

DIALOGIC CODING:  
A PERFORMANCE PRACTICE FOR CO-  
CREATIVE COMPUTER IMPROVISATION

J F HUMMEL  
PHD 2017

DIALOGIC CODING:  
A PERFORMANCE PRACTICE FOR CO-  
CREATIVE COMPUTER IMPROVISATION

JONAS FREDERIK HUMMEL

A thesis submitted in partial fulfilment of the requirements of the  
Manchester Metropolitan University for the degree of Doctor of  
Philosophy

Department of Contemporary Arts  
the Manchester Metropolitan University

March 2017



# Abstract

This research project explores *Dialogic Coding* – a performance practice situated within the field of live computer music which works towards a dialogic relationship with the computer as a programmable musical instrument.

The writing articulates a Practice-as-Research (PaR) inquiry that places my practice within specific contextual, analytical and philosophical frameworks. The point of departure is the assumption that following the concept of dialogue a more reflexive way of performing music with a computer becomes possible. This approach may produce innovative results through transformations of musical ideas, embodied interactions as well as the performer's self-concept within a situation of improvised group performance. Dialogic Coding employs the concept of nontriviality to create an independent but at the same time programmable musical agent – the apparatus – which so becomes a co-creator of the improvised music.

As a context for Dialogic Coding practice serve other dialogic forms of music making such as free improvised music as well as dynamic performances of programming found in live coding practice. A dialogic approach in music performance is based on listening and the ability to speak one's voice in response to the situation. Here, listening is understood beyond the auditory domain on the level of abstract thinking and physical interaction (interface affordance).

This research presents a first-hand account of a computer performance praxis and thus makes a contribution to academic knowledge. For this it makes some implicit or tacit 'knowings' contained in the practice accessible for an outside community through this writing. Dialogic Coding practice was developed through participating in free improvised music 'sessions' with other musicians as well as composing pieces in program code with which I then performed live (solo and group). This writing contextualizes the developed practice in a historic lineage, discusses it within the conceptual framework of dialogism and delineated how a dialogic approach fosters creativity, learning, surprise and flow. As a conclusion I summarise the ethical dimension of Dialogic Coding as a form of human-computer interaction (HCI).

# Acknowledgements

In this research I developed a subjective approach to composition and improvisation with a laptop. Thus, as no human is an island this project would not have been possible without the help of others. I am indebted to my supervisory team, Adam Fairhall, Jason Woolley and particularly my director of studies, Martin Blain, whose advice has been indispensable in completing this thesis, for their patience, critical questions and encouragement in the process.

This research was made possible through the Award of a Postgraduate Studentship from The Institute for Performance Research at Manchester Metropolitan University. I was lucky to have access to the facilities of the Department of Contemporary Arts on MMU Cheshire campus including a personal office space.

The dialogic approach taken in my project relied on the contributions of other musicians and improvisers I have been fortunate to work with. Particularly inspiring have been the collaborations with Mike Walsh and Adam Fairhall as well as with Niko Lefort. I want to also thank those early projects not referenced here which are the foundation on which my practice has developed: Graham Booth (Rebellious Devices and Chinese Whispers), Jasmine Guffond and Federica Furlani (INPLD) and numerous improvisations with Hui-Chun Lin and her guests. Additionally I want to thank the band powerbooks unplugged, particularly Julian Rohrerhuber, Alberto De Campo and Hannes Hoelzl who have introduced me to live coding and networked music teaching me many useful coding techniques which have inspired the development of my personal code virtuosity.

I want to thank the colleagues who have helped in the reviewing of this thesis with their critical feedback: Philipp Nielsen, Matt Wilson, Thomas McColgan, Mike Walsh, Shelly Knotts and Jasmine Guffond.

And finally, this work would not have been possible without the multi-level support of my parents and my family. Particularly, I am indebted to Miriam Giesecking for her patience, understanding and support, and our son Juri Giesecking who has kept me alert to the passing of time and incited me to finish this project.

# Table of Contents

<b>0 – Introduction</b>	<b>1</b>
0.1. The Role of the Writing	1
0.2. The Research Inquiry	2
0.3. The Role of The Practice	3
0.4. Structure of the Thesis	4
0.5. Summary of Outcomes	5
<b>1 – Dialogic Coding Practice and its Lineages</b>	<b>7</b>
1.1. Setting The Frame	7
1.1.1 Live Ensemble Performance	7
1.1.2 The Apparatus: a Dialogic Musical Instrument	8
1.2. Processes of Creation with a Programmable Musical Instrument	10
1.2.1 Materials	10
1.2.2 Dialogic Code Improvisation	12
1.2.3 Performer Roles	15
1.3. Dialogic Coding Practice as a Way of Knowing	16
1.3.1 Dialogue/Dialogic	17
1.3.2 Nontriviality	19
1.3.3 Dialogic Identity	19
1.3.4 Challenging and Immersing Performance	20
1.4. The Cosmology of Dialogic Coding	21
1.4.1 Free Improvised Music	22
1.4.2 Interactive Composing	24
1.4.3 Live Coding	26
1.4.4 Participative and Networked Performance	28
1.4.5 Post-Cagean Listening	31
1.4.6 Hacking and Glitch	33
1.5. Distinctions and Defining Features	36
<b>2 – Methodology, Conceptual Framework and Practice</b>	<b>38</b>
2.1. What is the Methodological Approach?	38
2.1.1 Nelson's Practice-as-Research Model in my Research Enquiry	40
2.1.2 Additional Input	42
2.2. Developing the Practice	43
2.2.1 Trajectories Through the Multi-Mode Inquiry	44
2.2.2 Adopting New Directions Through (Mid-)Course Corrections	45

2.2.3 A Family of Practices	46
2.2.4 Free Sessions as Laboratory	47
2.2.5 Rehearsing Dialogue	49
2.2.6 Creating Dialogic Spaces	49
2.3. Overview of Selected Cases	51
2.3.1 #A Free Sessions	51
2.3.2 #B Rehearsed Collaborations: Technospaetzle	52
2.3.3 #C1 Networked Piece: MindYourOwnBusiness	53
2.3.4 #C2 Participatory Piece: Projectionist Orchestra	54
2.3.5 #C3 Interactive Piece: SUM	55
2.4. The Conceptual Framework	56
2.4.1 Buber's Philosophy of Dialogue	56
2.4.1.1 The Dialogic Principle	57
2.4.1.2 The Twofold Existence of I-It and I-You: Distance and Relation	57
2.4.1.3 Dialogic Space: The Interhuman	58
2.4.1.4 The Problem of Mutuality in Unequal Relations	59
2.4.2 Bakhtin's Dialogism	60
2.4.3 Dialogic Intersubjectivity in the context of this research	62
2.4.3.1 Heteroglossia through algorithms	63
2.4.4 Flusser's Telematics	64
2.4.4.1 Apparatus	67
2.4.4.2 Functionary, Programmer, Envisioner	67
2.4.5 Ethics and Second-Order Cybernetics	69
2.5. Towards an Analytical Framework	71
2.5.1 Questions Towards the Dialogic in the Situation	72
2.5.2 Questions Towards the Apparatus (the nonhuman agent)	73
2.5.3 Performance & Musical Questions relating to Reflexivity	73
2.6. Summary	74
<b>3 – Simultaneous Interactions in Dialogic Coding</b>	<b>76</b>
3.1. Four Layers of Simultaneous Communication	76
3.2. Meta: Dialogue with the Non-Present	78
3.2.1 A Dialogic Sonic Vocabulary	78
3.3. Macro: Interacting with Performance Environment and Audience	81
3.3.1 Creating Dialogic Spaces through Participation	81
3.4. Meso: Performance Collaborators, Improvising Group	83
3.4.1 Compatibility of Players	83

3.4.2	Listening to the Yarn	84
3.4.3	Amplification/Silence	85
3.4.4	Strategies for Interaction	87
3.4.5	Group Intimacy as Facilitator	88
3.5.	Micro: Interacting with the Apparatus	90
3.5.1	Press, Push, Click and Type: The Ontology of Interactions	90
3.5.2	Listening to the Apparatus	92
3.5.3	Speaking to the Apparatus	94
3.6.	The Apparatus as Dialogic Partner	96
3.6.1	The Problem of Computational Perception	96
3.6.2	The Problem of Reflexivity and Action (Responsibility)	97
3.6.3	The Problem of Nontriviality (Alterity)	98
3.7.	Summary	99
<b>4</b>	<b>– Responsibilities of the Algorithmic Performer</b>	<b>100</b>
4.1.	A Heteroglot Performer	100
4.2.	The Listener	104
4.2.1	Multimodal Listening	105
4.2.2	The Calculating and Computing Consciousness	107
4.3.	The Programmer	108
4.3.1	Programmer and Functionary	109
4.4.	The Actor	110
4.4.1	Visibility	111
4.4.2	Participation	113
4.4.3	Reanimating the Inanimate	114
4.5.	Responsibilities for Dialogue	115
4.5.1	Dialogic Listening	115
4.5.2	The Dialogic Programmer	116
4.5.3	The Dialogic Functionary	117
4.5.4	The Dialogic Actor	118
4.6.	Negotiating Responsibilities	119
4.7.	Summary	120
<b>5</b>	<b>– Dialogic Coding as a Reflexive Practice</b>	<b>122</b>
5.1.	Dialogic Quality	122
5.1.1	Outsideness and Immersion	124
5.1.2	Transformation Through a Reflexive Practice	125
5.1.3	Forms of Immersion	126

5.1.4 Forms of Challenge	126
5.2. Interruption and Flow with Agencies in Dialogic Coding Practice	127
5.2.1 Free Sessions	127
5.2.2 Musical Structure vs. Algorithmic Freedom: Technospaetzle	129
5.2.3 Overpowering Complexity: MindYourOwnBusiness	131
5.2.4 Participation as Autonomous Agency: Projectionist Orchestra	131
5.2.5 SUM: Acting-Performing Against the Apparatus	132
5.2.6 Summary of Immersion/Challenges in Dialogic Coding Practice	134
5.3. Virtuosity in Algorithmic Performance	138
5.3.1 Essential Skills	138
5.3.1.1 Presence	138
5.3.1.2 Preparation	139
5.3.1.3 Virtuositic Bricolage Programming	140
5.3.1.4 Fluent Translation, Programming Efficiency	140
5.3.1.5 Empathic Imagination	141
5.3.2 Creative Strategies For Change	142
5.3.2.1 Decision-Making	143
5.3.2.2 Divergent Thinking and Embracing Ambiguity	143
5.3.2.3 Risk-Taking and Problem-Seeking	144
5.3.2.4 Accountability and Self-Assessment	145
5.4. Conclusion: Responsibilities for Change	145
5.4.1 Slow Programming or to Embrace Asynchrony	145
5.4.2 Increase The Number of Choices	147
5.5. Summary	148
<b>6 – Conclusion and Summary</b>	<b>149</b>
6.1. Chamber Music	149
6.2. Improvising Artistic Technoscience	152
6.3. The Five Aspects of Dialogic Coding	153
6.4. Future Outlooks	156
<b>7 – Appendix</b>	<b>157</b>
7.1. Glossary of Central Terms	158
7.2. List of Online Repositories for Practice Documentation (code/audio/visual)	161
7.3. Notes from Noise Concert Experience (Ryan Jordan, 2013):	162
7.4. Conversation/Questionnaire Hans Tammen, December 2014	163
7.5. Notes and Memos from Developing the Practice (on Technospätzle)	167
7.6. Score Example for Technospaetzle (2014)	170

7.7. Memo on Free improvisation Session 'Kuehlspot'	172
7.8. Memo of 'The process of live coding' in the Studio	174
7.9. Text Score for 'MindYourOwnBusiness'	175
<b>8 – References</b>	<b>176</b>

## Table of Figures

Figure 1: Nelson's multi-mode epistemological model' for PaR (2006).....	40
Figure 2: Information Sheet #A Free Sessions.....	51
Figure 3: Information Sheet #B: 'Technospaetzle'.....	52
Figure 4: Information Sheet #C1 'MindYourOwnBusiness'.....	53
Figure 5: Information Sheet #C2 'Projectionist Orchestra'.....	54
Figure 6: Information Sheet #C3 'SUM'.....	55
Figure 7: Macro level interaction in performance.....	80
Figure 8: Meso level interaction between performer (P), apparatus (A) and group (G).....	90
Figure 9: Text piece 'MIMICRY' (2013).....	107
Figure 10: Summary of Dialogic Qualities in Selected Cases.....	134
Figure 11: Structural Model of Flow.....	137

## Preface: How to Approach this Thesis

This Practice-as-Research (PaR) thesis consists of a written component, video footage, an archive of program code, as well as the Dialogic Coding practice itself. The written component is structured in six chapters which introduce the practice, its lineage and artistic context, its conceptual and philosophical framework and a discussion of central aspects. The Appendix section extends this with a glossary of terms and documents from the development of the practice.

The practice is supported and evidenced by video clips given in web links. The complete list of clips can be viewed online in a youtube playlist.<sup>1</sup> While this video footage serves to give a better impression of the practice and its aesthetics, the text is intended to be comprehensible also without these recordings. Furthermore, some of the code files which were used for performance can be found online.<sup>2</sup> This code does not work as a standalone: it first requires the installation of the elementary software (SuperCollider, version 3.7 or higher) including a few external libraries ('quarks'). The details for this can be found in the code files. Finally, as the most recent document of Dialogic Coding practice, the recordings of the performance before the oral PhD examination can be viewed online as well.<sup>3</sup> This includes six videos which present practice 'demonstrations' of the main concepts discussed in this thesis. The recordings connect to the writing as follows:

- video 1 & 2 interpret the roles of listener and programmer as discussed in chapter 4.1 and 4.3;
- video 3 comments on the finding a shared pulse or meter between human and nonhuman improvising agents. This refers to the general challenge of finding a balance between the individual freedoms of all agents which is a topic throughout the thesis (see summary below or chapter 6.3);
- video 4 & 5 interpret the concept of the apparatus and its implications on the micro (individual player) and the meso (group) level. This links to chapters 1.1.2; 1.3.2; 3.5 and 3.6;
- video 6 is a group improvisation following up on the idea of identity as dialogically constructed. This corresponds to chapters 1.3.3 and 5.

---

<sup>1</sup>[https://www.youtube.com/playlist?list=PLKRUGJaqJkAfIS7EXUfK8RpL1\\_s-ykeB\\_](https://www.youtube.com/playlist?list=PLKRUGJaqJkAfIS7EXUfK8RpL1_s-ykeB_)

<sup>2</sup><http://jonashummel.de/archives/code/>

<sup>3</sup><https://www.youtube.com/playlist?list=PLKRUGJaqJkAc76nEf-D8NfVSC4Q2Abew5>

*The page intentionally left blank*

## 0 – Introduction

This thesis explores the development, the conceptual framework and the practical experience of a mode of live computer music practice which I term ***Dialogic Coding***. This practice comprises improvised interactions between a programming performer, her *apparatus* – the programmable yet independent musical instrument<sup>4</sup> – and other human improvisers following the concept of a dialogic relationship (as a compositional strategy).

### 0.1. The Role of the Writing

The function of this thesis, as complementary writing to the research-practice, is to provide access to the new 'knowings' emerging from and contained in the Dialogic Coding practice. Therefore the practice will be positioned, clarified and analysed according to the 'knowings' and insights which have arisen by and through making work in this way. Through this process Dialogic Coding is presented as a distinct mode of 'doing-thinking' (Nelson 2013) – a performance practice which is a form of generating not only works of art but also knowledge in itself.

The main insight gained through the development of the research practice is that the dialogic approach may produce new and innovative musical results and offer a reflexive way of making music for a performer. The Dialogic Coding practice provides on one hand immersing but on the other hand challenging and distancing experiences for the performer and, through this, facilitates a transformative process guided by learning and discovery which contributes to a development of the performer's identity. The main challenge for the performer who works towards dialogic relationships with *all* agents participating in improvised group performance is one of *mediation and negotiation*: to create a balance between the autonomies of all agents involved, the human as well as nonhuman. This is the new dialogue-oriented musicianship of the 'algorithmic'<sup>5</sup> performer.

The emergent knowings in this thesis address the gap in current computer music scholarship by providing a conceptual approach for improvised performance in

---

<sup>4</sup>See section 1.1.2 for an introduction and explication of the term 'apparatus'.

<sup>5</sup>The use of the term 'algorithmic' here is intended to highlight the aspect of the performer as an inventor of computational structures which the apparatus transforms into music, thus, it points to the shift of action in performance from physical to intellectual. To make music in this way implies not only to 'perform' it on a musical instrument but also to invent and articulate it in an abstract form as part of the performance.

practice-based form in three ways: Firstly, the conceptual framework of dialogue (combining ideas of Buber, 1965; 1970; Bakhtin, 1981; Flusser, 2011) is brought into play with a creative practice resulting in a new analytical perspective<sup>6</sup> which is, following Nelson (2006) 'innovative' on the level of research. Secondly, the thesis represents an addition to ethnographic research in the field of live coding performance practice (Magnusson, 2014; Mori, 2015). It presents 'insider insights' into the typically obscure creative processes taking place through interactions between a programming performer and her programmable musical instrument in situations of group performance. It thus intends to provide access for an outside reader to the 'tacit knowledge' of the practitioner (Nelson, 2006) contained in the practice. Thirdly, this thesis examines the human-computer interaction from the standpoint of the programming performer and follows a focus on roles, identities and relationships between human (performers) and nonhuman (apparatus) agents which are created and changed as part of the performance<sup>7</sup>. As an outcome, this research offers a set of intellectual, organisational, and technical strategies for *responsible* live computer music performance in the pursuit of 'genuine' dialogic relationships (an ethics of the practice). Lastly, the intention of this project is furthermore, that the insights developed may become valuable beyond the context of music/arts performance for other situations of everyday human-computer interaction as well.

## **0.2. The Research Inquiry**

This research project pursued the application of theoretical ideas of dialogue to live computer music performance with a programmable musical instrument through a practice-based approach. The terrains explored in the developed Dialogic Coding practice in practical terms are new creative musical-technological interactions, an attainment of immersive states (flow), the encounter of challenges for one's habitual ways of acting/thinking and a new identification with both the technological other and the improvising group. In one phrase the main research question for this project can be expressed as:

---

<sup>6</sup> This includes new understandings of conceptual terms (i.e. dialogue) obtained from applying them in the context of human-apparatus, or nonhuman, interaction.

<sup>7</sup>With this orientation it can be considered an actor- or performance-focused more than a technology-focused research project (even though it is necessarily founded on programmable technology).

Does a dialogic approach to the computer as musical instrument in situations of group improvisation enable interactions which provide possibilities for reflexivity and transformation for the performer?

The aspects which need to be examined further in this thesis are:

- How can the computer be a dialogic partner? In which symbolic or non-symbolic forms does this manifest? How is the musical interaction with other performers dialogic?
- Which prerequisites and skills does a performer need to participate in the dialogic interactions with computer and, simultaneously, the improvising group?
- How exactly does Dialogic Coding practice create possibilities for reflexivity and transformation? What situations or conditions make these possible? What kind of transformations take place?

As a conclusion I propose a set of *ethics* or guiding principles for Dialogic Coding practice: the *responsibilities for dialogue* of the performer (see section 4.5 and 5.3).

### **0.3. The Role of The Practice**

The projects of the Dialogic Coding practice created as part of this doctoral project, hold the emergent knowledge and contribute each a distinct portion of the general experience (visual- and sonically), to the main research inquiry. The arguments pursued are necessarily exclusive in their nature – they do not attempt to condense and encompass all that is present within the work. Similarly the video recordings of performances referenced throughout the text do not attempt to present the full experience of a Dialogic Coding performance event but singular aspects of a complex whole. In the same way the recorded examination performance appended to this submission demonstrates a selection of partial aspects or 'knowings' from this thesis.<sup>8</sup> This selectiveness reveals my personal interests, engagements as well as frustrations with the practice I developed, while at the same time working towards its articulation as new knowledge in this written form.

As a practicing researcher I am privileged to speak and criticize my own work from the double perspective of being 'inside' as well as outside of the process. To

---

<sup>8</sup>See the previous section 'How To Approach this Thesis' and the list of online resources for the recordings (appendix, section 7.1).

counteract an overly subjective account however, this writing draws on a range of voices, perspectives and analytical moments in time to create a productive discourse which articulates the insights and knowings from this research project. These voices are in the form of personal journal reflections, audio/video documentation of past performances or rehearsals, informal conversations with collaborating improvisers and participating audience as well as the voices and practices of theorists and practitioners whose ideas and work resonate with my own.

#### **0.4. Structure of the Thesis**

Structurally, the writing begins with an overview of the Dialogic Coding practice, its ontological form, the modes of creation, underlying theoretical aspects and terms, and how it is positioned in the context of other practices. This serves to draw out resonances and differences with the work of others and make clear the intentions of Dialogic Coding practice. The second chapter outlines the methodology used in this project, and shows how its concepts have been applied within the making of the practice. It presents an overview of selected examples from my practice which serve as a reference in the discussions of the subsequent chapters. The chapter further expands on the *conceptual framework* of Dialogic Coding practice by introducing the the history of the terminology use and the philosophical ideas in more detail.

The more discursive sections of the writing follow in chapters 3, 4 and 5. Chapter 3 focuses on the different communicative interactions in performance, with the computer instrument as well as with the other improvisers, and whether these are dialogic in nature. Chapter 4 retraces the position of the performer to identify the distinct responsibilities for dialogic interaction, particularly in relation to the programmability of the musical instrument. Chapter 5 explores how Dialogic Coding is a reflexive practice and enables transformative experiences for the performer by revealing the factors which may facilitate or inhibit these. Finally, the conclusion draws together threads from all chapters towards a new formulation of Dialogic Coding practice in order to mark the contribution this thesis offers to the field.

In the appendix (section 7) the reader is provided with a glossary of the central terms such as 'functionary', 'programmer', 'dialogue' or 'nontriviality'. This is followed by a list of available documentation audio-visual material online (see

section 7.2) and completed by selected notes and journal entries from the development of Dialog Coding practice, i.e. early critical reflections, documents from rehearsal processes such as scores, conversations with participants.

## 0.5. Summary of Outcomes

Within this thesis Dialogic Coding is presented as a practice providing new points of experience within the field of contemporary live computer music performance and thus represents a form of new knowledge. The insights gained through 'doing' Dialogic Coding are located in a number of positions within the thesis as a whole. For ease of reference I categorise and position them here as follows:

- **The Dialogic Coding performance practice** – This is a distinctive and reflexive practice, which through its doing produces insights (an epistemological tool). It is in itself a form of new 'knowing' in the field, produced by the application of a philosophical framework to a field of practice. See chapter 1 for an overview and lineage, chapter 3 for an analysis of its performance situation, chapter 5 for a discussion of its reflexive qualities. How Dialogic Coding is generated and practised will become clear throughout the whole writing based on the given references.
- **Responsible programming in dialogic code improvisation** – a specific form of making music with a programmable musical instrument based on improvised interactions. This approach brings together and extends both the categories of live algorithmic composition and of improvised electronic music. Furthermore, it introduces an ethical perspective for the programming performer following the intention to facilitate dialogic relationships. See chapter 1.2 for overview; chapter 4 for the role of the performer; chapter 5.4 for conclusions; and code examples online.<sup>9</sup>
- **The algorithmic musicianship of the performer** – a deconstruction of the performer's experience into four central role(s) and their re-synthesis into a new 'dialogue-oriented' musicianship which intends reflexivity and self-challenge by ways of doing the practice: the performer's virtuosity for dialogue. See chapter 4 and 5.4.

---

<sup>9</sup>See Appendix 7.1 and online here: <http://jonashummel.de/archives/code/>

- **Insider insights** – new personally informed perspectives on the process of creating and performing with programmable instruments in live situations and within a group of other performers. See Chapters 2 and 5.
- **References for Dialogic Coding practice** – This is the recorded documentation of the practice. It comprises rehearsals and performances<sup>10</sup>, categorised into three approaches: (#A) free improvised sessions, (#B) structured and rehearsed improvisations, (#C) improvised performance with composed/programmed systems. The example 'cases' referred to in this writing are the following:
  - The piece *MIMICRY* (2013)
  - The *Drums'n'Algorhythms* session (2013)
  - The *Kuehlspot* sessions (2014)
  - The *SUNPYX* project (2014)
  - The *Technospaetzle* project (2014)
  - The piece *MindYourOwnBusiness* (2013) and other performances with the Birmingham Laptop Ensemble (BiLE)
  - The piece *Projectionist Orchestra* (2014-15)
  - The *SUM* project (2014-15)
  - The recording of the 'Live Dialogic Coding' performance before oral examination (2017)

A detailed overview of these can be found in the 'information sheets' in section 2.3. In addition, links to excerpts from performances online will also be given in footnotes in the discussion. A complete list of all online repositories for these references (including relevant program code) can be found in the Appendix.

---

<sup>10</sup>A detailed list of all the online resources with documentary evidence of my practice can be found in the Appendix 7.2.

## 1 – Dialogic Coding Practice and its Lineages

This chapter serves to establish the main concepts of my Dialogic Coding practice and position it within a lineage of relevant practices. The first three sections introduce its three main aspects: a) the situation of group performance with a programmable musical instrument which serves as the framework for the practice, b) the processes of creation and interaction in this situation and c) how this practice becomes a way of knowing on the basis of dialogic interaction. Dialogic Coding will then be positioned within a lineage of other related music and performance arts practices. Through such comparisons with resonant works in the field the distinctive nature of this approach will be delineated which at the same time represents the new knowledge gained through practice. The term definitions and context given in this chapter take up the main threads of inquiry and reflection presented in the introduction and will be elaborated on in subsequent chapters.

### 1.1. Setting The Frame

This research-practice has been developed in the situation of group performance, which brings together the improvised interactions between the performer and her interactive musical instrument with interactions with a group of improvisers.

#### 1.1.1 Live Ensemble Performance

Dialogic Coding practice was purposefully developed in a context of group performance and collaboration with other improvising musicians to challenge the strong focus of the coding performer on the computer screen. The dialogic approach with an attitude of *listening being* (see section 1.3.1) demands direct attention towards *all* agents in the situation: the group of human improvisers as well as a performer's own interactive apparatus. Additionally, in those events featuring a co-present audience, the attention is simultaneously directed to these *outside* listeners or towards the atmosphere in the performance space in general. Hence the flows of communication and interaction create multiple dialogic spaces between different partners in parallel. I have categorized these in four main levels. First, there is the asynchronous **Meta** level. Communication here takes place between a performer and the history and traditions of her performance practice in the form of musical sounds, performative gestures as well as direct or indirect

references in the program code (less visible).<sup>11</sup> Second, there are the three levels of synchronous communication between co-present partners: The **Macro** level contains the communication between performer and audience in the form of an 'atmosphere'<sup>12</sup> (Böhme, 2013) in the space. The **Meso** level refers to the exchanges between performer and other human improvisers in the group in the form of musical sounds, embodied or visual gestures (looks, nods etc). The **Micro** level contains the communication between performer and her programmable yet independent instrument in multiple forms (visual, embodied, sonic *and* written code/data). This last level is a distinctiveness of live computer music performance with interactive instruments in general and the main site of analysis for this thesis. The details of this situated framework, particularly towards how the interactions on any level might be or become dialogic, will be analysed in chapter 3 in more detail.

### 1.1.2 The Apparatus: a Dialogic Musical Instrument

The musical instrument is particularly significant in Dialogic Coding practice because it is a programmable tool for the performer but also functions as a potentially autonomous partner in performance to be interacted with. Only this changing character and functionality makes the reflexivity and transformative experience possible. The complex role of this instrument termed 'apparatus'<sup>13</sup> in this thesis, stands in contrast to a 'traditional'<sup>14</sup> musical instrument. While the latter can be regarded as an extension of the performer's body who, as result of long

---

<sup>11</sup>This form of interaction between performances through time and space follows Bakhtin's (1981: 276) interpretation of dialogue, when he writes: 'the living utterance, having taken meaning and shape at a particular historical moment in a socially specific environment, cannot fail to brush up against thousands of living dialogic threads, woven by socio-ideological consciousness around the given object of an utterance; it cannot fail to become an active participant in social dialogue.'

<sup>12</sup>Böhme (2013: 263) defines the 'atmosphere' in the context of performance arts and theatre as 'quasi-objective emotions, spread out into the space in indeterminate ways' which happen in a specific time and require a perceiving subject to exist. It is the 'being in-between' and its tendency to bring the subject into a particular mood which is the particularity of an 'atmosphere'. This position is grounded in the philosophical understanding of *listening* as proposed by Nancy (cited in Heble and Caines, 2014: 21): 'To listen is to enter that spatiality by which, at the same time, I am penetrated, for it opens up in me as well as around me, and from me as well as toward me: it opens me inside me as well as outside, and it is through such a double, quadruple, or sextuple opening that a "self" can take place.'

<sup>13</sup>The term 'apparatus' in this thesis follows Flusser's understanding of a 'plaything or game that simulates thought (...) [or an] organization or system that enables something to function' (Flusser, 2000). It describes a structure which operates on a symbolic level, transforming thoughts, in automated and programmed ways. In this thesis the term 'apparatus' refers to the device(s) to make music with – ontologically (as the object on stage with all its external devices, controllers etc.) as well as conceptually (the algorithms and rule-based procedures inscribed through the activity of programming).

years of repeated rehearsing, plays the instrument in a virtuosic manner to express or convey ideas contained in the music, the former does not extend the performer's body in the same way. On one hand the interfaces built into the computer typically involve small and hardly visible body gestures (e.g. with the hands and fingers when typing), on the other hand the apparatus extends the performer in an *intellectual* way (again hardly visible). Because of these problems of opacity, it is necessary to deconstruct the various technological media and interfaces which make up the apparatus and 'follow the actors' (Latour, 1987) who invented it: the programming performer.

The apparatus comprises a dynamic assemblage of different objects and materials which affords the performer multiple modes of embodied as well as intellectual interaction. Ontologically, this assemblage centres around the laptop computer with its built-in keyboard, touchpad and screen. It is extended by a variety of external controllers or human interface devices (HID) ranging from MIDI controllers (o.e. a box of faders), to gamepad, mouse, or other self-built HID. Often there will be also additional external audio hardware used which sends the audio output generated by the computer to an amplification system in the performance space. The form and combinations of these objects in the practice have shifted throughout the project and are different for each of the examples provided in chapter 2. In each of these interfaces a different mode of interaction takes place:

- the computer screen affords visual monitoring of displayed information (feedback on running processes or communication with other human players);
- the computer keyboard affords the pressing of specific keys (for activation/deactivation or altering of running sound processes, for writing code words to create new algorithms);
- the touchpad/mouse pointer interacts with graphical user interfaces (GUIs) in both of the ways mentioned;

---

<sup>14</sup>The distinction between the apparatus and 'traditional' musical instruments serves to separate two principally different groups of objects to make music. 'Digital music instruments' (DMIs) are systems for controlling digital sound synthesis (after Wessel, Wright and Schott, 2002) and typically exist of a multi-layered setup which 'begins with some form of input from a human performer and results in audio output, usually via algorithmic sound synthesis on a laptop computer' (Emerson, 2015: 1). The main differences are that a DMI can have any number of different input-/output relationships, as determined by the mapping design and the sound is electronically produced and needs amplification to be audible. The apparatus qualifies as a digital music instrument in principle, however, it includes the sound-generating computer and is also a conceptual term.

- the external controllers afford specific embodied and gestural interaction which typically control dedicated sound processes running in the computer.
- In all cases the audible sound plays an important role in the feedback loop which drives the evaluation of previous actions and decision-making on future actions of the performer.

The way how these interfaces with their respective modes may be interconnected in a functional structure is key for the improvised human-apparatus interaction. One of the distinctions in Dialogic Coding practice is the fact that through live coding<sup>15</sup> these interconnections can and shall be changed dynamically in response to the affordances of any moment in a performance. Thus, the apparatus as a whole 'ecosystem' of interacting objects is only momentary stable. Its structural changes are themselves subject of the improvised performance. In this way the apparatus provides the possibility to actually change and shift the very frame of performance within which the practice operates.

## **1.2. Processes of Creation with a Programmable Musical Instrument**

In this section I will highlight the processes of creation underpinning Dialogic coding practice: Exactly which materials are composed, deconstructed and re-composed in real-time? How does this happen and in which modes of interaction? Such processes are not fixed but have shifted throughout the project. The features which have remained will be introduced here to indicate how this work is a distinctive computer music practice.

### **1.2.1 Materials**

The material most often used in my performances are fragments of SuperCollider (SC) code which specify single event synthesis sound processes (SC SynthDef objects or 'synths'): In addition, in some projects pre-recorded sounds are used as samples. Both of these elements are then played (triggered and sequenced) live through programmed event patterns (with SC Pdef/Pbind Objects). These patterns can be written live during performance and further be flexibly re-used with different parameters to achieve changes in the sonic material or the musical aspects of timbre, amplitude/dynamics, rhythm or others.

---

<sup>15</sup>'live coding' describes the practice of writing algorithms during performance and changing them while they are running. As a performance practice within live computer music it works with generative structures (sounds are created automatically/algorithmically by the computer) but includes a live programming performer who 'navigates' the algorithms on a musically feasible course.

Another type of code used are specifications for continuous synthesis sound processes (as SC Ndef objects). They are 'played' through activation and then changing the input parameters of the process either through manipulating the interactive elements on a chosen interface (the direct, embodied way of playing). Alternatively as well: processes may be changed by a programmed automatic pattern (the algorithmic way of playing

embodied interaction (on some HID) or by programming a pattern which changes the parameters through automation (meta approach).

While in the early development of this research-practice my activities as performer were focusing more on the aspect of sound design and timbral development of the actual sonic material played, this has shifted to how these sounds are activated and arranged in time during performance. The development of Dialogic Coding practice produced numerous fragments of program code from improvised (live coding) studio work (SC patterns, synths, Ndefs) which I have collected and documented.<sup>16</sup> Similar perhaps to how an improviser with a traditional acoustic instrument trains her memory through practice, I have trained my own, however relying on the artificial memory of the apparatus in the form of an organised code archive accumulated during in the course of this project.

In a performance a large part of the algorithmic structure through which the sounds and patterns are activated is pre-programmed. In Dialogic Coding practice I have mainly used default GUIs (i.e. The ProxyMixer, TdefGui or PdefGui objects from SC-JITLib) for screen- and pointer-based interaction. Furthermore, the computer screen provides visual feedback about running sound processes and the states of interface devices. The functionality of interactive elements on external controllers (i.e. the knobs and faders of the KORG NanoKontrol2) are also pre-programmed to specific functions which acts as a proxy between the interface and the currently running sound processes (using SC MFunc objects). These functions can be flexibly assigned and re-assigned to perform any kind of sound-controlling action. The benefit of this is the possibility to control many different processes in parallel or navigate a single complex sound process with many control parameters by means of only few interaction elements. The improvisations in Dialogic Coding performance usually start from some prepared algorithmic structure offering

---

<sup>16</sup>See online for an archive of program code used in Dialogic Coding practice:  
<http://jonashummel.de/archives/code/>

selected interfaces for direct interaction with particular sounds. This structure for interaction may then be rearranged in the course of a performance.

Concluding, I subsume both the code fragments which specify the actual sounds as well as the code which specifies the flow of control through human-apparatus interaction under the term 'pre-existing elements'<sup>17</sup> (PEs). In Dialogic Coding practice the activation, combination and re-combination of these elements is the subject of performance.

### 1.2.2 Dialogic Code Improvisation

From the perspective of creation the apparatus has worked for me not only as a tool to develop and articulate a compositional idea in a coded form but also as an instrument to directly play a musical idea or gesture, that is to activate sound events which are prepared for this interaction. These two forms of instrumental interaction are the main modes of performing in Dialogic Coding practice: the mode of *programming* and that of *playing* the apparatus, which I term the mode of '*functioning*'<sup>18</sup> in this thesis. These modes correspond each to a 'role' of the performer which I outline in the following section 1.2.3.

Based on the concept of dialogic interaction I have developed a compositional practice, or a way of creating and performing in these two modes – the *dialogic code improvisation*. In the local context of this project 'composition' is understood as an activity which produces the conditions for performance – in form of the apparatus' algorithms – through acts of thinking and programming. The creation of the music in performance then happens through the interaction with these materialised conditions in 'improvised' way. *Composition* stands however not opposite to *improvisation* but describes another perspectives on the same process, comparable to Nettle (1974) distinction of 'rapid' and 'slow composition'. Following Butler (2014: 120) I describe the musical creation with the apparatus as a production of sonic outcomes on a continuum between those 'that are specified only *in time*' (of performance) and 'those that are maximally specified *ahead of*

---

<sup>17</sup>Butler (2014) proposes his concept of *preexistent elements* as a broad term to 'be applied flexibly and to address a wide variety of phenomena' (i.e. the fact of using a 'readymade' (Sawyer, 2003) of musical structure, or – similar to Pressing's (1988: 52) 'referent' – a structure 'which guides and aids the production of musical materials'.

<sup>18</sup>The 'functionary' is a term describing a specific way of human-apparatus interaction in which the human only 'calculate[s] and compute[s] instructions as instructed' (Flusser 2011:72), in other words 'to function' means 'to act' as intended by the inventor of the computer program (the programmer). Thus a functionary can be considered part of the apparatus' program.

*time*' by arranging the aforementioned PEs. Dialogic Coding practice works with the approach of 'bricolage programming' (Turkle and Papert, in McLean, 2011) but with the intention of creating a dialogic relationship. In *bricolage programming* the process of composing (in real-time) does not follow a preconceived plan but it navigates through the algorithmic structure by arranging and recombining readily available code material (the bricolage aspect<sup>19</sup>) in response to the immediate affordances (i.e. the generated sounds). Live coding can be regarded as a literal form of bricolage programming (see section 1.4.3). In dialogic code improvisation this approach can be used both as an improvisational method to develop a 'system-as-composition' and as an interactional strategy for exploring the possibilities in a system or interface. It works as a strategy for live performance as well as for private studio composition and practice. The dialogic aspect now extends this type of improvisation with code by giving a direction in which to travel: towards the discovery of the new and unexpected – in the musical events, the group-interaction, or the apparatus-interaction. This is has its origins in the basic concept of dialogue which pursues an open-ended encounter of the 'other' in mutual exchange (see also section 1.3.1 below). A prerequisite to 'discover' in this way is some degree of indeterminacy within the apparatus logic which describes the production of unpredictable and unexpected results through interaction. This is the feature of 'nontriviality' in the apparatus outlined in section 1.3.2. below. Consequently, dialogic code improvisation strategically utilizes chance or random-decision making operations in the mode of programming provides to create this kind of indeterminacy. When I interact with patterns or continuous sound processes I have often used code objects which generate random numbers based on probabilities within defined ranges. Once inserted into the algorithm, I then experimented with the values for probabilities, their defined ranges or moved these objects to a different position within the algorithmic structure – all guided by the general intentions: to discover unexpected sounds. An additional means to introduce uncertainty into the system is to improvise with code objects the performer knows little about (and thus cannot predict its outcomes). The programming environment used in this project has provided abundant possibilities

---

<sup>19</sup>Anthropologist Levi-Strauss termed the 'bricoleur' to describe practitioners who 'do not move abstractly and hierarchically from axiom to theorem to corollary (...) [but] construct theories by arranging and rearranging, by negotiating and renegotiating with a set of well-known materials' (cited in Turkle and Papert, 1990: 136).

for sound synthesis or interaction design as to always offer more unknown terrain for discovery. Moreover, there is the possibility to simply invent new 'black boxed' code objects which function in indeterminate ways. Thus, the flexibility of the environment does not only help to make actions easier and quicker by means of automation, but it provides at the same time the means to make the interaction more intricate. Dialogic code improvisation describes improvised interaction with building blocks of code which may be brought into effective action on the direct level of sound generation but also on the level of the interface, or the 'generation' of user interaction.

This *user interaction* describes the complementary side of dialogic code improvisation: the 'functioning' mode of interaction. Depending on the particular interface, the interaction takes on a more embodied and gestural form. Dialogic code improvisation is now actually a 'dialogic interface improvisation' in which the performer explores the interface, investigating all its elements in the pursuit of the dialogic purpose: the discovery of the unknown. The main difference to the programming mode can be found in the amount of algorithmic control the performer holds. The question whether there really exists a boundary between these two modes enacted by the same person is legitimate: Perhaps the adequate picture is this: The performer is in fact in a *functioning* mode of interaction when directly live coding in rapid cycles of code execution, sonic feedback, and re-programming the next idea into the algorithm. The planning horizon is short, the 'navigation' follows mid-course corrections, the interaction is improvised, immediate and flexible. But whenever she raises her head above this live coding action to look into the distance with a more strategic orientation, this becomes a programming mode of interaction. Thus, the ontological activity of programming (i.e. the typing of characters and code words into a computer) does not automatically indicate a creative mode of programming. The difference is in the specific consciousness or focus of attention a performer follows in any moment.

To navigate on course in dialogic code improvisation it is necessary to listen in a specific way, attending to a variety of inputs in the natural and the technological environment. This enables the performer to take various routes towards dialogic encounters – not only with the apparatus but also with the other human improvisers. Furthermore, it is helpful to have material ready-at-hand. For example by creating a pool of code fragments which allows easy retrieval and adding of

new elements to create the desired indeterminate structures and change them on-the-fly. Through this I was able to respond quicker to external inputs or introduce new challenges when required. The bricolage programming approach, situated within group improvisation, have shaped dialogic code improvisation into the direction of working with small, and repeated, but immediately audible changes rather than taking time to write lengthy code blocks which enters the improvised situation much delayed. Taking time to develop the building blocks and structures of code has happened during studio preparation or in private group improvisations with exactly this purpose.

### 1.2.3 Performer Roles

A characteristic of Dialogic Coding practice and a consequence of this mode of code improvisation is the specific musicianship for the performer which comprises several attitudes akin to De Bono (2000) concept of the 'thinking hats'<sup>20</sup>. I have termed these the *roles of the algorithmic performer* (see chapter 4). Even though they are enacted by one and the same person in reality, it is helpful to look at these four roles separately in order to better understand the creative processes in this practice and their relation to dialogue.

Following the precondition of listening for dialogue (see section 1.3.1) the first and basic role is that of the *listener*. It is present at the outset of any performer action and establishes the foundation on which decisions for actions are made. Second, in the direction of the outside audience the performer is concerned with a representation of ongoing human-apparatus interactions. This role I have termed the *actor*. It serves to communicate the negotiations of power, control and influence which are otherwise hidden from the outside. I need to note that in this research project I have focused on this attitude in only few projects (i.e. the *SUM* project). Additionally, there are the two roles previously mentioned. The *functionary* who interacts with the apparatus interfaces according to its possibilities. When performing in this attitude a dialogic relationship with the apparatus can be experienced in the most original understanding of the concept, that is to regard the technological other as an independent subject with whom the performer can interact but whose response she is not able to completely predict. Finally, there is the *inventor* of technology, what I term the *programmer*<sup>21</sup>, who invents the algorithms of the apparatus or the rules according to which the system

<sup>20</sup>See also [http://www.debonogroup.com/six\\_thinking\\_hats.php](http://www.debonogroup.com/six_thinking_hats.php) for a quick introduction.

operates. The scenarios for how a 'dialogic coder' acts in performance are various: as a programmer she may be focused on the invention of apparatus logic (while the system generates sound automatically) or as a functionary she may interact with the interfaces provided from the same logic to influence the sound generation directly/manually. This conceptual divide is further narrowed when the performer 'interacts' with the programming language in a functioning mode while simultaneously inventing new rules or when she discovers new possibilities for interaction by means of creative (or *dialogic*) exploration of a given interface.

This highlights the fundamental paradox of Dialogic Coding practice which is at the same time its potential: The performer can interact dialogically with an *independent other* (this may be the apparatus but also the other improvisers) while she may also *invent this other* to create new possibilities for dialogue. In other words: the process of performance describes a circular course starting from the invention of an autonomous nonhuman partner (possibly prior to performance) which then reveals new perspectives in the dialogic interaction with a functionary-performer which in turn motivate the programmer-performer to change the apparatus by inscribing her new intentions into it which are conditions for future transformations through practice. It is in this comprehensive and entangled way that the performer in Dialogic Coding practice develops a new musicianship in the pursuit of the unknown and unexpected – amalgamating the complementary roles of inventor and user of technology, as well as a listening focus and an embodied acting perspective. The apparatus becomes cocreative in this process in how it provides the medium through which this musicianship is developed.

### **1.3. Dialogic Coding Practice as a Way of Knowing**

The intention and a consequence of the *dialogic approach* is facilitate reflexivity and transformative experiences through practice for the performer. This happens in the form of encountering challenges in the improvised interaction which in turn may produce such insights. The encounter of unexpected events poses challenges in the interactional process which produce moments of reflection (listening and reconsideration). This reflection enables the performer to reconsider her action and learn about the process.

---

<sup>21</sup>Flusser (2011) originally uses the term 'envisioner' to describe the 'dialogic programmer', a complementary role to that of the 'functionary'. But since his concepts were developed in a scenario of producing 'technical images' (that is images and sounds) this term seemed inappropriate in the purely musical context of this research.

To make clear the terminology of dialogue used in this thesis, the following section will sketch out the underlying philosophical ideas. A more complete elaboration of the conceptual framework follows in chapter 2.4.

### 1.3.1 Dialogue/Dialogic

Dialogue or a dialogic relationship<sup>22</sup> is understood in this thesis to refer to a specific way of interaction between two autonomous subjects – communicating through some form of symbols in any of the media: image, gesture, sound, or text. This requires each to *listen and respond to* the other in an authentic way and without prejudices. In the conceptual framework of Dialogic Coding practice the ideas of Martin Buber, Mikhail Bakhtin and Vilém Flusser are brought together and complemented by perspectives from cybernetic theory and improvisation studies. This framework is then applied to practice in a distinctive way: following the assumption that a *dialogic relationship with technology* is possible and may (or needs to) further be programmed to emerge. A genuine dialogic relationship is described by Buber (1965b: 78) as:

I become aware of him, aware that he is different, essentially different from myself, in the definite, unique way which is peculiar to him, and I accept whom I thus see, so that in full earnestness I can direct what I say to him as the person he is.

In order to apply Buber's ideas to HCI it is necessary to create and perceive the technology as an *autonomous subject*. The apparatus in this practice is, strictly speaking, a *pseudo-subject*<sup>23</sup> for it does not perceive itself as autonomous in the way a human being would. This scenario classifies to what dialogic theory terms an 'unequal' relationship without 'mutuality'<sup>24</sup>, implying that the partners are not perceiving and 'confirming' each other in the same way. It is obvious that a relation between an autonomous human subject and the programmed apparatus is of a

---

<sup>22</sup>Bakhtin's theoretical concepts of dialogue are rather heterogenous and have also been referred to by scholar with the comprehensive term 'dialogism'. I avoid using this term here because it 'can mean, and indeed has meant, many things to many critics, sometimes without reference to Bakhtin' (de Man, 1983: 100)

<sup>23</sup>I term this 'pseudo-autonomy' for it is not completely independent and self-regulated in the complete sense of a human being, that is: a *nontrivial* or self-regulating system. Still, the term 'autonomy' or 'independence' is used without constraints to refer to the functioning of the nonhuman agency.

<sup>24</sup>Phillips (2011: 32) state that 'through presentness, authenticity and confirmation, mutuality can occur – rather than the instrumentality of subject-object relations. From mutuality, dialogue emerges. (...) We can only have a degree of mutuality, even in unequal relationships.'

different type. Based on Buber's perspective that a subject-subject relation is possible to nonhuman objects, i.e. a tree,<sup>25</sup> Dialogic Coding practice needs to invent the nonhuman agent with a kind of autonomy which brings the encounter closer to *mutuality*: by creating a *nontrivial* system (see next section). Presuming that the performer holds the capability to change how the nonhuman pseudo-subject is able to relate, an opportunity and a responsibility towards the facilitation of dialogue as inventor and the participation in dialogue as autonomous subject emerges. Through *dialogic code improvisation* the performer is able to work with this dilemma in creative and productive ways by means of creating varying degrees of technological pseudo-autonomy efficiently.

Any dialogic relation rests on an ability to listen in an attentive and present way, what Lipari has termed an attitude of:

**listening being** [which] requires a willingness to suspend already familiar conceptions, beliefs, and understandings. (...) This form of inner emptiness facilitates a focus and attention that enables one to really absorb the other's words beyond the confines of what has already been thought, believed, or understood. (2010:356)

Again transferring this to an HCI situation the question is: What does a *listening being* towards the apparatus mean if the other 'speaks' not only in the medium of musical sounds (which allows for ambiguous meaning already), but also in other coded forms such as the feedback from the programming language or the computer's operating system? And what does a 'listening apparatus' imply?

It becomes clear that a *listening being* attitude in Dialogic Coding requires a variety of modes of perception, understanding, and interaction (embodied, intellectual, visual and sonic) in order to enable parallel dialogic relations with both the human improvising group *as well as* the nonhuman apparatus in all its interfaces. Consequently, in this thesis the performance of the attitude of listening is understood much more broadly beyond the auditory domain alone, rather in a phenomenological sense of a being present and attending towards the entire perceptible environment and all objects therein – the audible, visible and tangible.

---

<sup>25</sup>Buber (1937) gives the example of experiencing a tree in its single elements through an 'I-It relation' as opposed to encountering the tree in its wholeness through a dialogic relation, that is 'to glide with one's own feeling into the dynamic structure of an object, a pillar or a crystal or the branch of a tree, or even an animal or a man, and as it were to trace it from within, understanding the formation and the motoriality of the object with the perceptions of one's own muscles.' See section 2.4 for more detail.

This understanding is described in detail in the algorithmic performer's role of the *listener*, expanded on in section 4.2.

### 1.3.2 Nontriviality

A dialogic relationship is based on the independence of both partners. The behaviour and responses are autonomous and not predictable by the other. This property is described by cybernetics – the 'science of control and communication, in the animal and the machine' (Wiener cited in Ashby, 1956) – as the 'nontriviality' of a system. Von Foerster (2003: 208) explains the two main systemic principles of trivial/nontrivial:

A trivial machine is characterized by a one-to-one relationship between its 'input' (stimulus, cause) and its 'output' (response, effect). (...) this is a deterministic system; (...) this is also a predictable system. Non-trivial machines, however, are quite different creatures. Their input- output relationship is not invariant, but is determined by the machine's previous output. In other words, its previous steps determine its present reactions. (...) they are unpredictable: an output once observed for a given input will most likely be not the same for the same input given later.

Therefore, in order to function in a nontrivial way (as humans do), the apparatus needs to be programmed in a way which produces a new and different result for a given input. This may be done by means of creating some kind of feedback structure which takes the output of a calculation back into the next calculation, in other words: to build circularity into the system. In Dialogic Coding practice this concept forms the basis of a performer's compositional strategy for the programming interaction with the apparatus.

### 1.3.3 Dialogic Identity

The reflexivity of Dialogic Coding practice is directed/oriented towards transformation (i.e. processes of learning). Following the theories of dialogue any genuine dialogic encounter holds a potential to substantially influence each partner towards a transformation of her self-concept.

Genuine dialogue (...) entails a risk, the 'danger' that by truly listening to the other – be the other individual, a text, a work of art – that one might, indeed, be changed transformed cognitively and existentially (Mendes-Flohr, 2015: 3).

This perspective points to how identity is determined intersubjectively – through interaction of subjects. Peeren (2007) differentiates further 'dialogic intersubjectivity' which acknowledges the existence of the *other as different* from oneself.

Dialogic intersubjectivity marks dissonance and distance rather than harmony and closeness; it is not about recognizing the other as the same, but about respecting the other as different and taking responsibility for this difference. The other does not become an object, but is recognized as another subject. Dialogism means responding to alterity without negation or assimilation.

Kanellopoulos (2011:124) concludes from this that improvised music practices are not per se a dialogic form of music making but that this should be regarded as 'a consciously sought after condition'. It is not enough to bring different previously formed identities into contact, but to collectively create something new and thereby produce alterity. Transferring this conceptual view to the local context suggests that the dialogic approach may indeed offer ways to create and alter a performer's identity through the intersubjective interaction with technology. The precondition for this being that the apparatus is programmed in a form which can be 'recognized as other' (i.e. by implementing the principle of nontriviality).

#### 1.3.4 Challenging and Immersing Performance

Suggesting Dialogic Coding as a reflexive practice in this form is based on the specific experiences I have made through making music in this way. A more autonomous musical instrument poses a challenge for the performer who interacts with this agent: it becomes uncertain whether she can express her intentions (the responses to the musical situation) in a direct way. In other words: the apparatus challenges the performer's musicianship and mediates her musical ideas. The interfaces of the apparatus transform into sites of negotiation where the performer's flow of thinking or embodied actions is supported or interrupted (i.e. a specific manual action produces an expected and desired change in the sound, a vaguely imagined abstract musical idea is realised in code and in this way refined and developed further). The states of consciousness encountered in these interactions may be described in Bakhtin's terms of 'empathizing' with or 'stepping outside' (cited in Kanellopoulos, 2011: 125). Accordingly, I describe the experiences in Dialogic Coding practice as states of *immersion* or of *outsideness*. Csikszentmihalyi's model of flow (2002) provides a theoretical frame which outlines the conditions for an immersion to emerge. Any performance event consists of many such momentary states which continuously interchange with each other. While an immersive state creates an experience of losing one's self in any of the activities of listening, functioning or programming, the outside state supports a critical reflection of one's actions and thoughts and thus, an opportunity

to act differently. The details of this will be discussed in chapter 5. In the development of my practice the challenging (frustrating but also engaging) experiences with a strongly independent apparatus may have hindered immediate expression of ideas in a creative or interactional flow but have sparked off new ideas in the attempt of resolving such conflicts. Similarly, the dialogic interactions with the rest of the improvising group may produce states of immersion and outsidership leading to a reflexive quality as well. One improvisational strategy may have 'worked' with one group of improvisers while it failed to produce musically interesting results (that is, with a potential for dialogical self-transformation) in another situation.

#### **1.4. The Cosmology of Dialogic Coding**

As suggested previously Dialogic Coding practice brings together a wide variety of topics:

- historic (interactive composing) and contemporary (live coding) artistic strategies in computer music performance;
- virtual (code) and physical/gestural (controllers) computer interfaces;
- instrument building through explorative performance (*sound as well as* interaction design);
- live algorithmic composition and free improvisation with pre-existent elements (code);
- the performance modes of programming (inventing, composing) and functioning (playing, acting) based on listening (observing);
- human and artificial improvisers (nontrivial and programmed systems);
- music-making as a self-reflexive and knowledge generating practice.

These topics bring in an array of contextual influences which will be delineated in the following paragraphs. According to the three main aspects of Dialogic Coding practice as outlined above I have grouped the lineages in three categories: (1) improvised (and dialogic) live performance, (2) algorithmic and interactive composition and (3) music making as a reflexive practice. An overlap between these categories is unavoidable for some established practices cross-over into adjacent fields.

The most influential lineage is the conglomerate of practices summarized under the umbrella term of 'non-idiomatic' or *free improvised music* (FIM) which provides a paradigmatic example of a dialogic approach to music making between

independent subjects in the pursuit of new musical or social interactions with a similar potential for transformation. The second important context is the diverse approach of *live coding* which focuses on real-time improvised interactions with programmable technology, typically in the written form of program code.<sup>26</sup> This leads to other fields of algorithmic and technology-based composition such as interactive computer music and experimental electronic music. These lineages are complemented by a selection of music practices which have a potential for reflexivity and transformation based on my personal experiences as listener, spectator as well as participating performer.

#### 1.4.1 Free Improvised Music

Following well-known guitar improviser Bailey (1992: 192) 'non-idiomatic or free improvised music (FIM) does not present a coherent aesthetic, rather it is 'a method of working (...) [which] won't necessarily indicate a particular style, or even presuppose an artistic attitude' (1992: 192). Corbett (2016: 137) continues that 'free improvisation is a method for making music; it isn't an independent value or quality'. The openness towards aesthetic results with a focus on process rather than product is how FIM informs the Dialogic Coding practice. In relation to the proposed three conceptual categories above, FIM typically operates simultaneously on the aspect of musical creation or composition as well as that of social interaction between musicians in performance.

Following 'improvisation' as an aesthetic category, Dialogic Coding is inspired by FIM's interest in the discovery of new and complex sonic textures through strategic exploration of the sonic material. This may explain the use of an atonal sound world in my Dialogic Coding practice (i.e. in *Technospaetzle*, *MYOB* or *SUM* projects). A well-known historic reference for this approach is the group *AMM* who pursued an equality of instrumental roles realised through a 'music in which the collective production of texture is a fundamental organising feature' (Bowers, 2002: 22). One effect of this 'laminal' approach (Prévost, in *ibid.*) being that any of the participants could lose themselves or their sonic identity in the laminal texture requiring that improviser to make clearly legible gesture in order to differentiate one's own playing from that of others. Dialogic Coding inherits from this approach particularly through the sound producing possibilities of the computer which

---

<sup>26</sup>I consider this the 'home category' of the Dialogic Coding practice for its conceptual proximity – even the examples presented as part of this research do not always or exclusively involve live coding interaction during performance.

effortlessly allow for continuous sounds with complex timbres. Such a form of aesthetically guided improvisation paradigmatically supports a state of immersion in the very activity of the sonic exploration. But it can also lead to the aforementioned disorientation in a group of improvisers which works against a dialogic encounter of other human improvisers in the group.

A second example provides a contrary approach: since FIM resists adherence to particular musical idioms (Kanellopoulos, 2011) it invites participants to develop highly idiomatic individual as well as group styles of playing. As a consequence, some FIM practitioners have developed a compositional practice which attempts to integrate these individuality of styles. The work of *John Zorn* is one such example. *Cobra*, part of his *game pieces* series, creates a scenario allowing for any combination of players to play at any time based on negotiations in the group. The concrete combinations take shape through the interaction between the individual improvisers and the 'prompter' – a kind of conductor of the group who negotiates individual requests for playing. The piece 'deal[s] with *form*, not with *content*, with *relationships*, not with *sound* (...) [with] musicians on the stage relating to *each other*.' (Zorn, in Brackett, 2010: 62), original emphasis). Consequently, a great importance and the dominant compositional influence lies in the act of 'picking the band'. Zorn observes how the playing becomes a negotiation of social issues.

I basically create a small society and everybody finds their own position in that society. It really becomes like psycho drama. People are given power and it's very interesting to see which people like to run away from it, who are very docile and just do what they are told, others try very hard to get more control and more power (...) [it is] a political arena in a certain kind of sense. (Zorn in Bailey, 1992: 78).

In this approach the dynamic creation of relationships between the participants of the improvisation becomes, too, the subject of the actions, not only the actual music that is being produced. FIM might thus be seen as a form of establishing a micro-society rather than as simply a metaphor for conversation (Horn, 2000; Monson, 1996). Improviser Joelle Leandre describes it adequately:

Three musicians meet, for example. It's a microcosm, a little society with its stakes, its tensions, its courtesies, its harshnesses, its silences, its fears, its powers (2009, in Fischlin et al., 2013: 21).

This proposes FIM as a concrete form of dialogic music in the intersubjective understanding of this thesis. Thus it serves as an interactional as well as conceptual paradigm for this practice in situations of group performance. To

paraphrase Leandre the society created through Dialogic Coding practice allows for direct, interpersonal exchange between improvisers as well as for retreating into the collectively created soundscapes. This microcosm offers various dialogic spaces for dwelling with a listening being attitude. Exactly how the relationships between improvisers are created depends on the role of the nonhuman agency, the apparatus in the creation of this space.

#### 1.4.2 Interactive Composing

Another lineage from which Dialogic Coding inherits is live and interactive electronic music in its various forms. It is not only the exchange between personalities or subjectivities of other human improvisers, but obviously also the exchange between the performer and her instrument which influences the process music making. The concept of 'interactive composing' (Chadabe, 1984) describes a 'method for using performable, real-time computer music systems in composing and performing music' (ibid.: 22). Here feedback is often used as a strategy or method in the real-time compositional process to determine the next actions. The feedback may be purely sonic (i.e. the sonic response from the space of performance) or it may already be built into the interactive instrument on the level of signal processing in the computer.<sup>27</sup> Chadabe (1984: 23) defines 'interactive composing' as a two-stage process in which an interactive composition system is first created (composed), and then interacted with as it functions (performed). The relevance of Chadabe's approach to Dialogic Coding practice becomes even more obvious when he goes on to explain the role of the computer in the interaction with the performer:

The performer (...) shares control of the music with information that is automatically generated by the computer, and that information contains unpredictable elements to which the performer reacts while performing. The computer responds to the performer and the performer reacts to the computer, and the music takes its form through that mutually influential, interactive relationship. The primary goal of interactive composing is to place a performer in an unusually challenging performing environment (ibid.).

*Interactive composing* can thus be regarded as the historic conceptual predecessor of Dialogic Coding practice in the pursuit of creating a challenging performer experience. My practice, however, pursues these goals through contemporary means, such as a high level programming language and small

---

<sup>27</sup>See Sanfilippo and Valle (2013) for a more complete overview of feedback systems.

external and reconfigurable MIDI controllers. Dialogic Coding departs from interactive composing in how it more fundamentally explores the different performers roles in performance. With the possibility to radically change the algorithmic structure in real-time, the performer in my practice acts (in Chadabe's terms) not only as a performer of the system, but also as a composer who is able to reinvent the structure with which she (almost) simultaneously performs. The goal of Dialogic Coding in extension of Chadabe's approach is thus to place not only the 'performer' in an 'unusually challenging environment' but also the 'programmer.'

Another example of an interactive, and instrument-oriented approach is the work of guitar improviser *Hans Tammen*. In the tradition experimental electronic instrument performance, such as Michel Waiszvisz' performing with 'the hands', he developed an interactive augmented version of Keith Rowe's prepared table-top guitar. Tammen describes a performance with his *endangered guitar* as 'a journey through the land of unending sonic operations, an interactive hybrid between a guitar and a computer' (Tammen, no date: online). The software patch running on the computer<sup>28</sup> (built in Cycling 74's software *Max/MSP*<sup>29</sup>) 'listens' to Tammen's guitar playing and determines the resulting computational processes based on the analysis of that input. Tammen describes this as an

*interactive instrument*, in that it draws different conclusions from my playing, and processes my sounds differently every time. That does not mean it is completely independent (and therefore fully unpredictable), but I use various grades of *controlled randomness* to surprise me during performance (Tammen, 2014: no page).

The amount of randomness built into the algorithms of the instrument he refers to as the 'independence' of the instrument which is adjusted to the situation of playing in *degrees of interactivity*.

In ensemble performances I allow less interactivity — my co-performers surprise me enough. In solo shows the computer became over the years more independent in its responses to my playing (Tammen, 2000: online)

This perspective points to the challenge of negotiating the different interactions in an improvised group performance: there is not only the programmed system with

---

<sup>28</sup>For more information and technical detail, see Tammen's description online: <http://tammen.org/musical-instruments-in-the-21st-century-endangered-guitar/>

<sup>29</sup><https://cycling74.com/products/max/>

which to interact, there is also the group of others with whom to possibly interact. This challenge has accompanied the development of Dialogic Coding practice from its beginning and resulted in the creation of different scenarios for each successive project, each responding to the previous experience. Much in the way Tammen proposes, the adjusting of interactivity, in other words the unpredictability of the performer's own instrument, becomes the central element to negotiate a balance between an improvising group and a partially independent technological setup.

#### 1.4.3 Live Coding

Shifting the focus of a performer's activity from performing with an interactive system to programming in real-time of performance leads to the practice of live coding. This approach brings the performer as programmer (thus as interactive composer) on the stage of performance. This field of practices is already over a decade old (Blackwell and Collins, 2005; Collins et al., 2003; Ward et al., 2004). What they all share is the focus on text-based HCI beyond any particular aesthetic. Magnusson (2014:14) provides an inclusive definition:

Live coding is the writing and performance, in real time, of music or other art forms, including games, where algorithms are the primary form of notation; it involves designing step-by-step rules for machines, humans, or others (e.g., animals or nature) to execute. An effective performance technique, live coding addresses the problems of improvisation and on-the-fly decision-making in live performance with machines.

In this definition there are the important aspects: a) the focus on the process of constructing an algorithm through improvised interaction and b) its relevance to (and inclusion of) all agents, human and nonhuman in the process. Thus, live coding can be said to create a space where a programming performer explores decision-making responsibilities by distributing them between human and nonhuman agencies.

Dialogic Coding borrows not only its name from this approach but the focus on the dynamic real-time construction of nonhuman agencies including the possibility to change these in performance, and in consequence re-compose the very situation of performance. The mode of creation –*bricolage programming*– is a direct inheritance from live coding practice. While in live coding performance a common scenario might be to start a performance from a 'blank slate' and slowly build up a more complex algorithmic structure (performance as the composition of a system

in real-time), Dialogic Coding typically begins with a pre-composed structure of some kind. In addition, the interaction with the algorithmic system in my practice is not as open-ended and purely guided by the developing music. Rather it follows the conceptual approach of dialogue, in other words: performing with an eagerness to find creative inspiration through challenge *in* the apparatus interfaces, the utterances of other improvisers, in one's own thinking *as well as* the emerging sounds. It may be added that the exploitation of error as a creative tactic plays an important role in Dialogic Coding. The apparatus is not only regarded as a tool which can be shaped and made to express the musical ideas of the programmer-performer but it functions as a source of inspiration and a partner which makes concrete a performer's musical ideas in a cocreative way. Dialogic Coding thus brings both modes of interaction together: that of interactive composing with that of live coding. The performer either responds to her composed setup as a functionary-performer, within the defined possibilities, or responds as a programmer-performer by means of re-writing the structure to the algorithmic system and thus changing again the possibilities for future interaction. One dilemma in live coding performance is the visibility of interactions between the nonhuman and the human agencies. Following TOPLAP's credo: 'Obscurantism is dangerous. Show us your screens' (Ward et al., 2004: 247) live coding practice projects the computer screen for an audience. However, this is not necessarily a sufficient means to communicate the intricacies taking place between the two main actors, human and computer, in this drama. *Wrongheaded*, a collaboration between live coders Click Nilson and Matt Yee-King, is an example with a more theatrical approach. In their concerts they emphasize performative aspects of coding in a typically humorous way, as they 'operate their bodies and computers at the mercy of live coded instructions, experimenting in algorithmic choreography' (Yee-King and Collins, 2012: online) Such aspects range from expressive dancing, to the presentation of kitsch objects and social media type of imagery while the performers simultaneously write SuperCollider code which defines the sound playing live. The aim is to 'unify human action and computation' (ibd.). In an event at the Networkmusicfestival 2012<sup>30</sup>, in Birmingham (UK), I personally experienced their quality of acting and provocations interrogating established ways of live coding performance. For example, the 'narrative' of the performance was

---

<sup>30</sup>A recording of the event can be found here: <https://vimeo.com/55313177>

reversed: instead of adding musical elements through a slow 'building-up' of code, Wrongheaded started from a full complex soundscape and arrived at silence. The group's focus on representative and entertaining performance actions has informed the development of the featured projects *Mind Your Own Business*, *SUM* (see chapter 2.3) and earlier 'grand gesture' laptop performances.<sup>31</sup> In each of these projects the stage setup conveys certain aspects of the human-apparatus relationships and thus emphasises the performers' roles as *actors* (see chapter 4.4 for more details).

#### 1.4.4 Participative and Networked Performance

Apart from the use of degrees of indeterminacy in the algorithmic structure of the apparatus, there are alternative means through which a form of autonomy and nontriviality can be introduced into performance. The lineage which exemplifies this is related to creating complex feedback structures on the interactional level, in other words: the construction of circularity in communication and interaction (cybernetics). From a system design perspective this equals the creation of interdependencies between improvisers through a technological medium or in other words, through a network structure which allows to create connections between the actions of each performer. In Dialogic Coding practice I have used two approaches: a) a networked structure to interconnect the improvisers through technology and b) participative music to include the outside audience into the interaction with the apparatus.

'Network music' typically refers to the approach of transmitting and receiving signals across a distance to make music together remotely. However, the same type of communication is also possible when all participants are co-present in one location. Drawing inspiration from the work of the pioneering computer ensemble *the HUB* (Gresham-Lancaster, 1998) network music, in this context, is used only in the form of a local area network (LAN) and not for a remote connection. The musical instruments share control or audio data in a local network with co-present performers. This setup allows to create complexity through interdependency. Communication takes place not only through sonic or visual gestures but through sharing computer data of any type. The laptop ensembles *BiLE*, *BEER* or *PBUP* provide relevant examples in the tradition of the *HUB*. While all these groups have

---

<sup>31</sup> A recording of this event, playing the 'hendrix instrument', can be found here: <https://vimeo.com/69715280>

programmed their own code architectures to communicate and share data in the network there are some differences in the compositional and performance approaches taken. On the one side there is a composer's approach found in the Birmingham Laptop Ensemble (BiLE). The group 'prioritise musical or artistic ability over technical proficiency and reject the need for rigid social organisation' (Booth and Gurevich, 2012: no number) in order to provide a space for each ensemble member to develop individual compositions. This stands in the tradition of the *HUB*. The Birmingham Ensemble for Electroacoustic Research (BEER) is further positioned into FIM territory exploring 'networked music systems for structured improvisation' (Wilson et al., 2014: 55).<sup>32</sup> While BiLE and BEER still use individual audio amplification for each ensemble member to suggest a personal relation between the sounds and a performer, this relationship is even further dissolved in the work of *powerbooks unplugged* (PBUP). While all groups use live coding as a technique to explore the pre-programmed system and written chat messages to discuss the course of the improvisations, there are, however, two significant differences found in PBUP's approach. First is the stage setup: They sit with their laptop computers directly within the audience and play music only through the built-in speakers. There is no stage and no amplification of the sound. Second is the radically shared and networked nature of the network architecture: every improviser can 'play' on any other computer remotely and all the code used to synthesize and sequence sound is shared – and can thus be copied by others.<sup>33</sup> This produces a disorientation for the listeners, and sometimes even for the players: it is not clear how the sounds heard relate to the actions of any of the performers. The personal authorship of a sound is 'delocalised' in the computer network (Rohrhuber et al., 2005; Rohrhuber et al., 2007). At the same time the 'unplugged' sounds played only through the small built-in laptop speakers call for a listening focus towards the quiet end of the loudness spectrum. The disorientation and the aesthetic quality of PBUP's performance approach propose a meditative type of listening oriented towards the inner.

---

<sup>32</sup>An example of a BEER performance can be found here: <https://www.youtube.com/watch?v=YhT600JYYE4>

<sup>33</sup>A recording of a PBUP performance (2009) can be viewed here: [https://www.youtube.com/watch?v=-ooTTNkVBDS&list=PLKRUGJaQJkAfU\\_PaYUeiRzD5qh5tkwvdj&index=9](https://www.youtube.com/watch?v=-ooTTNkVBDS&list=PLKRUGJaQJkAfU_PaYUeiRzD5qh5tkwvdj&index=9) and here: <https://vimeo.com/68073332>

This resonates with Dialogic Coding practice in how it brings a listening focus to the foreground. All of the three approaches outlined have inspired the strategic setting up of interdependencies in a network in order to facilitate dialogue (i.e. in the piece MYOB, developed in cooperation with BiLE). Sharing control data for parameters of any individual's sound processes is a means to introduce autonomy into each player's apparatus and thereby increase the unpredictability in the interaction. This works particularly well in groups with computer-based instruments who share the same data protocol.

Dialogic Coding does not dissolve the individuality of the improvisers as radically in the network architecture as PBUP does, but it pursues a negotiation of the individual human voices within and in relation to the nonhuman agency/-ies (which may be localised in the object of an individual performer's computer but are also delocalised in a shared data structure). The dialogic relationships on *all* levels, taking place in a triangular relationship between performer, other human improvisers and the nonhuman apparatus, provide the main orientation for navigating forward. In some cases this has resulted in the creation of a complete system to interact with (i.e. *SUM*), related to BEER's approach, in other cases the network only creates some interdependence between more autonomous nonhuman agencies (i.e. the *SUNPYX* project) – related to a BiLE/HUB approach. A related strategy to create circularity and interdependence is the participative approach which includes an outside (and possibly non-expert) audience. A relevant example piece here is '*Make a Baby*' by the group *Lucky Dragons* which I experienced at CTM Festival 2014:<sup>34</sup> Two performers on stage play an electronic setup which also has an interface open for exploration through physical contact by audience members. This works in the way that the ongoing sound can be altered or 'played' by connecting several loose ends of wires on stage through the human body. The more people join in the chain by touching each other (i.e. holding hands) a different sound may result. This setup is explorative and focused on discovery because the audience members first need to learn how to interact with this unusual interface and secondly need to find out what sound manipulations are possible. The encounter of this approach has inspired Dialogic Coding practice, particularly the development of the piece the *Projectionist Orchestra* as well as the compositional process of the *SUM* interface and its performance. Capacitive

---

<sup>34</sup>[http://www.ctm-festival.de/festival14/2014/01/31/sounds\\_like\\_kurenniemi/](http://www.ctm-festival.de/festival14/2014/01/31/sounds_like_kurenniemi/)

sensing technology (measuring the conductivity of human skin through touch) exposed properties similar to the effects of complex interdependent networks and thus offered a way to build nontriviality into the physical interface of the apparatus (in a system-as-composition approach). More importantly, viewed from a position of composition/programming, however, was the possibility to transfer the decision-making responsibility not only to the apparatus but also to a participating audience members and thus create new spaces for dialogic encounters. This step changes the relationship between performer, an audience, and the nonhuman agency by providing direct access to the apparatus system for performers other than the composer of the system. Effectively, the experience of the Dialogic Coding performer may be re-experienced in partial aspects (i.e. excluding the challenge of being able and responsible to alter the algorithmic structure).

#### 1.4.5 Post-Cagean Listening

The third aspect of Dialogic Coding emphasises reflexivity and transformation based on the subjective experiences of immersion and outsidership. This focus positions this practice within the post-Cagean tradition of American experimental music since the 60ies.<sup>35</sup> Dialogic Coding practice inherits from this tradition not only Cage's credo of indeterminacy and chance – a realisation of a piece 'is in some ways not pre-determined' (Collins et al., 2013: 129) and random elements are used to vary the result – but more fundamentally the focus on the listening experience as a means to reflect and transform the listener's own judgement, in other words: listening as challenge and a way of knowing.

On these grounds I introduce here particular practices which I have personally experienced in a transformative way, as listener and spectator: *noise music* performance. Collins et al. (2013: 141) describe 'noise music [as] one of those wide nets whose skirts can hold many interfaces to other areas', its historic origins reaching back to the works of composer Xenakis (i.e. *Bohor*, 1962) or even to the Italian Futurist Art Movement of the 'Intonarumori' (Russolo, 1986) who created acoustic instruments to (re-)produce the sounds of industrial work processes. Noise music performances celebrate the energy contained in the (typically atonal) sounds by confronting the spectator with its raw qualities in small audiences (Klett

---

<sup>35</sup>By this I refer to works such as David Tudor's *Rainforest IV*, David Behrman's *Runthrough*, Alvin Lucier's *I am sitting a room* or Nicolas Collins' *Pea Soup*.

and Gerber, 2014). This was the scenario in which I experienced a performance by British noise artist Ryan Jordan:

[I entered] his East London studio space, a dark and concrete wall space. The atmosphere during the event, one in a series of noise and other experimental concerts, was conspiratorial. All listeners seemed to be part of the event and also presenting. The place was turned into a stage setup in a very ad hoc-improvised manner, full of scrap elements, randomly organized shelves, empty beer cans, a cash desk improvised on a table. When he started it was pitch dark, all lights had been turned off. It started with an incredibly bright flash of light with a synchronous loud burst of noise, echoing off. And then again. And again. It was rather repetitive, but still my body seemed to be drawn to it, I could not get away. Even though the light was too bright to look at, the sound too loud and harsh to listen to! These extremely physical 'energy bursts' felt like a massage. I felt dizzy and lost, with my eyes closed while I was listening 'inside' of me, standing in the small crowd of people. The massive bursts of sound and light would start slowly and accelerate up to the point when it became a steady and pulsing rhythmic texture<sup>36</sup> (Hummel, 2013).

Based on this personal experience, noise music came to inspire the development of my practice following two of its leading aspects, that of 'raw physical sensation' and 'cultural dissonance' (Collins et al., *ibid.*). This influence may be identifiable in Dialogic Coding practice on an aesthetic level (the raw listening experience) as well as on a meta-level (to challenge traditional performer-instrument-audience relationships).

One example for a *culturally dissonant* noise performance is the performance piece 'monologue for taxidermy totem (conductive materials and electronics)' by Mexican sound artist and improviser *Mario De Vega*. In a 2014 solo performance<sup>37</sup> he used a DJ mixer and a cassette tape player, connected through some form of feedback not clearly visible to the audience. The devices were played by him in a 'mystical' attitude: He seemed to act as if he was part of his own circular setup and we, the spectators, were only watching it from a distance, as if looking through a glass window into his world. The performance was aesthetically intense and raw, at moments piercingly loud, but overall less physical than Ryan Jordan's. What I experienced as 'dissonant' here was his acting performance: He was dangling on his stool, surrounded by wires and strange 'shamanic' objects on the table, while operating the mixing desk through seemingly random gestures. The sounds

---

<sup>36</sup>A recording of a similar performance can be listened to here: [https://www.youtube.com/watch?v=CEm6jY1c9vQ&list=PLKRUGJaqJkAfIS7EXUfK8RpL1\\_s-ykeB\\_&index=15](https://www.youtube.com/watch?v=CEm6jY1c9vQ&list=PLKRUGJaqJkAfIS7EXUfK8RpL1_s-ykeB_&index=15)

<sup>37</sup>A recording of the same performance from 2013 can be viewed here: <https://youtu.be/jpjrYTCTLGg>

changed sometimes following his gestures, but not consistently. The performance thus created an irritation through its clear stage setup focusing on the performer and his instruments and the unclear causal connections between performative actions and sounds. In combination with the physical energy, DeVega's performance was engaging for me both on an embodied (listening, seeing) as well as on an intellectual level (observing, reasoning). Emmerson (2007) described live electronic music practice as a 're-animation of sound' and the composer as 'shaman' who is the director or master at the centre of this kind of ceremony. DeVega's performance literally demonstrated this potential of noise music to create such 'techno-shamanism'.

Dialogic Coding inherits from this a playful approach to creating performer-sound relationships mediated by and through the apparatus. The raw energy of sound in noise music emphasises in a very physical way the importance of listening as an influence for the compositional process. My practice does not necessarily employ extreme loudness but it pursues a similar intention: that of producing a dissonance in perception and expectation. In contrast to the rather technical postulation of live coding's 'Show Us Your Screens', DeVega's performance offered another way towards engaging an audience: through an other-wordly presentation of entangled performer-technology relationships. While Dialogic Coding practice is much less spectacular than this it has the potential to produce similar irritations and disorientation regarding the relationships between performer, gesture, and sound.

#### 1.4.6 Hacking and Glitch

The last lineage I want to mention in this section complements the above by focusing on Dialogic Coding as a reflexive practice – not through listening and observing but through direct and exploratory action by the performer. Hacking and Glitch are two terms which describe related artistic and musical practices which share this perspective.

Hacking (Erickson, 2008; Levy, 1984; McKenzie, 2004) the act of programming a computer, not only as a technological or functional activity but as an artistic practice in itself. Furthermore, the computer is viewed as a powerful tool to create new things with the goal of learning and increasing knowledge. The hacker is considered to be an enthusiast and explorer of technological possibilities who makes possible 'the possibility of new things entering the world.' (McKenzie, 2004: online). From an abstracted standpoint hacking 'is really just the act of finding a

clever and *counterintuitive* solution to a problem' (Erickson, 2008: 5). This approach of 'artistic' programming focuses on elegance in the pursuit and challenge of finding the best possible solution in a given technological situation. The computer code of any Hacking practice is thus not only an instrument to create but also part of the aesthetic itself. Dialogic Coding practice draws from this position in how it regards the activity of programming as an aesthetic act to value the program code in itself as a valuable product of performance (alongside the music and interactions). The interactions in performance are guided through the dialogic encounter with the apparatus and may similarly result in clever and counterintuitive solutions.

Hacking practices are however different in their claim of a political position of resistance. Putting exploration, learning and innovation over profit and efficiency (McKenzie, 2004) which uses the work and often the body of the artist to address, to interrogate and critique contemporary society, culture or politics. Software art focuses on the 'artistic activity which, in the medium —or rather, the material— of software, allows for critical reflection on software (and its cultural impact)' (Arns, 2005: 2) aesthetic and political subtexts of seemingly neutral technical commands. Dialogic Coding does not primarily follow this interventionist thrust to reveal, however it draws from the origins of the 'Do-It-Yourself-culture' (Friebe and Range, 2008; Holtzman et al., 2007) non-mainstream sub-cultural praxis. It employs accessible open-source technologies and shares its results openly with other practitioners – during and outside of performance. The building of a community around it is part of the practice itself.

A concrete application of a Hacking practice in electronic music is the approach of *circuit-bending*. Driven by the same goals of exploration and innovation, this activity uses pre-existing audio circuitry, such as sound producing toys, transistor radio receivers, miniature pianos, small drum computers and similar, and turns them into instruments to perform music. This is done by physically opening the objects and exploring their electronic circuitry, discovering new connections which generate new sonic results. Rheed Ghazala, the self-proclaimed 'father of Circuit-Bending', calls it an 'amazingly simple' process (Ghazala, no date: online) which can be done by anyone, without prior knowledge of electronics. It is a 'hands-on' practice accessible to all who have the curiosity to experiment and discover. Ghazala (2004: 104) draws the picture:

[c]ircuit-benders are, in a sense, ethnomusicologists exploring villages ringing with electronic folk music, indigenous in this case to circuit and time rather than people and time.

Similar to how the circuit-benders here travel the sonic world to discover new forms of sound, the 'dialogic coders' are in the pursuit of new forms of interaction as well as sounds, operating on both levels simultaneously. Their material is not in the physical form of discarded consumer electronics but in the form of program code, perhaps collected on the worldwide web, to experiment with, compose, and re-compose it in order to produce new results. The aesthetics of the result serve as guidance in musical form as well as in program code.

*Glitch* is a form of electronic music is another applied form of hacking by means of digital technology.

Artists have recorded their crashing computers, deliberately skewed sample values at high zoom in audio editors, connected outputs to inputs, loaded the audio editor program binary into an audio editor as if it were a raw audio file, and sought innumerable ways to subvert by interesting failure the normal behavior of the technologies essential to electronic music (Collins et al, 2013: 142).

The exploiting errors of user interaction or the system itself in a deliberate way is an artistic strategy shared with Dialogic Coding practice. In contrast to both circuit-bending and glitch music, however, the difference in my practice is that the technological material used in performance is not necessarily built by someone else than the exploring artist. Still all practices follow the goal of discovering new unexpected results in a practice-based and aesthetically-driven process. As such, Dialogic Coding may be associated with the loose term of 'experimental electronica' in how it attempts to discover new aspects about technology by subverting its uses and testing its boundaries (perceptually, structurally, socially). Guided by the dialogic principle the intention is furthermore to create and change relationships to technology and initiate transformations.

The potential of live coding to recreate any part of the technological agency on the fly during performance introduces many options for a 'hackability' of the musical instrument. Even though this brings flexibility, the problem sketched out above remains. In Dialogic Coding practice it is thus more difficult to perform in a 'hacking mindset' because all structures are created by the performer herself. It is difficult to subvert them effectively. As a consequence, this leads us back to the beginning of this journey, to the creative impulses gained from dialogic improvisation in a group

of improvisers: 'creative decision-making, risk-taking, and collaborative problem-solving' (Lewis, 2013: 255) which offer the potential to generate self-reflection and constitute identities in and as a result of the process.

Improvisation functions as a maker, and marker, of difference, an emergent, collective, processual, and modulatory agency in which freedoms are at stake, freedoms predicated on creation and expressive gestures that test the capacity of the commons to change, self-critique, and evolve new forms and social practices (Kannelopoulos cited in Fischlin et al, 2013: 19).

Whether or not this summary of the qualities of improvisation holds true cannot be discussed here – what is important however is the intention of the practice contained in this articulation: Improvised music is ultimately about creating the possibility for change, self-critique and development of new forms of social practices through interaction. For Dialogic Coding practice this includes both the human and nonhuman agencies. To paraphrase improviser Albert Ayler, Dialogic Coding may be performed to fulfil the following statement:

We, the performers, are the music we write and activate. The commitment is to understanding of life through technological entanglement. The objective is to purify ourselves in order to move us to higher levels of understanding.<sup>38</sup>

Dialogic Coding practice may in this way provide a means of creating identity through technological discovery and reflection about one's interaction with the apparatus.

### 1.5. Distinctions and Defining Features

The Dialogic Coding practice resonates with and draws on the traditions and performance practices as outlined above. However, it does not sit easily within any of these lineages but transgresses the disciplines. Similar to how Pinch and Trocco (2004: 308) in recurrence to Turner's concept of liminality have termed the synthesizer a:

boundary object, a liminal entity (...) [which is] neither here nor there (...) betwixt and between the positions assigned and arrayed by law, custom, convention and ceremony.

Dialogic Coding produces the apparatus – a comparably *liminal entity*. The practice itself may be termed *liminal* for it moves between the different practices

---

<sup>38</sup>The original quote by Ayler (cited in Fischlin et al, 2013: 26) reads: 'We are the music we play. And our commitment is to peace, to understanding of life. And we keep trying to purify our music, to purify ourselves, so that we can move ourselves-and those who hear us-to higher levels of understanding. You have to purify and crystallize your sound in order to hypnotize.'

and contexts, not exposing a clear or stable identity. The advantage of this position is to reflexively and critically interrogate concepts associated with each of these traditions. As a summary to this overview of Dialogic Coding practice and its artistic influences I list the defining features of this approach as follows:

- Any interaction between agents – human as well as nonhuman – follows the principles of dialogue to create the possibility for an exchange of personalities.
- This approach requires an attitude of *listening being* towards all partners and all technical media simultaneously. This affords the performer the skill of translating between different technological media and communicative codes in order to mediate between the different symbolic worlds of musical sounds, performative gestures and computer code.
- The nonhuman agencies (apparatus) participate in such dialogue based on the principle of nontriviality, or their ability to interact pseudo-autonomously. This needs to be prepared and subsequently changed through acts of *programming-as-composition*. Thus the interaction with the apparatus as well as the process of inventing it are both the subject of performance.
- Being the user as well as the inventor of the apparatus a performer in Dialogic Coding generates multiple specific responsibilities towards the emergence of a dialogic relationship. For each of these roles another skill set (or virtuosity) is helpful to reach the goal of reflexivity through practice.
- Based on the autonomy of the apparatus and the improvised mode of interaction a performer experiences states of outsidership and immersion and through these possibilities for reflection on interactions and self. Dialogic Coding can from this standpoint be considered an 'epistemological tool' with which a performer is able to generate certain kinds of knowledge from reflexivity.
- The main challenge for the performer in Dialogic Coding practice is to create a dynamic equilibrium in the triangle of: a) the group of independent improvising agents (humans or nonhumans), b) her own independent, yet programmable apparatus, and c) her own 'double-independence' – as a functionary-performer in interaction with that apparatus and as a programmer-composer who re-invents it.

## 2 – Methodology, Conceptual Framework and Practice

### 2.1. What is the Methodological Approach?

This research was conducted as a practice-based inquiry which enables the practitioner-researcher to 'position their practice within an academic context' with the aim to ultimately 'expose creative insights into the making of their work as well as making these insights into creative practices available to both academic and non-academic communities' (Blain, 2016: 81). Motivated by the obscurity of interaction concealed behind computer screens and the illegibility of computer code for non-specialist spectators in live coding performances, moving closer to this site of HCI was an obvious choice: to actually participate and develop a practice on my own. PaR provided the dynamic framework with which I could investigate the live coding practice by means of a multi-mode inquiry. Through this I was able to access that 'liminal space' between rational and embodied knowledge (Nelson, 2006: 108). Exactly how such knowledge in and through research can be articulated is at the core of the PaR initiative. Haseman (2006: 100) holds that practitioner-researchers normally construct 'experiential starting points from which practice follows'. In practice this means that research questions normally emerge after the practice element has begun. In what Haseman terms 'performative research', a third paradigm besides qualitative and quantitative research, the practitioner-researcher uses a mix of methods for her investigation such as 'reflective practice, participant observation, performance ethnography, ethnodrama, biographical/autobiographical/narrative inquiry, and the inquiry cycle from action research' (Haseman, 2006: 104). While Haseman holds the practical outcome, *not* in the form of discursive text to be most valuable, others have insisted on the importance of a written commentary to share and disseminate the research findings with an academic community (Nelson, 2006; Bolt, 2010). Through articulating the 'tacit' (Nelson) or 'praxical' (Bolt) knowledge in written form it is possible to expose the creative insights contained in the *processes* which have *led to the making* of the practice, the artwork – which is not the same as the knowledge about the work of art. Nelson's 'multi-modal research inquiry' refers to all these activities: the doing, the reading, the writing and the articulating. The purpose of the written submission he sees not in a commenting on or even worse, a transposition:

of the artwork from its own medium into that of words (...) [but] to assist in the articulation and evidencing of the research inquiry (Nelson, 2013: 36).

The inquiry and its insights should be *disseminatable*, that is, communicable to and shareable with both specialist and non-specialist audiences. Nelson (2013: 37) sees the purposes of complementary writing in PaR firstly in giving a 'clue' toward the research inquiry and secondly, in giving a 'clew'<sup>39</sup> toward the practice through its complementary and documenting role.

### 2.1.1 Nelson's Practice-as-Research Model in my Research Enquiry

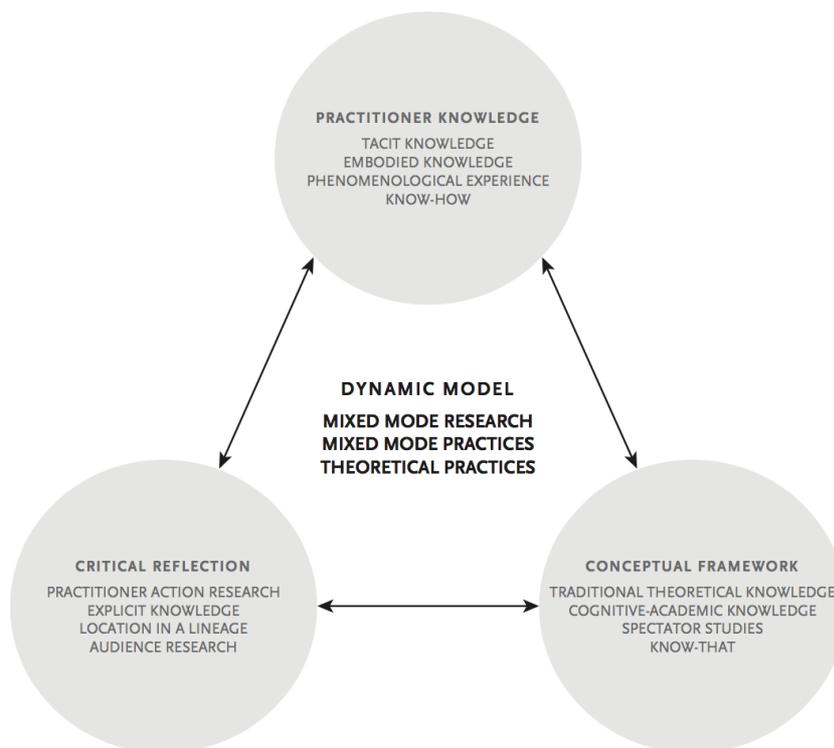


Figure 1: Nelson's multi-mode epistemological model' for PaR (2006)

This research follows the methodological approach proposed by Nelson. Thus I will briefly describe his 'triangular model' (see Fig.1) in more detail. He positions the 'arts praxis' within three areas of activity.

The three areas—the *know-that* or conceptual framework, *know-what* or critical reflection and *know-how* or practitioner knowledge (Nelson, 2013: 37) – are each pointing towards a different *kind of* knowledge.

<sup>39</sup>The 'clew' is Nelson's proposed term for specifically 'drawing attention to the thread of a researcher's doing-thinking' in the complementary documentation (Nelson, 2013: 11).

Firstly, my *know-how* was developed through and informed by studio-based learning, composing, improvising but further developed through rehearsing and performing in different line-ups. I identify my practitioner knowledge as the understanding for the potential inherent in code to build musical structures and how to practically apply this knowledge; how code objects can contribute to composition on a rhythmic, melodic and harmonic level and help develop my musical language. But the practitioner knowledge is also very much about the situatedness of this musical 'know-how': How can I prepare this material in a way to be able to use it in situations of improvised music with others?

Secondly, the *know-that* was approached through critical reading as well as my own experience of artistic works in the form of experiencing concerts and listening to recorded media. The most influential of these experiences were outlined in the lineages in section 1.4: FIM, live coding and noise performance. The 'conceptual framework' is informed by scholarly theories and concepts. In the development of the research-practice I have engaged with an array of conceptual perspectives. Some of these theoretical ideas have contributed significantly to my development, these include: dialogic theory (Buber, Bakhtin); related concepts from media theory (Flusser); second-order cybernetics (Von Foerster); improvisation studies (Fischlin et al) and will be delineated in section 2.4 below.

Lastly, in the important area of *know-what* I scrutinized my practice through critical reflection in order to further the development of the research project (performance practice and critical writing). I did this on the basis of field notes written before, during and after compositional processes, informal conversations held with audience members after concerts and through interviews with my musical collaborators, recorded during the duration of each of the projects presented in 'information sheets' in section 2.3. In addition, I recorded rehearsals and performances for most of the projects to analyse the interactions and forms of communicative exchanges taking place. These reviews enabled a comparison between the various projects of Dialogic Coding practice from a more distanced perspective in order to find out their shared aspects and better articulate my practice in this writing. Moreover, the recordings have helped to advance the practice itself by revealing those aspects which needed more work to become understood by an outsider. The recordings of the improvised sessions thus play an

important role for the development of Dialogic Coding practice, particularly for the critical analysis and setting of future development goals.

The PaR methodology relies on post-modern or post-structuralist thinking which fosters a sceptical and radical mode of thought. This provides not only an understanding of the factual world but as Nelson points out:

resonates with experimentation in arts practices insofar as play is a method of inquiry, aiming not to establish findings by way of data to support a demonstrable and finite answer to a research question, but to put in play elements in a bricolage which afford insights through deliberate and careful juxtaposition (Nelson, 2006: 109).

In this 'pursuit of knowing' it is worthwhile to embrace all possibilities (and methods) with the intention to find out how they '*resonate*' with each other through 'dialogic interrelation' (Nelson 2013:38).

### 2.1.2 Additional Input

In its early phase this research has had additional conceptual and methodological influence from the fields of ethnomethodology (Garfinkel, 1967, 1988) and science and technology studies (Latour, 1987, 2005; Suchman, 1987, 2007; Hutchins, 1995) without following their respective methodological programmes in a rigorous way. Both these fields may be regarded as groundwork for the multimodal approach of PaR. Ethnomethodology informed my research-practice through its emphasis on the question of how people *do things* and construct relations through this, particularly in everyday practices. These relations and their construction may only be revealed by a researcher's participation in the situation and activity. As this PaR project focuses on the development of a personal performance practice it is closely related to an *auto-ethnographic* perspective which provides the resources to

retrospectively and selectively write about epiphanies that stem from, or are made possible by, being part of a culture and/or by possessing a particular cultural identity (Ellis, Adams and Bochner, 2016)

In the dialogue with these readings, particularly the works of Lucy Suchman (1987, 2007) and Antoine Hennion (2003, 2007) the importance of *reflexivity* in my practice was revealed. This proposed a relational point of view and paved the way for a conceptual framework of dialogism as a way to critically reflect on and better understand my research-practice. The reflexive construction of meaning as well as

aesthetic (musical) form in improvised performance lies at the core of a dialogic approach to programming.

## 2.2. Developing the Practice

In this section I will briefly present how I have developed the practice within this PaR project. As evidence for my thesis I have selected a few examples as cases to be discussed in the subsequent chapters in relation to my conceptual framework (see section 2.4). Here I will introduce the path which led to these example projects (section 2.3) in order to give an impression of their poesis and aesthetic. This serves to highlight how the developing practice has helped my insights and conclusions to emerge in the course of the research. As Nelson (2013) states the key to accessing the knowledge of the 'doing-thinking' is a rigorous process of documenting, close observation and critical reflection in an ongoing, dialogic and cyclic interaction. The critical reading of aforementioned philosophical and theoretical texts has helped me to better understand the dialogic principle, as well as to identify this in an applied form within my practice. With each new musical collaboration I had a further developed and informed understanding of my own improvisational abilities for dialogic coding. At the same time each such encounter introduced new challenges i.e. new roles and tasks in preparation and performance, unfamiliar technological setups, and unexpected performance by collaborators. Reviewing my experience of live concerts and rehearsals, the moments of failure were particularly insightful and helped to re-adjust my focus into the inquiry. I did this through a method of recording performances audio-visually and analysing this data through my analytical framework introduced in section 2.5. These recordings were complemented by notes taken during the process of rehearsing and performing a particular project. The dialogue between my conceptual framework and the critical reflection of my practice has helped me to develop a more informed position as a practitioner. Additionally I was able to advance my understanding of the practice by participating in conferences as well as experiencing concerts or club events of other practitioners. The challenge being the object of the research as well as the subject analysing it was certainly to maintain that 'reflexive' condition which acknowledges the subjectivity of a researcher's own judgement while articulating insights which are valuable to the field. I achieved this by returning to critical reading and reflection in order to view the praxis from a distance. Following Nelson (2013: 8) that the 'key method of

inquiry' is the practice itself, any insight gained in the process consequently resulted in a new iteration of further development. This was realized also by sharing the work with others and opening it up to their participation and responses through conversations after performances as well as discussing the recordings of performances with my collaborators.

### 2.2.1 Trajectories Through the Multi-Mode Inquiry

My phd 'journey' advanced in interleaved periods of learning in these stages: a) the live coding practice in the studio (algorithmic rehearsing), b) jamming and performing with other musicians, c) reviewing these practices and d) critically reading in relation to my developing conceptual framework. In my developing practice I was able to identify three main approaches towards a dialogical performance approach:

1. **Strand #A** or 'free sessions' is the approach of participating in free improvised music sessions
2. **Strand #B** or 'rehearsed collaborations' is the approach of working with other artists (mainly laptop-based) for a longer period of time (between 2 weeks and 2 months)
3. **Strand #C** or 'performing with pre-programmed systems' follows Chadabe's approach of *interactive composing* in which an interactive system is composed/programmed first in the studio and then performed with live

These categories did not emerge in a strict linear or chronological fashion but rather, they developed from each successive performance experience. Initially I began the practice during free improvised sessions. I therefore needed to prepare myself for such a context. This took the form of learning and practising to live code, mainly at private studio rehearsals and through practical exploration. Only after an initial period of performing in free, improvised sessions, did I diversify my approach. As a reaction to constantly changing line-ups, I began to work with other performers for longer periods of time in a rehearsal-based approach. Thirdly, as a consequence of an overwhelming number of musical choices limited only by each participant's instrumental abilities, I began to compose/program interactive systems to be played collectively rather than by a single performer alone. This is not to say that I gave up playing free sessions altogether, the practice developed rather in the form of extending my activities into all three of these areas in parallel.

### 2.2.2 Adopting New Directions Through (Mid-)Course Corrections

Reviewing the course of my developing practice in regards to the insights it produced, the most significant changes in all these practice 'strands' was a shift from group-oriented communication and performance toward a more self-oriented standpoint. At the early stage of my inquiry I was focused on 'how (...) performers of laptop ensembles communicate with each other'. This changed to focus on 'how a performer communicates with the laptop and other performers.' The data I could access from this perspective (i.e. my own notes and critical performance reflection) were better able to support an informed practice. Furthermore, from this position I was better able to integrate the theoretical ideas from my conceptual framework into the developing practice. In a group-oriented scenario every idea will be negotiated with all other practitioners which may lead away from initial ideas into new directions (not necessarily a bad development). Similarly, the strand C which introduces constraints in how music is improvised and performed may be regarded as an attempt to focus on particular concepts in regards to interaction and improvised performance by transforming them into varying degrees of fixed forms (concealed into the algorithmic structure).

One consequence of this programming-as-composing approach was a shift in the roles for myself as a performer. With this I was able to gain insights into the affordances of programming which greatly extended my performer's perspective. At the same time the character of the live performance changed: not all elements and parameters of the music or interaction would or could be changed during a performance, but a performer needed to work within the programmed constraints. In other words: the meaning of performing changed in relation to the specific interface object. This change in the computer interfaces (moving away from the keyboard as input only) can be considered a result of feeding back the experiences of both strand #B and #C into the development of the practice. While the early sessions mainly used live coding techniques with the built-in keyboard, screen and touchpad of the laptop, the later performances resulted in interfaces which explored other forms of interaction (i.e. more gestural or tactile). Finally the practice changed not only the ways in which to interact with, and relate to the computer (interfaces) but also who was able to do this. While in the free sessions each improviser plays for herself with an orientation toward the group, in the other strands there are possibilities for more performers to interact with the same

system or in which the individual instruments are already more entangled and inter-dependent.

### 2.2.3 A Family of Practices

As I have outlined the practice in this research project developed in multi-faceted ways within three structural categories. This diversity is unified by the conceptual framework which spans across all these approaches. Consequently, the concept of a dialogic approach to live coding presented in this thesis is grounded on evidence from different places within my practice and is the result of not one particular project but the complete development within this research. This is to say that one element of the dialogic approach might be evidenced in a 'strand A project' while another might be found in a 'strand C project'. My practice as the sum of the different musical collaborations and compositional approaches, is a 'family' of (related) projects rather than a single coherent stream. What all projects share is the *umbrella* of the dialogic principle.<sup>40</sup> The particular differences will be highlighted in the later discussion. The common aspect found in all my projects is the focus on the role of the apparatus, more specifically in the use of the tools (the programming environment) for the development of my musical ideas. Logically, my own coding abilities are directly indebted to the SuperCollider language and all of its authors as well as to other live coding practitioners who shared code publicly. Another common feature is the involvement of my own person – however, as outlined in the previous chapter, with varying roles afforded by the different constellations and systems in performance. However, Dialogic Coding practice as it is presented here, does not include solo performances (even though much preparation time spent in the studio is solitary – solely with the nonhuman partner in dialogue). I ruled out the solo performance setup mainly because it had frequently been explored in the live coding community already and more importantly, because my research inquiry intentionally included the influence of other human improvisers response to my own actions. In the next sections I will outline in more detail what influence each one of the 'strands' played in regards to my research methodology.

---

<sup>40</sup>Similarly, if experienced live coding practitioners might wonder: While live coding techniques are at the basis of my improvisation/programming practice, they are not necessarily part of a live performance in all the examples mentioned here.

#### 2.2.4 Free Sessions as Laboratory

Throughout this research I have played in a total of about 25 different free sessions. Playing in such sessions is also referred to as jamming. In jazz practice jam sessions are considered 'venerable' by the community for they provide:

informal musical get-togethers [in which] improvisers are free of the constraints that commercial engagements place upon repertory, length of performance, and the freedom to take artistic risks. (Berliner, 1994: 42).

Some of the sessions I participated in arose spontaneously (i.e. on MMU Cheshire campus) while others were embedded in a regular and recurring structure. The ceremony is usually open otherwise there may be a predefined structure of who plays.<sup>41</sup> The improvisational aspect of jamming is what sets it apart from other musical practices. The session gives the opportunity to explore individual (instrument-related) and collective sonic spaces by bringing the skills and experience of each improviser to the table. Thus a jam session as an open structure provided me with the possibility to develop not only my improvisatory musicianship but also my algorithmic performance abilities through social practice. It is for this reason that I chose to put myself in such a situation. To a session I would usually bring the code that I had last worked on, together with a controller or interface. The testing of a performance patch developed for a specific scenario would therefore also contain the character of the session.

As common features of the 'session scenario' I identified:

- An intimate environment: a session was held usually in a small rehearsal room or studio space with only a very small co-present audience of up to 10 people or none at all.
- Rules/strategies for playing: there were no mutual constraints toward what and how to play. The only constraint in this sense was what I had defined by programming it in preparation.
- Unrehearsed: the participants of a session did not play together before on a regular basis.

Because of the exploratory and experimental character within a public context I did not document or record my session activity in full detail. The insights gained from

---

<sup>41</sup>I participated in the regular improvisation event 'The Noise Upstairs' in Manchester (UK) organised by Anton Hunter and Rodrigo Constanzo. At this session the participating improvisers are divided into smaller groups of three and expected to play for a short amount of time each (up to 5 minutes) in order to give everyone the opportunity to play in the course of a 2-3 hour session night.

participating in jam sessions directly influence my developing practice (i.e. by contributing new sounds and ways of playing to my live coding skills) or took the form of critical reflection based on notes and memos I had written down afterwards.

The experience of session playing has strongly shaped my understanding of how 'responsive' improvised interaction may be and what it affords the performer, particularly with an algorithmic setup. The main challenge to my practitioner skills arising from this was that of responsiveness and immediacy: I frequently experienced myself as too slow in response to the group interaction. It was only through the critical readings of my conceptual framework that I discovered how to deal with this adequately; Namely, by adjusting the focus within my dialogic relationships, or in conceptual terms: to value the dialogic interaction with apparatus more in the same way as those with the improvising group. In a concrete way, this meant not to respond to rather quick calls by other improvisers but take a step out to *listen* more attentively – also perhaps into the direction of the apparatus and what its live interfaces suggested – to then develop a more profound response.

At the same time, a consequence of the perceived need for immediately responding action during group improvisation was the introduction of other interfacing techniques into my apparatus setup. As outlined above I began with a purely live coded 'white screen' approach<sup>42</sup> which slowly diversified into a multi-modal setup which integrated possibilities for diverse interactions: coding (the laptop's built-in keyboard), touch (Apple Magic TrackPad), tactile fader and knob control (KORG Nano Kontrol II MIDI controller) as well as gestural interaction with arm or body movement (gyroscope/accelerometer data of a smart phone). This provided me with a greater flexibility to respond to the actions of the collaborating improvisers on all levels: musical-structural, expressive gestural or the intimate instrumental.<sup>43</sup> In the words of Stowell and McLean (2013: 3) 'the combination of continuous organic expression with symbolic abstraction'<sup>44</sup> has advanced my

---

<sup>42</sup>See the *Drums'n'Algorithms* project for an example

<sup>43</sup>By 'intimate instrumental' I mean the ability to play a musical instrument in nuanced ways; not necessarily through expressive gestures but with rather small and intimate actions.

<sup>44</sup>'Symbolic abstraction' may be understood here to mean program code as a form of abstract symbolic notation for music.

performance practice 'to provide immediate multimodal access to the multiple levels of musical ideas.'

### 2.2.5 Rehearsing Dialogue

As mentioned above it was difficult to successfully record the relevant interactions in public performances, not only because of a co-present audience, but also because I was fully occupied with my own coding performance leaving little attention for managing recordings in parallel. Accordingly, I began to create situations in a more controllable context working with the same partners for longer periods. Given that the principle of dialogue in intersubjective relations is considered to be of a fleeting quality and cannot be arranged to happen (Buber, 1965a), the reasoning for changing the setup was that a more stable line-up would allow me to explore the interactions more profoundly in the direction of dialogic relationships (i.e. by advancing my understanding of such interactions with each successive rehearsal and/or performance).

### 2.2.6 Creating Dialogic Spaces

Another methodological conclusion I gained from both the session experience as well as my first rehearsed collaborations was that only in a scenario of approximately equivalent musical and technical practitioner know-how could a dialogic and empowering interaction develop. In the previous scenarios the emergence of such dialogue had been hindered by various factors. On the one hand the technology itself stood in the way because it would neither operate as expected nor contribute or trigger new ideas for the creative process when interacting with it. On the other hand the social situation of the group proved difficult because there was a lack of listening and responding to each other. My response to this problem was to introduce constraints in the form of programmed structures into the improvisational interaction which were beyond the control of any individual performer. On the basis of my developing programming skills I designed interactive systems for music performance which would provide specific scenarios in which theoretically anyone could perform. My assertion behind this was that with a limited amount of decisions to take, the interaction in performance might expose more dialogic qualities. In practice the projects are not necessarily accessible to

anyone. They still require 'kairos' and 'metis'<sup>45</sup> – a feeling for when to act in an improvisation and the ability to seize those moments. In some cases a minimum knowledge of programming or live coding in the SuperCollider language is required.<sup>46</sup> The point of designing interactive systems within my practice was the idea for the performer to be less occupied with intellectual reasoning during performance but more with the physical and social interaction. The site of interaction shifted from being mainly 'behind the screen' of the laptop – the virtual space of programmer and apparatus – into an area which extends between all performers or participants of an improvisation. Thus in two cases (PO and SUM) there is no longer literal live coding performed, but the interaction happens in a gestural way within a pre-programmed frame. These projects have contributed significantly to my imagination of a *dialogic* space going back to Buber's concept of the 'interhuman' (see section 2.4.1.3). For Wegerif (2011: 180) such a space refers:

to any multi-dimensional map or graph, like the idea of a 'Search Space' in computing which is the set of all possible solutions for a search (...) or the widely used idea of a Design Space, which maps the many dimensions required for the design of any given artifact.

Wegerif follows Bakhtin (1986: 171) when he continues that the important difference between a dialogic space and other kinds is that:

the specification of the exact meaning of each position in a dialogue depends on succeeding utterances and so can never be closed down. In other words 'Dialogic Space' is more of a dynamic continuous emergence of meaning than a static 'space' (2011: 180).

The rich interface of an open programmable system (such as the audio-oriented programming environment of SuperCollider) facilitates the creation of such spaces by providing many tools for interaction and the generation of sound. In my practice the focus was on the creation of dialogic spaces in the form of new interfaces in which not only the human performer interacts with the agency of the nonhuman apparatus (the programmed system) but may also interact with other human performers (in the case of collective or shared instruments).

<sup>45</sup>The greek 'kairos' refer to an opportunity or propitious moment for an action or a decision to act within an ongoing stream of actions or time. It describes a moment in time or a 'timeliness' which may be seized through 'metis': a 'form of wily intelligence capable of seizing the opportunities' (Cocker, 2013: 74).

<sup>46</sup>The piece *MindYourOwnBusiness* is one such example. See the information sheet in section 2.3 for more details

## 2.3. Overview of Selected Cases

### 2.3.1 #A Free Sessions

# STRAND #A: FREE SESSIONS

## Drums'n'Algorhythms (Crewe)

**Date:** 20th June 2013

**Participants:** Mike Walsh (drums)

**Location:** MMU Cheshire, Crewe

**Duration:** 15 min



**Link:** <https://vimeo.com/71763130>

*Configuration:* The stage was set at the entrance of the contemporary arts building. The event took place as part of an end-of-year performance presentation by theatre students. MW played drumset with a few objects to damp or extend the sound quality. I live coded on a laptop without any other controllers and amplified through a speaker.

*Tech Setup:* I used a collection of synthesised percussion sounds and rhythm patterns developed in two studio sessions before. This provided the material with which he improvised.

*Structure:* The performance was free improvised. We had agreed to start from a very basic rhythm and then complexify successively.

## Kuehlspot sessions (Berlin)

**Date:** 13th+14th Nov 2014

**Participants:** Hui-Chun Lin (cello), Gerhard Gschlössl (trombone), Shi-Yang Lee (piano), Gioele Pagliacci (drums)

**Location:** Kuehlspot gallery, Berlin

**Duration:** 30–45 min



**Link:** <https://www.youtube.com/playlist?list=PLKRUGJaQJkAc3GWzIMj9uNiqGBUq8uKGV>

*Configuration:* A quintett of five different instruments seated like a regular ensemble in front of the audience. The venue is an art gallery space with high ceiling and pictures hanging from the walls.

*Tech Setup:* I had the laptop to live code and generate all sounds, extended by an 8-channel fader control through which I directly controlled effects and volumes of the sounds and one external wireless touchpad allowing for gestural interaction with synthesized sound processes.

*Structure:* The performance was free improvised. All performers had a background in jazz and non-idiomatic improvisation. The session consists of two halves.

## SUNPYX sessions (Berlin)

**Date:** 18th July 2014

**Participants:** Ruben, Liang, Paul, Martin, Annika

**Location:** Berlin, University of performing arts (UdK)

**Duration:** 8 min

*Configuration:* Six performers with laptops were sat in half-circle around a table on stage. The event took place in the patio of the university building.

*Tech Setup:* Each player used one laptop and some external controllers. Every used their own prepared sound 'set-up'. I used the laptop and an external wireless touchpad to control three different synthesis sounds.

*Structure:* The performance was structured by a ball of yarn which was stretched out between all performers. The 'pushes' or 'pulls' each player 'received' or initiated into the yarn net were interpreted as cues to play by.



**Link:** <https://www.youtube.com/playlist?list=PLKRUGJaQJkAf0OSIcQvRtwT-eYkNo4nOI>

Figure 2: Information Sheet #A Free Sessions

### 2.3.2 #B Rehearsed Collaborations: Technospaetzle

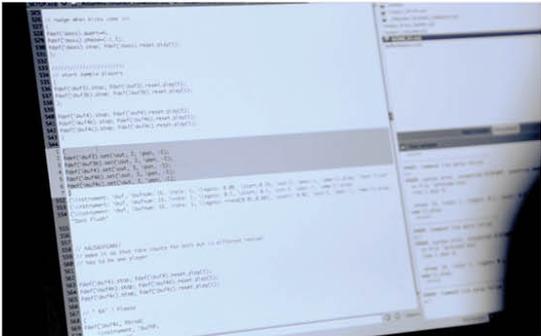
# STRAND #B: TECHNOSPAETZLE

**Date/Period:** 03/2014-07/2014  
**Participants/Instruments:** Tom Orr, Jonas Hummel  
**Location:** rehearsal/studio space, Berlin  
**Duration:** 37 min for 6 tracks

*Configuration:*  
The two performers faced each other. The audience was sat around them. The laptops were amplified through a stereo pa system. The venue was very so that the audience could see all performer actions at close distance. We used visual communication for some cues and announced the beginnings of tracks.



*Tech Setup:*  
Both laptops are sharing a MIDI clock via wireless network. Each sends 4 audio signals to the mixer. In the mixer, all 8 channels are mixed live and sent to effects. My setup is focused on live coding interaction with prepared sounds (for bass, percussion, vocal samples and atonal drones). This is extended by an 8-channel fader control, a wireless touchpad and a gamepad. I switch between these hands-on controls and the activation and live coding of patterns through the computer keyboard.



*Structural idea:*  
The performance was developed in a series of free improvised rehearsals into the fixed form of six tracks of about 5-6 minutes. The elements in each track (sounds and patterns) were prepared and activated in a coordinated way, the sections of the tracks have no defined length and allow for flexible transitions. The technical setup allowed for my sounds to be mixed live by my partner and manipulated through effects. My partner's role is concerned with creating the musical form while I provided sonic elements and timbres to play with.



**Link:**  
(All 6 Tracks) – <https://www.youtube.com/playlist?list=PLKRUGJaqJkAepPLO98g0Ovjd4lqvO7fnw>

Figure 3: Information Sheet #B: 'Technospaetzle'

### 2.3.3 #C1 Networked Piece: MindYourOwnBusiness

## STRAND #C1: MINDYOUROWNBUSINESS

**Date/Period:** 12&13.Nov.2013 + 27.09.2014 + 15.07.2015

**Participants/Instruments:** Holger Ballweg, Les Hutchins, Jonas Hummel, Shelly Knotts (Birmingham Laptop Ensemble)

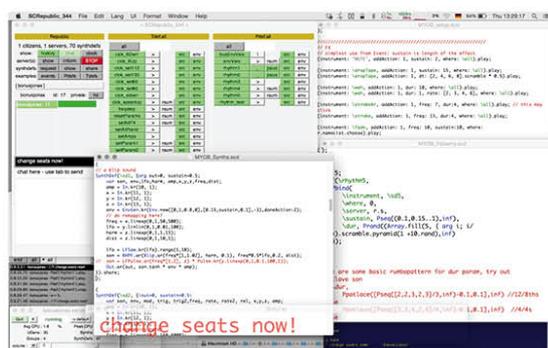
**Location:** London, Crewe, Birmingham, Leeds

**Duration:** 13-14 min

*Configuration:* The four performers with laptops are sat around a table facing each other. The audience is sat around the performers in a half circle. During the performance the performers 'rotate' seats and change their position on the table while the laptops remain fixed. The sound is projected from a quadrophonic pa system.



*Tech Setup:* All laptops are connected through a wireless network and send their sounds to one pa speaker each. All performers are live coding, one laptop additionally provided a fader controller to interface. The piece provides a set of three synthesis sounds and patterns as well as audio effects which are the basis for the improvisation. The laptops synchronise and 'share' code in the network so that any performer action may affect all machines simultaneously.



*Structural idea:* MYOB attempts a 'horizontal' distribution of musical responsibilities in the group, e.g. one player plays all melodic elements while another plays rhythmic – across all instruments. But it is interdependent in that a sound will only become audible if all players are active. In the course of performance the 'responsibilities' are rotated in the group. Beyond these 'rules' the performance is free improvised based on the sound material provided.

**Link:** Crewe event – <https://vimeo.com/90767434>

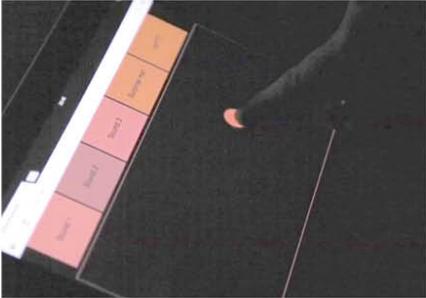
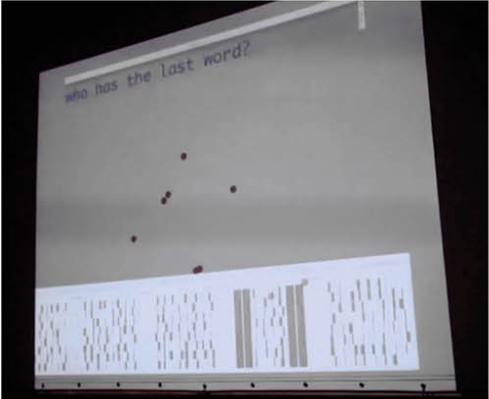
Figure 4: Information Sheet #C1 'MindYourOwnBusiness'

### 2.3.4 #C2 Participatory Piece: Projectionist Orchestra

## STRAND #C2: PROJECTIONIST ORCHESTRA

**Date/Period:** 2nd Feb 2014, 10th Oct 2014, 10th May 2015  
**Participants:** Audience  
**Location:** CTM Festival Preview, NK Project premiere, Originaltöne Festival (all Berlin)  
**Duration:** 22-25 minutes

*Configuration:* The audience was sat in in front of a projection screen with the performer on the same side with his laptop and a fader controller. The room of the venue was very dark in order to see the screen. The audience was invited to participate with their individual handheld computing devices or tablet computers via a wireless network. The sound was projected from a large pa system in the front. On the projected screen participants would see a visual feedback (a coloured dot) representing their own as well as all others' activities. Floating on top of this was a text window for chat communication.



*Tech Setup:* One laptop as a communication server (via nodejs + OSC) together with a router provides the WIFI network with which the participants connect. The server forwards messages to my laptop which controlled the sounds. I interacted through a fader controller to adapt the volumes of sounds and through the keyboard to send text messages visible on the projection screen. Each participant performed on a two dimensional (x/y) touch interface with 3 touch buttons. My laptop generated sound as well as visual feedback from the live control data. The mapping of action to sound was a combination of 1-to-1 as well as a complex, probabilistic mapping.

*Structural idea:* The performance was free improvised based on the sonic and interational possibilities of the system. The participants were free to select and play 1 of 3 different sounds. The piece has three main sections: It starts from a 1-to-1 mapping with only individual, pointillist voices, morphs into a complex mapping with multiple players influencing more sustained sound processes collectively and then finishes with a combination of the two. During the performance I used the text messages to comment on the activity and propose changes or announce the beginning of a new section.

**Link:** NK Performance 2014 – <https://vimeo.com/114578144>

Figure 5: Information Sheet #C2 'Projectionist Orchestra'

## 2.3.5 #C3 Interactive Piece: SUM

## STRAND #C3: SUM

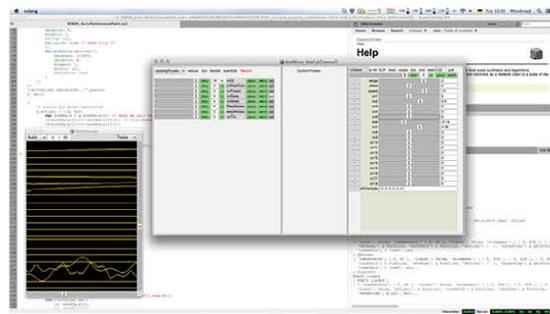
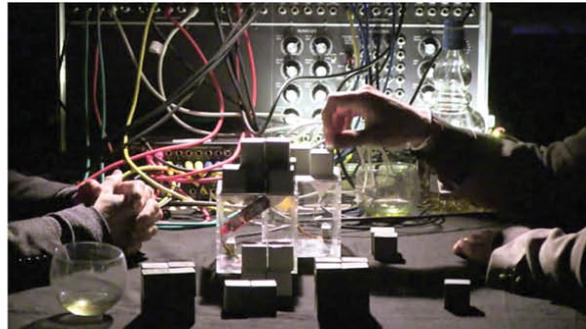
**Date/Period:** 12th Feb + 9th June + 21st Nov 2015

**Participants:** Nikolas Lefort, Jonas Hummel

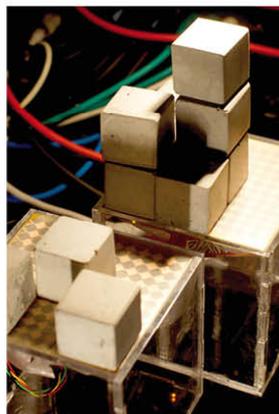
**Location:** LEAPknecht, designtransfer UdK, Spektrum (all Berlin)

**Duration:** 12-15 minutes

*Configuration:* On the stage the two performers wearing dresses were sat at a table facing each other. In the background was the analog modular synthesizer which produced sound. On the table were two cubic touch interfaces and a number of concrete cubic game tokens. The focus of the lighting was on the table, everything else in darkness.



*Tech Setup:* The cubic interface provides a conductive surface which allows for interaction through touch or with the game tokens (containing ground iron). The capacitance of the touch measured on the cube control various control voltage (CV) oscillators generated by the hidden laptop computer. These signals are sent through audio hardware to the modular synthesizer where they control a complex sound patch generating the sounds. The complex, multi-level setup does not allow full control for the performers.



*Structural idea:* The performance is free improvised based on the possibilities of the system. The actions of the performer follow a strict turn taking to add or take away game cubes. The performer's actions are performed in a dramatic posture reminiscent of a competition. They follow the intention to discover new 'sonic constellations' or change the sonic development into new directions.



**Links:**

Designtransfer event: <https://www.youtube.com/watch?v=eTjekBrzngQ>

LEAPknect event: <https://www.youtube.com/watch?v=DFDQkyjEtSA>

Figure 6: Information Sheet #C3 'SUM'

## 2.4. The Conceptual Framework

In the following section I present what Nelson (2013: 14) refers to as the 'conceptual framework' in PaR methodology. This is the theoretical horizon which is 'imbricated within practice' to tease out the tacit knowledge in the 'thinking-doing' of the research-practice. It picks up from the introduction of terms in chapter one and expands these into a complete framework of dialogism underlying the research inquiry. In this framework conceptual ideas from diverse backgrounds are brought together. They centre around the idea of dialogue, understood as an open-ended encounter with the other, in respect and curiosity towards generating new perspectives of understanding. This idea applies not only to other human beings but more importantly to programmable technology as well. The mode of being and observing required for such an encounter is a relational or reflexive one. One which does not identify absolute or 'objective' truths but which seeks to build connections to the research object regarding one's own position, knowledge and history. This perspective resonates with some conceptual traditions of academic discourse such as the postmodern and poststructuralist turns, or the reflexive standpoints of ethnomethodology and the 'sociology of associations' (Latour, 2005). Common in all these perspectives is the intention to recognize and integrate the influence of *all* participating actors – whether human or nonhuman, observer or observed – towards the consequences of an interaction. The consequences of such *dialogic* encounter may initiate transformations for the individual in both directions, toward 'inner' (self) and 'outer' (group) aspects of performance:

- inner: towards self-conception (identity), intellectual thinking (creativity and learning) and subjective experience of process (flow)
- outer: awareness of each other (listening focus), aesthetic process/product (collective improvisation/emergent) and ethical implications for action.

As a summary of the conceptual framework, I introduce an analytical framework which I will use for the discussion of the Dialogic Coding practice in the following chapters.

### 2.4.1 Buber's Philosophy of Dialogue

The 'philosophical anthropology' of Martin Buber provides the cornerstone of the conceptual framework because it focuses on the conditions of existence in the *individual* person with their thoughts, emotions, actions and responsibilities and

how these are reflexively constructed. It is thus well suited to analyse the highly individual processes taking place in creative arts performance via exploratory and experimental routes. Buber originally developed his concept of dialogue as a religious philosophy which was ultimately directed toward an encounter with the divine 'other', but he also applied it in the context of education. Consequently, his ideas need to be adapted to the local situation.

#### 2.4.1.1 The Dialogic Principle

Buber's basic assumption understands dialogue as a holistic concept through which the completeness of the other can be encountered. Even though dialogue is commonly understood as synonymous to conversation, Buber defines the term more narrowly. As introduced before, Buber (1965b) emphasises that the peculiarity in a dialogic relation or relationship lies in the fact of realising the difference of the other. Thus, in a conversation one may hear what the other says and reply with one's own utterance, however dialogue requires the listener to identify the other (and her standpoint) as different from oneself and accept it in exactly this way. This implies being more open toward what someone else has to say and not to impose one's own language, concepts, and interpretive schemes on the other. This kind of relating requires a very attentive kind of listening. Gordon (2011: 207) calls this a 'genuine listening' through which the other is encouraged 'to create his or her own meanings, which may be very different from one's own.' What is important to note is that dialogue does not take place within each of the participants nor in a world that binds them together, but rather between the participants in a realm that is shared only by those same partners. Buber (1957) also referred to this as the 'interhuman'.

#### 2.4.1.2 The Twofold Existence of I-It and I-You: Distance and Relation

Buber developed the concept of dialogue on his model of human existence as being 'twofold', in one of two modes of relating: The realm of 'I-It' and that of the 'I-You' (Buber, 1970: 53). These 'primary words' or 'word pairs' as Buber calls them do not signify anything, but rather they point toward different modes of relating to the world. Without speaking these 'words' nothing would exist (ibid.). In the world of I-It exists the subject-object relation, the relation of persons to things as objects, and to other persons as objects. This is the realm of 'experience'. Through acts of consciously directing attention towards objects we receive something from them. But this is an act of separation. It introduces a distance between the person who observes and the thing itself.

The man who experiences has not part in the world. For it is 'in him' and not between him and the world that the experience arises. The world has no part in the experience. It permits itself to be experienced, but has no concern in the matter. For it does nothing to the experience, and the experience does nothing to it (Buber, 1937: 5).

While it is impossible for a person to grasp an object in its entirety in I-It relating, completeness is the only way of relating in the world of I-You, the world of dialogue. This I-You relation or dialogue signifies 'the act of one whole being encountering or confronting another whole being' (Gordon, 2011: 209). Lipari (2004: 125) calls this an 'intersubjective, ethical, dialogical relation' in contrast to 'the instrumental, goal-oriented, monological relation of the I-It.' Buber describes the primitive dialogic event:

there are in it (...) only the two partners, the man and that which confronts him, in their full actuality, and while the world becomes in it a dual system, the man, without yet perceiving the I itself, is already aware of that cosmic path (Buber, 1970: 20).

Gordon clarifies that 'confronting' is what happens to a person in a dialogic encounter: the principal affirmation of 'the other's basic difference' while standing 'firm in his or her own being' Gordon (2011: 209)

#### 2.4.1.3 Dialogic Space: The Interhuman

As described dialogue is something that happens between persons.

What is essential does not take place in each of the participants or in a neutral world which includes the two and all other things; but it takes place between them in the most precise sense, as it were in a dimension which is accessible only to them both (Buber, 1969: 203-204).

However such a dialogic space 'between' the dialogic partners is transient and fleeting. It may be considered, as Phillips holds, an 'emergent quality of communication that occurs fleetingly and often in surprising rather than in neat, orderly ways' (2011: 32). Most importantly it cannot be planned as Buber holds: 'no one can know in advance what it is that he has to say; genuine dialogue cannot be arranged beforehand' (Buber 1965, in Phillips 2011: 32). For Buber it is important to note that this space 'cannot be reduced to something that happens within an individual's psyche or the dynamics of a group' (Gordon, 2011: 211).

The question is then what makes such genuine dialogue possible? Listening, although never directly addressed by Buber, is considered to be an essential quality in dialogue (Lipari, 2010; Gordon, 2011). A verbal dialogue consists of a mutual relationship which includes both, listening and speaking. But for a dialogic

relation to take place a focused attention or *presence* towards the other, which sets aside one's personal thoughts and opinions, is necessary.

There is genuine dialogue – no matter whether spoken or silent – where each of the participants really has in mind the other or others in their present and particular being and turns to them with the intention of establishing a living mutual relation between himself and them (Buber, 1965c: 19).

Buber referred to this also as awareness: 'To be aware of a man means (...) to perceive his wholeness as a person determined by the spirit' Buber (1965a: 70). On the basis of such a listening attitude a speaking partner may freely unfold their reality. Phillips (2011: 32) concludes that:

Through presentness, authenticity and confirmation, mutuality can occur – rather than the instrumentality of subject-object relations. From mutuality, dialogue emerges.

Following Buber (1965a: 75) authenticity means to say what is really on one's mind and not hold anything back. This is the precondition for perceiving the other through 'inclusion' (also referred to as 'imagining the real' or 'embracing'): the attempt to adopt the other's lived reality without in any way forfeiting one's own. And on the basis of an inclusion perception one is able to *confirm* the other in one's essence even while one opposes him. This is what Buber termed 'true confirmation' (Friedman, 2002). The act of *inclusion* is opposed to what Buber calls 'empathy' – 'the temporary 'suspension' of one's own concreteness for the sake of understanding the other' (Gordon, 2011: 212). To perceive the other through inclusion (or embracing) means *not* to give up one's own reality.

#### 2.4.1.4 The Problem of Mutuality in Unequal Relations

The mutuality or reciprocity in a dialogic relationship, however, does not presuppose equality. Buber was aware of unequal relationships. Cissna and Anderson differentiate that 'we can only have a degree of mutuality, even in unequal relationships' (1998, in Phillips 2011: 32). An example might be that between the educator and a student. It is only through embracing that more mutuality and thus a higher quality of dialogue can be achieved. Dialogue is particularly dependent on mutuality, that is the ability to listen actively and engage deeply. This entails a willingness to hear the speech as if one was addressed and an eagerness to interpret a meaning from it. At the same 'it does not require that all listeners heed the same message' (Gordon, 2011: 215, original emphasis). This is listening not in a 'passive and superficial' way (ibid.), but one that attends:

to the alterity of the other [which] involves giving the other meaning-making rights by renouncing one's own inclinations to control and master. (Lipari, 2004:138)

Such dialogue-focused listening attempts 'to create a space in which the other's unique voice can resonate' (Gordon, 2011: 217). To listen in this way means to be open – to be influenced by the words of the other. This is Lipari's (2010) attitude or state of 'listening being' – introduced in the first chapter.

I am being empty of possession and of all intentions other than the intention of engagement with you and of the what-will-happen (Lipari, 2010: 355).

The *listening being* requires one to suspend one's conceptions or presumptions about a situation to be open for the other. Such Listening becomes an ethical act:

that embodies responsibility, an expression of contingent encounter, a co-dependent and co-creative ethical relation. It arises from self as a function of otherness (Fischlin, 2014: 294).

I have delineated Buber's philosophical anthropology about dialogue at length here to build a basis on which any interaction between performer, computer, other musicians and the environment (audience and space) can be examined. This also serves to make clear how Buber's original concepts influenced the terminology of other thinkers introduced here: Bakhtin, Flusser and scholars from Improvisation studies.

#### 2.4.2 Bakhtin's Dialogism

Russian philosopher Mikhail Bakhtin applied the dialogic principle to language in the context of literature to examine how meaning is constructed through written and verbal speech acts. In his writings Bakhtin proposes several understandings of 'dialogue'. On a principal level a 'dialogue' refers to an act of relational meaning-making which functions through an interplay of multiple, often contradictory voices. Such voices are not equivalent to the singular voice of a speaking subject, but rather they are elements which are contained within a single utterance or speaking subject. As Bakhtin writes, 'every utterance (...) is a contradiction-ridden, tension-filled unity of two embattled tendencies in the life of language' (Bakhtin, 1981: 272). These two tendencies or forces are the 'centrifugal' and the 'centripetal'. The former pushes toward difference or decentralization and the other toward unity or centralization (ibid.).

This demonstrates how Bakhtin's focus is on the performative aspects of language as communication: how meaning is produced through verbal and written speech

acts. Such meanings, including the understanding the self and the other, are produced dialectically, in the tension between the two opposing forces. Bakhtin considers each voice a discourse in itself: It holds an ideology, a perspective or theme and is not just a medium for speech or the uttered speech of an embodied person (Clark and Holquist, in Kanellopoulos, 2011). This interplay of multiple voices is the dialogic aspect of the meaning-making process: Unity is formed through the interplay, but it may be one of contradictions. This specific kind of unity is described in Bakhtin's concept of Heteroglossia which he defines as

another's speech in another's language, serving to express authorial intentions but in a refracted way. Such speech constitutes a special type of *double-voiced discourse* (Bakhtin, 1981: 324).

This double-voiced discourse 'is always internally dialogised' (ibid.) between the intention of the author of the text and the author of the speech act within the text, the actual character who speaks.

In summary we may identify three main understandings of dialogue: Firstly, there exists the 'internal dialogism' of the word (Morson and Emerson, 1990: 138). This is an immanent attribute of any utterance and of language, its medium. Any word or utterance is in a dialogic relationship with the 'history' of their use: The sum of all prior utterances infuses traces of meanings into the speech act (Bakhtin, 1981: 91) . Further an utterance will be dialogised through the context of its speech. This is the situatedness of language. The words are in play with anticipated responses as Bakhtin writes:

The word in living conversation is directly, blatantly, oriented toward a future answer-word: it provokes an answer, anticipates it and structures itself in the answer's direction. Forming itself in an atmosphere of the already spoken, the word is at the same time determined by that which has not yet been said but which is needed and in fact anticipated by the answering word. Such is the situation in any living dialogue (ibid.: 280).

Thus language is dialogically responsive to things said before and in anticipation of things that will be said in response to it.

A second principal understanding of dialogue refers to a speaker's reflexivity, that is an awareness of the sources of her words and her intention to make its sources transparent to the listener. According to Morson and Emerson, this 'allows some utterances to be dialogic and some to be nondialogic (or monologic)' (in Kanellopoulos, 2011: 124). Transferring this to the situation of improvised music Kanellopoulos (ibid.) describes dialogue then as

that quality of interaction (between people, texts, sentences, musical gestures (...)) that is rooted in openness, that does not wish to assume a privileged point of view, or a definitiveness of meaning and closeness of sense, but is also aware of its debt to other speakers, points of view, or historical and cultural significations of particular modes of invention.

A consequence of the *dialogic* form of improvised interactions is then the possibility that these very people, texts, sentences and musical gestures may be altered in the process. Thus dialogic improvisation holds the potential to develop any involved medium towards the open-ended, or towards what Kanellopoulos citing Bakhtin has termed 'unfinalizability' (ibid.).

A third meaning of Dialogue refers to a hermeneutic model for understanding the world. This emphasises the inter-subjective quality of all meaning derived from communicative acts. Meaning is produced neither from the meaning contained in the utterance and single symbols of a medium (i.e. the structure of language), nor through the predisposition of the listener who interprets these signs alone as an autonomous subject, but between these two through acts of reflexive interaction. Dialogue thus entails a co-creative process of meaning-making between mutual partners. For Bakhtin 'Dialogization' is a process which a word, discourse, language or culture undergo and through which these become relativised, de-privileged and aware of competing definitions for the same things (1981: 427).

#### 2.4.3 Dialogic Intersubjectivity in the context of this research

Following Kanellopoulos (2011), I identify the second of Bakhtin's understandings of dialogue (which is actually closest to Buber's original concept) as most resonant for the context of this research. Dialogue taken as a quality of interaction, between utterances, musical gestures, speakers or selves and identities which introduces the possibility of alteration for all partners in dialogue without actually becoming the other. Focusing on the (self-) transformative aspect of a dialogic encounter Peeren's term of the 'dialogic intersubjectivity' applies. FIM is considered to create a space which makes this kind of dialogic intersubjectivity possible. The aesthetics of the practice resist fixity and closure and provide a possibility to question criteria of musical correctness. Fischlin, Heble and Lipsitz (2013: xii) go as far to propose 'musical improvisation as a generative yet largely unexamined model for political, cultural, and ethical dialogue and action' which calls into being an 'ethics of cocreation'. In their proposal the emphasis on the socio-cultural aspect of the music making process is evident. It may be concluded that a music which is

produced through dialogue is an intersubjective product, unique through the individuality of every participating actor and – following Buber – in its completeness only accessible to the dialogic partners. However, this should not imply that all music-making contexts including the collaborative ways of learning, creating, and performing are necessarily dialogic. Similar to Buber's notion of dialogue as a 'genuine encounter' it is necessary to look for the transitory moments of such quality within any musical improvised interaction.

#### 2.4.3.1 Heteroglossia through algorithms

At the same time, Bakhtin's theory of the internal dialogism of the word, may be used as an indicator to identify a dialogic quality in intersubjective acts of communication. A Bakhtinian perspective of improvisation (as dialogic practice), following Kanellopoulos, emphasises the obligation to create music not with reference to a particular way of developing musical ideas (i.e. an accepted musical style), but with reference to a particular attitude towards the music-making process: to follow 'one's own responding word' (Kanellopoulos, 2011: 122). This poses two difficulties: (1) a free improvising musician does not know in advance what this 'responding word' might be and (2) the context of its utterance is dynamic and changing. The identity and meaning of a musical gesture is always determined by the collective sound of all improvising partners at that moment. It is impossible to imagine or apprehend the 'word' in isolation. This challenge produces otherness or the dialogic quality of free improvised music:

In improvisation, I must, on the one hand, work intensely to discover ways of continuation that do not merely rest on ready-made formulas, and on the other to develop a kind of passivity, of readiness to be carried away through the sounds of my fellow players, which might take me to places hitherto unexpected: I must learn to surrender to the moment, and in this way I am othering myself (ibid.:123).

Similar to a musical gesture, also a code word in the programming language can only be understood in its situated context: the same code object changes its 'meaning', that is it will produce a different sound, depending on how and in which hierarchical order it is put in a context of the other code words. Adapting Bakhtin's obligation could in this situation translate into following one's own responding code word. Such an approach allows the development of a musical gesture not only on the basis of what the performer hears but also on the basis of what the code word itself may suggest (the code objects and structures of the programming language). Now we can see that the process of articulating a musical thought into code simultaneously

contains a mixture of internal and intersubjective dialogism: Intersubjectively between performer and computer, the former acts (or writes) code in silence while the latter interprets and transforms the code commands into sound. During the very process of writing (which is a different gesture than speaking or playing an instrument) the code words interact dialogically with each other *and* the performer's thought to force their multiple traces of meanings onto the performers abstract thoughts.<sup>47</sup> Following Bakhtin (1981), the musical gestures resulting from this human-computer-code interplay have a heteroglot quality even before they are 'dialogized' by the group of other improvisers. They contain two (different) intentions at the same time: the 'direct' intention of the sound produced by the computer and the 'refracted' intention of the programmer-performer hidden within it. The musical thought of a programming performer therefore undergoes a two-level 'dialogization' before it arrives in the collective sound of the performance. Firstly through the live interaction with the computer in textual or gestural form and secondly through the interaction with the other improvisers in a group. Both of these take place almost simultaneously, yet not directly responding to each other. Extending this view, any sound produced on a computer might be considered heteroglot for it always contains the 'voice' of the machine who produced it (which is in itself made up of many different elements in the form of various software modules used together with the actual hardware and the circuitry of the computer from which these softwares operate) and the 'voice' of the programmer who actuated it. Following Bakhtin's various types of heteroglossia, I propose this specific form of heteroglossia produced through acts of interactive programming-composing as *algorithmic heteroglossia*. This can be found in improvised live coded performance where the use of specific software environments (comprising the thoughts or 'voices' of others) makes the program code a mediator of the performer's thoughts and thus a cocreator of meaning.

#### 2.4.4 Flusser's Telematics

The next theoretical pillar I will introduce is the scenario of the 'telematic society' by media theorist Vilém Flusser (2011). His model of communication describes

---

<sup>47</sup>I may illustrate such a case of 'dialogism': A code word (i.e. a sine wave oscillator) may already 'propose' its 'meaning', that is, its use to generate a sine wave sound, even though a performer perhaps intended this object for a different use (i.e. to modulate the filter frequency of another subtractive synthesis process). The example may also be perceived in reverse. The importance lies in how the imagined use of a code object influences the choice of the programmer in the process of constructing a code phrase.

how not only humans interact and relate *dialogically* to each other, but how this is realised in *mutuality* with programmed machines. In my conceptual framework Flusser's ideas close the gap between Buber's anthropological and Bakhtin's semiotic position to acknowledge the presence of a computational agency in Dialogic Coding practice. Through Flusser's technology-oriented yet anthropocentric media theory perspective I was able to develop a dialogic understanding of my live coding practice.

Flusser applies the principle of a dialogic relationship to the interaction between humans and machines (which he refers to as 'apparatuses'). In his books 'Toward a philosophy of photography' (originally published 1983) and 'Into the Universe of Technical Images' (originally published 1985), he develops the metaphor of 'Telematics': a society in which humans communicate with each other *through* apparatuses, yet in a 'free' and democratic, that is, a dialogic way. This vision is founded on the understanding that human existence in this world is a struggle against entropy, that is against decay in nature and our own mortality. Flusser sees civilisation and culture as means to work against entropy which becomes particular in the history and development of media technologies (painting, language/writing, electronics). Flusser considers culture to be the effort to become immortal through communication, at least temporarily when he writes:

human communication is an artistic technique whose intention is to make us forget the brutal meaninglessness of a life condemned to death' (Flusser, 2002: 4).

Flusser developed his 'telematic' vision as a response to what he identified as the last developmental stage in the history of media technologies: the digital revolution, or what he terms the 'universe of technical images' (2011). By this he refers to a world which has been rendered entirely abstract by science (a part of human culture) and which needs to be made concrete again for humans to be able to really grasp it. Our world consists of particles which need to be computed into 'technical images' through the use of an 'apparatus'.

What remains are particles without dimension that can be neither grasped nor represented nor understood. They are inaccessible to hands, eyes, or fingers. But they can be calculated (calculus, "pebbles") and can, by means of special apparatuses equipped with keys, be computed. The gesture of tapping with the fingertips on the keys of an apparatus can be called "calculate and compute." It makes mosaic-like combinations of particles possible, technical images, a computed universe in which particles are assembled into visible images. This emerging universe, this dimensionless, imagined universe of technical images,

is meant to render our circumstances conceivable, representable, and comprehensible (Flusser, 2011: 10).

Flusser sees this effort of grasping through making-concrete only possible with the use of computers which he terms apparatuses: the machines which compute and calculate. These apparatuses are themselves a product of culture and follow specific pre-defined rules: the 'program'.<sup>48</sup> The challenge for the human, following Flusser, lies in how these apparatuses are being used. To create genuinely new products<sup>49</sup> one need to play creatively with the program of the apparatus and turn it against its own logic ('envisioning'), instead of simply using it in the way it was intended ('functioning'). The apparatuses produce 'technical images' which 'inform' the world and propose a way in which to make sense of it. But they do so only in collaboration with a human operator. 'Functionaries' operate an apparatus in accordance with their program, 'envisioners' work with the apparatus to infuse their own intention into the program. To act as envisioner and program the apparatus accordingly, a consciousness of 'calculating and computing' is necessary. Flusser refers to the product of the collective effort of the human and the machine as 'technical images'. He does not specify their form in more detail, only their properties in relation to the traditional images. These new images do not symbolize anything in the world, Flusser terms this 'superficial', but they represent a way of looking at the world. They therefore represent a program, the outcome of the intentional effort of the human together with the apparatus. Flusser regards them as 'computations of concepts' not 'observation of objects' Flusser (ibid.). Considering Flusser's concept of the 'technical image' as a representational form to refer to the products of computing technology, I regard the sonic products of the apparatus which result from the interaction with a performer as as another form of a 'technical image'.

The relevance of Flusser's ideas for this research-practice lies in how it offers a perspective of critical reflection towards how the computer is used in performance, towards how and by whom decisions are made and towards what goal the interaction is aiming at. In the description of the 'telematic society' Flusser proposes for all members of a technologised society to become *envisioners* and

---

<sup>48</sup>In this writing the 'program' of a computer is synonymous with the term 'algorithm' to describe the rules based on which the apparatus functions.

<sup>49</sup>Flusser speaks of synthesizing or constructing 'new information' (i.e. see 2011:90).

communicate to each other through apparatuses in a *dialogic* way. This serves to remind users of technology, to pursue active ways of interaction with the apparatus and moreover, to send the products of such interaction out to other humans in communicative acts. By pointing to the essence of what it means to act 'freely' Flusser's thinking paved the way to an ethical approach for Dialogic Coding practice as presented in chapter 5. In the following sections I will briefly explicate some of Flusser's central terms which are relevant to the discussion later in this writing.

#### 2.4.4.1 Apparatus

Flusser defines an 'apparatus' as a 'plaything or game that simulates thought (...) [or an] organization or system that enables something to function' (Flusser, 2000: 83). The concept therefore, applies to non-human agencies (like his prototypical example of a photo-camera) as well as larger social constructs such as the 'apparatus' of the state or the market economy. Every apparatus is invented by a *programmer* who inscribes the program—an automation (a set of rules, an algorithm)—into it. Also every apparatus needs a *functionary* to operate it. Even though automation and the capabilities of the apparatus may be highly developed, it still needs the collaboration with a functionary because otherwise the results 'produced in such a way would be redundant, that is, predictable, uninformative situations from the standpoint of the apparatus's program' Flusser (2011: 19). We find ourselves in a deeply entangled situation with the apparatus. Humans and apparatuses are so closely interconnected that it becomes meaningless to try to distinguish between the human and artificial agencies (Flusser speaks of 'memories') involved in the production of information. 'A man gives an apparatus instructions that the apparatus has instructed him to give' (ibid.: 74). It is a circular, interdependent relationship as described by second-order cybernetics (von Foerster, 2003). Who controls whom becomes a question of perspective. In this entanglement, 'we will face the unpleasant choice between humanizing artificial intelligences and making human ones more like apparatuses' (Flusser, 2011: 113).

#### 2.4.4.2 Functionary, Programmer, Envisioner

Flusser proposes a notion for how to interact with the apparatus in order to produce new information: Either to 'function' according to the logic of the program or to 'envision', that is to transform one's own intentions into a concrete form (i.e. digital sound) in a collaborative act with the apparatus. To be able to *envision* a human need to think with a calculating and computing consciousness, that is to

see 'something solid in the most abstract things (particles)' (ibid.: 170). A *functionary*, however, only 'calculate[s] and compute[s] instructions as instructed' (ibid.:72). She acts as it was intended by the sender (the inventor of the program) and can thus be considered part of the apparatus itself.

[Envisioners] are people who press the keys of an apparatus to make it stop at an intentionally informative situation, people determined to control the apparatus in spite of its tendency to become more and more automated and so to preserve human judgment over the machine. [They] are people who try to turn an automatic apparatus against its own condition of being automatic. (ibid.:19)

Consequently, to *envision* is equal to the ability to stop an apparatus in its automated procedure. Flusser terms the *competence to recognize* the desirable moment for synthesising truly new and non-redundant information, the *human freedom* of the envisioner. For Flusser (2000: 80) the purpose lies in 'consciously creating unpredictable information'<sup>50</sup> or 'to transform redundant coincidence into the unforeseeable, into an adventure' (2011:111) in order to advance culture and come closer to immortality (through new information). Envisioning may be considered either as a type of interaction (to 'push keys' to operate an apparatus or she might also become a programmer who invents apparatuses. Flusser considered his own practice of writing (pushing keys on a typewriter) as a practice of *freedom* in this sense.<sup>51</sup> The responsibility of the

engaged envisioners is to direct instruction dialogically against the apparatus, to arrive then not at a programmed democracy but democratic programming (ibid.: 77).

---

<sup>50</sup>This quote may be better understood in the context of Flusser's view on entropy. He writes that improbable situations are 'erroneous exception[s] to the general rule of increasing entropy' (2011:17). 'Inevitable' to him is that 'the possible becomes probable' and 'impossible' is that the 'possible becomes improbable'. The consequence of entropy (our own mortality) is therefore probable and exceptions to it are improbable. It is in this sense that Flusser sees the activity of envisioning as a means to work against entropy.

<sup>51</sup>Flusser writes: 'I know, when I strike a key, that I am dealing with a programmed instrument that reaches into the swirl of particles and packages them into texts. I know that a word processor can do this automatically, a chimpanzee can do it accidentally, and a stenographer can do it by copying an existing pattern, and that in all cases, the same text as mine will appear. I know, therefore, that my keys are inviting me into a determined mesh of accident and necessity. And in spite of it all, I experience my writing gesture concretely as a free gesture, in fact, free to such an extent that I would rather give up my life than give up my typewriter. "Writing is necessary, living is not." (attributed to Heinrich the Sailor (1394–1460)) For my being is concentrated on my fingertips when I am writing: my entire will, thought, and behavior flow into them and through them, past the keys, past the particle universe those keys command, past the typewriter and the paper and into the public sphere. This, my "political freedom," my key-striking, publicizing gesture, is my concrete experience of keys' (Flusser, 2011:28).

This makes clear how Flusser considered this ambiguous collaboration between humans and nonhuman apparatuses as a historical opportunity to change the structure of society toward a more democratic and participative one. He sees the people of the future as programmers rather than functionaries (ibid.: 154). Particularly this emancipatory leap from being programmed to (re-)programming, from a passive to a more active position is of relevance to this research. In technologically mediated music performance the challenge whether to follow or to lead (control) the 'apparatus' is always present. In this conflict Dialogic coding practice enables the performer to address exactly this aspect of technological entanglement: The very process of programming becomes a possible topic for public discussion.

In this thesis Flusser's term of the 'envisioner' is not used for it seems out of context – given its original relation to the creation of imagery. Nevertheless, the concept of 'envisioning' has much in common to my general proposal for the performer responsibilities in Dialogic Coding practice as will become clearer in the later chapters of this thesis.

#### 2.4.5 Ethics and Second-Order Cybernetics

Knowing when and how to act within the improvising moment is an ethical issue; and the answer to this cannot be given in a set of rules (Kanellopoulos, 2011).

Improvisation as a dialogic activity is considered to have an ethical dimension to it (Kanellopoulos, 2011; Nicholls, 2012; Fischlin et al., 2013; Heble and Caines, 2014). Fischlin et al. (2013: xviii) have called this an 'ethics of cocreation' which refers to the 'mutual recognition and responsibility, [and](...) cultural inclusion and belonging' on which the ability to exercise choices in an improvisation depends. Ethics are the guiding principles through which a practice, such as improvised music, can become 'a symbolic and (...) embodied representation of human potential and freedom' (ibid.: 15). The question is how do we find and articulate these principles in an open situation such as this? From what position is it possible to deduct these and translate them into guidance for others? This is a problem of perspective, a problem of the observer and the observed. Cybernetics, the science of communications and automatic control systems, and its reflexive sister *second-order cybernetics* (von Foerster, 2003).

There are two positions from which to look at or talk about any activity:

- a) that of an independent observer who watches the world go by; in a role: *the discoverer*
- b) that of a participant actor who influences the course of the play; in a role: *the inventor* (ibid.: 294)

From the observing position moral codes are deducted. To tell others how to think and act in the form “Thou shalt not”. From the participating position ethics are deducted, because I can essentially only tell *myself* how to think and act: “I shall not”. According to von Foerster, ethical behaviour is particularly relevant in the interaction with a problem or question which is 'in principle undecidable' to the agent (the performer). Von Foerster defines that the in principle undecidable questions are those to which there is not a single answer. These are *nontrivial* questions. Von Foerster's prototypical example is the question how the universe came into being: because nobody was there to watch it, nobody can account for it, hence, there is no absolute 'truth', no single answer. Von Foerster (ibid.: 293) sees these questions opposite to

the 'decidable questions' [which] are already decided by the choice of the framework in which they are asked, and by the choice of the rules used to connect what we label 'the question' with what we take for an 'answer'.

In computer terms the 'decidable questions' are deterministic or *trivial* systems because there is, by the definition of the framework, only one answer to them. The consequence of this is, following von Foerster, that in the interaction with nontrivial systems:

there is no external necessity that forces us to answer such questions one way or another. We are free! The compliment to necessity is not chance, it is choice! We can choose who we wish to become when we have decided on an in principle undecidable question. (...) With this freedom of choice we are now responsible for the choice we make (ibid.)

Objectivity, the position of an *independent* observer, can thus be considered as a device for *avoiding responsibility* for one's actions. The way out of this dilemma is then to account for one's own actions and communicate this through language: a performance *of* ethics developed in a dialogic interaction. In the context of dialogic laptop performance the articulation of an ethics has the purpose to assist the programmer in making decisions for the rules of a system in performance. The guiding principles are entirely based on my own personal observations and can, in the line of this argument, be recognized as my own *inventions* on the subject. As

such they are *my* responsibility and logically also *my* ethics. For the practice of dialogic coding the perspective of second-order cybernetics holds a valuable insight: that even though the interactions take place between a human performer and a programmable apparatus they are a result of how the programmer inserts her own purpose into the system (of the apparatus). The apparatus is thus her invention and consequently her responsibility, too. Von Foerster (ibid.: 227) has summarized the challenge of observing in a closed circular system in these two principles:

- The ethical imperative: Act always so as to increase the number of choices.
- The aesthetical imperative: If you desire to see, learn how to act.

By postulating ambiguous action as well as intentional action Von Foerster emphasises the constructed nature of our reality and the responsibility that comes with it. When von Foerster (2003: 286) says 'if we don't determine a purpose, someone else will do for us!' he pushes for an awareness of one's own position as observer and inventor of actions which should lead into an active contribution to one's reality yet with an acknowledgement of its subjectivity.

In Dialogic coding practice the performer acts in modes of programming, functioning, and listening (as will be analysed in chapter 4). Thus, it is a good example of a second-order cybernetic practice in which the act of inventing is combined with a participative observation of a system.

## **2.5. Towards an Analytical Framework**

An agent-orientated interpretation would look for interactions enjoyed between players and/or machines, by regarding the sound materials themselves as agents that work together in the wider landscape of the music (e.g. how one musical behaviour appears to instigate or respond to another). This is sonic evidence of a quasi-social praxis, to the extent that musical actions can be readily mapped to a particular human or machine contributor (Young, 2016: 97).

Michael Young describes exactly what this analytical framework intends to discover: the interactions enjoyed between players. While his perspective begins from an analysis of social interaction evidenced in the form of sound, the analytical perspective developed here goes further to give insight into more forms of interaction, such as the evidence found in code, gestural or visual actions. I am

able to develop this perspective from the privileged position of the practitioner who is able to look 'behind the screen' of the apparatus. To perform this analysis with the informed view of my conceptual framework of dialogue may help to bring out the tacit knowledge contained in the 'doing-thinking' of Dialogic Coding practice. Buber's understanding of dialogue is used as a method to identify the quality of a dialogic relation in any interaction, while the understanding of Bakhtin may function as an indicator whether the relation is generally of dialogic or monologic nature. Flusser's adaption of the principle toward the interaction of human-apparatus entanglements intends to highlight the ethical dimensions in the dialogic interaction regarding the freedom of both partners, the human and nonhuman. And further this perspective examines the role and responsibility of the human entangled with the apparatus (to program/envision or to function).

Lastly the reflexive aspects of Dialogic Coding practice point to the fact that an answer to this must be assessed on an individual basis. Any possibility for immersive and challenging experiences will always be dependent on individual knowledge and abilities as well as the intentions of the participating performer. Therefore it needs to be stressed that – even if I propose generalised conclusions from the analysis of my practice – these are limited and cannot simply be applied out of context. Nevertheless, the insights gained through analysis may still be informative to other practitioners or researchers.

### 2.5.1 Questions Towards the Dialogic in the Situation

The following questions were developed from the framework and used to critically review my own performance experiences and analyse recordings of rehearsals and performances in the course of this research. They approximately follow the main three aspects of Dialogic Coding practice: (1) improvised group performance, (2) dialogic code improvisation with an apparatus, and (3) the production of transformations for the performer through a reflexive practice.

#### 1) Bakhtin's Dialogue: Dialogism of the utterances (Intertextuality)

- Who are the speakers? Who are the listeners?
- Does the interaction take place sequentially or simultaneously?
- How does an interaction correspond to each other?
- What voices are articulated and what knowledges and identities (self and others) are constructed in the different voices?
- How are multiple voices contained within one speech?

- What are the relations between the different voices?
- To what extent, when and how is there an opening up for voices that articulate plurality (or singularity), or: how is the 'freedom' for any of these voices?

## 2) Buber's Dialogue: Dialogic Intersubjectivity

- Can we identify dialogic moments in concrete instances of social interaction whereby an 'I-You' (subject-subject) relating occurs?
- How is the confirmation of otherness performed: to accept and affirm the difference of the other without completely giving up one's own standpoint?
- How does a qualitative dialogic interaction change one's self-conception and identity?
- What kind of dialogic space, the between, does this dialogic interaction create? Where can we locate it?

### 2.5.2 Questions Towards the Apparatus (the nonhuman agent)

#### From Flusser's Dialogic Programming

- How is the apparatus programmed?
- Does the performer *operate* or *play* the apparatus?
- How does the apparatus 'program' the performer? How is the interface structured? Is the computer code easy to use?
- Does the interaction and musical result surprise the performer? Or the listeners?
- Does interaction and musical result expose novelty or predictability?
- Does the performance allow for and encourage machine autonomy (freedom)?
- Does the practice give hands-on access to the performer?
- Does a performance question authority? In musical, organisational or technical terms.
- Does it allow for an *enjoyable* experience of sound and technological interaction? For a performer *and* the listeners??

### 2.5.3 Performance & Musical Questions relating to Reflexivity

- Does the practice make the performer's thinking or the computational processes transparent to an outsider?

- Does a performance create ambiguous action-perception relations?  
(provide a perceptual challenge & difference, a dialogic experience)
- Are the performative gestures legible?  
Does the performance allow for ambiguous interpretations?
- What is the structural complexity in the music? Does it allow for a multiplicity of voices?
- How transparent are music and interaction?

## 2.6. Summary

In this chapter I introduced the leading methodology of this research project: Practice-as-Research. I highlighted the methods which I have used in the development of my own practice such as critical reflection, analysis of field notes and audiovisual recordings throughout the process. The development was interleaved with periods of conceptual reading and reflection on my practice to develop both, the practice and the framework further.

In the second section I identified how I have developed my research practice under the PaR methodology: from session-based coding to more structured collaborations in laptop duets to performing with pre-programmed interactive systems. The practice does not comprise a unified coherent approach. Rather each case developed in response to individual circumstances and concepts with the intention to realise different human-apparatus relations – from trivial, directly controllable systems to nontrivial and alienating ones. In my overview of these different approaches I have demonstrated how the 'knowings' associated with the practice have successively emerged and related them to my methodological framework. This was complemented by an overview of selected examples in the form of 'information sheets' with descriptions and references to online repositories/websites to make the projects more accessible to the reader.

In the last section I introduce the conceptual framework used to discuss and contextualize my practice within the domain. This framework is centred around the principle of dialogue as introduced by Martin Buber in a context of encounters between 'whole beings' and then extended by Mikhail Bakhtin for the analysis of language, particularly the relationships between spoken as well as written words or utterances. This is complemented by a media theoretical interpretation of dialogue found at the basis of the theory of 'telematics' by Vilém Flusser. From Flusser's conceptual world I adopt the terminology of 'programmer', 'functionary'

and 'apparatus' to describe the central actors in my inquiry. Flusser also suggests the term 'dialogical programming' which influenced the naming of the practice to 'dialogic coding'.

The developing 'knowings' of my practice will, in the following chapters, be placed in resonance to the conceptual framework above. The dialogic interplay between the practice and these discourses serves to identify the distinct aspects of Dialogic Coding practice in order to make clear the substantial new insights this research through its practice may give to the field.

### 3 – Simultaneous Interactions in Dialogic Coding

In this chapter I will analyse the different interactions between improvising partners within my practice. First I identify the four levels on which these interactions take place – the meta, macro, meso and micro – and show their different forms. For each form I exemplify how a dialogic relationship between the participants may be created. In the second half I focus on the specific interaction the laptop performer has with her apparatus on the micro-level. From this focus I will analyse how the apparatus can be created to function as a dialogic partner through the use of indeterminacy. These observations are discussed throughout in relation to the conceptual framework.

#### 3.1. Four Layers of Simultaneous Communication

This analysis begins with an identification of the agents in the practice: *Who is speaking and who is listening in a group improvisation with computers? To whom does the performer relate in a dialogic way across time and space?* Based on my research-practice I have developed a four-layered model of the communicative interactions in group performance – as introduced in chapter one. Each level corresponds to a different group of addressees in communication. The highest and most remote level is the *meta-level*. This refers to any action or utterance a performer may have in relation to other events or performances in a distant time and space (i.e. in the past). This follows the notion of intertextuality or what Bakhtin (1986) described as the 'internal dialogism of the word'.<sup>52</sup> The other three levels are more synchronous and take place in the time and space of the performance event itself. Most distant to the performer are relations on the *macro-level*. Here I locate all the interaction and exchange with the environment of performance including the co-present audience. Moving closer to the performer is the *meso-level*. Here I locate the other co-present musicians who actively co-create the emerging improvised music. Finally, at the *micro-level* all interactions with the apparatus take place. In the next sections I will examine these layers in detail to identify the potential dialogic relations entailed. I have provided diagrams

---

<sup>52</sup>Any word or utterance is in a dialogic relationship with the 'history' of their use: The sum of all prior utterances infuses traces of meanings into the speech act (Bakhtin 1986:91). See also section 2.4.2 in the last chapter.

for each level which visualise the participants and their communicative interactions.

### 3.2. Meta: Dialogue with the Non-Present

On the *meta-level* an algorithmic performer interacts dialogically with art, history or culture across distant time and space. Even though the focus of this research-practice is on the synchronous, intersubjective interactions during performance, it is worthwhile to mention this level here. A dialogic approach to performance is never a solipsistic practice. Thus a focus on the remote and asynchronous connections may explain how the sonic materials used in Dialogic Coding performance relate to other practices. Bakhtin demonstrated the intertextuality of meaning and how each utterance always carries its prior uses with it. Similarly, will such a reuse of a 'readymade' in program code i.e. the recycling of pre existing code, introduce other meanings and contexts of meaning into local performance situations. Such creation of symbolic relationships through musical gestures with a hybrid style has been termed signifying or heteroglossic playing in jazz improvisation (Monson, 1994; Horn, 2000).

#### 3.2.1 A Dialogic Sonic Vocabulary

Computing technology enables processes of creation, manipulation and a recombination of various materials – readymade or newly synthesized – in real-time. This makes it possible to create dialogic relationships with the original context and meaning of the materials used in a live performance. While this situation is obvious in the case of concrete sound, the connections are less easy to see in cases of re-used program code. In my performance practice each successive collaboration has contributed specific elements to my 'sound-code-archive'<sup>53</sup> on my computer. This is facilitated by the programming environment I use: SuperCollider stores all code as text files which means that they are easily editable and shareable with others<sup>54</sup>. Through my participation in a few networked performances of *powerbooks\_unplugged* (pbup) (see section 1.6.4) I was able to work with a collection of sounds which the band had developed through their own

---

<sup>53</sup>This code can be found in my repository online – <https://jonashummel.de/archives/code>

<sup>54</sup>The reader may imagine that, similar to any other program code, complex actions or sound processes may be expressed only by letters. A large collection of, for example, descriptions for sound synthesis will only have the size of a few Kilobytes and can as a consequence be easily sent via email in order to then be executed on the local machine which generates the same sounds but in a different time and space.

performance practice over the years. The sharing of this source material in coded form lies at the core of pbup's live performance practice. I continued to use parts of these sounds during studio rehearsals as well as in subsequent improvised performances.

Similarly my sound collection was influenced by other encounters in the process of learning to live code with the SuperCollider language. For example, the website/platform 'sccode.org' provides an example of a resource which contributed some sounds to my archive. Also the publicly available tutorial sessions by pioneer live coder Nick Collins left traces. And lastly, in many situations when developing a performance in an improvised mode of interaction through live coding techniques, I turned to the integrated 'help files' of the software itself which provide starting points for how a code object may be used to create sound. This strategy might seem plagiaristic at first but the reader should be reminded that from the code for the generation of the source material a myriad of other possibilities for actuating these sounds are possible. Furthermore, as a possibility of the accessibility of the synthesis process in the SuperCollider software it is very easy (or even afforded) to change any source code into something new every time it is used. This is a strategy used within the live coding community.

Another insight gained from the practice concerns how such sounds or synthesis code is actually produced. When working (compositionally) on the timbre of a sound for example, this is an iterative process: One may change one aspect of the code, i.e. replace one code object with another which produces a similar but different effect. Alternatively, one may also change the input to the process to explore how the sound changes for different inputs. This code exploration follows an approach of sonic approximation: To get closer to an imagined sound or an area of interest one needs to apply radical changes first and the closer the sound gets to the idea, the smaller the changes in the code will be. When such a dialogically coded sound is used during a concert, the final activity during a performance is actually one of making only these little changes and adjustments during the running synthesis process. Based on such experience I identify a connection in my practice to what Fischlin et al. (2013: 58) have called the notion of 'riffence':

The capacity of improvised music to invoke differential ways of being in the world across multiple contingencies that include politics, ideology, history, spirituality, ethnicity, and alternative forms of social and musical practice.

Here, Fischlin et al. refer to how improvised performance demonstrates a particular kind of otherness<sup>55</sup> through the act of performing. From this viewpoint I have developed a coding strategy which combines non-obvious copy & paste (the 'sampling' of program code) with small incremental changes through live coding – this can be seen as 'riffifferential' way of music making. A minimalistic live coding praxis may be riffifferential, perhaps not in the full cultural or political dimension of Fischlin et al., yet as a provocation<sup>56</sup> within the rituals of traditional music performance.

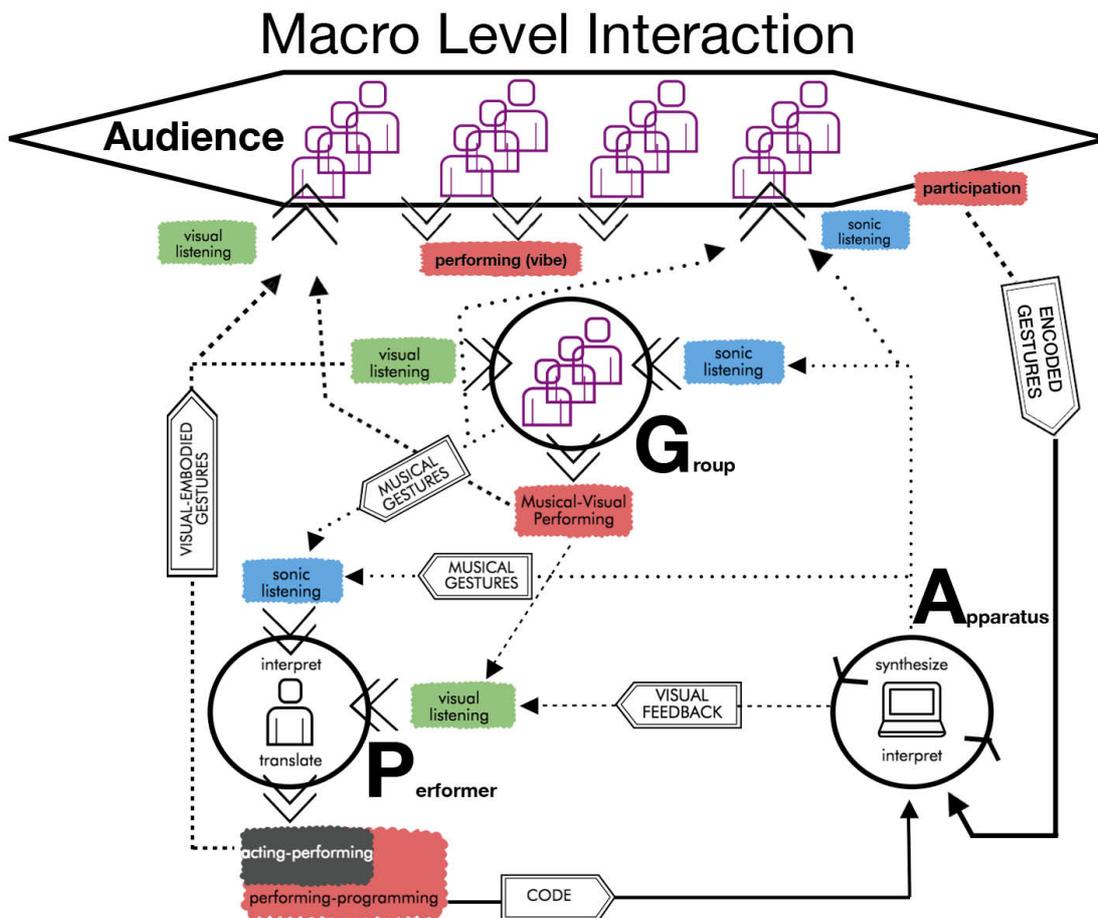


Figure 7: Macro level interaction in performance

<sup>55</sup>'Otherness' or 'alterity' describes the basic experience of the 'other' in a dialogic relation, the dialogic partner as different to oneself.

<sup>56</sup>By 'provocation' I refer to the challenges which laptop performance has introduced into the rituals of music performance. See the discussion on liveness for examples (Cascone, 2002; Auslander, 2008; Sanden, 2013).

### 3.3. Macro: Interacting with Performance Environment and Audience

In popular live electronic music practices such as Dj'ing and EDM<sup>57</sup> performance the relationship between the performers and the audience (and space) plays a crucial role for the conduct of performance. The 'vibe' or atmosphere in the space produced by the co-present (listening and possible dancing) audience is taken as an indicator for the direction of performance. If the feedback perceived by the performer is not as expected she needs to change the direction of her musical activities, if this vibe is positive then one may continue. Even though this research-practice does not share the aesthetic or rituals of EDM, this 'vibe' still plays a role in how an improvised performance develops. In the dialogic approach of FIM the atmosphere of the performance event naturally influences the decision making process (i.e. who is the audience? how is the climate? how is the light? how does the space resonate and so forth). In addition, the programmable technology provides the possibility to integrate a co-present audience directly through participation. The interactions on this level are visualised in the diagram in Fig.7. This aims to show the complexity of the situation in which the improvising performers are mainly focused on themselves and their interactions. The 'vibe' of the audience is a diffuse and non-personal kind of atmosphere emanating from the audience and space in the direction of the performers.

#### 3.3.1 Creating Dialogic Spaces through Participation

The basic problem in the communicative exchange between a programming performer and a listening audience lies in the fact that the relations between programmer and apparatus are not represented in an expressive or explicit way. A consequence of this problem may be that the listening audience needs to change their viewpoint to come closer. Only in close proximity may an outside spectator be able to experience the performance as an actual display of dynamic human-apparatus interactions in order to see the liveness of the actions more clearly. Live coders have responded to this challenge by the visual display of the computer screen onto a large screen in order to make the performance of typing code more visible. From my own experience however, the best understanding of the computer interaction results from first-hand participation.

---

<sup>57</sup>EDM refers to the genre of 'Electronic Dance Music' which 'features electronic synthesized and sampled instrumentation, with at least some parts of a percussive nature, in tracks designed for dancing' (Collins et al, 2013: 102).

In my practice I have attempted few such participative scenarios<sup>58</sup>. They helped me to understand that a low-level access to the technological interface is helpful for the participant to respond in a meaningful way and this has enabled the experience to become empowering. An overly complex interface with layers of automation built into the system will be opaque for an outsider, unlike for myself the designer and author of the interactive system. This recalls Buber's call for *mutual authenticity* as a precondition for a dialogic relation. To speak in an authentic voice may, in this scenario, translate to giving the participant and user of the interactive system at least a hint of how input and output of a system relate to each other in order for her to make sense of the apparatus' opaque interactions. For the *Projectionist Orchestra* project, I attempted another scenario which invited anyone independent of their level of practical knowledge and experience to participate. I had imagined this piece to enable a dialogic encounter between the participants themselves *through* the interactive system.

However, the result seem from my performer's perspective was different. While it did successfully integrate the outside audience 'into' an interaction with the apparatus, the coordination amongst the participants turned out to be difficult. Participants seemed unable to act in direct response to each other – the majority of users of the system focused on the free exploration of the afforded sonic possibilities. The beginnings of each performance were chaotic or in dialogic terms: no one seemed to listen *to* each other, there was no obvious *responsibility* in the music and interaction. In response I felt a need to facilitate more structural coherence and proposed an interaction with the participants through written instructions: I, the algorithmic performer, would write text in the form of suggestions for action or commentary on the emerging structures of the music, and the participants could respond to these by changing their gestural interactions. In summary my text-based intervention revealed a new way of performer–audience interaction in live coded performance which produced a specific proximity in the relationship between myself as the algorithmic performer and the audience in the room<sup>59</sup>. On top of that the feedback from the participants indicated

---

<sup>58</sup>Most prominently features the piece *The Projectionist Orchestra* (see section 2.3.4) as well as a two week residency in the OpenSpace of the AxisArts Centre, MMU Cheshire in february 2014 during which I created an on-going interactive installation interrupted by occasional performative interventions in the space.

<sup>59</sup>The mutuality of this experience is based on conversations I had with participants of the performance who expressed the described kind of involvement in the process.

that their experience of the system was similar to how I, as the author, had conceived of it. By this I mean how in the process of interacting with the provided touch interface of each mobile device the same questions came up which face the free improvising performer: How to proceed? What to do or play? How to exactly interact with others? In this way the changes in the spatiality of the performance environment and the roles of audience and performers made it possible for the participants to get a first hand experience of the human-apparatus interaction in free improvised performance.

### **3.4. Meso: Performance Collaborators, Improvising Group**

On the meso-level all communicative exchanges within the improvising group take place. As introduced from the lineage in section 1.4, in FIM:

[d]ialogue sits nicely in a situation where many people are ‘speaking’ musically to each other around a common ‘topic’ or referent (Lewis, 2013: 255).

The question is then who is actually speaking and how are they interacting? Based on research into the cocreative aspects of group improvisation this project has taken a methodological approach which values this influence of the musical collaborators during performance as a method to develop the dialogic coding performance.<sup>60</sup> The following analysis aims to highlight how interactions on this meso-level may create dialogic relationships between the algorithmic performer (with her apparatus) and the other improvising performers with their respective instruments. To illustrate the various simultaneous threads of communication in different forms occurring in my improvisation practice I present a diagram of the ‘meso level interaction’ (see fig. 8). This helps to show how each agent communicates in different ways with the multiple partners in dialogue.

#### **3.4.1 Compatibility of Players**

Following the approaches of Zorn and others group improvisation puts a particular focus on the *personality* of the participants in group interaction. The musical interaction is regarded as an exchange between improvising subjects which is ultimately about articulating personal differences and human freedom, as Fischlin et al. (2013: 11) assert:

---

<sup>60</sup>For example see Berliner, 1994; Prévost, 1995; Sawyer, 2003; Nicholls, 2012; Fischlin et al., 2013; Heble and Wallace, 2013; Waterman and Dawn Smith, 2013; Wilson and MacDonald, 2015.

The power of improvised music is to articulate and embody an event horizon of what is creatively possible and thus gives voice to a fundamental right to speak freely, to speak in compelling ways about the human condition.

In this sense my strategy for creating different encounters with others may be viewed as another way to articulate and 'discuss' subjective versions of 'free speech' with others through music. Out of the variety of instruments I improvised with the performances with other laptop performers stood out for several reasons.<sup>61</sup> Firstly, these were more successful in creating an empowering experience for myself because they often allowed for a 'laminar'<sup>62</sup> form of improvisation. All voices of the participants could be 'spoken' freely and develop in a productive and continuous coexistence, an aesthetic of an infinite soundscape reminiscent of the work of AMM. This stands opposed to the pointillist or atomic aesthetic of continuous interruption of emerging musical structure – common in some traditions of European improvisation.<sup>63</sup> The affordance of live coding as an interface strategy makes it less suited to participate effectively in a group improvisation with many sudden changes of direction. From the perspective of the instrument the key difference between laptop improvisation and an improvisation with acoustic instruments was that in the former both (or all) performers would *mutually* develop their instrument's character *while* they were simultaneously playing it, while in the latter the instrument was *more* fixed and constrained to begin with.

#### 3.4.2 Listening to the Yarn

From a performance scenario of all-laptop ensembles, that is strictly 'voices' of electronic sound, a new challenge for interaction originated: the ability to hear all other performers as well as oneself. In a traditional rock band or jazz ensemble every instrument's (or player's) 'voice' is clearly distinguishable because usually

---

<sup>61</sup>In my practice I have played with different laptop ensembles between 2 and 6 players: PBUP, BiLE, SUNPYX study group (Berlin), MMULE (MMU Laptop Ensemble) as well as jam sessions with laptop and non-laptop players. Not all but the majority of these performances were live coded.

<sup>62</sup>The laminar approach of 'textural improvising' (Bowers, 2002: 22) stands in contrast to an 'atomic' approach which 'consist[s] of microscopic elements arranged in relationship to each other' (ibid.).

<sup>63</sup>One such example is the particular approach of playing freely improvised music associated with the European or British improv scene. Such a 'style' includes frequent changes in tempo and pulse, high density and virtuosic playing – the music is generally transitional, highly interactive but resists to demonstrate obvious structures (rhythmical-, melodic- or formally). Because of the character of the laptop as instrument through live coding I found it difficult to follow this type of virtuosic playing. Any of my actions would often be 'too late' in regards to the interaction of other instrumentalists and thereby somehow disturb the collective emergent music.

only one or two instruments of the same kind are present. Within a laptop ensemble particularly in free improvisations this is different. In the case of *powerbooks unplugged* shows this challenge may also be intentionally exploited as an artistic choice. If the apparatus is able to create *any* type of sound how does one know who plays which sound at any given moment? In the *SUNPYX* project I faced exactly this scenario: Of the six laptop performers playing in parallel no one was able to say with certainty who was playing which sound. The effect was aggravated by mixing them into two channels of a stereo PA system.<sup>64</sup> Even though each participant knew the individual timbres<sup>65</sup> more or less, the sheer amount of full-spectrum electronic (and mostly atonal) sounds made it difficult to attentively listen with a focus towards each particular one. As a consequence we decided to introduce means of structuring our interaction with the intention of 'freeing up' space in the sonic landscape. One approach used a limited set of three juggling balls thrown around the group, a second approach explored the use of a ball of yarn. Only the participants who possessed a ball were allowed to play/make sounds. In the second approach each performer had to stay sensitive to pulls 'coming in' on the yarn wrapped around his or her wrist and then play in response to the pull of the yarn. The performers were allowed to pull 'back' into the yarn-structure to trigger sonic responses from others. As a consequence, for the performers (as well as the audience), the introduction of a visual structure into this performance rendered the musical interactions more legible. This worked on the level of the performers as well as on the level of the audience who commented positively. The image of the net of yarn represented a fitting metaphor of the technological entanglement all performers had with their apparatuses.<sup>66</sup>

### 3.4.3 Amplification/Silence

The experience of the *SUNPYX* project revealed a further insight into the interactions between algorithmic performers in an improvising group. While a laminar approach made it easy to be immersed in the experience, the same

<sup>64</sup>This situation must be imagined as follows: The performers were seated around a round table while the two speakers of the amplification system were on one side of the room. The multiple performer's voices were thus placed within the narrow auditory space between the speakers so that their position did not correspond to their physical position in the room – which would have helped the localisation process.

<sup>65</sup>We had agreed in the process of preparing our performance that everyone should develop a consistent and recognizable 'instrumental voice' as to make it easier for others to recognize it.

<sup>66</sup>Find a recording of this performance here: <https://www.youtube.com/watch?v=nJxbz1ATDgQ>

situation could produce an experience of disorientation when more direct interaction with others was aimed for. As the aforementioned case demonstrated, the solution was to devise a context in which the interaction could take place (i.e. by introducing a rule for when to play or not). In musical terms to play or not to play relates to the dichotomy of sound and silence. For the case of an acoustic instrument which always needs an energetic input (cause) to produce a sound (effect) this perspective might not be obvious. The problem is raised because of a computer's effortless sound production. The apparatus will simply continue to play what the program defines until it is programmed in a different way. When working with repeating structures (looping patterns) or sound processes which play continuously (drones) it is easy to lose sight of the actual possibility of stopping the sound. However in some situations there was a demand for communicative exchanges between players which were perceived as interrelated – for example when the music was intended to be more structured through coordinated actions (i.e. a collective gesture of starting and stopping a pattern in the *Technospaetzle* project).

The required ability to not-play (or play silence) operates on three levels: silence marks the end of the act and similarly for the musical gesture, silence marks beginnings or ends and the loudness helps to express an affective quality. Thirdly, silence or playing at low volumes offers space for others to play in and increases the overall transparency of the collective music from the listening perspective.

Throughout the development of my practice the parameter of volume came to be more and more important for the quality of the improvisational experience. While in the *Drums'n'Algorithms* sessions the musical interaction developed into a form of 'musician follows algorithms' with the human drummer at best figuratively complementing the independent playing of the apparatus in regards to rhythmic structure and volume dynamics. This changed significantly later. The solution was to provide direct hands-on control of the amplification to the performer in the improvisation. This embodied control enabled a much more flexible and 'instrument-like' way of playing. As a consequence, subsequent group improvisations with the *Kuehlspot* ensemble were more successful in how they allowed for more directly responding musical interaction during the performance. The musical parameter of amplification was a facilitating factor in these moments of intersubjective dialogue in group improvisation.

### 3.4.4 Strategies for Interaction

The context of FIM already proposes a number of interaction strategies which may support dialogic interaction. At the basis of any improvised interaction lies the ability to know when to act. This is expressed in the greek terms of *kairos* and *metis* (Dell, 2002). *Kairos* refers to an opportunity or propitious moment for an action or a decision to act within an ongoing stream of actions or time. For Eric Charles White (1987) *kairos* is a principle of invention or a 'radical principle of occasionality that implies a conception of the production of meaning in language as a process of continuous adjustment to and creation of the present occasion (in Cocker, 2013: 74). The speech or thought uttered is alert to its own occasion. To perform in a *kairotic* way is necessarily interventionist and inventive rather than obedient. However, *kairos* has little power on its own. It requires the perceptions and actions of an individual capable of seizing its potential. This capability is *metis*, 'a wily intelligence capable of seizing the opportunities (*kairos*) made momentarily visible.' (Cocker, 2013: 75). This is the 'art' of preparing for what could not have been anticipated or planned for in advance. Both *kairos* and *metis* are the fundamental skills or the real virtuosity of the improviser: to know when to play and what or how to play. What a concrete action in a *kairotic* moment might be remains open to the specific situation. In group improvisation the perception and recognition of the actual moment for when to perform an action is led by how the performer perceives the emerging structure of the music, or what Bakhtin has called the responsiveness of one's own responding word. In my subjective experiences of Dialogic Coding practice *kairotic* moments from the improvising group often posed a challenge to respond to because they occurred unexpectedly and seemed to demand immediate sonic action. This unpreparedness resulted from acting in a mode more distanced from the social group: that of programming. Through functionary mode of interaction, simply activating any of the sounds readily available might have worked out – yet with the risk the result of such action might be too uncontrolled or too predictable (musically uninteresting). The point here is that on one hand, the different modes of apparatus interaction follow different perceptions of *kairos/metis* in relation to the different virtuosities and the difficulty is in any case to seize the propitious moment – musically and/or algorithmically – before it has vanished in the dynamic forward flow of any improvisation.

### 3.4.5 Group Intimacy as Facilitator

Wilson and MacDonald (2015: 11) further proposed that contributions of improvisers are constrained by the identities they perceive of their improvising partners: 'they choose to play or sing material that they view as consistent with fellow improvisers' musical tastes and objectives'. The choice of decisions depend on how well the performers know each other and for how long a performance is agreed to last. In shorter encounters a more cooperative exchange dominates. This perspective resonates with my own participation in the 'Improv Summit' sessions at the AxisArtsCentre, March 2014. For this event the organiser Adam Fairhall proposed to play in groups of three or sometimes four and offer short performances of 3-10 minutes. In this specific situation my participation with the apparatus was strongly constrained by the affordance of the short time frame. As a consequence I would start with one musical idea, translated into code and then play with it in response to the others actions. When I attempted to introduce new elements into the performance they were responded to only occasionally. However, when reviewing a recording of the event<sup>67</sup> I discovered that – contrary to my immediate experience of the performance – there were many more related moments of dialogic interaction in the music than I had perceived. From the more distanced position of listening I was now able to perceive connections also between musical gestures in a more asynchronous way. A conclusion from this observation might be that the perception of a dialogic relation as a performer in the immediacy of an improvisation (who may be actively seeking interactional as well as musical responses) may be different to that of an outside listener (who is able to attend differently to the relations within the musical exchanges).

My experience from these short improvisations may be described in what Stewart (Heble and Caines, 2014: 213-218) termed a concept of *improvised dissonance* which are 'processes of musical negotiation' initiated by changes such as 'the proposal of new pitch material, rhythmic concept, new sonorities or perhaps a new volume level.' For Stewart all improvised music alternates between acts of improvised consonance (the sounding together) and improvised dissonance. However it needs to be noted that an improviser's decision to play in a dissonant way might not be recognized as such by a listener (from within the group as well as from the outside). Whether any action is considered as a response to what was

---

<sup>67</sup>A recording of this jam can be seen here: <https://www.youtube.com/watch?v=8nzSHxj6LYs> (session1) and <https://www.youtube.com/watch?v=GME-6OFdp-s> (session 3)

played before – contrasting or continuing the previous idea – is always the interpretive choice of the listener as Wilson and MacDonald (2015) note. For the improvisers themselves, the choice for how to act (in response) is not always clear. Many improvisers describe an improvisation as beyond them – 'a thing (...) shaping itself, rather than arising from the conscious efforts or influence of those taking part' (ibid.: 7) they also perceive their own agency in relation to the developing music as different and changing throughout an improvisation. This resonates with my own experience of a fluctuating agency in relation to the collective process, particularly in configurations of laptop and non-laptop performers. Importantly, Wilson and MacDonald (ibid.: 11) ground this perception of agency not only in the musical interaction but, 'in the social context and the tastes and identities [an improviser] constructed for other members of the group'. This thereby shapes her evaluative process. In other words: the social structure in the improvising group strongly influences how an individual improviser perceives her agency in relation to the rest of the group. This perspective may relate to notions of mutual trust in a group (which includes technological agency): In a social environment with a high level of trust I felt more confident to perform and make decisions for contrasting actions with the intention of achieving novel directions within the improvisation. However, the same actions could have a different character in another, less trustful environment. Similarly in well-known technological environments I feel confident to play whereas an unknown or not-understood environment will reduce my willingness to take risks. The experience of performing the piece *MindYourOwnBusiness* with the Birmingham Laptop Ensemble provides an example of how differently setup keyboards interrupted the flow of live coded interactions in a non-productive way.

# Meso Level Interaction

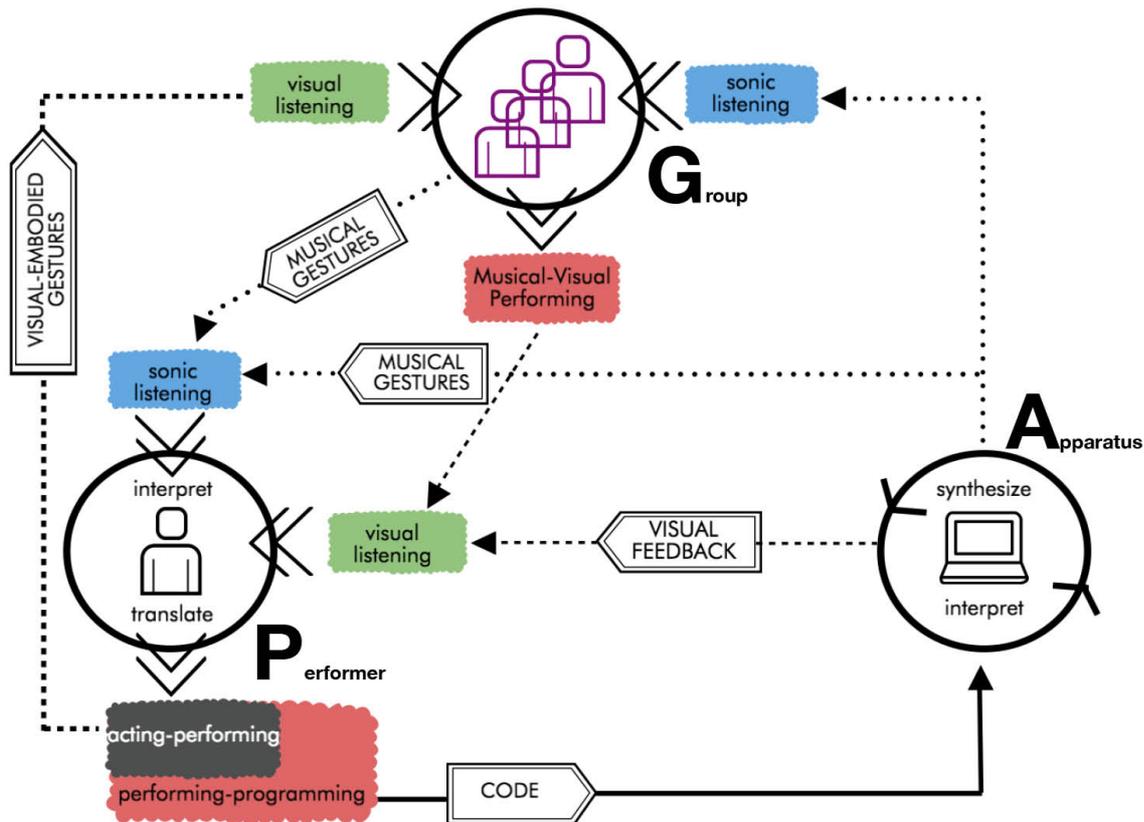


Figure 8: Meso level interaction between performer (P), apparatus (A) and group (G)

## 3.5. Micro: Interacting with the Apparatus

On the micro-level the interactions between the performer and her instrument (the apparatus) take place. Most of the details of this interaction are hidden from an outside perspective behind the screen of the laptop.

### 3.5.1 Press, Push, Click and Type: The Ontology of Interactions

The interaction on the micro-level takes place in the interfaces of the apparatus which include:

- a) the keyboard;
- b) the mouse pointer;
- c) the computer screen;
- d) external devices.

External devices may include remote controls which provide more haptic or tactile access (i.e. through knobs, faders, buttons or touchable surfaces).

The 'language' spoken in the physical interaction with the computer is thus dependent on what kind of gesture the interface affords the performer for input: a) key strokes to write phrases or directly execute pre-defined commands in the

programming language, b) moving the pointer with the mouse to c) click some graphical representation of a computational process. This interaction with the apparatus is physical and includes movements of the eyes, arms and fingers. However, these gestures are received by the apparatus as discrete commands for actions (i.e. a key stroke or button press). A performer is unable to express and convey a specific emotion through a change in how that gesture is performed (i.e. through a grand or reduced gesture). In the context of musical performance this is equivalent to expressiveness in playing – like the tone of voice in speaking. There are several solutions to cope with this problem. In my practice I have used (MIDI) controllers to provide the possibility for a continuous control input (i.e. faders and knobs). While these have the advantage of providing a tactile object for the fingers to interact with (and maybe hang on to), the diversity of the actual touch gesture is, again, not represented. The controller, however, did allow me to manipulate sound processes in such a way that I perceived the sound as flowing 'directly under' my fingers. Through this interaction an immersion in the process of improvisation was greatly increased – as will be discussed in chapter 5.

Still unsatisfied with the rigidity of the controller, I attempted a second approach: to work with actual touch interfaces. In the majority of handheld computing devices touch (electric capacitance or human skin conductivity) is the main sensing technology for human-apparatus interaction. But this is usually programmed to respond to discrete commands (key strokes or clicking of symbols). In the *SUM* project the focus was on the 'pure touch' interaction. I developed a mapping for the digital sound synthesis which 'transferred' the continuous control signal and did not reduce it to simple on/off switching functionality. As a consequence, the *SUM* cube interface makes it impossible to do a discrete action – cause and effect is disrupted for the performer. A touch gesture will always result in an array of possible trigger actions. This type of interaction brought me closer to a more dialogic interaction with the apparatus. The possibility for disambiguation is, as will be shown, a central aspect of dialogue.

Another dimension of this challenge to the discreteness of the apparatus' input is the communication on the macro-level with spectators who are enthusiastic not only to hear, but also to see a performative display of actions that somehow relate to what they are hearing. Bowers (2002) calls this the 'legibility' of performative interactions which depends on the organisation of the electronic devices on the

table: the 'performance ecology'. He recommends that the design of this ecology should make it possible 'for performers to engage with that aesthetic [of FIM] in a publicly accountable fashion' (Bowers, 2002: 57). In an examination of more dance-oriented practices by Djs and EDM live performers, Butler (2014) brought Bower's concept of legibility together with the approach of 'conveying liveness' in laptop-based performance. A live electronic performer will enact 'performance' as a ritual through her own performative actions with visible objects. In this situation the musician's body serves as a carrier of meaning to express involvement in the music as well as personality and style. Butler describes this as the 'passion of the knob' (Butler, 2014: 102) particular style of physically playing the apparatus. Particularly in the *Technospaetzle* project the different *ecologies* of myself (laptop computer, small MIDI 'faderbox' controller and gamepad) and my collaborator (16-channel mixing desk, a few hardware effect panels and joystick controller for the laptop) produced different kinds of expressivity in our playing. While his playing succeeded to convey liveness in a more obvious sense, my live coding interaction on the other hand was not easily visible for the audience.<sup>68</sup> As a consequence I developed a more 'expressive', that is more visible way of typing on the computer keyboard in order to express at least my personal state of involvement in the process and more clearly mark the actual key strokes of code execution (as a triggering action). In conclusion, the adoption of a more expressive 'coding style' – together with the earlier mentioned possibility for more tactile control – actually helped my immersion in the improvised process while interacting with the apparatus.

### 3.5.2 Listening to the Apparatus

A computer can be interacted with in both, dialogic and monologic ways. A dialogic moment may be identified whenever an interaction with that apparatus produces a change in the performers own thinking and acting. Computer 'errors' – whenever something does not work or operate as the user expects – during the process of interaction may thus be considered moments which facilitate such critical reflection. Whenever something does not work or operate as the user expects. On the other side there is the aspect of immersion and attention to the other. When the performer is able to surrender to the moment (and to the apparatus), she

---

<sup>68</sup>This analysis is based on informal conversations held with the audience members after the performance.

consequently others herself. If a laptop performance, live coded or not, merely consists of a virtuosic execution of rehearsed interactions, i.e. a programmed sequence of gestures, then it does not necessarily facilitate a truly dialogic relationship. Of course this holds true for the reversed relationship: if the apparatus dominates the interaction in which the performer only 'functions' in the logic of the program (she reacts in a predictable way from the perspective of the apparatus), then it may be difficult for the performer to experience this as a dialogic interaction (it does not allow for action against the apparatus logic).

In my practice I have experienced similar 'uninspired' moments in performance. This struck me most in a solo performance<sup>69</sup> which I perceived as disoriented and lost while I developed the improvisation through live coding. At the time I identified a lack of solo performance practice, musically as well as technologically, as the reason for the perceived failure. However, from the perspective of dialogic theory, the reason may actually have been that I was not surrendering myself completely or in other words, I was not listening attentively enough (with my whole body) to what was 'out there'. Such a listener orientation may discover inspiration for action not only in the sonic environment but also in the virtual environment of the computer and its interface. This comprises the programming language of code objects, the interfaces on the computer screen, the saved code files in a folder that one may choose from, or a choice to use and interact with a specific available input control (keyboard, mouse, touchpad, camera or microphone). It is obvious that the apparatus may not 'act' in the same way as another human partner would. The apparatus does not normally propose a choice or an action. This particular 'listening' by the performer is thus rather a self-aware listening to what one's consciousness may propose as a response to the visual environment on the screen. Similarly to dialogic solo improvisation, here listening may point into two directions: towards the inner (one's imagination) as well towards the emerging structures in the sounds (see Lewis, 2013). In conclusion, *listening* on the micro-level of the apparatus is a multi-modal approach. It includes hearing and interpreting auditory information, as well as integrating all other stimuli, in the various technological media.

---

<sup>69</sup>View a recoding of the solo live coding performance at Hope University (2013) here: <https://www.youtube.com/watch?v=EBi0DYOUb6Y>

Furthermore, a dialogic encounter with the apparatus implies that a performer's conception of self (identity) is constructed in relation to the nonhuman other. In other words: the challenge for the performer is to *imagine* the apparatus as a dialogic partner. With this perceptual shift a performer is able to see her own actions as responses to this other (and not only as her own actions) – and vice versa.

### 3.5.3 Speaking to the Apparatus

The counterpart of listening as a fundamental element for dialogue is the ability to *speak*, that is, to actively participate in a communicative exchange through the contribution of one's voice. In exchange with the programmed apparatus such a voice must be formatted in a language which can be decoded to be 'made understood' by the nonhuman partner. In most live coding cases, this is the specific programming language which one chooses to work in. Any (i.e. musical) idea that is created by a performer's imagination needs to be *translated* into this particular language in order to be *expressed* as sound by the apparatus. The way in which the apparatus processes these code phrases presents another barrier of discreteness: The interpreter<sup>70</sup> of the programming language receives the executed code commands from the human programmer and checks this for syntactical errors. It will only begin to transform the code phrases once all symbols can be related to a unique meaning – a programming language does not allow for ambiguous or fuzzy meanings. A decision is binary – yes or no. This describes a fundamental difference between a spoken languages (i.e. English) and the abstract functional language of code:<sup>71</sup> An English phrase may be interpreted to mean different things; a 'code phrase' will, in principle, always yields the same response. This points to the basic asymmetry between a human improviser and the nonhuman apparatus: A computer can only operate with unambiguous utterances, while meaning-making in human communicative interaction is never exact. On the basis of their experience human actors are also able to 'repair' syntactically incorrect or incomplete utterances by inferring their meanings – which

---

<sup>70</sup>The interpreter is a function in a programming environment which analyses and executes a program line by line. By 'executing' a line or a block of program code, the programmer activates the software's *interpreter* which then performs the transformation of the written code phrase into action on the hardware.

<sup>71</sup>It needs to be noted that while the described functionality is the predominant paradigm, there exist programming languages which operate differently as well.

may also lead to misunderstanding. Moreover, the intentional misinterpretation of utterances has also been used successfully as creative tactic – making this problem even more complex.

As I have hinted above dialogue may produce a transformation in one's thinking through challenges. Simple obstacles in the human habitual ways of interacting such as the pressing of an unintended key on the keyboard or the unexpected result of a wrongly understood code object may provide such challenges. In Dialogic Coding both dimensions of the interaction, the physical and the intellectual, provide such obstacles for human interaction. The programming language is rather non-intuitive to 'speak' for a human performer, it needs to be carefully constructed and the activity of pressing keys on the keyboard does not produce direct (sonic) results but is somewhat delayed – thus it only works indirectly. The complete act of articulating a performer's voice in the language of code is not synchronous to the performer's thinking but the opposite: it is highly mediated and delayed. It is no surprise that bio-feedback sensing technologies such as the measuring of brain (as early as Alvin Lucier's piece *Music For Solo Performer*, 1965) or muscle activity (i.e. Marco Donnarumma's *Music For Flesh II*, 2015) are popular ways to interface with the computer with the benefit of shortcutting the described delay in the process of articulating a speech act.

In Dialogic Coding practice I have come to value the importance of Bower's notion of the *performance ecology* for being able to smoothly articulate myself through code. In the piece MYOB the original concept was the radical sharing of various musical parameters amongst the group of performers. These individual roles of 'melody', 'rhythm', 'timbre' and 'effects' were then rotated in the group and in the course of one performance – every performer should 'play' each part at least once. As a consequence we collectively came up with the idea to rotate seating positions in order to make the role-switching more legible for the audience audience, this being a development resulting from the *SUNPYX* project. From this an unexpected new challenge emerged: each performer was required to perform through a variety of different laptop interfaces as set up by each member of the ensemble. Even though these interfaces were not fundamentally different for each performer within the ensemble (all built-in keyboards, three Apple machines, one IBM), the complication arose through differently set up keyboard languages (American, British, German). In the live coding interaction with the computer, my own habitual

ways of locating the symbols that I needed to strike led me to repeatedly pressing the 'wrong' keys in another language setup. Accordingly, the intellectual impulses for live coding action were massively blocked by typing errors. The experience of that performance was frustrating in how little I was able to directly articulate my thoughts into sounds, and at the same time the way in which the interaction with the computer was so provocatively difficult, made for an invigorating experience.

In conclusion, the ability to *speak*, that is, to interact with the apparatus in articulated ways must be learned and practised, particularly in complex chains of actions such as live coding. Barriers in the process may also productively challenge the improvised interaction to generate other forms of expression for both the embodied and the musical gestures.

### 3.6. The Apparatus as Dialogic Partner

In the previous sections we have examined the various forms of interactions between the human performer and other improvisers, as well as the apparatus. The discussion now shifts to the perspective of the apparatus. *How is the apparatus able to listen and speak? How may it interact in a dialogic way?*

#### 3.6.1 The Problem of Computational Perception

The most basic condition for dialogue is the ability to hear and listen. In other words, to perceive someone else speaking. To consider the programmable computer capable of a subject-subject relation suggests that it should have that potential for being self-aware, independent and able to respond just as a human. Cybernetics (von Foerster, 2003) describes this functionality in living beings as *autopoietic*: a self-regulating system which is capable of computing its own organisation. The autopoietic system is open to its environment, but organisationally closed. Obviously, this model does not straight-forwardly apply to computers.<sup>72</sup> However a computer may approximate such a systemic behaviour and then also become able to interact dialogically with an algorithmic performer. As illustrated in Fig. 7 (meso level interaction), the communication between algorithmic performer and apparatus is asymmetric in relation to the improvising group. While the performer is listening to the *musical* improvised activity, she interacts with the apparatus through program code. On the opposite side, the

---

<sup>72</sup>The ongoing development of artificial intelligence and robotics may eventually close this gap.

improvising performers perceive both, the visual-embodied coding as well as the sonic activity of the apparatus as the contributions of the algorithmic performer to the improvisation. I have termed this configuration the 'mediation triangle' for how the communication works in indirect ways between group and apparatus, mediated *by* the performer – as a kind of 'human filter'.

### 3.6.2 The Problem of Reflexivity and Action (Responsibility)

The second challenge for the dialogic apparatus is the condition of reflexivity and responsibility. These are similarly important elements of dialogic interaction. It is the ability to perceive oneself 'through' the other, or rather: in relation to the other. Reflexivity is based on perceptual and interpretive abilities at the same time. It requires an awareness of oneself as well as the perception of the other. This reflexivity lies at the basis for every act of meaning making (to interpret an utterance in relation to oneself). Furthermore, without reflexivity it may be impossible to decide when to contribute to an improvisation (*kairos*). Following this standpoint it would not be possible for the apparatus to participate in an improvised dialogue in the way a human performer would respond. Again this suggests the performer as a necessary mediator between the apparatus and the performance environment in order to create an 'algorithmic' self or identity of the apparatus. The concrete affordances of a particular performance situation in regards to the interactivity and agency of the improvisers are difficult to imagine outside of the immediate event. Following Kanellopoulos (2011: 122):

the process of improvisation creates an attitude of consciousness and a mode of being in the world where art and life are united through a deep sense of responsibility to sounds and musicians.

I propose that a reflexivity or responsibility of the apparatus (toward the improvising group) can only be developed adequately in collaboration with a programming performer (who senses deeply) in the situation of performance. The programming performer then interacts with the apparatus to create its algorithmic responsibility in the live. In other words: the performer holds various responsibilities through the different modes of interaction: That of the functionary-performer (her own) and that of the programmer-performer (of the apparatus). Algorithmic *kairos*, or the ability of the apparatus to act reflexively and self-responsibly, does not exist prior to performance, but needs to be constructed and

refined during performance – driven by the programming interaction of the performer.

Beyond this direct intersubjective action there is the linguistic dimension in which an algorithmic kairos may well contribute to a dialogic apparatus interaction. On the syntactic level of coding automatic suggestions for code words may be proposed by the apparatus to the typing performer. This can propose unexpected directions in the programming – a form of auto-completion. This function is significant for live coding efficiency during performance. In my practice this has facilitated a quicker development of live coded sound (reducing the aforementioned delay through typing) and in this way made the immersion in the activity of programming (as flow) easier.

### 3.6.3 The Problem of Nontriviality (Alterity)

Live coding is a non trivial activity anyhow, as in many cases (like sound synthesis), the relation between code and result is necessarily nonobvious – the resulting surprises or frictions with intuition or convention are a large part of its benefit (equally for learning, art, and science) (Rohrhuber, in Blackwell et al., 2014: 142).

Following Rohrhuber, the notion of the non-obvious or nontrivial relationship between action and result is not only a benefit of live coding practice, but in this context also a central pre-condition for dialogic interaction. If creativity is understood as 'the ability to come up with ideas or artefacts that are *new, surprising, and valuable*' (Boden, 2004: 1, original emphasis), non-trivial apparatus behaviour may facilitate exactly such creative thinking for the performer – or the group. Care needs to be taken for the implementation however. In some cases of Dialogic Coding practice I experienced the exact opposite situation: the very predictable apparatus behaviour constrained my personal creative thinking process. This became particularly evident in the use of repeatedly playing patterns. Part of live coding techniques is the approach of creating players which 'just play' based on given rules, often endlessly because they will be changed and updated continuously – so there is no need to explicitly start and stop them. The problematic consequence of this approach is that the (silent) musical and compositional thinking needs to be performed alongside the endlessly repeating (audible) sound patterns. The repetition of these sounds may draw a performer's attention so as to interfere with the silent articulation of an imagined new sonic idea. More exactly, it distracts the attention needed for completing the translation

processes for articulating this prototypical idea into apparatus language (code). As a consequence, the influence of hearing the repeating loop slows down or even completely blocks the imagination and thought process. It becomes hard to transform the imagined sonic idea into concrete sound.

As this example reveals, nontrivial behaviour may only support the development of new ideas in the process of improvisation when it appears in a non-obtrusive form. At the same time, if the behaviour is strongly predictable it similarly influences the performer's action and thinking process.

### **3.7. Summary**

In this chapter I have outlined the interactions of the performer during improvised group performance reaching out to all agents actively involved. These interactions with the different agents were categorized into four levels. On each level I have identified particular forms of listening, speaking or acting as forms of dialogic performance. Witnessed by a co-present audience, the relationships not only unfold in the close temporal proximity of a live performance event in the improvising group, but they also extend through time and space to past utterances in various symbolic forms: sound, music, code, gesture, rituals. More importantly for this research are the interactions taking place between the performer and the apparatus. I have described how this interaction takes place and how to develop it into a dialogic relationship. This describes the properties of the apparatus in Dialogic Coding practice: It needs to be able to 'listen', 'reflect' in some ways and thus functions in nontrivial ways. These conditions can be created through programming but need to be implemented by the programming-performer carefully and in response to the situation. The viewpoint of the performer and the various roles she performs in the pursuit of creating this dialogic agent will be the focus of the next chapter.

## 4 – Responsibilities of the Algorithmic Performer

In the last section I have demonstrated how the interaction between the performer – whom I term the 'algorithmic performer' – other improvisers, the audience, and the apparatus takes place in Dialogic Coding performance. In this chapter, I will analyse how these interactions afford the performer multiple roles through which she is able to interact on each level (meta, macro, meso, micro). The four different perspectives I present here are that of the listener, the programmer, the functionary and the actor. These are distinct conceptual roles which are, however, performed all by the same person and thus overlap each other. This describes one of the original characteristics of the *algorithmic* performer: the virtuosic internal negotiation between these perspectives through processes of translation between different media of communication (sound, code, performative gestures, vision) and impulses for action. The chapter will close by highlighting the different responsibilities these roles imply.

### 4.1. A Heteroglot Performer

As I have outlined in the conceptual framework (section 2.4.1) the elementary conditions for a dialogic relationship are the abilities to listen and speak, that is, to express oneself in the specific medium of communication in response to the other's communicative action. Formulated in terms of subjectivity, dialogue requires an ability to see who the other is and hear what she has to say while being aware of who I am and articulate this to the other. The situation for a performer in live coding performance is challenging for it brings together various divergent tasks of 'speaking' or acting: interacting with the apparatus through program code, 'speaking' to collaborating musicians through musical gestures and possibly also 'speaking' to an audience through performative gestures. These 'speech acts' are complemented by a listening perspective which attempts to attend to all these streams of communication simultaneously. Bringing these different, and possibly, contradictory *acts of listening and speaking* into a productive exchange presents a challenge in itself. The benefit here, following Stowell and McLean, is the 'rich' access to music making which such a multimodal approach to music performance provides.

The cognitive load for one performer carrying out both roles [a) livecoding: abstracting and scheduling, and b) gesture-based expression: playing in the

trad. sense] is high, but the combination of continuous organic expression with symbolic abstraction helps to provide immediate multimodal access to the multiple levels of musical ideas (Stowell and McLean, 2013: 3)

The simultaneity of the two perspectives – live coding and gesture-based expression – is a characteristic found in much live electronic music performance with programmable music apparatuses. Butler (2014) has termed this the 'listener orientation' which allows the performers – freed from the physical necessity of continuously *playing* the instrument (through gestures) – to listen more attentively and take an outside viewpoint to their own music making. Butler (2014: 106), suggests that:

a laptop set is simultaneously performance-based and interpretive; it encompasses both the production and consumption of sound. The musician's attitude is reflective (...) characterized by a dual consciousness.

It is particularly this notion of a 'dual consciousness' which resonates with Dialogic Coding practice. Kanellopoulos (2011: 125, emphasis added) sees the performer alternating between two 'positions of consciousness', that of 'empathizing and stepping outside (*immersion* and *outsideness*)'. Which of the performer's activities takes place from the immersed position and which is outside is contingent on the situation. In one moment the very act of listening might be immersing, while the 'programming' is performed from a distance (particularly the intellectual process of developing a code phrase). In another moment programming – as the actual activity of typing and actuating that code phrase – is immersive (i.e. in the form of an intellectual flow with/through programming) while listening views the musical results from a critical distance. The framing of this last case can be reframed differently, too: Such an immersive programming interaction might also be regarded as an interaction in the functionary mode: It is a direct interaction with the apparatus through the *interface of code* which can be performed in an 'empathizing' way. Thus, even though ontologically similar, the difference rests in the intentions behind the actions and the possibilities for action provided in the interface: a) does the action intend to re-invent the algorithmic structure? (programming mode) or b) does the action explore the interface without substantially changing it? (functionary mode). Interaction in the *functionary* mode of interaction is typically through 'empathizing' through action. This position shifts to an outside perspective as soon as the interface requires or affords many cognitive resources of the performer. In other words: the interface then needs to

be understood to use it. This explains how I was only able to attain immersive states in the very activity of programming when I did not need to think about the very interface and my interaction with it (the keyboard, or how to *write* syntactically correct code).

In summary, the shifting between these modes is not exactly conscious and strategic, but rather a motion of gliding into the other position, sometimes realised only after the perspective has changed. This resonates with Bakhtin who, cited in Kanellopoulos (2011: 125) reminds us that:

both of these moments are inseparable in reality. Pure empathizing is an abstract moment of the unitary act of aesthetic activity, and it should not be thought of as a temporal period; the moments of empathizing and of objectifying interpenetrate each other.

In the same way the different modes of action of the algorithmic performer are inseparable. They constantly mix, overlap and alternate in response to the improvisational process and the momentary attention of the performer.

A live coder's multiplicity of roles may be expressed in another dialogic term: Bakhtin's concept of *heteroglossia*, or 'another's speech in another's language' (Bakhtin, 1981: 324). The concept aptly describes how a fragmented consciousness of multiple 'voices' may be expressed in a single utterance. Previously, I have proposed the term *algorithmic heteroglossia* as adaption to the local context. There, I highlight how in the musical gestures of a performer produced through live coding interaction with her instrument multiple voices may be present: that of the human as well as that of the nonhuman. I may shift the ground slightly now by positing the human performer as the exclusive author of the musical gestures in performance. *Algorithmic heteroglossia* may then be understood as a presence of the multiple, possibly contradictory perspectives (voices) of a *performer* within the musical utterances produced through the interaction with the apparatus. The voice which I attributed earlier to the apparatus may be regarded to actually be that of the *programmer*-performer who inscribed it 'into' the algorithmic agency. This perspective is joined by the voice of the *functionary*-performer<sup>73</sup> produced in the live interaction between performer and apparatus. In such hybrid, 'algorithmic utterances' the two main voices – and to a

---

<sup>73</sup>The distinction between programmer and functionary goes back to Flusser (2011) as introduced in section 2.4.4.2. A *functionary* does not alter the structure of the apparatus through re-programming it, but interacts with the algorithms within the given limitations.

lesser extent the voices of *listener* and *actor* as well – bring to bear 'processes of centralization and decentralization, of unification and disunification' (Bakhtin, 1981: 272).

## 4.2. The Listener

It has repeatedly been claimed that a specific form of listening is essential for dialogue to happen; logically it is similarly important for Dialogic Coding practice. The introduced concept of 'listening being' (Lipari, 2010: 348, emphasis added) proposes a new perspective on the activity:

which resides beyond the limitations of language, dualism, and conceptual thought. As a dwelling place for human being, *listening being* can reveal the ethical possibilities that arise when listening begins not from a speaking, but from the emptiness of awareness itself.

This perspective emphasises the self-transcending potential inherent in listening. Lipari argues that while hearing emphasises perception and sensation of sound, listening points to attention and giving to another. Thus, listening is completing the act of hearing by intentionally responding to a call and in this way acknowledging it – or as Heidegger put it: 'We have heard when we *belong* to the matter addressed' (in *ibid.*: 349, original emphasis). Following Lipari this 'belonging' is the self-transcendence of listening – to:

make a space where I am not — where I have, however, temporarily, renounced my projects, goals, and understandings in order to *listen be* with the other (*ibid.*: 350, original emphasis).

This act of being or rather becoming does not imply 'to agree, or to see things the same way, or even come to understand in the same way', but to 'share the experience of being listening' (*ibid.*). Only when understood in this way, listening will make possible dialogic relationships with a transformative potential. The practice of FIM embraces a similar holistic understanding of listening which is at the foundation for making music collectively. Waterman and Dawn-Smith have termed this a 'revolutionary, decentered listening' which, in recurrence to Kristeva's psychoanalytical view, requires 'a certain openness in one's own psychical apparatus, a flexibility that ultimately represents an aptitude for revolt' (2013: 82). Following this perspective, listening becomes a continuous journey into the open on which the listener embarks and:

for which she has no map or compass, pouring herself into the contours of the sounds and gestures of the performance, allowing for detours into unfamiliar territory and possible encounters with the unknown, the unconscious, self, and other (ibid.: 83).

Summarizing these two conceptual perspectives, the conditions for reflexivity and transformation are founded on and expressed in the very act of listening: namely to be open toward the unknown and willing to respond to the calls heard.

#### 4.2.1 Multimodal Listening

In Dialogic Coding practice I follow the conceptual viewpoint of *listening being* and apply this to the situation of human-apparatus interaction. This raises the question what *listening* might mean in this context (understood as a belonging to the matter addressed)?

Given that the interaction on the micro level takes different forms – visual, text-based, embodied and sonic – the 'matters adressed' are diverse, accordingly. Thus, I need to stress that my understanding of the term listening in this thesis is much more inclusive beyond the auditory operating in the visual and haptic domain as well. This perspective serves to highlight the aspect of attending to or consciously giving as elementary basis of a dialogic relationship.

Listening in Dialogic Coding practice attends to a simultaneity of internal and external voices which all 'speak' in a different language. To extend Butler (2014: 106) this attitude is characterised not only by a 'dual consciousness' but rather by multiple modes of consciousness. Following Lewis' (2013) proposal of 'directionalities' in listening based on Gustavsen's (1999) concept of the 'art of listening', I have identified the following five directionalities (or modes of consciousness) in Dialogic Coding practice:

1. **Towards the sounds:** This refers to a listening in the auditory domain where the multiple voices of the apparatus, the other performers and the acoustic environment simultaneously resound. Here, the emergent quality of the music and the individual voices within it are perceived.
2. **Towards the embodied interface:** This listening attends to the properties of the physical interface. Like any other instrument, the specific elements for interaction facilitate some ways of physical gestures over others. In the screen-based interaction this overlaps into the directionality:
3. **Towards the 'intellectual interface':** This 'listening' attending to all activities on the computer screen and making sense of them. In Dialogic

Coding practice the computer screen brings together a complex combination of feedback from the programming environment (editor, console, interaction GUIs) the operating system, virtual communication with other players (chat).

4. **Towards the abstract (Flusser's 'calculating and computing' consciousness):** This is a specific form of listening in a 'programming mindset'. It refers to how musical gestures or sounds are expressed or translated in abstract program code. Some code objects may suggest themselves more than others for such translations. It is a mode of perceiving the world in particles and discrete units.
5. **Towards the inner:** This describes a performer's awareness and attention towards her inner imagination. This listening may discover intuitive or emotional responses to a dialogic situation. Such responses may be triggered by any of the modes above.

The role of the listener of the algorithmic performer in relation to the experiential states of outsidership or immersion in performative actions is two-fold. In the original context of improvisation as well as Butler's laptop performance, listening is the activity which enables a performer to step outside and view the activity from a more distanced perspective. This implies that listening is not an immersive experience. In contrast, listening becomes an immediate experience when performed with an intention of relating to the other, of actively making space for a *listening being with* the other. Based on my own practitioner experience, listening does in fact operate on both sides: as a way to immerse oneself and become only aware of what one is listening to, as well as a critical and interpretive attitude which analyses the perception in an objectifying way. Thus the performer as listener may experience both, an immersed (in a dialogic relation) or a distanced listening.

In the next section I will continue this analysis with a detailed description of the particular consciousness at work in the 'calculating and computing' mode of listening. While the other directionalities outlined above can similarly be found in improvised performance with acoustic instruments, this mode stands out as a unique feature of live coding performance.

#### 4.2.2 The Calculating and Computing Consciousness

In a dialogic relationship with an apparatus, the human may be considered as deeply entangled with the computer in an almost inseparable way. To account for the differences in perceptual capabilities the human programmer performs an *embracing listening* (Gordon, 2011) or a *listening being* (Lipari, 2010). In other words, the performer perceives the world in the categories of the apparatus. This is what Flusser (2011: 18) termed the 'calculating and computing consciousness' which recognises the world as a universe of dimensionless particles which 'can be neither grasped nor represented nor understood' (ibid.: 10). Such a view attempts to deconstruct and analyse the analog, continuous, and immediate reality into singular elements. In the medium of sound, for example, this may mean to interpret any sound as a type of a periodic function within a frequency range, reoccurring in a particular temporal arrangement which may be expressed in a set of abstract rules. This is an 'engineering perspective' of 'hearing' sounds in numeric frequency values rather than the pitches in a music scale or a hearing of a fading sound as an exponential decay rather than as a 'decrescendo'.

Jonas Hummel

+++ April 2013 +++

#### 4) MIMICKRY - 9 minutes of deep listening

For live coding performance in the field. (Unplugged)

*Needs: Portable chairs or blankets to sit on*

The group of performers agrees on a location outside where they want to perform. The criteria for choosing a location are set by the group. At best it is a location which provides a 'rich' soundscape or at least some characteristic sound-elements.

They group should sit not too far apart so they can hear the location and its soundscape as well as each other. Perhaps a sound check is necessary.

A) 60 sec of Listening: All (incl. the audience) should listen closely to the soundscape and what it consists of. Then silently choose one element he/she wants to mimic.

B) Rephrasing: Start writing your sound synthesis recipe which mimics the intended sound.  
duration: 3-4 minutes

C) React: While playing back and refining your sound the performers may also pick up on sounds that others are playing (instead of the soundscape)  
duration: 3-4 minutes

The piece is over after exactly 9 minutes. The use of a global counter/clock is recommended.

#### Variation I

Do not synthesize the sound but record a sample directly from the location and manipulate it during performance.

*Figure 9: Text piece 'MIMICKRY' (2013)*

Such a calculating and computing consciousness requires practice and attention and most importantly a profound knowledge of the technical syntax in which it operates (in the local context this is the SuperCollider programming language). In an act of 'coding the world' the algorithmic performer combines the ability of hearing with the ability to find adequate representations of it in a coded form. The study *MIMICRY – 9 minutes of deep listening* (see Fig. 9) provides an example work in my practice which intends to explore and rehearse this mode of consciousness. The focus of the piece is on the conscious and attentive act of translation from real-world sounds into digital coded representations. As it turned out such translations are never exact. The 'coded' sound will not sound identical to the original sound. This, however, provides a creative potential which may be exploited in performance. The non-obvious character of code and the limited ability of a human programmer to imagine a *literal* translation of a real-world acoustic phenomenon in code form function as a means of innovation in the creative musical process. Such an innovation works well as a contribution (a response) in the improvisatory process for a performer does not necessarily perceive it as her own act for the intention (the imagined sonic result) was different than the actual. In this way the result of this interaction between the calculating and computing consciousness and the apparatus in a process of live coding may be perceived as the 'voice' of the apparatus. This is of course based on the precondition that a performer is open to let herself be influenced and 'follow' in the direction of what the actual result of this translation suggests. In this sense *MIMICRY* provided a way to integrate an acoustic environment into my performance practice while at the same time being in a dialogic interaction with the >apparatus which eventually would lead away from the original soundscape into new sonic territories.

### **4.3. The Programmer**

While the multi-modal role of listening creates an understanding of any utterance or action, there are multiple perspectives for the 'speaking' side as well. I have subdivided this in three further modes: the *programmer*, *functionary* and *actor*. These modes make are distinct towards whom the performer addresses in any interactions and with what intentions. The *programmer* and the *functionary* describe two modes of interaction in relation to the apparatus while the *actor* describes a mode which is concerned with representation of that interaction

towards audience and improvising group. It is important to keep in mind that all of these modes are conceptual categories which overlap in practice. Any specific ontological activity is not exclusively attributed to one of these (e.g. using a keyboard to type is taken to be a gesture in a programmer mode). The purpose of these terms is that they will help make the responsibilities of the algorithmic performer in Dialogic Coding practice better understood.

#### 4.3.1 Programmer and Functionary

Theoretically, the *programmer* holds absolute control over the apparatus. She can completely invent this nonhuman agency by defining its logical rules and functional properties. In other words, she can change the rules of the game which is to be played. In live performance this can be done instantly as the game unfolds. The *functionary* on the other hand describes the counterpart of the programmer. She is acting in response to the possibilities which the programmed system offers her. As the user of the apparatus, she participates in a game of which the rules are fixed, only allowing for an algorithmically defined *freedom of action*. Both of these roles are mutually exclusive. A programmer cannot be the functionary (or user) of the system *while* she simultaneously creates it. Vice versa a functionary cannot change the rules of a system *while* she interacts with it. One is a consequence of the other. In the reality of Dialogic Coding practice, however, these positions rather converge. First, the performer writes (she invents) an algorithm (as *programmer*) to then 'use' it for the musical interaction through acts of changing the control parameters which this algorithm provides (as *functionary*). Thus the performer continuously oscillates between both roles or modes. Based on my own practitioner experience, these modes also overlap in the temporal order of events: the idea for a change in the algorithmic structure may appear while 'using it'. More exactly, the idea for a 'programming' change may be directly informed *by* the use of the running algorithm. This can be considered a dialogic form of interaction with the apparatus in a mixture of both modes of programming and functioning together.

In relation to the positions of consciousness in Dialogic Coding practice – immersion and outsidership – the mode of programming may be considered as an activity of stepping outside, of objectifying and making something understood through intellectual scrutiny while the mode of functioning corresponds to the empathising and being immersed in the computational activity. I need to note that I

have also experienced the mode of programming as immersion in cases when the activity of writing code merges synchronously with the creative thinking into a focused flow of musical-technical thinking. The 'rhythm' of oscillation between the two modes is driven by the quality of this immersion, of the flow of actions excluding any external stimuli: The deeper a performer is immersed she will most likely not interrupt herself through a conscious act of stepping outside. In some cases this act of stepping outside may not be perceived consciously or as one's own initiative, i.e. when the emerging music as a driver for the immersive flow does not engage a performer's consciousness as much as necessary so that she suddenly finds herself listening to the music from the outside. Similarly in the other direction: a performer may 'slip' into an immersive state of either functionary or programmer when the dynamic of interactions involves her (conscious) thinking-doing sufficiently.

#### 4.4. The Actor

The fourth role of the algorithmic performer is that of an *actor*. The *acting*<sup>74</sup> algorithmic performer attempts to provide clues on the interactions of the performer with the apparatus which are largely hidden to outside observers. Additionally the *actor* may help to convey the *liveness* of the performer's actions. *Acting*-performance is directed at the improvising group (meso-level) but similarly to the audience (macro-level). Based on my practitioner experience, however, it is quite challenging to perform this role adequately. The complexity of the situation produces a bias towards the multi-modal apparatus interaction which already consumes a significant portion of cognitive resources making it difficult to direct attention or even physical gestures to other subjects.

In response to this there exists a widespread praxis within live coding performance to provide other means of visual display than the performer's own body: to show a projection of the computer screen in order to expose the typing activity of the programming performer and be able to follow the execution of code lines more synchronously.

---

<sup>74</sup>In recurrence to Kirby (in Zarrilli, 2002) 'acting' is understood as to 'be aware of an audience – to be 'on stage' (...) by energetically projecting ideas, emotions, and elements of their personality, underlining and theatricalizing it for the sake of the audience' (ibid.: 43). This highlights the 'basic psychic or emotional component' which intends to convey 'certain attitudes and emotions' even without physical action. In this context therefore 'acting' refers to a conveying of the performer's attitudes and emotions in the dialogic interactions of the performance situation, particularly with the apparatus.

In Dialogic Coding I have rarely used this method in its simplest form because it does not effectively communicate the performer-apparatus relationship but rather introduces a disorientation for non-expert spectators who attempt to make sense of the technical symbols displayed and are thus distracted from the music and musical interactions.<sup>75</sup> Furthermore, it establishes a polarity in the stage setup of group performance. Suddenly one musical agent is much more visible than others (the apparatus) – however only on a very superficial level because without the knowledge about the compositional/programming processes which have happened before a visitor cannot fully make the interface understood. Thus the information it is capable of conveying is reduced.

However every 'rule' has its exception, thus I did use visual projections in two cases: Firstly, in the participative approach of the *Projectionist Orchestra* the computer screen was projected to provide feedback to other performers on a large scale (in a similar way how the computer screen works for the performer in other cases) and to increase the 'group focus' of the collective improvisation by exposing everyone's activity visually on the same display. Secondly, in the case of the final performance of this research 'live dialogic coding', I showed the screen of my computer in order to expose the different interfaces I was interacting with. The intention here was to expose the rhythm of switching between those and make clear how differently the interfaces are structured. This visual display was however not optimal to represent the group interaction adequately because it showed only one 'site' of performer-apparatus interaction while there were three possible others as well. Furthermore the interdependencies in the networked architecture were not obvious, as well as abstractions in the code used for performances (i.e. clearly indicating which elements were pre-existent and which were rather invented in the time of performance).

In summary, I have developed smaller forms of representation in Dialogic Coding practice which do not need large and potentially distractful projection: the stage setup and the practical visibility of gestures.

#### 4.4.1 Visibility

One form of 'acting performance' in Dialogic Coding practice is that of visibility. Many gestures in a live coding approach are typically so small as to be almost not

<sup>75</sup>It needs to be noted as well that in this research the focus was not set on the development of new ways of expression in performance in the visual domain but rather on the implications for a performer when performing with a programmable but independent instrument.

noticeable (e.g. typing on a built-in keyboard). The solution for me was augmentation, that is, to perform them with exaggeration. Therefore, in more recent Dialogic Coding practice I use an 'expressive typing' gesture:<sup>76</sup> For example the end of a musical train of thought and translation into code will be emphasised with a strong hit of the 'enter' key (actuating the code). This further represents an embodied response to a certain energy perceived in the emerging improvised music (e.g. a fast musical pulse or an energetically played phrase by someone might 'call for' a similarly energetic response). Even though the key presses are indifferent to the velocity of my finger's movement, such more expressive gestures have helped to communicate my own involvement and the liveness of the situation, rendering my performance more legible for outsiders.<sup>77</sup>

The face and upper parts of the performer's body provide other possibilities for visual communication. The *Technospaetzle* project, for example, has revealed through the analysis of performance recordings how different each performer's body language works as a form of acting-performing. My collaborator, Tom Orr, was moving his body in synchrony with the music and operating the mixing desk with clear gestures similar to how EDM performers 'perform performance' in order to signify (and proof) the liveness of their activity (Butler, 2014: 95-105). In contrast, my own body movement was much smaller and could be perceived by an outsider as less emotionally 'involved' in the music.<sup>78</sup> The legibility of such gestures can help to create meaning, particularly in visually or musically structurally-poor situations such as live coded free improvisation. In this way, some of the facial expressions of the live coding performers (e.g. of bewilderment and frustration) which I was able to observe during performances of *MindYourOwnBusiness* have a high potential to work as acting-performance – if they were better communicated to an outsider. The stage configuration between performers and audience is one way to open up the space to come closer and, in this sense, see more for an outside audience. A next step then would be to directly involve the audience not as

---

<sup>76</sup>The reader may, for example, compare the typing performance in the early example of session #A1 with the later session #A2. The stage setup and seating orientation adds to the visibility or opacity of these gestures (Does the laptop lid face the audience or is the keyboard interaction visible?).

<sup>77</sup>What I mean by this 'performative' gesture of typing can be viewed here: <https://www.youtube.com/watch?v=QWXqIP2MHqw>

<sup>78</sup>This difference is evident in this moment: <https://www.youtube.com/watch?v=qVqNo4M1rSI&index=24>

observers but as participants to better communicate the intricate human-apparatus relationships.

#### 4.4.2 Participation

The *Projectionist Orchestra* brings together two ideas: 1) to communicate the specific experience of interaction with an apparatus in a situation of free improvised performance (in the sense of acting-performing), 2) for the programmer-performer to create a scenario in which the power over the interaction is distributed between performer and audience. The the audience controls the musical events, while the performer may exercise only the course of performance. Through the use of handheld electronic devices the piece established an everyday technological experience with an arts context of music improvisation. This is the 'social location' that improvised music practice is able to provide where the performers can review and rephrase their everyday experience.

The performative dimension of improvisation is that of reframing everyday human experience, creating new possibilities for awareness and empathy in musicians and listeners alike (Waterman and Dawn Smith, 2013: 73).

While the technological gestures of swiping may be are trivial and widespread, the experience of playing directly with sound synthesis is not. Furthermore, the *Projectionist Orchestra* attempted to create a dialogic space for interaction in which every participant would be able to interact with others through sound as well as through movement<sup>79</sup>. None of the performances of the *Projectionist Orchestra*, however, showed much collaborative interaction aesthetically. The interface seemed to afford free exploration over sociable interaction. From the point of view of acting-performing it is, however, exactly this aspect in which the *Projectionist Orchestra* was successful. Through the participative approach it managed to communicate the challenging situation of the dialogic coder as a free improviser, that is, how to create a musically coherent structure with the given possibilities of the apparatus interface.

Another way in which the *Projectionist Orchestra* was a successful form of acting-performing was the direct interaction between me and the participants. I was in the position of a 'virtual' conductor (not clearly visible/present) who states recommendations for action on the projected screen in the form of text. The

---

<sup>79</sup>Even the visual space on the projection screen provided unexpected possibilities for interaction: some participants told me after the performance that they had coordinated their actions in relation to the visual activity on the screen and not in relation to sound.

participants were free to follow them. Even though the medium of text was inaccessible for the performers they were still able to respond – through their playing as well as other actions (e.g. speaking, laughing). By establishing this channel of communication it was suddenly much easier to directly interact: in other words, my visibility as a performer was greatly increased by being able to use 'the big screen' which everyone else was able to see (and interpret). Furthermore by transforming the 'acting-performing' from an embodied form into the medium of text it was now much easier for me to communicate because this was on the same level of the interface as the program code.

#### 4.4.3 Reanimating the Inanimate

The last example of my practice for a form of acting-performing is the project *SUM*. In this project I substituted live coding interaction for a more gestural and embodied performance in the interaction with the apparatus as a dialogic partner. As a consequence *SUM* plays with the interpretation of visible performer gestures and the expectations toward their relationship to the acoustic events. The apparatus programmed in *SUM* is partially indeterminate, therefore the sonic effects in result to a performer's gesture are not exactly predictable. For example, if a performer touches the interface at a particular position, the same touch performed later may not return the same sonic result. As a consequence of this arbitrariness, such gestures may be performed in a new 'freedom': that of acting-performing. By this I mean that the autonomy of the apparatus enables the performer to focus her attention on performing these gestures as signifying gestures toward the outsider. The audience is here left to wonder about action and response relate, causally or not, while the performers take turns to put new pieces on the interface. As a performer, I experienced this game-like setup as an oscillation between listening to the emerging structures in the music with an immersive consciousness and intellectually making the system understood to myself in order to decide on future actions. This experience was made possible because I had limited knowledge myself of the internal algorithmic structures of the apparatus, in other words: I could not fully experience the role of the programmer-performer. I was only able to perform as functionary and actor with the apparatus. Again, this made it possible to devote more attention to the performance of these roles which would normally be used for the programming-performing.

## 4.5. Responsibilities for Dialogue

This last section re-examines the roles elaborated above towards how they support the emergence of dialogue. Following von Foerster's (2003: 244) deduction of ethics from freedom and self-regulation, *responsibility* is a consequence of self-autonomy or the fact that 'the causes of my actions are within myself: I am my own regulator'. From the perspective of intersubjective dialogue this may be expressed as: 'in each and every moment I can decide who I am' (ibid.). Fischlin et al. (2013) attribute the act of dialogue-focused listening also an inherent ethical dimension, or responsibility, in how it acknowledges the other. For every role of the performer as outlined I propose a corresponding responsibility for dialogue. It is the specific challenge for the algorithmic performer to bring these into a productive coexistence through practice. These responsibilities are the ethical dimensions of Dialogic Coding practice.

### 4.5.1 Dialogic Listening

As a listener a dialogic attitude 'requires both an awareness of our habitual categories and a willingness to go beyond them' (Lipari, 2010: 354). This works for each of the five directionalities in the act of listening:

1. Towards the **sonic** a dialogic approach implies to be moved, touched or challenged by what one hears. In my own practice such a listening experience has worked particularly well in the medium of electronic sound. Noise music holds a great potential to produce encounters of difference and in consequence de-centre the listener for a transformative experience (see my mentioning in the lineage sections 1.4).
2. Towards the **embodied interface** a dialogic approach may mean to recognize difference in form of interruptions in the flow of actions as a proposal to rethink the ways in which the performer acts on the interface, e.g. to change the gesture of typing or follow what might emerge in the physical interaction.
3. Towards the **intellectual interface** a dialogic approach may refer to how the appearance of unexpected and novel elements in the process of inventing (programming) a musical gesture can be used as a moment of reflection. The performer is open to adapt her creative thought process following 'suggestions' of the interface, e.g. to use a code which the 'help'

reference of the software provided instead of the word which the imagination of the performer had generated.

4. Towards the **calculating and computing consciousness** a dialogic approach operates in a linguistic way. Following Bakhtin any code 'word' already carries all the prior meanings from how it was used before. It is oriented towards specific meanings (or future answer-words).

In my live coding practice I have found that this ambiguity of code words can be a problem and a potential at the same time. It makes it possible to create radically different sounds and interactions from the same code material. The reward from such innovative coding requires curiosity and risk-taking, however, the creative coder may fall victim to the limitations of her own imagination: to play only what she is already familiar with.

5. Towards the **inner self** a dialogic approach means to be open to encounter the unknown, unconscious *of* oneself through close attending of one's imagination. This describes a form of learning about oneself in relation to what one perceives.

#### 4.5.2 The Dialogic Programmer

As outlined in section 4.1 the agency of the apparatus relates to the role of the programmer of the algorithmic performer, that is to say the apparatus' agency *is* the programmer's agency inscribed into it in a different, coded form. The responsibilities for the roles of functionary and programmer are then different and potentially contradictory. The programmer accounts for her actions as inventor of the agency of the apparatus, prior to and during performance.

Following my analysis of how the apparatus may be a partner in dialogue it becomes apparent that this particular responsibility accounts for the autonomy or nontriviality of the apparatus.

In relation to computing machines, autonomy needs to be understood in a more nuanced way – neither only in the solipsistic way of being in a completely self-constructed world, nor in the opposite sense of actively relating to the outside world on the basis of a concept of 'self' and recognizing this world as 'other'. Lacking a self-concept and awareness, the apparatus, can only exercise a constrained freedom to adopt either of the two positions. A radically 'free' apparatus may autonomously decide *not* to do something, i.e. to stop running or participate in an interaction, based on its own logic. The constrained 'freedom' of

the apparatus (its 'pseudo-autonomy') is one which makes it impossible to fully control the system by means of building some form of circularity (a form of 'technological awareness') about the environment (its inputs) into it. This is the responsibility of the programmer's role: to invent algorithms which allow for this kind of 'pseudo-autonomous' behaviour on the side of the apparatus. In turn a *dialogic space of constrained freedom* is created in which the functionary-performer may interact with the apparatus dialogically. This space is constantly challenged by the interactions with improvising group and changed by the programming performer.

Concluding from an ethical point of view the programmer-performer should stipulate autonomy for the apparatus in order to facilitate dialogue. However this call contradicts the programmer's own intentions for reprogramming. If the apparatus is conceived and realised as a 'black box', it does not allow for modification without potential destruction. Such design only allows for an interaction in the mode of the functionary-performer. A black box design thus constrains the autonomy of the performer in the *programmer's* mode. The apparatus for dialogic live coding performance should consequently be re-programmable to allow for a freedom in programming. Thus, the ideal *mutual* Dialogic Coding practice needs to work along the principles of an open and programmable apparatus system. Only on this path the apparatus can be invented *as well as* interacted with dialogically during performance. In summary, the dialogic responsibility of the programmer implements two goals: it combines the invention of algorithmic autonomy with the possibility to re-program (or change) it. This is the essential precondition for a dialogic relationship between performer and apparatus.

#### 4.5.3 The Dialogic Functionary

The functionary accounts for the performer's actions in response to the apparatus *as the other* as well to the actions with the improvising group, all taking place during performance. This implies that the functionary holds the responsibility for all her intersubjectively oriented interactions with a *consciousness of immersion* in the improvisatory process. These interactions follow what Kanellopoulos (2011: 119) has called the 'oughtness of freedom': an obligation to be present to the activity and act accordingly in order to keep it going. In FIM – as a dialogic practice – this obligation is understood that the improviser 'must pursue that which [she] do[es]

not know' (ibid.:122) and resist habitual ways of playing as a form of affirming her musical identity. The functionary-performer must 'internally' collaborate with the listener-performer in order to discover new possible ways of 'functioning'. This relates to the apparatus interaction and simultaneously also to the group interaction. In this way the role of the functionary negotiates between these differently structured levels of interaction.

In my own practice the encounters in free sessions proved to be most innovative for my own functionary-performance. The truly dialogic moments in the improvising with other players proposed changes in my performance ecology and a redevelopment of the apparatus interface in order to be better able to respond to the musical gestures from the group. This experience was informed by the fact that the activity of live coding (the *programming* mode of performance) often functioned as a means of distancing myself from the immediate social and musical group interaction. This subjective distance to the collaborators was significantly reduced by a more embodied interface to control a sound (a tactile controller). In the *Technospaetzle* project I developed a 'fluid' solution to make more immediacy possible without losing the programmer's freedom: I could control important parameters of the sound processes through an external controller while I was also able to dynamically program new mappings for the very sounds or parameters to be controlled. This method allowed me to play some sounds in a tactile *hands-on* manner alongside a live coding approach which, in consequence, led to a more engaging and immersive experience of performance.

#### 4.5.4 The Dialogic Actor

The dialogic responsibility of the functionary-performer extends into the role of the actor-performer when the performative actions are viewed in regards to their potential to signify the improvised dialogues of the performance. As I have outlined in section 4.4 the potential of the dialogic coding approach may be that a performer is able to devote more attention to the acting-performing and thus communicate more actively with an outside audience. As a consequence this may make the experience of the performance as event more engaging for an audience. The dialogic responsibility for the role of the actor-performer is expressed through an attentive listening and presence toward the co-present audience as well as the space of the performance. The effect of such listening may inspire new directions

in the improvised in the interaction on the other levels. My piece *MIMICRY* provides one example of practising such listening.

While listening plays a very important role (the audience-performer communication is often asymmetric), the physical performance may develop towards dialogic communication in the directions of visibility and signification as I have outlined above. The combining of the programmer's with the actor's role may inspire the algorithmic performer to program an apparatus system which enables the participation of the audience in new ways – gesturally, visibly or technologically. The *Projectionist Orchestra* has presented one example for this.

#### **4.6. Negotiating Responsibilities**

The algorithmic performer brings all the responsibilities outlined above together in her person. The challenge is to negotiate the different needs and intentions of each position towards the emergence of dialogic relationships on all synchronous levels. This act of balancing with the aim of creating dialogic spaces which allow for multiple simultaneous 'freedoms' to exist – that of the algorithmic performer, of the apparatus, of all other improvisers and of the audience – is the main responsibility in Dialogic Coding practice.

From this perspective *responsible programming* is a way to *prepare* the algorithmic agency for a particular performance situation by *composing* rules and constraints in algorithmic form within which improvised dialogues can take place. These acts of programming as preparation take place prior to the situation for which they are invented. It is very helpful if the programmer has a good imagination of this very situation in order to define the requirements of the interactions and design the apparatus' functionality accordingly. This includes an expectation of which 'languages', musical idioms, or gestures might be used for communication and how this may be best performed. The virtuosity in this process of designing algorithmic agency describes the thin line between predictable behaviour and alienating complexity. In other words such a dialogic programming preparation aims towards a minimum of prepared material but at the same time a maximal flexibility on *how* these prepared elements may be used and changed in the course of performance. The ability to access and change material as well as control structures can be viewed as belonging to the freedom of the programmer-performer as outlined above. Nevertheless, a preparation can only be a proposal which will be revised and discussed collectively in the improvised group

interaction. In any case a minimum of preparation is needed. Based on my practitioner experience it is hardly possible to participate in dialogic interaction on the intersubjective macro and meso levels without sounds or musical gestures to play ready-at-hand. On the other hand, the programmer needs to pay attention not to 'over-prepare'. This refers to situations in which the algorithmic structures are too fixed and inaccessible during performance so that they impose their own principles (e.g. in relation to harmonic texture, rhythm/pulse or timbre as well as the interaction) onto the group performance. My session experience in the *Drums'n'Algorhythms* project may account for this. Consequently in the development of my Dialogic Coding practice I have pursued a direction which puts change or 'change-ability' as its main goal. It may be understood from this point of view why the restriction on only working with the SuperCollider programming environment deemed feasible: Many other available software environments already come with assumptions about musical process and structure built into their algorithmic form. Each software obviously works best within the frame for which it was invented. SuperCollider as an open environment – not only for sound synthesis but also the design of control interactions – provides a flexible tool for the responsible programmer-performer to develop the algorithmic agency of the apparatus. From my practitioner's point of view, interested in dialogic learning processes, this outweighs the drawbacks<sup>80</sup> of the unspecificity of the tool. A further reward may be that such a tool allows for the development of an individual, idiosyncratic approach to technology-based music performance based on the dynamically developing skills of the programmer. This provides a flexibility also in how the complexity of the programmed system may adapt to the growing expert knowledge of the programmer-performer. In addition to that every new performance situation will actually contribute to this learning process as well – as my methodological approach has revealed.

#### **4.7. Summary**

In this chapter I have described the different roles of a performer in Dialogic Coding practice. The combination of these potentially conflicting activities of listener, programmer, functionary and actor is one of the personal challenges which a performer needs to manage. This negotiation requires the ability to move

---

<sup>80</sup>See Blackwell and Collins (2005) for an analysis and comparison of several software environments for live laptop performance

fluently between the different perspectives and their respective languages of articulation and forms of interaction. I have highlighted, based on my own practice, how listening as well as programming, functioning and acting may be performed towards the emergence of dialogue. In particular, this requires the balancing of the intersubjective dialogic interactions in the improvising group against the dialogic interaction with the pseudo-autonomous apparatus. The possibility to re-program the apparatus establishes a particularly intricate 'double' responsibility: The programmer is responsible for programming a dialogic apparatus which only enables herself *as functionary* to interact dialogically in more direct ways. This all happens simultaneously to the group interaction. The actor carries here a responsibility to signify the dialogic interactions with the apparatus towards the outside (audience and group), while the listener is responsible for a general attention on all levels of interaction in any moment. I have provided examples of how this analysis has emerged from my practice. In the next chapter I will examine the reflexive qualities of the interaction in Dialogic Coding practice in order to see how it may produce transformations for the performer.

## 5 – Dialogic Coding as a Reflexive Practice

In this last chapter of the discussion I will now take a closer look towards how the dialogic interactions produce transformations of the performer in a reflexive way. This follows the notion of dialogic quality in a 'genuine encounter' as outlined in section 2.4. I distinguish two forms through which transformations may be produced: a) through immersion and b) through outsidersness, that is challenge and interruption of flow. I will then propose, based on evidence from my practice, what necessary conditions are for dialogue to emerge – in structural (for the situation and the apparatus) as well as in personal terms (for the actions of the performer). The chapter concludes by proposing strategies for the facilitation of dialogue based on the insights gained through my Dialogic Coding practice.

### 5.1. Dialogic Quality

One central quality of a dialogic relationship pertains to the how the self is created as a function of the other. This resonates with what Bakhtin cited in Nikulin (1998: 382) has called the 'the independence of a character and the unfinalizability of the human being' in a literary context. Following this notion Kanellopoulos sees the dialogic quality of improvised music practice not in how it allows for a productive encounter of previously formed positions or identities but in how this exchange produces alterity:

through the process of collective musical creation where (...) one's identity is constantly being lost the very moment it is being found (Kanellopoulos, 2011: 124)

Dialogic Coding practice is from this standpoint not automatically dialogic. The indicator for 'true' dialogue depends on whether the practice creates a 'form of interpersonality or interculturality' as the 'most productive relationship between self and other' (Peeren, 2007: 18). This points to the essential quality in a dialogic relation: Through an interaction with a different (independent and free) other, I recognize the other's difference and become thus able to alter my own position in response. This changing of oneself is an open-ended and continuous (unfinalisable) process. It may take place between people, texts, sentences, musical gestures – and in the local context between human, group, and apparatus. A transformation, even if only temporary, can take place on the level of thinking or embodied action but also emotionally through a feeling of togetherness or isolation

(immersion and outsidersness). It is this *transformative* quality which I propose as an indicator of the quality of a dialogic relationship in improvised musical interaction. Such a transformation is *liminal* as proposed by Turner (1969): it transgresses between states, categories, or identities. The question is when and how do such transformations happen? Phillips (2011: 32) cites Buber making the point that

Dialogue is an emergent quality of communication that occurs fleetingly and often in surprising rather than in neat, orderly ways, and it cannot be planned or facilitated. 'no one can know in advance what it is that he has to say; genuine dialogue cannot be arranged beforehand'.

Thus, the ephemeral character of any dialogic encounter makes it difficult to clearly identify such changes. It is therefore necessary to look into my personal accounts of performance situations as well as consider the actual entities which are produced in Dialogic Coding practice: the algorithmic structure of the apparatus.

#### 5.1.1 Outsidersness and Immersion

As I introduced earlier, Kanellopoulos (2011: 125) has proposed the concept of 'outsidersness' and its opposite 'immersion to better understand the 'aesthetic act' within free improvised music. *Outsidersness* in the context of FIM suggests:

a need [in improvisation for] (...) temporary finalizations which occur through the presence of the audience and the musicians' struggle to raise their minds 'above' their playing, looking at their music from outside (ibid.: 113).

The act of distancing oneself or of stepping outside is considered the complementary other to the act of empathising with the other individual, text or work of art (music). Through empathising the improvising performer is immersed 'in the invention of sound structures and interaction between musicians' which 'demands their attention and creates an obligation (...) to be present' (ibid.: 119). Similarly in Dialogic Coding practice I have identified these two positions of consciousness – immersion and outsidersness – between which the performer moves during a performance. The experience of these states becomes a reflexive process through the consecutive shifts between those. In other words: a dialogic relationship may first produce an immersive state for the performer continuously *responding to* another's action and may then develop toward a distanced position, in an act of stepping outside the activity and reflecting on it and through this create a transformation in the performer. Formulated into an analytical category I may express the criteria for dialogic quality in Dialogic Coding practice as follows:

- Does the activity provide an *immersing experience*? – This interaction is characterised by a dissolving of boundaries of time, space and self, in the intuitive performance of the task at hand. The awareness for actions disappears, a performer becomes 'one' with the activity.
- Does the activity provide a *challenge/experience of outsidersness*? – This interaction is characterised by disrupting the flow of creative thinking or habitual ways of acting. It challenges how a performer makes sense of what she perceives by confronting her with something novel, surprising, unexpected or unknown. As a consequence of the interruption a performer becomes aware of her own conscious self and may, from this position, reflect on her own actions in relation to an ongoing improvisation.

These two experiences are mutually exclusive for it is impossible to be immersed and unaware of one's actions in an activity yet reflect upon them at the same time. However, they may appear in rapid succession to each other, or as Cox and McLean (2013: 25) put it '[p]rogrammers oscillate between these modes at the level of the human-computer interface, translating between machine behavior and human perception.' In other words, the temporal distance between these two modes of interaction may be rather small. Both, immersion and outsidersness manifest in dialogic *moments* rather than in a continuous way.

#### 5.1.2 Transformation Through a Reflexive Practice

The transformation of the self (as well as of utterances, texts or musical gestures) through dialogic interaction is founded in the fact that it is a practice which transgresses borders of disciplines and conventions, and shifts traditional performer roles and ways of improvising. Turner's (1969) concept of *liminality* has been used to describe such practices highlighting their transformative potential through performative action. Following the pseudo-autonomy of the apparatus, a performer's habitual flow of actions and thinking is interrupted through encountering unexpected behaviour. The more such 'responses' by the programmed agent can be *perceived* as 'other' (in dialogic terms), that is independent of the performer's actions, the easier it will be to accept these, work with them and thus turn the interaction into a learning experience. Recalling the double responsibility of the performer, the difficulty is contained in the fact that the 'programmer-performer' is, of course, the inventor of that same apparatus algorithm and may infer the functioning of the apparatus. However, it is necessary

to 'take a step back' or switch the role/perspective into that of a functionary to make a transformation through performative action possible. Ideally, these insights through doing the practice may make the performer stronger and more confident as a reflected practitioner. I adopt a term by Lewis (2013) to consider Dialogic Coding practice a form of an 'epistemology of self'. This describes the capacity of the practice to provide methods for *producing* knowledge as well as being a form of knowledge in itself.

### 5.1.3 Forms of Immersion

The experience of immersion in the improvisatory activity is aptly described in the concept of flow (Csikszentmihalyi, 2002). The state of flow describes the state of total immersion, a holistic feeling in which the awareness for actions disappears. It is characterised as a:

unified flowing from one moment to the next, in which [the person] is in control of his actions, and in which there is little distinction between self and environment, between stimulus and response, or between past, present, and future (Csikszentmihalyi, 2014: 151).

Attaining a state of flow blurs the boundaries to our environment and our own self. It creates a state of timelessness in which only the momentary present exists. This description reminds of how Buber conceived of a dialogic relation with another individual (see section 2.4.1.3). The game or the activity of *playing* is considered 'the flow experience *par excellence*' (ibid.) yet the playing of a game, on the contrary, does not automatically guarantee an experience of flow. The key sign for a flow state is the merging of action and awareness. This includes the awareness of one's self and identity. Logically, it is impossible to reflect on one's own activity while in flow. The actual flow experience is therefore not constant but rather a moving in and out-of-this mental state. This description resonates with the concepts of immersion and outsidership in the improvisatory process and how they rapidly alternate (experiencing oneself as in- and outside of the activity).

### 5.1.4 Forms of Challenge

The second step in a transformation through Dialogic Coding, is the experience of a productive challenge<sup>81</sup> through the interaction with other subjects, with symbols

---

<sup>81</sup>The term 'challenge' may be understood here as a situation in which the performer is surprised as well as irritated by an unexpected response or behaviour from the partner in the interaction. This response may also interrupt or block the ongoing activity (e.g. the pressing of keys on the keyboard).

of textual or musical form or with the interfaces of the apparatus. Such a challenge takes place in the specific 'language' of the interaction with the other. This includes the language of program code, sound and musical gestures, of performative gestures (the visual and non-verbal). The consequence of being faced with a challenge is an interruption in the flow of activities – thinking or playing/acting. This interruption becomes productive if it directs a performer's awareness towards her own activity and makes reflection possible. These interruptive moments hold the potential to inspire the performer's creativity towards inventing a new successive action. However, if the interruption does not produce a moment of reflection and inspiration it may rather feel like being forcefully restricted by someone else in one's own freedom to act/think. The realisation of such *outsideness* must subjectively be perceived as an active gesture of distancing oneself from the activity and not as a passive one of being excluded from it. It needs to be stressed again that this is a subjective and relative viewpoint: a question of perception and its interpretation. Consequentially the quality of a dialogic interaction depends on the responsibility of the performer to perceive any interaction as an opportunity to dialogically relate.

## **5.2. Interruption and Flow with Agencies in Dialogic Coding Practice**

In the following sections I will analyse how experiences of flow or outsideness were created in my practice and how these resulted in moments of learning and transformation.

### **5.2.1 Free Sessions**

A free improvisation session is a very particular environment. There may be no rules except 'to keep it going.' The interaction is not scripted in any way and the participants have possibly never played together before. It is an arena of almost infinite possibilities in the beginning and then narrowed down by each successive musical contribution in the course of a performance – given that improvisers respond to each other. This 'body of rules and conventions', the collective 'emergent', is the effort of the improvising group in a performance (Sawyer, 2003). My immersion as a participant in the improvisatory process in the first sessions I played was rather challenging – and dependent greatly on the other improvisers (how and what they played). The immersive moments were characterised by a *synchrony* of the 'pulse' of my own interactions with the apparatus with that of the improvising group. I felt as part of the process because my actions seemed to be

able to respond 'intuitively' within an appropriate timeframe to the actions of others. Such moments were facilitated by a better knowledge about the other improvisers – how they would interact and what musical styles they might play. The *Drums'n'Algorhythms*' Project was successful in how it generated immersive states because through few rehearsals we had found a common musical ground on which we could interact in meaningful ways. This helped to 'feel part of the music'. This implies that a particular shared way of playing or common aesthetic may help to experience the performance as engaging. It was in moments when I was playing sounds with a similar timbre when I felt immersed in the collective improvisation. It seemed to me that by using similar sound material I was able to perceive the interaction as more related or responding to each other.<sup>82</sup> This is not say, that I used only sound materials which sounded similar to the instruments of the performance. On the contrary: in the search of a personal 'timbral space' in the musical spectrum of the group it was often helpful to play in a strong contrast to the others. The finding of a shared common language (of any form) in the group proved to be an important condition for immersion. In these cases, reflexivity was driven by a group-focused process through which I reviewed my sounds and apparatus interactivity to eventually change them towards this shared musical language. But actually more often a feeling of 'being part' was difficult to develop and my changes in the setup to discover a common ground received no response by the rest of the group. However, the long term compositional work on my performance ecology, in particular the development of a multi-modal interface to the apparatus, may be seen as a consequence of these challenging experiences. In the early sessions I was relying only on live coding techniques which resulted in shifts of attention every time I mistyped a word or executed a syntactically incorrect phrase. I was not able to stay within my own musical thinking process<sup>83</sup> and also my contributions to the improvised music were interrupted. In response I introduced more embodied ways to play sounds with the computer. The new challenge from this was a lack of flexibility relating to how I could change my sound material or musical gestures in order to actually find a 'free space' in the

---

<sup>82</sup>This is an example from the *Kuehlspot* session in which I used an additive synthesis process reminiscent of an organ sound with a random pitch material distributed on a diatonic scale:  
<https://www.youtube.com/watch?v=BPjN9OjDMhk>

<sup>83</sup>Such a case is illustrated in this example: [https://www.youtube.com/watch?v=1k-NwLN4uMQ&list=PLKRUGJaQJkAfiS7EXUfK8RpL1\\_s-ykeB\\_&index=12](https://www.youtube.com/watch?v=1k-NwLN4uMQ&list=PLKRUGJaQJkAfiS7EXUfK8RpL1_s-ykeB_&index=12)

group – musically. Proposing a musical gesture but not perceiving any response to it produces, at first, an experience of outsidersness in a coercive way. The *Kuehlspot* session with a group of five improvisers had many such moments.<sup>84</sup> This is perhaps no different to the experience of non-laptop improvisers struggling for a dialogic interaction, however in the case of performing with a programmable machine this proposes a change not only in interaction but also for programming as well (i.e. to change what the instrument is). The multi-modal embodied apparatus interface (a touchpad and a fader box, together with the computer) proved to be valuable in such moments for it offered other ways or sounds to interact with ready at hand. These made it possible to flexibly react to the ongoing interaction and shift my own perspective of being outside into a more immersive experience.

### 5.2.2 Musical Structure vs. Algorithmic Freedom: *Technospaetzle*

In the *Technospaetzle* project, contrary to the rest of my practice, a much more coherent aesthetic idiom framed the musical development. The collaboration was started on the basis of the idea 'to create repetitive music to dance to' (Orr, 2014: personal communication). This already implied constraints on sonic material, their organization as well as social organization of tasks (who plays which material?). The distinctiveness of this case for my practice lies in how it revealed the importance of the groove for attaining states of immersion/flow. The omnipresent 4/4-kick drum beat served to 'align' all other sonic elements rhythmically, providing them with a kind of contextual 'bed' in which it was easier to get immersed in. But even beyond this, the rhythmic 'grid' worked as a temporal reference for all performative actions as well, this way exposing differences in body posture and performance of the two performers. While my partner had organised his apparatus setup with a stronger tactile focus, I was focused more towards the screen when live coding patterns which conveyed a different posture from an outside perspective.<sup>85</sup>

I experienced a flow state most frequently during rehearsals, when the playing was not constrained by a self-imposed time frame. Any actions during such sessions

---

<sup>84</sup>This can be viewed here, a section in which there is some interaction but new elements are introduced rapidly: [https://www.youtube.com/watch?v=BPjN9OjDMhk&list=PLKRUGJaQJkAfIS7EXUfK8RpL1\\_s-ykeB\\_&index=25](https://www.youtube.com/watch?v=BPjN9OjDMhk&list=PLKRUGJaQJkAfIS7EXUfK8RpL1_s-ykeB_&index=25)

<sup>85</sup>This analysis is based on observations from reviewing video recordings from rehearsals and performance.

could be executed 'in their own time', according to my subjective perception and readiness. This changed half way through the rehearsal process when the decision was made to create a more rigid and fixed structure toward a more finalised 'position'. Our decision to create more structure not only goes back to what our expectations of a performance were (to show more finished pieces, not exploratory sessions), but it also might be seen as a reaction to the experiences of dissonant outsideness we had in the early sessions. Being lost and disoriented by technological agency (errors and incomprehensible unintended apparatus behaviour) led us into states of worry or even anxiety towards the confidence to act and perform.

For me, the challenges to interact with the computer through live coding in parallel to tactile interaction with HID<sup>86</sup> interfaces, while following a script with loosely set cues for action, were, more often than not, too complex to experience flow. The most successful moments were those when I was 'riding the sound' directly (Butler, 2014: 220). The bass sound was playing on its own while I could directly manipulate the timbral quality with my hands on the MIDI controller to create slow development of the sound on top of the rhythm.<sup>87</sup> Such interaction enabled a feeling of immersion within the music and was supported by a tactile and embodied interaction with the apparatus much more than by the live coding interaction.

Concluding, similar to the experiences in free sessions, live coding interaction with the apparatus exposed a potential to produce an unproductive distance from the dialogic interaction with other performers because of its asynchrony or delay in relation to the ongoing musical actions. The *SUNPYX* project as well as my session experience with other laptop performers may account for this: the group interactions here were much less directly responding to each other, everyone was playing (programming) in his or her own pulse. Through this laminal improvisation approach, I was, in fact, able to be immersed in the musical process often, only with the main focus on the apparatus interaction and not towards the group.

---

<sup>86</sup>HID: Human Interface device, commonly a joystick, gamepad and similar

<sup>87</sup>Examples of this 'riding' of the sound are a) <https://www.youtube.com/watch?v=7AEmoeBxqWg> and b) <https://www.youtube.com/watch?v=U5FOaRtVmEg>

### 5.2.3 Overpowering Complexity: *MindYourOwnBusiness*

The realisation of *MindYourOwnBusiness* provides an interesting case to analyse. In all of its performances it was very rarely possible for me to experience moments of immersion. As reasons for this I identify the complexity of the interdependent system together with the temporal organisation of the piece as well as the unexpected challenges in the physical interface (keyboards). First, the interdependency of the setup posed a challenge because it put an obstacle into the live coding interaction. In other words, the performers were challenged because their habitual live coding interaction would not produce results as expected. It was necessary to adapt or in dialogic terms listen attentively and with a presence to the forms of interaction the apparatus/system actually offered. As a group it seems we did not collaborate successfully enough to find new ways for effectively functioning in the given structure. Second, this challenge was aggravated by choosing to impose a rigid time structure for the piece (to play a maximum of 15 minutes and rotate every 3-4 minutes). The short time span every performer was left with for playing in one 'role' made it difficult to develop a clear musical idea which could be responded to by the others. Each role change interrupted one's ongoing activity of thinking as well as 'functioning'/programming. The pressure was high to quickly generate results which led to lack of confidence in the actions and more errors in the improvisatory actions (i.e. in typing). Third, differences in the immediate interface (languages of the computer keyboard) added to these challenges. Reassessing the experience of MYOB offered a good example of how the algorithmic structure may inhibit dialogue between the performers – even if they all interact in the same 'language' (of live coding). In other words: the disruptive challenges in the interactional flow were so high that it was required to adapt one's own expectations of it.<sup>88</sup> Thus, the piece also achieved a transformation of selves through a challenging and negative experience.

### 5.2.4 Participation as Autonomous Agency: *Projectionist Orchestra*

*The Projectionist Orchestra* follows a very different approach and thus represents a special case in my practice. The system was to a large extent programmed prior to performance and then interacted with by the audience and myself. My main focus was on the new performance situation I found myself in: namely to perform

---

<sup>88</sup>In fact, in the later performances of MYOB I did experience immersive moments in the interaction even though the aesthetic of the piece did not support this. I attribute this to changes in my expectation and a different way of playing (more pointillist and effects-oriented).

with a composed system and many agents which I could not control directly. In this piece my experience from the 'outside' position as a conductor of the events offered insightful moments of dialogic interaction. Particularly I refer to the interaction between my written chat and the sonic responses by the participants. The asymmetry<sup>89</sup> in this interaction produced a certain kind of flow which was facilitated by the difference in technological media. I was able to influence the course of performance through my written instructions, but I could not control what the participants were doing. This provoked a playful exploration of this dialogic space created between myself and the others in order to see how utterances would be understood and responded to.

Based on informal conversations with the participants and my own experience as a listener, the visual interface of the apparatus posed some challenges. While the local touch-interface on each device afforded simple gestures with a direct, one-to-one mapping, the second section of the piece which connected these gestures to a complex probabilistic mapping was experienced as disruptive and challenging. The visual and sonic feedback from the sounds directly could not fully communicate these functionalities in the apparatus and certainly not how they changed. In this sense I was not able to provide optimal possibilities to the participating performers through which they overcome the disorientation created from the interfaces. On the contrary, though, there were always a few participants which would play enthusiastically and until the very end – suggesting that they felt very involved.

#### 5.2.5 *SUM*: Acting-Performing Against the Apparatus

The *SUM* project followed yet another algorithmic design approach taking a further step away from live coding (programming) interaction toward dialogic acting-performing and functioning. In the technological configuration on stage the laptop with its familiar keyboard as well as other controllers had disappeared (now hidden) to make space for a new touch interface. The surprising discovery as a live coding practitioner was that it seemed to be exactly this change which enabled some moments of qualitative dialogue. As I outlined in the last chapter, this may be ascribed to a surplus of conscious attention during performance (because no live programming needed to be performed). An immersion into the music seemed

---

<sup>89</sup>By 'asymmetry' I refer to the different media used for the communication: While I was typing text in the English language, the others were producing musical gestures with their apparatuses.

to be facilitated by the fact that the music was generated automatically yet in not entirely predictable ways from the synthesizer (providing both a groove and interruptions to it). Further an immersion in the thinking-acting process was supported by the imposed structure of the performing interaction (to strictly take turns). Similar to the experience of playing game, I was already consciously 'preparing' my next move, pondering which cubic pieces I would add, re-position or take away. This thinking was only possible because I did not need to continuously act. The combination of these two 'rhythms' – in the thinking and the emerging music – would help to sustain this immersion.<sup>90</sup> From another angle this performance might also be regarded as a collaborative effort to play the apparatus, together in the pursuit of engaging and novel musical structures. This view offers an example of what Flusser called a 'plus-sum game':

Both opponents ally themselves against the problem: polemic becomes dialogue. (...) Their strategy is now to compute the bits of information distributed in the unexpected situation to new levels. And they are inspired. (...) Both players have gained new information (Flusser, 2011: 100)

Both of us performers were fascinated by the challenging situation which the apparatus provided us with. To create an interesting musical result it was necessary to explore the *possibilities* concealed in the interface of the apparatus in a collaborative manner through *playing the game*. As noted the missing programmer's-perspective greatly helped to enable such a way to perform: It opened the door for being surprised in performance. This might imply that it may be worthwhile to reconsider how the process of programming preparation takes place. In the case of SUM the apparatus was built in a collaborative effort<sup>91</sup> with the result that both performers did not know about the work of the other. Embracing this fact as a contingent result of the technologies involved helped us to develop qualitative dialogic relations to the apparatus as well as between ourselves.

---

<sup>90</sup>An example moment from performance was this here: [https://www.youtube.com/watch?v=503OZAn-BH4&list=PLKRUGJaqJkAfIS7EXUfk8RpL1\\_s-ykeB\\_&index=22](https://www.youtube.com/watch?v=503OZAn-BH4&list=PLKRUGJaqJkAfIS7EXUfk8RpL1_s-ykeB_&index=22) [accessed 02.07.16]

<sup>91</sup>My partner had created a 'patch' on the modular synthesizer which defines the sound palette while I had realised the mapping for the interface, which defines interactions on the interface relate to the sounds.

### 5.2.6 Summary of Immersion/Challenges in Dialogic Coding Practice

Project	Dialogic Layers	Translations	Degree of Nontriviality	Preparedness	Performativity	Ethics / responsibilities
	<i>in order of interactions</i>	<i>from language → language</i>	<i>Of algorithmic agency, From 0 (low) to 10 (high)</i>	<i>Of system/code, from 0 to 10</i>	<i>Obscure ↔ obvious Non-acted ↔ acted</i>	<i>Performer / apparatus freedom (constrained, some, autonomous)</i>
A1 'Drums'n'Algorhythms'	1. meso 2. micro	Sound → Code	4 (medium to low)	1 (low)	Visually: obscure Sonically: transparent Embodied: not-acted	Perf.: autonomous Apprt.: some
A2 Free Sessions	1. meso 2. micro 3. macro	Sound → Code Visual → Sound	3 (low to medium)	3 (some)	Visually: some Sonically: complex Embodied: acted (gestural)	Perf.: autonomous Apprt.: some
A3 SUNPYX group	1.meso 2.macro 3.micro	Action → Sound	5 (medium)	7 (high)	Visually: some Sonically: complex Embodied: acted	Perf.: some Apprt.: some
B Technospaetzle	1.micro & meso 2. macro	Sound → Action Sound → Code	4 (medium to low)	9 (very high)	Visually: obscure Sonically: complex Embodied: not-acted	Perf.: some Apprt.: constrained
C1 MYOB	1.micro (=meso) 2.macro	Sound → Action → Code (+reverse)	7 (high to medium)	6 (some to high)	Visually: obscure Sonically: complex Embodied: not-acted	Perf.: constrained Apprt.: autonomous
C2 Projectionist Orchestra	1.meso(=macro) 2.micro	Sound → text (chat) Code → Action	4..8 (medium + high)	3 (some to low)	Visually: obvious Sonically: complex Embodied: not-acted	Perf.: constrained Apprt.: constrained / autonomous
C3 SUM	1.meso(=micro) & macro	Sound → Action (+reverse)	6 (medium to high)	6 (some to high)	Visually: obscure Sonically: complex Embodied: acted	Perf.: constrained Apprt.: some

Figure 10: Summary of Dialogic Qualities in Selected Cases

In Fig. 10 I have provided an overview of the examples from Dialogic Coding practice featured in this writing. This may help to draw a comparative image of their differences and similarities. One recurring aspect throughout the practice was the contingency of the immersive experience as a programmer-performer – across all different performance situations. My own responses to this are built into the successive developments of my 'performance ecology' – the different configurations of technology with the computer at the core. From this viewpoint the trivial conclusion is that every partner in a dialogue actually requires a different configuration related to the individual's skill sets and way of playing which is then the foundation for being immersed in the improvised interaction. Of particular tension in this respect was the concealed, personally-focused or distanced performance character of the programming activity<sup>92</sup> in relation to the improvising group. This kind of performance attitude produced less perceived outsidersness in a scenario of other laptop performers (which were similarly in need of focusing on the computer screen). A dialogic interaction within the emerging musical dialogue

<sup>92</sup>This refers to the typical performance setup in live coding practice in which the performer is sitting behind the laptop, focusing solely on the computer interacting through the keyboard interface and communicating very little directly to others.

was nevertheless possible, perhaps, with the restraint of delayed responses through the coding 'translations' and typing. Such a delay in the response and more generally a lack of human perception and responsiveness in the apparatus (i.e. to adapt to a new musical direction in the improvisation) produced a dissonant kind of playing at times which in turn resulted in an adaptation of the human improvisers to the 'less resonant' playing of the nonhuman agency. This points to the aspect of time or the 'interactional synchrony' (Sawyer, 2003) in the actions which plays a central role for the creation of dialogic quality in improvised interaction. The immersion experienced in programming happens on a linguistic level, that is in the interaction with the algorithms on a symbolic or syntactic level. Within such a 'programming flow' the performer may entirely lose a sense of physical space and time.

Perhaps a programmer is thinking in algorithmic time, attending to control flow as it replays over and over in their imagination, and not to the world around them. Or perhaps they are not attending to the passage of time at all, thinking entirely of declarative abstract logic, in a timeless state of building. In either case, it would seem that the human is entering time relationships of their software, rather than the opposite, anthropocentric direction of software entering human time (McLean and Wiggins, 2010: no page number).

The 'algorithmic time' McLean and Wiggins describe here is that which is often asynchronous to the pulse or time of the performative interactions with the improvising group. Thus in one situation an algorithmic performer may be immersed in the interactions with the apparatus in the 'algorithmic time-space' and consequently be less present in the physical time-space of performance. In another situation it may be exactly opposite: the performance does not allow her to enter that time-space because it requires too much presence. In such a situation the algorithmic performer may intentionally adopt a strong visual and cognitive focus towards the computer screen in order not to be overly distracted by other stimuli and 'taken out' of the 'algorithmic time-space'. On the other hand, it is also the responsibility of the algorithmic performer to enable dialogue with the group. Therefore it may be necessary to *bend* the algorithmic structures towards the rest of the improvising group. In other words: to better synchronize the algorithmic and the physical time-space. The performer performs as an intermediary between apparatus and the improvising group.

In the different scenarios above I discovered for myself that the role of programmed indeterminacy is the central parameter with which this mediation can

be realised. Randomness is a common and important parameter in interactive computer music. In the *Voyager* system Lewis (2000) uses randomness to control various musical parameters such as melody, harmony, orchestration, ornamentation, pacing, transposition, rhythm, and even internal behaviour decisions in order to 'avoid the kind of uniformity where the same kind of input routinely leads to the same result' (Lewis, 2007: online).

The use of white noise allows me to filter the resulting stream of numbers in a conceptually simple and orthogonal fashion, where "tuning" the program for a desired musical result, or setting up input response parameters for interaction with improvisers, ultimately becomes a process of setting percentages. This manipulation of the many program variables whose values are determined by various filterings of white noise results in the emergence of complex global personality (Lewis, 1996: 106).

In my Dialogic Coding practice I use white noise in a similar way. By changing the amount of noise I can continuously adjust the degree of nontriviality of the apparatus I perform with. Sometimes the system is programmed in a way so it allows for flexible re-programming (the ideal case), sometimes it is actually 'black boxed' and allows for interaction only within the given constraints. In the latter case this may be expressed in that the apparatus may only be *influenced* not controlled. Put in a simplified way, the more improvisers participate in a situation, the more trivial functionality of the apparatus is necessary to be able to create a dialogic interaction with these players. This conclusion may be familiar for experienced improvisers in larger ensembles, too. A larger group size exposes a tendency to constrain the individual freedom of each participant to make a meaningful interaction between *all* participants possible. In other words: this situation apparently affords more structure, i.e. in the form of a score which organises when or what to play or in the form of a rule that everyone plays less in order to leave space for anyone else to play. Wilson and MacDonald (2015: 11) revealed that:

improvisers within an ensemble do not always perceive themselves as having agency in the direction of the improvisation. Their evaluative processes were shaped by the social context and the tastes and identities they constructed for other members of the group.

This raises the question for future research what the taste and identity might be that members of the improvising group construct *for* the apparatus (entangled, but not identical with the algorithmic performer).

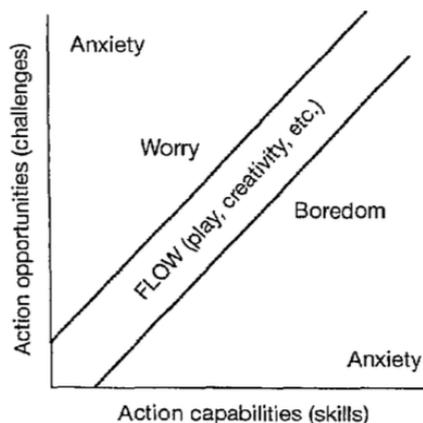


Figure 11: Structural Model of Flow

agency (control) towards the apparatus. This works similar to how Csikszentmihalyi (2014) has described the conditions which are necessary for a flow state to be attained. In his structural model (Fig.11) he describes a 'channel of flow', a space in which the skills ('action capabilities') and challenges ('action opportunities') of an agent match.

Csikszentmihalyi (2014: 159) emphasises, however, that this model is relative to the agent's own perception of what she considers to be the challenges and skills in a situation. I may adapt this model to this research context: If the challenges of apparatus interaction (against algorithmic agency) as well as the group interaction (responding to the collective improvisation) feel beyond the skills of the algorithmic performer, she may fall in a state of worry and in succession anxiety. And on the other hand, if the (programming) skills are greater than the opportunities for using them (i.e. the situation does not allow for reprogramming), boredom will follow. The conclusion I draw from this is that the indeterminate agency of the apparatus ideally roams within a small corridor between the extremes of predictable and controllable (low challenge) and unpredictable and uncontrollable behaviour (high challenge). In such a scenario the algorithmic performer may successfully be enabled to empowering experiences in the dialogic interactions of all levels, the audience, the improvising group as well as the apparatus. The productive challenges as well as the immersion in the activity and the music may be produced by all participants. While the indeterminate apparatus may challenge the functionary-performance in the interactional flow or the programmer-performance in the intellectual flow, the audience interaction may challenge the actor-performance as well as the improvising group interaction challenges the listener-

performance by unexpected responses or unanswered calls. Any of these impulses may inspire a moment of reflection and a new direction for how to re-programme the agency of the apparatus in order to contribute something else to the ongoing improvisation.

### **5.3. Virtuosity in Algorithmic Performance**

From this analysis of dialogic quality through immersion and outsidersness I may draw conclusions towards the aspect of performer virtuosity. *Virtuosity* points to *virtue* which can, etymologically be understood as a great skill in an artistic pursuit. But it also carries the meaning of an *ethical* conduct of virtue in that activity. Cox and McLean (2013: 59) translates this to 'not just doing something, but doing it "well", the act becomes an end in itself.' Considering virtuosity in dialogic coding may help to reveal how the skilled performance of playing with autonomies of performer and apparatus may be able to create transformative experiences. In the following paragraphs I have compiled a list of essential skills for algorithmic performance based on my practitioner experience.

#### **5.3.1 Essential Skills**

##### **5.3.1.1 Presence**

In line with dialogic theory, presentness and attention towards the situation on all levels of performance is a general requirement. This may seem trivial, but it is necessary to bring this to mind for the 'algorithmic time-space' and the actual physical performance situation often diverge in the kind of presence they afford the algorithmic improviser.

Furthermore, the aspect of presence is an essential condition for what has been termed the skills of *kairos* and *metis* within improvised performance. Again, it is easy for the programming-performer to get lost in the virtual apparatus world and lose sight of the fact that she still needs to play and interact in an embodied way. Dialogic Coding practice has revealed that a positive, immersive and transformative experience is often facilitated if more of the body is involved in the multiple simultaneous activities. Therefore, presence in algorithmic performance carries the responsibility to bring the cognition-heavy activity of programming into a coexistent balance with a more embodied type of physical interacting as well as a listening presence towards the 'speaking' of other dialogic partners.

### 5.3.1.2 Preparation

The ready availability of materials or in other words their *preparation* is of key importance for improvised performance to achieve flexibility and responsiveness in the improvised interactions. For the algorithmic performer this may have the form of preparing a collection of sounds (recorded samples or recipes for synthesis) which covers a wide range of sonic timbres. And additionally a collection of musical phrases for how to play such sounds<sup>93</sup>. Blackwell and Collins (2005: 8) assert that:

in the context of live programming, existing bases of code by the performer and by others are an extremely valuable resource. Fragments of code are assembled like jazz licks or scratch samples.

This implies that an algorithmic improviser is also only able to play what she 'rehearsed' or prepared before performance. While this holds true in principle, the virtuosic apparatus interaction may provide further options of how to exploit the capabilities of the artificial apparatus memory to create novel and surprising results – and thus not be constrained by one's own imagination. The comparison to jazz licks provides another dimension of preparation: the physical performance. Similar to how acoustic instrumental performers rehearse in order to train their 'muscle memory' for playing particular musical phrases, the algorithmic performer may train her fingers to minimize typing errors and maximise speed in order to program faster. Additionally, the power of associative thinking for the linguistic activities of translating sound ideas into code needs to be nurtured. To train such thinking skills it helps to learn from the performance practice of other coders as well as share and discuss code with others in order to find new ways for combining code objects and constructing code 'phrases'. In my own practice several online resources and mailing lists<sup>94</sup> have provided helpful inspiration and starting points for my own code-sound-archive as well as the live performance practice.

### 5.3.1.3 Virtuosic Bricolage Programming

The term 'bricolage programming' was already introduced in section 1.6.4. The abilities of the 'bricoleur' describe well the particular mindset an algorithmic

<sup>93</sup>This may be in the forms of melodic or rhythmic phrases, harmonic progressions, sounds in every register (high, mid, low pitched), frequency-limiting filters, atonal sounds, pointillist or sustained sounds and so forth.

<sup>94</sup>Amongst these are the SuperCollider mailing lists (<http://supercollider.github.io/community/mailling-lists>), the SCCode forum (<http://sccode.org/>) as well as SC code found on twitter.org (#140), (<https://twitter.com/search?q=%23sc140&src=typd>). Furthermore, I was able to learn from powerbooks\_unplugged's code (<http://wertlos.org/pbup/>).

performer may perform with. This in some respect is no different to any other form of improvisation: this model, however, specifically suits the programming perspective which thinks and articulates in an algorithmic, a 'planned', form. Turkle and Papert provide a description of this approach in the context of computer programming:

The bricoleur resembles the painter who stands back between brushstrokes, looks at the canvas, and only after this contemplation, decides what to do next. Bricoleurs use a mastery of associations and interactions. For planners, mistakes are missteps; bricoleurs use a navigation of mid-course corrections. For planners, a program is an instrument for premeditated control; bricoleurs have goals but set out to realize them in the spirit of a collaborative venture with the machine. For planners, getting a program to work is like "saying one's piece"; for bricoleurs, it is more like a conversation than a monologue (Turkle and Papert, 1990: 136).

The key for virtuosic dialogic code performance is the unplanned '*navigation of mid-course corrections*' which enables for reflexivity and change of directions. This implies that a virtuosity in programming does not only mean a fast and efficient typing or a fluent 'thinking in code' but rather that it integrates the goals of openness and reprogrammability in order to realise such flexibility. The creation of such 'access points' to the algorithmic agency makes it further possible to readily respond to the intuitive impulses from a kairotic perception within the improvisation. For the programmer-performer a moment for action may mean to take action towards the apparatus (i.e. by changing its behaviours) and not literally or directly to respond to any 'call' from the improvising group.

#### 5.3.1.4 Fluent Translation, Programming Efficiency

The ability to translate requires a good knowledge of the linguistic systems in play and a particular type of perception: the 'calculating and computing consciousness' (see section 4.2.2). It is helpful for the virtuosic algorithmic performer to rehearse these linguistic and perceptual abilities (as mentioned above) through practical exploration (see my piece *MIMICRY*, Fig.9).

Furthermore the integration of the automation capabilities of the apparatus is feasible. That means for example, to program with 'macros' – little text editing functions which propose automated completion for started phrases in order to speed up the sound production process. This automation of the programming activity is facilitated by specific software tools such as a powerful, configurable text editor (i.e. emacs, Atom). It starts, however, with the universally available functions of 'copy' and 'paste' which allow for quick multiplication of code words or numbers.

The dynamics of live improvised performance afford quickness and immediacy in actions. Therefore shorter linguistic expressions in the programming language may be preferred. Some practitioners have consequentially programmed their own idiosyncratic live coding languages.<sup>95</sup> Such systems are based on the ability in programming languages to abstract more complex behaviour into new functions which may be summoned by a single key press. In this sense abstraction and automation should be exploited for virtuosity for they allow the performer to create complex results with few actions.

A further aspect from the linguistic, syntactic possibilities in a programming language such as SuperCollider is the possibility of modularity and recombination. A code object may be used in different contexts for different purposes (i.e. an oscillator may be used to generate audio or control another signal). Also code objects or sound processes may be recursively combined ('nested') to create complex, self-influencing behaviours in the code. This may not only create surprising results but it also saves time to type new code words.

Lastly, similar to any other inter-language performance, good translational skills require good resources. There are, of course, also 'dictionaries' for the programming language and other look-up resources which demonstrate the use of code words. In my live practice I have frequently relied on code examples provided by the integrated 'help' reference system in the SuperCollider language. Further in studio sessions the other resources for sharing code mentioned above as well as available tutorials such as the SuperCollider book (Wilson et al., 2011) have supported my programming virtuosity. Such resources demonstrate how specific code objects may sound, which helps the ability to 'think' in the 'calculating and computing consciousness' mentioned above.

#### 5.3.1.5 Empathic Imagination

The last essential skill I propose here is an ability to empathise with the particular performance situation for which a performer prepares. To be able to imagine what the interactional, musical and social requirements of that situation are helps to choose the suitable algorithmic structures in the process of programming apparatus agency. This may then facilitate immersion into the improvisation through attending the creative musical processes more than the technical processes of problem solving (see my experience of the *Drums'n'Algorhythms*

<sup>95</sup>See Daniel Mayer's software ([http://daniel-mayer.at/software\\_en.htm](http://daniel-mayer.at/software_en.htm)), Thor Magnusson's Ixi Lang (<http://www.ixi-audio.net/ixilang/>) amongst others.

session). However, this imagination must be combined with the effort to balance indeterminacy and predictability in order to propose novel directions for the particular situation (see section 4.6).

### 5.3.2 Creative Strategies For Change

Besides these more obvious skills, I want to propose a set of different, more *dissonant* strategies for thinking and performing which have helped me to transform creatively 'jammed' situations into satisfactory results. The benefit of entering with a computing (binary) apparatus into the dynamic (organic) process of improvised performance lies in the contrasts and challenges which such a misfit automatically produces. To use the technology in ways that challenge one's thinking by pointing to the fact how software is not automatic but written as well as used by humans, is an approach found in software art. Artist-programmer Amy Alexander refers to this when she states that software art is interested in the production of 'programming accidents.' A computer will not produce an 'error' in itself; it is only through intentional use that a surprising result may be discovered, this is the 'intentionality of editing' (Alexander, 2009: 123). To develop such intentionality for laptop performance I want to propose a set of dialogue-oriented thinking skills – an *epistemology of self*. Lewis (2013) claims that dialogic interaction with oneself requires four essential 'socio-cognitive skills': 1) decision-making; 2) divergent thinking and embracing ambiguity; 3) risk-taking and problem-seeking; and 4) accountability and self-assessment. The successful application of these skills is based on the ability to attentively listen in the modality of self-awareness towards the inner and any of the outward directed modalities mentioned in section 4.2.1. This listening makes the dialogic interaction possible. Lewis' model seems particularly suited because the interaction between the algorithmic performer and the apparatus shares many aspects, in particular the fact that ultimately the human performer is the author of any actions in this dialogue (see 4.5.2) – even if they are performed by the apparatus (which she programmed before). I may now articulate these four socio-cognitive skills adapted to the context of algorithmic performance:

#### 5.3.2.1 Decision-Making

While Lewis mainly refers to what is understood in the terms of *kairos/metis*, or the ability to know when to act and do it, I want to extend this view of an ability to decide towards the perspective of programming. Deciding as a programmer-

performer takes place under the influence of multiple responsibilities, as we have seen in the previous chapters: for herself and her own actions but at the same time for the actions of the apparatus. Furthermore, there are also the performer's other responsibilities of the functionary- and the actor-performer. In this context the skill to decide might thus not only imply to decide for oneself and one's other inanimate self but perhaps also to decide 'not to decide' – and delegate this decision-making authority to the algorithmic autonomy. The benefit of dialogic coding performance is the actual possibility to decide on how get's decided, to be able to design the decision-making process. In my practice I understand this skill as a reminder to consider the decision-making process (a necessity for the programmer), but to do this with the intention of apparatus freedom while retaining the performer's own freedom (to change the program).

#### 5.3.2.2 Divergent Thinking and Embracing Ambiguity

To embrace ambiguity when using a programming language is a nontrivial task as I have shown before. When looking from a situated viewpoint, however, ambiguity might be understood as a performance strategy for the human agent. Following the intention of divergent thinking the aim may be to overcome a performer's own limited imagination and habitual ways of performing.

For the programmer this may for example suggest to use code objects in the programming process which one has not used before or does not know the results of. Or to use known code objects in combinations which are new and untested in order that something new and unexpected results. A rich programming language provides an infinite number of possibilities for this approach. I term this an approach of 'experimental programming'. This approach may also be pursued from the reverse perspective through a kind of 'naïve approach', in other words: to programme as if one did not know. Or in the words of Amy Alexander (2009: 123): 'when I write software as art, I understand everything and nothing about what I'm doing'. The programmer may also intentionally create ambiguity and surprise through the possibilities of abstraction. Such abstractions would provide productive challenges, they are not-easy to use or produce unexpected results.

This approach may even extend to the interface of the apparatus. A controller can be remapped to control the sound in various ways dynamically responding to the situation. The live programmability of the mapping could as well be exploited to play with the degree of indeterminacy of the algorithmic agency. De Campo (2014) presented a solution in form of a probability based mapping rather than direct

linear (causal) connections, termed the 'Influx matrix'.<sup>96</sup> This system follows the idea of delegating *control* to the algorithmic system in order to gain *influence*. Because of its real-time control possibilities a performer can directly *play* the nontriviality of a system instead of its actual responses. This makes it ideally suited as a flexible tool for a dialogical performance setup. In my practice I have used this system in the PO and the SUM project.

#### 5.3.2.3 Risk-Taking and Problem-Seeking

Risk-taking in improvisation is understood much in a similar way as I have outlined the creative strategy of using untested code words above. Problem-seeking follows a willingness to discover new things and behaviours through performance by challenging oneself in the habitual ways of playing and thinking. For the algorithmic performer this happens in all the roles of listening, programming, functioning and acting. I have suggested some ways of how to perform differently and more oriented towards taking risks in favour of dialogic interaction and self-transformation in section 4.5 in the last chapter. Problem-seeking further hints at the basic dissonances mentioned above of the binary and unresponsive algorithmic structure in a situation of highly resonant and fluid improvised performing. In other words: what I have perceived as 'problems' in my practice may as well be used towards self-transformation in a future approach. In my methodology I already started a strategy of problem-seeking by inserting myself with the apparatus in the situations described. The key aspect in this particular skill set is the mindset of the performer which enables her to perceive a disruption or challenge as an opportunity: for playing with and elaborating on *difference* (musically and socially) as well as for learning and her own identity.

#### 5.3.2.4 Accountability and Self-Assessment

This skill lies at the foundation of dialogue. It outlines the ethical dimensions of dialogic interaction. The 'inherent and comprehensive indeterminacy' of improvisation calls upon the improviser to take ultimate responsibility for the outcome (Kanellopoulos, 2011: 119). The accountability and responsibility of the algorithmic performer is actually four-fold (see section 4.5) and it includes the apparatus with whom she at the same time interacts as her dialogic other.

I will continue and extend this last dimension of the socio-cognitive skills of the algorithmic improviser in the conclusion to this chapter.

---

<sup>96</sup><http://www.3dmin.org/research/open-development-and-design/influx/>

## 5.4. Conclusion: Responsibilities for Change

Concluding this list of skills of the virtuosic algorithmic performer I want to return to the notion of negotiation and balance. I have outlined in the previous chapter how the performer balances her different perspectives and responsibilities. Expanding on this I propose now how this negotiation is actually one between the experiences of immersion and outsidersness. As I demonstrated earlier the immersive experience precedes that of stepping outside but with minimal distance between them in a movement of oscillation between the states. The performance situation of the improvising group puts much pressure toward interactional synchrony which stands in contrast to the 'algorithmic time' in which the apparatus operates. As another suggestion for approaching this conflicting opposition I offer this final strategy into the direction of a creative solution.

### 5.4.1 Slow Programming or to Embrace Asynchrony

This strategy is dedicated to the challenge of interactional synchrony in performance. As I described above, the performance situation of free group improvisation require a certain readiness in action and an eagerness to participate and respond to others. This puts a pressure on the activities of the algorithmic performer, particularly for live programming. In my practice I have frequently experienced typing errors and interruption in the creative thinking process as a *consequence* of this pressure. Such pressure has not been helpful for the creative musical process: it inhibits well-considered thinking and suggests the use of simple phrases or premature inventions. But the fact of 'being too late' is actually built into the hierarchy in live programming as Julian Rohrhuber concludes:

because programs are plans, or propositions for the future, changing such a plan in its unfolding cannot be immediate and transparent at the same time. From the beginning, Just-in-time-programming<sup>97</sup> has therefore been a tongue in cheek cover term for the truth: a program is always too late. (Rohrhuber in Blackwell et al., 2014: 151).

What Rohrhuber insinuates is that 'just-in-time-programming' is in fact just-in-time *enough* for it to be *perceived* as synchronous action.<sup>98</sup> However for a creative

<sup>97</sup>A synonymous term for live coding.

<sup>98</sup>In praxis, computer music applications can deceive the human listener's perception into experiencing an immediate gesture-sound (kinetic-sonic) connection. This is the aspect of *latency* in audio software— the time that passes between a (kinetic, energetic) control event and its resulting (sonic) output caused by the processing time of the algorithms and the computer hardware. If it is significantly small then the action and its result will be perceived as synchronous.

strategy I extrapolate from this that a performer may also turn things around and embrace the inherent asynchrony to *actually* perform 'out-of-time'. This requires the performer to place herself outside the ongoing 'synchronous' improvisatory activity and inhabit a position which allows for the articulation of more nuanced utterances, from a distance so to say. The computational architecture in this sense, supports and, one may say, even invites a slower and asynchronous interaction which takes time to attentively listen before responding. At best it may also result into more complex and intricate musical gestures.

Such a kind of 'slow programming' in fact starts from the affordances of the physical interface of the computer keyboard. Exactly because the typing is an embodied activity the performer is able to think simultaneously to it. Within the timeframe it takes to type a particular code phrase the algorithmic performer is free to reflect on the invented phrase – whether it was really a good idea or not or whether it should be changed into something different. In other words: the delay introduced by the embodied typing may already provide a moment for reflection to silently examine and imagine the sonic result of the code phrase which is being written and then possibly refine it in some way almost 'on-the-fly'.<sup>99</sup> I have experienced such moments of 'just-in-time re-thinking' frequently in my practice.<sup>100</sup>

The 'slow programming' strategy is furthered by the fact that the apparatus does the 'hard work' of sound generation and may do so in automatic ways. Slow programming thus enables the algorithmic performer to more attentively listen to the emerging music and perform a more considered programming which enables the apparatus to play autonomously and in interesting ways, possibly interacting with the improvising group in a dialogue. This strategy will produce a productive form of outsidersness which may be changed at the performer's will to change the course of actions towards more satisfactory ways.

---

<sup>99</sup>It has to be said that this interaction is made possible by the non-linear function of text editing on a computer: One can always jump to any particular position in the text and change elements (characters) in the written text—different to typing a text on a typewriter.

<sup>100</sup>To illustrate this case, let me give a description: the typing of a code phrase would start with one or two code words and then be filled with the actual numbers feeding the input parameters of the sound. Halfway through the process I would change my idea – perhaps because of insecurity with the initial idea or of a change in the parallel musical activity in the group – to replace them with an adapted version of the initial idea. In the worst case, however, this silent 'struggle' to articulate a musical phrase in code, in dynamic response to what is being played by other collaborators, may result in never actually actuating a phrase because it seems 'unfit' at the moment when the act of typing has finished.

#### 5.4.2 Increase The Number of Choices

A 'slow' approach may be a creative answer to the demands of immediacy. This is based on a kind of 'intentional subjectivity'. By this I mean to exploit Dialogic Coding practice as a form of 'self-programming'. Through the activity of programming in dialogue the algorithmic performer becomes aware of her own self as well as the 'identity' of the apparatus (as perceived). These identities are the product of mutual interaction and thus cocreative. These selves may only develop freely if diversity and variety are possible, in other words: freedom. For such freedom to manifest a greater number of choices is needed. Diversity and freedom need to be created for both performer and apparatus in conscious acts of programming and through a play and interplay between the participating agencies. This is the performer's *virtuosity* and her *responsibility*: the skill to efficiently play with the provided tools and materials to create challenges for herself and the other partners in dialogue, switching between their own perspective and the imagined perspective of the apparatus identity they are about to invent. Following Flusser (2011), this is a 'creative play' with the agency of the algorithms (the apparatus), in a 'controlled game of chance' (dialogue). To paraphrase Flusser this describes the 'envisioner' which I hold equal to the algorithmic performer in Dialogic Coding practice who is deeply entangled with the technology:

[envisioners] are people who press the keys of an apparatus to make it stop at an intentionally informative situation, people determined to control the apparatus in spite of its tendency to become more and more automated and so to preserve human judgment over the machine. Envisioners are people who try to turn an automatic apparatus against its own condition of being automatic. They cannot create illusions without the automatic apparatus, for the stuff to be envisioned, the particles, are neither visible nor graspable nor comprehensible without the apparatus's keys. But they can't turn the envisioning over to the automatic apparatus either, for the technical images produced in such a way would be redundant, that is, predictable, uninformative situations from the standpoint of the apparatus's program (Flusser, 2011: 19).

He aptly summarizes this symbiotic, yet ambiguous relationship of the human with the apparatus which may also be called a 'two-getherness' (Buber, in von Foerster, 2003: 297). These two agents are an inseparable unity which can achieve impressive (and novel) artistic results but is always at the risk of, on the one hand, falling into a redundancy of results (through algorithmic determinacy) or on the other hand, into a self-alienation of the performer as a *functionary* for the apparatus (performer determinacy).

## **5.5. Summary**

In this chapter I have presented how the dialogic quality in my practice may enable reflexivity and through this produce transformations. I have related the concepts of outsidership and empathising to my practitioner experiences of immersion in the immediate activity or a stepping out of it.

I have elaborated on how the experiences are evident in my practice and concluded how these have informed the development of the insights gained through practice. Finally, I have proposed strategies for 'virtuosic' performance which works toward transformation through dialogue.

These conclusions will be summarised in the following chapter to highlight the distinctiveness of Dialogic Coding practice as a form of 'ethical' performance practice in the live computer music field.

## 6 – Conclusion and Summary

The present research has followed a PaR methodology to develop a distinctive live computer music practice in situations of group improvisation termed Dialogic Coding. The development of this practice was experimental in how it attempted to create a dialogic relationship to the nonhuman agency of the apparatus in response to the situation of performance. Dialogic Coding practice, in its various realisations, is ongoing, based on the successes and failures in the process. So as endings are beginnings, conclusions also serve as introductions. In the following, I will conclude the present cycle of research, summarize its central insights, and propose directions for the following future cycle.

### 6.1. Chamber Music

In chamber music, there is no director, no government. The one who sets the tempo is only temporarily directing things. And yet chamber music demands an exceptionally close adherence to rules. It is cybernetic. Chamber music is pure play, by and for the players, for whom listeners are superfluous and intrusive. It employs participation (strategy) rather than observation (theory). Precisely to play as though it were playing solo, each instrument plays as though it were an accompaniment. To play for himself, each player plays for all the others. Each improvises together with all the others, which is to say, each adheres to precise rules (consensus) to jointly change them in the course of the playing. Each player is both a sender and a receiver of information. His goal is to synthesize new information to become more than the playing (Flusser, 2011: 162).

Flusser uses the scenario of chamber music as a model for the 'dialogic communication' in his envisioned 'telematic society'. The quote shows what a surprisingly apt candidate dialogic coding is as a concrete realization of this scenario. It seems justified to replace the media that Flusser specifies, namely 'technical images', with the media used in my research, namely concrete (digital) sound, considering that Flusser himself sees the composing of music synonymous with the activity of calculating and computing 'technical images. Based on the idea that the human is a player, a *homo ludens*<sup>101</sup>, it is only a logical conclusion to call on every member of society to participate in this communication; a conversation with others through sound and images, facilitated by apparatuses.

What is central for the sociable dimension of dialogic coding performance in this, is the change of perspective from observation to participation. The 'purest' form of

<sup>101</sup>This term was coined by Huizinga (1980) who describes play as a behavioural strategy in our society

this artistic practice might indeed be such where there is no outside audience anymore, but only active and participating performers. This should not imply that a scenario as I have explored it through practice with other acoustic instrumental improvisers is undesirable. Rather it points to the fact that a large share of the significance and self-affirmation in the activity is created through the active, involved position of 'doing things'. This presents Dialogic Coding practice as a form of a 'DIY' activity. Do-it-yourself practices are a cultural tactics that offers alternatives to mainstream capitalist consumer culture by creating spaces of exchange (Holtzman et al., 2007). These spaces function in a dialogic way by actually getting together, that is encountering human 'others' with their idiosyncratic apparatuses to share skills, ideas or experience.

At the foundation of this possibility lies an essential condition: To be able to not only read digital media, that is, to use it in a passive consuming way, but to be able to *write* it, or in Flusser's terms to use it in an intentional way as envisioners. Douglas Rushkoff (2010: 128) describes this *digital literacy* as the challenge for the 21<sup>st</sup> century when he programmatically writes:

Digital technology is programmed. This makes it biased toward those with the capacity to write the code. In a digital age, we must learn how to make the software, or risk becoming the software. It is not too difficult or too late to learn the code behind the things we use – or at least to understand that there is code behind their interfaces. Otherwise, we are at the mercy of those who do the programming, the people paying them, or even the technology itself.

Rushkoff alerts us that such asymmetric distribution of knowledge of the means of production results in social asymmetry: an unequal distribution of wealth and power in society. The consequence of this is that the content of messages, the code, is dominated by a minority of *senders*. Flusser (2011: 83-84) describes this as 'totalitarian discursive society' which he sees as the predominant model of communication in our society today. From the centralized senders the messages are sent to the periphery of receivers, more or less uni-directionally. Since the original publication of Flusser's text in 1985, much has changed in the media technologies and how we use them in society today. There exist now many more bi-directional (potentially dialogic) connections, particularly in the social media technologies of the web. Yet in the sphere of media performance and even stronger in music performance, where this research project locates itself, traditional models of performers *sending* and audience *receiving* (or at least expecting to receive) are still predominant.

Based on my experience of developing a dialogic coding practice I propose that we perhaps rethink the ways in which such technologically mediated music is presented and experienced. The format of a *participative workshop* to share knowledge and experience may actually be much more appropriate to communicate artistic ideas in a performative format. I understand the *workshop* to be a format which incorporates spoken as well as other forms of presentation (sonic, visual) and is characterised through the possibility of receiving not only a presentation as a 'black box' to experience but also instructions how this presentation was constructed, including options to become active. At first such a format may imply a step towards exclusivity which enables only those to participate who have the skill (the practitioner knowledge) to do so. However, I wish to advocate the opposite picture: A workshop group of perhaps 10 participants and 1 or 2 instructors may be the centre, but the boundaries are permeable—other interested people may come and go as well. This has to be imagined as a scene in which few involved people will work and create 'hands-on' while an undefined number of others are standing behind them, watching curiously over their shoulders, following the discussion between instructors and participants as by-standers. In this way, a participative workshop might be able to disseminate knowledge in an inclusive and less distant way through informal peer-learning and close observation. The activity is then not only accessible for learners (as workshop participants) with a certain level of 'literacy' or technical knowledge but it may be possible for others (as by-standers) to gain an experience, too.

Certainly the possibilities for dissemination of the experience of coding have not been exhausted. The recent initiation of the live code research network<sup>102</sup> and the first academic conference on live coding, the first performance focused festival for live coding<sup>103</sup> or the growth of algorave concerts<sup>104</sup> bear witness to this developing potential.

---

<sup>102</sup><http://www.livecodenetwork.org/>

<sup>103</sup><http://imwi.hfm.eu/livecode/2013/2013.html>

<sup>104</sup><http://algorave.com/>

## 6.2. Improvising Artistic Technoscience

As a second conclusion this practice may have insights to offer outside the sphere of computer music performance. The bringing to bear of the technologies of programmable machine culture on the practices of improvisation (its heritage and influence) carries a substantial potential.

The exploration of spontaneous, explorative, and surprise-seeking interaction with code has provided me with new perspectives and skills for *performing* technology. These insights may be applied not only in the context of the performing arts, but also in the everyday of our technologised society. As this research project suggests, users of technology may rethink their intentions and relations with the ubiquitous computing devices and perhaps 'play' (or even program) them differently. The first step towards this is to seek and create possibilities for such play in a sociable situation of creative collaboration. Ultimately, Dialogic Coding practice reminds us of the possibilities in new technologies to encounter 'the other' – the technological other, myself-as-other, as well as my human collaborators in a mediated way in dialogic spaces. From this viewpoint the aim of this research project has been to bring closer together the different cultural worlds it draws from – digital and programming culture, the free improvised music scene, acousmatic and experimental music – in the form of a unified practice. Improviser G.E. Lewis (2007: online) describes the importance of such an interdisciplinary approach from his perspective:

The direct study of improvisation will be vital to the production of new ways of using information technology - not only in the arts, but also across the board. Human identity, particularly in negotiation with new technologies, is continually reinscribed through processes of interactivity and improvisation, thereby demonstrating its centrality to our birthright as human beings. Thus, the interdisciplinary study of how meaning is exchanged in real-time interaction will be crucial to the development of new user interfaces, new forms of art, more sophisticated interactive computer applications, and much more (...).

Code as a medium does apply not only to the field of performance art and music but it is relevant in many other sectors of society built upon software technology as well. Therefore this approach also has a distinct political<sup>105</sup> dimension proposing a way in which a digitalised society may be reorganised. Instead of accepting the

---

<sup>105</sup>'political' is here understood to refer to the process of making decisions applying to all members of society. If the nonhuman agents are included in this process, then Dialogic Coding taken as a proposal for strategic and creative use of these artificial subjects, may contribute valuable ideas towards a dialogically organised society.

apparatus as more intelligent models of ourselves the nonhuman agents should be invented and interacted with as our 'other' selves – dialogically and in playful ways. Flusser (2011: 85) calls this a 'society of artists' which is 'in dialogue through images':

[This society] would dialogically envision, in images, situations that have never been seen and could not be predicted. It would be a society of players who would constantly generate new relationships by playing off moves against countermoves, a society of *Homines ludentes* in which inconceivable possibilities would open to human existence.

A dialogic approach highlights the importance of immersion and outsidership, of actively taking part and intentionally stepping out. It calls to mind that technological interaction is at best a game played between human players in which everyone can reflect and learn about herself in the pursuit of personal freedom. This may be interpreted as a response to the ongoing tendency to automate based on computational models of human intelligence. This research intends to 'reserve a seat' for the human performer on the digitalised and automated stage of music performance, not to play musical instruments in a traditional (embodied) way in exchange with automated computer-based composer-performers, but to contribute the 'human factor' to an otherwise inaccessible algorithmic structure and develop technology dialogically in mutual exchange.

### **6.3. The Five Aspects of Dialogic Coding**

The formulation of this complementary writing has involved placing concepts, practices and theories in relation to my dialogic coding practice. As such it has functioned through noting the insights, surprises and 'knowings' that emerge from the arts praxis or the 'theory imbricated within practice' (Nelson, 2013: 5). In this section, I summarize the five central insights of this thesis gained from Dialogic Coding practice. In a gesture of circular reconnection I may articulate these first as a response to the initial question of the research inquiry<sup>106</sup> from the introduction.

Dialogic Coding practice is a reflexive and potentially transformative way of making music which regards the apparatus – a programmable and interactive musical instrument – as a cocreative partner in group improvisation. In order to achieve moments of reflection and transformation through challenging interactions (acting/thinking) the programmer-performer needs to program the

---

<sup>106</sup>For the ease of reading I repeat the question here: 'Does a dialogic approach to the computer as musical instrument in situations of group improvisation enable interactions which provide possibilities for reflexivity and transformation for the performer?'

apparatus to act with a relative freedom on one hand, yet on the other hand this needs to be balanced against the freedom of the other improvisers including her own as a *functionary*-performer. To enable dialogic relationships and mutual subjectivity in this scenario the algorithmic performer needs to develop a '*virtuosity*' for *dialogue* which listens and speaks in many tongues.

Unpacking this dense summary the following aspects need to be assessed.

1. **Nontrivial interaction:** To bring about a dialogic performer-instrument relationship, the apparatus, needs to function in an independent way which makes interacting with it unpredictable and surprising for the performer. This is a relative freedom or a 'pseudo-autonomy' based on the concept of nontriviality and must be created and adapted by the performer by means of live coding the apparatus. In Dialogic Coding practice I have achieved this through inscribing various types of randomness (unweighted as well as weighted probabilities) into the algorithmic process but also by extending the apparatus structure out into a network integrating the interactions of other improvisers or the participating audience.
2. **Balancing different freedoms:** Dialogic Coding practice is situated in group improvisation. This implies an 'ethics of cocreation': to leave space for all participating agents, including the nonhuman entities. The particular challenge for the algorithmic performer lies in the double responsibility of the programmer – who invents the nontrivial apparatus but at the same time retains the possibility to reprogram it (her own freedom) – fulfilled only in dialogic relationship with the improvising group.
3. **Virtuosity of translating:** The basis for navigating this difficult course is a specific attentiveness or presence expressed in the term 'listening being'. Only by *giving* attention a performer is able to *respond* to the other and encounter difference. In Dialogic Coding practice it is necessary to perform such attention on multiple simultaneous levels: the intersubjective interactions with the audience, the improvising group and the autonomous instrument. For each of those the 'language' is different – taking the form of visual or musical gestures or written-text/program-code. The *virtuosity* of the dialogic coder in listening is to translate efficiently and creatively between these codes.
4. **Reflexivity/Transformation through immersion/challenge:** Dialogic Coding practice creates possibilities for reflexivity through experiences of being immersed or outside of the immediate actions of a performance.

Firstly, the experience of immersion in the improvised interactions and the emerging music facilitates dialogic relationships with the improvising group. On top of that Dialogic Coding allows for an immersion into the intellectual process of compositional thinking (the programming) as well. Secondly, this immersion is interrupted by challenges introduced through the functioning of the nontrivial apparatus which questions habitual interaction. In turn, a transformation of the performer's thinking or acting becomes possible which constitutes a development of her identity.

This leads ultimately to regard **Dialogic Coding as an epistemological tool**: It offers the possibility for reflection and transformation of one's habitual ways of making music and interacting with a programmable instrument (i.e. the computer) based on the following four attitudes:

1. An eagerness to interpret and decide and to do this quickly. This attitude values informed action as a way of understanding – which may be likened to von Foerster's aesthetical imperative: 'If you desire to see, learn how to act'. (2003:227);
2. To embrace and seek ambiguity which, from a programmer's perspective who is in constant need to make decisions, can be expressed in von Foerster's ethical imperative: 'always act so as to increase the number of choices' (ibid.);
3. To take risk and explore the unknown. To seek problems in order to find questions. To act in unexpected ways (the functionary's responsibility);
4. To take responsibility for one's actions; to invent structures and also re-invent them in response to a perceived call in performance;

To close this summarizing investigation I propose that the main responsibility of the algorithmic performer is concerned with freedom in order to make dynamic and dialogic human-apparatus as well as human-apparatus-human relationships possible. This is the ethical dimension of Dialogic Coding: to create a balance between the freedom of the apparatus, the multi-faceted freedom of the performer and the freedom of other partners in performance. Dialogic Coding practice may be understood as a realisation of Von Foerster's therapeutic imperative which says: 'If you want to be yourself, change!' (ibid.: 303). In paraphrase: if a live performance practice with technology is understood as a dynamic articulation and exchange of 'selves' then it should be oriented toward change, of the technological

users as well as technology itself. Dialogic Coding practice makes possible such changes in dynamic and flexible ways.

#### 6.4. Future Outlooks

As this project has progressed, the work has become more outward-facing and interactive, more about public engagement and how a performer may actually represent the relationships of power between herself and the technology. Following this motion, I am interested in how Dialogic Coding practice might be received by other, particularly younger audiences. In this direction it is worthwhile to consider how the problem of expert (programming) knowledge as a requirement for Dialogic Coding practice may be resolved to open up the practice for concerns, issues, thoughts and feelings of diverse groups and communities previously excluded. For this a comparative perspective which applies the insights from this research to other live coding languages may be helpful. This approach might create new idiosyncratic tools for laptop performance which are built for dialogue. I can imagine that a collection of such 'tools for Dialogic Coding practice' go beyond the use of text as the main form of interaction to realise a flexible approach which glides between various registers of interaction and provide a truly multi-modal access and experience of computer interaction through practice.

From a programming perspective there are technological paths worth exploring in order to create a more intricate and engaging dialogic exchange between performer and apparatus. The integration and application of more advanced computational models of artificial intelligence – such as neural nets, cellular automata or the dynamic field of machine listening and music information retrieval – provides interesting potentials to continue this investigation.

Another direction opening up is the integration of an outsider's viewpoint through spectator studies of Dialogic Coding practice. Such a methodological focus may help to develop the dialogic quality in the *acting* perspective of an algorithmic performer – an area which has not been in the centre of this research project. A performance practice with an orientation towards acting and representation may have interesting contributions to make in the ongoing discussion around liveness and laptop performance.

## 7 – Appendix

First, there is the glossary of the main terms used in this thesis. Second, follow supplementary materials to illustrate my research methods and programming processes. I have provided not provided any examples of actual SuperCollider program code here because it makes more sense to use them in digital form. All code files which supplement this thesis can be found online here:

<http://jonashummel.de/archives/code/>

### Appendix Items

0.1.Glossary of Central Terms	2
0.2.List of Online Repositories for Practice Documentation (code/audio/visual)	4
0.3.Notes from Noise Concert Experience (Ryan Jordan, 2013):	5
0.4.Conversation/Questionnaire Hans Tammen, December 2014	6
0.5.Notes and Memos from Developing the Practice (on Technospätzle)	9
0.6.Score Example for Technospaetzle (2014)	12
0.7.Memo on Free improvisation Session 'Kuehlspot'	14
0.8.Memo of 'The process of live coding' in the Studio	16
0.9.Text Score for 'MindYourOwnBusiness'	17

## 7.1. Glossary of Central Terms

Acting	refers to an aspect of performance which is concerned with 'representation, simulation, impersonation' (Kirby, 1974:46). To act is to 'be aware of an audience – to be 'on stage' (...) by energetically projecting ideas, emotions, and elements of their personality, underlining and theatricalizing it for the sake of the audience' (ibid.). In this thesis acting refers to an impersonation of processes which run in the apparatus through the performer's body.
Algorithm (Program)	is a set of rules to be followed in automatic calculations by a computer. The 'program' inscribed in the apparatus is synonymous. For example, in live coding praxis these are the rules for sound synthesis, processing and mapping of inputs onto the sound processes. 'Algorithmic' refers to any aspect related to the computational structure in the apparatus.
Apparatus/ Computer/ Machine	The terms machine, computer and apparatus used in this writing all refer to the computer as the instrument. 'Machine' represents a parent category for any automatic system, 'computer' refers to the digital programmable machine capable of generating synthesised sound in real-time. 'Apparatus' is a term introduced by Flusser (2000) to address the cultural and symbolic dimension of a computing machine: An 'apparatus' is a complex 'plaything (...) that simulates thought' (2000: 83), a programmed 'non-human agency' with an intention 'to change the meaning of the world' (ibid.: 25).
Authenticity/ Mutuality	in dialogic theory refers to the honest presentation of self to the other, or 'saying what's really on [one's] mind' (Buber 1965b:75). is the basis on which 'genuine' dialogue can emerge. It is based on authenticity and two further qualities: a 'presentness' to the other, or to listen to what she has to say, as well as an act of empathising which Buber calls a 'confirmation' of the other, or: to acknowledge and accept that the other's position is different to one's own. Cissna and Anderson (1998, in Phillips, 2011:32) hold that 'mutuality does not presuppose quality. This is impossible in any relationship. We can only have a degree of mutuality, even in unequal relationships.'
Bricolage (programming)	as a term was introduced first by Lévi-Strauss (1968), then applied to programming by Turkle and Papert (1990) and linked to live coding practice by Wiggins and McLean (2010). It describes the creative process as a 'mastery of associations and interactions' navigating by midcourse corrections. This approach is opposite to planning as a careful formulation of steps towards a specified goal. In contrast, 'bricoleurs have goals but set out to realize them in the spirit of a collaborative venture with the machine' (Turkle and Papert, ibid.:136). The process is likened to 'a conversation than a monologue.'
Creativity	describes 'the ability to come up with ideas or artefacts that are new, surprising, and valuable' (Boden, 2004: 1). In this context this happens during the activity of programming the apparatus: 'algorithmic creativity', or an invention of new computational structures. Furthermore there is 'musical creativity' involved in the process of improvising music between performers, i.e. the invention of new musical ideas and gestures.
(to) code	To write signs in a sign system arranged in a regular pattern
Composition	is understood as an activity which establishes the conditions for performance through which music is generated in a written form. In this context it is synonymous to the activity of programming understood as a transformation of specifications for musical structure in time, into the abstracted form of program code. It describes one end of a continuum of specifications for sonic outcomes related to performance time: while <i>composition</i> specifies 'ahead of time' (of performance), <i>improvisation</i> is understood to specify 'in time' (Butler 2014: 120). Both of these modes of creation work with 'pre-existent elements' (in form of code/recorded sound).
Dialogue (dialogic)	Buber (1965b: 78) defines the dialogue as a situation in which 'each should regard his partner as the very one he is. I become aware of him, aware that he is different, essentially different from myself, in the definite, unique way which is peculiar to him, and I accept whom I thus see, so that in full earnestness I can direct what I say to him as the person he is'. This 'dialogic encounter' is open to <i>listen</i> to alterity or the difference of the other. It thus emphasises coexistence of different standpoints over synthesis of ideas and describes a <i>quality</i> of interaction 'between people, texts, sentences, musical gestures' (Kanellopoulos, 2011: 124)
Dialogism	as term associated more with Bakhtin's theories about dialogue. For Bakhtin (1981) 'dialogue' is referring to the process of meaning-making, particularly in written or verbal speech acts. He introduces the ' <i>internal dialogism of the word</i> ' to refer to the 'traces of meaning' any speech act carries beyond the direct intentions by the speaker as the author of that speech. 'Dialogism' refers to all the meanings which this same word has incorporated through prior utterances through time and space. Additionally, any word proposes a specific expectation for a response in its medium of language.
Musicianship	describes the knowledge, skill, and artistic sensitivity in performing music. In this context this specifically includes the knowledge about computational systems and languages as well as the skill to fluently translate between different media in Dialogic Coding practice (music, code, visual gestures). Furthermore, in an improvised music context, it refers to the abilities of Kairos and Metis: to recognize

the propitious moment for playing a musical gesture – or executing a line of code.

Freedom	is understood in this context as the ability to exercise choices in improvised music performance. Thus it implies the capacity to account for one's own actions founded in this freedom.
Functionary	describes a specific way of human-apparatus interaction in which the human only 'calculate[s] and compute[s] instructions as instructed' (Flusser 2011:72), in other words the functionary acts as intended by the inventor of the computer program (the programmer) and can thus be considered part of the apparatus' program. In this thesis, it refers to an attitude of the performer which is present in the direct and immediate interaction with the musical process on any of the apparatus interfaces.
Programmer	is someone who 'inscribes an intention into an apparatus' (Flusser, 2011) by means of writing an algorithm in a specific symbolic system (a programming language). In this thesis, it describes that attitude or mode of apparatus-interaction of the performer which is concerned with the invention of (musical) structures (the compositional perspective). In this sense, the 'programmer's' attitude is present in the direct interaction with algorithms and therefore only indirectly related to the immediate musical process. It may materialise in the form of abstract thinking but also in the activity of pressing keys.
Immersion (Flow)	is considered to be what Csikszentmihalyi, cited in (Heble and Caines 2014: 151) has called 'the holistic sensation that people feel when they act with total involvement'. When in flow, one loses awareness of passing time, space and even self, being highly focused on the activity at hand. In this thesis immersion/flow has been identified while playing music directly, while listening intensely or while programming/live coding (an embodied <i>and</i> intellectual activity).
Improvisation	is understood as a form of composition, but 'just-in-time' not ahead of it. It describes the interplay between real-time creativity and the preexistent musical elements in the program code or apparatus interface. As such it is a model for action based on the two understandings: 1. <i>improvisation as intent</i> : to improvise in and for performance follows the goal to find new or compelling musical expressions or modes of interaction. 2. <i>improvisation as a tool</i> : to use improvised interaction as a way to explore new sounds or algorithmic structures.
Heteroglossia	is 'another's speech in another's language, serving to express authorial intentions but in a refracted way' (Bakhtin, 1981: 324). It is a form of a 'double-voiced discourse' which 'is always internally dialogized' between the intention of the author of the text and the 'author' of the speech act within the text, the actual speaker (ibid.).
Hacking	as a programming practice in a computer security context refers to someone who searches and exploits the weaknesses of a programmed system. In the local context 'hacking' describes the approach of exploring readymade technological structures (e.g. an electronic sound device but similarly an algorithm) with the intention to discover new ways of functioning and unexpected (sonic) results.
Liminality	is a concept by Turner who describes 'liminal entities [as] neither here nor there; they are betwixt and between the positions assigned and arrayed by law, custom, convention, and ceremonial' (1969: 95). A liminal entity, person or practice is characterised by this 'being between' the structural categories.
Transformation	refers to a change of status or the moving between the categories of a liminal practice. In the context of this thesis transformations may be produced through any dialogic interaction. Thus they apply to texts, sentences, program code, musical gestures and people, that is their intersubjectively constructed identities or self-concepts.
Live Coding	is a term which refers to the practice of writing algorithms and changing them while they are running during a performance. It is a performance practice within live computer music which works with generative structures (the sounds are created automatically by the computer) but includes a live programming performer who 'navigates' the algorithms on a musically feasible course. In live coding the algorithms are the means to produce music as well as the product of performance itself.
Listening	Is here understood to describe a certain attentiveness or presence towards one's surrounding environment. This is not confined to the acoustic environment but may include visual gestures, images or program code as well. The act of listening forms the basis of any dialogic relationship where it is the ethical gesture which 'embodies responsibility, an expression of contingent encounter, a co- dependent and co-creative ethical relation. It arises from self as a function of otherness' (Fischlin, 2014: 294).
Outsideness	is the act of distancing oneself from or of stepping outside a creative process. This is complementary to the act of being immersed or 'empathising with the other' – individual, text or work of art (music) – through practice. In a FIM context it can be likened to 'musicians struggle to raise their minds 'above' their playing, looking at their music from outside' (Kanellopoulos, 2011: 113).
Reflexivity	of a practice means that 'it' takes account of itself, or is aware of its constructedness. The practice becomes at once subject and direct object. For the performer this implies to acknowledge the effect of her actions for the products of that practice.
SuperCollider	is a real time audio synthesis programming language. The Language combines the object oriented structure of Smalltalk, features from functional programming languages, with a C programming language family syntax. Originating as proprietary software, it was released in 2002 by its author James McCartney under the free software GPL license. ( <a href="http://supercollider.github.io/">http://supercollider.github.io/</a> )

*Hummel, J. Dialogic Coding*

- Translation (to translate) describes here the process of transforming messages or code words from one language into another, a process of decoding and encoding from one symbolic 'universe' (Flusser) into another.
- Triviality / Nontriviality Triviality describes the property of a system which is defined by a stable relationship between input (stimulus) and output (response). A trivial system is deterministic: a particular input will always produce the same output. A Nontrivial system, on the contrary is defined by an unstable and dynamic relationship between input and output. Nontrivial systems are self-regulating in a way that 'its previous steps determine its present reactions' (von Foerster, 2003: 208). In other words, the outcome of the system cannot be calculated, it is unpredictable. Human beings are classified as nontrivial systems.

## 7.2. List of Online Repositories for Practice Documentation (code/audio/visual)

1. Code Repository for PHD: <http://jonashummel.de/archives/code/>
2. YouTube Channel:  
<https://www.youtube.com/channel/UCTvwLZW5E4pD7jP0L4mDncQ>
3. Vimeo Channel: <https://vimeo.com/monodread>
4. Soundcloud Channel with studio live coding:  
[https://soundcloud.com/fred\\_r\\_froyd](https://soundcloud.com/fred_r_froyd)
5. Playlist with all supplementary video recordings and excerpts:  
[https://www.youtube.com/playlist?list=PLKRUGJaqJkAfIS7EXUfK8RpL1\\_s-ykeB\\_](https://www.youtube.com/playlist?list=PLKRUGJaqJkAfIS7EXUfK8RpL1_s-ykeB_)
6. *Drums'n'Algorhythms* session: <https://vimeo.com/71763130>
7. Improvisation Sessions Berlin: <https://www.youtube.com/playlist?list=PLKRUGJaqJkAc3GWzIMj9uNiqGBUq8uKGV>
8. *SUNPYX* Project: <https://www.youtube.com/playlist?list=PLKRUGJaqJkAf0OSIcQvRtwT-eYkNo4nOI>
9. *Technospaetzle* project: <https://www.youtube.com/playlist?list=PLKRUGJaqJkAepPLO98g0Ovjd4lqvO7fnw>
10. *MindYourOwnBusiness* project: <https://vimeo.com/90767434>
11. *Projectionist Orchestra* project: <https://vimeo.com/114578144>
12. *SUM* project: <https://www.youtube.com/playlist?list=PLKRUGJaqJkAdjMNq4qHEmzaCoBD2IlnWd>
13. Additional improvisation sessions, Crewe: <https://www.youtube.com/playlist?list=PLKRUGJaqJkAcbRnYpxoyFAglGXIAxRsxK>
14. Playlist with all PhD-related recordings: <https://www.youtube.com/playlist?list=PLKRUGJaqJkAd1mSciFbzenRXf122LCDiE>

### 7.3. Notes from Noise Concert Experience (Ryan Jordan, 2013):

2.11.13

Yesterday, the noise=noise gig. Ryan Jordan, the organiser, was very nice. Very helpful. It is a strange little folk that listen to that kind of music. Somewhere between Punk and Zen. His performance was especially interesting. Involved a bright flashing light which was triggering sounds. So the ultimate synchronous strobo effect. The light was so bright it felt immediately physical. You could not look, had to close the eyes it was still very bright. But somehow a force was drawing me to it as well. It was a nice experience. Somehow dizzying and numbing, on the verge of trembling.

The other acts weren't as ecstatic. One was quite rhythmical. Another SC-bases but not a lot of movement. The microphone in the vagina spectacularly sounding - not really.

She took some time till getting into her. But then it sounded like something. All in all an interesting sight.

1.11.13 – Im Bus zu Les, Nachts

Aufm Heimweg vom Noise Concert. Schöner Raum. Im 7. Stockwerk der Regent Studios. Tolle Aussicht vom Balkon. Aber ich denke auch an Teneriffa.

Ryan Jordan, der Veranstalter, ein netter Typ. Zugänglich. Das Event ist natürlich punkig im Sinne von Bier muss fließen. Es ist ein bisschen Impro alles, ein erster gig mit Mikrowellen und Bierdosen drin, die krachen. Ryans Set selbst war das krasseste. Er hat ein massives Strobotlight verwendet. ultrahell, eine physische Erfahrung. hat auch Spass gemacht. Ich traute mich nicht die Augen zu öffnen, vorher hat er noch ein bisschen Atmo verbreitet und Sandelholz abgebrannt. Gute Idee. Dann gab's noch die Vagina.Mikro-Frau. Die war ganz Cool als Person, aber die performance war nur so halb cool.

Aber eigentlich wollte ich von meinem Stress schreiben. Dass ich mich total unter Strom fühle. Immer unterwegs und überall zu spät. Es ist komisch, ich sehen mich so nach Pause, ob das dann kommt? God knows. Ruhe. I look forward. Und mit Miri und Ju und Co?

Es ist alles ziemlich abgebrüht. momentan. So wenig Energy da. In Momenten, wo es nicht, so unter Stress steht, weiss ich plötzlich gar nicht was tun. Die plötzliche Leere

7.11.13 – Putting down a more thorough description of this experience in English:

Ryan Jordan, a British London-based noise artist, invited me to a performance in his East London studio space, a dark and concrete wall space. The atmosphere during the event, one in a series of noise and other experimental concerts, was conspiratorial. All listeners seemed to be part of the event and also presenting. The place was turned into a stage setup in a very ad hoc-improvised manner, full of scrap elements, randomly organized shelves, empty beer cans, a cash desk improvised on a table. When he started it was pitch dark, all lights had been turned off. It started with an incredibly bright flash of light with a synchronous loud burst of noise, echoing off. And then again. And again. It was rather repetitive, but still my body seemed to be drawn to it, I could not get away. Even though the light was too bright to look at, the sound too loud and harsh to listen to! These extremely physical 'energy bursts' felt like a massage. I felt dizzy and lost, with my eyes closed while I was listening 'inside' of me, standing in the small crowd of people. The massive bursts of sound and light would start slowly and accelerate up to the point when it became a steady and pulsing rhythmic texture.

## 7.4. Conversation/Questionnaire Hans Tammen, December 2014

### Hans Tammen - ENDANGERED GUITAR

<http://tammen.org/endedangered-guitar/> - with lots of performance excerpts  
<http://tammen.org/cycling74-interview/> - some older interview that may help  
<http://tammen.org/multichannel-endedangered-guitar-works/> - multichannel versions

#### 1. Who are you?

I started playing guitar in 1972, and in the 70s I wanted to be the next Richie Blackmore, playing Rock. I also studied classical guitar with Olivier Restellini. In the 80s I was a devoted Jazz guitarist, and in the 90s associated myself with the British Improvisers style. Since the 2000's I perform with my hybrid interactive guitar instrument, more in line of what is called Electroacoustic Improvisation.

#### 2a. What music do you make and why do you do it? Please explain, in as much detail as you see necessary, the way in which you perform. e.g. What kind of musical experience or aesthetic do you want to create?

Since the end of the 1980s I am interested in sound and rhythm, after 10 years of writing for jazz ensembles I lost interest in melody and harmony. As for sound or timbre, I see it less in Cage's than in Stockhausen's way: in that I juxtapose sounds, morph from one into the other, and generally organize them on a musical trajectory. In terms of rhythm I follow a similar strategy, in that I often superimpose different rhythms on top of each other. A third important dimension is dynamics, I often use radical shifts in volume.

#### 2b. How do you interact with the technology you use?

I created an interactive instrument, in that it draws different conclusions from my playing, and processes my sounds differently every time. That does not mean it is completely independent (and therefore fully unpredictable), but I use various grades of controlled randomness to surprise me during performance.

#### 2c: Why?

If I am not surprised during performance, my improvisations become dull and predictable. Improvisation means to handle unforeseeable situations, merely rearranging your musical ideas with every performance is no improvisation. In a group situation my collaborators may provide the surprises, with a computerized instrument you have to program the surprises into the machine.

#### 3. Describe your creative use of technology with your performance setup: a) Technologically:

See the slides attached which show the basic routing of the signal flow. Note: The "Control Analysis" part can affect any parameter in the software, not only the ones that arrows are pointing to. In recent years I have also processed my collaborators, so they would plug into the "Additional Input". See an example

1 here: <http://tammen.org/oxide-irmer-tammen/>

#### b) Strategically/musically:

I did not choose that setup before I started programming, it grew out of an already 10-year old practice as a (sound-oriented) guitar player. As it is often so, the first strategy (starting in 1999) was to simply mirror my setup in the computer, just because it is easier to carry around. But it takes only a few weeks to figure out that

there is much more possible. From then the hybrid interactive guitar setup emerged.

As for the musical possibilities, I deliberately sought out very different kinds of music collaborators, to see what I can do to make the system work in all situations. Well, obviously not everybody (country & western?), but everything that would allow for sound-oriented playing without being relegated to a position where I would just provide some sonic icing on the cake. I worked with contemporary classical performers, free jazz, british improviser types, DJs, electronica, berlin reductionists, electroacoustic improv. I used it extensively solo, but also in ensembles of up to 10 people.

#### **4. How did this setup develop?**

The setup varied constantly, from 2000 to around 2010 I have not played a single concert with exactly the same software. After every concert (and that includes being on tour where we played every night) there was always something that could be improved or changed, even if it was just something I needed to have more prominently on my screen.

What you see in my current software is probably just 10% of everything I ever programmed into it. Changes were often needed when I played with a very different set of players, different music demands different approaches to the instrument.

Generally if I didn't use a routine for a couple of months I threw it out. Some tools (such as my proximity sensor) stayed for many years, until I got tired of it. The basic structure (revolving buffers that allows for sample manipulation) is the same from the very beginning.

The interface changes regularly up until around 2006, I can see that from the screenshots I did save with the backups. From then on the basic interface did not change much, but still some of the underlying routines did. Since around 2010 I have only done small changes, the software did not change significantly.

#### **5. What kind of computer autonomy / intelligence do you use? Where in your setup do you use them?**

Controlled randomness by using the table object in Max, mostly. I use them everywhere - the software is written in a modular way (currently around 200 nested abstractions, bpatchers and poly~ objects, the maximum is 7 levels deep if I remember correctly). If a new routine is established, it takes around 5 minutes to have it available as a menu item, complete with reset strategies, all parameters accessible from any controller available (external and internal), etc.

The answer is then that it can be technically applied to every parameter, but of course it doesn't make sense to have it everywhere.

#### **Which parameters of the sound or modules in your setup work best with random decision making?**

For example I don't want any volume change to be uncontrollable. Dynamics is a very tricky issue, and here I need full control. But most other parameters work fine with uncertainty.

#### **does this usage change in different performance contexts?**

Yes. In ensembles I need more control. Solo is a completely different animal, here I can be as uncertain as I want.

#### **6. Where do you see the threshold towards too much complexity? or: When is it too little control?**

see the answer above - depends on the size of the ensemble. Of course, also on the music - in a fast-reaction context such as the british improv style you may want to react in a more precise and quick way.

**7.) While you perform how and in what terms do you "think"?**

**e.g. are you using the formalized terms of your computer setup (e.g. envelope attack-decay-sustain-release times or filter freq and resonance or playbackspeed of sample etc) or more traditional musical terms (e.g. intervals, transposition, variation etc)?**

I don't think in technical terms, and I also don't think in terms of pitch. "Variation" is a good one, though, because sometimes I come back to a previous idea and play a variation of it. I mainly think in terms of form, and dynamic changes.

Another issue is frequency - you want to stay out of the frequency ranges of the other players.

**8.) What role(s) do you (the performer) take on and what role(s) does the computer play? collaborator, mediator, input generator, improviser, (real-time) composer, mimicker, challenger or other (please describe)**

Well, in any case it's me who is performing, and the computer is more or less commenting on me? I would like to say that we could be equal, but in fact the computer can't continue if I don't want it to do so.

**9.) Do these rolesets change? when and how? perhaps give an example of a situation or a group setting which changed particular ways of playing your setup**

I can't remember any, although I am sure that in the beginning there were many changes that came by playing with different kinds of players.

**10.) More generally: For a digital/electronic improviser what skills and attributes do you consider to be essential?**

Probably the same as if you'd play the clarinet. Plus, since we can inhibit all the frequency bands, I am looking for sensible players who understand how to play in a transparent way, and stay out of other's bands.

**What potential and advantages do you see for using a computer?**

I am not so interested in the part where the computer just saves time and weight. Interactivity is something the computer can really bring to the performance.

**in other words: can you conceptualize your practitioners experience? how much of this is influenced by your personal performance style and knowledge of playing particular instruments (the guitar)?**

Everything. It grew out of a 10 year performance practice, and was an extension of it. It could also be said that my personal approach to music is still the same, this could probably be demonstrated in my chamber orchestra works, electronic orchestra works, and other non-guitar projects as well.

**11.) Is it important for the audience to understand how you are creating the performance for them to understand/enjoy/appreciate it?**

No. Nice to have sometimes, but not important.

**Please articulate why.**

Hummel, J. **Dialogic Coding**

This is not concept art, in which the concept is more important than the actual art work. If the music sucks I do not *want* to know how it is done. If the music is great, I don't *need* to know it.

## 7.5. Notes and Memos from Developing the Practice (on *Technospätzle*)

(August 2015, unedited original)

These rehearsals are more interesting than the actual concert, as they reveal the process how things evolved and how things were tried out. (as tom mentioned in the conversation: he took the course of some rehearsals to develop his particular “mixing desk technique” for performing). In the same way, did I try out different things.

My emotional involvement in the music is visible only very momentarily. Generally I am very much tunneled to my laptop screen. Whereas tom is very active operating his different devices in front of him. I am less doing so.

Insights from the playing (fully only after the project).

You need volume control at your hands, for every channel! This thought comes from the fact, that we wanted to be able to quickly transit from one thing to the other. As described below: this goes against the idea of navigating in indirect manner. But you want to be able quickly turn. Yet, then just like in sailing you need to prepare a few things, some necessary steps, before you can make the turn. Its not a simple turn. Whereas in traditional instrument, you simply stop moving or blowing. Its a simple, intuitive and fast action. Here, with machines, you need to prepare such quick transitions. And I did not do this in the beginning.

Then over the course of some rehearsals, material was collected. Now you wanted to be able to quickly execute those materials, so you would need a pool to which you had quick access. So I built a structure for that. In SuperCollider, this was a dictionary. From this arose a secondary problem: All entries to the dictionary needed to be uniform. They would have a name and a content. Now you needed to be able to remember them only by name and then know what lies behind the name to put them into the corresponding function, e.g. a pattern rhythm, a melody line, a filter function etc. This posed a challenge to my ability of memorizing things by names only. Clearly there are easier ways or assistive functions to remember things by names. It's a) a question of practice and memory, just like any vocabulary practice, and b) a question of clear and consistent naming – something which I am not good at, as this means, you need systematic naming. Because names are in this case symbols. So they point to their actual meaning, to what they refer to. For a particular content any number of symbols is possible but in practice only one kind feasible. These are some of the pitfalls of code design in development. And usually this goes very quickly in rehearsals. Hardly anyone spends many hours at home optimizing these insights after the session.

Same for effects: They need to be at hands for channel. Ideally.

In this project The code base grew bigger and bigger. Not necessary to my liking as it makes things more complicated for startup and more buggy, less error-prone.

But I threw in more and more modules. So there was the MasterFX Quark which seemed a handy way to find a **wrapped and pre-built version to quickly add effects** on a particular audio stream where I could **use my own selection**, and **most importantly, get a visual interface for it**, so that operation by mouse was also possible. (In fact, even though the code lines would have been much faster in the execution, I hardly ever used them. Mostly I controlled the effects by mouse on the screen or by assigning the slider/continuous value parameter of any effect to a hardware controller (NK2).

In rehearsals, especially the sculpting of bass sound through effects, like bit reduction and distortion and compression satisfied our common musical aesthetic. For high pitched sounds also some hard EQ-ing and reverberation was a good fit.

In rehearsals it was always a fight to focus on what I was doing, and hear actually what the differences were. I was, then, not sure enough about my activity, so I desperately needed some feedback to it. This is why I tunnel-focus on the screen so much, to be there and be sure. Tom made it more difficult at times because he had control over my signals in terms of volume and about effecting them with reverberation or other processes that would make clear audible feedback more difficult.

Hummel, J. *Dialogic Coding*

Tom mostly executed his role as “the last in the chain” to end a section when I he felt, it was over. By pulling the channel fader down quickly.

Good, equally leveled Volumes were a problem for the Free environment. If you are mixing stuff, you want to work with same levels to be able to swap one sound for the other without losing pressure!

And

Sync was hard one to get to.

Technically The MIDI Network abilities of Apple were useful and the feature of Ableton to interlink machines on that MIDI network clock. Within SuperCollider this posed to be more challenging. I had to realize that the building blocks I was using: That of JITLib with NodeProxies on the server and JITLib modules for playing patterns. It took me a while to understand how the SC architecture actually works. That processes running on the server dont follow a master sync unless you explicitly design them like that. The language editor follows a common master clock is therefore able to listen to incoming clock messaging.

Livecoding is mostly capable of doing micro adjustments, because it is a process of refining with successive steps. One follows the next. And sudden jumps are unexpected.

Most importantly: you need time to code and this is only possible if you leave things running, which again means, that things are changing slowly (slow iteration cycle) in general.

Wiederholbarkeit. *Repeatability*. Turns out to be a challenge really. From the improvising corner coming' and the complexity of the system in mind, nothing is exactly repeatable as played. Even when a rough approximation of it would be enough to work on something, to develop it, compositionally, it was not really possible to do so.

This, I reckon, mainly comes from the level of complexity and how code is written (overwritten). So that it is difficult to remember every single step that you did. Every single number that you wrote. You could reconstruct it with the help of some tools like the *History* class, but still this is too slow to be useful in rehearsals.

And also: your own not-being-satisfied with something in a way as to keep it.

Also sometimes loudness-wise: a sound wouldnt have enough punch, but there are many times when you “overdesigned” and you arrived at some sound, totally different thing and not useful for the initial purpose. File this under

Overdesigning or when to stop:

T e m p o            a d j u s t m e n t            a n d            r h y t h m i c a l            r e p e t i t i o n :

Sometimes I also would have liked to be able to change a tempo or the offset of a running loop like you would on a turntable. (Even beyond this the idea of a turntable as an ideal object representing loops (a circular thing) became clear to me). There this is normally called the *nudge* functionality. As a human performer can do this at an instant: just halt and continue where you left off, or (a little more difficult) start from beginning of loop. And also I realized through soundpainting: halting and continueing (pause and resume) and stutter-repetetion in a rhythmical way (following a perceived global tempo) are aesthetics introduced by turntablists and samplers, based on the fact that you have recorded music as material to play with. Now soundpainting says: we dont need recorded material, we can do that ourselves, we then just mimic the machine behaviour.

Astoningly difficult to do for a machinic setup if you are not prepared for that!

Why were you not prepared? Because I tried not to follow popular electronic aesthetics but rather see what is possible beyond that!

Because you are unable (with most interfaces) to play something manually at real-time speed, one likes to use repeating structures, so that you don't have to execute once for every sound. And also because you can do that with a machine.

Difficulty lies therein: Once you set the things running, you have difficulty going out of them again. Then you are running next to them, like letting something roll downhill. A bunch of logs rolling downhill. You can only stop one of them. Its difficult to do for all.

So I needed a “stop all” and “run all” thing. The “run all” was rarely used. The “stop” function was.

But usually you dont just pause and resume. You usually break and continue differently.

“Melodicality”

It turns out to be difficult to play a repeating note pattern in the grid of an overall track structure. And then if one pattern runs you just leave it running because it takes some attention to watch and sculpt it. All the while you need to prepare something else. So you are only able to do so much in your actual real-time

Complexity

A wise word. Complex term. In essence: all you can see, oversee, understand. So code is quickly complex, because you cannot oversee everything. Just by means of having no visual representation of what is running & playing at that instant.

This is a re-occurring problem in all.

## 7.6. Score Example for Technospaetzle (2014)

- Jonas beginnt mit Fahren-Drone **[CH 3]**
- Jonas spielt dazu Vocal-Samples (Trainmanager und Tesco vending machine) **[CH 4]**
- Tom fährt die Samples langsam in den Hall und Sidechaned dann den Hall mit Kick
- beginn BULLBUB

### **BULLBUB (123)**

- Jonas startet Bass **[CH 1]** (Variante mit 16tel Basslauf)
- Mindestens ein Breakdown
- > AUF SIGNAL HALFTIME
- Jonas startet hohen Bass auf **[CH1]** —> SIGNAL FÜR ABBAU
- Jonas beendet Lied mit unrhythmischem Bassgeklacker

### **SIDEKICK (128)**

- Tom faded in Bassgeklacker Hihat/Shaker ein
- Alles andere AUS! paar Sekunden nur Hihat + Evtl. FX
- Jonas Spielt Tomsound in Vierteltrioen **[CH 2]**
- Fetter Bass auf [CH 1] den Jonas immer knackiger macht
- Jonas blubbert immer mehr Tomsounds dazu
- Jonas Macht FX auf die Tomsounds
- ende Bass (nur Grundton) und SD
- Tom Zieht am Ende abrupt raus
- > Jonas beginnt schon während Bass am Ende mit "Please don't flush" Samples **[CH 4]**

### **FRICKY (119)**

- Jonas: "dummdudumm" bass auf **[CH 1] !!! TOM BASS QUANTISIERUNG ANGLEICHEN!!!**
- Jonas: Spielt Collins Amp **[CH 2]** (!! Bässe bisschen raus)
- Jonas: auf **[CH 3]** punktierte-8tel-perc-sound
- Jonas: "Please don't flush" and "burial check" - Samples **[CH 4]**
- Tom Beat dazu (mit Percs anfangen! später Hat dazu)
- Tom Szene weiterschlafen -> "dummdudumm"-Sound kommt dazu
- TOM: SIDECHAIN AUF HALL MIT GATE
- TOM: MIT FILTER AUF PERCS ARBEITEN
- ENDE UNKLAR!
-

### **DRIP (128) (Triolen feel)**

- Tom: Bass Sidechain!!!
  - Jonas Startet BASS am Anfang!
  - Jonas: Fetter Basston **[CH1]** mit dem Jonas auf dem Nano so rumspielt:) (—> HIER GRAINDELAY)  
—> Auf Zeichen dann irgendwann den "dudeligen" Bass als fill
  - Jonas: Tomspund Triolisch **[CH2]**
  - Jonas: rythmischer Bass **[CH 3]**
  - Jonas: Hihats auf **[CH4]**
- TOM: mit GRAIN\_DELAY arbeiten (Kick Side-Chain kann mit "R"-Taster (bei Fader 3) aktivier werden!!)

### **SQUILLYBONES (124) (Deepphouse)**

- Tom: startet mit Groove (erst nur Hats)!!
- Jonas: 8tel-Bass auf **[CH 1]** —> mit Moll-Tönen (Tom: Mit Graindelay zum Bass Terz dazu!!)
- Jonas: 4tel "Rhodes"-Sound auf **[CH 2]**
- Jonas: Synthsound **[CH 3]** —> Tom in Hall preset 1
- Jonas: Rauschsounds + Blip Sounds **[CH 4]**

### **SIDEKICK (126) (Dreams or Goldfish)**

- Kick beginnt!! —> FX von Jonas
- Rauschen mit "Destroy"-FX Kanal erzeugen
- Jonas: "BASSDROP" **[CH 1]** (Kann heftig von Jonas gefiltert werden)
- Jonas: SNARE auf **[CH 2]**
- Jonas: "boiboiiiiii"-Sound **[CH 3]**
- Jonas: Hit-Sounds **[CH 4]** auch oneshots!!
- Tom: SD mit Rechner-Reverb bearbeiten (ganz kurz einstellen und mit Freese arbeiten)

—> Am Ende alles immer Noisiger, Beat dann raus, alles rhythmische raus —> Ende

## 7.7. Memo on Free improvisation Session 'Kuehlspot'

(15.11.14, unedited original)

(...)

So How did it go?

Miri said she was able to recognize the different sounds I was using as a sort of "shades of my instrument". Different characters. A nice observations as it quite fits my own perspective on the whole thing.

Also did she say this morning: Do you many other people playing in this kind of setting? Which means: a setting which is rooted in jazz improvisation and free improvisation by acoustic instruments. Where basically, each instrument is playing on its own, no amplification as a "whole". And in this sense: the computer that I bring is truly an "instrument", as object, as the other instruments in the round.

And secondly the style of playing.

Which is sometimes very vivid and fast-changing so it affords a need to quick replies and responsivity. Not in the sense of close imitation but rather in timbre or expression character maybe (at least this would be my intention or aim).

On a very general note, I felt the thursday session to be flowing much better for me. Was it partly because of the acoustics of the place (not so reverberant, all musicians on the stage which is another part of the concert room, so with own acoustics) or as well because of the more relative darkness and the reclusiveness of my position (far stage right, corner).

Yesterday there were not so many moments of being "inside the music" really. Which I could not exactly say why this is. Again maybe the seating position was disturbing (facing away from group and audience in the first half). and then there were longer parts which worked with melodic lines as well as expressive (romantiv) cadenza style, harmonic progression. Which always seem so excluding to me, as the computer musician, who has not prepared any such style of musical interaction (no keyboard at hand). This leaves me mainly with the contribution of noise-based sound and processed qualities (reverb, distortion) which especially in this line-up was not of particular interest.

In line with my prejudice about this style of jazz-rooted free improvised music, it is a lot about "playing", meaning noodling-doodling of tonal sequences and turn-arounds. well, lets not be so accusing. It is in parts about that. Or enjoys being active in Tongirlanden or Tonketten. And to me this is not a choice. As I do not have the ability to play singular notes.

So in general very much attentive again to myself, and my playing, my sound. As noticed in previous sessions, but very much with a critical focus. So: What does not work? Or what am I NOT able to do?

One of them being the tonal playing and the other one being the groove-playing.

Melody and groove are two aspects which the machine does not automatically do as it requires a lot of previous musical knowledge about tone relationships and grooves common to any well-educated musician but not to the machine.

Ergo, only if such knowledge has been programmed into the machine, it is able to perform such things live in the concert situation.

And I have not devoted energy to this.

So. What did work out well then?

I think in terms of my own setup, I was quite satisfied with having master Effects of processing "at hand", that is mapped to my controller and being able to slowly modify it with knobs rather than Mouse-based GUIs, and also not having to code the progression, but being able to extract it from or control it through this body knowledge of my fingers, led by my ears.

This works together with my decided-upon interface of the multitouchpad interface. Which is essentially finger based.

Sometimes it felt too small for me, the area of interaction which is just a rectangle of about 10x12 cm. But I was happy with the fact that I understood the control mapping better, to better be able to meaningfully control sounds rather than indeterminately running behind the sound which is leading the thing.

So responding to the affordance of that situation: playing in this jazz-rooted free improvised music with lots of quick changes and fast call-response dialogues.

I wondered sometimes why in a particular case the particular ongoing "motivik" or playing stopped and something else emerged from (below) it. Sometimes I continued my doing (yesterday) deliberately against it. Sometimes I went along with the group. But especially yesterday in comparison to thursday I had more moments were I felt that there was a difference in "feeling" or decision in such moments. Where I felt alone because of having missed the collective point of decision to stop or not continue something (my assumption). At least it was not so much "in sync". Which is only negative in the fact that it does

not enable me as much to "enter the flow" of a collective experience. Which I am not sure of, noted on the side, whether this is an aimed-for goal on anyone else's side, really.

Could also be just me.

What else do I remember. Hm, difficult to say from this moment (Saturday afternoon).

Too many impressions have already moved away to some extent. memory faded.

I remember (again self-focused vision) that I had some thoughts were I wanted to play percussive or drums-like with live coded stuff, but I wasn't all too able to do so.

Still I used some livecoding to play rhythmical structures which I however was not able to terminate or change quite abruptly (as noticed by Miriam as well) (solo intro of second half with Hui Chun). And Also I noticed how massive the dynamic range of all the acoustic instruments together is and how difficult this is for me with the speaker to move along.

Still, even with having put the volume control out on the faderbox, I could not hold up to this.

Sometimes, not often, it was too loud for the silent moments, more often times I could not be loud enough for the screaming drums and trombones next to me. The only way to "survive" (meaning cutting through the noise floor) was to navigate to "free" corners of the frequency spectrum. mostly in the very upper range.

Also, remember thoughts of me moving to other "free corners" of the lineup, like missing instruments in the band. Bass player being the other one. so the very low end. Even though my speaker was not necessarily powerful enough or my sounds were not optimized for a "focused" (pragnant) sound in the lower ranges. But at some point I also wished to be able to throw in a moving bass line, like a swing bass or some funkier line. just to "underlay" the other activities in the band.

Also, together with this impression of excludedness or isolation I felt that my sounds make it especially possible for me to "throw in comments" like momentary motifs, not meant for development, because mostly noise based which make it more difficult to "develop" in the classical sense (variation of a motif). So in other words I could decorate or color the whole sound.

## 7.8. Memo of 'The process of live coding' in the Studio

(23.11.15, unedited original)

An ad hoc approach

Starting Condition: I want to make a sound with laptop.

0. What are your actions inspired from?

Singular Musical Elements

- by a rhythm figure that you hear
- by a melodic motif that you hear
- by a sound that you hear (timbral qualities)

Their Mix

- by a fragment of music (the combination of singular elements)
- need to deconstruct this 'mix'

The Body Dimension

- by a choregraphic (visual) idea that you see (a movement sequence)
- by a body impulse to use particular body parts

narrative aspects

- by a change in music over time: harmonic progression, timbral development (e.g. an EDM build-up)

1. How do I want to "make" the sound?

-> Then how should it be triggered, through which action?

a) automated

By an algorithmic process (a pattern sequencer)

By design: e.g. a self-triggering process

b) some gestural interaction

a computer key press

a mouse click

a press on touch pad or some other touch sensitive surface

c) some other sensor interaction

hand movements affects ambient light -> built-in light sensing

hand or other object affect acoustic feedback -> built-in microphone

combined with live audio analysis (pitch, volume, FFT analysis)

This approach gets more complicated quickly:

Because one has to build an interactive system first, and define it's functional properties (at which sensor value does it react, what is the connected reaction?). It then becomes a question of mapping.

Underlying question to this, is: What is my input to the machine as performer?

a) bodily actions

b) conceptual thoughts (composition)

2. What sound should it be?

Different ways to create a sound with machine

a) synthesized

- what 'ingredients to the recipe'?

- what should the aesthetic qualities of the sound be

freq spectrum

internal dynamics ('life')

b) sample-playback

- What is the important content of sample? Rhythm, timbre, location sound, soundscape

qualities

- does the origin matter? or is it just 'material'

- a single event ('tone') or longer sample?

c) additional process stages

- tonal qualities: filter, harmonics, noiselevel (saturation)

- spatial qualities (reverberation)

- temporal qualities (delay, echo, amp-modulation)

3. How do you proceed in playing with the sound?

a) automated scoring

sequencer live programming

algorithmic sequencing

b) dramaturgies (scores) to go by

what are the formal elements in it?

beginning, end, chorus, verse, variation, build-up, climax, release, repetition

## 7.9. Text Score for 'MindYourOwnBusiness'

Jonas Hummel

+++ April 2013 +++

### 6) CONTROL CONSTRAINTS (mind your own business)

For a networked setup in the form of the SuperCollider Republic quark.

There are as many sounds/instruments/synths as participants available.

Each performer can use as many patterns (Pdef) and tasks (Tdef) as participants.

Now the different musical parameters are distributed to each performer in the group:

Everyone has a different responsibility('job'):

- a) Take care of sound: Write sound synthesis recipes or manage buffer contents  
(the parameterNames of sounds should be generic like x,y,z)
- b) Take care of timbre: modify the parameters x,y,z
- c) Take care of rhythm: modify only the dur/sustain parameters
- d) Take care of Volumes: modify the amp parameter
- e) Take care of spatial placement: modify only the 'where' parameter

The jobs can be less/more according to group size.

It will all be live coded. It starts from writing the sounds/setting the buffers.

The jobs can & will be shifted around the group if more than half of the performers request to do so (via Chat Communication or visually / by gesture).

The piece is over if everyone has done every job once.

#### **/\* QUESTIONS:**

Everyone plays on a central server so that there are not duplicate events, patterns, etc.?

Or else: how to integrate different inputs from different players?

**Possibly: play NDEFs and setup OSCresponders**

**\*/**

## 8 – References

- Alexander, A. (2009) 'Programming by Ear (or How to Give up Control and Still be a Control Freak).' *In Contemporary Music Review*, 28(1), pp. 115-128.
- Arns, I. (2005) *Code as performative speech act*. Unknown place of publication: #Artnodes [online] [accessed on 3<sup>rd</sup> March 2015] Available from: <http://www.uoc.edu/artnodes/eng/arns0505.pdf>
- Ashby, W. R. (1956) *An Introduction to Cybernetics*. London: Chapman & Hall.  
Internet (1999) [Online] [Accessed on 1<sup>st</sup> March 2017] Available from: <http://pcp.vub.ac.be/books/IntroCyb.pdf>
- Auslander, P. (2008). *Liveness: Performance in a mediatized culture*. Routledge, New York.
- Böhme, G. (2013) *Atmosphäre. Essays zur neuen Ästhetik*. (Atmosphere. Essays for the new Aesthetics.) 7<sup>th</sup> ed., Frankfurt a.M.: Suhrkamp.
- Bailey, D. (1992) *Improvisation. Its Nature and Practice in Music*. Cambridge, MA: DaCapo Press.
- Bakhtin, M. (1981) *The dialogic imagination. Four Essays*. trans. Emerson, C. and Holquist, M., ed. Holquist, M., Austin: University of Texas Press.
- Bakhtin, M. (1986) *Speech genres and other late essays*. trans. McGee, V.M., eds. Emerson, E. C. and Holquist, M., Austin: University of Texas Press.
- Bakhtin, M. M. (1993), *Toward a philosophy of the act*, trans. Liapunov, V., eds. Liapunov, V. and Holquist, M., Austin: University of Texas Press.
- Berliner, P. F. (1994) *Thinking in Jazz: The Infinite Art of Improvisation*. Chicago: University of Chicago Press.
- Blackwell, A. and Collins, N. (2005) 'The programming language as a musical instrument.' *Proceedings of PPIG05 (Psychology of Programming Interest Group)*, 3 pp. 284-289.

- Blackwell, A., McLean, A., Noble, J. and Rohrerhuber, J. (2014) 'Collaboration and learning through live coding (Dagstuhl Seminar 13382).' *Dagstuhl Reports*, 3(9) pp. 130–168.
- Blain, M. (2016) 'Practice-as-research: A method for articulating creativity for practitioner-researchers.' In Haddon, E. and Burnard, P. (eds.) *Creative Teaching for Creative Learning in Higher Music Education*. Abingdon: Routledge, pp. 79-92.
- Boden, M. A. (2004) *The Creative Mind. Myths and mechanisms*. 2<sup>nd</sup> ed., London: Routledge.
- Bolt, B. (2007) 'The magic is in handling.' In Barrelet, E. and Bolt, B. (eds.) *Practice as Research: Approaches to Creative Arts Enquiry*. London and New York: IB Tauris & Co Ltd., pp. 27.34
- Booth, G. and Gurevich, M. (2012) 'Collaborative composition and socially constructed instruments: Ensemble laptop performance through the lens of ethnography.' In NIME 2012: Proceedings of the International Conference on New Interfaces for Musical Expression, Ann Arbor: University of Michigan. pp: 21-23.
- Bowers, J. (2002) *Improvising Machines. Ethnographically Informed Design For Improvised Electro-Acoustic Music*. M.A., University of East Anglia, Norwich, UK.
- Brackett, J. J. (2010) 'Some Notes on John Zorn's Cobra.' *American Music VL*, 28(1) pp. 44-75.
- Buber, M. (1937) *I and Thou*. trans. Smith, R.G., Edinburgh: T & T Clark.
- Buber, M. (1957) 'Elements of the interhuman.' *Psychiatry*, 20 pp. 105-113.
- Buber, M. (1965a) *The Knowledge of Man: A philosophy of the interhuman*. Friedman, M. (ed.), New York: Harper & Row.
- Buber, M. (1965b) *The knowledge of man*. trans. Friedman, M. and Smith, R. G., London: George Allen and Unwin, Ltd.

- Buber, M. (1965c) *Between man and man*. trans. Smith, R. G., New York: Macmillan.
- Buber, M. (1969) *Between Man and Man*. trans. Smith, R. G., New York: Macmillan.
- Buber, M. (1970) *I and Thou*. trans. Kaufman, W., New York: Charles Scribner's Sons.
- Butler, M. J. (2014) *Playing with Something that Runs: Technology, Improvisation, and Composition in DJ and Laptop Performance*. Oxford: Oxford University Press.
- de Campo, A. (2014) 'Lose control, gain influence - Concepts for Metacontrol.' In *Proceedings of ICMC/SMC. 2014*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library.
- Cascone, K. (2002). 'Laptop music-counterfeiting aura in the age of infinite reproduction', *Parachute: Contemporary Art Magazine*, pp. 52-59.
- Chadabe, J. (1984) 'Interactive Composing: An Overview.' *Computer Music Journal*, 8(1) pp. 22-27.
- Cocker, E. (2013) 'Live Notation: -- Reflections on a Kairotic Practice.' *Performance Research*, 18(5) pp. 69-76.
- Collins, N., McLean, A., Rohrhuber, J. and Ward, A. (2003) 'Live coding in laptop performance.' *Organised sound*, 8(3) pp. 321-330.
- Collins, N. (2003). 'Generative Music and Laptop Performance', In *Contemporary Music Review*, 22 pp. 67-79.
- Collins, N. (2011). 'Live coding of consequence', *Leonardo*, 44 pp. 207-211.
- Collins, N., Schedel, M. and Wilson, S. (2013) *Electronic music*. Cambridge and New York: Cambridge University Press.
- Collins, N., (2006), *Handmade electronic music: the art of hardware hacking*. Taylor & Francis.

- Corbett, J. (2016) *A Listener's Guide to Free Improvisation*. Chicago: University of Chicago Press.
- Cox, G. and McLean, A. (2013) *Speaking Code: Coding as Aesthetic and Political Expression*. Cambridge: MIT Press.
- Csikszentmihalyi, M. (2002) *Flow: The classic work on how to achieve happiness*. London: Random House.
- Csikszentmihalyi, M. (2014) 'A Theoretical Model For Enjoyment.' In Heble, A. and Caines, R. (eds.) *The Improvisation Studies Reader: Spontaneous Acts*. London: Routledge, pp. 150-162.
- De Bono, E. (2000) *Six Thinking Hats*. 2<sup>nd</sup> ed., London: Penguin Books.
- Dell, C. (2002) *Prinzip Improvisation*. (the principle of improvisation). Koeln: Walther König.
- Ellis, C., Adams, T. E. and Bochner, A. P. (2016) *Autoethnography: An Overview*. [Online] [Accessed on 04.06.2016] Available from: <http://www.qualitative-research.net/index.php/fqs/article/view/1589/3095>
- Emerson, G. (2015) *Gesture-Sound Causality from the Audience's Perspective: An Investigation of the Influence of Mapping Perceptibility on the Reception of New Digital Musical Instruments*. M.A. Humboldt-Universität Berlin.
- Emmerson, S. (2007) *Living electronic music*. London: Ashgate.
- Erickson, J. (2008) *Hacking: The Art of Exploitation*. 2<sup>nd</sup> ed., San Francisco: No Starch Press.
- Fairhall, A. L. (2008), *Intertextuality and the Dialogic Principle in Jazz*. Ph.D. Manchester Metropolitan University.
- Fischlin, D. (2014) 'Improvised Responsibility.' In Heble, A. and Caines, R. (eds.) *The Improvisation Studies Reader: Spontaneous Acts*. London: Routledge, pp. 289-295.

- Fischlin, D., Heble, A. and Lipsitz, G. (2013) *The fierce urgency of now: improvisation, rights, and the ethics of cocreation*. Durham and London: Duke University Press.
- Flusser, V. (2000 [1983]) *Towards a philosophy of photography*. trans. Matthews, A., London: Reaktion Books.
- Flusser, V. (2002) 'What is Communication.' In Andreas Ströhl (ed.) *Writings*. Trans. Eisel, E., Minneapolis: University of Minnesota Press, pp. 3-7.
- Flusser, V. (2011 [1985]) *Into the Universe of Technical Images*. Minneapolis: University of Minnesota Press.
- von Foerster, H. (2003) *Understanding Understanding - Essays on Cybernetics and Cognition*. New York: Springer.
- Friebe, H. and Ramge, T. (2008) *Marke Eigenbau: Der Aufstand der Massen gegen die Massenproduktion*. Frankfurt and New York: Campus.
- Friedman, M. S. (2002) *Martin Buber: The life of dialogue*. Psychology Press.
- Garfinkel, H. (1967) *Studies in Ethnomethodology*. Englewood Cliffs, NJ: Prentice Hall.
- Garfinkel, H. (1988) 'Evidence for Locally Produced, Naturally Accountable Phenomena of Order, Logic, Reason, Meaning, Method, etc. In and as of the Essential Quiddity of Immortal Ordinary Society, (I of IV): An Announcement of Studies.' *Sociological Theory*, 6(1) pp. 103-109.
- Ghazala, Q. R. (2004) 'The Folk Music of Chance Electronics, Circuit-Bending the Modern Coconut.' *Leonardo Music Journal*, 14 pp. 96-104.
- Ghazala, Q. R. (no date) *Anti-Theory*. [Online] [Accessed on 1<sup>st</sup> April 2015] <http://www.anti-theory.com/bio/>
- Gordon, M. (2011) 'Listening as embracing the other: Martin Buber's philosophy of dialogue.' *Educational Theory*, 61(2) pp. 207-220.

Gresham-Lancaster, S. (1998) 'The Aesthetics and History of the Hub: The Effects of Changing Technology on Network Computer Music.' *Leonardo Music Journal*, 8(1) pp. 39-44.

Gustavsen, T. (1999) *The dialectical eroticism of improvisation*. Ph.D. University of Oslo.

Haseman, B. (2006) 'A Manifesto for Performative Research.' *Media International Australia incorporating Culture and Policy*, 118 pp. 98-106.

Heble, A. and Caines, R. (2014) *The Improvisation Studies Reader: Spontaneous Acts*. London: Routledge.

Hennion, A. (2003) 'Music and mediation: Towards a new sociology of music.' In Clayton, M., Herbert, T. and Middleton, R. (eds.) *The cultural study of music: A critical introduction*, London: Routledge, pp. 80-91. Available from: <https://halshs.archives-ouvertes.fr/halshs-00193130>

Hennion, A. (2007) 'Those things that hold us together: taste and sociology.' *Cultural sociology*, 1(1) pp. 97-114.

Holtzman, B., Hughes, C. and Van Meter, K. (2007) 'Do it yourself... and the Movement Beyond Capitalism.' In Graeber, D. and Shukaitis, S. and Biddle, E. (eds.) *Constituent Imagination. Militant Investigations, Collective Theorization*. Oakland, Edinburgh und West Virginia: AK Press

Horn, D. (2000) 'The Sound World of Art Tatum.' *Black Music Research Journal*, 20(2) pp. 237-257.

Huizinga, J. (1980) *Homo Ludens*. 1949 London: Routledge.

Hummel, J. (2013) *Notes from improvised electronic music 2013*. unpublished.

Hutchins, E. (1995) *Cognition in the Wild*. 1995. Cambridge, USA: MIT Press.

Kanellopoulos, P. A. (2011) 'Freedom and Responsibility: The Aesthetics of Free Musical Improvisation and Its Educational Implications---A View from Bakhtin.' *Philosophy of Music Education*, Review 19(2) pp. 113-135.

- Klett, J. and Gerber, A. (2014) 'The Meaning of Indeterminacy: Noise Music as Performance.' *Cultural Sociology*, 8(3) pp. 275-290.
- Kirby, M. (1995 [1972]), 'On acting and not-acting', in P. Zarrilli (ed.), *Acting (Re)considered*, London: Routledge, pp. 43–58.
- Latour, B. (1987) *Science in action: How to follow scientists and engineers through society*. Boston: Harvard University Press.
- Latour, B. (2005) *Reassembling the Social: An Introduction to Actor-Network-Theory*. Oxford: Oxford University Press.
- Levy, S. (1984) *Hackers: heroes of the computer revolution*. Harmondsworth: Penguin.
- Lewis, G. E. (1996) 'Improvised music after 1950: Afrological and Eurological perspectives.' *Black Music Research Journal* pp. 91-122.
- Lewis, G. E. (1999). 'Interacting with latter-day musical automata', *Contemporary Music Review*, 18 pp. 99-112.
- Lewis, G. E. (2007) 'Live Algorithms And The Future Of Music.' *CTWatch Quarterly*, 3(2) [online] [accessed 10<sup>th</sup> October 2014] Available from: <http://www.ctwatch.org/quarterly/articles/2007/05/live-algorithms-and-the-future-of-music/>
- Lewis, J. (2013) 'Dialogue As a Way of Knowing: Understanding Solo Improvisation and Its Implications for an Education for Freedom..' *Psychomusicology: Music, Mind & Brain*, 23(4) pp. 255-261.
- Lipari, L. (2004) 'Listening for the Other: Ethical Implications of the Buber-Levinas Encounter.' *Communication Theory*, 14(2) pp. 122-141.
- Lipari, L. (2010) 'Listening, Thinking, Being.' *Communication Theory*, 20(3) pp. 348-362.
- Magnusson, T. (2011). 'Algorithms as scores: Coding live music', *Leonardo Music Journal*, 21 pp. 19-23.

- Magnusson, T. (2014) 'Herding cats: Observing live coding in the wild.' *Computer Music Journal*, 38(1) pp. 8-16.
- de Man, P. (1983) 'Dialogue and Dialogism.' *Poetics Today*, 4(1) pp. 99-107.
- McKenzie, W. (2004) *A Hacker Manifesto*. Boston: Harvard University Press.
- McLean, A. (2011) *Artist-Programmers and Programming Languages for the Arts*. Ph.D. Goldsmiths, University of London.
- McLean A. and Wiggins, G. (2010) 'Bricolage Programming in the Creative Arts.' *Proceedings of the Psychology of Programming Interest Group*, [online] [accessed on 30<sup>th</sup> May 2012] Available from: <http://yaxu.org/bricolage-programming-2/>
- Mendes-Flohr, P. (2015) *Dialogue as a Trans-disciplinary Concept: Martin Buber's Philosophy of Dialogue and its Contemporary Reception*. Berlin, Munich and Boston: Walter de Gruyter GmbH & Co KG.
- Monson, I. (1994) 'Doubleness and Jazz Improvisation: Irony, Parody, and Ethnomusicology.' *Critical Inquiry*, 20(2) pp. 283-313.
- Monson, I. (1996) *Saying something - jazz improvisation and interaction*. Chicago: The University of Chicago Press.
- Mori, G. (2015) 'Analysing Live Coding with Ethnographical Approach.' In . *Proceedings of the First International Conference on Live Coding 2015*. McLean, A., Magnusson, T., Ng, K., Knotts, S., Armitage J. (eds.)
- Morson, G. S. and Emerson, C. (1990) *Mikhail Bakhtin: Creation of a prosaics*. Stanford: Stanford University Press.
- Nelson, R. (2006) 'Practice-as-research and the Problem of Knowledge.' *Performance research*, 11(4) pp. 105-116.
- Nelson, R. (2013) *Practice as research in the arts: principles, protocols, pedagogies, resistances*. London: Palgrave Macmillan.
- Nettl, B. (1974) 'Thoughts on Improvisation: A Comparative Approach.' *The Musical Quarterly*, 60(1) pp. 1-19.

- Nicholls, T. (2012) *An ethics of improvisation: Aesthetic possibilities for a political future*. London: Lexington Books.
- Nikulin, D. (1998) 'Mikhail Bakhtin: A Theory of Dialogue.' *Constellations*, 5(3) pp. 381-402.
- Orr, T. (2014) *Evaluating Performance 2014*. unpublished.
- Peeren, E. (2007) *Intersubjectivities and popular culture: Bakhtin and Beyond*. Stanford: Stanford University Press.
- Phillips, L. (2011) *The promise of dialogue: The dialogic turn in the production and communication of knowledge*. Amsterdam and Philadelphia: John Benjamins Publishing.
- Pinch, T. and Trocco, F. (2004) *Analog days: The invention and impact of the Moog synthesizer*. Cambridge, MA: Harvard University Press.
- Pressing, J. (1988) 'Improvisation: Methods and models.' In Sloboda, J. A. (ed.) *Generative processes in music: The psychology of performance, improvisation, and composition*. Oxford: Oxford University Press, pp. 129-178.
- Rohrhuber, J., de Campo, A. and Wieser, R. (2005) 'Algorithms today notes on language design for just in time programming.' In Proceedings of the 2005 International Computer Music Conference.
- Rohrhuber, J., de Campo, A., Wieser, R., van Kampen, J.-K., Ho, E. and Hölzl, H. (2007) 'Purloined letters and distributed persons.' In Music in the Global Village Conference (Budapest). 2007.
- Rushkoff, D. (2010) *Program or be programmed: Ten commands for a digital age*. New York: Or Books.
- Russolo, L. (1986) 'The Art of Noises: Futurist Manifesto.' In Brown, B. (ed.) *The Art of Noises*. New York: Pendragon Press, pp. 23-30.
- Sanden, P. (2013) *Liveness in Modern Music. Musicians, Technology, and the Perception of Performance*. Routledge: New York.

- Sanfilippo, D. and Valle, A. (2013) 'Feedback Systems: An Analytical Framework.' *Computer Music Journal*, 37(2) pp. 12-27.
- Sawyer, R. K. (2003) *Group creativity: Music, theater, collaboration*. 10<sup>th</sup> ed., Mahwah, NJ: Lawrence Erlbaum Associates.
- Small, C. (1998) *Musicking: The meanings of performing and listening*. Hanover, NH: Wesleyan University Press.
- Stowell, D. and McLean, A. (2013) 'Live music-making: a rich open task requires a rich open interface.' *Music and Human-Computer Interaction*, pp. 139-142.
- Suchman, L. (1987) *Plans and situated actions: The problem of human-machine communication*. Cambridge: Cambridge university press.
- Suchman, L. (2007) *Human-machine reconfigurations: Plans and situated actions*. ed., : Cambridge University Press.
- Tammen, H. (no date) *Endangered Guitar*. [Online] [Accessed on 10<sup>th</sup> December 2015] <http://tammen.org/endangered-guitar/>
- Tammen, H. (no date) *Hybrid Interactive Instruments - thoughts on the Endangered Guitar*. . [Online] [Accessed on 2014] <http://tammen.org/berlin-almanac-on-contemporary-musical-instruments/>
- Tammen, H. (2014) *Conversation 2014*. Email to Jonas Hummel, 11<sup>th</sup> November.
- Turkle, S. and Papert, S. (1990) 'Epistemological Pluralism: Styles and Voices within the Computer Culture.' *Signs*, 16(1) pp. 128-157.
- Turner, V. W. (1969) *The Ritual Process: Structure and Anti-Structure*. 7 ed., Ithaca, NY: Cornell University Press.
- Ward, A., Rohrhuber, J., Olofsson, F., McLean, A., Griffiths, D., Collins, N. and Alexander, A. (2004) 'Live Algorithm Programming and a Temporary Organisation for its Promotion.' *In Proceedings of the README software art conference*. 289, Aarhus, Denmark, 2004.

- Waterman, E. and Dawn Smith, J. (2013) 'Listening Trust: The Everyday Politics of George Lewis's "Dream Team".' In Heble, A. and Wallace, R. (ed.) *People get ready: the future of jazz is now!*. London and Durham: Duke University Press.
- Wegerif, R. (2011), 'Towards a dialogic theory of how children learn to think', *Thinking Skills and Creativity*, 6 pp. 179-190.
- Wessel, D., Wright, M. and Schott, J. (2002) 'Intimate Musical Control of Computers with a Variety of Controllers and Gesture Mapping Metaphors.' In NIME 2002. Proceedings of the International Conference on New Interfaces for Musical Expression. 2002.
- White, E. C. (1987) *Kaironomia: On the will to invent*. Ithaca, NY and London: Cornell University Press.
- Wilson, G. B. and MacDonald, R. A. R. (2015) 'Musical choices during group free improvisation: A qualitative psychological investigation.' *Psychology of Music*, (15) pp. .
- Wilson, S., Cottle, D. and Collins, N. (2011) *The SuperCollider Book*. Boston: The MIT Press.
- Wilson, S., Lorway, N., Coull, R., Vasilakos, K. and Moyers, T. (2014) 'Free as in BEER: Some Explorations into Structured Improvisation Using Networked Live-Coding Systems.' *Computer Music Journal*, 38(1) pp. 54-64.
- Winthrop-Young, G. (2013). 'Cultural Techniques: Preliminary Remarks', *Theory, Culture & Society*, 30 pp. 3-19.
- Yee-King, M. and Collins, N. (2012) *Artist Statement, Network Music Festival*. [Online] [Accessed on 29<sup>th</sup> June 2016] <http://networkmusicfestival.org/nmf2012/programme/performances/wrongheaded>
- Young, M. (2016) 'Interactive and generative music: a quagmire for the musical analyst' In Emmerson, S. and Landy, L. (eds.) *Expanding the Horizon of Electroacoustic Music Analysis*. Cambridge: Cambridge University Press, pp. 58-79.