッチ(音高/音階)を持つものであり、背景で鳴っているBGMの持っている**調的/和声的枠組み**に対して矛盾しないnoteを選ばないと、不快な不協和音となってしまう。例えばある瞬間の和音が**CM7**で、メロディー音も**CM7**に対して音楽的に選ばれたchord notesやtension notesが鳴るべき時に、例えば**F**音とか**F#**音のように音楽的に**avoid(禁止)**とされる音が一瞬でも鳴ることは避けなければならない。

効果音とは**楽音**以外のサウンドであり、特定の音楽的ピッチ(音階)を持たないパーカッション(打楽器)系の音、風や水やジェットなどのノイズ系の音、物体が衝突する打撃音や擦れる摩擦音、鐘や鈴の音や時計のカチカチ音、動物の鳴き声、人の話し声や叫び声や笑い声、ピッチを感じない重低音の衝突ノイズ、など多種の**騒音/自然音**が該当する。これらは基本的にBGMパートの持っている音楽的/調的枠組みとは異種なので、同時に鳴らされても人間は音楽的に不快な不協和音とは知覚しない。

4.4. 全体システムの構想

上述のように、各種のレーダセンサに対応して**楽音**に属するサウンド(ジングルのようなある種の短いフレーズ/メロディー)を刹那的に生成するためには、背景で鳴っているBGMの持っている調的枠組みを考慮する必要があるが、BGMには調的枠組み(ピッチ方向)とともに時間的構造もあり、本システムでは基本的に**4小節の周期でループ**している事を考慮する必要がある。つまり、レーダセンサから周囲の刹那的な変化情報が届いたその瞬間、BGMがいまどこ(何小節目/何拍目)を演奏しているのかを考慮する必要がある。ランダムに選ばれる4小節のコード進行の途中であれば、「今鳴っているコード」「次に鳴るコード」という和声的情報を考慮すればいいが、4小節のほぼ最後あたりの拍に来た場合、次の4小節の最初のコードがまだランダム選択されていない瞬間には、その決定を待つまでイベント検出状態を保持する必要があり、イベント検出の直後にすぐサウンド生成を始めてしまわないようにする必要がある。

そこで本研究から発展して現実的なリアルタイム作曲システムを実装に向けて開発していくためには、試作開発したBGM自動生成システムに加えて「リアルタイム・レーダセンシング」・「パラメータ・マッピング2」・「リアルタイム効果音生成」・「リアルタイム楽音生成」が有機的に結合した、図6のようなシステムを構築する必要があるという結論に達した。本稿で次節から「本システム」として紹介報告するのは、実際に試作開発したBGM自動生成システムの部分(図6の左端3ブロック)だけである。

Vol.2018-MUS-118 No.11
Vol.2018-SLP-120 No.11
2018/2/20

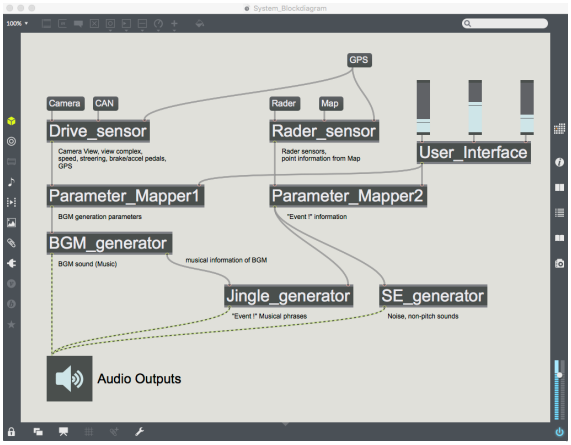

図 6 全体システムの構想
Figure 6 Total System Block Diagram.

5. リアルタイムBGM生成システムの概要

5.1. 開発と試験走行の切り分け

筆者が本システムを開発したのは静岡県浜松市の静岡文化芸術大学(主要部分は2016年の欧露ツアー[27]道中にプログラミング)であり、トヨタ実験車が本システムを搭載して市街地を走行したのは愛知県長久手市の豊田中央研究所の周辺である。筆者が実際に実験車に乗車したのは2回だけであり、それ以外は全てMax7パッチの送付交換と以下の手法によって開発と実験の同時進行を実現した。

図 7 メインBGM生成モジュール
Figure 7 Main BGM Generator Module.

最終的にサウンド出力する下流から遡っていくと、図6でBGM_ganaratorとあるのが図7の「自動作曲モジュールパッチAutoComposer020.maxpat」である。外見はとてもシンプルでコンパクトに見えるが、サブパッチを叩いて下部の階層構造に入っていくと、筆者の持つMaxテクニックの全てを極限まで満載したとんでもない密林となっている。このモジュールは起動するだけでdefaultの音楽生成パラメータ初期値に従って単調でなく快適に進行するBGMを刻々と自動演奏し続けるが、シーケンスデータの読み出しではなくリアルタイムにアルゴリズム作曲して生成したMIDI情報によって(パソコン内部の)MIDI音源を駆動する。

その上流、図6でParameter_Mapper1とあるのが「パラメータ変換用中間パッチparameter_convert010.maxpat」(図8)である。これは、上流のセンシング情報取得ブロック(後述)から刻々と送られてくるMaxの内部パラメータを受け取り、後段の自動BGM生成のAutoComposer020.maxpatに対して音楽的な変化をもたらすパラメータにマッピングする複数のマッピングパターンブロックを選択して、刻々とBGM生成パラメータを変更しつつ表示しているブロックである。これら2つのMaxパッチの連携動作が今回の試作開発システムのほぼ全てであるが、遠隔地である開発/実験拠点を結ぶためにもう1つのブロックが必要となった。

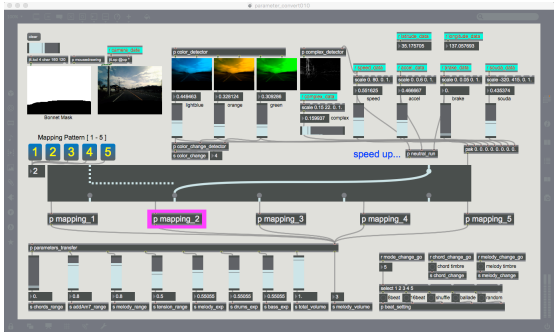

図 8 パラメータ変換用中間パッチ
Figure 8 Parameter Mapping Patch.

開発プロジェクトの冒頭、筆者は図6でDrive_sensorとある「各種センサ情報(カメラ画像を含む)を刻々とロギングしてカメラ画像をmovieファイルに、他センサ全データをplain textファイルに書き出す」という**ロガーパッチ**を豊田中研に開発提供した。このパッチの記録部分を除いたモジュールが**ライブセンシング・モジュール**として実験車内においてparameter_convert010.maxpatに所定のパラメータを送れば、それで後段2ブロック全体からなる筆者の開発したシステム全体の検証を行える。

そしてこのロガーパッチの動作形態を**記録**から**再生**に修正変更したパッチは、実験車とは離れて「記録されたデータを刻々と送り出す(実験車の走行のシミュレーション)」という**センサデータ再生モジュール**として機能することになり、これで現場にいなくても開発を進められる。図9はそのような「車載センサ情報再生モジュールパッチcar_data_player.maxpat」の一例であり、実際に実験路を走行してロギングしたデータファイルcar_camera.mov(車載カメラ情報を記録したmovieデータ)とcar_data.txt(車載センサ群情報を記録したデータ)を刻々と読み込んで、あたかも実際に実験車の車載**ライブセンシング・モジュール**がparameter_convert010.maxpatに送るのと同様にパラメータをライブ生成(再生)して、遠隔地での筆者のプログラミングを支援した。

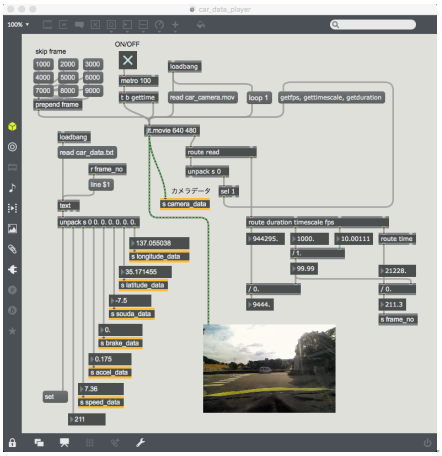

図 9 センサデータ再生モジュール
Figure 9 Sensor Data Playback Module.

自動作曲モジュールパッチAutoComposer020.maxpatとパラメータ変換用中間パッチparameter_convert010.maxpatとの関係をここで改めて整理確認する。パラメータ変換用中間パッチはcar_data_player.maxpatから運転記録データ/movieを再生して送られる情報、あるいは実車において送られてくるライブ情報/画像から、下流に位置する自動作曲モジュールが音楽生成において参照するパラメータを変化させたり対応関係のマッピングを設定する。すなわち、センサ情報への対応を変更する際に**自動作曲モジュールMax7パッチ自身は原則として改編する必要がない**というように切り分けた構成である。これにより、将来的に運転状況/カーナビ等から得られるセンシング情報が増えた場合にも、パラメータ変換用中間パッチでの関係性マッピングの部分を改良するだけで対応できる。

システム下流に位置する自動作曲モジュールMax7パッチAutoComposer020.maxpatの音楽生成アルゴリズムは、本稿で詳細に後述するように、音楽理論や積み上げられてきた音楽的ヒューリスティクスの塊である。一方、上流の運転状況に関する多種のセンサデータの取得/変換や、それらを音楽生成パラメータとしてマッピング/スケーリングするparameter_convert010.maxpatの部分は、原理的には音楽的ヒューリスティクスとは無縁の**数理的操作**となっている。深層学習AIなどを導入し多数の被験者の評価データを積み上げて改良されるのは後者の部分であり、将来的なシステム改良に際しても、この切り分けは重要である。

5.2. 起動時に用意されるデータ

本システムを稼働させるためには、上記3つのMaxパッチと、car_data_player.maxpatによってシミュレーションする場合には上記2つのデータファイルcar_camera.movとcar_data.txtが同じディレクトリに置かれる必要がある。そしてさらに本システム稼働のためにはあと2つ、codes.txt(コード進行用データ)とnotes.txt(メロディー用テンションノートデータ)という、まさに音楽的ヒューリスティクスの塊の真髄である2つのplainテキストファイルも同じディレクトリに置かれる必要がある。このうち前者のcodes.txtは図10のようなもので、”FMC3”[3]では**53種類**であったコード進行パターンが**152種類**に拡張されている。この内容は第7節で詳細に後述する。

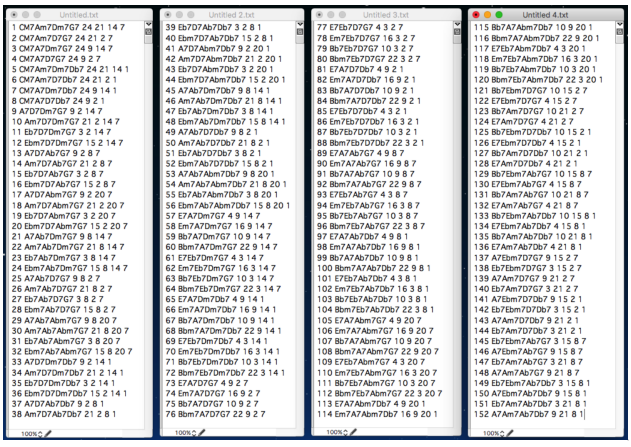

図 10 152種類のコード進行パターンデータ
Figure 10 codes.txt.

5.3. メインBGM生成モジュールパッチの概要

ここで本研究の心臓部である、図6のBGM_ganarator、図7の自動作曲モジュールパッチAutoComposer020.maxpatの中にある8個の下部階層のサブパッチの役割を本システムの音楽的枠組みと合わせて紹介する。

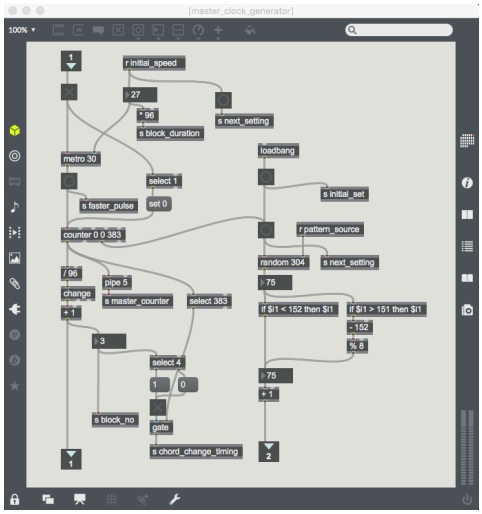

図 11 「master_clock_generator」サブパッチ
Figure 11 master_clock_generator.

図11の「master_clock_generator」はその名の通り、システム全体を駆動する4小節384クロックのマスタークロックを生成したり、小節番号に対応した自動作曲パラメータを送出するブロックである。注目すべき点として、内部的に「initial_speed」変数となっているテンポ設定値は、4種類のBGMスタイルごとに異なる初期値に設定されるものの、その後は運転状況パラメータによって変更されない、つまり本システムでは、**生成するBGMのテンポは変化しない**という音楽的仕様を定めた。

音楽演奏表現において、テンポの変化(**アゴーギク**)は、音量の変化(**デュナーミク**)や音色の変化(**コロリート**)と並んで重要な要素であるが、敢えて運転状況(加速/減速やカーブなど)に応じてテンポを動かさない事にした。クラシック音楽で微妙にテンポが変化するのは芸術的表現として至高の喜びであるが、基本的にJazz/Pops/Rock的に**インテンポ**(ある意味でdanceable)で進む本システムのBGM部分のテンポが、ちょっとした運転状況の変化にいちいち対応して速くなったり遅くなったりするのは、やってみれば判るがとても気持ち悪いのである。

英国Forkswagenが2013年に発表した”Play the Road”[2]においては、コードはたった2種類(Key=Cとすればそれ**CM7**と**FM7**とを交互にひたすら繰り返す)で進行し、テンポについては作曲を依頼されたミュージシャンは運転状況(加速減速/ステアリング操作)が変化しても生成する音楽のテンポを一定にしたまま、個々の楽器パートのフレーズの音数や拍の分割法やパート音量等だけを変化させていた。コードについては全く立場が異なるが、テンポについては、クルマが停止したらどうするか(音楽が止まってはブツ切れになる)、という単純な理由よりも深い音楽的考察において、筆者と見解が一致した点である。

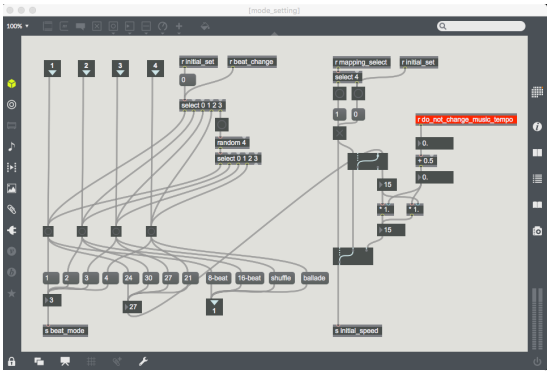

図 12 「mode_setting」サブパッチ
Figure 12 mode_setting.

図12の「mode_setting」はその名の通り、4種類の「8beat」「16beat」「shuffle」「ballade」という生成BGMスタイルを切り替えるブロックである。これらの名称は音楽において明確に定義/分離されているものではないが、”FMC3”[3]では「8beat」「16beat」「shuffle」の3種類だったものを4種類に増設してみた。音楽的には当然のことであるが、例えばある8ビートの生成BGMを突然に倍に刻んで16ビートにしてみると全く**違和感の塊**になるので使いものにならず、このサブパッチの右側ではそれぞれのスタイルに対応した良好なテンポ(default値)に瞬時に切り替えている。赤い「do_not_change_music_tempo」という「テンポを実験的に変化させるパラメータ」を受けているが、これは実験車で実際に「走行中に運転状況に対応してテンポを敢えて変化させる」という気持ち悪い体験のパラメータマッピングのために用意したものであり、実際にはここに変更値が飛び込んでくることはない。

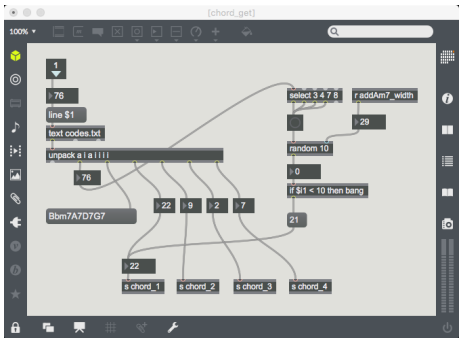

図 13 「chord_get」サブパッチ
Figure 13 chord_get.

図13の「chord_get」はその名の通り、刻々と4小節ループで進行する毎回の**コード進行**を読み込むサブパッチである。Maxのtextオブジェクトは起動時に図10のcodes.txtを読み込んでいるので、ここに行番号を与えれば、1行ごとの情報として「確認用のコード名表示」「コード進行番号」「各小節のコード4つ」が得られる。本システムでは後述するように、コードタイプとしては、Tonal Centerのrootの「CM7」以外は全て「**マイナー7thコード**」または「**ドミナント7thコード**」のいずれかとなっている。ブルース進行(12小節ループ)などで登場するサブドミナントの「FM7」は「4小節ループ/循環コード系」の繰り返しを基本とする本システムには登場しない。

内部データベースに4小節コード進行のパターンを格納するためのエンコードルールは以下である。

コードデータ = root(0-11) + type(0/12/24)
type = 0 はドミナント7thコード
type = 12 はマイナー7thコード
type = 24 はメジャー7thコード

例えば、G7=7、Dm7=2+12=14、CM7=0+24=24、であり、コードデータは0～24の値となる。自動作曲システムの研究では、コードタイプとして「dim7」「aug」「7sus4」「69」「7b5」その他のタイプまでなるべく多く盛り込んで「網羅」したつもりになっている発表も少なくないが、これらは全て**イロモノ**(退屈な音楽が進行する際の変化を齎すための不協和音)である。本システムは後述する音楽的理由からコード進行自体に**音楽美学的ダイナミクス**を盛り込んでいるので、**イロモノ**は不要である。

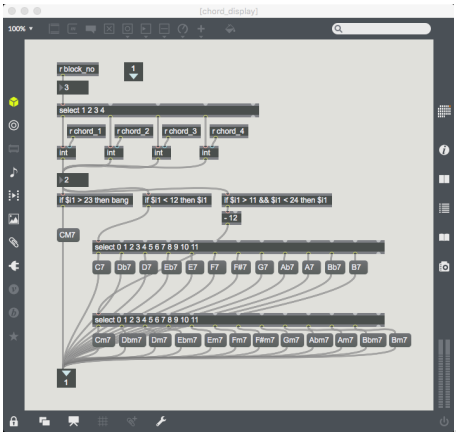

図 14 「chord_display」サブパッチ
Figure 14 chord_display.

図14の「chord_display」はその名の通り表示変換専用のサブパッチで、図13の「chord_get」で得られた「各小節のコード4つ」を保持しておいて現在の小節番号(1-4)によってトリガ出力することで選択し、計25種類のコードネーム文字列(定数)をメインパッチ内のメッセージオブジェクトに出力して表示するものである。codes.txtの各ライン冒頭にギッシリと詰まっている「確認用のコード名表示」から分離することも出来るのだが、刻々と現在の小節番号が飛んできて動作している事の確認も兼ねている。12個のドミナント7thコードと12個のマイナー7thコード、そしてただ1つのCM7を文字列として出力できる。

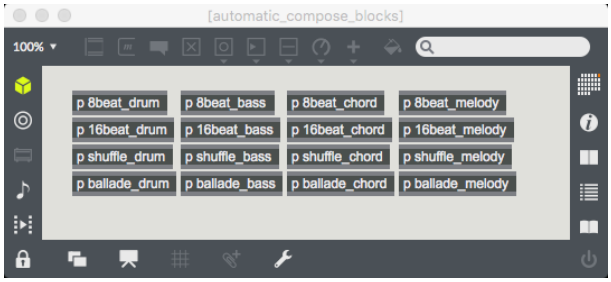

図 15 「automatic_compose_blocks」サブパッチ
Figure 15 automatic_compose_blocks.

図15の「automatic_compose_blocks」はその名の通り、本システムで実際に刻々とBGMを自動生成している心臓部である。見た目は16個のサブパッチが整然と並んでいるだけだが、このそれぞれを開くと広大な音楽生成アルゴリズ

Vol.2018-MUS-118 No.11
Vol.2018-SLP-120 No.11
2018/2/20

ムの世界が拓けてくる(→第8節)。このサブパッチ群は、4種類のBGMスタイル「8beat/16beat/shuffle/ballade」ごとに、それぞれ以下の4パートを生成するブロックとして分割されることで計16個となっている。

ドラムパート
ベースパート
コードパート
メロディーパート

ここではまず、簡単にそれぞれのパートの音楽的設計のポイントを紹介しておく。

ドラムパートでは、基本的に4種類のBGMスタイルごとに固定のドラムパターンを繰り返す。ただし一部で複数の候補からランダムに叩く(fill-in)パターンを選んで演奏の自然さを含ませている。また、カメラ画像からの画像認識情報として「**画面の複雑さ**」パラメータや**色認識情報**が与えられることを受けて、後述(第8節)するように主としてハイハットの細かい刻み音と、スネアドラムのゴーストノートを付加するデプスの「drums_exp」というパラメータ(0.0～1.0の実数値で、最小値0.0で効果ゼロ、最大値1.0で効果最大)を持っている。

ベースパートでは、後述(第7節)する**4小節コード進行**のルールで選ばれたそれぞれの小節ごとのコードの**root**と**5th**から4種類のBGMスタイルごとにシンプルなリズムでのベースパターンを演奏生成する。また、カメラ画像からの画像認識情報として「**画面の複雑さ**」パラメータや**色認識情報**が与えられることを受けて、わずかにチョッパー的なオブリガートを付加するデプスの「bass_exp」というパラメータ(0.0～1.0の実数値で、最小値0.0で効果ゼロ、最大値1.0で効果最大)を持っている。なお**16beatモード**では**7th音**も登場し、さらにコードが**CM7**である時にだけ**7th**(短7度)でなく**major7th**(長7度)となる。

コードパートではdefault音色を**エレピ**として、後述(第7節)する**4小節コード進行**のルールで選ばれたそれぞれの小節ごとのコードの構成音4音を、4種類のBGMスタイルごとに**アルペジオ的**に演奏して延ばすパターンとして演奏する。ただしアルペジオパターンとして4和音をどの順に連結するかは、4小節のループ単位で毎回ランダムに選択し、さらに4小節内の4つのコードでは全て同一パターンを演奏することで統一感を実現している。

メロディーパートとは本システムにおいては「擬似的にメロディー/アドリブソロ的に演奏される」単音パートの意味である。ベースやコードに対して際立つようにdefaultの音色を**ビブラフォン**としており、その生成ルールについて本項で次に詳細に紹介する。コード理論によれば、メロディー構成音中の一部のtension noteから、コードパートにおいて同時に鳴らされるコード構成音の一部(5th)を抜くよう**禁則**要請される場合がある。しかし本システムではコードパートの音色を減衰音であるエレピ等とし、またメロディーのリズムとしてコードパートと同時に鳴らないようタイミング生成する事で、この禁則要請を回避している。将来的にコードパートの音色に持続音を使用する場合には、この部分を考慮する必要がある。

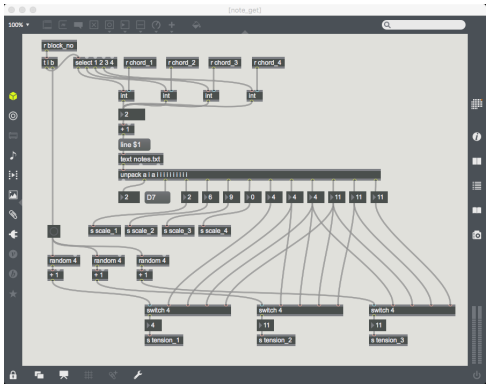

図 16 「note_get」サブパッチ
Figure 16 note_get.

図16の「**note_get**」はその名の通り、メロディパートが刻々と単音フレーズを生成する際にコードノートやテンションノートを選択するブロックである。メロディーを構成する音には2種類あり、chord note(コード構成音の4音)とtension noteからなる。**chord noteは常にメロディー音として使える。tension note**とは、**UST**[Upper Structured Triad:**上部構成3和音**](chord noteの4音の上部に3度音程で重ねたDiatonic Scaleからなる3和音)のうちavoid note[禁則により除外すべき音]を除いた残りの音である。なお、**NDC**(non diatonic chord)のドミナント7thコードのうち**Hmb5**スケール(和声的短音階)では短3度音程跳躍を補うために例外的にさらに候補に1音を加えてavoid note判定を行う。

図16の「**note_get**」では、Maxのtextオブジェクトは起動時に**notes.txt**を読み込んでいるが、これは25種類の**コードデータ**に対応して、以下のような内容である。

```
0 C7 0 4 7 10 99 99 99 99 99 99
1 Dd7 1 5 8 11 1 5 8 11 9 9
2 D7 2 6 9 0 4 4 4 11 11 11
3 Eb7 3 7 10 1 5 5 5 5 9 0
4 E7 4 8 11 2 4 8 11 2 6 6
5 F7 5 9 0 3 99 99 99 99 99 99
6 Gb7 6 10 1 4 99 99 99 99 99 99
7 G7 7 11 2 5 9 9 9 4 4 4
8 Ab7 8 0 3 6 8 0 3 6 8 4
9 A7 9 1 4 7 11 11 11 11 11 11
10 Bb7 10 2 5 8 0 0 0 0 4 7
11 B7 11 3 6 9 99 99 99 99 99 99
12 Cm7 0 3 7 10 99 99 99 99 99 99
13 Dbm7 1 4 8 11 99 99 99 99 99 99
14 Dm7 2 5 9 0 4 4 4 7 7 7
15 Ebm7 3 6 10 1 3 6 10 1 9 0
16 Em7 4 7 11 2 9 9 9 9 9 0
17 Fm7 5 8 0 3 99 99 99 99 99 99
18 Gbm7 6 9 1 4 99 99 99 99 99 99
19 Gm7 7 10 2 5 99 99 99 99 99 99
20 Abm7 8 11 3 6 8 11 3 6 8 2
21 Am7 9 0 4 7 11 11 11 2 2 2
22 Bbm7 10 1 5 8 10 1 5 8 10 7
23 Bm7 11 2 6 9 99 99 99 99 99 99
24 CM7 0 4 7 11 2 2 2 9 9 9
```

4小節コード進行の各小節において、このデータを**コードデータ**により参照すれば、**そのコードにおいて鳴らして良い音**が与えられる。この音楽的な根拠として、全てのコードに対してメロディー音の候補の検討を、Diatonic Chord、NDC- Dominant7th Chord、NDC- minor7th Chordの

3カテゴリごとにtension noteの判定理由について記述すると、以下のようになる。ここで上のnotes.txtは

コードデータ(0～24)
コード名
chord note(4データ)
tension note(6データ)
none=99、tension1種→各6個、
tension2種→各3個、tension3種→各2個

というルールでまとめたものである。なお、Non Diatonic Chordsのマイナー7thコードの一部で、基調のtension noteとして使いにくい(avoidでないもののchord toneとぶつかるのでchordパート側で鳴らさない要請が入る)ものについては、上記notes.txt中の出現確率をさらに抑制させている。

「Diatonic Chord」のtension note

CM7 = C E G B + D F A --- C Ionian → CM7(9,13)
11thのF音は3rdのE音とm9th音程のためavoid
コードデータ[24] → D A (2,9)

Dm7 = D F A C + E G B --- D Dorian → Dm7(9,11)
13thのB音は3rdのF音とtritoneでSD機能を失うのでavoid
コードデータ[14] → E G (4,7)

Em7 = E G B D + F A C --- E Phrygian → Em7(11,b13)
9thのF音はrootのE音とm9th音程のためavoid
※b13thのC音を使う場合はchordから5thのB音を抜く
コードデータ[16] → A C (9,0)

FM7 = F A C E + G B D --- F Lydian → FM7(9,#11,13)
no avoid
※本システムではコード進行として登場しない

G7 = G B D F + A C E --- G MixoLydian → G7(9,13)
11thのC音は3rdのB音とm9th音程のためavoid
コードデータ[7] → A E (9,4)

Am7 = A C E G + B D F --- A Aeorian → Am7(9,11)
13thのF音は5thのE音とm9th音程のためavoid
コードデータ[21] → B D (11,2)

Bm7b5 = B D F A + C E G --- B Locrian → Bm7b5(11,b13)
9thのC音はrootのB音とm9th音程のためavoid
※本システムではコード進行として登場しない

「NDC - Dominant7th Chord」のtension note

C7 ※V7/IVのSec.Dであるが本システムでは登場しない
Db7 = Db F Ab Cb + D F A → Db7(b13)
b9thのD音はrootのDb音とm9th音程のためavoid
b11thのF音は3rdにある
※b13thのA音を使う場合はchordから5thのAb音を抜く
コードデータ[1] → A (9)

D7 = D F# A C + E G B --- D MixoLydian → D7(9,13)
11thのG音は3rdのF#音とm9th音程のためavoid
コードデータ[2] → E B (4,11)

Eb7 = Eb G Bb Db + F A C → Eb7(9,11,13)
※#11thのA音を使う場合はchordから5thのBb音を抜く
コードデータ[3] → F A C (5,9,0)

E7 = E G# B D + F A C --- A Harmonic Minorなので
F#(#9)を補う → E7(9,#9,b13)
9thのF音はrootのE音とm9th音程のためavoid
11thのA音は3rdのG#音とm9th音程のためavoid
13thのC音は5thのB音とm9th音程のためavoid
コードデータ[4] → F# (6)

F7 ※本システムではコード進行として登場しない
Gb7 ※V7/VIIのSec.Dであるが本システムでは登場しない
Ab7 = Ab C Eb Gb + A C E → Ab7(b13)
b9thのA音はrootのAb音とm9th音程のためavoid
b11thのC音は3rdにある
※b13thのE音を使う場合はchordから5thのEb音を抜く
コードデータ[8] → E (4)

A7 = A C# E G + B D F --- D起点の
上行Melodic Minor → A7(9,b13)
11thのD音は3rdのC#音とm9th音程のためavoid
b13thのF音は5thのE音とm9th音程のためavoid
コードデータ[9] → B (11)

Bb7 = Bb D F Ab + C E G --- F起点の
上行Melodic Minor → Bb7(9,#11,13)
※#11thのE音を使う場合はchordから5thのF音を抜く
コードデータ[10] → C E G (0,4,7)

B7 ※V7/IIIのSec.Dであるが本システムでは登場しない

「NDC - minor7th Chord」のtension note

Cm7 ※本システムではコード進行として登場しない
Dbm7 ※本システムではコード進行として登場しない
Ebm7 = Eb Gb Bb Db + F A C --- Bb Harmonic Minorなので
G(#9)を補う → Ebm7(9,#11,13)
#9thのF音は3rdのGb音とm9th音程のためavoid
※#11thのA音を使う場合はchordから5thのBb音を抜く
コードデータ[15] → A C (9,0)

Fm7 ※本システムではコード進行として登場しない
Gbm7 ※本システムではコード進行として登場しない
Gm7 ※本システムではコード進行として登場しない
Abm7 = Ab Cb Eb Gb + B D E → Abm7(#11,13)
#9thのB音は3rdにある
13thのE音は5thのEb音とm9th音程のためavoid
※#11thのD音を使う場合はchordから5thのEb音を抜く
コードデータ[20] → D (2)

Bbm7 = Bb Db F Ab + C E G → Bbm7(9,#11,13)
※9thのC音を使う場合はchordから3rdのDb音を抜く。
※#11thのE音を使う場合はchordから5thのF音を抜く。
コードデータ[22] → G (7)

Bm7 ※本システムではコード進行として登場しない

上述のルールに従って、あるコードが演奏されている時にメロディーを構成する音の候補は、**chord noteの4音**と、許される**tension noteの1～3音**の、計5～7音から構成される。ここでchord noteのうちrootと5thの音はベースパートでも演奏される可能性が高いので、メロディー構成音としての出現確率をやや低下させている。またchord noteのうち3rdと7thの音はコードパートでも演奏される可能性が高いので、メロディー構成音としての出現確率をわずかに低下させている。

tension noteのメロディー構成音としての出現確率は、本システムにおいては外部環境センサから与えられたデータ「tension_range」(0.0～1.0の実数値)によって変化する。すなわち、運転状況に対応して**メロディーのテンション(緊張度/不協和度)**が変化する、というのが本システムの生成するBGMの大きな特徴である。tension_range=0.0ではメロディー音はchord noteからだけ選ばれ、そこからtension_range=1.0に向けて大きくなるに従って、chord note以外のtension noteからもメロディー構成音が選ばれ

る確率が増大していく。

コードパートにおいて小節ごとのコードの構成音4音をアルペジオ的に演奏するパターン(順序)が毎回ランダムに選択されて、4小節内の4つのコードでは全て同一パターンとなることで統一感を実現しているのと同じように、メロディーパートにおいて生成されるメロディーのリズムも、基本的には4小節内の4つのコードでは全て同一パターンとなるように生成し、単調さを避けるためにいくつかのランダム確率でリズムパターンを微細に変化させるオブリガートを施している。なおカメラ画像からの画像認識情報として「**画面の複雑さ**」パラメータや「**色認識情報**」が与えられることを受けて、このオブリガートを付加するデプスの「melody_exp」というパラメータ(0.0～1.0の実数値で、最小値0.0で効果ゼロ、最大値1.0で効果最大)を持っている。

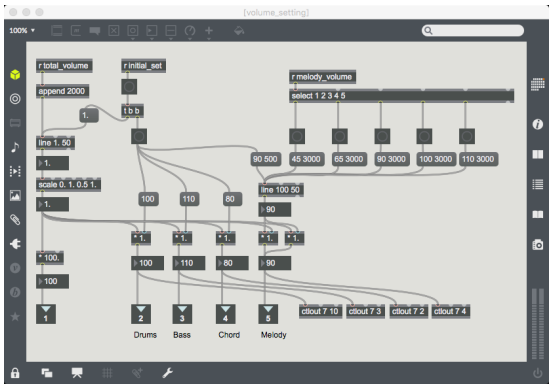

図 17 「volume_setting」サブパッチ
Figure 17 volume_setting.

図17の「volume_setting」はその名の通り、**ドラム/ベース/コード/メロディー**の各パートのトラックボリュームをコントロールするブロックである。初期値として**ドラム/ベース/コード**の3パートに100:110:80というバランスで設定されたところに、total_volumeというパラメータでの全体の音量コントロールが加わり、またmelody_volumeというパラメータによって、別個にメロディーパートの音量を5種類のシーンに対応してクレシェンド/デクレシェントさせるような処理を実装した。同時にこれら各パートの音量変化をメインパッチ内のスライダーオブジェクトに出力して表示しているが、このスライダー群はユーザの操作を禁止しているので「表示専用」である。

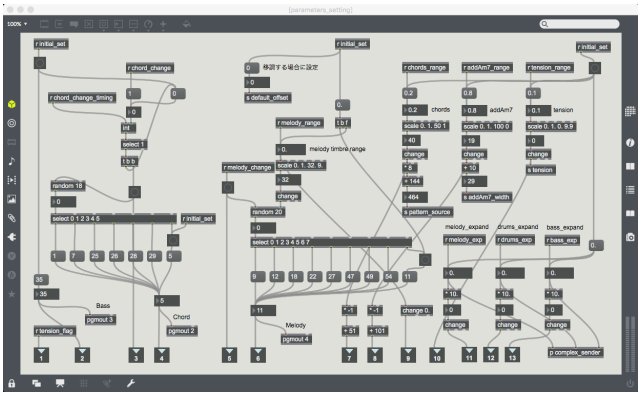

図 18 「parameters_setting」サブパッチ
Figure 18 parameters_setting.

図18の「parameters_setting」はその名の通り、メインパッチ右側に並ぶ各種の音楽演奏表現パラメータ群をスライダー表示したり現在の演奏状況を表示(ここもユーザからの操作は禁止)しているが、もちろん内部的にはこれら音楽演奏表現パラメータを上流のパラメータ変換用中間パッチparameter_convert010.maxpatから受け取って、BGM生成ブロックに供給するという、パラメータ分配の中継所となっている。

6. 音楽を駆動するダイナミクス

次の第7節では、本システムで採用した152種類の**4小節コード進行**の具体的内容(音楽的理論)を紹介するが、本節ではそれに先立ち、本研究で注目した**音楽を駆動するダイナミクス**という、BGM生成アルゴリズムの音楽的な基礎を確認する。音楽ヒューリスティクスと言っても、筆者は本研究で「これまで無かった新規な音楽理論」などと提唱するつもりは毛頭なくて、専門家には当たり前の王道の音楽的常識を整理しただけの事である。第3節「3.4 **楽典**」の項にあった音楽的内容からさらに発展していくが、詳細は「**音楽理論特訓集中講座**」[23]にあるので、ここではそのエッセンスを簡単に整理する。

第3節ではコードについて、Tonal CenterをCとすると、Diatonic Scaleは「C,D,E,F,G,A,B」、Diatonic Chordは「CM7,Dm7,Em7,FM7,G7,Am7,Bm7b5」、6種の主要和音はそれぞれ、Tonic[T](CM7,Am7)、Dominant[D](Em7,G7)、SubDominant[S](Dm7,FM7)の**機能Function**を持つとした。全てが4和音(7th chord)であり、今後、3和音は「たまたま7thが省略されたシンプルな(味わいの乏しい)和音」として軽く扱う。ここが考察の起点となる。

6.1. 循環コードとドミナントモーション

古典的な和声理論の教科書では、**カデンツァ**と呼ばれる「T→S→D→T」とか「T→S→T→D→T」の進行から話が始まる(ここではコードネームでなくFunctionであることに注意)。ただしこれらは**古風でダサくて使えない**。最後に**Tonicの終止**があるので、そこで終わってしまって音楽が続いていかないのである。本システムが対象としているBGM生成は、いくらでも続いていく、つまり**終止のない**金太郎飴のような**ループ音楽**なので、**カデンツァ**で終了しては駄目なのである。そこで「T→T→S→D」と枠組みを変更して、4小節ループの最後の「**DominantからTに戻る**」機能によって各4小節ループを無限に繰り返すようにする。これが定番中の定番「**循環コード**」の原点である。

「T→T→S→D」の最初の2つのTが同じコードでは面白くないので、最初が基調のCM7(T)、続いて並行和音(4音のうち3音が共通)のAm7(T)とすると、このコード進行は

CM7→Am7→FM7→G7

となる。一応これも循環コードの一種と言えるが、実際にはこれも**ダサくて使えない**。その理由は3番目のコードFM7にある。理由は後述するが、このSとして並行和音(4音のうち3音が共通)のDm7(S)とすると、このコード進行は

CM7→Am7→Dm7→G7 (★)

となる。実はこれこそ人類の歴史上、世界中の津々浦々で

定番進行であり続けている「**循環コード**」である。このコード進行が何故、いくら繰り返しても飽きない魅力を持っているのかを考察しよう。

まず、楽典では**カデンツァ**のところで解説されているように、Dominantとは「**Tonicに行きたい!**」と切に希求する機能である。ドミナントDは、基調(Key)のrootに**p5**下行する和音、つまりKey=**C**であれば**G7**である。この[**0-4-7-10**]の音程構成の和音を**ドミナント7th**と呼ぶが、ダイアトニックスケールの枠組みでは5の和音**だけ**に登場する。ピアノで「**(CM7→)G7→CM7**」と弾けば、小学校で「着席」と教わったものと一致する。つまり、「**Dominant→Tonic**」の動きは、これで解決した、終了した、安堵した、という**終止感**をもたらす。この「**Dominant→Tonic**」の動きのことを、ドミナントモーションDominant Motionと言い、音楽を動的に語る上での最大のキーワードである。ドミナントモーションを、その連結されたコードの**root**(ベース)の動きとして確認すると、**完全5度下行**あるいは転回して**完全4度上行**である。オクターブ移動によって同じことなので、これもまた**p5**(下行を省略)と呼ぶ。

ドミナントモーションとは、DominantがTonicに解決したい、解決しよう、解決するぞ‥という強いエネルギーであり、和声的に音楽を駆動しているダイナミクス原理と言える。そこでこの原理を、和音の構成音に分解して検討する。Dominant和音は**G-B-D-F**の**G7**コード、Tonicはここでは**M7th**のB音を省略した**C-E-G**の「素の」Cmajorとする。まずrootに注目する。**G7→C**の進行のrootは**G**から**C**への**p5**(**ソ→ド**)である。これは音組織のもっとも基本的な**p5**音程の跳躍ということで、**rootのp5進行**がドミナントモーションの重要な要素である。さて、rootの**ド**に対しては、**G7**の**B**から**C**への**半音上行**(**シ→ド**)もあり、これを「**導音**」という。**p5**と並んで音楽において自然な進行は、**半音進行**(最小基本単位の推移)である。そしてTonicの第3音**ミ**に対しては、**G7**の**F**から**E**への**半音下行**(**ファ→ミ**)もある。また第5音の**ソ**は、**G7**と**C**との共通音(和音の枠組み)である。これらの事実から、**G7→C**の動きは、全ての音に**解決**という性格が宿っており、この機能がより強い関係の和音は存在しない。これがドミナントモーションの第一の意味である。

Diatonic Chord「**CM7,Dm7,Em7,FM7,G7,Am7,Bm7b5**」の中で、唯一のドミナント7thコードが**G7**である事を思い出そう。(7は除外して)他の和音は、例えば**CM7**、**FM7**は、**下3音がmajor3和音、上3音がminor3和音**からなり、単独でも安定している。そして**Dm7**、**Em7**、**Am7**は、**上3音がmajor3和音、下3音がminor3和音**からなり、これも単独でも安定している。ところがドミナント7thの**G7**をよく眺めてみると、1度と5度は**p5**の枠組みであるが、3度と7度の音は**トライトーン**である。主要Diatonic Chord(1から6)の中で**唯一、G7だけ**が和音の内部に不協和音程を持つことで、**G7**はそれ自身で停滞することを拒み、**どこかに解決(進行)することを強く希求**している。トライトーンが最小単位の半音の推移で解決されるパターンは次の2つである。**シ-ファ**の減5度が半音1つずつ狭くなれば**ド-ミ**の長3度に**解決**する。**ファ-シ**の増4度が半音1つずつ広くなれば**ミ-ド**の短6度(長3度の転回形)に解決する。いずれにしても、トライトーンは**同時に解決**して、人間は無意識のうちにとても嬉しい。これがドミナントモーションの第二の意味である。

6.2. 「II-V」

さきの「**T→T→S→D**」で「**CM7→Am7→FM7→G7**」と★印の「**CM7→Am7→Dm7→G7**」の循環コードが登場したが、その最後の**G7**こそ、4小節ループの最後を受けて冒頭に戻りたいドミナントモーションの主役である。ではこの2種類のコード進行、具体的には3番目のコード**FM7**がダサくて駄目で**Dm7**がイケてる理由は何か。コード進行における「**root音の推移**」に注目するのがポイントとなる。4小節目の**G7**に進む直前の3番目のコードのrootの動きを見ると、**FM7→G7**では**全音上行**であるのに対して**Dm7→G7**では**p5下行**しているのが決定的な違いである。

生態学ecologyの視点から考察してみると、地球上の生物の遺伝子は、過去から未来に進む**時間**と、地球の中心に向かう(地表では鉛直方向)重力に支配された**空間**という時空間的制約の下で数億年かけて進化形成されてきた。従って我々の脳内プロセスは**時間**的には常に無意識的に過去の記憶を参照しつつ未来の状況を推測しているし、**空間**的には上下方向において無意識的に地表を基準としている。音楽聴取で言えば、例えば4小節コード進行の音楽を聞けば、直前に聞いた和音から現在の和音への推移、さらにはその前の和音からの推移を無意識に確認しつつ次の和音を無意識に**予測/期待**して聞いている。ドミナントモーションの嬉しさもここに由来する。最小単位の**半音**であればトライトーンの解決のように半音上行も半音下行もアリなのだが、**FM7→G7**の**全音上行**となると、無意識な上下方向感覚(ニュートンの林檎のように下に落ちるのが自然)としてはやや無理がある運動であり、root音がグイッと**全音上行**する動きは違和感として好まない(rootが**全音下行**するコード進行は歓迎)。

一方、もっとも基本的な**p5**による**Dm7→G7**の**p5下行**は、これこそ地球上で進化してきた我々が好む下に落ちる動きであり、**G7**がさらに次の**T**の**CM7**に戻るrootも**p5下行**なので「**2連発のp5下行**」として後半のドミナントモーションを強く実感できる。さらに★印の「**CM7→Am7→Dm7→G7**」の循環コードでは、2番目から3番目のコード進行**Am7→Dm7**もまた**p5下行**であり、つまり2-3-4番目のコードのrootの動きは「**3発連続のp5下行**」を期待させる強さがあり、これこそあまたあるコード進行の中で**循環コード**が王様と言われる所以なのである。

この循環コードの中盤にある**Am7→Dm7**は別にして、後半の**Dm7→G7**のrootの**p5の2段連鎖**は、聞き耳を立ててみるとそれだけでなかなか美しい進行である。有名なR&BやPopsの名曲に「**Dm7→G7**」だけを延々と繰り返すことで気持ち良くトリップ出来るという曲が何十曲も存在する。そこでより一般的に見ると「**あるドミナント7thコードにrootがp5進行するマイナー7th和音**」という進行は、かなりイケて、使えて、妥当である。これを「**II-V進行**」と呼ぶ。正確には「IIm7-V7」進行であるが、一般に**II-V進行**として普及しており、本システムでももちろん採用した。

6.3. ドミナントモーションの拡張とSec.D

ここまでDominantは、基調のrootのTonicに対する5のドミナント7th和音として見てきたが、ドミナントモーションの本質が**rootのp5進行**と**トライトーンの解決**であるとすれば、ドミナントモーションの原理は音楽のより広い局面に拡大して活用できる。例えば本システムでは登場しないものの「**C→C7→FM7→G7**」というあまりに有名なコード進行があるが、ここでは突然にDiatonic Chordでもない2番目の**C7**が登場する。実はこれはドミナントモーションの格好の応用例であり、C7の内部のトライトーン(E-Bb)は、**rootがp5進行する**相手に強く進行(解決)したいと希求している。その相手こそ次の**FM7**なのである。つまりこの原理を応用すると、調的な枠組みにおける位置付けを離れて、ある和音に対してその前に、**そこへrootがp5進行するドミナント7th和音**を持ってくれば、この進行はかなりイケてる/使える/妥当である、という可能性がとても高いのである。これがドミナントモーションの第三の意味付けである。

さて、ドミナントモーションの概念の拡張として、ここで★印の「**CM7→Am7→Dm7→G7**」から拡張していく。まず3番目の**Dm7**を**D7**にした

CM7→Am7→D7→G7

を考える。耳にしてみれば判るが、とても魅力的かつダイナミックである。これは何故か。理由の一つは、2番目から3番目の進行**Am7→D7**が、まさに**II-V進行**となっているからである。また3番目から4番目の進行**D7→G7**は、後ろのコード**G7**に対して、上で拡張した意味でのドミナントモーションになっている。この**D7→G7**での**D7**のように、**後ろのドミナント7thコードに対してrootがp5で進行するドミナント7thコード**を「**セカンダリードミナント(Sec.D)**」と言う。Sec.Dを知るとコード進行の持ち駒は飛躍的に増加する。ここでもrootは**A→D→G(→次のC)**という、**p5下行の3段連鎖(!!)**となっている。このコード進行は**NDC**(ノンダイアトニックコード)の**D7**というコード(Key=Cのスケールにおいて登場しない**F#**音)を含みながら、とても美味しいコード進行なのである。

さて、今度は★印の「**CM7→Am7→Dm7→G7**」の2番目のコード**Am7**を、これも**NDC**の**A7**というコード(Key=Cのスケールにおいて登場しない**C#**音を含む)にすると

CM7→A7→Dm7→G7

となる。これは次に続く**Dm7**はドミナント7thではないので**Sec.D**ではないが、前述の「**拡張されたドミナント7thコード**」と解釈でき、実際に音にしてみるとたまらなく美味しい進行である。後半3つのrootは同様に**p5下行の3段連鎖(!!)**であり、さらに前半3つのコードに**C→C#→D**という半音上行の美しいメロディーが隠れている。

そして最後に、これらの合体した

CM7→A7→D7→G7

がある。解釈すれば、**Sec.Dの2連発**ということになる。後半3つのrootは同様に**p5下行の3段連鎖**し、高揚した感じでドミナントモーションが続く、これまた定番の美味しいコード進行であり、スグにでも使える。これら各種パターンの4小節ループを適度に交代しながら繰り返すLoop Musicは、もはや単調とは言わせない豊富な表情をもった音楽になっているが、その基本原理は**ドミナントモーション**、II-V、Sec.D、というたった3つの基本ルールだけなのである。

ここで改めて最後の「**CM7→A7→D7→G7**」を眺めると、和声進行の(無意識的な)解釈とは、**後ろから前に説明がつくことの連鎖**とも言える。人間は時間とともに音楽を聴取しながら、過去の和声を短期記憶に格納して、新たに到着した和声との繋がりとして(無意識に)認知/解釈/反応している。例えば**CM7→A7**と進行したA7の瞬間は、前の**CM7**と後の**A7**との関係にちょっと戸惑う。ところが次に続いた**D7**により、**A7→D7**のSec.D進行によりナルホドと納得し、ここにさらに**G7**が続くと、**D7→G7**のSec.D進行、あるいは気付いてみればrootはなんと美しい**p5下行の3段連鎖**、そして過去に遡れば最初の3和音に**C→C#→D**という半音上行の美しいメロディーまであった、そこまで展開していくための**A7**なのだった、あぁぁ美しい、と**あと付け**で解釈して、このコード進行(の余韻、また繰り返されるかもしれない期待)を楽しむのである。Jazzミュージシャンはそのようなのちのちの効果を意図して、刻々とコードのバリエーションをアドリブで選んで演奏するのである。

6.4. 裏5度(Sub.D)とExt.D

クラシックのドミナントはトニックの**p5**上とあるが、Jazz屋のドミナントはこれだけでなく、**トニックの半音上**も同等にドミナントである。これを**裏5度**と言い、Sub.Dと記述する。Key=Cであれば古典的なドミナントは**G7**である。そしてこの裏5度は、**5度円**で**G7**のちょうど反対側、rootがトライトーン(3全音)だけ離れた**C#/Db**なので、コードネームとしては**C#7/Db7**である(同じなので今後は**Db7**)。ドミナントモーションの図式としては、古典的な**G7→CM7**に対するSub.Dとしては**Db7→CM7**となる。最後の**CM7**はCでも、よくあるC6でも構わない。maj7thも6thも9thもちょっとした味付けであり、重要なのは「**rootの半音上のドミナント7th**」である。5度円において基準音に順に作用させて12個の音の連鎖を生成できるのは、**p5**と基本単位の**半音**しかないと述べた。その**p5**と**半音**とが、ここで晴れて対等に並んだ。

ではどうしてSub.Dもドミナントなのか、これは構成音を調べてみれば明白である。Key=Cにおいてドミナントの**G7**を眺めると、**G-D**の**p5**と、**B-F**のトライトーンからなる。ところでSub.Dの**Db7**を眺めると、**Db-Ab**の**p5**と、**B-F**のトライトーンからなる。つまりこの2つのコードに共通するのは、**B-Fのトライトーン**+(**協和する**)**p5音程**なのである。そして前述のように「**B-F**のトライトーン」こそ、キー**C**における**G7**のドミナントモーションの存在理由であった。完全5度の枠組みというのはそれ自体の色彩は無色透明であり、色はトライトーンが付けている。これが**G7**と**Db7**で完全に共通であるために、両者は正当なドミナントなのである。**G7**は、**CM7**への解決の際にrootが**p5**跳躍という爽快さを持つが、ボイシングを変えた場合、**G→G**という面白くない経過音となる可能性も秘めている。一方、Sub.Dの**Db7**では、**Db-Ab**の**p5**は**CM7**の**C-G**に対して、**いずれも半音ずつズルッと下行することで解決**する。どうやってもズルッと

Vol.2018-MUS-118 No.11
Vol.2018-SLP-120 No.11
2018/2/20

半音下行する。これはp5ジャンプの潔い解決に比べてある意味では**色っぽい**(Jazzっぽい)動きであり、Jazz屋はむしろこちらを優先することも多い。

C#7/Db7は同じコードなのでここではDb7として、前述の循環コードのドミナントG7をSub.Dにすると

CM7→Am7→Dm7→Db7
CM7→Am7→D7→Db7
CM7→A7→Dm7→Db7
CM7→A7→D7→Db7

のようになる。そして本節の最後として、コード進行のバリエーションの可能性をさらに増大させるドミナントモーションの拡張を追加する。以下はその一例であり、これまでに紹介した理論では出てこないコード進行である。

CM7→D7→Dm7→G7

これはJazzコード理論に登場する「**コードをまたぐコード進行**」である。ここまでの説明では、あるコードに進行する**前の**コードとは、**直前**であった。しかし、人間の崇高な脳機能には、**直前**のコードからだけでなく、1つとばして**もう1つ前のコードから**のドミナント進行を、納得するコード進行の理由付けとして解釈してくれる能力もある。これを**拡張ドミナント**(Ext.D)と言う(文献によっては「Diatonic Chordに進行するNDCのドミナント7thコードのみをSec.Dと呼び、それ以外の全てのNDCに進行するNDCのドミナント7thコードをExt.Dと呼ぶ」という説もあるが本稿では上記の立場をとる)。上のD7→Dm7の部分だけを見ると、なんでドミナント7thコードから、同じrootのマイナー7thに行くのか分かりにくいが、実は文脈はD7→(パス)→G7という進行なのである。つまり上のコード進行の最後のG7は、II-VとしてDm7→G7と受けつつ、同時にExt.DとしてD7→G7と両方を受けている事になる。この例ではrootがD→Dのままで面白くないが、**裏5度**が等価なドミナントである、という拡張からD7をAb7に置換してこの単調さを打開する事は容易であり、コード進行のバリエーションは本システムにおいても飛躍的に拡大している。

6.5. FMC3との関係

FMC3では「新しい4小節ループに進行した場合、Tonal Centerをランダムに12音から変えてもよい」(局所的な移調の連続)としたが、本システムでは基本的にKey=CのTonal Centerを維持する。これは、移調がなくても本システムではコード進行パターンのバリエーションを全152種類とFMC3に比べて大幅に拡大したので、ループの先頭部分で**局所的な移調感**を持ち、その後に妥当なコード進行で解決に向うためである。

まとめると、本システムではここまで解説したような音楽的理論に基づき、次に続く4小節ループの冒頭にはTonic(調的中心)CM7に帰着進行することを期待させる(**Tritoneの解決**)というドミナントモーションの原理から、4小節目のコードは必ずドミナント7thコードのG7または裏5度のDb7の**いずれか**とした。この2つのコードは**共通のTritone:F音とB音**を持っており、F→EおよびB→Cというトライトーンの解決(**増4度→長3度**あるいは**減5度→完全5度**という音程の変化)を内包している。

7. 4小節コード進行のパターン

7.1. 循環コード系(8パターン)

コード進行パターン(1～152)のうちの先頭8種類(**循環コード系**)については、4小節コード進行の4小節目のコードであるG7またはDb7から、先頭のコードとして正直にドミナントモーションでrootトニックのCM7で受けた形で始まるものである。なおこの8パターンについては、他のパターンに対する優位性を持たせ、重み付けを変えるパラメータchords_rangeによって、この循環コード系グループの出現確率のpriorityを上げている(次項で解説)。

pattern = 1 「CM7→Am7→Dm7→G7」 (24, 21, 14, 7)
もっとも基本的な循環コード。後半2小節がII-V、2→3小節はドミナントモーションではないがrootが**p5下行**なので無難/良好に進行する。

pattern = 2 「CM7→Am7→D7→G7」 (24, 21, 2, 7)
循環コードの3小節目のDm7の代わりにノンダイアトニックコードのD7を用いた進行。Am7→D7がII-V、D7→G7がSec.Dとなっている。

pattern = 3 「CM7→A7→Dm7→G7」 (24, 9, 14, 7)
循環コードの2小節目のAm7の代わりにノンダイアトニックコードのA7を用いた進行。CM7→A7→Dm7にC→C#→Dという**半音上行**の進行が含まれ、後半2小節がII-V、A7→Dm7がSec.Dとなっている。

pattern = 4 「CM7→A7→D7→G7」 (24, 9, 2, 7)
循環コードの2小節目のAm7の代わりにノンダイアトニックコードのA7を、3小節目のDm7の代わりにノンダイアトニックコードのD7を用いた進行。A7→D7→G7がSec.Dの2連発になっている。

pattern = 5 「CM7→Am7→Dm7→Db7」 (24, 21, 14, 1)
pattern = 6 「CM7→Am7→D7→Db7」 (24, 21, 2, 1)
pattern = 7 「CM7→A7→Dm7→Db7」 (24, 9, 14, 1)
pattern = 8 「CM7→A7→D7→Db7」 (24, 9, 2, 1)
循環コード系の上の4パターンの4小節目のG7の代わりに、裏5度(Sub.D)であるDb7を用いた進行。

7.2. 出現確率の上げ方とAm7への置換

ここで、先頭コードがTonal Centerトニックの**CM7**であるもの(pattern=1からpattern=8まで)の出現確率のpriorityを上げる方法と、さらに「**CM7をAm7に置換する**」というバリエーションの実現方法について述べる。

8パターンの循環コード系進行の出現確率を全体のコード進行(152種類)に対して高める余地を設定するパラメータがchords_range(0.0～1.0の実数値)である。patten NO.は1～152の152種類であるが、内部的には0～151の剰余値pattern_selectとして扱う。4小節ごとにコード進行パターンを刻々とランダム選択するための乱数幅設定値pattern_sourceが152である場合には、そのままこの値がpattern_selectとなる。これは152種類のコード進行パターンが等確率で選ばれるため、**コード進行のバリエーションという意味で音楽の複雑さが最大**になる。これがパラメータ最大値「chords_range=1.0」の場合である。

そして、pattern_sourceの値が153以上である場合にはpattern_sourceの値から152を引き、残った差に対して8の剰余をとった値をpattern_selectとする。例えば「pattern_source=160」の場合には「残った差」は0～7となって、対象となる8パターンの循環コード系進行の出現確率は他144パターンの2倍となる。同様に「pattern_source=168」では他144パターンの3倍、さらに「pattern_source=176」では他パターンの4倍、・・・と続く。バリエーションのパラメータchords_rangeの最小値はchords_range=0.0であり、この時に内部的には「pattern_source=544」となって他144パターンの50倍となる。すなわちパラメータchords_range(0.0～1.0)が小さくなるほど、馴染みの8パターンの循環コード系進行の出現確率が大きくなり、コード進行のバリエーションという意味で音楽の複雑さが小さくなる。この処理を行っているサブパッチは図11の「master_clock_generator」である。

また、これら8パターンの循環コード系進行のうち2番目のコードがA7である「3・4・7・8」の4種類については、常に循環コード系進行の先頭コードがCM7である**単調さを避ける**ために、確率的にバリエーションを持たせるパラメータaddAm7_rangeがある。このルールは「**先頭のCM7を並行コード(同じTonic)のAm7に置換する**」というものである。実際にはこの特別な置換を実行する確率をパラメータaddAm7_range(0.0～1.0の実数値)を100～0の整数値にスケーリングした値に10を加算したデータをrandomの範囲として「10未満の時にCM7をAm7に置換する」と判定処理する。この処理を行っているのは図13の「chord_get」内である。そこで最大値パラメータaddAm7_range=1.0では常にCM7をAm7に置換し、最小値パラメータaddAm7_range=0.0では「確率10/110でCM7をAm7に置換」するので、めったにこのバリエーションが起きなくなる。なお「1・2・5・6」のパターンでは2番目のコードがAm7で**同一コードが連続**するので、この置換ルールは**適用しない**。

7.3. FMC3からのコード進行ルール(1)

コード進行に著作権は存在しない。そこでFMC3[3]では、多数の「色々な音楽に登場する**よくあるコード進行**」をまず網羅し、さらにドミナントモーションの連鎖で進行するジャズのコード進行ルールを参考にして、以下のような規則で考えられる組み合わせのコード進行も加えて計53種類の4小節コード進行パターンを考え出した。しかしこの考察には抜けがあったので、本システムではさらに考えられる組み合わせルールを盛り込んで拡張し、上述の循環コード系の8パターンと加えて計152種類となった。そして理屈抜きの「**よくあるコード進行**」はカットした。

4小節のコード進行は、「前2小節のパターン」と「後2小節のパターン」とを連結する

前後それぞれの「2小節パターン」において、後ろのコードは必ずドミナント7thコードとする

それぞれの「2小節パターン」において、前のコードは「ドミナント7th(II7→V7)かマイナー7th(IIm7→V7)のいずれか」とする。このrootは「完全5度下行か半音下行(裏5度進行)のいずれか」の2通りで進行する

ここで「前2小節のパターンと後2小節のパターンとを連結する関係」としては、「前2小節の後ろのコードと,後2小節の前のコードを隣接進行Sec.Dで連結」するか、「前2小節の後ろのコードと,後2小節の後ろのコードをExt.Dで連結」するか、のいずれかである。Key=Cとして、G7に進むのは、Dm7とD7(p5下行)と、Ab7とAbm7(半音下行)。Db7に進むのは、Abm7とAb7(p5下行)と、D7とDm7(半音下行)。つまり後半2小節の組み合わせは以下の2×4＝8通り。

Dm7→G7 / D7→G7 / Ab7→G7 / Abm7→G7

Abm7→Db7 / Ab7→Db7 / D7→Db7 / Dm7→Db7

ここへの前2小節との連結は、「前半2小節の後のコードからExt.Dで後半2小節の後のコードに進む」か、「前半2小節の後のコードからSec.Dで後半2小節の前のコードに進む」か、のいずれかなので、次項からこれらの組み合わせ方法と生成されるコード進行を示す。

7.4. FMC3からのコード進行ルール(2)

前半2小節の後のコードからExt.Dで後半2小節の後のコードに進む場合のうち、後半2小節の後のコード(4小節目のコード)がG7の場合は以下のようになる。

□→D7→◇→G7 / □→Ab7→◇→G7

◇にその直前と同じコードが入るものは除外するので、以下の6つだけが候補となる。

□→D7→Dm7→G7
□→D7→Ab7→G7
□→D7→Abm7→G7
□→Ab7→Dm7→G7
□→Ab7→D7→G7
□→Ab7→Abm7→G7

この□の部分(1小節目のコード)に、続くコードに対してII-VとSec.Dの各2通りが入る(p5下行と半音下行)ので、以下の24通りが全てOKとなる。

pattern = 9 「A7→D7→Dm7→G7」 (9, 2, 14, 7)
pattern = 10 「Am7→D7→Dm7→G7」 (21, 2, 14, 7)
pattern = 11 「Eb7→D7→Dm7→G7」 (3, 2, 14, 7)
pattern = 12 「Ebm7→D7→Dm7→G7」 (15, 2, 14, 7)
pattern = 13 「A7→D7→Ab7→G7」 (9, 2, 8, 7)
pattern = 14 「Am7→D7→Ab7→G7」 (21, 2, 8, 7)
pattern = 15 「Eb7→D7→Ab7→G7」 (3, 2, 8, 7)
pattern = 16 「Ebm7→D7→Ab7→G7」 (15, 2, 8, 7)
pattern = 17 「A7→D7→Abm7→G7」 (9, 2, 20, 7)
pattern = 18 「Am7→D7→Abm7→G7」 (21, 2, 20, 7)
pattern = 19 「Eb7→D7→Abm7→G7」 (3, 2, 20, 7)
pattern = 20 「Ebm7→D7→Abm7→G7」 (15, 2, 20, 7)
pattern = 21 「A7→Ab7→Dm7→G7」 (9, 8, 14, 7)
pattern = 22 「Am7→Ab7→Dm7→G7」 (21, 8, 14, 7)
pattern = 23 「Eb7→Ab7→Dm7→G7」 (3, 8, 14, 7)
pattern = 24 「Ebm7→Ab7→Dm7→G7」 (15, 8, 14, 7)
pattern = 25 「A7→Ab7→D7→G7」 (9, 8, 2, 7)
pattern = 26 「Am7→Ab7→D7→G7」 (21, 8, 2, 7)
pattern = 27 「Eb7→Ab7→D7→G7」 (3, 8, 2, 7)
pattern = 28 「Ebm7→Ab7→D7→G7」 (15, 8, 2, 7)

pattern = 29 「A7→Ab7→Abm7→G7」 (9, 8, 20, 7)
pattern = 30 「Am7→Ab7→Abm7→G7」 (21, 8, 20, 7)
pattern = 31 「Eb7→Ab7→Abm7→G7」 (3, 8, 20, 7)
pattern = 32 「Ebm7→Ab7→Abm7→G7」 (15, 8, 20, 7)

7.5. FMC3からのコード進行ルール(3)

前半2小節の後のコードからExt.Dで後半2小節の後のコードに進む場合のうち、後半2小節の後のコード(4小節目のコード)がDb7の場合は以下のようになる。

□→Ab7→◇→Db7 / □→D7→◇→Db7

◇にその直前と同じコードが入るものは除外するので、以下の6つだけが候補となる。

□→Ab7→Abm7→Db7
□→Ab7→D7→Db7
□→Ab7→Dm7→Db7
□→D7→Abm7→Db7
□→D7→Ab7→Db7
□→D7→Dm7→Db7

この□の部分(1小節目のコード)に、続くコードに対してII-VとSec.Dの各2通りが入る(p5下行と半音下行)ので、以下の24通りが全てOKとなる。

pattern = 33 「A7→D7→Dm7→Db7」 (9, 2, 14, 1)
pattern = 34 「Am7→D7→Dm7→Db7」 (21, 2, 14, 1)
pattern = 35 「Eb7→D7→Dm7→Db7」 (3, 2, 14, 1)
pattern = 36 「Ebm7→D7→Dm7→Db7」 (15, 2, 14, 1)
pattern = 37 「A7→D7→Ab7→Db7」 (9, 2, 8, 1)
pattern = 38 「Am7→D7→Ab7→Db7」 (21, 2, 8, 1)
pattern = 39 「Eb7→D7→Ab7→Db7」 (3, 2, 8, 1)
pattern = 40 「Ebm7→D7→Ab7→Db7」 (15, 2, 8, 1)
pattern = 41 「A7→D7→Abm7→Db7」 (9, 2, 20, 1)
pattern = 42 「Am7→D7→Abm7→Db7」 (21, 2, 20, 1)
pattern = 43 「Eb7→D7→Abm7→Db7」 (3, 2, 20, 1)
pattern = 44 「Ebm7→D7→Abm7→Db7」 (15, 2, 20, 1)
pattern = 45 「A7→Ab7→Dm7→Db7」 (9, 8, 14, 1)
pattern = 46 「Am7→Ab7→Dm7→Db7」 (21, 8, 14, 1)
pattern = 47 「Eb7→Ab7→Dm7→Db7」 (3, 8, 14, 1)
pattern = 48 「Ebm7→Ab7→Dm7→Db7」 (15, 8, 14, 1)
pattern = 49 「A7→Ab7→D7→Db7」 (9, 8, 2, 1)
pattern = 50 「Am7→Ab7→D7→Db7」 (21, 8, 2, 1)
pattern = 51 「Eb7→Ab7→D7→Db7」 (3, 8, 2, 1)
pattern = 52 「Ebm7→Ab7→D7→Db7」 (15, 8, 2, 1)
pattern = 53 「A7→Ab7→Abm7→Db7」 (9, 8, 20, 1)
pattern = 54 「Am7→Ab7→Abm7→Db7」 (21, 8, 20, 1)
pattern = 55 「Eb7→Ab7→Abm7→Db7」 (3, 8, 20, 1)
pattern = 56 「Ebm7→Ab7→Abm7→Db7」 (15, 8, 20, 1)

7.6. FMC3からのコード進行ルール(4)

前半2小節の後のコードからSec.Dで後半2小節の前のコードに進む場合のうち、後半2小節の前のコードがDm7の場合は以下のようになる。

□→◇→Dm7→G7 / □→◇→Dm7→Db7

◇はドミナントコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→A7→Dm7→G7
□→Eb7→Dm7→G7
□→A7→Dm7→Db7
□→Eb7→Dm7→Db7

この□の部分(1小節目のコード)に、続くコードに対してII-VとSec.Dの各2通りが入る(p5下行と半音下行)ので、以下の16通りが全てOKとなる。

pattern = 57 「E7→A7→Dm7→G7」 (4, 9, 14, 7)
pattern = 58 「Em7→A7→Dm7→G7」 (16, 9, 14, 7)
pattern = 59 「Bb7→A7→Dm7→G7」 (10, 9, 14, 7)
pattern = 60 「Bbm7→A7→Dm7→G7」 (12, 9, 14, 7)
pattern = 61 「E7→Eb7→Dm7→G7」 (4, 3, 14, 7)
pattern = 62 「Em7→Eb7→Dm7→G7」 (16, 3, 14, 7)
pattern = 63 「Bb7→Eb7→Dm7→G7」 (10, 3, 14, 7)
pattern = 64 「Bbm7→Eb7→Dm7→G7」 (22, 3, 14, 7)
pattern = 65 「E7→A7→Dm7→Db7」 (4, 9, 14, 1)
pattern = 66 「Em7→A7→Dm7→Db7」 (16, 9, 14, 1)
pattern = 67 「Bb7→A7→Dm7→Db7」 (10, 9, 14, 1)
pattern = 68 「Bbm7→A7→Dm7→Db7」 (22, 9, 14, 1)
pattern = 69 「E7→Eb7→Dm7→Db7」 (4, 3, 14, 1)
pattern = 70 「Em7→Eb7→Dm7→Db7」 (16, 3, 14, 1)
pattern = 71 「Bb7→Eb7→Dm7→Db7」 (10, 3, 14, 1)
pattern = 72 「Bbm7→Eb7→Dm7→Db7」 (22, 3, 14, 1)

7.7. FMC3からのコード進行ルール(5)

前半2小節の後のコードからSec.Dで後半2小節の前のコードに進む場合のうち、後半2小節の前のコードがD7の場合は以下のようになる。

□→◇→D7→G7 / □→◇→D7→Db7

◇はドミナントコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→A7→D7→G7
□→Eb7→D7→G7
□→A7→D7→Db7
□→Eb7→D7→Db7

この□の部分(1小節目のコード)に、続くコードに対してII-VとSec.Dの各2通りが入る(p5下行と半音下行)ので、以下の16通りが全てOKとなる。

pattern = 73 「E7→A7→D7→G7」 (4, 9, 2, 7)
pattern = 74 「Em7→A7→D7→G7」 (16, 9, 2, 7)
pattern = 75 「Bb7→A7→D7→G7」 (10, 9, 2, 7)
pattern = 76 「Bbm7→A7→D7→G7」 (22, 9, 2, 7)
pattern = 77 「E7→Eb7→D7→G7」 (4, 3, 2, 7)
pattern = 78 「Em7→Eb7→D7→G7」 (16, 3, 2, 7)
pattern = 79 「Bb7→Eb7→D7→G7」 (10, 3, 2, 7)
pattern = 80 「Bbm7→Eb7→D7→G7」 (22, 3, 2, 7)
pattern = 81 「E7→A7→D7→Db7」 (4, 9, 2, 1)
pattern = 82 「Em7→A7→D7→Db7」 (16, 9, 2, 1)
pattern = 83 「Bb7→A7→D7→Db7」 (10, 9, 2, 1)
pattern = 84 「Bbm7→A7→D7→Db7」 (22, 9, 2, 1)
pattern = 85 「E7→Eb7→D7→Db7」 (4, 3, 2, 1)
pattern = 86 「Em7→Eb7→D7→Db7」 (16, 3, 2, 1)
pattern = 87 「Bb7→Eb7→D7→Db7」 (10, 3, 2, 1)

pattern = 88 「Bbm7→Eb7→D7→Db7」 (22, 3, 2, 1)

7.8. FMC3からのコード進行ルール(6)

前半2小節の後のコードからSec.Dで後半2小節の前のコードに進む場合のうち、後半2小節の前のコードがAb7の場合は以下のようになる。

□→◇→Ab7→G7 / □→◇→Ab7→Db7

◇はドミナントコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→Eb7→Ab7→G7
□→A7→Ab7→G7
□→Eb7→Ab7→Db7
□→A7→Ab7→Db7

この□の部分(1小節目のコード)に、続くコードに対してII-VとSec.Dの各2通りが入る(p5下行と半音下行)ので、以下の16通りが全てOKとなる。

pattern = 89 「E7→A7→Ab7→G7」 (4, 9, 8, 7)
pattern = 90 「Em7→A7→Ab7→G7」 (16, 9, 8, 7)
pattern = 91 「Bb7→A7→Ab7→G7」 (10, 9, 8, 7)
pattern = 92 「Bbm7→A7→Ab7→G7」 (22, 9, 8, 7)
pattern = 93 「E7→Eb7→Ab7→G7」 (4, 3, 8, 7)
pattern = 94 「Em7→Eb7→Ab7→G7」 (16, 3, 8, 7)
pattern = 95 「Bb7→Eb7→Ab7→G7」 (10, 3, 8, 7)
pattern = 96 「Bbm7→Eb7→Ab7→G7」 (22, 3, 8, 7)
pattern = 97 「E7→A7→Ab7→Db7」 (4, 9, 8, 1)
pattern = 98 「Em7→A7→Ab7→Db7」 (16, 9, 8, 1)
pattern = 99 「Bb7→A7→Ab7→Db7」 (10, 9, 8, 1)
pattern = 100 「Bbm7→A7→Ab7→Db7」 (22, 9, 8, 1)
pattern = 101 「E7→Eb7→Ab7→Db7」 (4, 3, 8, 1)
pattern = 102 「Em7→Eb7→Ab7→Db7」 (16, 3, 8, 1)
pattern = 103 「Bb7→Eb7→Ab7→Db7」 (10, 3, 8, 1)
pattern = 104 「Bbm7→Eb7→Ab7→Db7」 (22, 3, 8, 1)

7.9. FMC3からのコード進行ルール(7)

前半2小節の後のコードからSec.Dで後半2小節の前のコードに進む場合のうち、後半2小節の前のコードがAbm7の場合は以下のようになる。

□→◇→Abm7→G7 / □→◇→Abm7→Db7

◇はドミナントコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→Eb7→Abm7→G7
□→A7→Abm7→G7
□→Eb7→Abm7→Db7
□→A7→Abm7→Db7

この□の部分(1小節目のコード)に、続くコードに対してII-VとSec.Dの各2通りが入る(p5下行と半音下行)ので、以下の16通りが全てOKとなる。

pattern = 105 「E7→A7→Abm7→G7」 (4, 9, 20, 7)
pattern = 106 「Em7→A7→Abm7→G7」 (16, 9, 20, 7)
pattern = 107 「Bb7→A7→Abm7→G7」 (10, 9, 20, 7)
pattern = 108 「Bbm7→A7→Abm7→G7」 (22, 9, 20, 7)
pattern = 109 「E7→Eb7→Abm7→G7」 (4, 3, 20, 7)
pattern = 110 「Em7→Eb7→Abm7→G7」 (16, 3, 20, 7)
pattern = 111 「Bb7→Eb7→Abm7→G7」 (10, 3, 20, 7)
pattern = 112 「Bbm7→Eb7→Abm7→G7」 (22, 3, 20, 7)
pattern = 113 「E7→A7→Abm7→Db7」 (4, 9, 20, 1)
pattern = 114 「Em7→A7→Abm7→Db7」 (16, 9, 20, 1)
pattern = 115 「Bb7→A7→Abm7→Db7」 (10, 9, 20, 1)
pattern = 116 「Bbm7→A7→Abm7→Db7」 (22, 9, 20, 1)
pattern = 117 「E7→Eb7→Abm7→Db7」 (4, 3, 20, 1)
pattern = 118 「Em7→Eb7→Abm7→Db7」 (16, 3, 20, 1)
pattern = 119 「Bb7→Eb7→Abm7→Db7」 (10, 3, 20, 1)
pattern = 120 「Bbm7→Eb7→Abm7→Db7」 (22, 3, 20, 1)

7.10. FMC3のコード進行ルールの拡張

FMC3のルールのうち「2小節パターンの、後ろのコードは**必ず**ドミナント7thコードとする」という条件は音楽的に必須なものではない。後半2小節の後ろは全体4小節の最後なのでドミナント7thコードが必須であるが、前半2小節の後ろのコードはマイナー7thコードであってもOKであり、この場合、前半2小節の前のコードは、マイナー7thコードの連続は除外しているので、ドミナント7thコードに限定される。つまり、**前半2小節を「ドミナント7thコード→マイナー7thコード」と逆順にしたパターン**のコード進行を、FMC3からの拡張として本システムに加えた。

このような前半2小節から繋がる後半2小節については、その先頭(3小節目)のコードに制限が出る。後半2小節の前(3小節目)がマイナー7thコードでは、2小節目→3小節目がマイナー7thコードの連続となり、コード進行の理由付けが出来ないので不適である(**循環コード**のみ**経験的例外**)。従って、ここで考える後半2小節は「ドミナント7thコード→ドミナント7thコード」というSec.Dになって、全体の3小節目はドミナント7thコードだけとなり、2小節目→3小節目の「マイナー7thコード→ドミナント7thコード」はII-V(またはその**裏5度**)として進行する。

そして、前半2小節の前、つまり全体の1小節目のパターンには、「1小節目のコードからSec.Dで2小節目のm7thコードに進む」というものと、「1小節目のコードからExt.Dで後半2小節の前のコードに進む」というものの2つがある。このrootは「p5下行か半音下行(裏5度進行)のいずれか」の2通りで進行する。全体の3小節目はドミナント7thコードだけなので、Key=Cとして、G7に進むのは、D7(p5下行)と、Ab7(半音下行)。Db7に進むのは、Ab7(p5下行)と、D7(半音下行)。つまり、組み合わせは以下の4通りである。

D7→G7 / Ab7→G7 / Ab7→Db7 / D7→Db7

ここから前半2小節の前、1小節目のコードの連結パターンは、「1小節目のコードからSec.Dで2小節目のマイナー7thコードに進む」か、「1小節目のコードからExt.Dで3小節目のドミナント7thコードに進む」か、のいずれかである。そこで、次項からこれらの組み合わせ方法と生成されるコード進行を示す。

7.11. FMC3から拡張したコード進行ルール(1)

1小節目のコードからSec.Dで2小節目のマイナー7thコー

ドに進む場合のうち、後半2小節の前のコードがD7の場合は以下のようになる。

□→◇→D7→G7 / □→◇→D7→Db7

◇はm7thコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→Ebm7→D7→G7
□→Am7→D7→G7
□→Ebm7→D7→Db7
□→Am7→D7→Db7

この□の部分(1小節目のコード)に、続くコードに対してSec.Dの2通りが入る(p5下行と半音下行)ので、以下の8通りが全てOKとなる。

pattern = 121 「Bb7→Ebm7→D7→G7」 (10, 15, 2, 7)
pattern = 122 「E7→Ebm7→D7→G7」 (4, 15, 2, 7)
pattern = 123 「Bb7→Am7→D7→G7」 (10, 21, 2, 7)
pattern = 124 「E7→Am7→D7→G7」 (4, 21, 2, 7)
pattern = 125 「Bb7→Ebm7→D7→Db7」 (10, 15, 2, 1)
pattern = 126 「E7→Ebm7→D7→Db7」 (4, 15, 2, 1)
pattern = 127 「Bb7→Am7→D7→Db7」 (10, 21, 2, 1)
pattern = 128 「E7→Am7→D7→Db7」 (4, 21, 2, 1)

7.12. FMC3から拡張したコード進行ルール(2)

1小節目のコードからSec.Dで2小節目のマイナー7thコードに進む場合のうち、後半2小節の前のコードがAb7の場合は以下のようになる。

□→◇→Ab7→G7 / □→◇→Ab7→Db7

◇はm7thコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→Ebm7→Ab7→G7
□→Am7→Ab7→G7
□→Ebm7→Ab7→Db7
□→Am7→Ab7→Db7

この□の部分(1小節目のコード)に、続くコードに対してSec.Dの2通りが入る(p5下行と半音下行)ので、以下の8通りが全てOKとなる。

pattern = 129 「Bb7→Ebm7→Ab7→G7」 (10, 15, 8, 7)
pattern = 130 「E7→Ebm7→Ab7→G7」 (4, 15, 8, 7)
pattern = 131 「Bb7→Am7→Ab7→G7」 (10, 21, 8, 7)
pattern = 132 「E7→Am7→Ab7→G7」 (4, 21, 8, 7)
pattern = 133 「Bb7→Ebm7→Ab7→Db7」 (10, 15, 8, 1)
pattern = 134 「E7→Ebm7→Ab7→Db7」 (4, 15, 8, 1)
pattern = 135 「Bb7→Am7→Ab7→Db7」 (10, 21, 8, 1)
pattern = 136 「E7→Am7→Ab7→Db7」 (4, 21, 8, 1)

7.13. FMC3から拡張したコード進行ルール(3)

1小節目のコードからExt.Dで3小節目のドミナント7thコードに進む場合のうち、後半2小節の前のコードがD7の場合は以下のようになる。

□→◇→D7→G7 / □→◇→D7→Db7

◇はm7thコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→Ebm7→D7→G7
□→Am7→D7→G7
□→Ebm7→D7→Db7
□→Am7→D7→Db7

この□の部分(1小節目のコード)に、1つとばした3小節目に対してExt.Dの2通りが入る(p5下行と半音下行)ので、以下の8通りが全てOKとなる。

pattern = 137 「A7→Ebm7→D7→G7」 (9, 15, 2, 7)
pattern = 138 「Eb7→Ebm7→D7→G7」 (3, 15, 2, 7)
pattern = 139 「A7→Am7→D7→G7」 (9, 21, 2, 7)
pattern = 140 「Eb7→Am7→D7→G7」 (3, 21, 2, 7)
pattern = 141 「A7→Ebm7→D7→Db7」 (9, 15, 2, 1)
pattern = 142 「Eb7→Ebm7→D7→Db7」 (3, 15, 2, 1)
pattern = 143 「A7→Am7→D7→Db7」 (9, 21, 2, 1)
pattern = 144 「Eb7→Am7→D7→Db7」 (3, 21, 2, 1)

7.14. FMC3から拡張したコード進行ルール(4)

1小節目のコードからExt.Dで3小節目のドミナント7thコードに進む場合のうち、後半2小節の前のコードがAb7の場合は以下のようになる。

□→◇→Ab7→G7 / □→◇→Ab7→Db7

◇はm7thコードに限定しているので、p5下行と半音下行で以下の4つだけが候補となる。

□→Ebm7→Ab7→G7
□→Am7→Ab7→G7
□→Ebm7→Ab7→Db7
□→Am7→Ab7→Db7

この□の部分(1小節目のコード)に、1つとばした3小節目に対してExt.Dの2通りが入る(p5下行と半音下行)ので、以下の8通りが全てOKとなる。

pattern = 145 「Eb7→Ebm7→Ab7→G7」 (3, 15, 8, 7)
pattern = 146 「A7→Ebm7→Ab7→G7」 (9, 15, 8, 7)
pattern = 147 「Eb7→Am7→Ab7→G7」 (3, 21, 8, 7)
pattern = 148 「A7→Am7→Ab7→G7」 (9, 21, 8, 7)
pattern = 149 「Eb7→Ebm7→Ab7→Db7」 (3, 15, 8, 1)
pattern = 150 「A7→Ebm7→Ab7→Db7」 (9, 15, 8, 1)
pattern = 151 「Eb7→Am7→Ab7→Db7」 (3, 21, 8, 1)
pattern = 152 「A7→Am7→Ab7→Db7」 (9, 21, 8, 1)

以上で、本システムで使用している152種類のコード進行全てについて、音楽的な解説は尽くされている。4小節の最後が必ずG7かDb7のいずれかになるというかなり限定された条件でも、ドミナントモーションを基盤とするコード進行における**音楽的ダイナミクス**で駆動される「いい感じに展開するコード進行」はこんなにあった事に驚かされる。読者はぜひ、どれでもいいので適当に選んで実際にキーボードで弾くか打ち込みソフトで鳴らして、その響きを体感して欲しいと強く希望する。専門家の耳でなくてもいいので、とにかく**音楽を愛して**聞きまくれば耳は誰でも成長して、本稿で紹介したような音楽の素晴らしさが**聞こえてくる**のである。かつて現代音楽に初めて触れた筆者はとても耐えられなくて閉口したが、ICMCコンサート等で繰り返しあれこれ聞き続けるうちに、現代音楽ならではの良さに**開眼**できた。何事も継続こそ力である。

8. リアルタイム音楽生成モジュールの実装

本システムの152種類のコード進行について解説したので、本節では第5節の図15「automatic_compose_blocks」のパッチ、すなわち実際に刻々とBGMを自動生成している心臓部に並んだ16個のサブパッチの中身について紹介する。4種類のBGMスタイル「8beat/16beat/shuffle/ballade」ごとに「**ドラム/ベース/コード/メロディー**」の4パートを生成するブロックとして分割されている。上述のコード進行が理路整然と設計されたのに対して、ここでは完全に筆者の音楽的ヒューリスティクス、つまりは個性や癖や好みの塊となっているので、「何故そうしているのか」という質問には「好きだから」「いいと思うから」というような回答しか用意できない事を了承いただきたい。

8.1. 「8beat-ドラム」生成パッチ

図19が「8beat-ドラム」生成パッチである。4小節ループは384発のmaster_counter(0-383)で回っているので1小節は96クロック、そこで基本的に8分音符のclosed-Hihatは12クロックごとに叩かれる。ただしOpen-Hihatのタイミングを抜いたり、1拍目に弱く16分音符の刻みを入れたりしている。バスドラムは4小節のそれぞれで異なる固定パターンを蹴っている。通常であれば2拍目と4拍目にだけ入るスネアドラムは、カメラ画像から与えられる「風景の複雑さ」パラメータに対応して裏拍で刻むゴーストノートの音量を変えている。クラッシュシンバルは4小節ループの先頭で必ず叩いている。

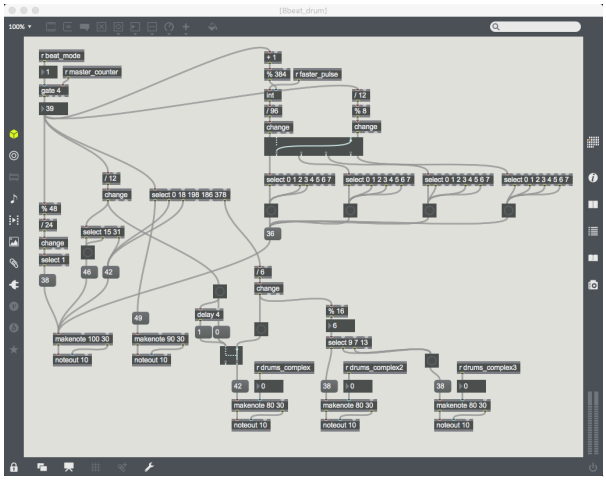

図 19 「8beat-ドラム」生成パッチ
Figure 19 8beat-Drum Generator.

8.2. 「16beat-ドラム」生成パッチ

図20が「16beat-ドラム」生成パッチである。基本的に16分音符のclosed-Hihatは6クロックごとに叩かれる。8beatとの大きな違いとして、バスドラム(16分音符の間隔で蹴るのが16beatの最大の特徴)は4種類の異なる固定パターンを用意しているが、これを固定的な順番で連ねるのでなく、4種類のパターンを**4:3:2:1**の異なる確率のランダムで選ぶことで面白い意外性を実現でき、16beatの刻みに特有の単調さを回避している。Open-Hihatやスネアドラムのゴーストノートの扱いは8beatとほぼ同様である。クラッシュシンバルは4小節ループの先頭で必ず叩いている。

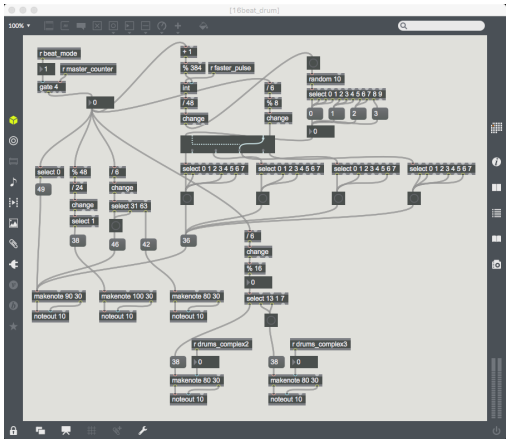

図 20 「16beat-ドラム」生成パッチ
Figure 20 16beat-Drum Generator.

8.3. 「shuffle-ドラム」生成パッチ

図21が「shuffle-ドラム」生成パッチである。shuffleというパターンは1小節を4つの3連符で分割するので基本的にclosed-Hihatは8クロックごとに叩かれる。スネアドラムでなくリムショットを使用してのんびり感を出している。バスドラム(3連符の1か3を蹴るのがshuffleの最大の特徴)は4種類の異なる固定パターンを用意しているが、これを固定的な順番で連ねるのでなく、例外的な**裏蹴り**を含む4種類のパターンを**6:3:2:1**の異なる確率のランダムで選んで単調さを回避している。クラッシュシンバルは4小節ループの先頭で必ず叩いている。

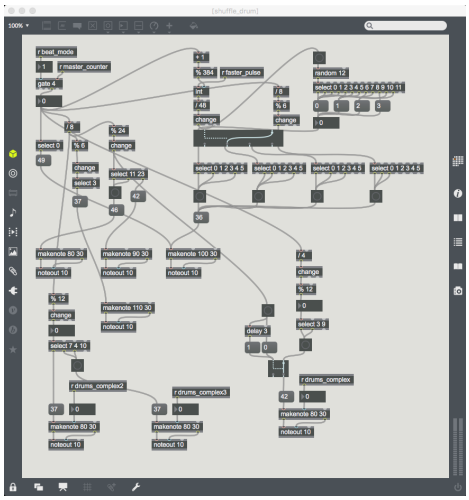

図 21「shuffle-ドラム」生成パッチ
Figure 21 shuffle-Drum Generator.

8.4. 「ballade-ドラム」生成パッチ

図22が「ballade-ドラム」生成パッチである。本システムにおいてballadeとは、shuffleと同様に1小節を4つの3連符で分割するビートであるが、こちらはスネアドラムを叩きやや速めのテンポで、closed-Hihatおよびライドシンバルを3連符の先頭だけ、つまり4分音符(24クロック)ごとにだけ叩くことでノリを演出している。バスドラムはshuffleと同様に**裏蹴り**を含む4種類の別パターンを**6:3:2:1**の異なる確率のランダムで選んで単調さを回避している。クラッシュシンバルは4小節ループの先頭で必ず叩いている。

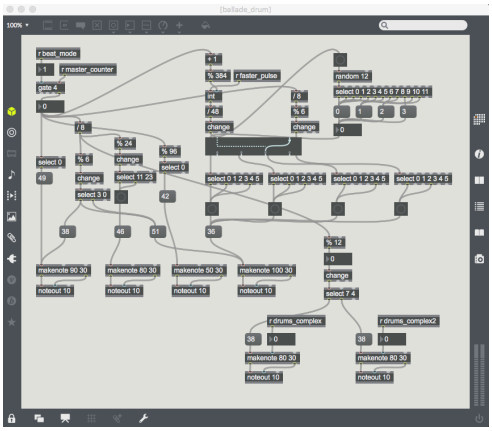

図 22 「ballade-ドラム」生成パッチ
Figure 22 ballade-Drum Generator.

8.5. 「8beat-ベース」生成パッチ

図23が「8beat-ベース」生成パッチである。8beatはシンプルに続く事がノリの命なので、ベースのフレーズのリズムはランダム要素のないシンプルな8分音符シンコペーションの固定パターンで4小節にわたって共通である。フレーズの各音(ノートはrootと5thとオクターブ上のroot)は全てアクセントの意味によって音量と音価が異なるので、異なったMIDIベロシティとdurationを設定してノートイベント化している。なお、本来8ビートであれば入らない16分音符が「master_counter%96=90」の地点にあるが、これはカメラ画像から与えられる「風景の複雑さ」パラメータに対応してベースをチョッパー気味に裏拍の7thで刻むノートのvelocityを変えて加えている。

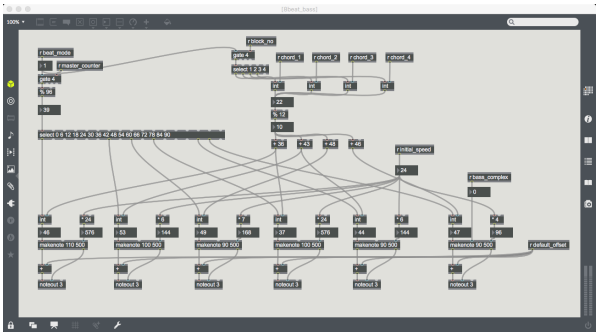

図 23 「8beat-ベース」生成パッチ
Figure 23 8beat-Bass Generator.

本システムでは4小節コード進行のパターンとしてKey=Cに対するコードを記述しているが、**車載警報音の固有ピッチ**などの関係でTonal Centerを移調する必要がある場合には、BGM生成システム全体に共通のオフセット(移調)値を設定すればよい。これが上のパッチの最終的なnoteoutの直前に共通に加算されている**default_offset**というパラメータである。パッチの起動時にはその値がサブパッチ「**param_set**」内で定数ゼロに初期化されているので、この初期化定数をプラスマイナス6以内に変更する。なお、演奏中に**default_offset**を変更すると、Note ONで発音開始した音に対応するNote OFFが来ないために音が鳴り続け(これはMIDI規格の仕様)、発音停止させるために音源システムを再起動する必要があるので、初期設定時(演奏開始前)に**のみ**設定する。

8.6. 「16beat-ベース」生成パッチ

図24が「16beat-ベース」生成パッチである。バスドラムと同様にベースのフレーズに16分音符の間隔がある事が16beatの最大の特徴であり、シンプルな16分音符シンコペーションの固定パターンを4小節にわたって共有する。フレーズの各音(ノートはrootと5thとオクターブ上のrootと7th)は全てアクセントの意味によって音量と音価が異なるので、異なったMIDIベロシティとdurationを設定してノートイベント化しているが、チョッパー気味に高音で刻むフレーズの音価を細かく調整して**味**を出している。このパターンだけ7th(短7度)が活躍するが、唯一の例外である**CM7**ではもちろんM7thに切り替えている。カメラ画像から与えられる「風景の複雑さ」パラメータに対応してベースを裏拍で刻むノートは**隠し味の長9度**である。

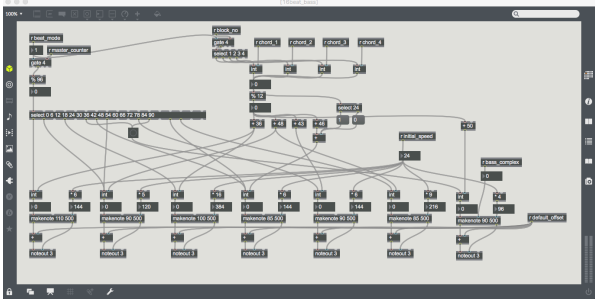

図 24 「16beat-ベース」生成パッチ
Figure 24 16beat-Bass Generator.

8.7. 「shuffle-ベース」生成パッチ

図25が「shuffle-ベース」生成パッチである。1小節を4つの3連符で分割した8クロックを3連符の単位とする。このパターンは各小節の冒頭のビートの部分のベースが強く長く(1拍+3連符2つ)伸びることでドラムのリムショットののんびり感とうまく絡むので、ほとんど小細工せずにシンプルなフレーズ(ノートはrootと5thとオクターブ上のroot)を繰り返している。

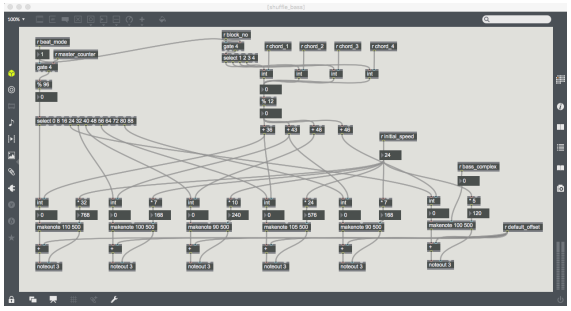

図 25 「shuffle-ベース」生成パッチ
Figure 25 shuffle-Bass Generator.

8.8. 「ballade-ベース」生成パッチ

図26が「ballade-ベース」生成パッチである。ドラムが生み出すshuffleより速めの12ビートのノリを生かすために、各小節の冒頭のビートの部分が強く長く(1拍+3連符2つ)伸びた後の細かい3連符のフレーズをスタカートに刻むようにdurationを設定しているのがポイントである。ここでもカメラ画像から与えられる「風景の複雑さ」パラメータに対応してベースを裏拍で刻むノートに**隠し味の長9度**を仕込んである。

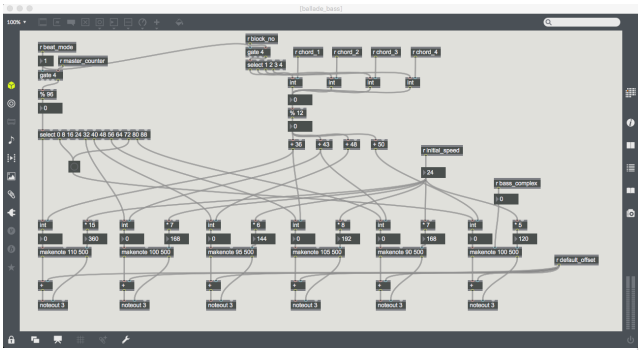

図 26 「ballade-ベース」生成パッチ
Figure 26 ballade-Bass Generator.

8.9. 「8beat-コード」生成パッチ

図27が「8beat-コード」生成パッチである。本システムでは、各小節の最初のビートは必ずシンバルとバスドラムとベースのrootが鳴るので、そこに4和音からなるコード音を同時に鳴らすと響きが濁るという経験則を生かして、FMC3と同様に「**コードは和音で刻まずアルペジオ**」という方針を継承した。そして新たなアイデアとして、4和音を鳴らす順番として6種類(1357, 1537, 5713, 7531, 7351, 3715)のパターンを用意して、4小節ループごとにランダムに選択して、その4小節内では各小節が同一パターンとなるようにして単調さを回避している。

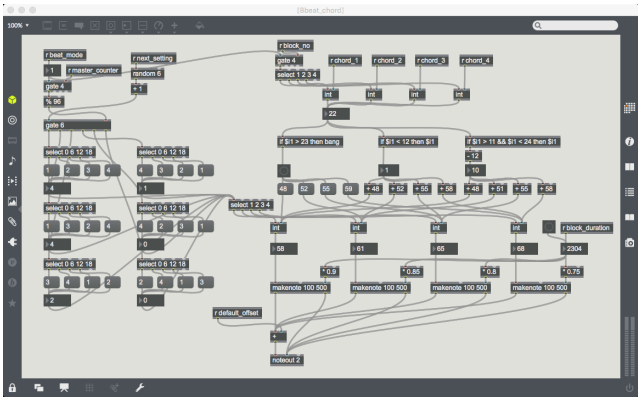

図 27 「8beat-コード」生成パッチ
Figure 27 8beat-Chord Generator.

コードパートのdefault音色は**エレピ**(MIDIプログラム番号=5)であるが、運転状況の変化に対応してコード音色を変更できる。ただしメロディーパートの音色はいつでも変更可能なのに対して、途中で変わるのが不自然となるコードパートは、音色変更要求が反映されるのが4パターンのコード進行が一周して新しくスタートする瞬間だけとしてある。音色はdefaultの5以外に「1/7/25/26/28/29」の6種類を用意した。この切り替えは内部的にはchord_changeにbangを送るとランダムに選択するが、6種類の音色候補に対してdefault=5の出現確率を約3倍に設定している。

8.10. 「16beat-コード」生成パッチ

図28が「16beat-コード」生成パッチである。コードパートについてはほとんど図27の「8beat-コード」と違わないが、アルペジオの各音を伸ばすdurationの演算部分をわずかに短めにして16beatの持つ**ソリッド感**を反映させる、というような味付けを施している。

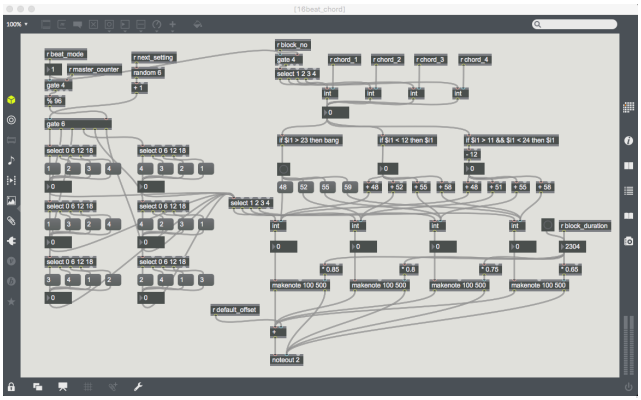

図 28 「16beat-コード」生成パッチ
Figure 28 16beat-Chord Generator.

8.11. 「shuffle-コード」生成パッチ

図29が「shuffle-コード」生成パッチである。1小節を4つの3連符で分割した8クロックを3連符の単位としている以外、コードパートについてはほとんど図27の「8beat-コード」と違わない。

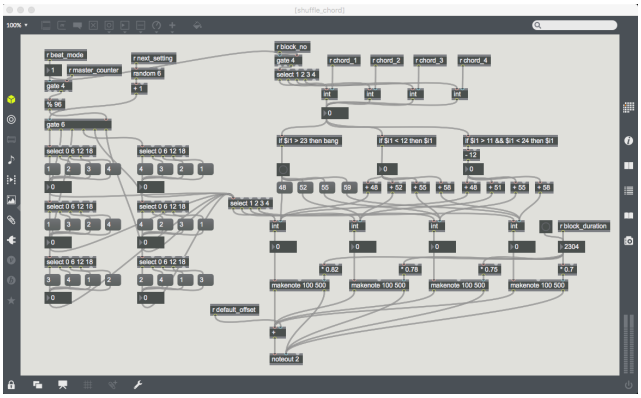

図 29 「shuffle-コード」生成パッチ
Figure 29 shuffle-Chord Generator.

8.12. 「ballade-コード」生成パッチ

図30が「ballade-コード」生成パッチである。shuffleと同様に1小節を4つの3連符で分割した8クロックを3連符の単位としている以外、コードパートについてはほとんど図27の「8beat-コード」と違わない。ただしテンポがshuffleよりも速い(ノリが良い)ので、アルペジオの各音を伸ばすdurationの演算部分をわずかに短めにして**歯ぎれ良さ**を演出した。

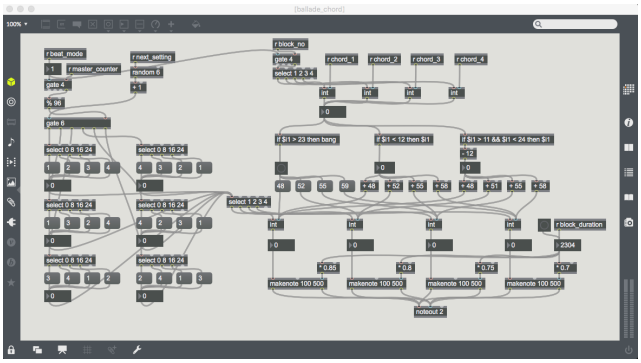

図 30 「ballade-コード」生成パッチ
Figure 30 ballade-Chord Generator.

8.13. 「8beat-メロディー」生成パッチ

図31が「8beat-メロディー」生成パッチである。本システムにおける「メロディー」パートという名称は便宜的なものであり、「まずメロディーを作り、そこにコード等のバッキングのアレンジを加える」という古典的な作曲スタイルとは**真逆**なので誤解なきよう注意されたい。本システムではまず最初に152種類の4小節コード進行から選び、ここにドラム/ベース/コードによってバッキングのスタイルが設定されて、そこに調的枠組みを壊さない範囲でいわば**アドリブソロとしての単音フレーズ**を追加するパートの事を便宜的に「メロディー」パートと呼んでいる。

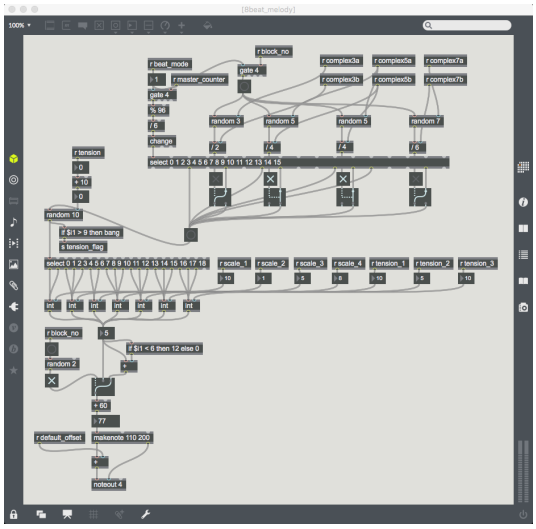

図 31 「8beat-メロディー」生成パッチ
Figure 31 8beat-Melody Generator.

4種類のBGMスタイル「8beat/16beat/shuffle/ballade」のいずれにおいても、「メロディー」パートは全て同じコンセプトで設計しているのでここで代表して解説すると、(擬似)メロディーにおいては「**音階(高さ)**」(空間的要素)と「**リズム**」(時間的要素)の両方とも重要である。このうち**音階(高さ)**については「**5.3メインBGM生成モジュールパッチの概要**」のところで詳解したように、その小節で鳴っているコードに対して、「**コード構成音4音**」+「available tension notes 1-3音」という候補音から**重み付けしたランダム**(→8.17で解説)によって選ばれた音から構成されている。

自動作曲システムの研究においてメロディーを生成する際にマルコフ連鎖などでよくある「直前の音からの**音高**移動方向と**音高**移動幅」という概念は本システムにおいては全く意味を持たない。後述するようにリズム生成も**重み付けしたランダム**(→そのタイミングでの発音の有無)なので直前の音や直後の音という時間的概念は捨象されている。むしろランダムの結果「**同じ高さの音がずっと連続する**」ような(擬似)メロディーが生成されることがあり、これは「**飾りじゃないのよ涙は**」(**井上陽水**)・「**柊**」(**Do As Infinity**)・**吉田拓郎**の数々の曲、など多くの名曲にある美味しいメロディー要素の一つであり、前の音から上下いずれかに動かないといけないようなルールのメロディー生成アルゴリズムは大切なものを欠いている。

リズムの生成について図31の「8beat-メロディー」生成パッチを眺めてみると、1小節を16分割したタイミング(0-15)のうち、「**2,6,10,12**」については**必ず発音**し、「**4,7,11,15**」についてはそれぞれ別個のパラメータで重み付けしたランダムで**発音するかしないか**を決定し、残りのタイミングでは**発音しない**。必ず発音するタイミングは全て8分音符の位置である事から**8beat**の枠組みをキープしつつ、**発音するかしないか**のタイミングの細かく裏拍で**16beat**的に刻むところで**ノリと緊張感**を生み出している。

8.14. 「16beat-メロディー」生成パッチ

図32が「16beat-メロディー」生成パッチである。**音階(高さ)**については前述の8beatと同様なので**リズムの生成**について調べてみると、1小節を16分割したタイミング(0-15)のうち、「**1,2,11,12**」については**必ず発音**し、「**5,9,10,14,15**」についてはまた別のパラメータで重み付けしたランダムで**発音するかしないか**を決定し、残りのタイミングでは**発音しない**。8beatとの違いは最初の2音にあり、16分休符に続いてベースと同様の**16beat**間隔で連続する冒頭リズムによって、このスタイルを確定させている。

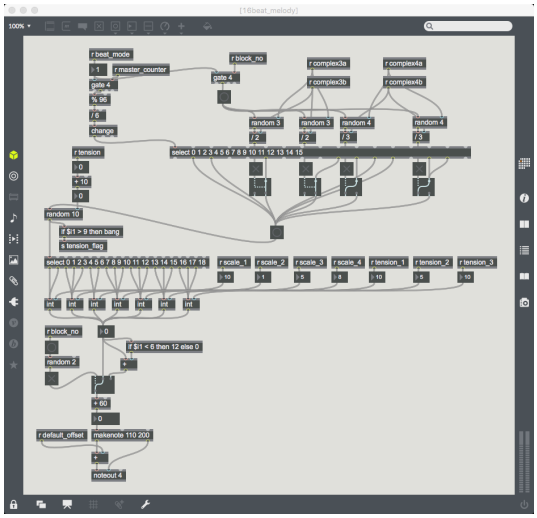

図 32 「16beat-メロディー」生成パッチ
Figure 32 16beat-Melody Generator.

8.15. 「shuffle-メロディー」生成パッチ

図33が「shuffle-メロディー」生成パッチである。**音階(高さ)**については前述の8beatと同様なので**リズムの生成**について調べてみると、1小節を12分割したタイミング(0-11)のうち、「**1,2,5,6**」については**必ず発音**し、「**3,7,9,11**」についてはまた別個のパラメータで重み付けしたランダムで**発音するかしないか**を決定し、残りのタイミングでは**発音しない**。ここでも最初の3連符の1音目は発音せず(休符)、続く3連符の2-3音が連続する冒頭リズムによって、shuffleのスタイルを確定させている。

本システムで全てのスタイルの(擬似)メロディーにおいて「**休符から始める**」というルールを徹底したのには二つの理由がある。最初のビートはバスドラムやコードパート第1音が鳴っているので重複の濁りを避けて敢えて抜いているというのがその第一、そして**ブレスタイム**のように感じることで息苦しさを避ける意味が第二である。ゲーム音楽などを打ち込む人(音楽演奏経験が無い)には、メロディーとして一切の休符が無い機関銃のようなフレーズを打ち込

む作曲を散見するが、これは音楽経験者からすれば非常に違和感のある**息苦しい音楽**となってあまり好ましくない。**「ボカロ音楽」の不自然さ**の原因の一つもこれであり、生身の人間であれば酸欠になるので歌えないようなメロディーをボカロに歌わせ(打ち込みなのでこれは容易)、カラオケではそのような非人間的なボカロ曲を人間が苦労して真似ているという滑稽な風景はちょっといかがなものか。

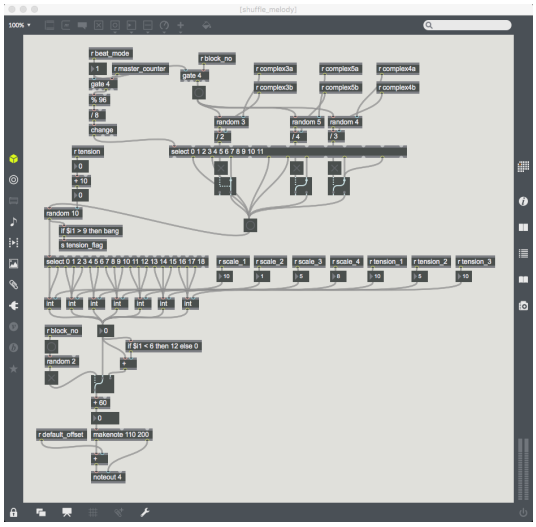

図 33 「shuffle-メロディー」生成パッチ
Figure 33 shuffle-MelodyGenerator.

8.16. 「ballade-メロディー」生成パッチ

図34が「ballade-メロディー」生成パッチである。**音階(高さ)**については前述の8beatと同様なので**リズムの生成**について調べてみると、1小節を12分割したタイミング(0-11)のうち、「2, 5, 6」については**必ず発音**し、「3, 7, 8, 10」についてはまた別個のパラメータで重み付けしたランダムで**発音するかしないか**を決定し、残りのタイミングでは**発音しない**。shuffleと似ているがアップテンポで**裏蹴り**がノリになっているballadeでは、メロディーの冒頭の休符を3連符の1-2音と長くとって3音目から始まることで**ノリ/タメ**として生きている。後半のタイミングもshuffleと違えて**早めに突っ込んでいる**ところが、まさに音楽的ヒューリスティクスの反映なのである。

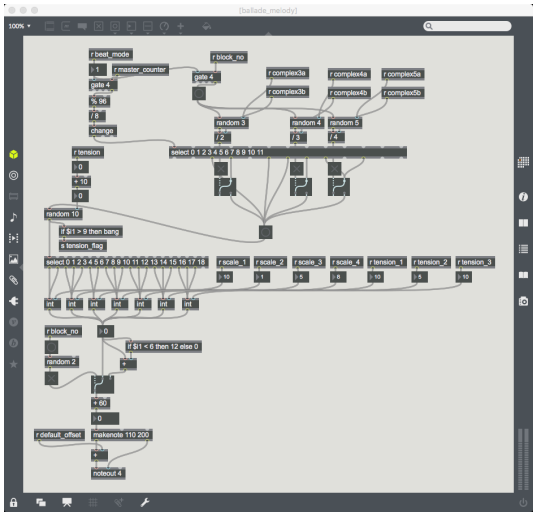

図 34 「ballade-メロディー」生成パッチ
Figure 34 ballade-Melody Generator.

8.17. メロディーノート重み付け生成パッチ

図35が「parameter_mapping」サブパッチ内にある「メロディーノート重み付け」生成パッチである。ここでは耳で聞いて試行錯誤しながら、ドラムのゴーストノートやハイハット分割などのバランス、そしてそれぞれのスタイルごとにメロディーの中で**発音するかしないか**を設定するタイミングのランダムの重み付けパラメータを経験的に設定している。自動車のセンサ情報から「**音色の変更**」のように直接に音楽的要素の生成に反映させるのでなく、このように**ランダム生成確率の重み付け**として間接的に反映させているので、[音楽的経験が豊富でない/微妙な差を聞き分ける耳が育っていない]豊田中研の研究者にはなかなかこの違いを理解してもらえなかったが、筆者としてはここはこだわりの部分であった。

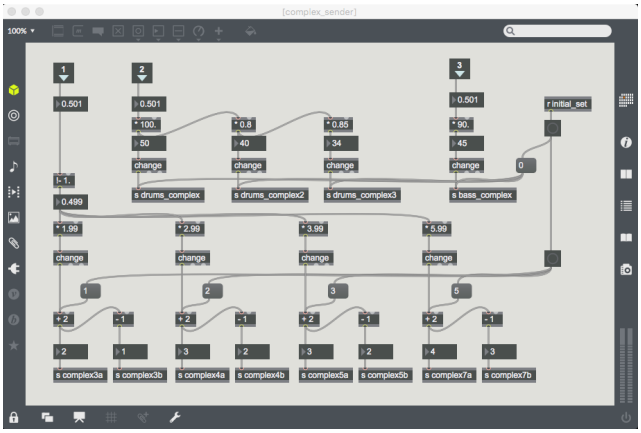

図 35 「メロディーノート重み付け」生成パッチ
Figure 35 Melody Notes Random-Weight.

9. パラメータ変換用中間パッチの概要

9.1. 車載センサ情報として受け取る情報

ここで、図9の「車載センサ情報再生モジュールパッチcar_data_player.maxpat」からの情報を受け取ってリアルタイムBGM生成モジュールに引き継ぐ図8の「パラメータ変換用中間パッチparameter_convert010.maxpat」について詳解する。図9のパッチcar_data_player.maxpatの出力として内部的にMaxパラメータとして「send」されているのは以下の7種類の情報である。

camera_data - カメラ動画
longitude_data - GPS情報「経度」
latitude_data - GPS情報「緯度」
souda_data - ステアリング(ハンドル)回転角
brake_data - ブレーキペダルの踏み込み量
accel_data - アクセルペダルの踏み込み量
speed_data - 速度

本稿ではこれ以上、上流に位置している図9のパッチcar_data_player.maxpatについては解説しない。その理由は、このパッチが取得する車載センサデータ処理の部分(実際にCANやカメラからの信号を受け取り変換送出する部分)は筆者でなく豊田中研側が実験車において開発しているためであり、共同研究の取り決めにより、筆者が学会発表するのは筆者が担当するMaxプログラミング部分に限定しているからである。

9.2. カメラ画像情報の処理

図8のparameter_convert010.maxpatには、上記の7種類の情報がそのまま「receive」入力情報として定義されている。ここではまず左上で、「camera_data:**カメラ動画**」のjitter(movie)情報に対して、lcdオブジェクトに可変の太さのペンで**マスク画像**を描画する。これはカメラ画面の下部にある実験車のボンネット部分を黒く塗り潰すマスクであり、ボンネットに周囲の光が反射して画像データに不自然なノイズが加わることを回避するためのものである。

このカメラ画像出力は「lightblue」(青空を検出)「orange」(朝陽/夕陽など)「green」(植物の緑)という3種類の色彩抽出パラメータとして、カメラ画像の風景のおよその傾向を出力する。この「color_change_detector」サブパッチでは、「lightblue/orange/green」(0.0～1.0)の3種類の画素平均強度(時間的な平滑化も施している)に対して、内部的に以下のアルゴリズムで簡易的な変化イベント検出を行い、出力として0～7の整数値color_changeを得ている。厳密には「<」の下にはイコールが付く。

```
lightblue > orange  かつ orange > green  かつ green > lightblue
        → color_change = 7
lightblue > orange  かつ orange > green  かつ green < lightblue
        → color_change = 6
lightblue > orange  かつ orange < green  かつ green > lightblue
        → color_change = 5
lightblue > orange  かつ orange < green  かつ green < lightblue
        → color_change = 4
lightblue < orange  かつ orange > green  かつ green > lightblue
        → color_change = 3
lightblue < orange  かつ orange > green  かつ green < lightblue
        → color_change = 2
lightblue < orange  かつ orange < green  かつ green > lightblue
        → color_change = 1
lightblue < orange  かつ orange < green  かつ green < lightblue
        → color_change = 0
```

また、風景画面をモノクロ化した上で**画像の複雑さ**を抽出するモジュールを豊田中研が開発したので(詳細は省略)、この出力も音楽情報生成に活用するためのパラメータ「complex_data」として出力している。

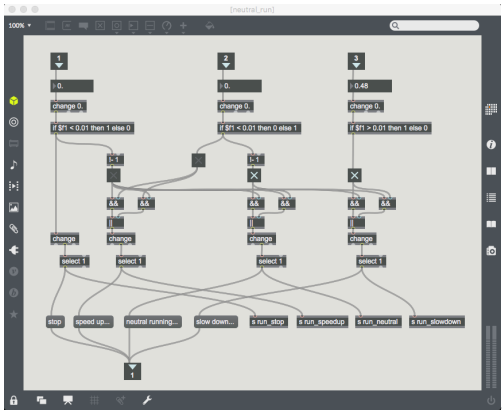

図 36 「走行状況推定」生成パッチ
Figure 36 natural_run.

「brake_data」と「accel_data」と「speed_data」という3種類の運転情報からは、図36の「natural_run」という走行状況推定パッチにおいて、「stop」「speed up」「neutral running」「slow down」という4種類の走行状況推定結果がパラメータとして出力される。これは車載センサそれぞれからの生データよりも、乗っている人間が体感するのは「停止してきた」とか「動き出した」などの**状況**だからである。例えば停止に伴って音楽をストップすることなく自然にメロディパートの音量が小さくなりつつもドラムとベースは継続している方が自然であるとか、信号で完全に停止していたところから動き出す時に各パートの音色を切り替えることで**自然な発車**を感じる、などの演出に効果を確認できた。

9.3. カメラ画像情報の処理

図8のparameter_convert010.maxpatからは、以下の情報が「send」出力情報としてMaxパラメータ出力されて、後述する内部マッピングブロックや、後段のBGM自動生成システムで利用される。既に解説されているものが大部分なので詳細な個別解説は省略する。

```
complex_data
beat_change
run_stop
run_speedup
run_neutral
run_slowdown
color_change
chords_range
addAm7_range
melody_range
tension_range
melody_exp
drums_exp
bass_exp
total_volume
melody_volume
chord_change
melody_change
```

図8のparameter_convert010.maxpatの画面の中段にある巨大な選択スイッチは、センサ情報に対して音源モジュールのパラメータをどのようにマッピングするか、という5種類の**mappingサンプル**を切り替えるためのものである。画面中段左端の「Mapping Pattern [1-5]」という部分に**「1」・「2」・「3」・「4」・「5」**というボタンがあり、これをクリックするとその番号のマッピングパターンに切り替わる。それぞれの**mappingサンプル**の内容(パラメータの対応関係アルゴリズム)は、その色のついた枠内のサブパッチを叩いて開くことで参照/改編できる。

なお「mapping_5」は**デバッグ/開発用のテンプレート**であり、叩いて中身を見てみればわかるように、受け取ったパラメータを展開するだけで、出力パラメータには何も接続されていないので、音源ブロックには**新たなパラメータは何も送出されない**。この「mapping_5」を選択しておけば、parameter_convert010.maxpatの下段にある多数のスライダーオブジェクトやボタン等を操作してパラメータを直接に変化させて、音源ブロックの動作を確認できる。

9.4. 周囲環境情報とのマッピング例(1)「mapping_1」

図37のmapping_1は、画像認識センサの**複雑さパラメータ**への反応を重視したセッティング例である。届いたcomplex_dataパラメータに対して、まず「コード進行のバリエーション」「**CM7→Am7置換**の重み付け」「メロディー音色切り替えのバリエーション」「メロディーのテンショ

ン含有率」の4つにそのままパラメータを送っている。

mapping_1ではさらにcomplex_dataパラメータに対して、「t01」というテーブルを参照し、グラフィカルに設定できる特性カーブを参照したパラメータを「メロディーのオブリガート生成確率」として送っている。この「t01」というテーブルはインスペクタ「Save Date with Patcher」ONなので、パッチとともにテーブル特性が保存される。パッチ実行モードで「t01」を叩くとテーブルが出てくるので、内容を自由に変更できる。同様に「t02」というテーブルを参照し、特性カーブを参照したパラメータを「ドラムパートのオブリガートの音量」として送っている。また「t03」というテーブルを参照し、特性カーブを参照したパラメータを「ベースパートのオブリガートの音量」として送っている。

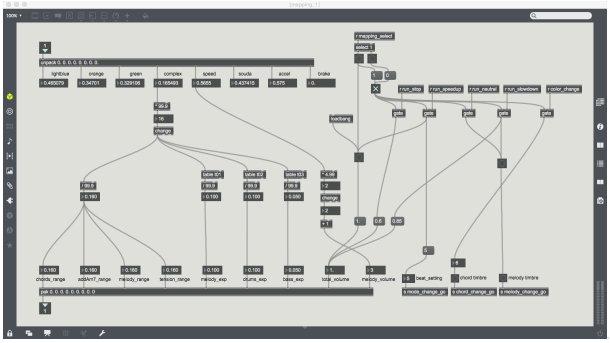

図 37 「マッピング例」(1)
Figure 37 mapping example (1).

mapping_1ではさらに、クルマのspeedパラメータを5種類の「メロディー音量変化」グループに割り当てている。この「メロディー音量変化」の値が小さい(速度が小さい)時には、メロディーパートの音量が小さく変化し、速度が大きいレンジではメロディーパートが他パートよりも大きなレベルになる。

また自動生成される音楽全体のtotal volumeに対して、(1)このマッピングが呼ばれた状態では最大値に、(2)運転状態が「**停止**」になると音量が60%に、(3)「**加速状態**」になると音量が100%に、(4)「**スローダウン**」になると音量が85%に、それぞれ**3秒間かけて次第に変化**する。さらに「加速状態」になったイベントで、4種類の音楽スタイルからランダムに選ぶトリガを行う。ランダムなので確率25%で「変化せず」という事もある。

mapping_1ではさらに、上記のカラー画像認識モジュールの簡易的な変化イベント検出の結果として、8種類のデータcolor_change(0～7)に変化があった時に、コード伴奏パートの音色変化イベントを送っている。ただしメロディーパートの音色変更はいつでも可能なのに対して、コードパートの音色変更要求が反映されるのは4パターンのコード進行が一周して新しくスタートする瞬間だけである。さらに、mapping_1では運転状態が「スローダウン」になった瞬間にメロディー音色の変化イベントを送っている。これに対してパラメータmelody_range(0.0～1.0)で重み付けされた確率で8種類の音色候補からメロディー音色がランダムに選ばれる。

9.5. 周囲環境情報とのマッピング例(2)「mapping_2」

図38のmapping_2は、画像認識センサからの「複雑さ」complex_dataとカラー画像認識モジュールのイベント検出を一切使わない代わりに、**運転状況**に対応した色々なバリエーションをあれこれ盛り込んでみたセッティング例である。mapping_2が選択されるとまず、「コード進行のバリエーション」(→0.0)・「**CM7→Am7置換**の重み付け」(→0.8)・「メロディー音色切り替えのバリエーション」(→0.8)・「メロディーのテンション含有率」(→0.5)の4つにそれぞれ定数をセットして、これらのパラメータは運転状況で変化しない。

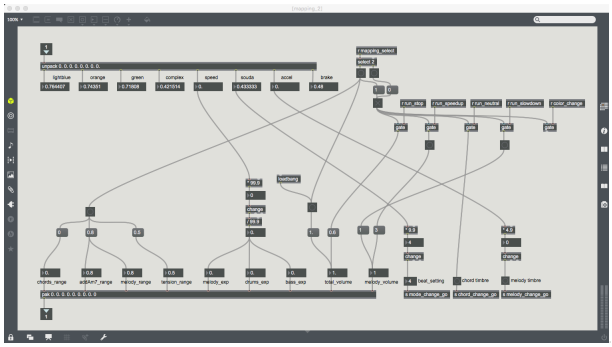

図 38 「マッピング例」(2)
Figure 38 mapping example (2).

mapping_2ではまず、クルマのspeedパラメータ(実数値)を100ステップの分解能に量子化した上で、「メロディーのオブリガート生成確率」「ドラムパートのオブリガートの音量」「ベースパートのオブリガートの音量」の3つに共通の値として送っている。つまり、車速の上昇に応じて単純に**音楽ニュアンスの複雑さ**が上がるようにした。

さらに自動生成される音楽全体のtotal volumeに対して、(1)このマッピングが呼ばれた状態では最大値に、(2)運転状態が「停止」になると音量が60%に、それぞれ3秒間かけて次第に変化する。また「加速状態」になったイベントでメロディーパートが他パートよりも大きなレベルに変化し、「スローダウン」になったイベントでメロディーパートが他パートよりも小さなレベルに変化する。

さらに「ステアリング(操舵)」(0.0～1.0)パラメータを10ステップの分解能に量子化した上で、その値が変化したというイベントで、4種類の音楽スタイルからランダムに選ぶトリガを行う。ランダムなので確率25%で「変化せず」という事もある。つまりこのセッティングでは、ハンドルを(ある程度)回さない時には音楽スタイルは変化せずに続くことになる。

そして「**ニュートラル/エンジンブレーキ**走行状態」、すなわち車速がゼロでなくかつアクセルを離して**惰性走行**、という状態になったイベント検出の結果として、コード伴奏パートの音色変化イベントを送る。ただしコードパートの音色変更要求が反映されるのは、コード進行が一周して次の4パターンの新しくスタートする瞬間だけである。

さらに「アクセル」(0.0～1.0)パラメータを5ステップの分解能に量子化した上で、その値が変化したというイベントで、メロディー音色の変化イベントを送る。これに対

して「0.8」に初期設定されたパラメータmelody_rangeで重み付けされた確率で8種類の音色候補からメロディー音色がランダムに選ばれる。

9.6. 周囲環境情報とのマッピング例(3)「mapping_3」

図39のmapping_3は、画像認識センサのカラー画像認識モジュールのイベント検出結果として、3種類のカラーフィルタのパラメータのうちもっとも優勢なものを抽出して、以下のようにランダムでなく4種類の音楽スタイル(**8beatモード/16beatモード/Shuffeモード/Balladeモード**)を固定的にトリガする、という部分に最大の特徴がある。これは将来的にカメラ画像から**外部の風景情報を抽出**するような画像認識アルゴリズムが登場した場合の参考となる。

「lightblue」が「orange」と「green」の両方に対して大きい
(lightblue優勢) → 「8beatモード」

「green」が「lightblue」と「orange」の両方に対して大きい
(green優勢) → 「16beatモード」

「orange」が「lightblue」と「green」の両方に対して大きい
(orange優勢) → 「Balladeモード」

それ以外の場合で「スローダウン」または「停止」のとき
→ 「Shuffeモード」

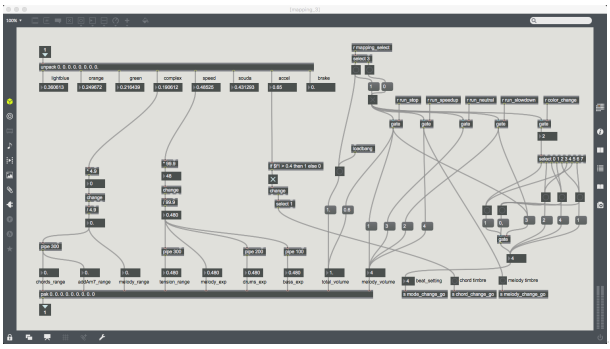

図 39 「マッピング例」(3)
Figure 39 mapping example (3).

さらに、画像認識センサからのcomplex_dataパラメータ(実数値)を5ステップの分解能に量子化した上で、遅延ゼロで「メロディー音色切り替えのバリエーション」パラメータ(0.0～1.0)として送り、さらに300msecの遅延を持たせて「コード進行のバリエーション」および「**CM7→Am7置換**の重み付け」パラメータ(0.0～1.0)として送っている。またクルマのspeedパラメータ(実数値)を100ステップの分解能に量子化した上で、遅延ゼロで「メロディーのオブリガート生成確率」パラメータ(0.0～1.0)として送り、さらに100msecの遅延を持たせて「ベースパートのオブリガートの音量」パラメータ(0.0～1.0)として送り、別途に200msecの遅延を持たせて「ドラムパートのオブリガートの音量」パラメータ(0.0～1.0)として送り、さらに別途に300msecの遅延を持たせて「メロディーのテンション含有率」パラメータ(0.0～1.0)として送ってみた。

mapping_3ではさらに、自動生成される音楽全体のtotal volumeに対して、(1)このマッピングが呼ばれた状態では最大値に、(2)運転状態が「停止」になると音量が60%に、それぞれ3秒間かけて次第に変化するよう送っている。

また「加速状態」になったイベントでメロディーパートが他パートよりも大きなレベルに変化し、「ニュートラル/エンジンブレーキ走行状態」ではそれよりやや小さめのレベルに変化し、「スローダウン」になったイベントでメロディーパートが他パートよりも小さなレベルに変化し、「ストップ」となったイベントでメロディーパートの音量を他に比べてかなり低下させる、という処理を試してみた。

mapping_3ではさらに、accelの踏み込みパラメータにaccel=0.4という閾値を設定して、これを越えたというイベントに対してコード伴奏パートの音色変化イベントを送る。ただしコードパートの音色変更要求が反映されるのは、コード進行が一周して次の4パターンの新しくスタートする瞬間だけである。また「加速状態」になったイベントでメロディー音色の変化イベントを送る。これに対してcomplex_dataパラメータに応じて設定されたパラメータmelody_rangeで重み付けされた確率で8種類の音色候補からメロディー音色がランダムに選ばれる。

9.7. 周囲環境情報とのマッピング例(4)「mapping_4」

本稿の5.3の中で、パラメータinitial_speedによって音楽のdefaultテンポは変更可能であるが、音楽にはスタイルごとに最適テンポがあるので筆者の立場としては安易にこの値を変更しないことを推奨すると書いた。これに対して図40のmapping_4は、この個人的な禁則を破って、「**運転状況に対応してテンポが変化するという仕様を体験してみる**」サンプルとして敢えて盛り込んでみた。パッチ内に赤い色で示したdo_not_change_music_tempoというパラメータがそれで、speedデータを100ステップの分解能に量子化/スケーリングして送る。AutoComposer020.maxpat内のmode_settingサブパッチ内でこのパラメータを受けて、「Mapping Pattern [1-5]」で「4」が選ばれたときだけ、default設定されたシステムクロックの値をspeedデータに対応して増減させるようにしている。このサンプルのように生成される音楽(クルマのスピードに対応してBGM生成される音楽のテンポが速くなったり遅くなったりを繰り返す)がいかに**不自然**であるかは、さすがに音楽の専門家でない豊田中研の研究者でも実験車で十分に実感できたようで、それ以降「テンポ漸次変化」案は廃棄処分となった。

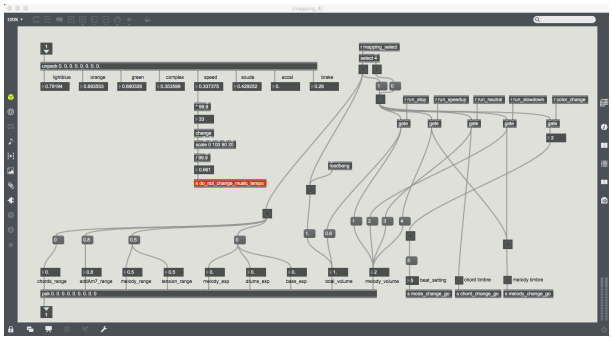

図 40 「マッピング例」(4)
Figure 40 mapping example (4).

mapping_4が選択されるとまず、「コード進行のバリエーション」(→0.0)「**CM7→Am7置換**の重み付け」(→0.8)「メロディー音色切り替えのバリエーション」(→0.5)「メロディーのテンション含有率」(→0.5)の4つ、さらに「メ

ロディーのオブリガート生成確率」「ドラムパートのオブリガートの音量」「ベースパートのオブリガートの音量」の3つに全てゼロの定数をセットして、これらのパラメータは運転状況では変化しない。そして自動生成される音楽全体のtotal volumeに対して、(1)このマッピングが呼ばれた状態では最大値に、(2)運転状態が「停止」になると音量が60%に、それぞれ3秒間かけて次第に変化する。

mapping_4ではさらに「加速状態」/「減速状態」になったイベントでメロディーパートが他パートよりも大きな/小さなレベルに変化するとともに、メロディー音色の変化イベントを送る。これに対して上述のように「0.0」に設定されたパラメータmelody_rangeで重み付けされた確率(defaultのビブラフォンの確率が最大)で8種類の音色候補からメロディー音色がランダムに選ばれる。また上記のカラー画像認識モジュールの簡易的な変化イベント検出の結果として、8種類のデータcolor_change(0～7)に変化があった時に、4種類の音楽スタイルからランダムに選ぶトリガを行う。ランダムなので確率25%で「変化せず」という事もある。また「ニュートラル/エンジンブレーキ走行状態」、すなわち車速がゼロでなくかつアクセルを離して惰性走行、という状態になったイベント検出の結果として、メロディーパートの音量を加速時よりやや小さな値に変化させるとともに、コード伴奏パートの音色変化イベントを送る。ただしコードパートの音色変更要求が反映されるのは、コード進行が一周して次の4パターンの新しくスタートする瞬間だけである。

9.8. 周囲環境情報とのマッピング例(5)「mapping_5」

図41のmapping_5は、上述のようにデバッグ/開発用のテンプレートであり、叩いて中身を見てみればわかるように、受け取ったパラメータを展開するだけで、出力パラメータには何も接続されていないので、音源ブロックには演奏状態やセンサ情報に対応した新たなパラメータは何も送出されない。このサブパッチをduplicateして新しいマッピングパターンを実験的に増設し、多くの被験者の好き嫌いを反映していけば、より有効な自動BGM生成パラメータのマッピングを発見/実装していけることになる。

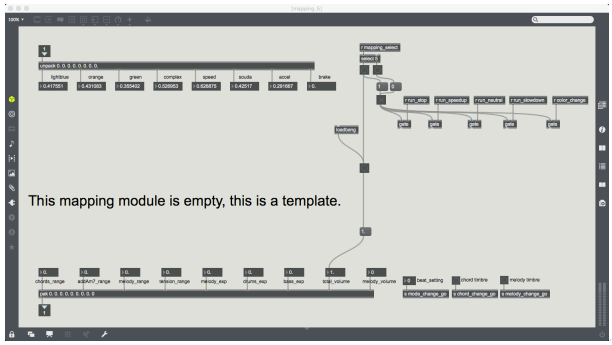

図 41 「マッピング例」(5)
Figure 41 mapping example (5).

10. 路上実験と評価/検討

研究の進展とともに、長久手の豊田中研から周辺のいろいろな状況(市街地、山間部、農村、高速道路など)で実験車が走行して得られた多種多量な記録データが筆者に届いてきたが、これは共同研究において豊田中研が取得した成果であり公表することは出来ない。筆者は終盤で実験車に実際に試験乗車も体験したが、市販ハイブリッド車をベースに図6のイラストにあるような未来系にフルカスタム改造した**素晴らしい実験車**の詳細も、残念ながらここに書くことは出来ない(ひたすら想像を逞しくして欲しい)。

およそ音楽に関する感情表現というのは個人差があるものなので、囲碁や将棋の勝ち負けのようにルール定義できず、この路上実験と評価検討の段階でも、双方の意見は完全には噛み合わなかった。本稿のあちこちに筆者が音楽的に「**いい感じに出来た**」と記述してきたポイントの大部分は、音楽の専門家でない豊田中研の研究者にはなかなか気付いてもらえず理解も得られず、一方で過度に明らかな変化(メロディーやコードパートの**音色切り替え**)に遭遇して喜んでもらうと、ちょっと複雑な心境にもなった。

大枠としてまず、本システムの自動生成BGMについて、**快適**であり適度に変化して**飽きない**、という点では妥当な評価を得たが、これは音楽ヒューリスティクスを駆使しているのでまぁ当然と言える。問題はその「**変化**」についてのリクエストと解決についての部分である。共同研究プロジェクトの初期段階でのフリーディスカッションにおいて両者が想定したイメージは、例えば「トンネルを抜ければ雪国になるとGPSを利用して予感させる」「周囲が海辺と青い空に晴れ渡る」「都会のコンクリートジャングルのような冷徹さ」「山間部を走行して森林浴のように緑を感じる」などど夢想していたのだが、最終的にはこれは**ちっとも実現できなかったね**、という感想であった。まぁ当然と言えば当然である。これは**ロマン派の標題音楽**の世界であり、本システムが生成する金太郎飴のようなdanceableな音楽でできるものであれば筆者もぜひ教えて欲しい。

ここに豊田中央研究所から静岡文化芸術大学に対して提出された共同研究の実績報告書をそのまま記載することは出来ないが、およその概要としては「クルマの周囲環境を感じさせながら一般的に聞きやすいアルゴリズムを作る」という**目的**に対して、「カメラ情報とCAN情報からコード進行に基づいた音楽を生成する」という**方法**をとり、結果として「際限なく連続する聞きやすい音楽アルゴリズムを開発することができた」・「しかし**周囲環境を感じ取れるまでには至らなかった**」・「プログラムと解説により豊田中研は音楽理論と自動作曲手法を学べた」という**結果**が得られたことになっている。筆者としてはSUACでの受託研究/共同研究は6例目となったが、なかなか面白い体験であったと感謝している。

11. おわりに

豊田中央研究所との共同研究の取り決めにより、成果物として筆者が納品した本システムの**Maxパッチは非公開**(提供不可)である。ただし本稿に載せた各Maxパッチ等の鮮明なスクリーンショットは[28]にあるので実装上のテクニックは解析可能である。筆者から豊田中研に最終試作版を2016年3月に提出した後に、豊田中研が特許出願の可能性

を検討する期間として半年以上、筆者の学界発表を控えていたので、完成/完了から約1年後の音楽情報科学研究会での発表となった次第である。もし本研究を発展させる応用とか本研究の詳細を検討したい場合には、静岡文化芸術大学の産学官連携プログラム[29]に従って「**共同研究**」ないし「**受託研究**」の枠組みで別途に契約することで対応可能である。ただし筆者は技術士[情報工学部門/電気電子部門]であり、技術士倫理規定から同業他社[自動車業界]から同一テーマでのofferはお断りする可能性も高い。現状、永遠のテーマとも言える自動作曲については、個人的にはまた10年ぐらい塩漬けにしたいという気持ちが強い。

謝辞

本研究にあたって、共同研究として実験/試作/検討を進めた、**藤枝延維/赤井良行/宇田尚典**(豊田中央研究所ヒューマンサイエンス研究領域/感覚拡張プログラム)の各氏に謝意を表する。

参考文献

1. 長嶋洋一. 新楽器へのアプローチ. 情報処理学会研究報告(2015-MUS-108). 情報処理学会. 2015.
2. “Play the Road”. http://www.youtube.com/watch?v=3flwZ8OpXBY
3. FMC3. http://nagasm.org/FMC3/
4. http://nagasm.org/ASL/paper/ICICE201803_nagasm.pdf
5. H.B.リンカーン編. 柴田南雄/徳丸英彦他訳. コンピューターと音楽. カワイ楽譜. 1972.
6. http://nagasm.org/ASL/paper/JMACS_SIGMUS.txt
7. 西村恕彦監修. 音楽情報科学研究会編. コンピュータと音楽(bit別冊). 共立出版. 1987.
8. 特集:計算機と音楽. 情報処理. VOL.29,NO.6 情報処理学会. 1988.
9. 特集:これがコンピュータミュージックだ!. Computer Today. NO.34. サイエンス社. 1989.
10. 音楽情報処理の技術的基盤. 平成4年度文部省科学研究費総合研究(B)音楽情報科学に関する総合的研究. 課題番号04352030(研究代表者:甲南大学理学部教授 田口友康. 1993.
11. 特集:音楽情報処理. 情報処理. VOL.35,NO.9 情報処理学会. 1994.
12. 長嶋洋一/橋本周司/平賀譲/平田圭二編. コンピュータと音楽の世界(bit別冊). 共立出版. 1998.
13. 長嶋洋一. コンピュータサウンドの世界. CQ出版社. 1999.
14. http://nagasm.org/ASL/books/C_sound.pdf
15. http://nagasm.org/ASL/mse/
16. Curtis Roads著. 青柳龍也/小坂直敏/平田圭二/堀内靖雄訳監修. 後藤真孝/引地孝文/平野砂峰旅/松島俊明訳. コンピュータ音楽 歴史・テクノロジー・アート. 東京電機大出版局. 2001.
17. http://nagasm.org/ASL/cmm/
18. http://nagasm.org/ASL/midi03/
19. http://nagasm.org/ASL/max02/
20. http://nagasm.org/ASL/max01/
21. http://nagasm.org/ASL/dspss2002/
22. http://nagasm.org/ASL/temper/
23. http://nagasm.org/1106/news5/docs/tokkou2017.html
24. 長嶋洋一. 身体に加わる加速度とサウンドの音像移動に関する心理学実験報告(1/4). 電子情報通信学会ヒューマン情報処理(HIP)研究会資料(技術研究報告)HCS2012-5. 電子情報通信学会. 2012.
25. 長嶋洋一. 身体に加わる加速度とサウンドの音像移動に関する心理学実験報告(2/4). 情報処理学会研究報告(2012-MUS-95). 情報処理学会. 2012.
26. 長嶋洋一. 身体に加わる加速度とサウンドの音像移動に関する心理学実験報告(4/4). 日本音楽知覚認知学会2012年春季研究発表会資料. 日本音楽知覚認知学会. 2012.
27. http://nagasm.org/1106/news5/docs/Tour2016.html
28. http://nagasm.org/ASL/paper/MaxPatch.zip
29. http://www.suac.ac.jp/researchcenter/interact/cooperation/