TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

EXPECTATION INCONGRUENCE IN MUSIC AND CODE READING: THE EYE-TRACKING APPROACH

Natalia Chitalkina

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. B OSA – TOM. 543 | HUMANIORA | TURKU 2021

TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

EXPECTATION INCONGRUENCE IN MUSIC AND CODE READING: THE EYE-TRACKING APPROACH

Natalia Chitalkina

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. B OSA – TOM. 543 | HUMANIORA | TURKU 2021

University of Turku

Faculty of Education
Department of Teacher Education
Doctoral programme on Learning, Teaching and Learning Environments Research (OPPI)

Supervised by

Adjunct Professor Marjaana Puurtinen
Department of Teacher Education
Faculty of Education
University of Turku, Finland

Professor Hans Gruber
Department of Educational Science
Faculty of Human Sciences
University of Regensburg, Germany

Associate Professor Roman Bednarik
School of Computing
Faculty of Science and Forestry
University of Eastern Finland

Reviewed by

Professor Markku Hannula
Department of Education
University of Helsinki, Finland

Professor Tamara van Gog
Department of Education
Utrecht University, The Netherlands

Opponent

Professor Tamara van Gog
Department of Education
Utrecht University, The Netherlands

The originality of this publication has been checked in accordance with the University of Turku quality assurance system using the Turnitin OriginalityCheck service.

ISBN 978-951-29-8515-9 (PDF)
ISSN 0082-6987 (Print)
ISSN 2343-3191 (Online)
Painosalama, Turku, Finland 2021

Dedicated to the beloved memory of my grandfather, Vladimir Chitalkin, who taught me to dream big and to my mother, Olga Chitalkina, who taught me to fight for my dreams.

UNIVERSITY OF TURKU
Faculty of Education
Department of Teacher Education
NATALIA CHITALKINA: Expectation incongruence in music and code reading: The eye-tracking approach
Doctoral Dissertation, 82 pp.
Doctoral Programme on Learning, Teaching and Learning Environments Research (OPPI)
August 2021

ABSTRACT

Humans create and use different kinds of languages in order to store, view, and convey various types of information. Natural languages, such as English, allow people to communicate with each other in everyday and professional contexts. In contrast, symbolic languages, such as Western music notation or programming languages, enable people to make use of technical devices like musical instruments or computers. Research on the eye movements of expert musicians and programmers has revealed certain similarities in how these symbolic languages are read: unlike text reading, experts read music and code with more regressive eye movements. The current dissertation is the first project that explores music and code reading together. It focuses on one of the aspects of music and code reading that is equally important for both symbolic languages—the skill of working with unexpected information from a notation. In music and programming, this skill is especially required in tasks such as handling surprising melodic patterns and debugging, respectively.

This dissertation had three main aims: (1) theoretical exploration of similarities and differences in creating expectations that help with pattern recognition in music and programming; (2) development (in music reading) and creation (in code reading) of research methodologies that can be applied in research on incongruent patterns; and (3) exploration of the cognitive processing of incongruent notation in experienced music readers and one experienced code reader.

Surprising elements in familiar patterns hamper their recognition. Article I presents a theoretical exploration of the similarities and differences in building expectations that allow pattern recognition in music and programming. The proposed prediction model, which serves as the solution to Aim 1, includes three components that are common for both music and programming: (1) knowledge of language systems, (2) knowledge of meaning, and (3) knowledge of context. In addition, it also contains two components that differ for music and programming: (4) translation of information and (5) temporal and motor requirements. Experiments presented in this dissertation can be considered to be the first steps toward looking at certain components of the proposed prediction model in detail when prediction works normally (congruent notation) and when prediction is violated (incongruent notation).

In order to study the reading of surprising incongruent patterns in music and code, special experimental settings, which provide the solution to Aim 2, were developed for music reading and created for code reading. Hence, the selected set-up for the music reading study was based on a prior study (Penttinen et al., 2015), where incongruences were introduced in the “Mary Had a Little Lamb” melody and

all music performances were temporarily controlled using a metronome. Experiment 1 developed this set-up by inserting incongruent notes into two different tonalities and by asking participants to play on a piano or sing from the notation. Thus, the music reading experiment focused on the first, second, fourth, and fifth components of the proposed model in music reading. It explored how the meaning of congruent and incongruent musical symbols is processed by experienced music readers. In addition, it also explored the translation of music information into two different performance ways (singing and playing piano) that have different motor requirements. Combining three different eye movement parameters allowed the researcher to describe different aspects of the cognitive processing of incongruent music reading—the temporal aspect, with the help of the eye-time span parameter (ETS), the cognitive effort aspect with the help of the mean pupil size parameter measured only in first-pass fixations, and the attention aspect with the help of the first-pass fixation duration. Experiment 2 on code reading was carefully designed on the basis of the music reading study. Consequently, incongruences were introduced in different parts of the familiar notation. The Bubble sort algorithm—a well-known sorting algorithm—was chosen as an analogue of the "Mary Had a Little Lamb" melody in programming. As in the music reading study, all code reading performances were temporarily controlled. The case code study provided some insights into the first and second components of the proposed model in programming by investigating how an experienced programmer reads sorting algorithms with and without surprising patterns. It particularly focuses on the phenomenon of an experienced reader overlooking the surprising pattern, which is considered to be the original one instead so-called proof-readers' error. In addition, this study explored the issue of the unit of code reading analysis by comparing two different options: lines and elements. The study introduced saccade velocity as a parameter of cognitive effort for the incongruent code reading analysis.

Research findings from these experimental studies provided the solution to Aim 3 and revealed that—in both music and code reading—incongruent patterns in the notation led to changes in fixation and cognitive effort parameters (pupil size and saccadic velocity). In contrast to code reading, strict temporal requirements for the processing of incongruence exist in music reading. The application of the eye-time span (ETS) parameter that describes the distance between the performer's gaze and musical time, allowed the researcher to investigate the temporal aspect of incongruence processing in the music reading experiment. Hence, experienced readers had longer ETS when they approached the incongruent part of the notation and shorter ETS when they were in the process of struggling with the incongruent part. In addition to incongruent reading, the difference in the performance mode of the same music task associated with the translation of information and motor requirements was studied in the music reading experiment by comparing singing and playing from music scores. Despite the fact that the participants played incongruent melodies better than they sang them, the analysis of eye movement parameters allowed the researcher to discover that singing might be less cognitively demanding than playing. These findings are discussed within the proposed theoretical model of prediction and associated expertise theories.

KEYWORDS: Music reading, code reading, expertise, eye tracking, incongruence.

TURUN YLIOPISTO
Kasvatustieteiden tiedekunta
Opettajankoulutuslaitos
NATALIA CHITALKINA: Odotusten vastaisten symbolien lukeminen: katseenseurantatutkimus nuotin- ja koodinluvusta
Väitöskirja, 82 s.
Oppimisen, opetuksen ja oppimisympäristöjen tutkimuksen tohtoriohjelma (OPPI)
heinäkuu 2021

TIIVISTELMÄ

Ihmiset luovat ja käyttävät erilaisia kieliä tallentaakseen, tarkastellakseen ja välittääkseen informaatiota. Luonnolliset kielet, kuten englanti, mahdollistavat ihmisten välisen kommunikaation arkisissa ja ammatillisissa tilanteissa. Sen sijaan symboliset kielet, kuten länsimainen nuottikirjoitus tai ohjelmointikielet, mahdollistavat erilaisten laitteiden, kuten soittimien tai tietokoneiden, operoinnin. Taitavien muusikkojen ja koodinlukijoiden silmänliikkeiden tutkimus on paljastanut joitakin samankaltaisuuksia siitä, miten näitä kahta symbolikieltä luetaan: toisin kuin tekstin lukemisessa, taitavat nuotin- ja koodinlukijat tekevät enemmän regressiivisiä, eli taaksepäin suuntautuvia silmänliikkeitä. Tämä väitöstutkimus on ensimmäinen tutkimushanke, jossa tarkastellaan nuotin- ja koodinlukua rinnakkain. Tutkimus keskittyy tiettyyn, molemmissa symbolikielissä tärkeään piirteeseen, eli taitoon selvitä lukemisen aikana notaatiossa havaittuun yllättävään informaatioon. Sekä musiikin että ohjelmoinnin aloilla tätä taitoa tarvitaan silloin, kun lukijan täytyy käsitellä yllättäviä melodisia kuvioita musiikkikappaletta lukiessaan tai etsiä virheitä koodista.

Tällä väitöstutkimuksella oli kolme päätavoitetta: (1) teoreettinen pohdinta nuotin- ja koodinluvun yhtäläisyyksistä ja eroavaisuuksista ja erityisesti siitä, miten taitavat lukijat muodostavat ennakko-oletuksia lukemastaan symbolien tunnistamisen helpottamiseksi; (2) menetelmien kehittäminen (nuotinluvussa) ja luominen (koodinluvussa) inkongruenttien, eli epäyhdenmukaisten, kuvioiden lukemisen tutkimukseen; ja (3) taitavien nuotinlukijoiden ja yhden taitavan koodinlukijan kognitiivisen prosessoinnin tutkiminen silloin, kun lukijat käsittelevät inkongruenttia informaatiota.

Yllättävät elementit tutussa visuaalisessa materiaalissa vaikeuttavat kyseessä olevan materiaalin prosessointia. Artikkelissa I pohditaan nuotin- ja koodinluvun teoreettisia eroja ja eroavaisuuksista ja sitä, miten taitavat lukijat muodostavat ennakko-oletuksia lukiessaan. Ehdotettu ennustusmalli, jonka avulla vastataan päätavoitteeseen 1, sisältää kolme molemmille symbolikielille yhteistä komponenttia: (1) tiedon kielijärjestelmästä, (2) tiedon merkityksestä ja (3) tiedon kontekstista. Tämän lisäksi malli sisältää kaksi komponenttia, joissa nuotin- ja koodinluku eroavat toisistaan: (4) informaation kääntäminen laitteelle ja (5) temporaaliset ja motoriset vaatimukset. Tässä väitöstutkimuksessa esiteltävät empiiriset osahankkeet olivat ensiaskelia ennustusmallin komponenttien tutkimuksessa. Kahdessa osahankkeessa tarkasteltiin yksityiskohtaisesti tilanteita, joissa ennakko-oletuksia voi hyödyntää tavalliseen tapaan (kongruentti notaatio) ja kun odotukset eivät toteudu (inkongruentti notaatio).

Tässä väitöstutkimuksessa kehitettiin koeasetelmia nuotin- ja koodinluvun aikaisten, yllättävien ja inkongruenttien kuvioiden lukemisen tutkimusta varten (päätavoite 2). Nuotinlukuaiheinen koeasetelma pohjautui aikaisempaan tutkimukseen (Penttinen et al., 2015), jossa epäyhdenmukaisuuksia sijoitettiin tuttuun ”Maijall’ oli karitsa” –melodiaan ja soittosuoritusten ajoitusta kontrolloitiin metronomin avulla. Osatutkimus 1 kehitti tätä asetelmaa edelleen esittämällä tutun kappaleen osallistujille kahdessa eri sävellajissa ja pyytämällä osallistujia toteuttamaan melodia kahdella eri tavalla, soittaen ja laulaen. Osatutkimus 1 keskittyi siis ennustusmallin ensimmäiseen, toiseen, neljänteen ja viidenteen komponenttiin nuotinlukemisen näkökulmasta. Osatutkimuksessa 1 tutkittiin kuinka taitavat nuotinlukijat prosessoivat kongruenttien ja inkongruenttien nuottisymbolien merkityksiä. Tämän lisäksi osahanke selvitti sitä, miten nuotti-informaatio käännettiin kahdelle motorisilta vaatimuksiltaan erilaiselle ”soittimelle” (pianonsoitto ja laulaminen). Kognitiivisia prosesseja inkongruentin materiaalin lukemisen aikana kuvailtiin kolmen eri silmänliikemuuttujan turvin: lukuprosessin ajallista etenemistä selvitettiin *eye-time span* –mittarin (ETS) avulla, kognitiivista työmäärää mittaamalla pupillin koon vaihtelua, ja fiksaatioiden kestot kertoivat huomion kohdistumisesta notaation eri osiin ensilukemisen aikana. Koodinlukuun keskittyvä osatutkimus 2 suunniteltiin osatutkimuksen 1 koeasetelman pohjalta ja epäyhdenmukaisuudet sijoitettiin jälleen tuttuun notaatioon. Koodinlukukokeessa nuotinlukukokeen ”Maijall’ oli karitsa” –melodian tilalle valittiin hyvin tunnettu kuplalajittelualgoritmi, ja myös koodinlukutehtävässä kontrolloitiin ajankäyttöä. Tapaustutkimuksessa selvitettiin, miten kokenut ohjelmoija luki lajittelualgoritmia silloin kun siinä joko oli tai ei ollut inkongruentteja kohtia. Näin voitiin tarkastella ennustusmallin ensimmäistä ja toista komponenttia koodinlukemisen näkökulmasta.

Tapaustutkimus keskittyi erityisesti tilanteeseen, jossa kokenut lukija ohitti yllättävät elementit notaatiossa ja tulkitsi inkongruentin algoritmin oikeaksi ja alkuperäiseksi (ns. *proof-readers’ error*). Tämän lisäksi osatutkimuksessa 2 testattiin koodinlukututkimuksiin sopivia analyysiyksiköitä vertaamalla kahta vaihtoehtoa, rivejä ja elementtejä, ja esiteltiin sakkadien nopeus kognitiivisen työmäärän tarkasteluun sopivana, koodinlukututkimuksille uutena mittarina.

Osatutkimusten 1 ja 2 perusteella vastattiin päätavoitteeseen 3. Osatutkimuksissa selvisi, että sekä nuotin- että koodinlukutilanteissa inkongruentit kohdat notaatiossa johtivat muutoksiin fiksaatio- ja kognitiivisen työmäärän mittareissa (pupillin koko ja sakkadin nopeus). Toisin kuin koodinluvussa, tiukat temporaaliset rajoitteet säätelevät inkongruenssin prosessointia nuotinluvun aikana. Tästä syystä nuotinlukukokeessa hyödynnettiin ETS-mittaria, joka kertoo katseen kohdan ja musiikillisen ajan välisestä etäisyydestä. Taitavien nuotinlukijoiden ETS oli pidempi, kun he lukiessaan lähestyivät inkongruenttia kohtaa, ja lyhyempi, kun he soittivat tätä samaista kohtaa. Inkongruentin kohdan lukemisen lisäksi tutkittiin kahta erilaista esitystapaa (laulaminen ja soittaminen), sillä esitystapa liittyy nuotti-informaation kääntämiseen oikeiksi motoriksiksi liikkeiksi. Vaikka osallistujat soittivat inkongruentit melodiat paremmin kuin he lauloivat ne, silmänliikkeiden tarkastelu osoitti että laulaminen saattoi silti olla osallistujille kognitiivisesti vähemmän vaativaa kuin soittaminen. Näitä havaintoja pohditaan väitöskirjatutkimuksessa esitetyn ennustusmallin sekä asiantuntijuusteorioiden valossa.

ASIASANAT: nuotinluku, koodinluku, asiantuntijuus, katseenseuranta, inkongruenssi

Acknowledgements

I want to thank all people who supported me on my PhD journey and contributed to my development as a better scientist. I am very grateful to my supervisors Adjunct Professor Marjaana Puurtinen, Professor Hans Gruber and Assistant Professor Roman Bednarik. Hans and Marjaana, thank you for giving me the opportunity to join the Reading music project that gave rise to this dissertation. Marjaana, thank you for your patience, positive attitude and help with scientific and practical aspects of my doctoral work. I really appreciate your feedback on my ideas and promotion of my scientific independence. Hans, thank you for your feedback, useful comments and suggestions on my work. Roman, thank you for taking risks and accepting a doctoral candidate from a different field. I appreciate our discussions and your guidance in the area that used to be unfamiliar to me. I am very grateful to Professor Tamara van Gog and Professor Markku Hannula for being reviewers for this dissertation and providing me with feedback and critical comments on the dissertation contents. I am honored to have Professor van Gog as my opponent.

I also want to thank doctoral candidates Linda Puppe and Manuel Längler for their peer-support of my writing of the dissertation text and their useful comments, suggestions and theory discussions. Your hard work on dissertations motivated me to progress faster with my own text. I wish you the best of luck with finalizing of your degrees and hope to be able to see your defenses. I would like to thank the members of the Reading music project for their friendly support and help in the beginning of my PhD studies. I wish to express special thanks to Dr. Anna-Kaisa Ylitalo who helped me with the data analysis for my music reading study and always found time to give advice and answer my questions. I am grateful to the members of Turku Eye Tracking Laboratories for their feedback and comments on my PhD work.

I also want to thank my former supervisors Dr. Thorsten Kolling, Dr. Jantina Hohmann and Professor Monika Knopf who influenced my development as a scientist and beginning eye-tracking researcher during my master studies at Goethe University in Frankfurt. I am grateful to my former supervisor Dr. Sergey Isaychev who influenced my interest to physiological sensors and taught me to record and analyze physiological signals during my studies at Lomonosov Moscow State University.

I would also like to express my deepest gratitude to my close friends Julia Konovalova and Anna Nikiforova. Julia, I want to thank you for your support in good and bad times during my PhD journey, for the opportunity to share with you my successes and failures and to hear your point of view and advice, for always making me feel welcomed in your home, for our travel adventures and for the opportunity to share with you some of my hobbies. Anna, I thank you for being by my side despite changing circumstances of our lives, for sharing with me your positive attitude towards life, for your words of motivation and support.

I am grateful to my father, Konstantin Chitalkin, for his love and support. Unfortunately, it is not always possible to say words of gratitude due to the finite nature of human life. However, this dissertation would not be possible without influence of my mother and grandfather. I deeply appreciate my mother, Olga Chitalkina, for teaching me how to work hard and start from the beginning in case of failures. I assume my mother could induce my interest to the dissertation topic by being the first person in my life who could both program and play music. I also want to express appreciation to my grandfather, Vladimir Chitalkin, who spent almost whole life working in aviation industry for being my source of inspiration and work engagement.

Turku, 19.06.2021
Natalia Chitalkina

Table of Contents

Tables

Figures

List of Original Publications

This dissertation is based on the following original articles:

I Chitalkina, N., Bednarik, R., Puurtinen, M., Gruber, H. (2019). Prediction as a prerequisite of skilled reading: The cases of source-code and music notation. In *Proceedings of the 19th Koli Calling international conference on computing education research* (pp. 1–9). New York, NY: Association for Computing Machinery. https://doi.org/10.1145/3364510.3364516

Chitalkina contributed to the conception of the paper and was responsible for the development of its theoretical position as well as for writing the article. Bednarik and Puurtinen contributed to the conception, writing, and editing of the paper. Gruber contributed to the revision of the manuscript.

II Chitalkina, N., Puurtinen, M., Gruber, H., & Bednarik, R. (2021). Handling of incongruences in music notation during singing or playing. *International Journal of Music Education, 39*(1), 18–38. https://doi.org/10.1177/0255761420944036

Chitalkina contributed to the conception and design of the study and was responsible for data collection, data analysis, and interpretation as well as for writing the article. Puurtinen contributed to the conception and design of the study, data analysis, and interpretation as well as to the writing and editing of the manuscript. Gruber and Bednarik contributed to the editing of the manuscript.

III Chitalkina, N., Bednarik, R., Puurtinen, M., & Gruber, H. (2020). When you ignore what you see: How to study proof-readers' error in pseudocode reading. In *Proceedings of the 12th ACM symposium on eye tracking research & applications* (pp. 1–5). New York, NY: Association for Computing Machinery. https://doi.org/10.1145/3379156.3391979

Chitalkina developed the idea for the topic of the paper, contributed to the conception and design of the study, and was responsible for data collection, data analysis, and writing the article. Bednarik and Puurtinen contributed to

the conception and design of the study, the interpretation of the findings as well as to the writing and editing of the paper. Gruber contributed to the revision of the manuscript.

In this dissertation, references to these publications are made using their Roman numerals. The original publications have been reproduced with the permission of the copyright holders.

1 Introduction

Humans create and use different kinds of languages in order to store, view, and convey various types of information. Natural languages, such as English or Finnish, allow people to communicate with each other in everyday and professional contexts so that they can live, work, teach, and learn. In contrast, some symbolic languages, such as Western music notation or programming languages, enable people to make use of technical devices like musical instruments or computers (Chitalkina, Bednarik, Puurtinen, & Gruber, 2019). Although when music students should study music notation is still a matter of debate in music education (Fautley, 2017), music reading skills are considered to be important for overall musical competence (Sloboda, 1978). Learning to read code is considered by some computer science educators to constitute a necessary foundation for learning how to write computer programs (Xie, Nelson, & Ko, 2018; Xie, Loksa, Nelson, Davidson, Dong, Kwik, Tan, Hwa, Li, & Ko, 2019). Interestingly, research on eye movements of expert musicians and programmers has revealed some similarities in the reading of these symbolic languages—unlike text reading, experts read music (Goolsby, 1994; Penttinen & Huovinen, 2011) and code (Busjahn, Bednarik, Begel, Crosby, Paterson, Schulte, Sharif, & Tamm, 2015) with more regressive eye movements. Taking into consideration their similar main function of giving commands to devices and certain similarities in cognitive processing while reading, it might be useful to try to study these different symbolic languages together as a next step in the development of symbolic language reading research. This joint research focuses on those aspects of music and code reading that are equally important for both symbolic languages. One of these aspects is the skill of working with unexpected information from a notation. In music and programming, this skill is especially required in tasks such as handling surprising melodic patterns and debugging. Therefore, the main focus of this dissertation is on expert processing of unexpected changes (so-called incongruences) in familiar music and code reading. Incongruent reading of music and code is studied here through careful analysis of eye movements, performance, and the cognitive parameters of experienced readers.

2 Symbolic Language Reading Expertise

Symbolic languages are used in different disciplines to represent field-specific meaning in non-textual format. For instance, mathematics uses the extremely specialized symbolic language to express numbers, equations, functions and formulas (Österholm, n.d.). Modern theoretical physics and mathematics are closely linked (Garber, 1999). However, Garber (1999) stresses that despite mathematics is the language of physics, theoretical physics is not mathematics. The mathematical language is used in physics to represent physical processes or a relationship between physical entities (Garber, 1999). In molecular biology, symbolic language includes, for instance, Greek letter symbols that are used to name components of enzymes or polypeptides building a protein (Gupthar, 2007). Letters and mathematical symbols are also used for the representation of genes and their properties (Gupthar, 2007). Chemistry uses a symbolic language to represent elements and compounds based on the use of ordinary letters associated with the Latin name of the elements (Childs, Markic, & Ryan, 2015). Therefore, symbolic languages in physics, chemistry and biology mainly represent natural processes. In contrast to them, symbolic notation in music represent properties of sounds usually produced by a musical instrument managed by a musician. In a similar way, symbolic notation in programming represent the way to get an output produced by a computer. Unlike mathematics, physics, chemistry or biology, devices are needed in order to get outputs or sounds from music (except for singing) and code notations. That makes programming languages, where the computation process is automatized, different from mathematics, where the symbolic language is used for manual computations.

This dissertation focuses on investigating experienced music and code reading while dealing with surprising patterns in a notation. Research on experienced reading can help novices evaluate their proficiency and understand what is needed to improve reading skills in their domain. However, different approaches exist for defining expertise in education research and, consequently, in music and computing education research. The current chapter provides an overview of four approaches that focus on aspects that are particularly important for the acquisition of symbolic language expertise.

Symbolic languages are always learned within a specific community (including online communities). Sociocultural theory proposes that natural language acquisition and development of verbal thinking are mediated by society and culture that a certain society represents (Vygotsky, 1934). Recently, Vygotsky's ideas were adopted for the acquisition of symbolic languages, such as mathematics (Berger, 2005). This theory considers symbolic language expertise to represent an acquisition of professional knowledge meaning in the form of professional concepts mediated by professional communities. Importantly, this acquisition is seen as a qualitative change—that is, as a change in the ways of meaning abstraction at different stages of language acquisition (Vygotsky, 1934). Novices might possess different meanings for professional categories determined by different ways of abstraction—that is, their ways of thinking differ from those of experts (Berger, 2005). Thus, development of language expertise not only changes the contents of professional knowledge but also ways of thinking. Although there is existing research on qualitative differences in terms of music (Penttinen, Huovinen, & Ylitalo, 2013) and code (Lister, Simon, Thompson, Whalley, & Prasad, 2006) notation comprehension among novices and experts, it has still not investigated the developmental aspect of professional thinking (Chitalkina et al., 2019).

The professional community that mediates the acquisition of both language expertise and professional thinking is not limited to formal tuition only; it also includes peers, colleagues, project members, and online community members. In this approach (Vygotsky, 1934), the difference in symbolic language expertise levels is seen as the difference in the access to the professional community and the available quality of support from its members (Berger, 2005; Chitalkina et al., 2019). In the music domain, support from professional community and family members plays an important role in the development of professional expertise (Gruber, Lehtinen, Palonen, & Degner, 2008; Längler, Nivala, Brouwer, & Gruber, 2021). However, it seems that support from peers, like band members, is more important for popular musicians (Längler, Nivala, & Gruber, 2018), whereas formal tuition and family are more important for classical musicians (Degner, Lehmann, & Gruber, 2003). In the computing domain, informal mentorship and accessibility to it were found to be surprisingly important for students, especially for underrepresented students, in terms of promoting their motivation and changing their beliefs about the subject (Ko & Davis, 2017; Ko, Hwa, Davis, & Yip, 2018).

In addition to having access to a related community, much practice is needed in order to acquire symbolic language expertise. However, not all kinds of practice are helpful for becoming an expert reader. The theory of expertise as a deliberate practice (Ericsson, Krampe, & Tesch-Römer, 1993; Ericsson, 2016) proposes that those who plan to reach an expert level of performance should participate in professional activities with a conscious goal to improve their performance. This approach also

places a special emphasis on the role of trainers and how trainers create learning instructions (Ericsson, 2018 a). Nevertheless, the deliberate practice approach tends to restrict the role of the community to formal tuition, which might not always be appropriate for programming. For instance, when programmers need to write code in order to solve a concrete computational problem, they might focus on successfully completing it before the deadline more than on obtaining a complete understanding of all code details that allow its successful implementation. Thus, they might use certain code snippets obtained from professional forums that they do not completely understand. Interestingly, different programming solutions for the same computational problem that can be found on programming websites are often rated based on whether they solve a problem rather than on whether their explanation or the expertise of the programmers who offered them is good.

For the acquisition of symbolic language expertise, it is also important to gain a solid understanding of the subfield for which the use of this language is intended and what its applications could be. According to the theory of knowledge encapsulation, expertise is based on continuously repeated handling of cases that are relevant to a specific domain (Boshuizen, Gruber, & Strasser, 2020). Expertise development is carried out in three stages: (1) knowledge accretion and validation, (2) knowledge encapsulation, and (3) formation of scripts (Boshuizen & Schmidt, 2008). Thus, working with a large number of relevant cases is what makes expert professional categories more detailed and easily accessible (Van de Wiel, Boshuizen, Schmidt, & Schaper, 1999). This theory proposes that not only is the time spent on professional activities important for becoming an expert but that this time should also be spent on relevant kinds of tasks and within a relevant professional subfield.

Research shows that knowledge of a musical genre and an application domain, which can be considered to be subfields within music and programming, are important for adequate interpretation of information from music and code notations (Ramalingam & Wiedenbeck, 1997; Puurtinen, Anto, Ylitalo, Huovinen, Gruber, Heinonen, & Turta, 2019). Thus, novice coders understand programs better when they know their application domains (Ramalingam & Wiedenbeck, 1997). Similarly, the same note symbol might be played with different rhythmic processing in different genres (Puurtinen et al., 2019).

The next important aspect of symbolic language expertise acquisition is the recognition of patterns in a notation. The theory of expertise as superior memory primarily focuses on memory and its role in the recognition of patterns in a notation by comparing expert and non-expert performances (Ericsson, 2018 b). The theory has its roots in research on chess playing expertise. Chase and Simon (1973) demonstrated that expert chess players can better remember figure positions from chess games than non-experts, however, this difference disappears when experts and non-experts have to memorize unstructured figure positions from non-game

situations. Therefore, the memory excellence of experts was observed only in professionally-related tasks. Chase and Simon (1973) assumed that the memory of professional players would be different from the memory of non-professionals, because experts store information about chess figures in a more complex way by keeping in mind chunks of associated figures in their long-term memory and can successfully recognize these patterns.

In music reading, Sloboda (1974) showed that experienced music readers could better recall patterns of notes from scores in comparison to unexperienced readers, which allowed them to continue their performances in an experimental situation in which they were only able to see several starting notes before the remainder of the notation became invisible. Expert programmers were also able to recall more lines of code from computer programs presented in random order (Adelson, 1981). Adelson (1981) assumed that experts, in comparison to novices, organize chunks of information in accordance with their semantics, whereas novices use syntax-based chunks.

The mental schema theory can be considered to represent a further extension of the superior memory approach in the domain of programming language expertise acquisition. This approach emphasizes that the knowledge of experts has a special structural organization—the so-called mental schemas of professional knowledge—in the form of generic professional concepts, which novices do not have (Détienne & Soloway, 1990). Programming plans and the rules of programming discourse are two components of generic professional concepts (Soloway, Ehrlich, Bonar, & Greenspan, 1982). The theory seems to split knowledge and cognitive processes that are needed to use it (Détienne & Soloway, 1990; Chitalkina et al., 2019). A further modification of the mental schema approach to programming expertise was that even experts need to use several comprehension approaches, depending on whether a particular code matches the schemas that they possess (Von Mayrhauser & Vans, 1995). Thus, if a code snippet matches the schema, top-down comprehension is used. In contrast, bottom-up comprehension is used to build a new mental representation.

The handling of unexpected changes in familiar patterns of music and code can also be considered to constitute a particular part of the pattern recognition issue, which is closely connected to symbolic language acquisition. Nevertheless, research on information recall cannot provide detailed insight into visual perception and information processing, which are also required for pattern recognition. Consequently, Chapter 3 presents the problem of incongruence in music and code notations as well as several traditional ways for studying visual perception. Chapter 4 introduces eye-tracking as a research method and discusses how it can be used to obtain insight into visual perception and information processing while reading music and code in general as well as how it can be used to study incongruent music and code notations.

3 Incongruence in Music and Code Reading

Expertise theories outlines in the previous chapter propose that symbolic language expertise acquisition affects cognitive processes—especially thinking, memory, and visual perception—and requires a lot of deliberate practice within a relevant community as well as familiarity with a subfield and its typical applications (Sloboda, 1974; Adelson, 1981; Ramalingam & Wiedenbeck, 1997; Längler et al., 2018; Puurtinen et al., 2019). In addition, expertise in symbolic language reading leads to better recognition of patterns in a notation (Sloboda, 1974; Adelson, 1981). However, improvements in pattern recognition create powerful expectations or predictions (in the current work these two terms are used interchangeably) that might damage performance in certain important tasks related to music and code reading (Sloboda, 1976; Hansen, Goldstone, & Lumsdaine, 2013). For instance, music students usually start to learn classical music and Western music notation. Western music notation is a universally used system among musicians from different countries (Puurtinen, 2018). Many of them could not perform their jobs without sufficient level of music reading skill (Sloboda, 1978). Some music educators consider reading of Western music notation as a prerequisite of musical learning, while others believe that pop, jazz and rock music does not rely on it (Fautley, 2017). Nevertheless, musicians need music reading skills in order to be better engaged in music community (Sloboda, 1978). As time progresses, music students might switch to performing in a different genre (for instance, from classical music to folk music) that has different conventions that define how music should be performed from music scores. Conventional music rules they became used to while learning classical music might not work in a different music context. Thus, familiar visual stimuli need to be treated differently. Knowledge on how experienced performers adapt their reading strategies while dealing with incongruent music notation can help students and music educators by shedding more light on how the switch from a familiar to an unknown context is done (Chitalkina, Puurtinen, Gruber, & Bednarik, 2021).

Another aspect of music reading with surprising patterns is that this task might be performed in different ways depending on the type of music instrument used or singing. Musicians do not need to translate information from music scores into the

language of keys or strings of the musical instrument while singing, which is quite similar to programming because programmers do not need to translate a program that is written as source code in one of the programming languages into machine code that is executed in the computer to get an output. Hence, the performance method can potentially affect how surprising patterns in a notation are dealt with. However, this issue has not previously been investigated.

In programming, coders spend much more time on debugging—that is, finding mistakes in computer programs—than on writing computer programs (Hansen et al., 2013). Debugging is one of core professional competences for programmers. They need to be able to independently find and fix errors in computer programs (Michaeli & Romeike, 2019). However, debugging can be twice as difficult as writing the program (Kernighan & Plauger, 1978). Indeed, even software developers spent between 20 and 40 percent of their working time on debugging (Perscheid, Siegmund, Taeumel, & Hirschfeld, 2017). One of the reasons for debugging difficulties might be the inability to notice an error in familiar patterns of code when programmers perceive a notation in accordance with their expectations and not in accordance with what is actually written on the screen (the so-called "proof-readers' error", see the description in Sloboda, 1978). Surprisingly, the processing of information while reading code with proof-readers' errors has never been studied before. To summarize, despite the fact that studying how musicians and programmers handle surprising patterns in a familiar notation has various practical applications, numerous knowledge gaps still exist in this research area.

Research on how musicians and programmers read music and code notations containing incongruent changes usually focuses on performance correctness and reaction time analysis. However, only performance correctness was analyzed in early music reading studies on incongruence, whereas incongruent code reading studies measured both parameters. In these music reading studies, experienced participants were asked to play music from misprinted notation and their performance correctness was analyzed (Sloboda, 1976; Wolf, 1976). Surprisingly, it turned out that participants often ignored these errors and played in accordance with their expectations of the music instead (Sloboda, 1976; Wolf, 1976). In a recent study (Brodsky & Kessler, 2017), researchers focused on investigating the time that is needed to comprehend congruent and incongruent notation. In contrast to earlier studies, where participants were asked to perform complete music pieces from notation, in the reaction time approach, the participants were required to read and react to short musical probes, that is, groups of notes. Musicians and non-musicians were asked to judge the direction of the notes or the line that connected a group of notes, which were either congruent or incongruent. Interestingly, musicians were more influenced by the direction of the notes than non-musicians while attending to the lines that connected a group of notes.

Similar to Sloboda's (1974) study on the perceptual span of musicians, which actually measured their expectations about the continuation of a musical phrase (see Section 2), Soloway and Ehrlich (1984) were the first to reveal that experienced programmers are strongly influenced by their expectations. Music studies on incongruence are similar to incongruent code reading studies in terms of the main parameters of analysis—that is, reaction time and correctness (Soloway & Ehrlich, 1984; Hansen et al., 2013; Ajami, Woodbridge, & Feitelson, 2019). Typically, in research on code incongruence, scholars create two versions of the same program with expectation congruent and incongruent code and present each participant with one of these versions (Soloway & Ehrlich, 1984; Hansen et al., 2013). Participants usually do not know the purpose of the code. However, experimental tasks used in early studies differ from the tasks used in recent ones.

In one of the first studies on code incongruence (Soloway & Ehrlich, 1984), novice and expert participants were asked to fill in a blank line in Pascal codes that were either congruent or incongruent with the rules of programming discourse. Interestingly, although the incongruent versions of programs induced more errors and required more time for participants to respond correctly, no interaction between expertise and program version was found. Even in incongruent programs, experts performed better than novices. In a second study (Soloway & Ehrlich, 1984), experts were asked to recall programs that were either congruent or incongruent with the rules of programming discourse. This study provided evidence that experts recall congruent programs better.

In recent studies on code incongruence, participants are typically asked to find out what the output of code snippets is. In this vein, Hansen et al. (2013) asked experienced programmers to find the output of basic Python programs, where each program type had two or three versions with notational changes. The study revealed that experience helped experts avoid the kinds of errors that are common for real programs; however, it could also disrupt performance when basic expectations about code statements were violated (such as, for instance, using + for addition and then, immediately after, for string concatenation). Ajami et al. (2019) studied how three different factors influence code complexity and its understanding: different control structures (if and for loops), different forms of conditionals, and violation of programming idioms in for loops. They compared a traditional for loop with a for loop that does not cover the full conventional range as well as a for loop counting up with a for loop counting down. For loops that violated programmers' expectations led to significantly more errors; this, however, did not lead to significant differences in processing time. Interestingly, it was also found that for loops are significantly more complicated than if loops.

A special place take studies by Casalnuovo, Lee, Wang, Devanbu, & Morgan. (2020), where participants were asked to choose a preferred snippet out of two

snippets of Java code with the same meaning written in both predictable and unpredictable forms in order to test whether language models align with programmers' preferences. These studies have shown that programmers have a preference for predictable code and that the language model is able to capture preferences for some conventions. Interestingly, studies also revealed that certain preferences, such as the removal of nonessential parentheses, are not as strong as others.

Overall, research on incongruence in music and code notations, which used reaction time and the performance correctness approach, showed that incongruent notations lead to processing difficulties. However, reaction time and performance correctness analysis cannot provide deeper insights into the course of information processing while reading, which the use of the eye movement analysis can provide.

4 Eye Movements and Incongruent Reading

4.1 Eye-Tracking as a Method for Studying Symbolic Language Reading

Eye-tracking is a method for studying eye movements and where humans look in different contexts. This method is used to study text reading, visual search, scene perception (Rayner, 2009), music reading (Puurtinen, 2018), code reading (Obaidellah, Al Haek, & Cheng, 2018; Sharafi, Sharif, Guéhéneuc, Begel, Bednarik, & Crosby, 2020), and many other activities. An eye tracker contains cameras that record gaze positions by tracking the reflection of near-infrared light that is emitted toward the pupil and the cornea. Two main parameters of interest in eye movement research are fixations and saccades (Figures 1 and 2). During fixations, a steady retinal image of the thing of interest is maintained and ocular drifts are suppressed (Komogortsev, Gobert, Jayarathna, & Gowda, 2010), while saccades refer to the quick movements between two fixations.

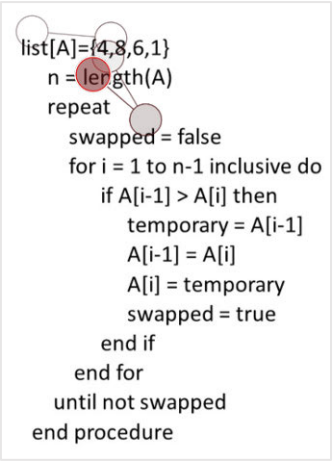


Figure 1. Example of first eye movements (circles = fixations, lines between circles = saccades) during expert reading of the Bubble sort algorithm. Data from experiment 2 (see chapter 6.3), extracted from Tobii Pro Lab.

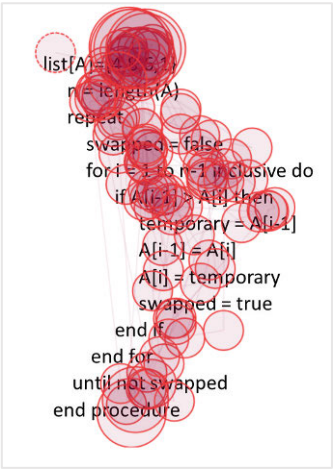


Figure 2. All eye movements (circles = fixations, the size of a circle = fixation duration, lines between circles = saccades) during expert reading of the Bubble sort algorithm for one minute. Data from experiment 2 (see chapter 6.3), extracted from Tobii Pro Lab.

The problem of how fixations and saccades should be separated from raw gaze data—the so-called event detection—remains an acute issue in eye-tracking research. Different approaches to the problem include the application of various velocity-based or dispersion-based detection algorithms as well as machine learning algorithms such as random forest (Zemblys, Niehorster, Komogortsev, & Holmqvist, 2017). Currently, no standard approach to the issue exist, although many eye-tracking companies offer a fixation filter that is based on the Velocity-Threshold Identification (I-VT) algorithm as a default. The chosen approach can affect the research findings in cognitive studies. In the I-VT fixation filter, a velocity value that is calculated for each gaze position sample is compared to the threshold—if it is lower than the threshold, a sample is considered to be a part of a fixation; otherwise, a sample is considered to be a part of a saccade (Komogortsev et al., 2010; Olsen, 2012).

A wide variety of eye movement parameters associated with fixations and saccades, such as the number and duration of fixations, velocity, length, and direction of saccades, are analyzed in eye-tracking research. The current dissertation project focuses on various fixation parameters, such as total fixation duration, number of fixations, duration of first-pass fixations (all fixations on the area until the first exit from it), as well as saccade velocity parameters. Fixation parameters can provide insights about attention (Just & Carpenter, 1980) and its distribution across a task as well as about the processing of visual information. In this dissertation, fixation parameters are applied in both music and code reading tasks. In addition, eye trackers allow pupil size to be measured, which is considered to be a parameter related to

emotional arousal and cognitive effort (Holmqvist, Nyström, Andersson, Dewhurst, Jarodzka, & Van de Weijer, 2011). Pupil dilation was found to be a useful cognitive workload parameter in education research, especially for young adults (Van Gerven, Paas, van Merrienboer, & Schmidt, 2004). In this dissertation, the pupil size parameter is used to assess cognitive effort in music reading tasks. However, two serious confounds are associated with the pupil size parameter: the pupillary light reflex and pupil foreshortening error (Hayes & Petrov, 2016). The first confound is linked with the luminance of the laboratory and experimental stimuli. The second is associated with changes in gaze position producing foreshortening of the pupillary image (Hayes & Petrov, 2016). In addition, changes in the pupil size parameter following a look at experimental stimuli occur with a delay (Katidioti, Borst, Bierens de Haan, Pepping, van Vugt, & Taatgen, 2016). In music, reading time is limited, because every note should be performed in line with musical tempo of the piece (Puurtinen, 2018; Chitalkina et al., 2019). Eyes of musicians are forced to move in accordance with the progress of the musical piece. Therefore, a registered change in the pupil size parameter can be attributed to the previous part of the melody. Presenting experimental melodies at the same location in the center of the screen makes it possible to control for the pupil foreshortening error while comparing pupil sizes in the same parts of melodies to each other, since they are at the same angle from the eye-tracker. In code reading, however, reading time is typically unlimited, and eye movements are not restricted by requirements of execution (Chitalkina et al., 2019). Thus, the eyes of a programmer might move to different locations from the same point of interest in code making difficult to control for the pupil foreshortening error and assume what in the code causes the change in the pupil size parameter. In this case, saccade velocity parameters can be a convenient alternative to the pupil size parameter, since they can also signalize the change in mental workload (or cognitive effort) that is needed in order to proceed with a task (Di Stasi, Antolí, & Cañas, 2011). In the current dissertation, saccade velocity parameters are used to assess cognitive effort in code reading tasks to avoid the above-mentioned issues with the application of the pupil size parameter in code reading tasks.

Finally, eye-tracking data provides an opportunity for creating additional mixed parameters that merge eye movements with other parameters that are important for a task in question, such as the eye-hand span that links eye movements and motor activation (Penttinen, Huovinen, & Ylitalo, 2015). The eye-hand span can, for instance, provide information about eye-hand coordination in tasks for which it is important. In music, temporal dimension is particularly important (Puurtinen, 2018) because musicians have to maintain a steady tempo during the entire music performance. Hence, this dissertation focuses on the application of the eye-time span parameter (Huovinen, Ylitalo, & Puurtinen, 2018) in music reading tasks, which is a mixed measure of the difference between the music time of the first fixation on the

note of interest and the music time at which it should be performed according to the music tempo of the piece. In eye movement research, eye movement parameters are usually calculated for notation excerpts—the so-called areas of interest. The length of these areas is defined by researchers in accordance with their research questions.

4.2 Eye-Tracking Studies on Music and Code Reading

Although the first eye-tracking studies on music reading appeared as early as the first half of the twentieth century (Jacobsen, 1928; Weaver, 1943) and the first study on code reading appeared in 1990 (Crosby & Stelovsky, 1990), eye-tracking was not among the most popular research methods in music (Penttinen, 2013; Puurtinen, 2018) and computing (Obaidellah et al., 2018) education until the end of the first decade of the twenty-first century.

Several similar lines of eye-tracking research can be identified in both music and code reading (Table 1). The first line focuses on the search for similarities and differences between natural and symbolic languages. For educational science, this line of research provides knowledge on whether the same learning strategies and instructions could be applied to teach both of them in case that their processing is similar. However, eye-tracking research also allows important processing differences to be identified between symbolic and natural languages.

Table 1. Similar Lines of Eye-Tracking Studies on Music and Code Reading.

Type of Research	Music Reading	Code Reading
Comparison to prose reading	More regressive eye movements and longer fixation durations in comparison to prose reading.	
Role of expertise (experts vs. novices)	The level of expertise is reflected in different distribution of attention on a notation. Experts fixate on more relevant areas of a notation.	
Task performance of participants with the same expertise level	Good and poor performers distribute attention onto a notation in different ways.	

Several studies pointed out that source code reading is fundamentally different from prose reading (Crosby & Stelovsky, 1990; Busjahn, Schulte, & Busjahn, 2011; Binkley, Davis, Lawrie, Maletic, Morrell, & Sharif, 2013; Busjahn et al. 2015; Jbara & Feitelson, 2017; Peachock, Iovino, & Sharif, 2017). The difference is reflected in eye movement behavior. Indeed, the mean fixation on a code element was found to be longer than the mean fixation on natural language, and programmers were also found to have more regressions while reading source code in comparison to natural

language (Busjahn et al., 2011). Code reading was found to be non-linear in general (Busjahn et al., 2015; Jbara & Feitelson, 2017). Moreover, studies that compared code and text reading revealed that even participants with low programming experience read code less linearly than text (Busjahn et al., 2015; Peachock et al., 2017). Both novices and non-novices read source code less linearly than text in a natural language (Peachock et al., 2017) and experts read source code less linearly than novices (Busjahn et al., 2015).

Among research on music reading, only one study directly compared syntactic incongruences in text and music reading (Ahken, Comeau, Hébert, & Balasubramaniam, 2012). In music, incongruences violated tonality; whereas, in text, incongruences disrupted grammar. The study revealed several similarities of eye movement behavior through two tasks: an increase in the mean proportion and duration of fixations on target areas. Other eye-tracking studies of music reading (Goolsby, 1994a, 1994b) did not compare music reading and text reading directly—instead, they compared music reading studies with other text reading studies. Important differences in information processing between these domains were revealed. In contrast to text reading, skilled music readers have more regressions and shorter saccadic lengths. Another important difference between the two domains was found in terms of the duration of fixations, which is longer for music reading.

To summarize, this line of eye-tracking research revealed interesting similarities between these two different symbolic languages. First, the reading of symbolic languages is conducted with more regressive eye movements in comparison to prose reading. Second, in both music and code reading, the processing of information required longer fixation durations than in prose reading. Therefore, due to the similarities in information processing mechanisms and differences to text reading, it might be promising to study different symbolic languages together.

The second line of eye-tracking research on music and code reading focuses on the role of expertise in music and programming and on the differences in information processing between experts and non-experts when performing different kinds of tasks. However, there is still a lack of longitudinal studies that explore the development of reading skills at different stages of formal tuition (with the only exception being the 2011 study by Penttinen and Huovinen).

In music reading studies, experts and non-experts differ in their ability to access fingering, that is, numbers from 1 to 5 added to music notation in order to indicate the finger that should perform a specific note (Drei-Zerbib, Baccino, & Bigand, 2012). Furthermore, the eye-hand spans and eye-time spans of experts were greater than the eye-hand spans and eye-time spans of amateurs (Furneaux & Land, 1999; Penttinen et al., 2015; Huovinen et al., 2018; Sheridan, Maturi, K. S., & Kleinsmith, 2020) and experts operated with shorter fixation durations than amateurs (Penttinen et al., 2015). A special place in this group of studies is taken by Penttinen and

Huovinen's (2011) study, which examined the development of sight-reading skills (reading of music that musicians have not read and played before) in novices at three different stages of their music course in comparison to a group that included amateur musicians. In contrast to novices, experts were found to be sensitive to the metrical division (how the notes are organized into bars) of the music piece, which was reflected in an increase in the total fixation duration toward the end of every bar. However, the allocation of fixation time by novices was closer to the allocation of amateurs as their music course progressed. In addition, the development of sight-reading skills in novices was reflected in dealing with the intervallic skip (a low-pitch note was followed with a high-pitch note, or vice versa)—the first-pass fixation duration increased for the first note before the skip and decreased for the first note after the skip.

The first eye-tracking study of programming tried to shed light on whether it is possible to identify scanning patterns (a special order of eye movements) associated with expertise (Crosby & Stelovsky, 1990). In this case study, two experts were compared to two novices. The study revealed a difficulty in identifying scanning patterns associated with programming expertise—that is, the same types of scanning patterns were found in both experts and non-experts. Nevertheless, two most differing patterns were also found in experts. In addition, it was found that, on average, experts spent more time on relevant areas and less time on comments (Crosby & Stelovsky, 1990). However, code-oriented and comment-oriented participants were found in both experts and novices (Crosby & Stelovsky, 1990).

Another study revealed that novices are less successful than experts in discriminating between areas of the program and that they do not use beacons, that is, stereotypical segments of code (Crosby, Scholtz, & Wiedenbeck, 2002). Several studies investigated how expertise influences code reading for the purpose of debugging by studying both source code and its representations. They reported that experts and non-experts use different debugging strategies, which is also reflected in how they use code representations (Romero, du Boulay, Cox, & Lutz, 2003; Bednarik, 2012; Hejmady & Narayanan, 2012). While debugging experts focused on systematically relating the code to the output, novices relied on both code and graphical representation (Bednarik, 2012). Experienced programmers tended to look at the output more often than those less experienced at the beginning of the debugging session (Hejmady & Narayanan, 2012). Another group of research studies (Uwano, Nakamura, Monden, & Matsumoto, 2006; Sharif, Falcone, & Maletic, 2012) focused on analyzing scan patterns for code review purposes—that is, on when a programmer reads the entire code before focusing on its details. It seems that experts have a different attention focus than novices and are able to associate special lines of code with potential problems (Sharif et al., 2012). Experts actively use information from extrafoveal vision (the region in the visual field, where acuity is

not so good; see Rayner, 1998) during code reading in comparison to non-experts (Orlov & Bednarik, 2017). When experts were deprived of extrafoveal vision, it not only resulted in a drop in their performance, but it also increased the duration of their fixations, similar to novices (Orlov & Bednarik, 2017).

Overall, this line of eye-tracking research demonstrates that experts and non-experts distribute their attention to music and code notations in different ways. In particular, experts fixate more on relevant areas of a notation.

The third line of eye-tracking research on music (Puurtinen, 2018) and code reading provides insight into the performance of different musical and programming tasks among readers with more or less equal level of expertise. Some studies compare successful and unsuccessful performances of the same task, while others analyze successful performances only.

In eye-tracking studies of music reading, researchers sometimes divide groups of good and poor readers not on the basis of their performance during a task but on how well the participants with more or less equal level of expertise performed in preliminary tests on either sight-reading (Gilman & Underwood, 2003) or music skill (Goolsby, 1994a, 1994b) and compare their subsequent performance in music reading tasks. Thus, Goolsby (1994 a, 1994b) described general processing differences between skilled and less skilled readers (and those at almost equal level of expertise) in sight-singing tasks. Skilled music readers were found to look ahead of the point of performance (Goolsby, 1994a). In addition, less skilled readers used longer fixations and processed melodies note-by-note, whereas skilled readers looked further ahead of the point of performance and had more regressions (Goolsby, 1994b). Gilman and Underwood (2003) analyzed the differences between good and poor sight-readers in sight-reading, transposition, and error-detection tasks. The study revealed that good sight-readers made fewer fixations in the sight-reading task that poor sight-readers and fewer fixations but with longer duration in the transposition task (the musicians had to perform the piece in a different key than the one it was written in). Interestingly, only good sight-readers were influenced by the task type, having longer fixations in the transposition task than in both sight-reading and error detection tasks. Another group of studies focused on how musical complexity and certain other factors influence the eye-hand span or eye-time span of experienced musicians (Wurtz, Mueri, & Wiesendanger, 2009; Rosemann, Altenmüller, & Fahle, 2016; Huovinen et al., 2018). Complexity of music notation was shown to have an impact on both the eye-hand span and eye-time span. Thus, violinists had lower eye-hand spans, longer fixation duration, and more regressive fixations when they had to perform more complicated music pieces (Wurtz et al., 2009). The eye-hand span significantly changed in accordance with the playing tempo and complexity of the music (Rosemann et al., 2016). A study of the eye-time span revealed that experienced readers approached upcoming complexities in music

notation with longer eye-time spans that were also affected by the playing tempo (Huovinen et al., 2018).

In eye-tracking studies of code reading, successful and unsuccessful performances of programmers were studied on different types of tasks, including debugging and reviewing of code with defects. It was found that programmers with correct and incorrect comprehensions have different attention focuses (Aschwanden & Crosby, 2006). Students with high performance tended to trace programs in accordance with their logic, whereas students with low performance tended to trace them line-by-line (Lin, Wu, Hou, Lin, Yang, & Chang, 2015). Programmers with poor debugging performance switched their attention between code, static, and dynamic representations, whereas programmers with good performance focused on code and dynamic representations (Hejmady & Narayanan, 2012). Intermediate students, who did not spend enough time to initially scan the code, tended to take more time to find defects during code review tasks (Uwano et al., 2006).

This line of eye-tracking research provided insights into what skilled performance is in relation to specific tasks associated with music and code reading. Research in both domains pays special attention to the notation reading of different complexity and error detection tasks. Eye-tracking studies on incongruent music and code notations also belong to this line of research. They can be thought of as special cases of research on notation with different complexity levels. In addition, the tasks of incongruent code reading are, to some degree, similar to the tasks of debugging.

However, several lines of eye-tracking research in music and code reading are different from each other (Table 2). In code reading, a growing number of studies apply machine learning algorithms on eye movement data to predict either programmer expertise (Lee, Hooshyar, Ji, Nam, & Lim, 2018; Ahsan & Obaidellah, 2020) or task difficulty (Lee et al., 2018; Fritz, Begel, Müller, Yigit-elliott, & Züger, 2014). Ahsan & Obaidellah (2020) studied whether three different machine learning algorithms (Decision Tree, Support Vector Machine and K-nearest Neighbors) were able to predict expertise differences in undergraduate computer science students. The study revealed that the decision tree was the most accurate classifier of expertise differences. Lee and colleagues (2018) demonstrated that Support Vector Machine algorithm was accurate to predict programmer expertise and less accurate to predict task difficulty from eye movement data. Fritz et al. (2014) assessed performance characteristics of the Naïve Bayes algorithm to predict task difficulty from eye movement data. Currently, no attempts of applying machine learning algorithms to predict either musician expertise or music reading task difficulty from eye movement data were done in music reading research.

Table 2. Different Lines of Eye-Tracking Studies on Music and Code Reading.

Type of Research	Music reading	Code reading
Machine learning to predict expertise or task difficulty from eye movements	Does not exist at the moment.	Algorithms can predict expertise more accurately than task difficulty
Eye movement modelling examples	One proposal to develop EMME to improve choral singing. No research on whether EMME can improve music reading for instrumental performance	Studies investigate the influence of EMME on problem solving and debugging as well as different ways to instruct expert before making EMME recording

The second line of eye-tracking research that is at the moment different in music and code reading is using eye movement data as a teaching tool. Eye-tracking provides an opportunity to create eye movement modeling examples (EMME)—that is, recordings of eye movements of expert readers along with an explanation of the task solution (Jarodzka, Van Gog, Dorr, Scheiter, & Gerjets, 2013; Bednarik, Schulte, Budde, Heinemann, & Vrzakova, 2018). In code reading, this is a rapidly developing line of research (Bednarik et al., 2018; Emhardt, Jarodzka, Drumm, van Gog, & Brand-Gruwel, 2020). Bednarik and colleagues (2018) showed that the use of EMME could improve problem solving while code reading. Emhardt and colleagues (2020) studied how two different ways of expert instruction (natural and didactic) in EMME influenced student learning of debugging. In contrast, little has been done to develop and study efficacy of EMME in music reading. At the moment only one experimental proposal on the development of EMME as a tool to improve choral singing was released in music reading field (Timoshenko, 2018). The development of effective EMME to improve music reading for instrumental performance can become a promising issue for the future of educational research in the field. The current work aims to provide more evidence for designing effective EMME tasks in music and code reading.

4.3 Eye-Tracking Research on Incongruence in Music and Code Reading

Eye movement studies on incongruent music reading focus on how musicians deal with unexpected alterations in either familiar (Penttinen et al., 2015) or unfamiliar (Ahken et al., 2012; Hadley, Sturt, Eerola, & Pickering, 2018) music notations. Penttinen and colleagues (2015) compared temporally controlled (a metronome was used to signal the tempo of the performance) amateur and expert performances of familiar songs both with and without incongruent bars. Expertise-related differences in eye movement behavior vanished when participants performed incongruent bars.

Thus, Penttinen et al. (2015) showed that, despite experts having shorter average fixation durations and using longer eye-hand spans, these differences disappear while performing from incongruent parts of music scores.

Ahken et al. (2012) demonstrated that incongruences related to musical tonality influence eye movement parameters, causing an increase in the mean proportion and duration of fixations. Hadley et al. (2018) showed that incongruence in one of the bars causes processing difficulties in experienced pianists—that is, an increase in total duration of fixations, a decrease in first-pass fixation duration, and an increase in the mean pupil size in the bar after the incongruent one. However, the temporal aspect of incongruent music reading—i.e., how participants read incongruent and surrounding notes in relation to musical time—has not been studied with the use of the eye-time span parameter. Previously, this aspect has been studied with the use of the eye-hand span parameter (Penttinen et al., 2015).

At the moment, although debugging is a popular topic in eye-tracking studies of code reading, there is still a lack of eye-tracking studies on incongruent code reading. Only two aspects of congruence have been studied with the use of eye movement registration thus far. Jbara and Feitelson (2017) studied how intermediate students read regular code—that is, code with repetitions of the same pattern. The study showed that repeated patterns are easier to understand because of their predictability: programmers spent more effort on initial repetitions than on successive repetitions. Eiroa-Lledo, Bechtel, Daskas, Foster, Pirzadeh, Rodeghiero, & Linstead (2020) investigated the incongruence of code and its corresponding comments. A small-sample eye-tracking study (Eiroa-Lledo et al., 2020) with beginning (up to four years of programming experience, no industry experience) and expert programmers revealed that when code snippets contained misleading comments about their contents, the experts seemed to tend to rely on them more than novices. Interestingly, experts also spent more time on code comments than beginning programmers. This finding was not, however, in line with the case study of Crosby & Stelovsky (1990), who found code-oriented and comment-oriented participants among both experts and novices. Therefore, it is possible that the sample of Eiroa-Lledo et al. (2020) included more comment-oriented experts than comment-oriented novices. Nevertheless, these findings need to be replicated in future research.

To summarize, eye-tracking studies demonstrated that predictable congruent music and code notations are easier to read, providing deeper insights into what cognitive processing of congruent and incongruent music and code (or code with comments) looks like.

5 Main Aims

As can be seen from the previous sections, music and code reading have never been studied together. Despite some existing research on incongruence in music and code reading, it exists separately (Ahken et al., 2012; Hansen et al., 2013; Penttinen et al., 2015; Hadley et al., 2018; Ajami et al., 2019; Eiroa-Lledo et al., 2020). There is no theory on how expectations are created in symbolic language reading and what similarities and differences can be found between these expectations in music and programming. In addition, despite the fact that eye-tracking is considered to be a suitable tool for gaining knowledge about cognitive processing during incongruent music and code reading (Penttinen et al., 2015; Hadley et al., 2018; Eiroa-Lledo et al., 2020), there is still a lack of eye-tracking studies on incongruence in music and code reading, which also leads to gaps in the development of research methodologies. Indeed, the current studies on incongruent music reading have created several paradigms through their investigation of incongruence in either familiar (Penttinen et al., 2015) or unfamiliar notations (Ahken et al., 2012; Hadley et al., 2018)—but no one has tried to develop these paradigms in future studies. Surprisingly, incongruent code reading is still primarily studied using the performance correctness and reaction time approach (Hansen et al., 2013; Ajami et al., 2019). Consequently, it is clear that there is a need for creating a research methodology for eye-tracking studies on incongruent code reading. This methodology can be adapted from another symbolic language on which it has previously been tested—music. Furthermore, little is known about the cognitive processing of incongruence in symbolic languages.

The current dissertation aims to cover these three identified gaps (Table 3). The general focus of this research project is on the reading of symbolic languages—that is, music and code notations. In particular, the dissertation places a special emphasis on the eye movements of experienced participants while reading surprising (incongruent) patterns in familiar music and code notations. The main aims of the dissertation project could be divided into the following two groups:

Table 3. Open Questions in Eye-Tracking Studies on Incongruent Reading.

Open Questions	Music	Code
Theoretical background	No theory on how expectations are created	
Research methodologies	Two approaches—studying incongruence in either familiar or unfamiliar music. No further development of these approaches.	Do not exist. Only two eye-tracking studies on related, but different topics.
Cognitive processing of incongruent notation	Incongruent notation causes changes in fixation and pupil size parameters. However, the temporal aspect of incongruent notation processing remains unknown.	Not known.

Both predictable music and code are easier to comprehend and perform or find an output for than unpredictable ones (Hansen et al., 2013; Penttinen et al., 2015; Jbara & Feitelson, 2017; Hadley et al., 2018; Ajami et al., 2019). Nevertheless, studies have not discussed how expectations about a notation are created nor what kinds of factors are important for the development of pattern recognition in music and code reading. **Therefore, the first aim of this project is theoretical exploration of similarities and differences in creating expectations that assist pattern recognition in music and programming.** The theoretical study represented in Article I explores the similarities and differences in how expectations are created in music and programming.

The second aim of this project is to develop (in music reading) and create (in code reading) research methodologies that can be applied in future research on incongruent patterns. The music reading experiment in Article II conducted for this project extends the study conducted by Penttinen et al. (2015) by introducing surprising patterns in two different tonalities and asking participants to perform from a notation in two different ways—by playing on a piano or by singing. This is the first study in the music reading field that investigated two different performance modes of the same musical task. In accordance with Penttinen et al. (2015), incongruent changes were introduced in a familiar melody, and all performances were temporally controlled with the help of a metronome. The experiment suggested improvements in the application of eye movement parameters for studying incongruent music reading. Thus, this study pays special attention to the temporal aspect of incongruent notation processing using the eye-time span parameter, which helps evaluate the distance between the first look at a specific part of a notation and the musical time at which it should be performed. In addition, the study uses the pupil size parameter as a measure of the

cognitive effort spent on the performance from an incongruent notation. In contrast to previous studies (Hadley et al., 2018), the mean pupil size was measured using the pupil size parameter for first-pass fixations only because first-pass fixations were found to reflect expertise during the processing of complex patterns in music notations (Penttinen & Huovinen, 2011).

A similar experimental setting was created for the code reading experiment in Article III. In contrast to previous research (Hansen et al., 2013; Ajami et al., 2019), where each participant was presented with one of two versions (congruent or incongruent, usually of a different length) of the same program—leading to the same output—and without knowledge of the purpose of the code, in the code reading case experiment developed for this dissertation, the participant read algorithms with incongruent changes in different parts of the code notation that led to different outputs and knew the purpose of the algorithm. Thus, by designing the task to have a known code purpose and surprising patterns that produce a different meaning and lead to a different output, the task we assigned the participant more closely resembled the real task of debugging. This design also allowed the researcher to use code snippets of the same length. In contrast to previous research (Hansen et al., 2013; Ajami et al., 2019), to reduce any advantages that some participants might have as a result of their experience with a specific programming language, pseudocode was used in order to maximally avoid any language-specific syntactic details. The case study investigated two different options (element or line) for a suitable unit of analysis in code reading that could also provide insights into computational thinking. The experiment explored the use of saccadic velocity parameters in order to study incongruent pseudocode reading. Hence, this study is the first study in the field of code reading that uses two saccadic velocity parameters to obtain additional information on cognitive effort during incongruent pseudocode processing. These parameters allow to avoid limitations associated with the application of the pupil size parameter in code reading tasks described in chapter 4.1.

To summarize, both research methodologies suggest placing incongruent elements into familiar notation because expectations are particularly strong in these kinds of notations. In addition, both settings apply eye movement parameters of cognitive effort: pupil size in the music experiment and saccadic velocity in the code experiment. The code reading study also had an additional subgoal of testing two possible options (code element and code line) for meaningful unit of analysis in code reading, merging together language (sign) and thinking (abstraction) aspects.

The third aim of this project is to explore the cognitive processing of incongruent notations in experienced music readers and one experienced code reader. The experimental study on music reading, presented in Article II,

investigates how experienced readers perform from a familiar music notation with incongruent patterns. The experimental case study on pseudocode reading, presented in Article III, serves as the first step for comparing incongruent music and code reading. It provides an exploratory look into how one experienced programmer visually processes incongruent pseudocode notation of a familiar sorting algorithm, which is called the Bubble sort algorithm.

6 Methods

This dissertation includes one theoretical study and two experiments. The theoretical study discusses similarities and differences in creating expectations. The first experiment focuses on the eye movements of experienced participants during playing and singing from an incongruent familiar notation in different tonalities. The second experiment addresses the eye movements of experienced participants during comprehension of an incongruent familiar pseudocode notation.

6.1 Article I: Similarities and Differences in Creating of Expectations in Music and Programming

This article focuses on how expectations are created in music and programming. It explores the similarities and differences in this process between these two domains by reviewing the relevant scientific papers from both domains with a special emphasis on visual attention. In addition, the study reviews the related scientific papers in order to compare three different theoretical paradigms (sociocultural, Action-Process-Object-Schema theory, and conceptual change) that can be used to study prediction mechanisms on the basis of expectations in both fields. The study provides a theoretical model of prediction in music and programming. Subsequent experimental studies represent the first attempts to obtain deeper insights into several components of the suggested prediction model when prediction works in normal (congruent notation) and wrong (incongruent notation) ways.

6.2 Article II: Playing and Singing Incongruent Music Notation

Research from the music perception domain revealed two particularly interesting findings for the set-up choice for Experiment 1. According to the first study (Gunter, Schmidt, & Besson, 2003), incongruence detection in music relies on modality-independent processing mechanisms—that is, incongruence detection is comparable in music reading and music hearing. According to the second study (Vuust, Dietz, Witek, & Kringelbach, 2018), conducted on rhythmic incongruence perception

modeling, prediction is less salient in unpredictable music. Hence, familiar music was selected for use in this experiment as being most predictable. The selected set-up was based on a prior study (Penttinen et al., 2015) in which incongruences were introduced and experimentally tested in the "Mary Had a Little Lamb" melody. All music performances were temporarily controlled using a metronome. Experiment 1 developed this set-up by inserting incongruent notes into two different tonalities and then asking participants to play them on a piano or to sing from this notation. It aimed to test the first, second, fourth, and fifth components of the prediction model in music reading that was proposed in the theoretical study.

6.2.1 Participants

Twenty-four musically experienced subjects, aged 16–57, participated in the study ($M = 32.50$, $SD = 10.18$, 20 females). Participants were recruited from the Turku area in Finland and included professional musicians as well as people in other occupations but with sufficient music experience. All participants had at least four years of music experience. The sample size was defined by the number of available volunteer participants. Two participants were not included in the data analyses because of calibration problems and rhythmically incorrect performances.

6.2.2 Stimulus Materials

A music notation of the popular children's song "Mary Had a Little Lamb," composed by L. Mason, was presented on a computer screen with a resolution of 1920 x 1080 pixels. Eight-bar-long music scores of the song in C major were used together with two variations (created by M. Puurtinen, see also Penttinen et al., 2015) that consisted of one bar being shifted down (that is, incongruence). The original melody and its variations were then transposed to B major, which is considered to be a more complicated tonality because of its five sharps (Appendices 1 and 2).

6.2.3 Reading and Memory Tests

In order to complement eye movement parameters, reading ability and auditory working memory capacity were measured using word and pseudoword reading tests obtained from the Assessment Battery for Reading Disabilities in Young and Adults (Nevala, Kairaluoma, Ahonen, Aro, & Holopainen, 2007) as well as a digit-span memory test obtained from the Dyslexia and Literacy International Charity website (https://www.dyslexia-and-literacy.international). Music reading skills are considered to be linked with text reading comprehension (Gromko, 2004; Reifinger, 2018) and working memory (Kopiez & Lee, 2006; Meinz & Hambrick, 2010).

Therefore, correlations between reading skills or working memory and the correctness of incongruent music performance were investigated in the study. All tests were conducted in the Finnish language. Participants were asked to read the list of words and pseudowords. In reading tests, the speed and accuracy of reading were assessed while the participants read the list of words and pseudowords. In the digit-span memory test, participants recalled increasing rows of numbers that were recorded on tape by a native Finnish speaker in both direct and reverse orders.

6.2.4 Apparatus

Participants played from a given music notation on a Yamaha electric piano. Playing and singing performances were recorded with two video cameras, one of which was adjusted to the computer screen, while the other one was placed in the corner of the laboratory to capture the computer screen and the participants' hands.

Eye movements during playing and singing from the notation were recorded using a remote binocular Tobii TX 300 eye tracker. The sampling rate of the eye tracker was 300 Hz. Fixations and saccades were defined in accordance with the default I-VT fixation classification algorithm (Olsen, 2012) from the Tobii Pro Studio using a threshold of 30 degrees/second.

To control for the pupillary light reflex in pupil size measurements, luminance in the laboratory and luminance of the experimental stimuli were kept at the same level for all recordings. Furthermore, identical black and white colors were used for calibration stimuli and calibration background as well as for stimuli and background in the music reading trials. Experimental melodies were presented at the same location in the center of the computer screen to control for the pupil foreshortening error. Participants were recorded without a chinrest in order not to restrict their natural hand and body movements as they played on the piano. Individual differences in the pupil size were taken into account in the statistical analysis (see 6.2.6).

6.2.5 Procedure

The study comprised three parts. In the beginning, participants were told about the purposes of the experiment and gave their informed consent for data use and storage. They were also informed that their performances would be videorecorded. Subsequently, participants took part in an eye-tracking practice session, where they were asked to play and then sing (or sing and then play; depending on whether they started the experiment with the singing or playing part) the original "Mary Had a Little Lamb" melody in two tonalities—C and B major—with a metronome set at 60 bpm. Melodies did not contain large melodic skips that might influence the eye-time span parameter (Huovinen et al., 2018). The tempo of 60 bpm was chosen in

accordance with the previous study (Penttinen et al., 2015). Participants were informed about the tonality and the first note of each melody. In addition, they were instructed that they should look at the fixation cross marking the beginning of each melody for two metronome beats before they would see the melody. When the melody appeared on the screen, the participant was asked to wait for two quarter beats and start performing the melody after that. Thus, they did not have time to examine the melody before the start of the performance. Participants played the melodies with their right hand or sang them with the syllable "la" instead of words. Participants did not use any instrumental accompaniment while singing.

After the practice session, participants were told that they would perform melodies during the experiment that would include certain changes from the original melody and that they should perform exactly what they see in the music notation. Each subject participated in one singing and one playing session that was preceded by a calibration, where he/she was invited to perform one out of four possible stimuli combinations (Table 4) with a total of eight melodies accompanied by a metronome. Participants did not know which melodies would be congruent or incongruent. Similarly to the practice session, participants were informed about the tonality and the first note of every melody, instructed to look at the fixation cross for two metronome beats and start performance after two metronome beats following the appearance of the melody on the screen.

Table 4. Design of the Music Reading Experiment (Cc = C Major Congruent, Ci = C Major Incongruent, Bc = B Major Congruent, Bi = B Major Incongruent). Source: Altered Table from Article II (Chitalkina et al., 2021).

Combination of Stimuli		Stimulus 1	Stimulus 2		Stimulus 3	Stimulus 4
1	Singing	Cc, Ci1	Bc, Bi2	Playing	Bc, Bi1	Cc, Ci2
2	Singing	Bc, Bi1	Cc, Ci2	Playing	Cc, Ci1	Bc, Bi2
3	Playing	Cc, Ci2	Bc, Bi1	Singing	Bc, Bi2	Cc, Ci1
4	Playing	Bc, Bi2	Cc, Ci1	Singing	Cc, Ci2	Bc, Bi1

At the end of the study, participants were asked to fill out a questionnaire about their music background and to evaluate their sight-reading and sight-singing skills. Next, they participated in reading (reading of words and pseudowords) and working memory (digit-span) tests. Finally, participants informally discussed their reading with the researcher and viewed their eye movement recordings. The entire experiment took approximately one hour.

6.2.6 Data analysis

Research data analysis included three stages: (1) performance analysis, (2) analysis of reading and memory parameters, and (3) analysis of eye movement parameters.

Performance data were co-analyzed by the author of the dissertation and her main supervisor, who is also an active musician (M. Puurtinen). The author analyzed the correctness of all music performances, and the most difficult cases were judged together with the main supervisor. Performance analysis focused exclusively on the correctness of the three bars of the incongruent melodies and the corresponding bars of the original melodies—the altered one, the preceding bar, and the succeeding bar. In addition, the number of errors in the four incongruent melodies was calculated for each participant. Pearson's chi-square tests were used to study the relationship between the following parameters: type of the melody, modality of performance, tonality of performance, and correctness of performance.

In order to analyze reading and memory parameters, the reading time of words, pseudowords, and the number of errors were measured using a timer. Furthermore, the number of correctly recalled digits in direct and reverse orders, their sum, and the standard score for the sum were also calculated, respectively. Data were further analyzed using non-parametric Kendall's tau b tests in SPSS in order to reveal any correlation between the number of errors in incongruent melodies and every cognitive parameter related to reading and memory: amount of time spent on reading words, amount of time spent on reading pseudowords, the number of errors in pseudowords, the number of correctly recalled digits in direct order, the number of correctly recalled digits in reverse order, the total number of recalled digits, and the standard score for the total number of recalled digits.

The analysis of eye movements focused on three parameters: (1) eye-time span (ETS, Figure 3), (2) duration of first-pass fixations, and (3) mean pupil size in first-pass fixations. A half-bar was chosen as a unit of analysis. Since the primary focus of the analysis was on the processing of incongruent changes, six squared areas of interest (AOIs) were created using the Tobii Pro Studio software around the half-bars of the changed bar, the preceding bar, and the succeeding bar in incongruent melodies and the corresponding bars in congruent melodies (pre-target a, pre-target b, target a, target b, post-target a, post-target b). The AOI data without fixations were handled as "missing" in the analysis. Custom written scripts in R were applied to calculate the first-pass fixation duration and ETS for six half-bars and the mean pupil size in first-pass fixations for four half-bars (pre-target b, target a, target b, post-target a). The mean pupil size in first-pass fixations was not calculated in the pre-target a half-bar, because the pupil size at that part of the stimulus does not reflect the handling of incongruence due to the delayed nature of changes in the parameter. In addition, mean pupil size in first-pass fixations was not calculated in the post-target b half-bar, because it was decided not to study catch-up processes for this

parameter, since this would require inclusion of additional half-bars in the analysis. Random intercepts for participants were introduced in linear mixed models to take into account individual differences in pupil dilation. ETS was calculated according to the algorithm reported in Huovinen et al. (2018) and in Article II.

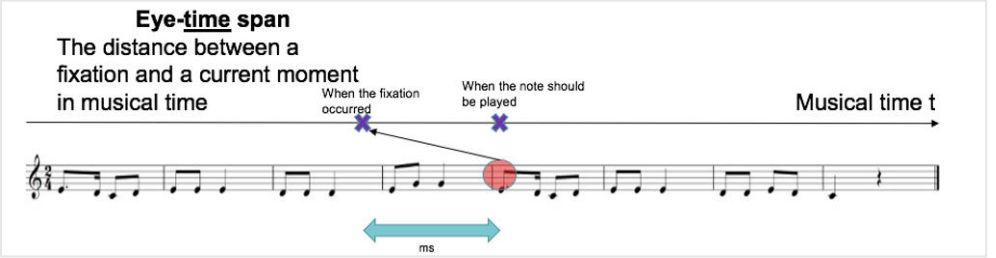


Figure 3. Eye-time span (Huovinen et al., 2018) parameter in Mary had a little lamb melody composed by L. Mason

Next, the relationship between the three above-described eye movement parameters and the modality, tonality, and congruence was investigated using linear mixed models in the lme4 package (Bates, Mächler, Bolker, & Walker, 2014) of R. The following fixed effects were inserted into the model: mistake, congruence, tonality, and modality; two-way interactions modality*tonality, modality*congruence, and tonality*modality; and three-way interactions tonality*modality*congruence. Intercepts for subjects and by-subject random slopes for the effect of mistakes were entered as random effects. The analysis focused on the interpretation of the highest-order interaction if it was found to be significant.

6.3 Article III: Comprehension of Incongruent Code Notation

The code reading experiment was carefully designed on the basis of the music reading study in order to make them as similar to one another as possible. Consequently, incongruences were also introduced here to different parts of a familiar notation. The Bubble sort algorithm—a well-known sorting algorithm—was chosen as an analogue of the "Mary Had a Little Lamb" melody in programming. Like in the music reading study, all code reading performances were temporarily controlled: participants had only one minute to read each code. The experiment aimed to test the first and second components of the prediction model in code reading that was introduced in the theoretical study.

6.3.1 Participant

One experienced participant was chosen from a sample for this case study. His data were recorded earlier than the data of the rest of the sample. (The sample data were not included in the dissertation, because the data collection was delayed due to the exceptional pandemic situation in 2020-2021). In addition, the participant completed the task in a different manner than the rest of the sample. He spoke his thoughts aloud even though he was not required to do so. The recording of the thinking-aloud process allowed researchers to verify the error types that the participant made. The participant was a programming teacher who has actively been programming for more than 10 years. At the time of measurement, he was actively programming several days per week. The participant self-assessed his general programming skills as extremely advanced, his main language programming skills as advanced, and his knowledge of the Bubble sort algorithm as extremely advanced.

6.3.2 Stimulus Materials

Fourteen pseudocodes of the Bubble sort algorithm were presented on a computer screen with a resolution of 1920 x 1080 pixels. The Bubble sort sorts a list of numbers by comparing two adjoining numbers and swapping them if they are in the wrong order. The pseudocode designed for this study looked like real code but also tried to avoid any language-specific features. The purpose of using a pseudocode instead of a real programming language was to ensure that different participants would not have an unfair advantage if they happen to be experts in a particular programming language. Out of fourteen algorithms, seven included one-unit long unexpected changes that also affected the output; the remaining seven algorithms were original Bubble sorts, four of which sorting in the ascending order while the remaining three sorting in the descending order (Appendix 3).

6.3.3 Apparatus

Eye movements during code reading were recorded using a remote binocular Tobii TX 300 eye tracker with a sampling rate of 300 Hz. Fixations and saccades were classified according to the default I-VT fixation filter (Olsen, 2012) from the Tobii Pro Studio using a threshold of 30 degrees/second.

The participant's performance was recorded with a video camera that was placed in the corner of the laboratory in order to capture the computer screen and the participant's responses.

6.3.4 Procedure

When the participant came to the laboratory, he was informed about the goals of the study and signed the informed consent form. Subsequently, he participated in the practice session during which he was asked to figure out the outputs of codes that can be created based on the pseudocodes that he sees. The participant read two pseudocodes of the Bubble sort algorithms—one sorting in the ascending order and the other sorting in the descending order—for one minute and had unlimited time to choose one out of four possible output options afterward. When the participant chose the output option, he pressed any key on the keyboard to move on to the next pseudocode on the screen.

The eye-tracking experiment was conducted following the practice session. The participant was informed that he would see fourteen pseudocodes of the Bubble sort algorithm and that some of them would include changes that could affect the output. He was asked to figure out the outputs on the basis of the notations that he would see on the screen. Before the experiment, black stimuli on a white background—equivalent to the stimuli and background colors of the experiment—were used for the calibration of the eye racker. Next, the participant read fourteen pseudocodes, one minute each, and chose an output from four possible options. Fourteen pseudocodes included four original ascending Bubble sorts, three original descending Bubble sorts, and seven variations containing one-unit-long changes.

After the eye-tracking part, an English spelling test and a digit-span memory test were conducted. In the end, the participant saw his eye movement recording and discussed the study task with the researcher. The entire experiment took around one hour.

6.3.5 Data analysis

Data analysis focused on the five pseudocodes read by the experienced participant: one original ascending, two incongruent variations that the participant performed correctly, and two incongruent variations for which the participant made proof-readers' errors—that is, he thought these variations were original ascending pseudocodes and chose the corresponding outputs (Appendix 3). Proof-readers' errors were verified with the help of the thinking-aloud videorecording.

The eye movement analysis performed in this study represented descriptive comparisons of eye movement parameters in the pseudocodes with proof-readers' errors and in the original ascending pseudocode as well as eye movement parameters in correct incongruent variations and original ascending pseudocode. The eye movement analysis focused on fixation and saccadic velocity parameters. This study also examined the problem of carefully choosing the meaningful analysis unit by comparing two alternative solutions—the AOIs created around the altered element

in the incongruent pseudocode and the corresponding element in the original pseudocode (Appendix 5) as well as the AOIs created around the line with the altered element in the incongruent pseudocode and the corresponding line in the original pseudocode (Appendix 4). Fixation parameters were analyzed for both AOI kinds. Saccade velocity parameters were analyzed for the time periods starting with the first saccade out of the line with the altered element in the incongruent pseudocode and the corresponding line in the original pseudocode.

6.4 Research Ethics

All participants participated in these experimental studies voluntarily and provided their informed consent to data use and storage. The obtained research material was used for scientific purposes only. All research data were collected and stored according to the ethical guidelines of the General Data Protection Regulation of the European Union and the Finnish Advisory Board of Research Integrity. The data did not include sensitive material that would require a statement from the University of Turku Ethics Council. Storage was made only for anonymized data, unless the participant had consented to appear in his or her own name. Participants of the music study were compensated for their participation with lunch tickets for the university cafe. The pseudocode reading study participant did not receive any compensation for his participation.

7 Overview of the main results

This dissertation is based on three articles and includes one theoretical and two empirical studies. The theoretical study explores how expectations about music and code are created and how they allow musicians and programmers to predict the contents of the notation and recognize familiar patterns. The first empirical study provides insights into the processing of incongruent music notation in skilled readers. The second empirical study investigates the processing of incongruent pseudocode in one skilled reader. The current section presents the summary of the main findings from each scientific article.

7.1 Article I: Similarities and Differences in Creating Expectations in Music and Programming

The study presented in Article I examined prediction and its underlying components as key principles for advances in music and code reading. Experienced music and code readers have expectations about notations, which allows them to predict what they will read and to better recognize familiar patterns. Neuroscientists consider prediction to be one of the main important principals of brain work (Friston, 2005; Bar, 2009; Bubic et al., 2010). In particular, our brain creates so-called predictive models in order to explain the upcoming sensory information (Bubic et al., 2010; Bar, 2009; Friston, 2005). The study theoretically explored the cognitive foundation of prediction in music and programming by comparing them to each other and by indicating the similarities and differences between them. The proposed prediction model includes three components that are common for both music and programming: (1) knowledge of language systems, (2) knowledge of meaning, and (3) knowledge of context. In addition, it also contains two components that are different for music and programming: (4) translation of information and (5) temporal and motor requirements. Therefore, expertise development can be considered as the development of the components of the prediction model.

The first component refers to the knowledge of various basic signs (such as notes, musical articulation signs, and dynamic markings in music; keywords, variables, operators in programming) and more complex sign-related concepts (such as key and

time signatures in music; variable declarations and array indexing in programming) used in non-natural notations. Knowledge of signs can be considered as a foundation for pattern recognition. It seems that while recognizing patterns, musicians and programmers tend to orient more toward the signs than to their meanings. However, knowledge of language signs is not enough for the successful execution of music and programming activities.

The second component emphasizes that, behind signs, there is always knowledge of their meaning that constantly develops over time. In reality, students might start to perform music or code before attaining a complete understanding of what all notational nuances mean for music performance (for instance, in terms of music expression) or how all stages of problem-solving should be expressed in code (by proceeding with working code snippets obtained from the web that they not fully understand). The sociocultural paradigm assumes that the development of concept meaning occurs in several stages (Vygotsky, 1934). In this sense, student misconceptions are not viewed as errors but rather as necessary steps for achieving expert knowledge level. The sociocultural paradigm considers society to be the driving force for the development of cognitive functions (Vygotsky, 1934). In particular, for music and programming, this driving force is the music or coding community that mediates the development of the meaning of music and code concepts required for musical and computational thinking. This influence is not limited to formal schooling but also includes all kinds of informal communication experiences, such as playing in bands or discussing code on online forums.

The third component pays attention to the importance of context in both music and programming. Every concept meaning appears within a specific context. Application domains and music genres might serve as examples of different contexts in programming and music, respectively. In terms of expertise development theories(Ericsson, 2016; Boshuizen et al., 2020), this component can be linked to both deliberate practice and knowledge encapsulation. A lot of deliberate practice is needed to learn and master knowledge of contexts. In addition, musicians and coders have to deal with many similar types of performance and computational tasks to achieve expertise.

The following two components differ in music and programming. The fourth component emphasizes that the information from a notation is translated to the language of musical instruments (for instance, from Western music notation to black and white piano keys) and computers (source code to machine code) in different ways. Code is compiled and executed automatically by computers, whereas music is compiled and executed by musicians. The last component refers to different temporal and motor requirements in music and programming. Music is compiled and executed by musicians in a limited timeframe because musicians have to perform in a constant tempo during the entire music piece and in compliance with the physical requirements of musical instruments. In contrast, timing (with the only exception of

work deadlines) and motor aspects of coding are not related to programmers but to the computer system instead.

Experiments presented in this dissertation are the first steps taken to examine some components of the proposed prediction model in detail when prediction works normally (congruent notation) and when prediction is violated (incongruent notation). Familiar music and code notations (a popular song and well-known sorting algorithm) were chosen as experimental stimuli in order to induce experienced readers' existing prediction models. Hence, the first experiment focuses on the first, second, fourth, and fifth components of the model in music reading. It explores the processing of the meaning of the congruent and incongruent musical symbols in experienced music readers. In addition, it also explores the translation of music information into two different performance ways (singing and playing piano) with different motor requirements. Furthermore, the first experiment imposes the same temporal requirements for all music performances by applying a metronome with a steady beat. The second experiment provides some insights into the first and second components of the model in programming. Similarly to the music experiment, it investigates the processing of the meaning of the congruent and incongruent code symbols in an experienced programmer. In addition, the second experiment introduces the same temporal requirements for all code reading tasks. The code experiment was simplified to avoid the influence of context (third component) using pseudocode instead of source code. Processing the meaning of the incongruent music and code symbols is expected to cause changes in fixational and cognitive effort parameters as well as in eye-time span during music reading.

7.2 Article II: Playing and Singing from Incongruent Music Notation

The music experiment explored how experienced performers deal with unexpected note changes in familiar melodies. These melodies were presented in two tonalities, and the participants were asked to sing one portion and play another. The only thing that was altered in the incongruent melodies was the pitch—that is, the notes on one bar were shifted down. The quantity and duration of the notes remained unchanged.

7.2.1 Cognitive Processing of Incongruent Notation in Experienced Music Readers

The analysis revealed that congruent melodies were performed correctly significantly more often than incongruent melodies (Table 5). Thus, it seems that the pitch anticipation of the performers was rather strong and that it affected the correctness of the incongruent melody performances. No significant correlations

were found between any of reading or memory parameters and the number of errors in incongruent melodies (Table 6).

Table 5. Quality of Performances. Source: Article II (Chitalkina et al., 2021).

Type of Melody	Number of Correct Singing Performances	Number of Incorrect Singing Performances	Number of Correct Playing Performances	Number of Incorrect Playing Performances
C congruent	22	0	21	1
C incongruent	13	9	17	5
B congruent	22	0	18	4
B incongruent	11	11	15	7
Total	68	20	71	17

Table 6. Correlations Between Reading, Memory Parameters, and the Number of Mistakes. Source: Article II (Chitalkina et al., 2021).

Type of Melody	*r*	*p*
Digit forwards raw	-0.02	0.91
Digit backwards raw	-0.01	0.95
Digit total raw score	-0.03	0.88
Digit total standardized score	-0.05	0.79
Time spent on reading words	0.20	0.23
Time spent on reading pseudowords	0.09	0.58
Number of mistakes in pseudowords	-0.26	0.14

In order to examine the time course of the detailed processing of incongruent notes, the performance analysis was complemented with the analysis of three eye movement parameters: eye-time span (ETS), duration of first-pass fixations, and mean pupil size in first-pass fixations.

In the beginning, these eye movement parameters were analyzed to find out how experienced participants approached incongruent notes. In agreement with previous research (Huovinen et al., 2018), the effect that was associated with upcoming difficulties was found for the ETS parameter on the music notation symbols preceding the most complex symbols. Thus, the ETS increased in the pre-target a

half-bar while performing from an incongruent notation. In addition, there was a significant two-way interaction between tonality and the melody type for the first-pass fixation duration parameter in the pre-target b half-bar (Figure 4). Performers made longer first-pass fixations to the half-bar immediately preceding the incongruent half-bars in the B major in comparison to the C major. Thus, it might be more complicated to approach an upcoming incongruence in the B major.

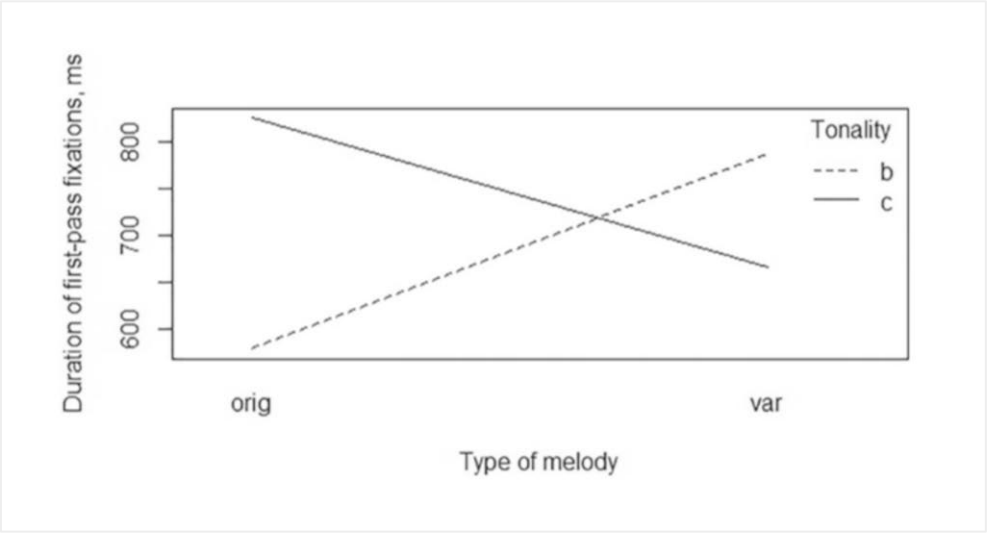


Figure 4. Mean duration of first-pass fixations measured for original and (incongruent) variation melodies in the C and B majors during the second half of the bar preceding the incongruent one. Source: Article II (Chitalkina et al., 2021).

Next, the significant effects of incongruence on the first-pass fixation duration in target a, on the ETS and the mean pupil size during the first-pass fixations in target b, and on the mean pupil size during the first-pass fixations in post-target a can reflect that the participants struggled with musical complexity. Thus, a decrease in the ETS in target b half-bar of incongruent melodies might signalize that the participants could not approach this half-bar as early as they reached the corresponding half-bar of the original melody because they already began to struggle with the preceding incongruent notation. In addition, participants reached the first incongruent half-bar with shorter first-pass fixations and the first half-bar after incongruent half-bars with longer first-pass fixations in comparison to original melodies. The changes in the pupil size parameter occurred relatively slowly (Katidioti et al., 2016). Thus, the increase in the pupil size parameter in both target b and post-target a probably reflect the challenges in the cognitive processing of the incongruent pitch that occurred earlier in the target half-bars.

Finally, significant effects of incongruence on eye movement parameters (except for mean pupil size) in post-target half-bars could reveal how the participants caught up with the continuation of reading after incongruent notes. Indeed, an increase in the first-pass fixation duration for incongruent melodies in post-target a might reflect the processing difficulty of this catch up. Also, a significant two-way interaction between tonality and incongruency for the ETS parameter was found in the post-target b half-bar (Figure 5). Performers had longer ETS when they performed the second half of the post-target bar of incongruent melodies in the B major in comparison to the C major.

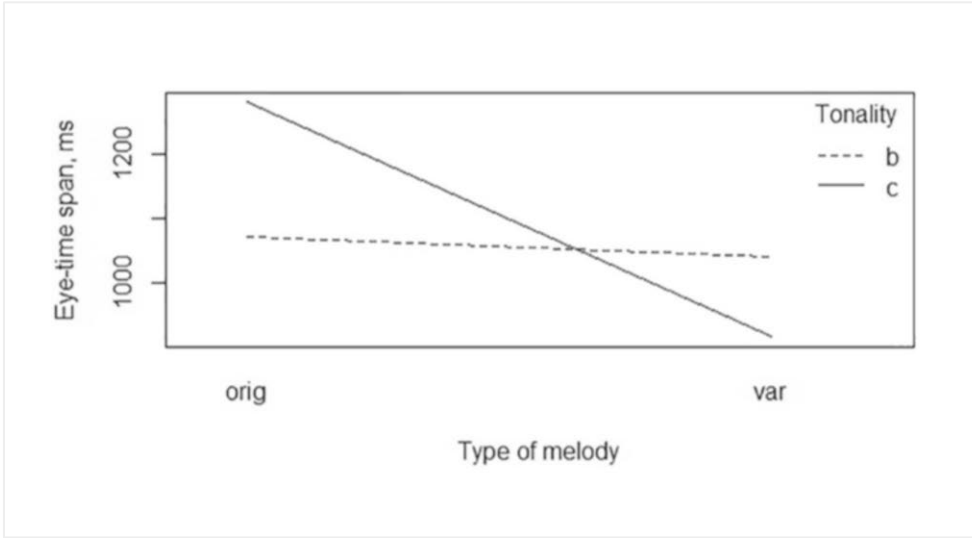


Figure 5. Mean eye-time spans measured for original and variation melodies in the C and B majors in the second half of the bar succeeding the incongruent one. Source: Article II (Chitalkina et al., 2021).

To summarize, the processing of the meaning of surprising note symbols caused various changes (Table 7) in the three analyzed eye movement parameters, which began one bar before the incongruent notes, continued during the incongruent bar, and finished one bar after the incongruent one. Furthermore, it seems that performing incongruent melodies in the B major was more difficult than in the C major, which was reflected in the approach to the incongruent bar and in the catch up with the subsequent notation.

Table 7. The Time Course of Significant Effects of Incongruence on Eye Movement Parameters in the Music Reading Tasks.

Half-Bar of the Melody	Eye Movement Parameter	Statistics and Effect
Pre-target a*	ETS	$\chi 2$ (1) = 14.41, p < .001; increase by 334.47 ± 85.15 ms
Pre-target b	-	-
Target a	First-pass fixation duration	$\chi 2$(1) = 7.35, p = .007; decrease by 199.95 ± 72.41 ms
Target b	ETS	$\chi 2$(1) = 4.09, p = .04; decrease by 158.59 ± 77.56 ms
	Mean pupil size in first-pass fixations	$\chi 2$(1) = 37.3, p < .001; increase by 0.16 ± 0.02 mm
Post-target a	First-pass fixation duration	$\chi 2$(1) = 4.79, p = .03; increase by 144.89 ± 64.37 ms
	Mean pupil size in first-pass fixations	$\chi 2$ (1) = 33.28, p < .001; increase by 0.14 ± 0.02 mm
Post-target b*	-	-

*Mean pupil size in first-pass fixations was not analyzed in these half-bars.

7.2.2 Research Methodology for Studying Incongruence in Music Reading

Using a combination of these three different eye movement parameters allowed the researcher to describe the time course of incongruent reading of familiar music notation in detail. Indeed, the eye-time span parameter applied in the study took into account the temporal aspect of incongruent music reading. The mean pupil size, which was measured only in first-pass fixations, allowed for an evaluation of the cognitive effort required to process an incongruent notation. The first-pass fixation duration can be considered to be a marker of attention needed to be paid to a notation. The use of melodies in two different tonalities provided an opportunity to compare incongruent reading in these tonalities and discover that the differences mainly affect the bars that precede and follow the incongruent notation bars.

The experiment also explored the translation of music information and the motor requirements in music reading by comparing the acts of playing on a piano and singing from music scores. No interaction of performance mode with incongruence was found. However, there were significant two-way modality*tonality interactions in target b, post-target a, and post-target b half-bars for the ETS parameter (Figures 6, 7, and 8).

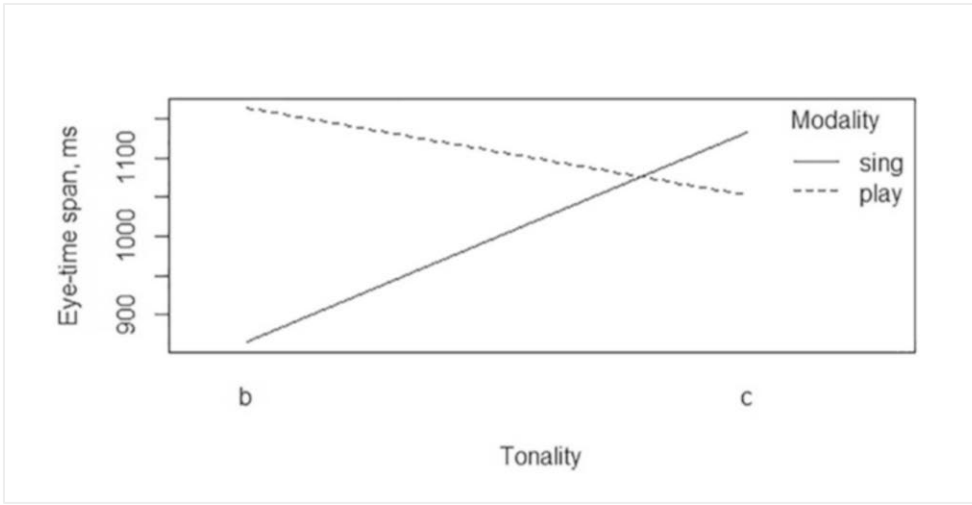


Figure 6. Mean eye-time spans measured for singing and playing performances in the C and B majors in the second half of the incongruent (target) bar. Source: Article II (Chitalkina et al., 2021)

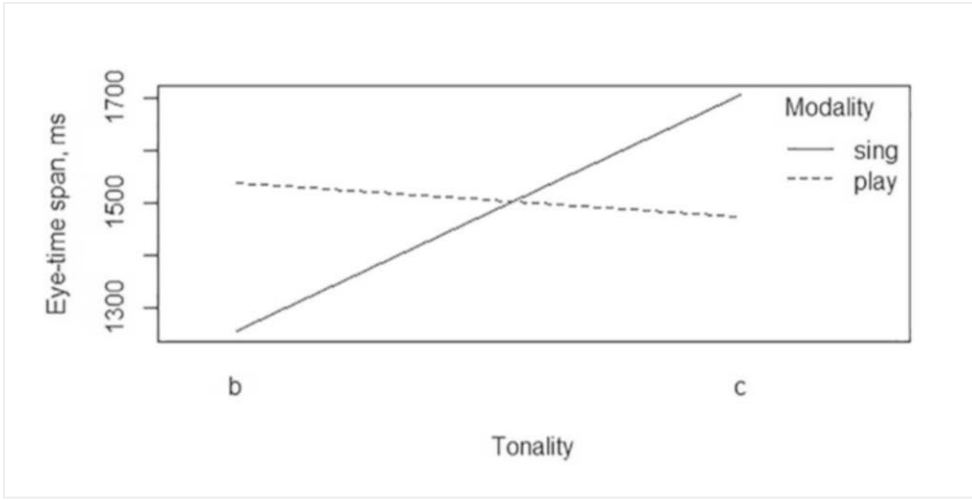


Figure 7. Mean eye-time spans measured for singing and playing performances in the C and B majors in the first half of the bar succeeding the incongruent one. Source: Article II (Chitalkina et al., 2021)

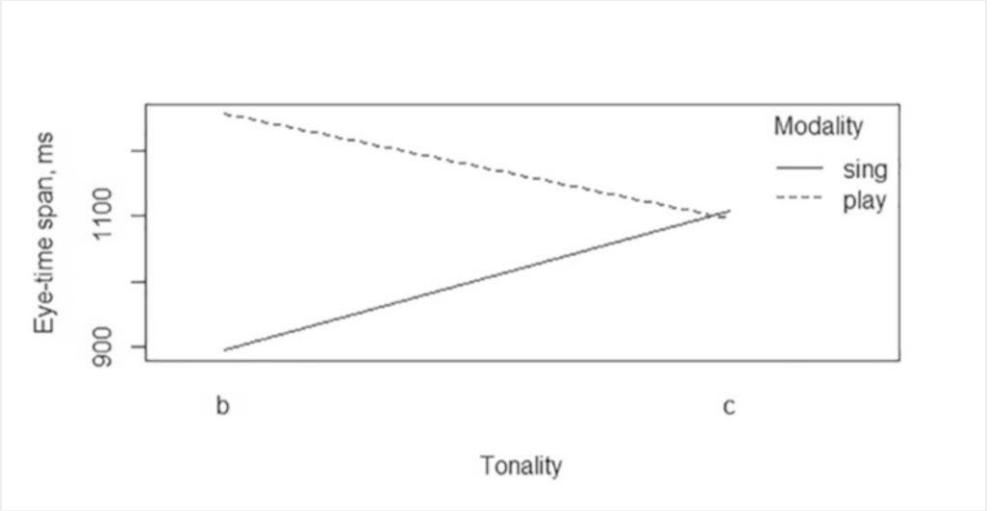


Figure 8. Mean eye-time spans measured for singing and playing performances in the C and B majors in the second half of the bar succeeding the incongruent one. Source: Article II (Chitalkina et al., 2021)

In all these half-bars, participants operated with lower ETS when they had to sing in a more complicated key of the B major than when they had to play in the B major. This finding might possibly be explained by saying that participants needed more time to translate the notation into "the musical instrument language" and to plan their motor sequences in the B major. Moreover, singing decreased the mean pupil size in first-pass fixations in pre-target b, target a, target b, and post-target a half-bars. The results of the pupil size analysis (Table 8) pointed out that playing the piano from music scores might be a more complicated cognitive task in comparison to singing from music scores. However, according to performance analysis, participants played incongruent melodies better than they sang them. Therefore, it seems that if participants have skills in both activities, then they are able to cope despite the fact that playing (especially in the B major) might be more cognitively demanding than singing (Chitalkina et al., 2021).

Table 8. The Time Course of Significant Effects of Singing on the Mean Pupil Size in First-Pass Fixations.

Half-Bar of the Melody	Statistics and Effect
Pre-target b	$\chi 2(1) = 8.41$, $p = .004$; decrease by 0.06 ± 0.02 mm
Target a	$\chi 2(1) = 9.95$, $p = .002$; decrease by 0.05 ± 0.02 mm
Target b	$\chi 2(1) = 7.75$, $p = .005$; decrease by 0.06 ± 0.02 mm
Post-target a	$\chi 2$ (1) = 7.24, $p = .007$; decrease by 0.06 ± 0.02 mm

7.3 Article III: Comprehension of Incongruent Code Notation

The code reading case study provides an exploratory look at how an experienced coder handled unexpected one-element-long code changes in a notation of the Bubble sort algorithm. It describes the processing of meaning in four algorithms with unexpected changes, including two algorithms in which the participant noticed the change and correctly figured out the output (Condition 1) and another two where the participant did not notice unexpected changes and thought that these algorithms were original Bubble sort algorithms (Condition 2). The readings of these four algorithms were compared to the correct reading of the original Bubble sort algorithm (Condition 3) with the help of fixation and saccadic eye movement parameters (Appendix 3). The particular focus of the study was on Condition 2 and its cognitive processing.

7.3.1 Cognitive Processing of Incongruent Notation in an Experienced Programmer

In this study, areas of interest were created in two different ways: (1) around separate code elements and (2) around code lines. According to the first way, areas of interest were drawn around the altered element in incongruent algorithms and their corresponding elements in the original algorithms. The analysis of fixation parameters revealed differences between Condition 3 and Condition 1, as well as between Condition 3 and Condition 2, reflected in the total fixation duration, number of fixations, total visit duration, and number of visits. These parameters increased for Condition 1 in comparison to Condition 3 and either decreased or remained the same for Condition 2 in comparison to Condition 3 (Table 9).

Table 9. Fixational Parameters in AOIs Created around Altered Code Elements in Correct Incongruent Codes (Condition 1), in Incongruent Codes with Proof-Readers' Errors (Condition 2), and in the Corresponding Code Elements of the Original Algorithm (Condition 3). Source: Altered Table from Article III (Chitalkina et al., 2020).

	Total Fixation Duration, ms	**Number of Fixations**	**Total Visit Duration, ms**	**Number of Visits**
Condition 1 (a)	607	7	2906	6
Condition 3 (a)	177	2	177	2
Condition 1 (b)	233	3	233	3
Condition 3 (b)	150	2	150	2
Condition 2 (c)	0	0	0	0
Condition 3 (c)	0	0	0	0
Condition 2 (d)	377	3	390	2
Condition 3 (d)	417	4	873	2

According to the second way, areas of interest were created around the lines with altered elements in incongruent algorithms and their corresponding lines in the original algorithm. The findings revealed that total fixation duration, number of fixations, and total visit duration in both Conditions 1 and 2 increased in comparison to Condition 3 (Table 10).

In addition to fixation parameters, saccadic velocity parameters were analyzed for the first saccades from the congruent and incongruent lines in algorithms. The analysis of saccadic velocity parameters revealed processing differences between the three studied conditions. Thus, the peak velocity of the first saccade and the average velocity of the first saccade, made out of a line with an incongruent element in Condition 1, were higher than the same parameters in the corresponding line of Condition 3, whereas these parameters in Condition 2 were lower than in Condition 3 (Table 11). According to prior studies, increased task difficulty (and cognitive effort needed to accomplish it) and increased arousal are associated with lower saccadic peak velocity (see Di Stasi, Catena, Canas, Macknik, & Martinez-Conde, 2013 for an overview). It is possible that the participant figured out that something was wrong with the code snippets in Condition 2, but was unable to find exactly what was wrong, which led to increase in arousal and cognitive effort.

Table 10. Fixational Parameters in AOIs Created around Code Lines with Altered Elements in Correct Incongruent Codes (Condition 1), in Incongruent Codes with Proof-Readers' Errors (Condition 2), and in the Corresponding Code Lines of the Original Algorithm (Condition 3). Source: Altered Table from Article III (Chitalkina et al., 2020).

	Total Fixation Duration, ms	**Number of Fixations**	**Total Visit Duration, ms**
Condition 1 (a)	1277	14	6249
Condition 3 (a)	313	4	3156
Condition 1 (b)	333	5	873
Condition 3 (b)	297	4	297
Condition 2 (c)	887	10	1143
Condition 3 (c)	233	3	233
Condition 2 (d)	1437	15	3506
Condition 3 (d)	763	9	2403

Table 11. Saccadic Parameters in TOIs Created for the First Saccade from Code Lines with Altered Elements in Correct Incongruent Codes (Condition 1), in Incongruent Codes with Proof-Readers' Errors (Condition 2), and from the Corresponding Code Lines of the Original Algorithm (Condition 3). Source: Altered Table from Article III (Chitalkina et al., 2020).

	Peak Velocity of the First Saccade, °/s	**Average Velocity of the First Saccade, °/s**
Condition 1 (a)	148.27	97.81
Condition 3 (a)	39.95	39.90
Condition 1 (b)	98.38	67.84
Condition 3 (b)	45.90	43.51
Condition 2 (c)	52.36	45.75
Condition 3 (c)	77.54	51.68
Condition 2 (d)	33.56	32.60
Condition 3 (d)	239.70	148.58

To summarize the findings, the eye movement analysis demonstrated that these three conditions differ in terms of their cognitive processing. However, these differences were more visible in the analysis of the code lines with surprising elements and in their corresponding lines in the original algorithm with the help of the two eye movement parameter categories that provide complementary information: fixation and saccadic velocity parameters.

7.3.2 Research Methodology for Studying Incongruence in Code Reading

This code reading study was developed to be as close to the music reading study as possible. Indeed, the familiar code of the Bubble sort algorithm was used, and the participants were informed about the purpose of the algorithm. Similar to the music study, incongruent changes were introduced in different parts of the familiar notation and all readings were temporarily controlled. The pseudocode form of the notation was used to reduce the influence of programming experience with specific programming languages.

The study examined two approaches for defining the units of analysis in code reading. By analyzing fixation parameters only in the code elements of interest, it is possible to conclude that the reading with proof-readers' errors is quite similar in terms of processing to the reading of the original algorithm. However, the analysis of fixation and saccadic parameters in the code lines of interest demonstrated that an expert did not read the code with proof-readers' errors in a manner similar to how he

read the original code—despite the fact that these readings appeared equal at the behavior level. Hence, at least for this code reading phenomenon, it seems that the choice of the analysis unit might affect the findings and lead to different conclusions. However, this finding needs to be replicated in a larger sample of experienced participants.

The experiment introduced saccadic velocity parameters in order to study incongruent pseudocode reading. These parameters were associated with cognitive effort (Di Stasi et al., 2011). However, they had never previously been used in the code reading field. The current study demonstrates that the peak velocity of the first saccade and the average velocity of the first saccade, made out of a line with an incongruent element, are two useful parameters for evaluating cognitive effort during incongruent reading. Indeed, these parameters revealed the differences that exist between correct and incorrect (with proof-readers' errors) readings of incongruent code.

8 Discussion

8.1 Cognitive Processing of Incongruence in Music and Code Reading

The issue of creating expectations in absence of a needed piece of information or errors in an available piece of information is not only limited to music and code reading. Indeed, anticipation is required even in such simple everyday tasks as movement and interaction with the elements in our environment (Abernethy, Farrow, & Mann, 2018). Neuroscientists consider prediction to be one of the main principles of brain work (Friston, 2005; Bar, 2009; Bubic, Von Cramon, & Schubotz, 2010). This means that the brain uses old available information in order to anticipate and explain new sensory information (Bubic et al., 2010). Furthermore, in many work and sport tasks, movements of our body are based on probabilistic information as a result of temporal limitations under which movements must be planned and executed (Abernethy et al., 2018). Therefore, the findings of this dissertation can also contribute to the development of knowledge on cognitive mechanisms of anticipation skills in general.

Experimental studies presented in the current dissertation provide an insight into the processing of incongruent patterns in music reading and an exploratory look at the processing of incongruent patterns in code reading. In both studies, incongruent patterns in music and code led to a change in cognitive processing that was reflected through several eye movement parameters: fixation parameters and parameters associated with cognitive effort (pupil size and saccadic velocity).

In music, first-pass fixation duration decreased in the first half of the target bar and increased in the first part of the post-target bar during the reading of incongruent melodies. It seems that performers might not need additional first-pass time for the first half of the target bar because their coping strategies were activated earlier, which was confirmed by the increase in another eye movement parameter—ETS. This finding was not in line with a previous study (Hadley et al., 2018) in which no such first-pass fixation duration effects were found. Moreover, the analysis of fixational parameters provided evidence that approaching incongruence in the B major might be more challenging in comparison to the C major. Thus, participants approached the second half of the pre-target bar with longer first-pass fixations for

incongruent melodies in the B major in comparison to incongruent melodies in the C major. In the exploratory code study, total fixation duration, number of fixations, and total visit duration in both correct and incorrect (with proof-readers' errors) readings of incongruent code lines increased when compared to the readings of the corresponding original code lines. Hence, it seems that an experienced participant did not process the incongruent patterns read with proof-readers' errors in a similar manner to the original algorithm.

With regard to the parameters associated with cognitive effort, pupil size changes were analyzed in the music experiment and saccadic velocity parameters were analyzed in the exploratory code study. In music, pupil size analysis revealed an increase in mean pupil size in first-pass fixations during the second half of the target bar and the first half of the post-target bar. This effect might indicate increased cognitive effort while processing the incongruent bar because changes in pupil size occur with a delay (Katidioti et al., 2016). This finding was consistent with Hadley et al. (2018). However, there might also be an alternative explanation for these changes in pupil size. One study from the music perception field (Liao, Yoneya, Kashino, & Furukawa, 2018) reported increase in pupil diameters while listening to surprising compared to unsurprising moments in music. Therefore, the finding might also reflect the feeling of surprise while listening to how incongruent parts of melodies sound while they are being performed. Interestingly, pupil dilation is larger while listening to vocal than to instrumental music (Weiss, Trehub, Schellenberg, & Habashi, 2016). Moreover, vocal demands such as speaking can cause an increase in pupil size compared to nonvocal movements or conditions without any vocal or muscular movements (Brych, Händel, Riechelmann, Pieczykolan, Huestegge, 2021). The experiment presented in Article II showed that producing both vocal and instrumental incongruent music caused an increase in the mean pupil size in first-pass fixations compared to producing vocal and instrumental congruent music. In the exploratory code study, saccadic parameters revealed differences between correct readings of incongruent code snippets and original code in comparison to incorrect readings (with proof-readers' errors) of incongruent code snippets and original code. Thus, two saccadic velocity parameters—the peak velocity of the first saccade and the average velocity of the first saccade, made out of a line with an incongruent element in correct readings of incongruent algorithms—were higher than those same parameters in the corresponding line of the original algorithm, whereas these parameters were lower in the original algorithm than in incongruent algorithms with proof-readers' errors.

In addition to these findings, the music reading study found no correlation between incongruent melody performances and reading parameters. This finding was consonant with Reifinger (2018) who also did not find a connection between reading fluency and sight-singing (Chitalkina et al., 2021).

The music experiment presented in this dissertation also examined the temporal requirements of performing from an incongruent music notation. In contrast to programming, music performers have to deal with incongruent notation within a strictly limited timeframe. Out of three eye movement parameters analyzed in the music study, the eye-time span provides information about the temporal aspect of the notation processing. In line with a previous study (Huovinen et al., 2018), the eye-time span effect of upcoming difficulties was registered in the first half of the pre-target bar. Thus, approaching the incongruent bar required longer eye-time spans. The opposite situation was registered in the second half of the incongruent bar—participants approached it with lower spans in comparison to the congruent one. It seems that performers might lack the time to look at it earlier as a result of struggling with the first half of the incongruent bar. In addition, the second half of the post-target bar was approached with a longer ETS for incongruent melodies in the B major in comparison to the C major, whereas the length of the ETS in this half bar for congruent melodies was similar to the incongruent ones, which could be evidence that the B major requires more planning.

8.2 Research Methodologies for Studying Incongruence in Music and Code Reading

Experimental studies presented in this dissertation assume that incongruence in music and code should be studied in the context of familiar notations. Expectations are particularly strong within familiar notations. Moreover, research from the music perception domain provides additional evidence for the use of familiar notations in music reading research on incongruence by indicating that incongruence detection in music relies on modality-independent processing mechanisms and that deviation from prediction is less salient in unpredictable music (Gunter et al., 2003; Vuust et al., 2018).

The music experiment outlined in this dissertation improved upon the experimental setting proposed by Penttinen and colleagues (2015), where surprising patterns in the notation were explored by introducing shifted-down notes in a familiar melody. The study presented in this dissertation extends this experiment by inserting surprising patterns in different tonalities and asking participants to play and sing from these music scores. This is the first study in the domain of music reading that explores different ways of performing the same musical task.

At the same time, investigating the playing and singing from a notation provided insights into aspects of music reading that make it different from programming, according to the theoretical model proposed in Article I—that is, translation of information and motor requirements. The results of the performance analysis revealed that participants better played incongruent melodies than they sang them.

However, this issue is not that simple. Eye movement analysis allowed the researcher to shed light on some hidden facets of processing information for playing and singing purposes. Hence, the analysis of pupil sizes revealed that the processing difficulty of singing might be lower in comparison to playing from music scores. Moreover, the ETS analysis showed that there is a significant interaction between modality and tonality in the second half of the target bar and in the post-target bar, which might mean that performers approached these half-bars with lower ETS during singing in the B major in comparison to playing in the B major. The opposite difference in ETS between playing and singing in the C major was smaller. To summarize, eye movement parameters provide evidence that singing from a notation might still be less cognitively demanding than playing. Consequently, performers might need additional resources to plan for motor sequences in piano performances. In addition, singing might be less cognitively demanding in terms of translation of information from music scores because performers do not have to compile information into a system of black and white piano keys and they do not have to understand this system well enough to do so.

The code reading case study presented in this dissertation used notation of a familiar Bubble sort algorithm that computer science students usually learn at the beginning of their studies. In contrast to previous research (Hansen et al., 2013; Ajami et al., 2019), in the current case study, the goal of the code was known and the surprising patterns introduced in different parts of the algorithm produced a different meaning and lead to a different output, making the experiment closer to the debugging task—when programmers debug a code, they know the goal of the code and look for bugs that do not allow it to reach this goal. This design also allows for the same length to be used in all algorithms. Furthermore, in contrast to previous research (Hansen et al., 2013; Ajami et al., 2019), the algorithm was presented in the form of pseudocode that tries to eliminate specific programming language features. This form of pseudocode was chosen in order to reduce differences between coders that might result from the amount of experience with a specific programming language.

The code study pays particular attention to the choice of the unit of analysis in code reading. This question is particularly important for drawing areas of interest in eye-tracking studies. Two options for creating areas of interest were explored—single elements and lines of code. The issue of finding "a meaningful unit" (Vygotsky, 1934) of language analysis that can also shed light on the thinking behind it is not as straightforward as it might appear upon first glance. Vygotsky (1934) proposed that a word should be a meaningful unit of natural language analysis because it represents a unity of a sound related to speech and a meaning related to verbal thinking.

In code reading, this unit should merge language (sign) and computational thinking (meaning) aspects of programming. It is difficult to decide how a unit of code reading analysis should be defined in, for instance, code that contains such complex code components as method declarations. However, the pseudocode that was used in Article III did not contain such complex components. A line of code might be considered to be a part of language by containing signs and a part of computational thinking by constituting a computational abstraction—that is, it expresses a thought about one action. In incongruent algorithms, the change in this one action (and its meaning) also changes the meaning of the entire algorithm and its output. It can be compared to a single element, which can also be considered to be a part of language and a very simple abstraction. Nevertheless, in contrast to a line of code, it does not express a thought about one computational action. Interestingly, the comparison between the two different units revealed different analysis results regarding the proof-readers' error phenomenon in code reading. In particular, it is possible to overlook that code reading with proof-readers' errors is different from the reading of the original notation by taking into account only single elements of code. Single elements might not maintain the properties of the entire code. Therefore, a code line can be chosen to represent a meaningful unit of code reading analysis for future studies, if this finding is replicated in a larger sample.

Experimental studies presented in Articles II and III proposed enhancements in how eye movement parameters are used to study incongruence in music and code reading. First, the ETS parameter was applied in the music reading study to explore the temporal aspect of incongruent notation reading. Second, the pupil size parameter—associated with cognitive effort—was measured only in first-pass fixations, which are associated with expertise in complex pattern processing (Penttinen & Huovinen, 2011). Finally, saccadic velocity parameters were used as a measure of cognitive effort in the code reading study.

To conclude, experimental studies presented in this dissertation suggest improvements to the design of experimental tasks and to the applications of eye movement parameters for research on incongruence in music and code reading. The case study on code reading also proposes the meaningful unit of code reading analysis. The music reading study provides the first exploration of the translation of information and motor requirements in music reading.

8.3 Expertise and expectations in symbolic language reading

The theoretical study presented in Article I provided a theoretical model for how expectations are created and how prediction works in music and programming. It can be used as a starting point to further study incongruence in symbolic languages—

that is, the violation of expectations in a notation. This theoretical model identifies three similarities (knowledge of language systems, knowledge of meaning, and knowledge of context) and two differences (translation of information as well as temporal and motor requirements) in prediction mechanisms in music and programming.

The experimental studies on incongruent music and code reading presented in Articles II and III primarily focused on two components of the model that are similar for both symbolic languages: knowledge of language systems and knowledge of meaning. Incongruent changes in familiar notations are expressed in different signs. Therefore, they contradict the knowledge of language systems and prevent the recognition of familiar patterns in notations. In terms of the mental schema theory (Détienne & Soloway, 1990), these changes disrupt current mental schemas of experienced readers. In addition to knowledge of language systems, incongruent patterns change the meaning of the notation by making it uncommon. This altered meaning contradicts the meaning that is commonly accepted in the professional community. Both the experimental study on music reading and the case study on code reading revealed that the processing of the meaning of incongruent symbols is reflected through changes in the fixation and cognitive effort parameters, unlike the meaning of congruent symbols. In addition, the code reading study tried to define and use a meaningful unit of analysis in code reading by adopting the criteria for such definition from the sociocultural theory of verbal thinking and natural language acquisition (Vygotsky, 1934) to programming. A code line that expresses a thought about one computational action that affect the entire program is proposed to be the meaningful unit of code reading analysis.

Knowledge of the context component might be considered to be associated with both deliberate practice and knowledge encapsulation. Indeed, a conscious effort, with a goal of performance improvement, is needed to obtain knowledge of a new context. Furthermore, this effort should be spent on relevant tasks and activities. The influence of different contexts on incongruent music and code reading is a topic that needs future scientific investigation. At the moment, there is no existing research regarding this issue in music and code reading.

In contrast to code reading, strict temporal requirements for the processing of incongruence exist in music reading. Each note should be performed in accordance with the music time of the music piece. The application of the ETS parameter that merges music time and eye movements allowed the researcher to investigate temporal aspect of incongruence processing in the music reading experiment. Participants had longer ETS prior to the incongruent part of the notation, which signalized the upcoming difficulties, while their ETS was shorter when they were in the process of struggling with the incongruent portion. In addition, a difference in

the catch-up strategy was registered in the bar after the incongruent one for the more difficult tonality, which was reflected in longer ETS.

Interestingly, the translation of information and motor requirements differs not only between music and programming but also between different music performance modes. The difference in the mode of performance of the same music task associated with translation of information and motor requirements was studied in the music reading experiment by comparing the acts of singing and playing from music scores. Despite the fact that the participants played incongruent melodies better than they sang them, the analysis of their eye movement parameters allowed the researcher to discover that singing might be less cognitively demanding than playing.

To summarize, the proposed theoretical model of prediction in music and code reading is closely linked to expertise theories and it provides a foundation from which to study incongruence and identify gaps for future studies.

8.4 Limitations

The experimental studies of the current research project have limitations. First, it only included an exploratory case study on incongruent code reading, which restricts opportunities to make conclusions about both music and code reading domains. Moreover, the findings of the case study need to be replicated with a larger sample. However, a second study, which includes a sample of eighteen participants who perform the same task, is currently in preparation. That study plans to explore the different ways in which surprising patterns in code reading are handled in accordance with the inserted type of incongruent change. Hence, the study continues the research began in this dissertation, providing more information about incongruence in code reading. Second, the experimental sample for the music reading study included only four males, which might affect the findings. Future studies should explore incongruent music reading on a more gender-balanced sample. Third, there was no full randomization in the music reading study that could influence behavior of participants. Fourth, further work is needed in terms of creating measures for levels of reading skill expertise relative to the domain. In this dissertation, music and programming expertise was measured as number of years, when participants were professionally active, which might not be an accurate measurement of the symbolic language reading skill. Fifth, it is possible that output options influence reading behavior in code reading tasks. For instance, output options that the participant in the case study saw after the first task might have guided his attention in the following tasks by providing a clue of what should be checked. Finally, there are some limitations concerning measuring of the pupil size in the music study. The analyzed half-bars of the melodies included stimuli of different visual complexity (notes of two different durations), which could affect both the mean pupil size in first-pass

fixations and the first-pass fixation duration parameters. In addition, in contrast to singing performances, participants might need to look at the piano keyboard while playing. The difference of luminance level between the stimuli and piano keyboard could also affect the observed differences in pupil sizes between playing and singing (Chitalkina et al., 2021). In addition, baseline differences in the pupil size were taken into account by introducing random intercepts for subjects in the linear mixed models instead of traditional normalization methods such as baseline subtraction, Z-score or percent changes in pupil size (see the description in Bednarik, Vrzakova, & Hradis, 2012). Future studies could also improve the control for the pupil foreshortening error by developing and applying algorithms for its correction. It is also possible that this issue will be fixed by eye-tracker producers, and these algorithms will be embedded in future versions of eye-trackers.

9 Educational Applications and Directions for Future Work

Previous research on music and code reading has either considered them separately or has compared them to the reading of natural languages (see Section 4). The current dissertation project represents the first attempt to look at these two symbolic languages together. It particularly focuses on the reading of incongruent symbolic language notations. Prior studies of incongruent music and code reading primarily explored incongruence in unfamiliar music notations (Ahken et al., 2012; Hadley et al., 2018) and programs with unknown goals (Hansen et al., 2013; Ajami et al., 2019). In contrast, the current project proposed that incongruent changes should be studied within familiar notations, thus contributing to the development and creation of appropriate experimental settings in music and code reading domains. Furthermore, this dissertation study introduced eye-time span and saccadic velocity eye movement parameters to incongruent music and code reading experiments, respectively. The approach employed in the dissertation allowed the researcher to provide an insight into how expectations are created and what happens when these expectations are violated in music and code reading tasks.

The current work informs both music and computer science educators about the processing of incongruent notation while reading. This knowledge is particularly related to issues in music and computing education—such as the switch to a different music genre with different conventions and debugging, respectively. At a more general level, research on incongruent reading allows researchers to find out how anticipation works in experienced readers. A superior anticipation skill is one of the characteristics of expert performance (Abernethy et al., 2018). Abernethy et al. (2018) indicated that particular focus of research on anticipation should be placed on the development of suitable training that could help develop prediction in professional tasks. In music and code reading, an ability to act against expectations is needed in certain professional tasks. Therefore, experimental tasks used in this work might serve as an example of learning exercises that could be developed and used to improve handling of surprising patterns in music (Chitalkina et al., 2021) and debugging.

Reading of source code is not an explicit activity in computing education, and the findings of Article III show that, for code reading, new types of didactic activities can be developed. For instance, research on possible applications of eye movement modeling examples recordings in computing education is increasingly gaining popularity (Bednarik et al., 2018; Emhardt et al., 2020). Eye movement modeling examples combine two channels of information: visual and auditory. It is possible that such a combination can lead to processing difficulties, especially in complicated tasks. Nevertheless, eye-tracking also provides opportunities for a different teaching approach—that of gaze-contingency. The main principle of the gaze-contingency approach is that the content of a screen is changed on the basis of the looking behavior of readers. This principle can also be applied to teach code reading by, for instance, showing an explanation to the part of the program that a reader fixates on for long time. Since research has revealed that readers with different expertise levels prefer to rely on different types of representations while debugging (Romero et al., 2003; Bednarik, 2012), these explanations can also be presented in different forms depending on the level of a reader's expertise. For instance, it is possible that a verbal explanation is more suitable for beginners because it seems that computational thinking develops from verbal thinking. More experienced readers can benefit from the use of more abstract explanation forms. However, in order to develop these kinds of learning materials, more understanding of how computational thinking develops is needed. A further interesting line of future research could be found in the comparison of eye movement modeling examples and gaze-contingency teaching tools.

Current research on music and code reading also tends to analyze only correct trials, excluding reading errors from analyses. In contrast, the experimental studies presented in this dissertation included errors in the analyses. Moreover, the case study on code reading specifically focused on one specific error type—when a reader could not notice a mistake in a notation. In addition, the music reading study compared two different ways of performing the same task: playing on a piano and singing. Despite the fact that the participants played incongruent melodies better than they sang them, eye movement analysis provided evidence that playing from a notation might require more effort than singing. Therefore, in terms of having music reading skills in both singing and playing, playing is not more complicated than singing, as one might think (Chitalkina et al., 2021).

Experimental studies presented here investigated how experienced music readers and one experienced programmer deal with the same type of task in both music and code reading. This research could be extended by comparing how the same participants who are experienced in both symbolic languages perform on similar types of tasks in music and code reading. At the moment, there is a lack of studies that make a direct comparison of reading in these two symbolic languages.

The proposed theoretical model for predictions in music and code reading points out certain gaps in symbolic language research. First, there is a lack of longitudinal research on concept formation and development of abstraction—that is, on music and computational thinking behind music and code reading. Here, longitudinal means a careful investigation of abstraction development at different stages of symbolic language expertise acquisition. Recently, some studies on the development of programming concepts have been conducted using interviews and questionnaires only (e.g., Herman, Kaczmarczyk, Loui, & Zilles, 2008; Kennedy & Kraemer, 2018). In music reading, a combination of eye-tracking and verbal descriptions was used to study the development of music concepts (Penttinen et al., 2013). In natural language acquisition, a deep divergence exists between the formation of a concept and its verbal definition—that is, the analysis of the reality based on the concept appears earlier than the analysis of the concept (Vygotsky, 1934). The same might be the case for symbolic languages. Consequently, experimental studies on how different ways of abstraction are associated with stages of concept development might be a better way in which to study concept formation in symbolic languages. Eye-tracking can provide additional opportunities to obtain insights into cognitive processing during different stages of concept formation in symbolic languages.

This line of research could also contribute to the comparison of natural and programming languages by changing the focus to the comparison between verbal and computational thinking and by discussing how computational thinking develops from verbal thinking. In addition to the practical importance of this issue for the development of new eye-tracking-based teaching tools mentioned above, it remains unclear whether computational thinking goes through similar stages of development as verbal thinking. The current dissertation takes the first step toward such research by discussing a meaningful unit of the code reading analysis. In addition, code reading in real life includes a combination of both verbal and computational thinking because the code includes both comments and a program. A new exploratory study on the incongruence of code and comments (Eiroa-Lledo et al., 2020) revealed that programming and code reading are both probably significantly influenced by verbal thinking of the programming community because expert coders act in accordance with comments even if they are incongruent with the code. Therefore, this line of research should be developed in future studies because it provides important knowledge about the interplay between verbal and computational thinking.

The current code study might further be extended using two different types of pseudocode: one that is closer to natural languages and another that is closer to programming languages. In Article III, only the second type of pseudocode was used. The application of the first type might obtain deeper insights into the differences between natural and programming languages.

Second, the theoretical model indicates that there is a lack of research on the influence of context knowledge on the reading of incongruent symbolic notations. Future studies could investigate how experts deal with incongruent notations in different music genres or in different programming application domains.

To summarize, the current dissertation project presents deeper theoretical and experimental insights into the creation of expectations and what happens when these expectations are disrupted in music and code reading. It suggests experimental settings in which incongruent symbolic language reading can be studied. The proposed theoretical model also allows research questions for future studies to be identified.

List of References

Abernethy, B., Farrow, D., & Mann, D. L. (2018). Superior anticipation. In K. A. Ericsson, R. R. Hoffman, A. Kozbelt, & A. M. Williams (Eds.), *The Cambridge handbook of expertise and expert performance* (2nd ed., pp. 389–412).

Adelson, B. (1981). Problem solving and the development of abstract categories in programming languages. *Memory & Cognition*, *9*(4), 422–433.

Ahken, S., Comeau, G., Hébert, S., & Balasubramaniam, R. (2012). Eye movement patterns during the processing of musical and linguistic syntactic incongruities. *Psychomusicology: Music, Mind, and Brain*, *22*(1), 18.

Ahsan, Z., & Obaidellah, U. (2020). Predicting expertise among novice programmers with prior knowledge on programming tasks. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)* (pp. 1008-1016). IEEE.

Ajami, S., Woodbridge, Y., & Feitelson, D. G. (2019). Syntax, predicates, idioms—What really affects code complexity? *Empirical Software Engineering*, *24*(1), 287–328.

Aschwanden, C., & Crosby, M. (2006). Code scanning patterns in program comprehension. In *Proceedings of the 39th Hawaii international conference on system sciences*.

Bar, M. (2009). Predictions: A universal principle in the operation of the human brain. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, *364*(1521), 1181–1182. https://doi.org/10.1098/rstb.2008.0321

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2014). Fitting linear mixed-effects models using lme4. *arXiv preprint arXiv:1406.5823*.

Bednarik, R. (2012). Expertise-dependent visual attention strategies develop over time during debugging with multiple code representations. *International Journal of Human-Computer Studies*, *70*(2), 143–155.

Bednarik, R., Vrzakova, H., & Hradis, M. (2012). What do you want to do next: a novel approach for intent prediction in gaze-based interaction. In *Proceedings of the symposium on eye tracking research and applications* (pp. 83-90).

Bednarik, R., Schulte, C., Budde, L., Heinemann, B., & Vrzakova, H. (2018). Eye-movement modeling examples in source code comprehension: A classroom study. In *Proceedings of the 18th Koli Calling international conference on computing education research* (pp. 1–8).

Berger, M. (2005). Vygotsky's theory of concept formation and mathematics education. *International Group for the Psychology of Mathematics Education*, *2*, 153–160.

Binkley, D., Davis, M., Lawrie, D., Maletic, J. I., Morrell, C., & Sharif, B. (2013). The impact of identifier style on effort and comprehension. *Empirical Software Engineering*, *18*(2), 219–276.

Boshuizen, H. P., & Schmidt, H. G. (2008). The development of clinical reasoning expertise. *Clinical Reasoning in the Health Professions*, *3*, 113–121.

Boshuizen, H. P., Gruber, H., & Strasser, J. (2020). Knowledge restructuring through case processing: The key to generalise expertise development theory across domains? *Educational Research Review*, *29*, 100310.

Brodsky, W., & Kessler, Y. (2017). The effect of beam slope on the perception of melodic contour. *Acta Psychologica*, *180*, 190–199.

Brych, M, Händel, BF, Riechelmann, E, Pieczykolan, A, Huestegge, L. Effects of vocal demands on pupil dilation. *Psychophysiology*. 2021; 58:e13729.

Bubic, A., Von Cramon, D. Y., & Schubotz, R. I. (2010). Prediction, cognition and the brain. *Frontiers in human neuroscience, 4,* 25.

Busjahn, T., Schulte, C., & Busjahn, A. (2011). Analysis of code reading to gain more insight in program comprehension. In *Proceedings of the 11th Koli Calling international conference on computing education research* (pp. 1–9).

Busjahn, T., Bednarik, R., Begel, A., Crosby, M., Paterson, J. H., Schulte, C., … & Tamm, S. (2015, May). Eye movements in code reading: Relaxing the linear order. In *2015 IEEE 23rd international conference on program comprehension* (pp. 255–265). IEEE.

Casalnuovo, C., Lee, K., Wang, H., Devanbu, P., & Morgan, E. (2020). Do programmers prefer predictable expressions in code? *Cognitive Science*, *44*(12), e12921.

Chase, W. G., & Simon, H. A. (1973). The mind's eye in chess. In *Visual information processing* (pp. 215–281). Academic Press.

Childs, P. E., Markic, S., & Ryan, M. C. (2015). The role of language in the teaching and learning of chemistry. *Chemistry Education: Best Practices, Opportunities and Trends (eds J. García-Martínez and E. Serrano-Torregrosa), Wiley-VCH Verlag GmbH and Co. KGaA, Weinheim, Germany.*

Chitalkina, N., Bednarik, R., Puurtinen, M., & Gruber, H. (2019). Prediction as a prerequisite of skilled reading: The cases of source-code and music notation. In *Proceedings of the 19th Koli Calling international conference on computing education research* (pp. 1–9).

Chitalkina, N., Bednarik, R., Puurtinen, M., & Gruber, H. (2020). When you ignore what you see: How to study proof-readers' error in pseudocode reading. In *Proceedings of the 12th ACM symposium on eye tracking research & applications* (pp. 1–5).

Chitalkina, N., Puurtinen, M., Gruber, H., & Bednarik, R. (2021). Handling of incongruences in music notation during singing or playing. *International Journal of Music Education, 39*(1), 18–38.

Crosby, M. E., & Stelovsky, J. (1990). How do we read algorithms? A case study. *Computer*, *23*(1), 25–35.

Crosby, M. E., Scholtz, J., & Wiedenbeck, S. (2002). The roles beacons play in comprehension for novice and expert programmers. In *PPIG* (p. 5).

Degner, S., Lehmann, A. C., & Gruber, H. (2003). Expert learning in the domain of jazz guitar music. In *Proceedings of the 5th Triennial ESCOM Conference* (Vol. 387).

Détienne, F., & Soloway, E. (1990). An empirically-derived control structure for the process of program understanding. *International Journal of Man-Machine Studies*, *33*(3), 323–342.

Di Stasi, L. L., Antolí, A., & Cañas, J. J. (2011). Main sequence: An index for detecting mental workload variation in complex tasks. *Applied Ergonomics*, *42*(6), 807–813.

Di Stasi, L. L., Catena, A., Canas, J. J., Macknik, S. L., & Martinez-Conde, S. (2013). Saccadic velocity as an arousal index in naturalistic tasks. *Neuroscience & Biobehavioral Reviews, 37*(5), 968-975.

Drai-Zerbib, V., Baccino, T., & Bigand, E. (2012). Sight-reading expertise: Cross-modality integration investigated using eye tracking. *Psychology of Music*, *40*(2), 216–235.

Eiroa-Lledo, E., Bechtel, A., Daskas, E., Foster, L., Pirzadeh, R., Rodeghiero, K., & Linstead, E. (2020). Do experienced programmers put too much confidence in comments? In *International conference on software engineering & knowledge engineering*.

Emhardt, S., Jarodzka, H., Brand-Gruwel, S., Drumm, C., & van Gog, T. (2020). Introducing eye movement modeling examples for programming education and the role of teacher's didactic guidance. In *Proceedings of the 12th ACM symposium on eye tracking research & applications* (pp. 1–4).

Ericsson, K. A. (2016). Summing up hours of any type of practice versus identifying optimal practice activities: Commentary on Macnamara, Moreau, & Hambrick (2016). *Perspectives on Psychological Science*, *11*(3), 351–354.

Ericsson, K. A. (2018 a). The differential influence of experience, practice, and deliberate practice on the development of superior individual performance of experts. In K. A. Ericsson, R. R. Hoffman, A. Kozbelt, & A. M. Williams (Eds.), *The Cambridge handbook of expertise and expert performance* (2nd ed., pp. 389–412). Cambridge University Press.

Ericsson, K. A. (2018 b). Superior working memory in experts. In K. A. Ericsson, R. R. Hoffman, A. Kozbelt, & A. M. Williams (Eds.), *The Cambridge handbook of expertise and expert performance* (2nd ed., pp. 389–412). Cambridge University Press.

Ericsson, K. A., Krampe, R. T., & Tesch-Römer, C. (1993). The role of deliberate practice in the acquisition of expert performance. *Psychological Review*, *100*(3), 363.

Fautley, M. (2017). Notation and music education. *British Journal of Music Education*, *34*(2), 123–126.

Fritz, T., Begel, A., Müller, S. C., Yigit-Elliott, S., & Züger, M. (2014). Using psycho-physiological measures to assess task difficulty in software development. In *Proceedings of the 36th international conference on software engineering* (pp. 402-413).

Friston, K. (2005). A theory of cortical responses. *Philosophical Transactions of the Royal Society B: Biological sciences, 360*(1456), 815–836.

Furneaux, S., & Land, M. F. (1999). The effects of skill on the eye–hand span during musical sight–reading. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, *266*(1436), 2435–2440.

Garber, E. (1999). *The language of physics: the calculus and the development of theoretical physics in Europe, 1750-1914.* Springer Science & Business Media.

Gilman, E., & Underwood, G. (2003). Restricting the field of view to investigate the perceptual spans of pianists. *Visual Cognition*, *10*(2), 201–232.

Goolsby, T. W. (1994 a). Eye movement in music reading: Effects of reading ability, notational complexity, and encounters. *Music Perception: An Interdisciplinary Journal, 12*(1), 77–96.

Goolsby, T. W. (1994 b). Profiles of processing: Eye movements during sightreading. *Music Perception*, *12*(1), 97–123.

Gunter, T. C., Schmidt, B. H., & Besson, M. (2003). Let's face the music: A behavioral and electrophysiological exploration of score reading. *Psychophysiology*, *40*(5), 742–751.

Gupthar, A. S. (2007). *Biochemistry students' difficulties with the symbolic and visual language used in molecular biology* (Doctoral dissertation).

Gromko, J. E. (2004). Predictors of music sight-reading ability in high school wind players. *Journal of research in music education, 52*(1), 6-15.

Gruber, H., Lehtinen, E., Palonen, T., & Degner, S. (2008). Persons in the shadow: Assessing the social context of high abilities. *Psychology Science*, *50*(2), 237.

Hadley, L. V., Sturt, P., Eerola, T., & Pickering, M. J. (2018). Incremental comprehension of pitch relationships in written music: Evidence from eye movements. *Quarterly Journal of Experimental Psychology*, *71*(1), 211–219.

Hansen, M., Goldstone, R. L., & Lumsdaine, A. (2013). What makes code hard to understand? *arXiv preprint arXiv:1304.5257*.

Hayes, T. R., & Petrov, A. A. (2016). Mapping and correcting the influence of gaze position on pupil size measurements. *Behavior Research Methods, 48*(2), 510-527.

Hejmady, P., & Narayanan, N. H. (2012). Visual attention patterns during program debugging with an IDE. In *Proceedings of the symposium on eye tracking research and applications* (pp. 197–200).

Herman, G. L., Kaczmarczyk, L., Loui, M. C., & Zilles, C. (2008). Proof by incomplete enumeration and other logical misconceptions. In *Proceedings of the fourth international workshop on computing education research* (pp. 59–70).

Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H., & Van de Weijer, J. (2011). *Eye tracking: A comprehensive guide to methods and measures.* OUP Oxford.

Huovinen, E., Ylitalo, A. K., & Puurtinen, M. (2018). Early attraction in temporally controlled sight reading of music. *Journal of Eye Movement Research*, *11*(2).

Jacobsen, O. I. (1928). An experimental study of photographing eye-movements in reading music. *Music Supervisors' Journal*, *14*(3), 63–69.

Jarodzka, H., Van Gog, T., Dorr, M., Scheiter, K., & Gerjets, P. (2013). Learning to see: Guiding students' attention via a model's eye movements fosters learning. *Learning and Instruction, 25*, 62-70.

Jbara, A., & Feitelson, D. G. (2017). How programmers read regular code: A controlled experiment using eye tracking. *Empirical Software Engineering*, *22*(3), 1440–1477.

Just, M. A., & Carpenter, P. A. (1980). A theory of reading: From eye fixations to comprehension. *Psychological review, 87*(4), 329.

Katidioti, I., Borst, J. P., Bierens de Haan, D. J., Pepping, T., van Vugt, M. K., & Taatgen, N. A. (2016). Interrupted by your pupil: An interruption management system based on pupil dilation. *International Journal of Human–Computer Interaction, 32*(10), 791-801.

Kennedy, C., & Kraemer, E. T. (2018). What are they thinking? Eliciting student reasoning about troublesome concepts in introductory computer science. In *Proceedings of the 18th Koli Calling International Conference on Computing Education Research* (pp. 1–10).

Kernighan, B. W., & Plauger, P. J. (1974). *Elements of programming style.* McGraw-Hill, Inc..

Ko, A. J., & Davis, K. (2017). Computing mentorship in a software boomtown: Relationships to adolescent interest and beliefs. In *Proceedings of the 2017 ACM conference on international computing education research* (pp. 236–244).

Ko, A. J., Hwa, L., Davis, K., & Yip, J. C. (2018). Informal mentoring of adolescents about computing: Relationships, roles, qualities, and impact. In *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 598–603).

Komogortsev, O. V., Gobert, D. V., Jayarathna, S., & Gowda, S. M. (2010). Standardization of automated analyses of oculomotor fixation and saccadic behaviors. *IEEE Transactions on Biomedical Engineering*, *57*(11), 2635–2645.

Kopiez, R., & In Lee, J. (2006). Towards a dynamic model of skills involved in sight reading music. *Music education research, 8*(1), 97-120.

Längler, M., Nivala, M., & Gruber, H. (2018). Peers, parents and teachers: A case study on how popular music guitarists perceive support for expertise development from "persons in the shadows." *Musicae Scientiae*, *22*(2), 224–243.

Längler, M., Nivala, M., Brouwer, J., & Gruber, H. (2021). Quality of network support for the deliberate practice of popular musicians [Manuscript in preparation].

Lee, S., Hooshyar, D., Ji, H., Nam, K., & Lim, H. (2018). Mining biometric data to predict programmer expertise and task difficulty. *Cluster Computing, 21*(1), 1097-1107.

Liao, H. I., Yoneya, M., Kashino, M., & Furukawa, S. (2018). Pupillary dilation response reflects surprising moments in music. *Journal of Eye Movement Research, 11*(2).

Lin, Y. T., Wu, C. C., Hou, T. Y., Lin, Y. C., Yang, F. Y., & Chang, C. H. (2015). Tracking students' cognitive processes during program debugging—An eye-movement approach. *IEEE Transactions on Education*, *59*(3), 175–186.

Lister, R., Simon, B., Thompson, E., Whalley, J. L., & Prasad, C. (2006). Not seeing the forest for the trees: Novice programmers and the SOLO taxonomy. *ACM SIGCSE Bulletin*, *38*(3), 118–122.

Meinz, E. J., & Hambrick, D. Z. (2010). Deliberate practice is necessary but not sufficient to explain individual differences in piano sight-reading skill: The role of working memory capacity. *Psychological science, 21*(7), 914-919.

Michaeli, T., & Romeike, R. (2019). Improving Debugging Skills in the Classroom: The Effects of Teaching a Systematic Debugging Process. In *Proceedings of the 14th Workshop in Primary and Secondary Computing Education* (pp. 1-7).

Nevala, J., Kairaluoma, L., Ahonen, T., Aro, M., & Holopainen, L. (2007). *Lukemis-ja kirjoittamistaitojen yksilötestistö nuorille ja aikuisille* [Assessment battery for reading disabilities in young and adults]. Jyväskylä: Niilo Mäki Institute.

Obaidellah, U., Al Haek, M., & Cheng, P. C. H. (2018). A survey on the usage of eye-tracking in computer programming. *ACM Computing Surveys (CSUR)*, *51*(1), 1–58.

Olsen, A. (2012). The Tobii I-VT fixation filter. *Tobii Technology*, 1–21.

Orlov, P. A., & Bednarik, R. (2017). The role of extrafoveal vision in source code comprehension. *Perception*, *46*(5), 541–565.

Österholm, M. (n.d.). The role of natural language when learning the symbolic language of mathematics. Retrieved from https://www.umu.se/en/research/projects/the-role-of-natural-language-when-learning-the-symbolic-language-of-mathematics/

Peachock, P., Iovino, N., & Sharif, B. (2017, July). Investigating eye movements in natural language and C++ source code-a replication experiment. In *International conference on augmented cognition* (pp. 206–218). Cham, Switzerland: Springer.

Penttinen, M. (2013). *Skill development in music reading: The eye-movement approach* [Doctoral dissertation, Department of Teacher Education and the Centre for Learning Research, University of Turku]. Turku, Finland: University of Turku.

Penttinen, M., & Huovinen, E. (2011). The early development of sight-reading skills in adulthood: A study of eye movements. *Journal of Research in Music Education*, *59*(2), 196–220.

Penttinen, M., Huovinen, E., & Ylitalo, A. K. (2013). Silent music reading: Amateur musicians' visual processing and descriptive skill. *Musicae Scientiae*, *17*(2), 198–216.

Penttinen, M., Huovinen, E., & Ylitalo, A. K. (2015). Reading ahead: Adult music students' eye movements in temporally controlled performances of a children's song. *International Journal of Music Education*, *33*(1), 36–50.

Perscheid, M., Siegmund, B., Taeumel, M., & Hirschfeld, R. (2017). Studying the advancement in debugging practice of professional software developers. *Software Quality Journal, 25*(1), 83-110.

Puurtinen, M. (2018). Eye on music reading: A methodological review of studies from 1994 to 2017. *Journal of Eye Movement Research*, *11*(2), 1–16.

Puurtinen, M., Anto, E., Ylitalo, A. K., Huovinen, E., Gruber, H., Heinonen, S., & Turta, H. (2019). Eye movements reflect musicians' planning for local embellishments during music reading [Unpublished conference presentation].

Ramalingam, V., & Wiedenbeck, S. (1997). An empirical study of novice program comprehension in the imperative and object-oriented styles. In *Papers presented at the seventh workshop on empirical studies of programmers* (pp. 124–139).

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological bulletin, 124*(3), 372.

Rayner, K. (2009). The 35th Sir Frederick Bartlett Lecture: Eye movements and attention in reading, scene perception, and visual search. *Quarterly Journal of Experimental Psychology*, *62*(8), 1457–1506.

Reifinger Jr, J. L. (2018). The relationship of pitch sight-singing skills with tonal discrimination, language reading skills, and academic ability in children. *Journal of Research in Music Education*, *66*(1), 71–91.

Romero, P., du Boulay, B., Cox, R., & Lutz, R. (2003). Java debugging strategies in multi-representational environments. In *PPIG* (p. 19).

Rosemann, S., Altenmüller, E., & Fahle, M. (2016). The art of sight-reading: Influence of practice, playing tempo, complexity and cognitive skills on the eye–hand span in pianists. *Psychology of Music*, *44*(4), 658–673.

Sharafi, Z., Sharif, B., Guéhéneuc, Y. G., Begel, A., Bednarik, R., & Crosby, M. (2020). A practical guide on conducting eye tracking studies in software engineering. *Empirical Software Engineering*, *25*(5), 3128–3174.

Sharif, B., Falcone, M., & Maletic, J. I. (2012). An eye-tracking study on the role of scan time in finding source code defects. In *Proceedings of the symposium on eye tracking research and applications* (pp. 381–384).

Sheridan, H., Maturi, K. S., & Kleinsmith, A. L. (2020). Eye movements during music reading: Toward a unified understanding of visual expertise. *Gazing Toward the Future: Advances in Eye Movement Theory and Applications, 73,* 119.

Soloway, E., & Ehrlich, K. (1984). Empirical studies of programming knowledge. *IEEE Transactions on Software Engineering, SE-10,* 595–609.

Soloway, E., Ehrlich, K., Bonar, J., & Greenspan, J. (1982). What do novices know about programming? In B. Shneiderman & A. Badre (Eds.), *Directions in human-computer interaction*. Norwood, NJ: Ablex.

Sloboda, J. (1974). The eye-hand span-an approach to the study of sight reading. *Psychology of Music*, *2*(2), 4–10.

Sloboda, J. A. (1976). The effect of item position on the likelihood of identification by inference in prose reading and music reading. *Canadian Journal of Psychology/Revue canadienne de psychologie, 30*(4), 228.

Sloboda, J. (1978). The psychology of music reading. *Psychology of Music*, *6*(2), 3–20.

Timoshenko, M. (2018). Seeing into the music score: eye-tracking and sight-reading in a choral context. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications* (pp. 1-2).

Uwano, H., Nakamura, M., Monden, A., & Matsumoto, K. I. (2006). Analyzing individual performance of source code review using reviewers' eye movement. In *Proceedings of the 2006 symposium on eye tracking research & applications* (pp. 133–140).

Van de Wiel, M. W. J., Boshuizen, H. P. A., Schmidt, H. G., & Schaper, N. C. (1999). The explanation of clinical concepts by expert physicians, clerks, and advanced students. *Teaching and Learning in Medicine, 11*, 153–163.

Van Gerven, P. W., Paas, F., Van Merriënboer, J. J., & Schmidt, H. G. (2004). Memory load and the cognitive pupillary response in aging. *Psychophysiology*, *41*(2), 167–174.

Von Mayrhauser, A., & Vans, A. M. (1995). Program comprehension during software maintenance and evolution. *Computer*, *28*(8), 44–55.

Vuust, P., Dietz, M. J., Witek, M., & Kringelbach, M. L. (2018). Now you hear it: A predictive coding model for understanding rhythmic incongruity. *Annals of the New York Academy of Sciences*, *1423*(1), 19–29.

Vygotsky, L. S. (1934). *Mushlenie i rech* [Thinking and speech]. Moscow, Russia: Socekgiz.

Weaver, H. E. (1943). A survey of visual processes in reading differently constructed musical selections. *Psychological Monographs*, *55*(1), 1–30.

Weiss, M. W., Trehub, S. E., Schellenberg, E. G., & Habashi, P. (2016). Pupils dilate for vocal or familiar music. *Journal of Experimental Psychology: Human Perception and Performance, 42*(8), 1061.

Wolf, T. (1976). A cognitive model of musical sight-reading. *Journal of Psycholinguistic Research*, *5*(2), 143–171.

Wurtz, P., Mueri, R. M., & Wiesendanger, M. (2009). Sight-reading of violinists: Eye movements anticipate the musical flow. *Experimental Brain Research*, *194*(3), 445–450.

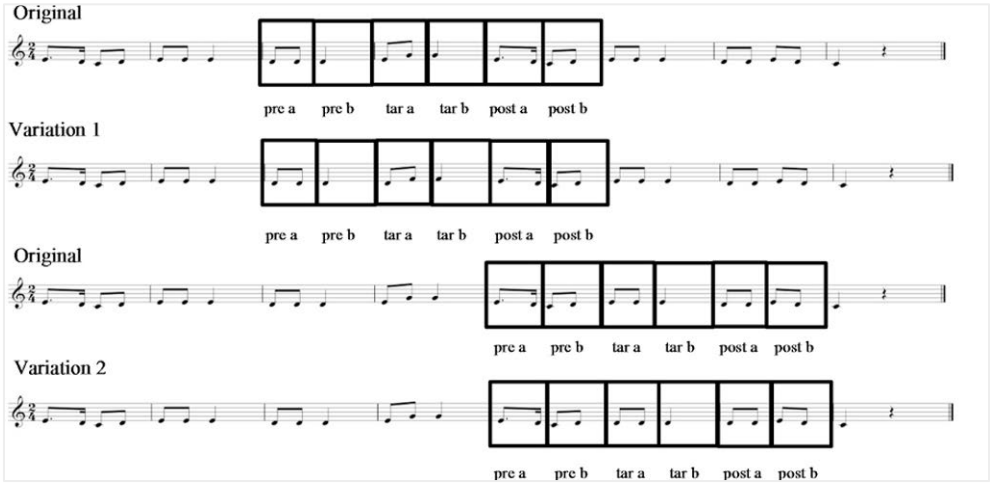
Xie, B., Nelson, G. L., & Ko, A. J. (2018). An explicit strategy to scaffold novice program tracing. In *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 344–349).

Xie, B., Loksa, D., Nelson, G. L., Davidson, M. J., Dong, D., Kwik, H., … & Ko, A. J. (2019). A theory of instruction for introductory programming skills. *Computer Science Education*, *29*(2–3), 205–253.

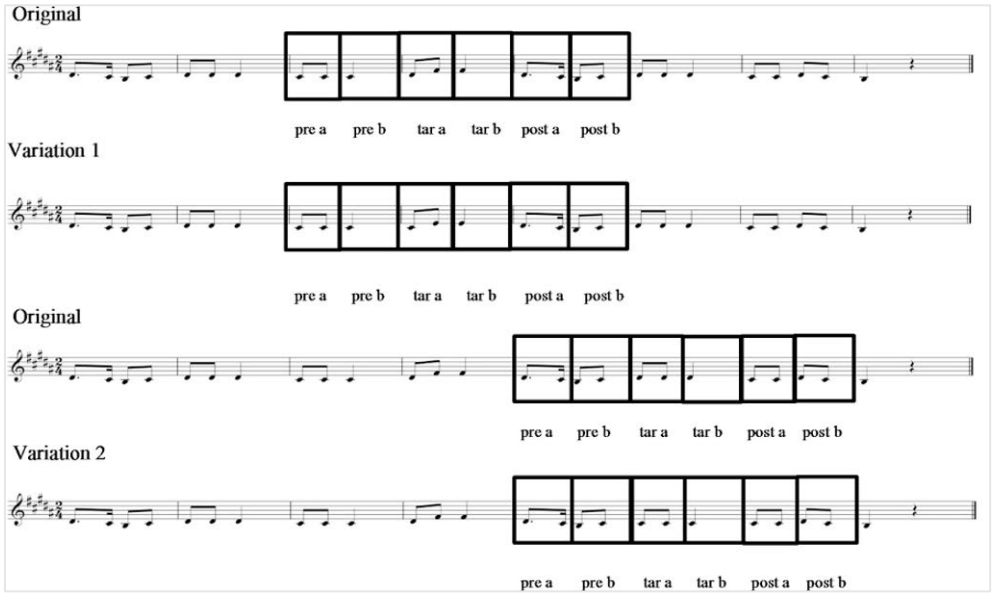
Zemblys, R., Niehorster, D. C., Komogortsev, O., & Holmqvist, K. (2018). Using machine learning to detect events in eye-tracking data. *Behavior Research Methods*, *50*(1), 160–181.

Appendices

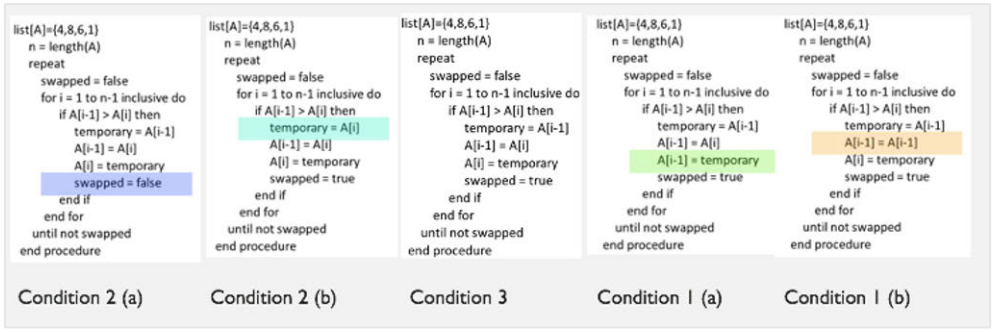
Appendix 1. “Mary Had a Little Lamb” that is used in Article II, composed by L. Mason in C major, and its two variations, composed by M. Puurtinen, with Areas of Interests (AOIs) applied in the data analysis. Source: Article II (Chitalkina et al., 2021)



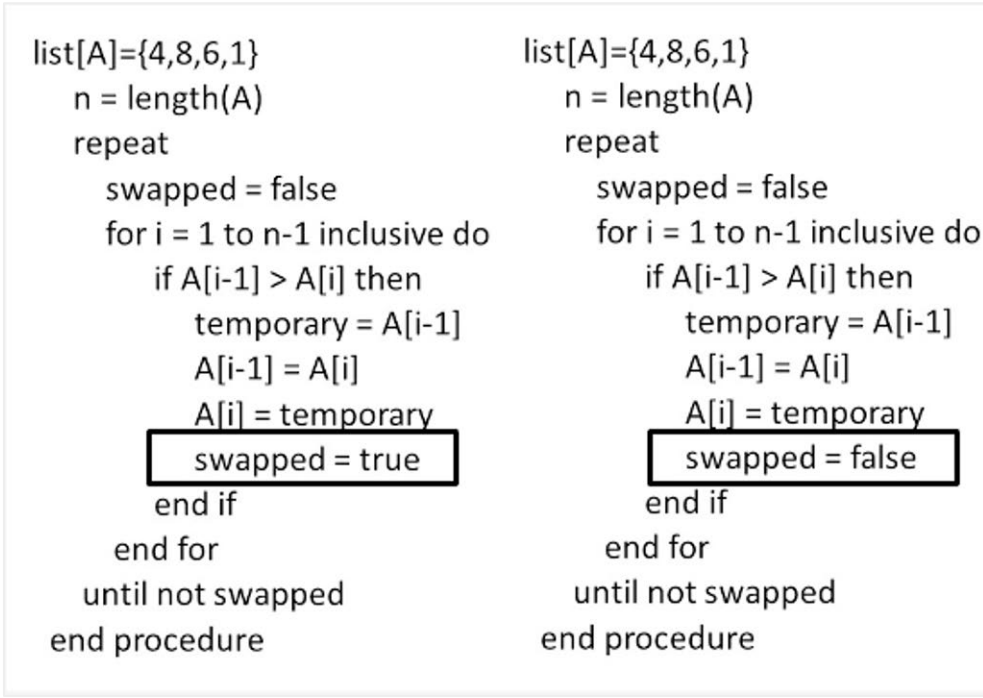
Appendix 2. “Mary Had a Little Lamb” (composed by L. Mason), that is used in Article II in B major and its two variations with Areas of Interests (AOIs) applied in the data analysis. Source: Article II (Chitalkina et al., 2021)



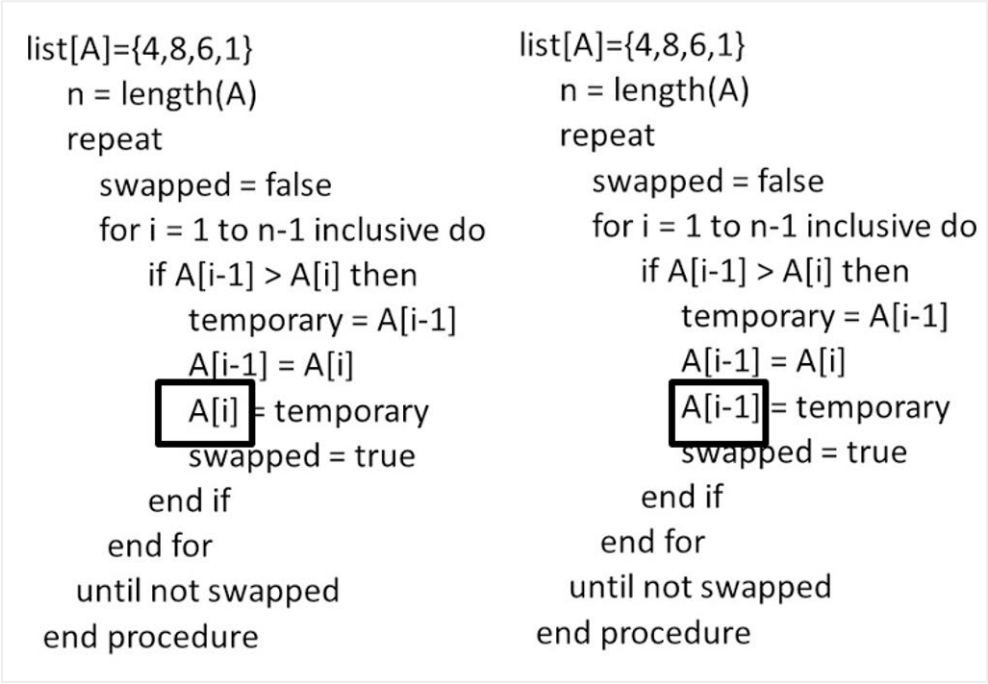
Appendix 3. The Bubble sort algorithm that is used in Article III (Condition 3) with two incongruent algorithms that the participant performed correctly (Conditions 1 a and b) and two incongruent algorithms that the participant performed with proof-readers’ errors (Conditions 2 a and b).



Appendix 4. The Bubble sort algorithms that are used in Article III with areas of interests created around the line containing the incongruent element in the incongruent algorithm (Condition 2 (a), on the right) and the corresponding line in the original algorithm (Condition 3, on the left).



Appendix 5. The Bubble sort algorithms that are used in Article III with areas of interests created around the incongruent element in the incongruent algorithm (Condition 1 (a), on the right) and the corresponding element in the original algorithm (Condition 3, on the left).



TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

Painosalama Oy, Turku, Finland 2021

ISBN 978-951-29-8515-9 (PDF)
ISSN 0082-6987 (Print)
ISSN 2343-3191 (Online)