

VERY NEGATIVE IMAGE EDITING

Regaining creative agency through designing
and developing VeryNeg, a bespoke, standalone
image editing tool for the conversion of scanned
film negatives.



Joonatan Koponen
Master's thesis for
Master of Arts
30 ECTS

Visual Communication Design
Department of Art and Media
School of Arts, Design and Architecture
Aalto University
2026

Tekijä Timo Joonatan Koponen.		
Työn nimi Erittäin negatiivista kuvanmuokkauksta: Luovan toimijuuden takaisinvaltaus filminegatiivien positiiveiksi kääntämistä varten mittatilaustyönä suunnitellun ja kehitetyn VeryNeg sovelluksen kautta.		
Koulutusohjelma Taiteen ja median maisteriohjelma		
Pääaine Visuaalisen viestinnän muotoilu		
Valvoja Prof. Rupesh Vyas		
Työn ohjaaja Heikki Lotvonen, TaM		
Päivämäärä 25.5.2026	Sivumäärä 80	Kieli Englanti

Tiivistelmä

Tämä käytäntöön perustuva soveltava tutkimus tutkii kuvankäsittelyä, filminegatiivien kääntämistä positiiveiksi, taiteellista tulkintaa kuvanmuokkauksessa, työkalujen tekemistä sekä luovaa toimijuutta. Tutkimus tapahtuu ilmaisen ja itsenäisen filminegatiivien kääntämiseen suunnitellun VeryNeg-sovelluksen kehittämistyön kautta.

Tämä opinnäyte käsittelee seuraavia haasteita: Suunnittelutyökalujen sisäisen toiminnan ymmärryksen puute, toimijuuden puute suunnittelutyökalujen luomisessa, liiallinen riippuvaisuus Adobesta, sekä tiedon ja toimijuuden puute filminegatiivin positiiviksi muuntamisessa. Työn motivaationa on kiinnostukseni suunnittelutyökalujen sisäisen toiminnan ymmärtämisestä sekä ohjelmoinnista sekä tarve säästää filminegatiivien kanssa työskentelyyn käytettyä aikaa sekä rahaa.

Opinnäytteen päätutkimuskysymys on: Kuinka rakentaa mittatilaustyönä tehty digitaalinen kuvankäsittelytyökalu skannattujen filminegatiivien positiiveiksi kääntämisen automatisointia varten?

Tutkimus käsittää kirjallisuuskatsauksen filmivalokuvaamisesta, digitaalisesta kuvanmuokkauksesta ja filminegatiivin positiiviksi kääntämisestä, vertailun erilaisten filminegatiivien käsittelyohjelmien välillä, iteratiivisen prototyyppien luomisen ja kokeilun VeryNegin kehitystyön aikana, kehitysprosessin pohdinnan sekä vertailevan tutkimuksen eri filminkäsittelyohjelmien, VeryNegin sekä laboratorion tuottamien kuvien välillä.

Tulokset viittaavat, että kooditasolla tapahtuvaa kuvanmuokkauksta voi ajatella matemaattisena funktiona, joka tuottaa jonkin kuvaajan. On myös olemassa suodattimia sumentamista ja terävöittämistä varten sekä lineaarinen normalisaatio, jota levels-työkalu hyödyntää. Automaattinen valkotasapaino voidaan saavuttaa ”harmaa maailma” -algoritmillä. Kuvan bittisyvyys, gamma sekä värienhallinta ovat tärkeitä kuvanmuokkauksessa huomioitavia asioita.

Produktion perusteella voi todeta, että itselleni mittatilauksena tekemäni työkalun voi luoda C++:n avulla, hyödyntäen OpenImageIO, SFML sekä RmlUi kirjastoja. Adobe Lightroom ja Photoshop sekä Fuji Frontier toimivat tärkeinä inspiraation lähteinä minulle suunniteltuun työkaluun. Mahdollinen arkkitehtuuri suunnittelutyökalulle on Model-View-Controller. Nopean demon tekeminen ennen varsinaista kehitystyötä on hyödyllistä. Tekoäly voi toimia kehittämisprosessissa avustajana.

Filmin positiiviksi kääntäminen on aina tulkintaa. Kahden olemassa olevan kääntämismetodin lisäksi esittelin oman kääntämismetodini. Negatiivin voi muuntaa monella eri tavalla, mutta silti saavuttaa lähes samanlaisen lopputuloksen ja monimutkaisempi kääntämismetodi ei välttämättä ole tarpeellinen. Kääntämisessä tärkeitä asioita ovat: gamma ja lineaarinen avaruus, oranssi filmimaski, filmin ominaiskäyrät sekä mustan ja valkoisen pisteen sekä keskisävyjen väritasapaino.

Vaikka omien työkalujen tekeminen on mahdollista, se ei ole kovin saavutettavaa tai helppoa. Se kuitenkin lisää toimijuutta. Kuitenkin, jopa tässä itsenäisessä työkalussa, Adoben työkalut välillisesti vaikuttivat päätöksiini tottumusten kautta. Filminegatiivin kääntäminen tasapainoilee rajoittavan tarkkuuden sekä yksilöllisten mieltymysten ja taiteellisen näkemyksen välillä.

Avainsanat	Työkalujen rakentaminen, kuvanmuokkaus, filmivalokuvaus, filminegatiivin kääntäminen positiiviksi
-------------------	---

Author Timo Joonatan Koponen		
Title of thesis Very Negative Image Editing: Regaining creative agency through designing and developing VeryNeg, a bespoke, standalone image editing tool for the conversion of scanned film negatives.		
Programme Master’s Programme in Art and Media		
Major Visual Communication Design		
Thesis supervisor Prof. Rupesh Vyas		
Thesis advisor Heikki Lotvonen, MA		
Date 25.5.2026	Number of pages 80	Language English

Abstract

This practice-based applied research explores image processing, film negative conversion, artistic interpretation in image editing, toolmaking and creative agency through the development of VeryNeg, an independent free application for film negative conversion.

The thesis addresses several problems: lack of knowledge on how design tools function internally, lack of agency in the creation of design tools, over-reliance on Adobe and a lack of knowledge and agency in film conversion. It is motivated by my interest in understanding the internals of design tools and programming and the need of saving time and money in film photography.

The thesis investigates the main question of how to build a bespoke digital image editing tool for automating scanned film negative conversion.

The research combines a literature review on film photography, digital image editing and film negative conversion with case studies of existing conversion software, iterative prototyping and experimentation when developing VeryNeg, reflection on the development process and a comparative analysis of conversion results from VeryNeg, existing software and lab scans.

The results indicate that digital image editing on the software level can be thought of as a mathematical function producing a curve. There also exists filtering for blurring and sharpening and linear normalization for the levels tool. For automatic white balance one can use the gray world algorithm. Image bit depth, gamma and color management are important aspects in image editing.

Based on the production, a bespoke tool for my photography practice can be developed in C++ with OpenImageIO, SFML and RmlUI libraries. Adobe Lightroom, Photoshop and Fuji Frontier are important inspirations for a tool for my workflow. A possible architecture for a design tool is Model-View-Controller. Making a quick demo before actual development is useful as is using artificial intelligence as a helper.

Film conversion is always an interpretation. I presented two existing conversion options and my own in addition. You can convert an image in different ways but still obtain almost the same image that is just as usable and that a more complex conversion might not be required. Important aspects to consider in a conversion are: gamma correction, linear space, orange film mask, characteristic curves, black and white point and midtone color correction.

Making one’s own tools is possible, but not very accessible or easy, but it does increase agency. However, even in this independent tool, familiarity with Adobe’s tools affected my decisions. Film conversion is a balance between restrictive correctness and individual preferences and artistic vision.

Keywords	Toolmaking, Image editing, Film Photography, Film negative conversion
-----------------	---

Contents

1. Introduction.....	1
1.1 Motivation and Research Problem.....	1
1.2 Methodology and Research Questions.....	2
1.3 Thesis Structure.....	4
2. Literature Review.....	6
2.1 Film Photography.....	8
2.1.1 A word on Light and Human Perception.....	8
2.1.2 Overview of film structure.....	8
2.1.3 Orange color mask.....	9
2.1.4 Film stocks and looks.....	9
2.1.5 Film negative as a raw image.....	10
2.1.6 Darkroom printing.....	10
2.2 Digital Image Editing.....	12
2.2.1 Digital Image.....	12
2.2.2 Color Management and ICC Profiles.....	13
2.2.3 Image Editing.....	13
2.3 Film Negative Conversion.....	18
2.3.1 Scanning.....	18
2.3.2 Issue of Correctness.....	19
2.3.3 Characteristic Curves and Gamma.....	19
2.3.4 Film Negative Conversion Approaches.....	20
2.4 Literature Review Summary.....	24
3. Case Studies of Scanning and Conversion Software.....	26
3.1 Negative Lab Pro.....	27
3.2 FilmLab.....	29
3.3 SmartConvert by Filmomat.....	31
3.4 Negfix8.....	32
3.5 Darktable Negadoctor.....	33
3.6 Fuji Frontier.....	36
3.7 Summary of the Case Studies.....	38
4. Production of VeryNeg.....	40
4.1 Production Description.....	41
4.1.1 Goals for VeryNeg.....	41
4.1.2 Choosing Technologies.....	42
4.1.3 Development Process.....	43
4.1.4 VeryNeg Overview.....	45
4.2 Reflection and Findings from the Production.....	49
4.2.1 Image processing and Algorithms.....	49
4.2.2 Tool Building.....	55
5. Comparison of Conversion Results.....	60
6. Conclusion.....	71
6.1 Summary.....	71
6.2 Findings.....	72
6.3 Reflection.....	73
6.4 Future.....	74
7. References.....	77
7.1 Image References.....	79

Introduction

1.1 Motivation and Research Problem

As long as I can remember, I have been fascinated by how things work. I love to learn how something works or how it's made, whatever it is. I have also always enjoyed building, drawing or designing things, to create something new. I think that this desire inside me is the driving force that led me to be a designer in the first place. It has also led me to study how to do things myself to gain more agency and to maybe save some money.

Lack of knowledge on how design tools work under the hood

As visual communication designers, we do most of our work with digital tools these days, but do we actually know how those tools work? In my quest to learn more about doing things myself and understanding how things work, I took on computer science as my minor to learn how computers and our graphic design tools work under the hood and to learn how to program. As a result I fell in love with programming and the prospect of automation. Before this, we had had some programming courses in our studies, where I remember being told that learning these principles you might be able to develop your own tools and not be dependent on big corporations and paid software. I think that a lot of designers can relate with the hope of not having to be dependent on monthly subscriptions and the idea of automating some repetitive aspects of one's practice or to have a say on the functionality of our tools to tailor them to each of our specific needs. But how possible is that? Based on my observations, knowledge of the inside workings of our tools could be improved and it's not a focus of many designers. For example, many of my fellow students (me included) struggle with understanding what color management even means. What else would we have to learn to actually make our own tools?

Lack of agency in the creation of design tools and the over-reliance on Adobe

Many design tools that we use have largely remained the same throughout the years. For example, Photoshop still resembles its first version quite much. Many design software tools emerged as research subjects on computer science. They later evolved into tools for graphic design but the decisions on their design were largely influenced by the existing standards of the industry. They were not created by designers and were mostly driven by decisions of computer scientists or the existing graphic design industry and

its needs. Thus, they inherited a lot from pre-existing workflows. After the advent of the personal computer, digital tools designed by designers started to emerge, but that has not shaken the industry standard defined by Adobe with their tools that haven't changed much. (Maudet, 2017)

For a profession that is centered on creativity, I find it interesting that professionally we have almost only one serious option for the digital tools that we use. Adobe has had a dominant position in design for a long time (Zeidenberg, 1996). Fluency in Adobe's design programs is expected for almost every design job and you won't be taken very seriously if you don't have experience with Adobe software even if you are a genius in using open source design apps such as GIMP or Inkscape. Due to this general unrivalled dependency of Adobe, they are able to price their applications really high and they have even made us more locked in with a yearly subscription model.

Adobe's applications are closed source, meaning the logic under the hood of the apps is hidden and a secret. This is in contrast to open source tools which are usually free and, where anyone can make copies of the software and customize it to their liking if they know how. Open source works on the basis of volunteerism and is shaped by the interests of the volunteers, not requirements of the industry or shareholders. (Ingo, 2005) Not being able to really own and tinker with your tools, as is the case with closed source applications, might prove a hindrance to designers: According to Maudet (2017), designers use many different design processes shaped by their way of working or the specific task at hand. Sometimes a designer would want to do something, but the tool they have available doesn't allow them to do it and they have to either change their original idea or to try to duct-tape their way around it. (Maudet, 2017) Blender is a good example of how an open source design program can become a tool to be taken seriously (Blender, 2026).

Lack of open knowledge and agency in film conversion

In addition to being interested in how things work, photography has been a dream and a hobby of mine from an early age. During the past few years I have started to do paid client work as well. When living abroad for a few years, I discovered film photography and vintage cameras. I really enjoy the unique aesthetics of film: The dynamic range, the grain, the halation and the organic feel and slow process associated with it.

But film has its downsides. Being an outdated process and technology, it has become expensive. The prices of film rolls themselves have gone up and also the development and scanning processes are expensive. Due to my thrifty "do it yourself" attitude

I don't want to pay for a lab to scan my negatives and have thus resorted to scanning the film myself. During the process, I have, however, noticed that scanning and editing the photos takes up a lot of time. Even after a lot of searching, I have struggled to find definitive answers on how to edit a scanned color negative to convert it to a positive Image. Due to this, I have struggled to make the colors right and to get perfectly pleasing results.

To speed up my scanning and conversion process, I have tried to automate the film editing process in Photoshop using photoshop actions. The process saves some time, but still requires a lot of manual editing from my part. On a general level, a lot of tools offer some options to customize and automate them, but you are still tied to that specific program. On the other hand a lot of the programs feature so many features that most of us might not use and can feel overwhelming or complicated to use the first time you open them. Due to both of these reasons, I don't particularly enjoy the process of converting my negatives in Photoshop and thus, a lot of my negatives are still sitting in their protective sleeves, waiting to be seen. However, I still insist on not spending the money to get them scanned at the lab or buying film conversion software. So they still sit in my drawer, waiting to be revealed to the world.

To conclude, I have thus identified three problems: Lack of knowledge on how design tools work under the hood, Lack of agency in the creation of design tools and the over-reliance on Adobe and Lack of knowledge and agency in film conversion.

1.2 Methodology and Research Questions

In this thesis, I will approach the problems mentioned above through practice-based applied research. I explore the possibilities of actually understanding, creating, owning and tinkering with the tools that we designers use to gain more creative agency. I realize that as hinted at in our small programming courses, that the potential tools that one could make themselves were probably small helper tools that could enhance one's workflow or artistic practice, but in this thesis I would like to dig a bit deeper. Can you actually build your own tools that you would use seriously as part of your workflow? Tools that are open source, free, customizable, and not dependent on Adobe or other subscription models? Tools that would be designed specifically for you by you? In this thesis I combine my interests of film photography, image editing and programming together to create an independent

film negative conversion tool VeryNeg to show how creating one's own design tools is possible and how it increases a designers agency. VeryNeg also provides a solution to my dilemma of scanning, converting, and editing film negatives myself.

I have chosen three lenses to use throughout this thesis: Technical image processing, artistic interpretation in image editing and toolmaking and agency.

The lens of technical image processing means that I focus on the technical aspects of image processing, the functions, math and algorithms used. A key part of visual communication design is how images and design are handled digitally. This research goes into a deep dive into what image processing and editing is and even though here I discuss it in relation to photography and especially film photography, I would argue that the principles can be applied and might be useful for many different aspects of visual communication design.

To balance the technicalities, I also approach image editing through the lens of artistic interpretation of images. This means that due to the analog nature of film, I also focus on different approaches to negative conversion, different interpretations of the process and the effect of different technicalities used and by the comparison of different results and by reflecting on the feelings they evoke in me.

The third lens of toolmaking and agency works on the larger scale of the assessment of tools and their understanding. I look critically at Adobe and consider my individual preferences but also look at how to share agency and power to individuals and communities. The topic of finding alternative methods and tools to suit one's specific workflow and not having to rely on Adobe's tools is important and current in today's design landscape. Even though I will not get into the micro level details of programming in this thesis, I will still provide overviews on the topics of software technologies, architecture and algorithms.

In my opinion, creative authorship is the reason why we as designers should understand and challenge the tools that we use. And nowhere else is authorship more strongly expressed than in tools that we design and create wholly by ourselves. For

such tools, I would propose the term bespoke tools. Such tools open up creative authorship, but also creative accessibility. They increase our agency in our work and craft.

By looking at the problems through the lenses just listed, in this thesis I will try to answer the following research question and sub-questions:

How to build a bespoke digital image editing tool for automating scanned film negative conversion?

- How does digital image editing work on the software level?
- How to develop a personal tool for my photography practice?
- How to correctly convert a film negative, what does that mean or is there a correct way?

To answer the questions, I will use multiple methods: A literature review was chosen to establish a solid technical understanding of the topics so that I can correctly develop a software application tool for converting scanned film negatives. The topic of film photography is essential to understanding how images form on film and how film is traditionally interpreted. This is essential for both technical processing and artistic interpretation. Image editing is also essential knowledge for software that works with scanned negatives, since after scanning they are digital images and have to be treated as such. It is an essential topic to increase the knowledge of internal workings of tools and to answer the research questions. Since it has been hard for me to find definitive answers about existing methods of film conversion, I want to find more sources and to analyze them on the actual topic of film negative conversion which is the core part of the application and a direct source to answer the research questions.

Case studies of existing conversion software were chosen as a part of the contextual study in applied situations and to gain inspiration for the tool, VeryNeg that I develop in the production part.

In order to fully understand the creation of an application, it has to be done in practice. Through iterative prototyping and experimentation when developing VeryNeg. I study how image editors are built and try to implement my own version of an image conversion algorithm. I document & communicate the choices of technologies and architecture used and then reflect on my findings from the whole development process.

A comparative analysis of visual results from VeryNeg and the other software from the performed case studies was performed to inspect different conversion results and to investigate different interpretations of film negatives. A preliminary example of this is seen in image 1.

1.3 Thesis Structure

This thesis is divided into six main chapters: this introduction, a literature review, case studies of conversion software, production of VeryNeg, a comparison of conversion results and the conclusion.

Chapter 2 consists of the literature review. Section 2.1 covers the topic of film photography: what film is, how it behaves to light, how it produces different colors, what affects its look and how it is printed in the darkroom. Section 2.2 discusses digital image editing, I will dig deep into image editing and what happens on the pixel level of common operations. The section covers topics of: digital images in general and their properties, color management and some image editing algorithms.

The final section 2.3 of the literature review combines both of the earlier topics of film photography and image editing and handles the topic of film negative conversion. I research what film scanners

and conversion software do when converting the negative. The section covers a common conversion method and then discusses more advanced methods based on the physical properties of the film's density and its characteristic curves. It also ponders on the correctness of a film conversion.

After the literature review, in chapter 3, I present case studies and comparisons of some existing film negative conversion software to learn how they approach the subject of conversion and to analyze their user experience.

The production part of this thesis is covered in chapter 4. Section 4.1 covers the design and creation of VeryNeg, a free, open source standalone image editing application designed and developed entirely by me for me. I describe the production, provide an initial plan and a review of the development process and timeline of the application. I present the final application VeryNeg.

In section 4.2 I reflect on my findings during the development process on the topics of image processing and algorithms used in VeryNeg and tool building aspects. The reflection tries to find answers to film negative conversion algorithms by practically implementing them and the same for edits required for the negatives. I also reflect on designing the user interface and how to organize such a big project software and usability of AI.

In chapter 5 the culmination of the production is a comparative representation and analysis of results from VeryNeg the software listed in the case study of tools and a film lab.

The final chapter 6 is the conclusion, where I provide answers to the research questions and reflect on the findings, listing limitations and providing ideas for further research.

I sincerely hope that all the information gained through this research will be useful to others. I also hope to show that it is possible to design and to create your own custom and independent bespoke digital tools that increase our creative agency. I also hope that I will be able to finally uncover some hopefully beautiful film photos to the world from that drawer of mine using a bespoke tool that I made for myself.

[Image 1] One negative, multiple interpretations. An excerpt taken from the comparison of different conversions from chapter 5.

“Experienced designers know that off-the-shelf, general software tools obscure the potential of software as a medium for expression and communication. Writing custom, unique tools with software opens new potentials for creative authorship”

(Reas, 2012, as cited in Maudet, 2017, p. 16).



2.

Literature Review

Before examining the topic through prototyping and iterative testing by diving deep into developing the film negative conversion application VeryNeg, I will cover the theory of essential topics for the functionality of the application in this section of the literature review. Since the final application I will develop in this thesis focuses on scanned film negatives and digital image editing performed on them, the literature review focuses on the general topics of film photography and digital image editing. Due to the fact that most film conversion software is closed source and their algorithms are kept secret, I have previously not found concluding answers on the algorithms used in film negative conversion. Because of this I will try to uncover some of those secrets by covering the topic of film negative conversion.



2.1 Film Photography

I will start the literature review by examining the process of film photography and how it records light and colors. This chapter establishes important considerations, connections and differences between human vision and film, discusses the film's layered structure, and ponders why it has an orange mask. It ponders the question whether film negatives have a look to them and how they can be interpreted. It provides an overview of the traditional way of acquiring a finalized image from a film negative by the means of darkroom printing. These aspects are fundamental for later understanding of the negative conversion process and its correctness. This thesis will not touch on positive film nor black-and-white film, but it will focus on color negative film specifically. This chapter will not go into all specific details of film but tries to pick the parts that are important to consider when scanning and converting the negative image to a positive digitally.

2.1.1 A Word on Light and Human Perception

To understand how an image is formed onto photographic film or even into a digital photo, we must first understand light. I won't go that much into the scientific details of the physicality of light, but will cover some properties of light that are important to the image-forming and interpretation process.

The intensity of light behaves linearly. What that basically means is that to get double the amount of light, you double the amount. If you want double the amount of light of one light bulb, you take two light bulbs, if you want to double that, you take another two light bulbs with a total of four. (Filmmaker IQ, 2018) That's quite simple math. How we perceive light is, however, different and our eyes don't perceive light linearly.

Human eyes do not see light linearly, as for example a digital sensor would. When the amount of light doubles, we do not perceive a doubled amount. Our vision adjusts our interpretation of the scene by showing just a slight increase. Human vision adjusts the highlights to not appear so bright as they should in relation to the shadows. The shadows appear lighter than they are in mathematical terms if one were to measure the absolute amount of reflected light. (Filmmaker IQ, 2018; Grey, 2004; Poynton, 2018; Gonzalez & Woods, 2018; Case, 2001) One could say that we have an internal image processing algorithm that dynamically adjusts our perception based on the scene.

Light is formed from different wavelengths of electromagnetic radiation. We as humans define visible light as the part of that radiation that we can see, that is the visible spectrum of light. Different wavelengths of light are perceived as different colors by our eyes. This ranges from red in the long wavelengths through orange, yellow, green, cyan, blue and finally to purple. (Langford, 2000; Grey, 2004; Sharma, 2018; Gonzalez & Woods, 2018; Kodak, 2007; Hirsch, 2015)

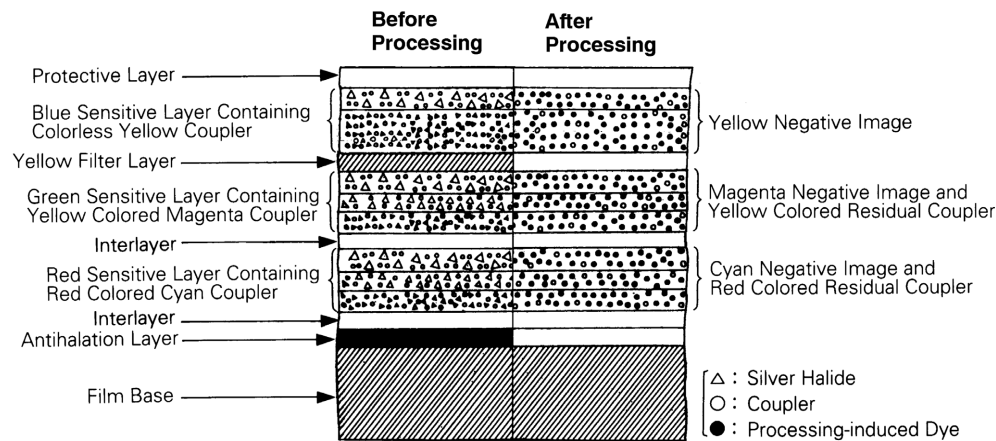
We humans also don't have vision cells for each specific color or wavelength of light. For sensing color, we only have three main cells in our eyes that respond to the incoming light and its wavelengths. They are mostly sensitive to specific areas of wavelengths that could be separated into short, medium and long, or blue, green and red. The final color that we see is then formed by the amount of stimulation that these cells receive, corresponding to the amount of the specific wavelength received. Thus it can be concluded that all colors can be represented with just three primary colors of red, green and blue. (Gonzalez & Woods, 2018; Hirsch, 2015; Kodak, 2007; Case, 2001) As we will see, this will be a fundamental principle for understanding color photography, both digital and film, and also the scanning process and the digital storage and image editing processes of images.

2.1.2 Overview of Film Structure

Film photography is an analog process. The basis for film is the photochemical process provided by silver halide crystals. These crystals react to light and when developed with specific chemicals, form dyes that represent the photographed scene. These crystals determine how sensitive the film is to light, they also create the grain that we associate with film images. (Hirsch, 2015; Langford, 2000; Case, 2001; The Cine Network, 2021; Kodak, 2007)

Negative film records an inverted image of the scene and it is used for projecting onto photographic paper that contains the inverse dyes of the negative. When the paper is developed, it forms a positive and final image as it would have been in real life. (Hirsch, 2015)

Early experimentations of color photos relied on recording three different black and white images of the scene separately using different masks for red, green and blue, that were then overlaid on top of each other in a complex projection system to show a color image. The current way of forming a color photo relies on this same process on the basic level, but is more sophisticated. Instead of taking three separate images with different colored filters, the separate



[Image 3] Film structure of Fujicolor C200 showing the different layers and their composition before and after processing.

filters and images are included inside the film as filter layers and silver halide emulsion layers. (Hirsch, 2015) As seen in image 3, these filters and emulsion layers are laid on top of each other inside the film so that you can just expose one photo, but it will expose those three separate layers at the same time. You are still basically taking three black and white images with different filters on top of them but everything happens “automatically”.

As image 3 shows, in reality the film structure is a bit more complex but for the purposes of this research, only the general idea is needed. The layers that we will focus on next are the silver halide emulsions that individually react to red, green and blue light. The blue-sensitive emulsion reacts to blue light but forms yellow dye, the green-sensitive emulsion reacts to green but forms magenta dye and the red-sensitive layer forms cyan dye (Hirsch, 2015; Langford, 2000; Case, 2001; Kodak, 2007, The Cine Network, 2021).

The dye layers react to light in a way that the more it is exposed, the more dye is formed, thus forming a negative image. Thus when talking about film negatives, the brightness of the image is usually referred to with density. The denser the image, the more it is exposed. The dyes are also the inverse of the color of light that it reacts to, thus also creating an inverse image in the color spectrum. (Kodak, 2007; The Cine Network, 2021; Case, 2001)

The selection of the film reacting to red, green and blue light is based on the theory that just as the human eye can recognize millions of colors with just cells that sense blue green and red light, we can also record color information using those main colors (Hirsch, 2015; Kodak, 2007). There are some films that include more color layers however, but these have not become the main standard (Hirsch, 2015).

2.1.3 Orange Color Mask

As everybody who has ever handled color film negatives has probably noticed, the negatives are not a direct inversion of the original image, but look orange. It is really hard to see any other color in the film than orange. The reason and structure of this overall orange color has been puzzling to me for a long time. According to multiple sources (Case, 2001; Taylor, 1983; The Cine Network, 2021; Dorsky, 2024; Kodak, 2007) the reason for the orange color is to actually help in accurate color recording and reproduction. The different filters do not actually filter all of that specific color and some of it bleeds to the other layers, affecting the color balance of the formed image. The orange color comes from a faint positive image of the specific color recorded in the layer in addition to the negative image. Thus it records the amount of “leaking” light of that specific color and when the image is compensated for the overall orange tint, the leaking color is removed from the other layers and the colors are thus accurate. (Case, 2001; Taylor, 1983; The Cine Network, 2021; Dorsky, 2024; Kodak, 2007)

2.1.4 Film Stocks and Looks

Different film stocks have been designed to give different results. They react to light in subtly different ways, creating slightly different images or colors. The amount of this effect is argued, since a film negative is never the final representation of an image, but rather an intermediate step and the color or look of the image can be edited extensively both in darkroom printing and digitally. Some argue that the

differences between film stocks are quite negligible (Buchler, 2023), but they do exist (PARVEC, 2025). We have a lot of film emulations for digital images that are supposed to give the feeling of a certain film stock being used, but again a lot of this can be adjusted in post processing (Fujifilm, n.d.-a; Alchemy Color, 2024). My personal experience is that different film stocks do give different results since the physical reactions are a bit different, but I would also argue that some of this has been exaggerated and emphasized in post production. A lot of the final images we see are either scanned or printed, and then scanned and this process affects a lot how the final image looks as we will further explore in this section and in section 2.3.

2.1.5 Film Negative as a Raw Image

We can now continue to the related topic of how a color negative film could be described as a medium between processes. A negative is not a final image, but rather a way to capture and store image data in order to obtain a finalized image (Verbeeck, 2025). A lot of this could be applied as a metaphor for a digital raw image file as a so-called “digital negative”. A raw camera file is just the raw data seen by the camera sensor. It’s not a final image in itself. It has to be interpreted to become an image. And it can be interpreted in many different ways. The interpreter can be the camera itself, an automatic program or a human editing it via a program. In the same way a film negative has to be interpreted to become a final image (Verbeeck, 2025). As we will cover next it can either be interpreted by the printing process, or the scanning process.

If you just shine a light through a film negative, you won’t see a final image. Everything you do to the film negative in the process of converting it into an image we are used to seeing involves some interpretation. The backlight used affects the result, so does the chemicals in the paper, the filters between the light and the negative, the length of exposure, the development of the paper. The same applies for scanning. (Alchemy Color, 2024)

One can try to do the whole process with calibrated equipment, color-managed workflows, according to the data acquired by the producer of that specific film stock to be as scientifically accurate as possible but it is still an interpretation. Thus I would argue that a film photo that we see has gone through a lot of interpretation and a single negative could be used to produce many kinds of different images as we will later see.

2.1.6 Darkroom Printing

The traditional way of acquiring a finalized photograph from a film negative is by making a positive print using a film enlarger and photographic paper. A film enlarger is basically just a film holder in front of a backlight. Just as the film itself is sensitive to light, the photographic paper is as well. Just like the negative, it has to be developed in order to activate the dyes that form the final image. However, unlike the negative, the paper produces a positive image. The printing paper’s color sensitive layers also have the corresponding response to the negative so that it removes the orange mask automatically. Many images will still often need additional color correction that is done by adjusting cyan, magenta and yellow filters on the film enlarger. Exposure can also be adjusted with the intensity or length of the exposure of the print paper. (Verbeeck, 2025; Taylor, 1983; Koraks Tinkers, 2023; Hirsch, 2015)

Often the papers have a specific way of reproducing the colors and contrast and thus a paper has a specific look (Hirsch, 2015). These looks may be emulated in film scanners, such as the Fuji Frontier. (Vandebroek, 2026) The printing process is traditionally out of necessity strongly tied to film photography in general and both have influenced each other (Kodak, 2007). We observe that once again the final look of a photograph is affected by a multitude of factors and how it is an interpretation of the original scene and that it is really hard to define the absolute correct way how a final photo from a film stock should look like. The backlight, the filter, the paper, the development process, all once again, influence the look of the photograph. And then in the darkroom, further creative image editing can be applied. Image manipulation didn’t start with digital software, but it began already in the darkroom with basic dodging and burning (Taylor, 1983, Hirsch, 2015) and then other more creative methods that allowed image manipulation and artistic departure from the negative (Stone, 2016).



2.2 Digital Image Editing

We have now familiarized ourselves with the properties of the analog medium of film. To eventually perform a digital conversion of a scanned negative, it is essential to look into the topic of digital images and the methods used to manipulate them in image editing. This is also directly a part of answering the research question of how does digital image editing work on the software level? I would claim that in order to build tools or just to use pre-existing tools to their full potential, it helps to know how they work internally.

This chapter establishes an overview of digital images, how they are stored using different channels and color spaces and bit depths. It also covers the sometimes hard to understand topic of color management, which is essential to maintain accurate color in a pipeline. Finally we take an in depth look into how digital images are edited using mathematical functions or algorithms for common edits such as exposure, levels, curves or white balance. The specific edits are chosen to help in the creation of the tool VeryNeg.

2.2.1 Digital Image

A digital image usually refers to a raster image. There are also vector images, but for photographic purposes and the purposes of this thesis we will exclusively be talking about raster images.

A raster image is defined by pixels in a rectangular grid. It has a specific width and height. Pixel (picture element) represents a defined point of an image. Each pixel stores data about the image for that particular region of the grid. (Foley et al., 1996; Gonzalez & Woods, 2018; Solomon & Breckon, 2001) A pixel can be binary, meaning on or off, or it can store one channel of information for gray-scale images, or it can store multiple channels of information for color images. It can even store more channels such as transparency or depth (Solomon & Breckon, 2001; OpenEXR, n.d.)

How many values each channel can store is called the bit depth of the image. When we increase the bit depth, we can store more shades per pixel and channel. (Solomon & Breckon, 2001; Gonzalez & Woods, 2018) This allows for more extensive editing of the image since it contains more information that we can shift around. However, it also increases the size of the image file (Walker, 2006).

Bit Depth per Channel

Bit depth refers to the amount of information that each pixel can store. Digital images are stored in a binary format using bits. (Solomon & Breckon, 2001; Gonzalez & Woods, 2018; Walker, 2006) When you increase the bits used to store the information of a channel, you increase the amount of different values or shades that each channel can store. An eight bit channel can store 256 values ranging from 0 to 255.

$$\begin{array}{ccc}
 \text{R} & \text{G} & \text{B} \\
 8\text{bit} & 8\text{bit} & 8\text{bit} \\
 \downarrow & & \\
 \text{R} & \text{G} & \text{B} \\
 256 & 256 & 256 \\
 \downarrow & & \\
 256 * 256 * 256 = & & \\
 16\,777\,216 \text{ colors} & &
 \end{array}$$

[Image 4] A visualization of how many colors an RGB image with 8bits per channel can store.

8 bits per channel is a common format for storing images for viewing since it can store 256 values in three different colors meaning a total of 16 777 216 different color shades as seen in image 4. If we increase the depth of each channel to 16 or 32 bits we can have much, much more information. This is important for image editing, since it allows for making much more edits to the image without reducing the image quality (Walker, 2006). An image with 16 bits per channel can store 65 536 different shades per channel versus the 256 of 8 bits.

When the channel is stored using 32 bits, the values are usually stored using floating point arithmetic. I won't go into detail on how floats work, but floats can store decimal data and they are designed to be accurate with smaller numbers. So we can store numbers such as 0.3456 or 0.9994 and do precise mathematical calculations and edits without losing information. 32 bit images don't clip at 1.0 and often may contain data that is above 1.0 for bright sources of light. (OpenEXR, n.d.)

The term bit depth regarding images often refers also to the total amount of bits that all the channels use together, so an image where each channel stores 8 bits, is usually also referred to as an 24-bit image (8bits per channel). For the purposes of this thesis I will be referring to an image's bit depth as bit depth by channel.

2.2.2 Color Management and ICC Profiles

In his book, *Color Confidence* (2004), Grey explains color management quite well. Below is my explanation, based on his book:

If you were to ask three people to look at an apple and to choose a specific red color and to paint a picture of the apple, you would probably get three different paintings. In the same way if you asked the people to take a digital image of the apple with their smartphones the image would look different on each of the phones screen if they were a different smartphone model. This is all really simplified and there is also much more at play here, but this is a good example, where a color management workflow is important if one wants to have consistency between different devices and even different programs inside of the device. (Grey, 2004)

If we show a digital image with RGB pixels on a screen and then on another screen, both without color management, the images will look different due to the actual LEDs or LCD crystals used on the pixels on the screen. The screens have no way of knowing how a red RGB(200, 2, 6) should look like. Color profiles are made so that each device and application “knows” how a specific color should look like and can try to show it like we expect. This happens because each device is calibrated with their own color profile and then we know how to convert from one color space to another by maintaining consistency. (Grey, 2004)

To over simplify, If a camera sensor sees the apple it might record the red as RGB(189, 4, 9). It, however, knows that we expect this color to look like RGB(200, 2, 6) in sRGB color space or RGB(171, 9, 14) in AdobeRGB and so when saving the image the

[Image 5] Left: Image with a gamma correction of 2.2
Right: A linear image without gamma correction looking dark



camera decides to write the color as RGB(200, 2, 6) but then embeds the information into the file that this is using the sRGB profile. When the smartphone then shows the image to the screen, it knows what a sRGB red of RGB(200, 2, 6) should look like, but in addition it knows that its screen is a bit greener so it increases the red and converts the color to its native display profile where the final values are RGB(212, 1, 5) in the display profile. (Grey, 2004)

2.2.3 Image Editing

Two easy to understand visuals in relation to image editing are the curves editor and the histogram. The curves editor represents the input values on the x-axis and the output values on the y-axis. If no edits are performed, the correlation between the two is a straight line. A value of 30 on the image corresponds to a value of 30 after the edit. If we want to brighten the darkest parts of the image, the curve would first grow quickly but then continue as an almost straight linear line the rest. If we don't want the whites to blow out, we would still want the line to end so that the brightest values are still the brightest values. So in this case an input value of 5 could output a value of 30 for example but a value of 70 could output a value of 75 for example and a value of 230 a value of 232. (Walker, 2006) The concept of the curves editor is quite fundamental to image editing and many image editing operations can be thought of along these lines: Some function that mathematically edits the image's pixel values, called an intensity transformation (Gonzalez & Woods, 2018).

The histogram, on the other hand shows the darkest values on the left and the brightest on the right usually in a bar chart and the height of the bars shows the amount of pixels in the image in that brightness. Sometimes the histogram is shown at the same time behind the curves in the x axis in order to help edit the curves. (Walker, 2006) I will next try to cover some of the basic image editing operations and algorithms that might be important for the application.

Linearity and Gamma

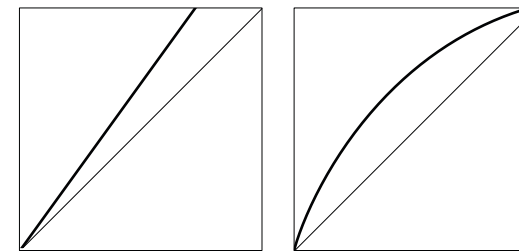
As discussed earlier in the chapter on film photography, the amount of light intensity increases linearly. This means that if we want to double the light we need to double the intensity. But we humans don't perceive the doubling since our eyes and brain adjust the perceived brightness and compress the bright amounts. We just observe a small increase. (Filmmaker IQ, 2018)

Images meant to be displayed have undergone a gamma adjustment. A gamma adjustment compresses the highlights and raises the shadows, thus emulating the way we humans would see the scene with our eyes. (Poynton, 2018) In image editing, if we want to emulate the physical nature of light, however, we have to deal with images that do not have that gamma adjustment but are linear. This simplifies some of the math in the image editing functions. (Alchemy Color, 2024). Image 5 shows examples of an image with and without a gamma correction

Intensity Adjustments:

Brightness and Exposure:

Brightness and exposure might mean the same thing or they may not in different contexts. The point of both however is to increase or decrease the light intensity of the image. Brightness in its most simple form is usually just adding the same value to each pixel, thus brightening the image, like Photoshop's legacy brightness edit (Adobe, 2026a). Exposure in its simple form is usually multiplication as can be seen from Photoshop's exposure edit. (Adobe, 2026a). These edits can be more sophisticated, for example a curve that preserves the highlights and shadows better if the image already has a gamma adjustment applied like Photoshop's current brightness edit (Adobe, 2026a). Examples of both curves can be seen in image 6.



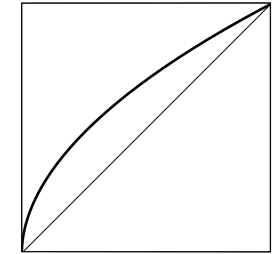
[Image 6] Two ways to edit brightness/exposure. Left: Multiplication exposure adjustment that clips whites. Right: A highlight preserving brightness adjustment.

Gamma Adjustment:

Gamma adjustment in its simplest form is a power function, meaning rising the input value to a power and forming a curve. Depending whether we increase or decrease gamma, the power function brightens the dark areas while keeping white at 1 thus compressing the highlights but not blowing them out to

pure white or it compresses the darks and darkens the highlights if it's increasing the gamma. (Gonzalez & Woods, 2018)

The Gamma correction function is applied to images in order to organize the data according to the logarithmic way humans see the world and to accompany the gamma adjustment required by the display. It redistributes the data inside the image to have more precision on the darks, where precision is more important due to the way we see. This is especially important for low bit depth images where the amount of different tones is limited. (Poynton, 2018)

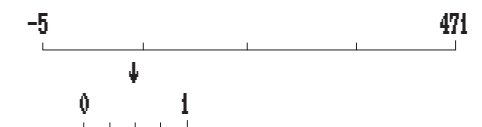


[Image 7] Example of a gamma correction power curve adjustment showing how the shadows are lifted and the highlights compressed.

Levels:

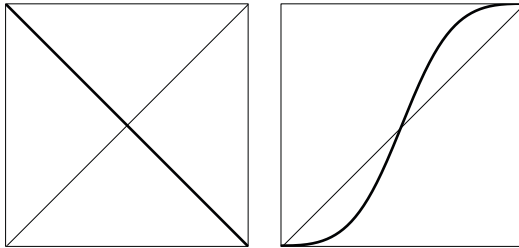
What I want to refer to here, it's easiest to refer to Photoshop's levels tool. This is practically taking the brightest values and darkest values and then remapping them to another range. If for example the darkest value is -5 and the brightest 471, and we would want the brightest areas to be fully white and the dark areas to be black then we would scale all the other values linearly so that the new darkest and brightest output values would be 0.0 and 1.0. (Walker, 2006) A visualization of this mapping can be seen in image 8. We can apply the levels adjustment to individual color channels to make sure that the brightest areas are actually white and the darkest black, if we want to (Lin & Tretter, 2004). In the curves tool this would mean that the input and output points would be moved on the x axis. A more general term for the action performed by the levels tool is called linear normalization (Fisher et al. 2003).

[Image 8] Linear normalization example between ranges.



Simple Invert:

A simple invert in terms of the curves tool, is that the lowest value on the input would become the brightest on the output and the brightest the lowest. This basically looks like an inverted line as seen in image 9.



[Image 9, left] Inversion curve.

[Image 10, right] Sigmoid function curve that decreases the darks and increases the brights, increasing contrast.

Contrast:

A gamma adjustment does affect the contrast of an image, but also the brightness. A traditional way of applying contrast is with a curve representing the character S. This can be achieved with a sigmoid function as shown in image 10. (Molina-Pérez & Rojas-López, 2024).

White Balance Algorithms:

White balance algorithms try to correct the color channels' intensities so that a neutral gray or white color in the image looks properly neutral and doesn't have a color cast, this will then also correct other colors of the image as well. The main problem with automatic white balance is determining what in the image should be the reference point for gray. White balance algorithms can be learning based or statistical. Learning, meaning that the algorithm has been trained beforehand with a dataset so that it can predict how an image should be adjusted and statistical that it just uses data that it can retrieve from the image that is being edited. (Koskinen, 2025) Due to the fact that I will want to implement some auto white balance myself for the production part, I won't be looking into the more advanced learning based algorithms, which could be a thesis on its own.

Here again we face the problem of what is "accurate" white balance? We can measure the conditions using a color checker and a calibrated workflow, but then again the photographer might have a different vision on what is a correct white balance.

Someone might think that the colors should be warmer or maybe colder. A white balance algorithm can be of help though and maybe it can be used as a basis but then to allow manual override to the artist if the algorithm gets the colors wrong.

Gray World Algorithm:

The gray world algorithm assumes that the average color of a scene is gray and then adjusts the intensities of each channel accordingly. At first a full average color from all of the pixels is calculated. After this a scaling factor for each channel is calculated, meaning the ratio between the full averaged and the average of an individual channel. Then each channel is scaled according to this ratio. The result is an image where the average of all pixels is assumed to be gray. If the scene is illuminated by one light source, this can give decent results, but if multiple differently colored light sources are present or if the actual color of the light source should be warm or cold or there is a big area of brightly colored objects such as greenery or forest, this may not balance the colors accurately. (Serteep, 2024)

Manual Neutral Patch Selection:

White balance can also be calculated from a manually sampled neutral area, like in Lightroom (Adobe, 2026b). A manual neutral patch white balance probably uses the gray world algorithm or some other algorithm with the data gained from that small sample area and then applies the scaling factors to the whole image. This should guarantee accurate white balance, but requires a user to determine which part of the image is neutral so it's not fully independent.

Grayness Index Algorithm:

This algorithm tries to look for pixels in the image, where the three channels are close to each other in value, meaning the area probably represents a neutral grey. It then calculates an illuminant color estimation from those pixels and then scales the color channels accordingly. (Qian et al., 2019)

2.3 Film Negative Conversion

This final section of the literature review combines the knowledge gained from the earlier sections on film photography and digital image editing. We are now approaching a tangible core problem that this thesis and the application VeryNeg tries to solve. How to correctly convert a film negative, what does that mean or is there a correct way?

As described earlier, the traditional way of acquiring a positive image from a negative is by printing an image in a darkroom by shining light through the negative onto photosensitive paper that will form a positive image. Another way to obtain a positive image is by shining light through the film and scanning it to a digital format and then performing the conversion to a positive image digitally with software (Kodak, 2007, Hirsch, 2015). Due to the rule of digital content at this point in time, scanning is usually the main way of obtaining and sharing a photo from a film negative. However, it seems like film negatives and print paper are practically married to each other and can be thought of as a system (Kodak, 2006). Digital scanning is like a third wheel trying to get in between them, but struggles since they are so used to each other. Due to this analog marriage of film and darkroom paper, obtaining a digital scan from a negative seemed quite hard for me at first. Some of the algorithms are trade secrets that seem hard to figure out the “proper way” as has been noted not only by me but others as well (Buchler, 2023) I have tried to find a formula for the conversion and will next look into what is needed, to obtain a digital negative scan from both hardware and software points of view. I will also discuss problems related to the process and if we can know how a converted film negative image “should” look like.

From the point of view of this research, film scanning and conversion can be split into two parts, the physical scanner hardware and the digital software. With scanning I refer to the process of acquiring a digital scan from the physical negative. With negative conversion, I refer to the digital image processing done by either the scanner itself or on a computer in the software, to get a pleasing digital positive image out of the scanned digital negative image.

2.3.1 Scanning

The choice of a scanning device and technique plays a big role in the final obtained image. The aspects of a scanner that affect the scan colors are: The backlight that will shine through the negative, the sensor that will be used to record the scanned color data of the negative and the file quality of the scanner output and any color processing that the scanner does. (Buchler, 2023; Alchemy Color, 2024) For the scanner backlight, it seems like the best light is a light that has distinct spikes in the red, green and blue frequencies that correspond to the different layers of the film negative (Alchemy Color, 2024). Some professional scanners such as the Fuji Frontier for example take three different scans with separate R, G, B, backlights (linusandhiscamera, 2023; McDougall, 2024). The sensor also plays a big role on the color accuracy and latitude and also the sharpness and resolution of the scan. The sensor can be a specifically designed sensor internal to a scanner such as a professional or dedicated film scanner or a flatbed scanner. It could also be a digital camera. The sensor, in combination with the internal software of the scanner also determines how the colors are captured into digital format. (Alchemy Color, 2024) Since the focus of this thesis will be on the software part of the scanning process, I will not go into detail regarding these aspects and how they affect the final outcome. I include them here even though our software cannot change them but we need to still acknowledge that they do have a strong influence in the final outcome.



2.3.2 Issue of Correctness

Due to the many steps required in the development, scanning and conversion process, it can be practically impossible to fully determine how the final image is supposed to look like. All steps of the scanning and conversion process affect the final outcome (Buchler, 2023, Captured by Sam, 2024, Alchemy Color, 2024). With all of these dimensions, the same negative can have many drastically different outcomes and determining the correct colors of a negative is really hard (Noriega et al., 2001; Alchemy Color, 2024). An example of many interpretations of a negative is shown in image 11. It has been said that a correct conversion can only be determined with darkroom printing (Buchler, 2023, Noriega et al.). But then again, there isn't a single truth to the correct print either (Buchler, 2023; Noriega et al., 2001; Tuijn, 1996; Alchemy Color, 2024; Verbeeck, 2023).

There are, however, some steps that can be taken to determine a base image based on the actual recorded data of the film, that should present a decent automatic conversion. There are different possible approaches to obtaining a usable final conversion, depending on whether one wants to be physically accurate or just to emulate the process.

For either approach, we need to take into consideration the characteristics of the negative as discussed in section 2.1 on film photography that have to be taken care of in the conversion: The

[Image 11] A screenshot from a video from Captured by Sam (2024) showing variations of the same negative from different scanning techniques and people editing the photo.

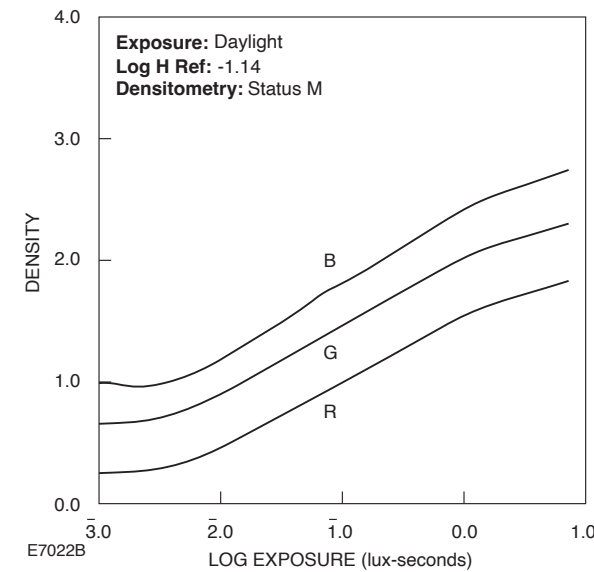


negative is a negative in the luminance and color aspects, that means that dark values represent light values, but also cyan represents red, magenta represents green and yellow represents blue. The negative also has a general orange color cast due to the masks in the blue and green layers compensating for the colors bleeding into other layers. The values are also not black or white but lie somewhere along the range in between.

2.3.3 Characteristic Curves and Gamma

The way that film records light is represented by the film's characteristic curves. Understanding this helps us to understand film's response to different intensities better but it also helps us to understand how the conversion process should work and to maybe figure out a "correct" conversion.

The characteristic curves come from densitometry. As seen in image 12, on the y-axis, the values represent density, which behaves logarithmically. On the x-axis, the values represent a logarithm of the original light intensity, being thus also logarithmic. The characteristic curves have a "toe" and a "shoulder", meaning that the curve starts flat for low intensities and then slowly becomes a straight line before flattening again for very bright intensities. For the linear section, the curve's slope is called gamma. If the slope is smaller than 1, it means that the curve



[Image 12] The characteristic curves of Kodak ColorPlus 200 film stock.

compresses intensities, meaning it has a high dynamic range. This happens in log-log space. (Kodak, 2006; Tuijn, 1996)

If we compare what actually happens in linear space, we can see why the straight line's slope is called gamma. A line in log-log space becomes a power curve in linear space as shown in image 13. As described in section 2.2 on image editing algorithms, this is the function that's associated with gamma correction. So a film's characteristic curve with a gamma smaller than 1, which it usually always is for negative films, raises the shadows and compresses highlights. This all allows a film to be really sensitive to small changes in the shadows, but not to blow out with highlights. As described earlier, this also corresponds to the way a human eye works. And it allows you to capture a wide dynamic range (Kodak, 2006).

2.3.4 Film Negative Conversion Approaches

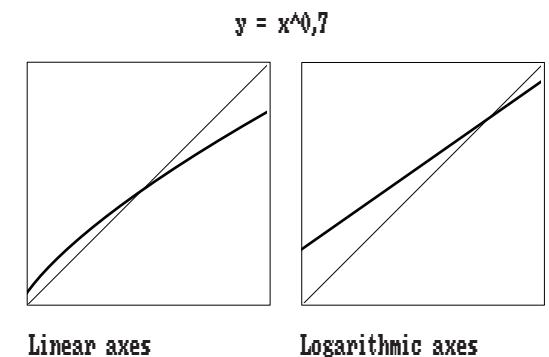
Based on my research, I will split film conversion into two categories. The first category is approaches that are not much based on maths, physics, or graphs but rather instinct and trial and error on Photoshop. These are often found on tutorials and are simple

for anyone to do using an image editor. The second category is based on the characteristic curves and thus the physical properties of how film records light and mathematical functions.

Simple Approaches from Tutorials:

Most tutorials online that I have come across show a variance of the same approach. This approach does not intentionally care about the logarithmic relationship of density and exposure. The approach can be summed up as follows: You first balance the white and black point, this is done by scaling the channels individually so that each channel's darkest pixel corresponds to a specified black value and the brightest pixel to a specified white value. This is done with either the levels tool or the curves tool. For the darkest point you could also just subtract a layer filled with the color of the film border. After this a simple inversion is performed, meaning white becomes black and black becomes white: 0 becomes 1 and 1 becomes 0. The inversion can either be performed before or after the balancing of the black and white points. After setting the black and white points, white balance is also adjusted either by adjusting each channel's curves or the overall color balance in the midtones so that the black and white points stay fixed but the curve of the channels is changed in order, to for example, remove blue by adding yellow. (Burke, 2019; Tran, 2019; RetroPixel, 2018; Santo, 2024)

[Image 13] A gamma adjustment power curve in log-log space compared to the same curve in linear space.



More Complex Approaches:

Finding sources on how actual film scanners or dedicated film conversion software do their conversion was really hard and quite nonexistent. It seems to be a trade secret that people want to keep secret as noted by Buchler (2023) also. Only a few sources describe a full algorithm in detail. The steps are also often quite hard to understand correctly.

In their paper, Lin & Tretter (2004) take a very similar approach to their algorithm as this simple approach. The approach is visualized in image 14. They do a simple invert, then based on the scan data, remap the black and white points of each channel to be almost white and almost black to avoid effects of noise and clipping. The white point is chosen with protection from extreme colors, such as for example if the brightest point is not white but actually a bright blue light. Then the target white point is changed to a more neutral target than the actual measured value. After this step they adjust contrast and midtone color balance with precomputed lookup tables. (Lin & Tretter, 2004)

Tuijn (1996) describes a more complex approach that is based on the characteristic curves of film described earlier. The method is visualized in image 15.

First step would be to expose a test gradient on the film stock to be scanned of different exposures from black to white and then to measure the characteristic curves from that test gradient (Tuijn, 1996).

When measuring densities we need to remember that the actual value the scanner produces is not density, but represents simply the measured light passing by the film. The measured transmission is converted to the logarithmic density by the formula:

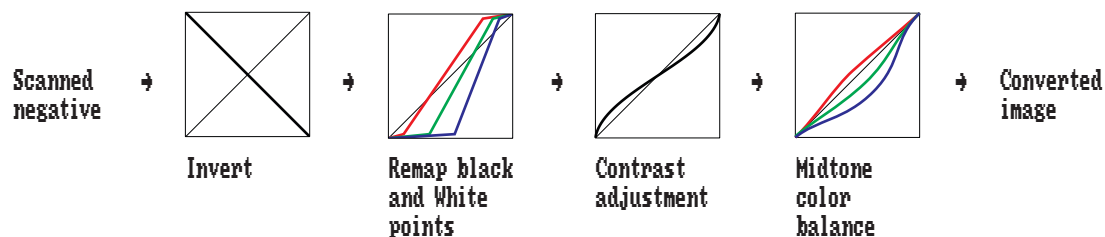
$$\text{density} = \log_{10}(1 / \text{scanner_transmittance})$$

which comes from the formula:

$$\text{transmittance} = \text{original_intensity} / \text{passed_through_intensity}$$

(Noriega et al., 2001)

[Image 14] Visualization of the conversion pipeline of Lin and Tretter (2004).



Continuing with the algorithm proposed by Tuijn (1996), then based on these characteristic curves, we would solve, which density value gives the largest log-exposure value, meaning which density translates to the actual brightest value. If it would happen that for example the brightest density would be 2.3 from the red channel from a log-exposure value of let's say 4, we would then balance all the other density channels based on that log-exposure value of 4. We would solve, using the characteristic curves, which density a log-exposure value of 4 would give from the blue and green channels. And we would mark that as the corrected brightest density. We would then repeat this process by figuring out the darkest log-exposure value and the darkest density and correct the other channels' densities. (Tuijn, 1996) As I understand, this is effectively balancing the black and white points to equal pure black or white, same as in the simple approach. The difference in this, however, is that it takes into account the fact that each channel has different characteristic curves that have different slopes and the toe and shoulder might be in slightly different places or amounts. It also takes into account the actual measured values in the conversion so that the final values represent actual intensities.

The rest of Tuijn's (1996) method is straightforward: After this balancing, we convert the densities of each channel to log-exposure using the characteristic curves and then to the actual exposure values by converting from log to linear and thus obtain the original intensities of the final image (Tuijn, 1996).

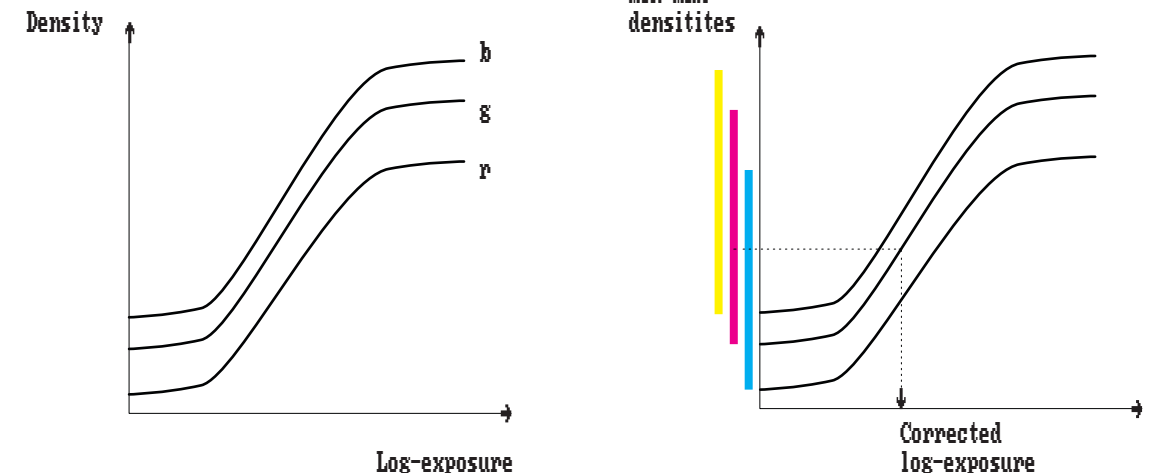
Buchler, 2023 (2023) chooses to ignore the curve parts of the characteristic curves, and simply convert from scanned intensity to density and then from log-exposure to exposure. This he does because for most of the time, the characteristic curves represent a line. (Buchler, 2023) With this combined, the result is basically:

$$\text{exposure} = 1 / \text{scanner_transmittance}$$

If we inspect the code of the open source Negfix8 script, we see that it also approaches the conversion with the same formula (JaZ99, 2013). Then Buchler (2023) balances the colors by setting the camera white balance to the film border and then adjusting the slopes of each "characteristic curve" line in the density to log-exposure space so that the white and black points match. (Buchler, 2023)

For me it's interesting how many different approaches and confusion there is to the conversion. It's also frustrating. Since this deals with moving from one coordinate space to another and logarithms and physics etc. it was quite hard to understand each formula and then to understand why they do it differently. I will further test these and reflect more on them in the production section when I compare the different approaches when developing my own negative conversion application VeryNeg.

[image 15] Visualizing Tuijn's method. Left: Characteristic curves for each color channel. Top: Measuring the maximum densities and corresponding log-exposures and choosing the largest log-exposure. Middle: Correcting other channels' max densities with the largest log-exposure value. Bottom: Correcting the lowest densities the same way and then acquiring the corrected positive image using the characteristic curves.





2.4 Literature Review Summary

The goal of this literature review was to provide a solid basis upon which to build a film conversion software. I set out on my research journey by looking at everything through the lenses of technical image processing combined with artistic interpretation and creative agency and independence. The literature review was mostly focused on the balance of technicality and artistic interpretation. I will next provide an overview of the most important findings to the two relevant research questions:

How does digital image editing work on the software level?

In section 2.2, Digital image editing, I went through important image editing aspects. Many basic image editing actions, such as intensity and color channel adjustments can be understood with curves produced by mathematical functions that are then applied to individual pixels. The functionality of the levels tool can be achieved with linear normalization. Bit depth is an important consideration to avoid losing quality of the image being edited. I also identified options for auto white balance algorithms: gray world, neutral patch and grayness index.

How to correctly convert a film negative, what does that mean or is there a correct way?

in sections 2.1 and 2.3, Film Photography and Film Negative Conversion. I highlighted that in the film pipeline, interpretation happens at every step and is affected by the tools and technology used, choices and personal preferences. I established that a negative is a raw image that is interpreted. Film negative conversion can be darkroom printing or scanning to a digital form. Important aspects to consider in the conversion are the orange film mask and the characteristic curves. I highlighted a general simple inversion from tutorials and more advanced conversion methods.

3.

Case Studies of Scanning and Conversion Software

I have now covered the theory of film photography, digital image editing and film negative conversion in the literature review in chapter 2. As a final step before developing a film negative conversion application in chapter 4, I will examine a few pre-existing commercial closed-source and free software options for film negative conversion. With the technical details of image processing as a prevalent lens in the literature review along with the artistic interpretation in image editing, in this comparison I now introduce more of the lens of toolmaking and agency when assessing the tools.

Case studies of relevant software help to get a more practical assessment of the conversion process and its variations and the usability of tools and their functions. In the case of closed-source software, we can inspect and guess how it performs the conversion technically. In the case of open source, we can try to look into the code for its implementation of the conversion.

The goal of assessing these applications is to perform a contextual study of film conversion in applied situations and to gather inspiration for my own application, picking the functionality I like. Trying out different applications is an essential step to answer the question of How to develop a personal tool for my photography practice?

Before going deep into any of the tools listed below, I will lay out guidelines on how I am assessing the tools. The main focus is on the conversion of TIFF scans from a flatbed scanner. This decision is due to the fact that I don't have a good setup for scanning negatives with a digital camera and thus I don't have camera raw files to process. Due to this the final tool that I will be developing and documenting later in this thesis is also made to process TIFF files from a scanner. Thus, to compare the software, we will judge it from its ability to convert tiff negatives. At this point, I'm mainly assessing my observations from using or observing the tool. The negative conversion results from some of these tools will be compared later in the thesis in chapter 5.

The aspects of the tools that I will be assessing and comparing are the following:

- ▮ User interface and user experience
- ▮ Editing options
- ▮ Conversion approach
- ▮ Preliminary conversion results
- ▮ Closed- or open source, paid or free
- ▮ The overall happiness it gives to me during use

3.1 Negative Lab Pro

I would argue that Negative Lab Pro (Nate Photographic, 2024) is quite famous among modern film photographers. It was the first film conversion method I heard about. It's been around for quite a long time and it has a lot of users. You can find a lot of information or tutorials around it.

Negative Lab Pro is not a standalone program, but a plugin for Adobe Lightroom. It works for both camera raw files and TIFF files. Negative Lab Pro at the time of writing itself is a one-time purchase, but since it's a Lightroom plugin, you have to have a creative cloud subscription of Lightroom in addition to Negative Lab Pro. (Nate Photographic, 2024) It's not open source, so we don't know for certain what happens behind the scenes when it converts a negative.

The user interface could be improved in my opinion, functionality wise and also design wise. I like that I can have a lot of control, but at first there are almost too many options to choose from that it's almost distracting. It's also distracting to have some settings where you have no idea what they do. For example, a pre-conversion saturation or a color model before conversion and then a preset after the conversion. There are also so many different white balance algorithms/presets. But once you use the plugin for some time, these start to make more sense luckily. The user interface also feels ugly and not really intuitive to use. The whole thing feels a little like a hack, working in some ways around Lightroom. The idea of having presets is cool, but then again that and the fact that you have so many options, makes it sometimes hard to choose what you want to do.

My guess on the basic inversion that Negative Lab Pro does is that it does the simple

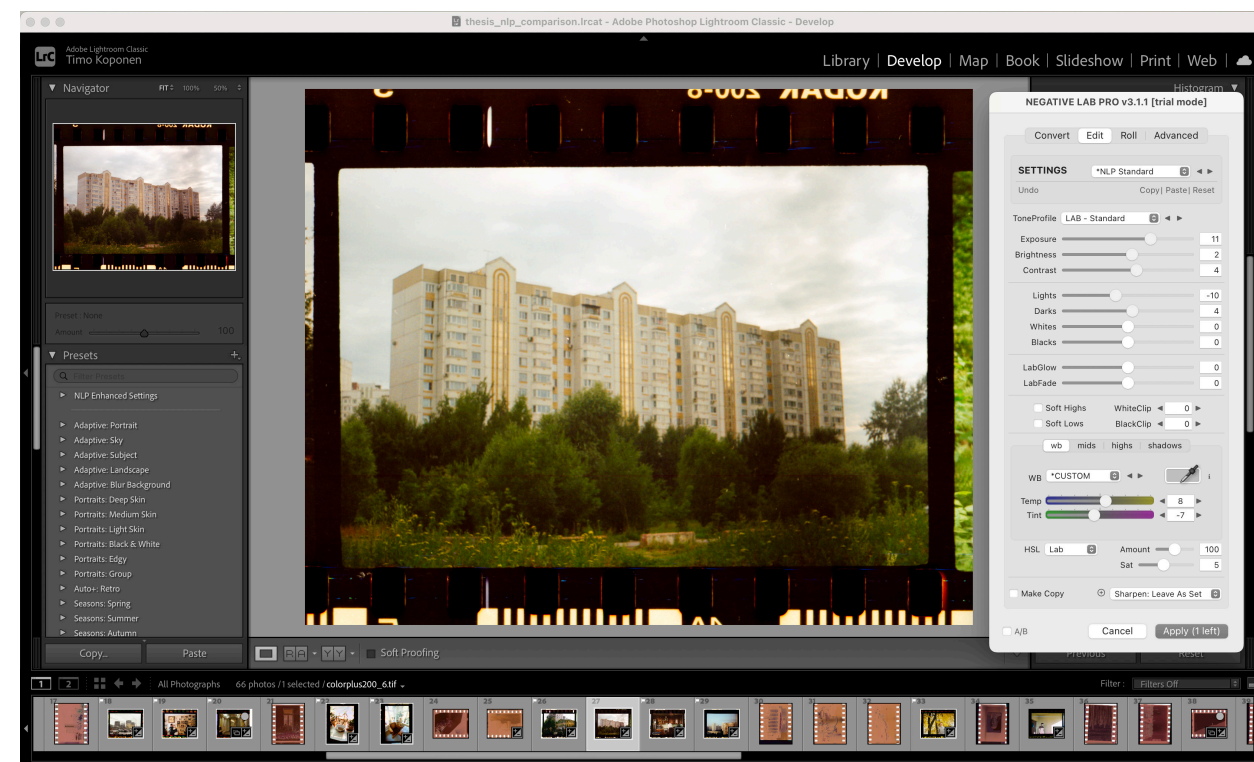
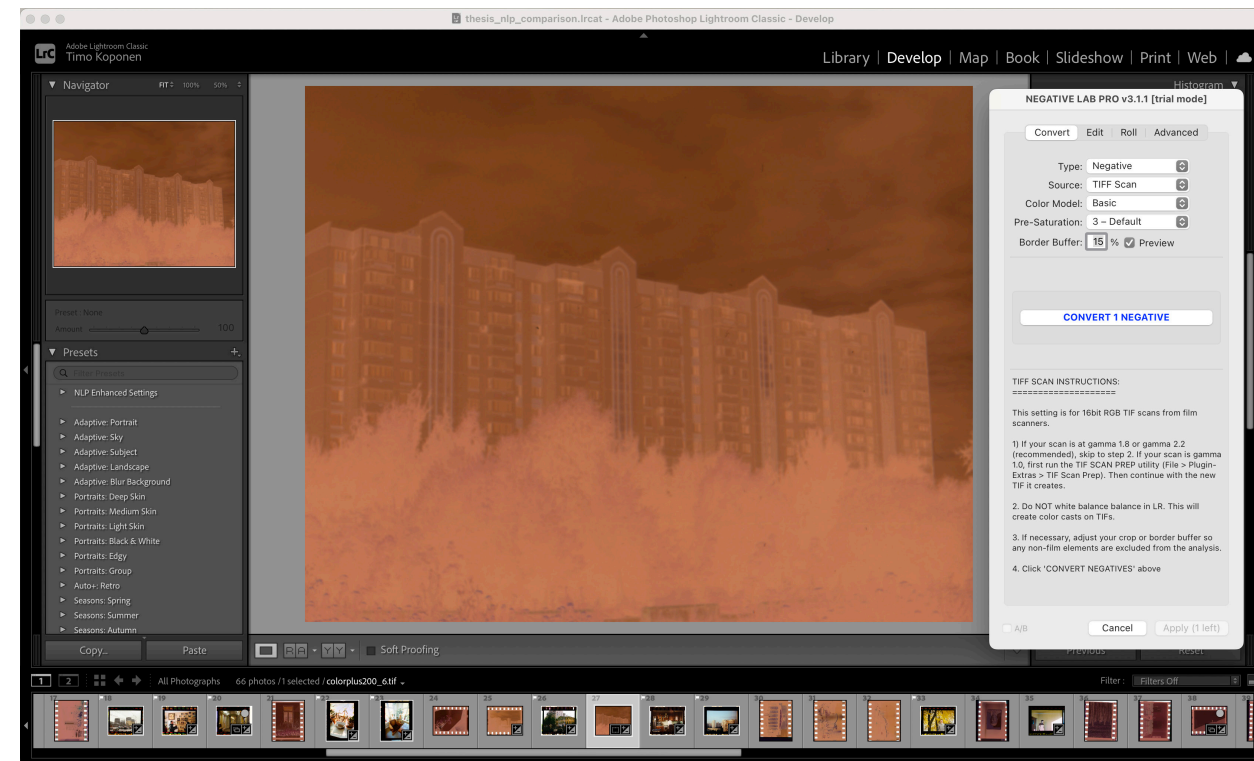
inversion, balancing the black and white points. It works on top of Lightroom so I don't know how much it can do internally and probably has to do everything using the Lightroom's exposed slider functionality. Since it is not publicly revealed how it works or what it does we cannot know for sure.

Interesting to note, if I scanned a negative without color management, meaning I didn't include a color profile from my scanner and convert from that to some other profile, the results using Negative Lab Pro weren't that good. But everything started working when the source image had a color profile tagged with it. Of course this makes sense but I was surprised by the amount of difference it makes.

Negative Lab Pro is a bit more complex to use than the other software. I had to read a manual to use it or watch a video. A plus is that it does explain the things before pressing the buttons in the actual software. It just seems hard that if you have a TIFF, you need to manually apply a different preset and not balance to the film border, whereas on a raw photo you need to balance to the film border.

My initial impressions of Negative Lab Pro are that it yields good and pleasing results that I like. I like that it emulates different scanner styles and you can choose different styles or presets. The default conversions seemed to be quite good already for most images. I can see why people like it and why it's popular. Most photographers use Lightroom already and are fine with it. Due to this, it's easy to just buy a plugin that integrates into an application that they already know. Then after doing the conversion with Negative Lab Pro, you still have all the features of Lightroom at hand as well.

[Image 17] Screenshots showing the user interface of Negative Lab Pro on top of the UI of Adobe Lightroom



3.2 FilmLab

FilmLab (Develop and Fix, inc., 2025) is a stand-alone application. FilmLab provides either a monthly or yearly subscription model or a buy-it-for-life option. It works for camera RAW files and TIFF files. (Develop and Fix, inc., 2025) Like Negative Lab Pro, it’s also not open source so we don’t know for sure what happens under the hood. I have a feeling that a lot of these also intentionally keep the exact logic secret.

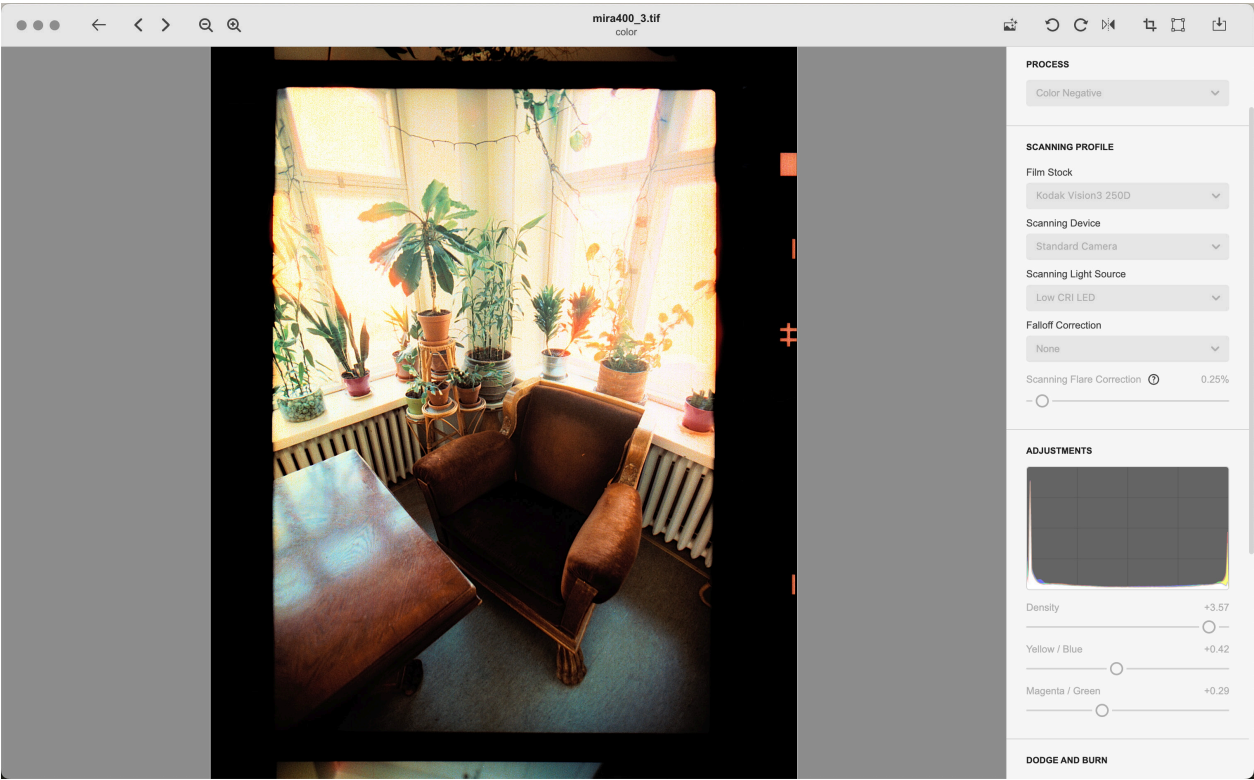
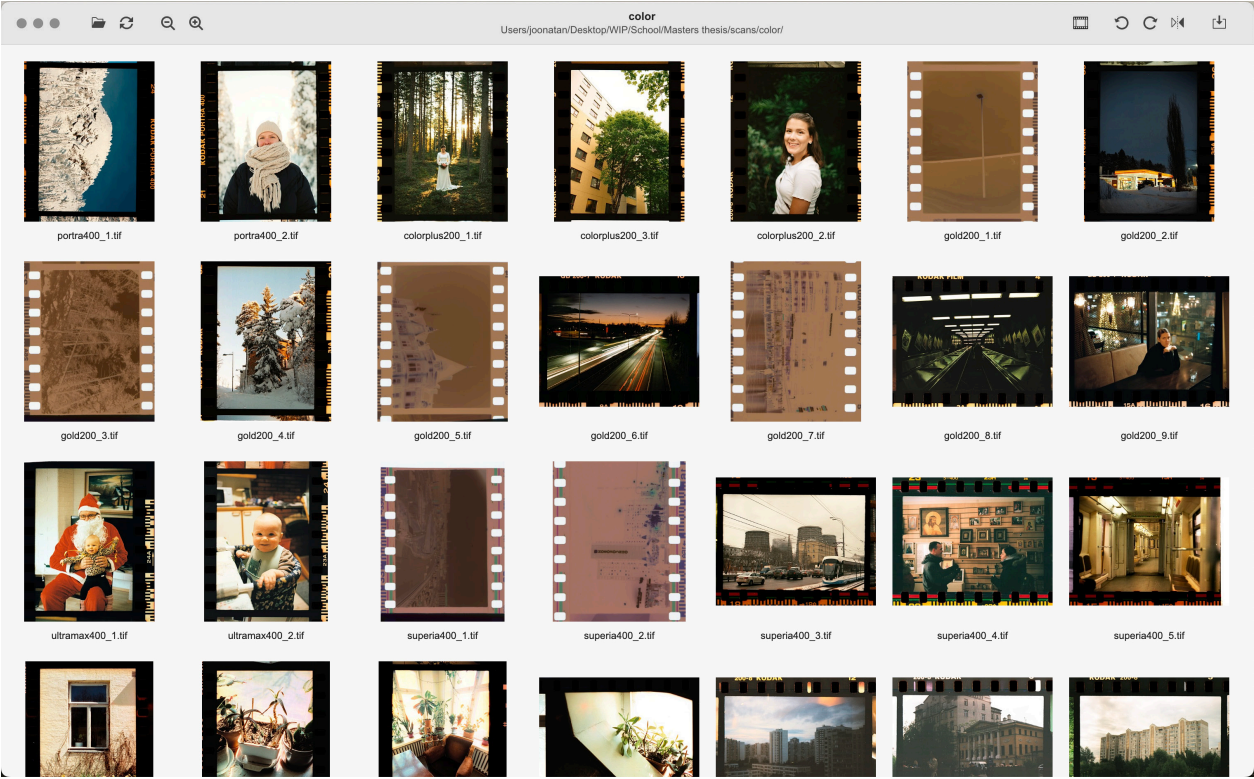
FilmLab feels much more simple and straightforward to use than Negative Lab Pro. You import a folder of images and then you can convert or apply some edits to them all as a batch or choose individual negatives and edit them individually in more detail. You choose to process the negative as color negative and it immediately does the conversion for you and sets an automatic color balance. Often the conversion is quite good. You then have simple options to edit the general density or highlight and shadow densities, contrast and white balance. It does have good cropping and sharpening functionalities.

FilmLab did a really nice job for some images. It didn’t perform so well with others, especially underexposed images, but for the majority of images, it does give nice results. For me FilmLab stands out from the rest of the apps with the colors it produces. It seems to apply a style or look, like a look up table, since the colors have something different about them. I respect that it adds a look. I might have liked to have the option of no look applied for some images. but maybe that’s the charm of the application that it owns its look.

The usage and UI in general feels much more polished but sometimes shows that it is still a bit of a work in progress. Even though I complained that Negative Lab Pro felt cluttered with all the different options then, with FilmLab, I feel like I am missing some options that I would like to have, for example the black point and white point adjustments. So it goes to the extreme of having fewer options than I would like versus Negative Lab Pro having too many. A nice feature that FilmLab has, is a nice full-size preview with zooming that feels smooth. Its auto crop and rotate functionalities worked surprisingly well as well. I like the batch editing functionality and the option to copy settings from one image to the other quickly. With regards to the edits, the density slider doesn’t behave as I would expect, almost feeling like a gamma adjustment. The shadow and highlight sliders feel like they don’t really do anything meaningful. The auto density calculations often yielded too dark images that I had to manually adjust.

An interesting feature of FilmLab is that it offers a selection of different film stocks that you should match to the film stock of the negative. There is no explanation how it affects the conversion and what to do if the film stock used isn’t listed. It would be nice to know what the film stock option does. Does it actually take into account the characteristic curves of the film or some data derived from them? If so, then it’s clearly a different approach than all the other applications listed except for the Fuji Frontier’s internal software.

[Image 18] Screensots showing the user interface of FilmLab. Top: the batch editing view. Bottom: Individual negative editing window.



3.3 SmartConvert by Filmomat

Like FilmLab, SmartConvert (Filmomat, n.d.) is a standalone application. If you want to purchase it, you pay once. The application is aimed towards camera raw files, but works with TIFF files as well. (Filmomat, n.d.) Like the earlier programs, it is not open source, so we cannot inspect the algorithms it uses.

When using SmartConvert, you simply drag and drop images to the app, which are then shown on the left. In the middle is a preview of the current image and on the right are the edits. The edits are done with buttons for orientation, cropping, white balance and contrast, density and saturation. The conversion is calculated automatically and you have the option of using automatic white balance.

The best thing for me about SmartConvert is its user interface (image 19) and the approach it takes to editing. The UI is simple and clear and is inspired

[Image 19] A screenshot of SmartConvert's user interface



by the Fuji Frontier's approach to negative editing (Filmomat, n.d.). I really like the layout of the UI and how it emulates the Frontier. Editing with a mouse is a bit clumsy, but SmartConvert relies a lot on keyboard shortcuts like the Frontier. With the keyboard shortcuts the editing becomes quite fast and intuitive.

The results it produces are decent. They are a bit too aggressive with contrast and saturation but luckily those can be changed with presets. You can also save your own presets. The app seems to struggle a bit with underexposed images and causes a blue color cast in the shadows. The default preset was a bit too saturated for me, and too contrasty, but it hides the color cast in the shadows. If you use the flat preset, then you can see it. SmartConvert markets itself as not applying any looks or lookup tables (Filmomat, n.d.). I would suspect that it does some form of the basic inversion for the negative.

The lack of shadow, highlight or tonality adjustments in general is a minus for me. They would have helped with the underexposed images for example. But contrast, brightness, and saturation can luckily be fixed. Overall, the experience of using the app is probably the best, but the results are not.

3.4 Negfix8

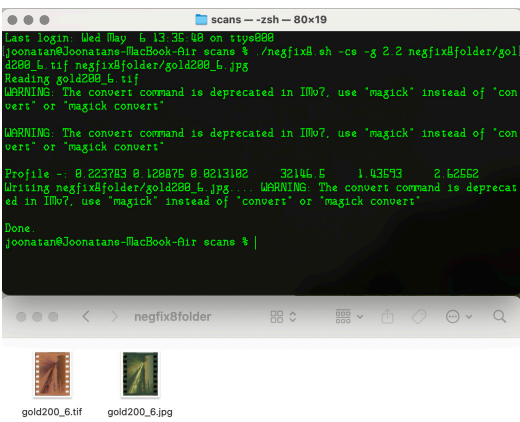
Negfix8 (JaZ99, 2013) isn't really an application. It's a bash script that automates the program ImageMagick for film negative conversion.

The script is free and open source and so is ImageMagick as well. It's nice that you can inspect the script and see exactly what it does and get inspiration from it. But bash can be quite hard to understand. Since it's a bash script, it works from the command line as seen in image 20 and can be a little complicated to use for someone who isn't used to command line applications.

You use the script by giving it an image file path as an argument and then you can optionally adjust the name of the output image or tell the program to do contrast stretching or to apply a different gamma to the final image.

My success with Negfix8 was quite hit and miss. For some negatives, it did a great job, but often it performed poorly for others. I tried to follow the instructions as best as I could, but it seems like things like dust or debris that cause dark spots that are darker than the densest spots of the negative distract the algorithm since most of the issues seemed to be in calculating the white point, which caused a considerable color shift in the highlights. After manually removing the debris from the scan in Photoshop, then Negfix8 converted the image perfectly. Example of this is shown in image 21. This indicates that Negfix8 just samples the brightest individual pixels without averaging out any irregularities. The instructions say to use a scanned image with a gamma of 1.0. My scans have baked in-gamma of 2.2 so that affects the end result as well, but only in the tonality.

Since Negfix8 is a basic command line script, there is no UI and no interactive editing. If it does a good job with conversion, you would still probably want to use an image editor for adjusting the contrast or brightness. A good use of this would be batch processing multiple negatives. It's a shame that small debris or dust skews the results so much and the batch processing isn't very quick.



[Image 20] Screenshot of using Negfix8 from the command line

[Image 21] Example of Negfix8 getting distracted from bright spots caused by debris or dust. Left is uncorrected negative, right is negative with those bright spots removed



3.5 Darktable Negadoctor

Darktable (2026) is an open source general raw file editor a little bit like Adobe Lightroom. Negadoctor is just a module inside Darktable designed for converting film negatives. It's primarily designed for camera RAW images, but works for TIFFs as well. (Darktable developers, 2026) As Darktable is open source, it's free and we can potentially inspect exactly what it does for the conversion.

Usage of the module itself is quite straightforward, but it does contain some parts that can seem hard at first. The module is visualized in image 23. Luckily, there aren't a lot of options in general in the negadoctor module. The module works mostly manually. You set the film border by sampling the film base color. Then you set the dynamic range of the film, meaning basically the white point with a slider. It tries to automatically guess the film base, but most often you will have to manually set it. After this, you manually sample a shadows area and highlights area to correct for shadow color cast and highlight white balance. This works surprisingly well. You can also tweak the sampled colors with sliders. After these steps you already have a color-corrected image. Then you can tweak the final density and contrast. I think that this workflow is quite nice actually, but if you want to make other adjustments, for example adjusting highlights or shadows for an underexposed image, you would have to use other modules inside Darktable which brings us to the topic of darktable in general.

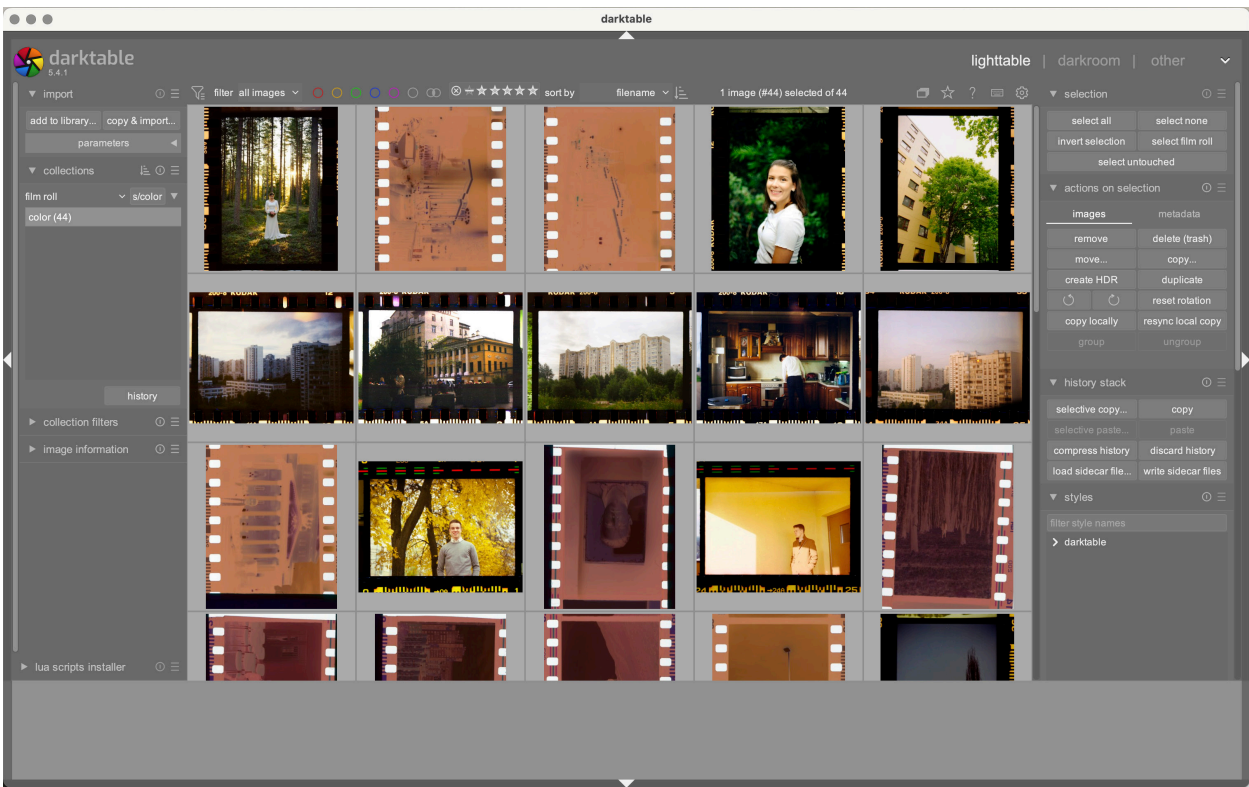
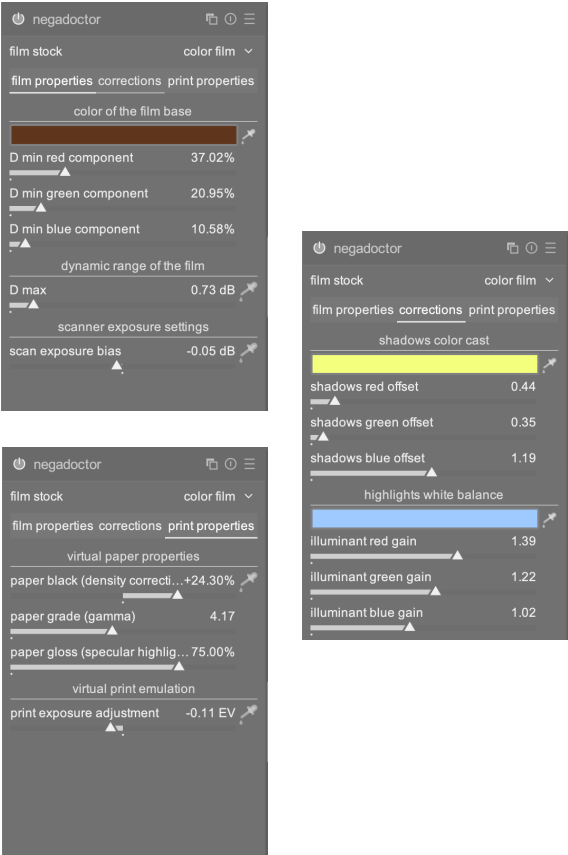
The Darktable application itself then is a completely different story. An overview of the UI can be seen in image 22. If you want to use other features than Negadoctor, it can feel really complicated at first since it's a whole multi option raw editing software that gives the user a lot of control. Without studying instruction manuals or tutorials, Darktable seems really unintuitive to use and to learn and navigate through, so for a first user it doesn't feel very easy to use or intuitive. This can feel frustrating. The problem again, is getting lost in too many options in the rest of the program.

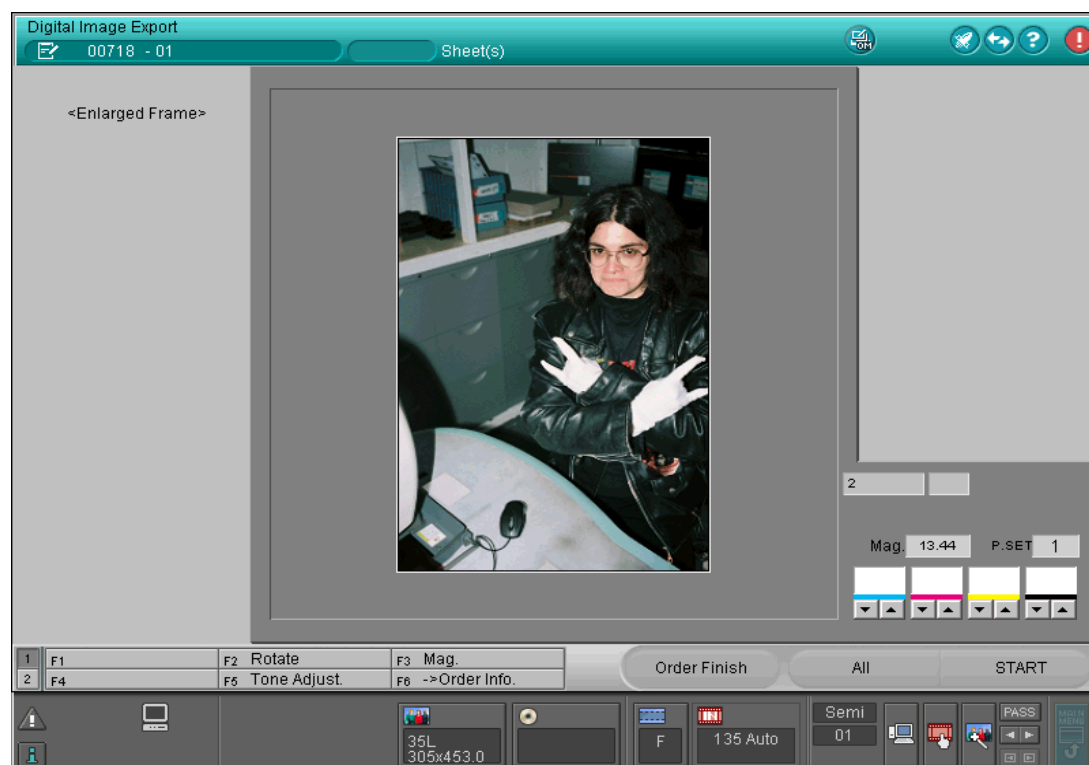
The Negadoctor module does give nice final results that are comparable to manual inversion in Photoshop or to Negative Lab Pro. Its color correction is really powerful. And once you understand the negadoctor module, the conversion process itself is quite easy, but it's not really automatic and thus takes a long time.

I really like the results. I also like the approach of using the eyedroppers, although for batch processing it is tedious and takes almost the same amount of time as the manual method in Photoshop. The Darktable application itself is a complex program that would need some time to learn, but then I am sure it's really powerful.

[Image 22, right] Screenshots from the user interface of Darktable. Top: Darktable's general view of all images. Bottom: Individual image selected with the Negadoctor module active

[Image 23] Screenshots of the Negadoctor module's different tabs showing the different conversion steps





3.6 Fuji Frontier

Fuji Frontier scanners differ from the other entries in this chapter. There are a few different models, but the basic technology they have is the same. A Fuji Frontier is an actual professional film scanner designed for lab use (image 25). They are not made anymore, but still actively used by many film labs. Since it's a physical scanner it's not really comparable to the other software entries in the list, but I include it here due to its strong influence in the film world. It's a favorite among many analog photographers today (McDougall, 2024; linusandhiscamera, 2023; Verbeeck, 2025) and often used as inspiration or a source of emulation for conversion software and their processing styles (Filmomat, n.d.; Nate Photographic, 2024). There are other film scanners such as Noritsu, Hasselblad or Pakon scanners, but I will only focus on the Frontier. The Frontier has its own internal image processing and editing software as well, seen in image 24.

The Frontier cleans dust automatically from the negatives. (Verbeeck, 2025) It uses a special approach to scanning. It projects red, green, and blue backlights separately through the negative and then combines the scan results of those into one image. When you edit the color balance in the software it actually adjusts the intensity of those backlights (Linusandhiscamera, 2023; McDougall, 2024)

In the software, you set both the scanning settings such as resolution and also edit color and tone adjustments. It focuses heavily on automatization and speed. It automatically corrects the scanned images' color and tones and then lets you manually correct the three RGB channels or the overall density of the scan. In addition, you can select what tone profile to use. These define the contrast and if shadows or highlights need to be lifted or lowered, etc. You can also set the sharpening amount. Aspects that increase the speed is a little dedicated keyboard with physical buttons dedicated for the adjustments. Another speed boost comes from the option of holding settings from one image to another. (Linusandhiscamera, 2023; Verbeeck, 2025; McDougall, 2024; Safelight Berlin, 2024) Both of these speed-boosting aspects are features that I like and I think I would really enjoy using the Fuji frontier scanners and software.

Since this is a commercial product completely made by Fujifilm, there aren't a lot of details on how the algorithms inside the software work. It would be really interesting to know how it approaches the conversion and what steps and automation it uses.

Many people praise the frontier for its color rendition (Linusandhiscamera, 2023; Verbeeck, 2025; McDougall, 2024;). It delivers a finalized looking image with an interpretation of the colors. Some compare the look of photographs from a Frontier to photos printed on photographic paper (Vandebroek, 2026). For me, I like the general look of images made with a Frontier. I don't have personal experience from it and I haven't had my images scanned with it so I cannot personally say if I prefer the colors as is or if i would like to do some adjustments.

[Image 24, left]: Screenshots of the user interface of the software of Fuji Frontier. Top: A general view of images with the tone adjustment window open on top. Bottom: Edit window of an individual image.

[Image 25, below]: The Fuji Frontier sp3000 scanner, a big dedicated and old film scanner



Application	Negative Lab Pro	FilmLab	Smart-Convert	Darktable	Negfix8	Fuji frontier
User Interface and user experience	Confusing	Okay	Nice	Negadoctor module okay, otherwise confusing	Lacking, no user interface	Nice
Editing options	Lots of options, maybe too many	Too few, lacking white and black point editing	Too few	A bit too few, lacking proper tonality control, which could be hidden somewhere	None, only auto contrast stretching	Almost enough, would like more manual options for tone adjustments
Conversion approach	Probably auto White and Black point balancing + auto white balance	Probably more complex, considers characteristic curves + applies a lookup table	Probably auto White and Black point balancing + auto white balance	Samples film border, shadow color cast and highlight white balance	Takes logarithmic density into account, probably auto White and Black point balancing + auto white balance	Probably more complex, considers characteristic curves, + applies Lookup table
Preliminary conversion results	Very nice	Okay, heavy look applied	Okay, but a bit lacking	Very nice	Okay or bad	Very nice
Closed or open source, paid or free	Closed, paid, Lightroom plugin	Closed, paid, independent	Closed, paid, independent	Open source, free, independent	Open source, free, script for imagemagick	Closed, paid, Scanner software
The overall happiness it gives to me during use	Okay	Okay	Quite happy	Quite happy, but slow	Not very happy	Probably really happy

3.7 Summary of the Case Studies

Having gone through each application separately, a summary of my findings from the studied software is presented in table 1. I include the Fuji Frontier in the comparison, but acknowledge that I have not personally used it and thus the conclusions are not certain.

I will next provide an overview of the most important findings in relation to the relevant research questions.

How to correctly convert a film negative, what does that mean or is there a correct way?

The reviewed applications are all tools that people actively use, and some of them are paid products but many of them produce results that differ from each other and use different approaches of conversion. We can thus conclude that there are many different ways to convert a film negative. It is interesting to note that some are more pleasing to me than others.

How to develop a personal tool for my photography practice?

Key takeaways in regard to toolmaking are that I enjoy the approach to an editing UI provided by the Fuji Frontier or UIs inspired from it like SmartConvert. However, I also want to have manual control of the highlights, shadows, whites and blacks and not just pre-assigned presets for tonality control, which are still a nice thing to have. There also has to be some simplicity to the UI since too much clutter makes it confusing. I also seem to value an application that does not apply too strong a look and if it does, it has the option to disable it.

[Table 1] Comparison between the different software reviewed in the case studies



4.

Production
of VeryNeg

*[Image 26] A converted scanned negative without gamma
correction or color or tonal adjustments*

4.1 Production Description

The production part of my thesis focuses on applying the learned theory from the literature review and findings from the case studies into an actual bespoke software application, named VeryNeg, that is designed specifically for converting scanned film negatives into positive and final images in the way that I like. I approach the development process as a method of iterative prototyping and experimentation to find out more about the technicalities of image processing, practicalities of tool building, and pondering different ways to approach the conversion. I also document and communicate the choices of technologies and architecture used and then reflect on my findings from the development process.

VeryNeg version 0.5.0's source code is attached to this thesis. Future source code and releases can be accessed from VeryNeg's GitHub repository: <https://github.com/tootainz/VeryNeg>

The general steps involved in the production can be summarized as follows: First I had to make a plan and choose technologies to use. After this I had to come up with an editing workflow for the required steps to convert an image from negative to positive properly and accurately and then to program these into algorithms. Then I had to develop a software application that handles the negative image opening and saving, and applying the image editing algorithms to the negative images. After this an UI and interactive editor were implemented to make it feel like a complete application. Throughout the process, I was iteratively testing the functionality and fixing bugs and implementing missing features, refining the application.

This section is divided into subsections that describe the goals of the application, the process of choosing technologies to use, how the development was carried out and culminating in the introduction and overview of VeryNeg. The actual implementation of VeryNeg's features and findings from the making are discussed separately in the next section 4.2.

4.1.1 Goals for VeryNeg

I will next list the goals I set for the development of VeryNeg:

General goals:

- Can be used as part of a workflow
- Standalone application with a user interface
- Decent performance and usability

Application specific:

- Support flatbed scanner generated 16-bit TIFF images
- Automated conversion process
- Convert multiple negatives easily
- Ability to tweak the conversion process if needed
- Edit density and tones after the conversion
- Edit the color balance manually and automatically
- Edit sharpening
- Export good quality images
- Yield pleasing results for my practice and preference

4.1.2 Choosing Technologies

Before doing any programming, I had to choose which technologies to initially use for the application. Some of these initial decisions changed during the development process, with the main one being the way the UI was implemented.

I first started to search for plausible libraries that would work well with my project. The first one I looked at was ImageMagick, which is an old command line image editor that could also be used programmatically from different languages. My first idea was to use it with JavaScript and WebAssembly, but the WebAssembly versions of ImageMagick don't seem to be well supported. It seemed that using ImageMagick could become a performance issue, and also it would not be so practical to design another program around it.

Since I have been focusing mainly on web development in my studies, I was looking first into different ways of using web technologies to implement the application. This would mean programming in JavaScript, either for the browser directly or a Node.js or Deno JavaScript runtime application or a separate web app framework such as Electron. However, I wasn't sure if I could get the libraries I wanted using those technologies. I also looked into other JavaScript image editing libraries, but I wasn't sure if they would be powerful enough for large images or camera raw images if I decided to use those. In fact using JavaScript for the image editing logic might be too slow since image files are so huge. There is the option of using the GPU through WebGPU, but I didn't want to dedicate time to learning it.

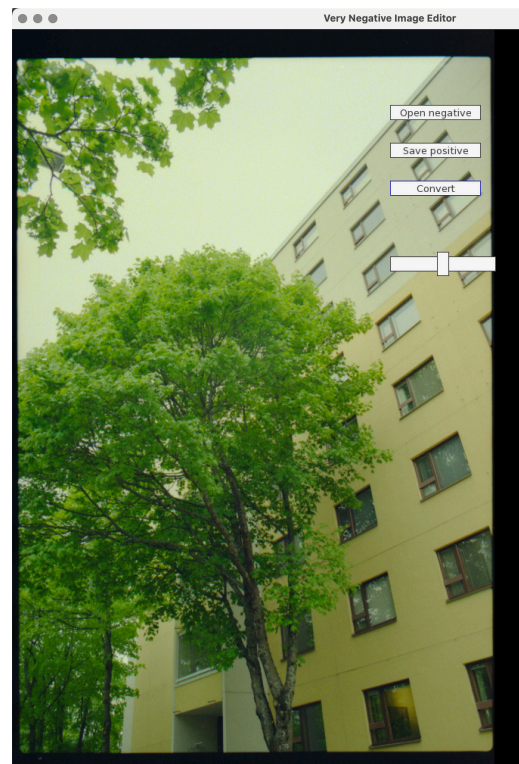
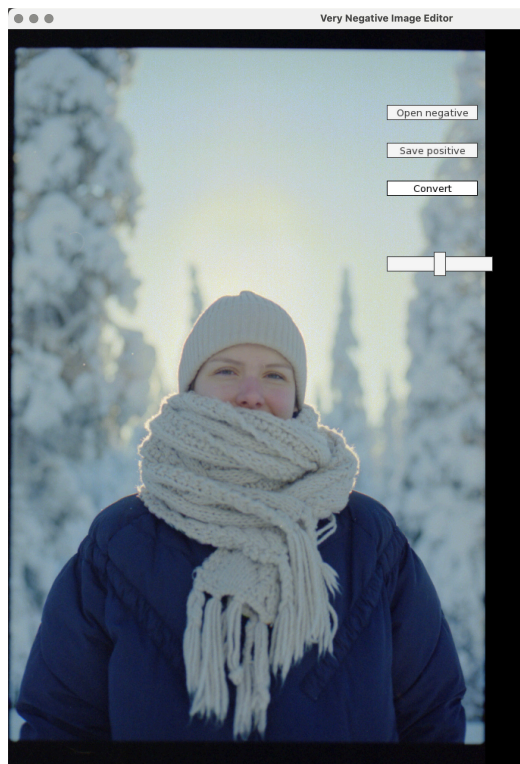
The final option that I considered and ended up choosing was to use a lower level language to gain performance required for image editing. I had touched on C++ before and used it to program a microcontroller so I had some experience using it. At first I was a little intimidated by the idea of using C++ since it is such a vast language and it seemed a bit cumbersome to write and use and one really needs to be aware of what the program does and how it uses memory. But I wasn't scared. And since C++ compiles directly into machine code and gives manual control over the memory, it can be really fast and performant. C++ or C are the languages most of the image editing programs are written in and especially if I wanted to make the application responsive in real-time, speed would be important. C++ also has a lot of good libraries for image editing such as OpenImageIO for handling image opening and saving in a vast number of different image formats. I also considered using

ImageMagick as a C++ library or libvips or OpenCV for the image editing algorithms part.

With this initial research I came up with the plan of making the application logic in C++ but maybe utilizing web technologies for the user interface. This would mean a C++ backend with an HTML, CSS, and JavaScript frontend or user interface. I tested using the operating system's native WebView with a library called Saucer, which is basically a mini browser inside each operating system. It worked, but the problem was that sending the preview image from the C++ code to the WebView was slow and a bottleneck for the real-time editor. I decided to abandon that plan as well. There were other options that I found that directly rendered HTML using the GPU but those weren't fully open source or had some limitations.

After intense research and hard decisions, I decided to program the whole application using C++. Since I didn't want to start writing code for the GPU I searched for a 2D graphics library that would hide the process of drawing images on the screen. I ended up with Simple Fast Media Library (SFML). I also chose OpenImageIO for helping me to open and save different file formats and to convert between formats or different bit depths or image sizes. I eventually decided not to use a premade library for the core image editing algorithms since those are what I wanted to focus on myself as part of the research.

I found a GUI framework called TGUI for implementing all the buttons and menus. It was originally designed for SFML and had good documentation and seemed quite easy to use. This worked fine but after implementing it, I finally found the GUI library that I was looking for from the beginning. There actually existed a library for rendering a GUI with C++ and using HTML- and CSS-derived languages for defining the GUI. This framework was called RmlUI. The HTML- and CSS-inspired languages it uses, RML and RCSS, don't have all the features of regular web technologies, but it was enough for what I wanted to do and made the development of the UI much easier for me. However, the setup time for it was much more difficult and took a long time.



[Image 27] Screenshots of the demo showing conversion results and the crude user interface

4.1.3 Development Process

The development process for the application can be split into two main parts: the demo and release development. During a computer science course, before actually developing the application properly, we were given one week to develop a rough demo of the application that we could show as an example when discussing the idea and then to make changes to the plan if needed. This approach worked well for me and it was a good way to kickstart the development process. Thus, I decided to take the same approach with this project. I first developed a really rough demo in one week and then dedicated an entire month for the release development for the full application. The goal was not to make it a final version, but a polished beta version that could already be used as a part of my workflow.

Demo Development:

The demo was done in a week in February and already represented the application quite well. The demo, as shown in image 27, allowed you to import one scanned negative image and to convert it with the press of a button and to edit its exposure. You could change the imported scan. It also allowed the user to

export the photo. On the code level, the basic building blocks and classes were in place. It had the Model, View, Controller and Negative classes. The negative already generated a lower-resolution preview image and performed the conversion and edits non destructively. The algorithms used in the conversion and edits were in their own files under “image algorithms”. The app was using SFML and OpenImageIO. The crude UI of a few buttons and one slider was done using TGUI, which was changed in the production development.

The conversion itself was what I labeled as a crude inversion. This basically meant to find the brightest and darkest pixels for each RGB channel and then to make the darkest to be 0 and the brightest to be 1. After this it's just invert 1 to be 0 and 0 to be 1. And to be honest this already looks almost the same as the final basic inversion as well later see. I tried some basic color balance with applying a gamma function to each color channel with a hardcoded value, but the results of that weren't quite there yet. The real-time editing was slow since all the edits were applied to the full sized image and the algorithms were all single threaded. The exposure adjustment was just multiplication so the images highlights blew up almost instantly.

Release dDevelopment:

After the demo it was time to start the real development. Based on what I did in the demo and what I noticed I needed or wanted to do, I wrote a plan to develop the full application. These were immediately some of things that would need to be implemented:

- Speed up the preview process by introducing a smaller working image instead of always using the full sized image and to parallelize the algorithms
- Manage the state of the image edits and also how are the edits applied to the image
- Allow to have multiple negatives
- Proper conversion based on film's physical properties
- Auto white balance function
- Implement an actual UI
- Implement more edits like color balance, shadows, blacks, contrast, etc

I had the whole month of March reserved for the development process. I listed out the planned features and divided them into different days and kept adjusting and improving the timetable as time went on. For the most part, the production went well, but due to sometimes debugging a small error that took a few hours or implementing a feature took a few days longer than expected, the month wasn't quite enough and more had to be done in April. As time went on, I realized that I still had some more features that I had to add to my plan and those were:

- Color management for color correct workflow
- Scan area for the conversion
- Copying the edits from one negative to another
- Presets for the edits
- Sharpening
- UI refinement

4.1.4 VeryNeg Overview

I decided to name the final application VeryNeg. The application is simple with a large preview of the current image on the left, the settings for the conversion and post-convert edits on the right, and a film roll of the opened negatives on the bottom and the ability to import, export, remove, and navigate between them. Screenshots from VeryNeg’s user interface can be seen in image 28. A more detailed walkthrough of the usage and features is shown in the walkthrough video attached to this thesis.

Once a user has opened an image or multiple images at once by pressing the import button, they can set options for the conversion. An important information for the conversion is to know what gamma the scanned image contains and this can be set using the scan gamma dropdown menu. The conversion itself is fully automatic, but if one wants, they can activate custom conversion settings. They can set a custom sample of the film border or the densest area or a custom sample area if for example the image contains the film perforations or scan holders, that might disrupt the conversion process. The user can also set a crop area to be cropped to upon export and to edit the orientation of the negative.

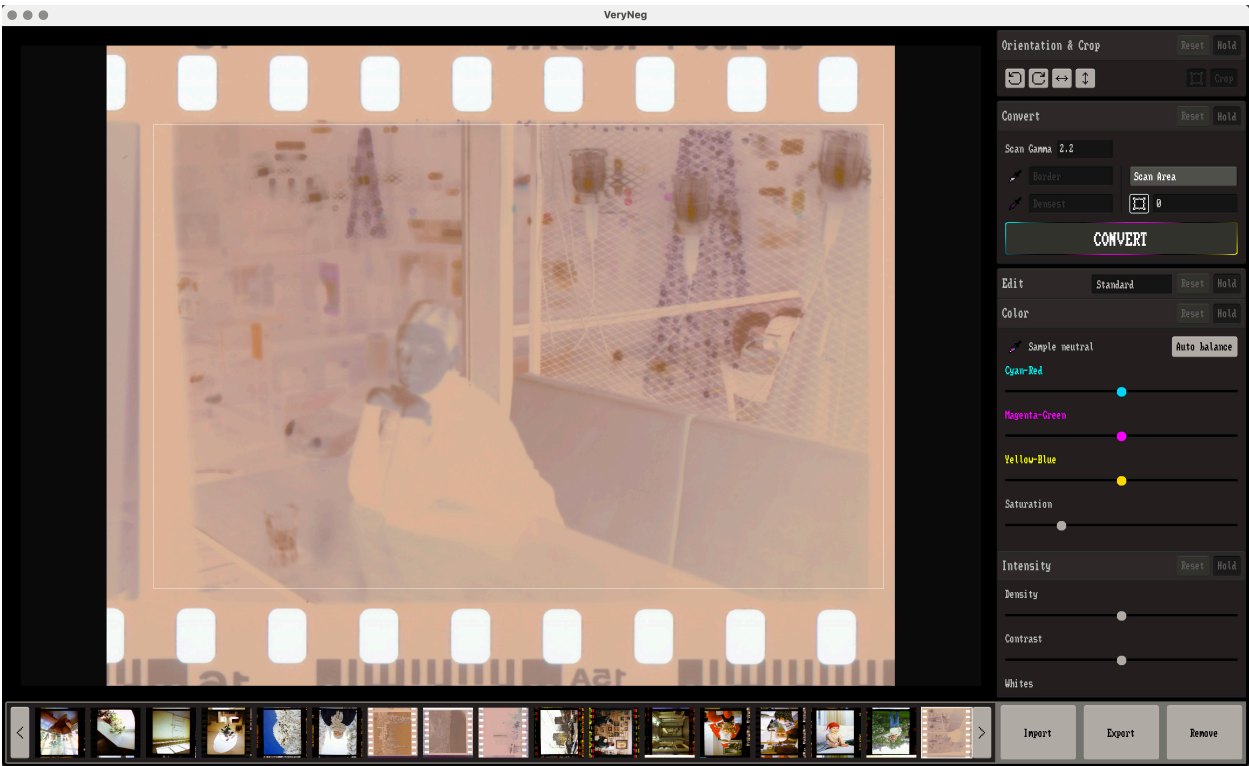
After the automatic conversion, the user can choose a starting point from predefined presets. Using the sliders, the user can edit the color balance of the image manually, with an eyedropper on a neutral area or to let it be automatically calculated. They can also edit the density and other intensity values to their liking. In addition to this the user can set the amount of sharpening if desired. The sharpening preview above the sharpening sliders shows a magnified sample of the image so as to see the sharpening applied on the pixel level on a full resolution image.

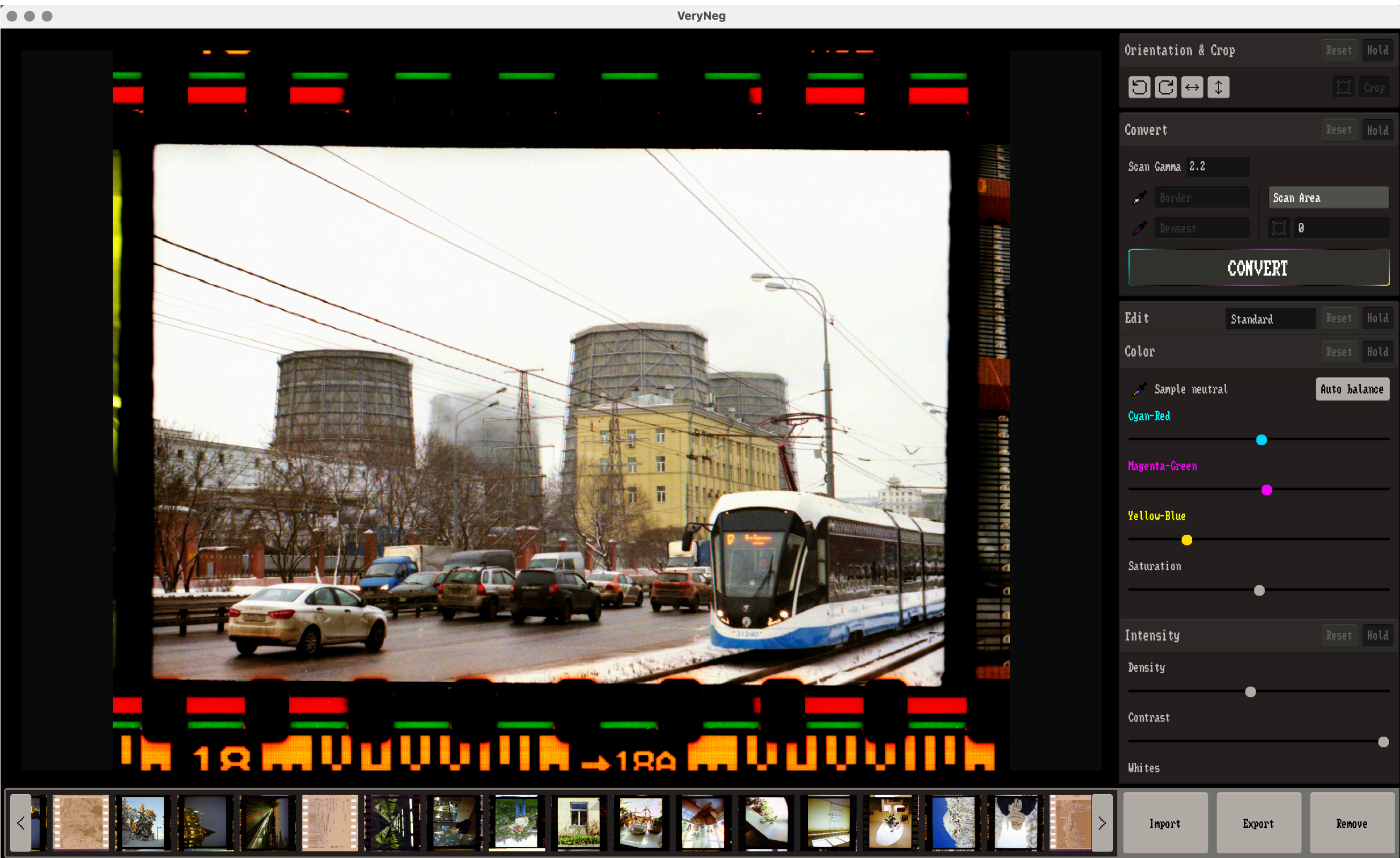
Performed actions can be canceled by undoing or they can be redone by pressing cmd + z or shift + cmd + z.

To make the editing process faster for many images, the user can select to hold settings. This means that the settings are effectively copied and can be pasted to other images. The conversion settings can be held, meaning whether to perform a fully automatic conversion or to use a custom area or a specific sample. If you want to use the same absolute conversion values for many images, the user should use the custom samples. You can also hold all of the edits at once or choose to only hold a subcategory of color, intensity or sharpening. Settings can be held with the hold buttons or by pressing command + c to hold all settings. Held settings can be applied to other images simply by pressing cmd + v

Once the user is satisfied with the results, they can press the export button to open the export menu. Images can either be exported as a 16bit TIFF or an 8bit jpeg file. The user can also select which color profile they want the exported image to be converted to. When the user is editing the negative, the application automatically saves the edits next to the original file in a .neg file that will automatically be loaded with the negative when opening if the user wants to change their edits later.

[Image 28] Screenshots of the user interface of VeryNeg showing the preview on the left, sliders and settings on the right and film roll at the bottom. Top: A negative being prepared for scanning by selecting a scan area. Bottom: A converted negative with the sharpening preview and settings on the right, Next page: A converted image showing the editing sliders on the right





4.2 Reflection and Findings from the Production

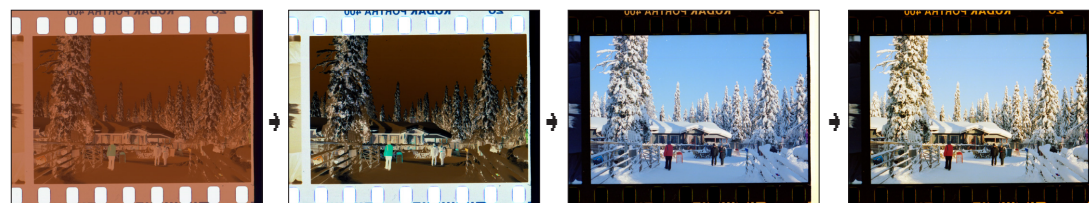
This section is divided into two main subsections. The first discusses the actual conversion and image editing functions and algorithms implemented into veryNeg and the color management workflow. The main focus here is on the technical image processing but I also discuss the artistic interpretation of images when discussing the different conversion methods and the reasons I chose specific technical functions for the edits. The second section discusses the toolmaking process. Here again I do focus on the technicalities but it is balanced by reflections through the lens of toolmaking and agency. The findings are all results of the iterative process and experimentation of the development and design process.

4.2.1 Image Processing and Algorithms

Conversion

I would argue that the core and the most important part of the application is the algorithm for the conversion. Figuring it out was complex, fun, irritating and also surprising and rewarding all at the same time. I tried different techniques for the conversion in Photoshop, within its limits. I also played around with my application trying different ways to convert the image and would divide my findings into three different algorithms: simple conversion, advanced conversion and my conversion.

[Image 29] Simple conversion method's steps visualized.



Original negative

Sample white and black

Simple invert

Apply edits

Simple Conversion Method

The simple conversion is something that can be done in Photoshop and that I described in the literature review. My version of this “simple” way of converting a negative as visualized in image 29 is basically this: With the levels tool or the curves tool, for each RGB channel, you balance the black and white points by manually dragging them just to the edge where they clip. This means that for black every channel is black and for white, every channel is white. This removes the orange color cast and already produces an okay image. Then you convert it by a simple inversion: dipping the max value of 255 or 1 to be 0 and 0 to be 1 or 255. After this you choose two channels and in the curves editor, you change the curves slightly, so that you have basically balanced the color balance of the image. In the beginning I thought that I had to do all of this by hand manually for each channel, but then I found out that Photoshop’s curves tool has an eyedropper for blacks, whites and midtones. You sample the correct region in the image and the eyedropper balances all the channels automatically. So in the end, the editing process for this method in Photoshop wasn’t that complex or time consuming after all. However, some sources (Buchler, 2023) claim that this is not the “proper” method for converting the negative, since the conversion calculations should take into account the fact that a negative stores its data in densities, which are logarithmic and this method does not take that into account.

Advanced Conversion Method

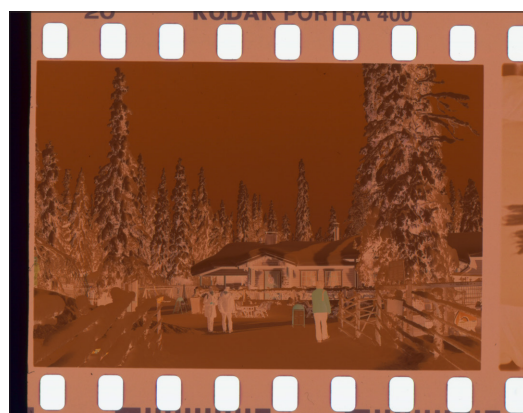
The more advanced approach to convert a film negative is already described in the background research. It includes converting the measured values that represent transmission into density, which is

logarithmic, then you would follow the film’s characteristic curves for each of the channels both ways to balance the black and white points and to eventually get the actual intensities measured by the film. I tried to implement this method as described by Tuijn (1996) and it worked and produced nice results. I traced the characteristic curves of Kodak Gold from the official graphs from Kodak into a dataset and then created a curve function based on them with a curve fitting tool from a dataset. For this to work consistently, you would need the original characteristic curves of the current filmstock or somehow simulate them or use generic ones. What I was thoroughly surprised by, though, was that the results weren’t actually that different from the easy method. The colors were basically exactly the same, the only difference was that the gamma correction was off and of course the final values did not represent the actual captured physical values, but that was about it. And the gamma is easy to correct in post. I couldn’t see a substantial difference that would make me want to implement this fully since it was way harder and takes a big performance toll for the conversion. The reason that this is the proper way is that as far as I understand, you can get the original intensity values of the light with this method as opposed to the simple method, since the math is physically correct. Maybe there are some small intensity differences in the highlights and blacks but this didn’t really seem that visible to me. But then again, I don’t really need those to be mathematically correct, since I will just be scaling them to the range of 0-1 anyways for digital images.

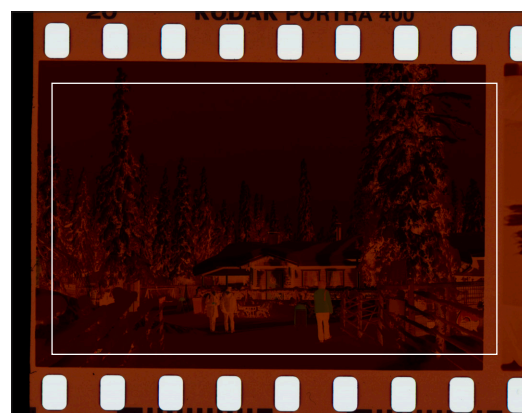
My Conversion Method

For the final approach I decided to use a combination of the two methods described above. I still take into consideration the conversion from transmission, which means the pixel values to the logarithmic density. and then back to the non logarithmic original intensity. I just keep the Characteristic curves as a linear line for the whole time as suggested by Buchler (2023) and as Negfix8 (JaZ99, 2013) does. The reason why I still go through the curves even though they don’t really do anything at the moment is that maybe in the future if I want to include actual curves, it’s possible to just edit the line function into a curve. I then balance the white and black points for each channel and perform the inversion through “going” back through the linear “characteristic curve”. I thus obtain a positive image for the editing part. I decided not to include any white balance correction baked into this part as it can be subjective and depend on the user if they want to change it, so I decided to do all the white balance adjustments after the conversion with the edits.

[Image 31] Visualization of the conversion steps of my method.



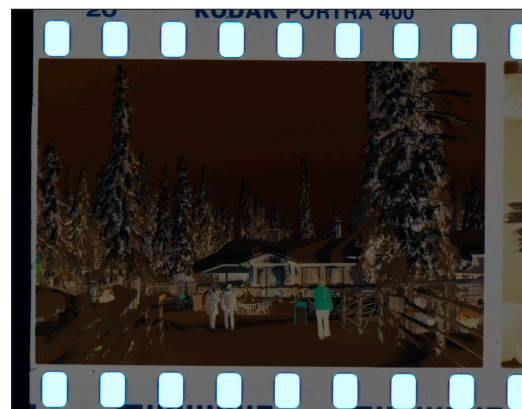
Original negative



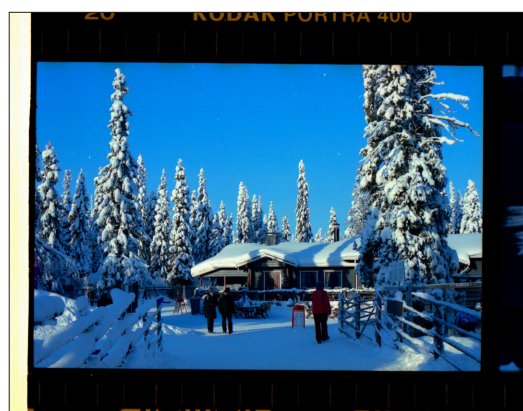
Convert to linear, sample only selected area



Blur the sample area and measure brightest and darkest values



Balance channels with linear normalization



Invert with my method and normalize to 0.0-1.0



Apply gamma correction

The inversion is only a part of the full conversion algorithm. The general algorithm, illustrated in image 31, for the fully automated version in the final version goes like this:

1. If the scanned image has a gamma applied, remove it and convert the intensities to linear space
2. Generate a copy of the negative and apply a box blur filter to average out singular sharp values so that they don't interfere with the automatic black and white value sampling,
3. Sample the darkest and brightest values for every channel from the blurred image
4. Convert these values from measured intensity to density with the formula:
$$\text{density} = \log_{10}(1/\text{intensity})$$
5. Balance all channels using the darkest and brightest values so that every channel's darkest point has the same value and the brightest as well
6. Perform the inversion with my method and normalize the values to span the values from 0 to 1

This algorithm meant that I also had to implement automatic sampling and a box blur filter. I won't go into full detail on how the algorithm works, but I followed the method described in Gonzalez & Woods (2018). The basic principle is:

1. Go through all pixels in the image
2. While processing a pixel, count the average of all pixels in a specific surrounding area. For example if the filter radius is 2, calculate the average from all pixels that are in a box 2 pixels away from the current pixel

3. In a copy of the original image, assign the calculated average to be the new value of this pixel
4. Repeat for every pixel

This box blur algorithm generates a basic blurred image. That is useful since if we have just one pixel that has a really bright value, for example a dust particle or a scratch, it evens it out since that should not be the brightest value for the conversion since it is not part of the actual densities of the negative. It also removes noise and grain from distracting the measurements. This way we get much more consistent results. The simple algorithm described above does the job, but it is really slow. For performance reasons, I had to use an integral image for calculating the averages, but I won't go into detail how that works. It's considerably faster, but doesn't visually affect the outcome at all.

In the conversion algorithm, I also do conversion from one range of numbers to another. This is called linear normalization and it's the same thing that a levels tool does. It means setting the black and white points in the image or to map from one range to another using the formula as explained by Fisher et al. (2003):

$$\text{targetValue} = \text{targetMin} + ((\text{originalValue} - \text{originalMin})(\text{targetMax} - \text{targetMin})) / (\text{originalMax} - \text{originalMin})$$

The way this is done when balancing the channels in step 5 of the conversion algorithm is that we balance the actual values of the scan, not the density values but the actual pixel values not in a logarithmic scale.

Editing

After the conversion, I wanted to have the ability to do basic edits to the photos. The main goal being that I would not have to use any other software if I want to process negatives and publish them. The edits I chose are split into categories of color, intensity and sharpening. The color balance edits are: Saturation and balance for each of the three RGB channels. For intensity edits, I have: density, or in other words, exposure, contrast, and then more specific ones: whites, blacks, highlights and shadows. These are sufficient for me for this purpose. I would also like to add cropping, but for the time being I don't have time for it. For the edits, I had to come up with algorithms or in most cases a math function that I would apply for the whole image that performs that edit. For most of the edits I just had to figure out a function that generates the desirable curve between 0 and 1 as described in the background research. The process of coming up with the final editing curves was to use google, ask AI for help and to play around with different curves in a graphing calculator.



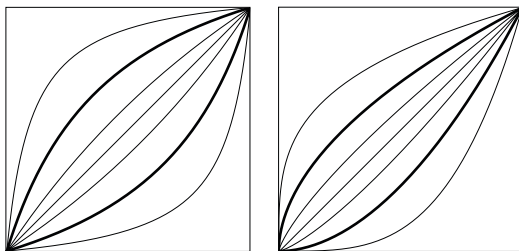
Apply edits

Colors

For adjusting the color balance, I initially used a gamma adjustment for each color channel. The reason for this choice is that I wanted to keep the white and black points unaffected since we set those to be black and white in the conversion. I noticed that a power curve used by the gamma adjustment has a bad habit of clipping the black values when it's large enough, causing considerable color casts in the shadows. With the help of ChatGPT, I eventually found a rational function that basically performs symmetrically for the whites and blacks and keeps them fixed, but increases or decreases the midtones. The final function and a comparison to a gamma adjustment can be seen in image 32:

I also implemented a simple automatic white balance algorithm that I think works surprisingly well. It is basically the gray world algorithm that is described in the background research, but it does not scale the whole channels, but instead applies the scaling factors to the above described color balance adjustments. For this reason, just directly applying the scale factors wasn't enough, but I had to experiment with scaling the scale factors a bit more to get pleasing results. I also adjusted the target neutral grey to not be fully grey, but to get a bit warmer, since that's what I have noticed from my editing workflow, that I prefer my white balance to be a bit on the warmer side and a basic gray world produces too cold white balance for my taste. When I had implemented the gray world auto white balance it was quite easy to implement the neutral spot white balance as well.

[Image 32] My color balance function compared to a gamma correction function as a color balance function



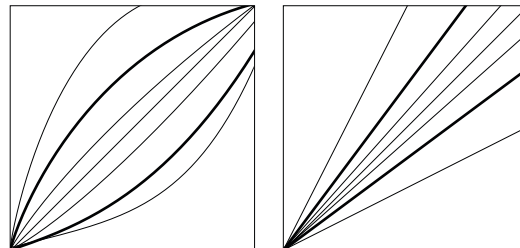
Final color balance function $y = x(x + g(1 - x))$; Gamma adjustment $y = x^g$

where g is the correction amount

Density

For the intensity edits it was harder to find the correct curves. I didn't want the density adjustment to be just a basic multiplication like a physically correct exposure adjustment would be. I wanted a curve that didn't straightway clip all the highlights. It would have to be a function that protects the highlights. I like how Adobe Photoshop's (2026a) brightness adjustment works and I tried to emulate its behavior. When increased, the curve would raise strongly and almost linearly in the shadow area but then not so strongly and linearly when approaching white, preserving the white point for a long time, until eventually changing it a bit. When decreased, however, it should not straightway start to lower the white point, but again decreasing almost linearly in the shadows, but then decreasing less when approaching whites. It should start to reduce the whites though at some point and not be as aggressive in protecting them as when increasing (Adobe, 2026a). I had to use a different curve for increasing and decreasing the density. The final curve compared to a multiplication is shown in image 33.

[Image 33] My density adjustment function compared to a basic multiplication for exposure adjustment.



Final density function Linear multiplication $y = vx$

Increasing
 $y = (1 + v)x - vx^s$

Decreasing
 $y = 0.5vx + 0.5(1 - (1 - v)^3) * x^{(\min(1/v^3, 0.5 + 1/v^2, 2 + 1/v, 7))}$

where s is the "sharpness of the curve" and v is the input value
 $s = 1.14$
 $v = 1.075^v$

Blacks, Whites, Shadows, Highlights

When approaching the Shadows, Highlights, Blacks and Whites adjustments, I again tried to emulate what I was used to in Adobe Lightroom (2026a). I also tested and inspected what happens to the histogram when applying those adjustments in Lightroom. I tested some curves with the curves tool, to look for general direction, where I would want to go when finding curves that would make these adjustments. Based on my inspection of the histogram and my thoughts, I would want the black and white adjustments to change the actual black and white points in the image, but not bluntly as in what the levels tool does. It seems that they also protect the whites and blacks a little before actually starting to adjust the actual point. (Adobe, 2026a) So I knew I wanted a curve that is mostly linear, but at the extreme bends a little, but is still not fully fixed to just one point. After experimentation, I found again that the curve needs to be split into two different curves for the increasing and decreasing. One good thing was that I didn't have to come up with a fully different curve for whites after figuring out the blacks, since I could just move the black curves in the coordinate space to get the white curves, since these curves mirror at (0,0). So the white curves are basically the same curves as the blacks but just mirrored along the line $y = x$ and flipped so that mirrored increasing black is decreasing whites and same for the mirrored decreasing black.

The curves that I came up with for the blacks and shadows adjustments, were handy in the way that with minor tweaks and adjusting the sharpness, I could apply them to the shadow and highlight adjustments as well. The curves had to be adjusted in a way so that they affect a larger area, meaning that they are less linear and more curved and also in addition they shouldn't move the black and white points. The curves for blacks, white, shadows and highlights are shown in image 34.

Final Highlights function

Increasing
 $y = 1 + p(x - 1) + (1 - p)(s(x - 1) / (s - 1 + |x - 1|^{-(v)}))$

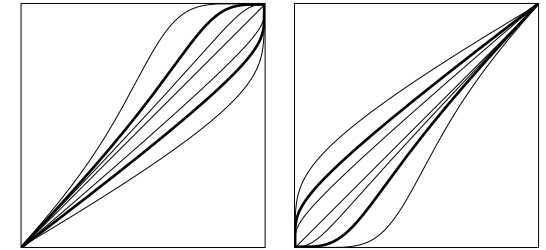
where p is the slope of the section near extremes and s is the "sharpness of the curve" and v is the input value

$p = 0.08$
 $s = 9$

Decreasing
 $y = (1 + v)x - vx^s$

where s is the "sharpness of the curve" and v is the input value
 $s = 15$
 $v = 0.1v$

[Image 34] Functions and graphs for the whites, blacks, shadows and highlights.



Final Whites function

Increasing
 $y = 1 + s(x - 1) / (s - 1 + |x - 1|^{-(v)})$

where s is the "sharpness of the curve" and v is the input value
 $s = 40$

Decreasing
 $y = 1 + (1 + v)(x - 1) - v(x - 1)(1 / s)$

where s is the "sharpness of the curve" and v is the input value
 $s = 5$
 $v = 0.1v$

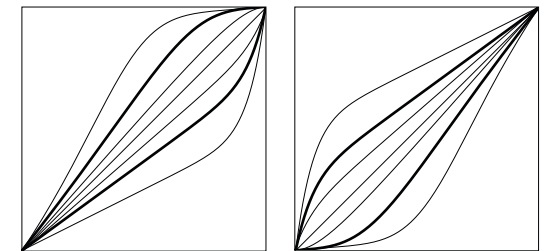
Final Blacks function

Increasing
 $y = (1 - v)x + vx^s(1 / s)$

where s is the "sharpness of the curve" and v is the input value
 $s = 5$
 $v = 0.1v$

Decreasing
 $y = sx / (s - 1 + |x|^{-(v)})$

where s is the "sharpness of the curve" and v is the input value
 $s = 40$



Final Shadows function

Increasing
 $y = 1 + (1 - v)(x - 1) + v(x - 1)^s$

where s is the "sharpness of the curve" and v is the input value
 $s = 15$
 $v = 0.1v$

Decreasing
 $y = px + (1 - p)(sx / (s - 1 + |x|^{-(v)}))$

where p is the slope of the section near extremes and s is the "sharpness of the curve" and v is the input value
 $p = 0.08$
 $s = 9$

Finally, we combine all of the edits together and reapply all of them to the unedited image after every adjustment so as to allow nondestructive editing as stated earlier. When combining the edits together, order matters. I first apply the color balance adjustments since I want them to be applied to the unedited linear conversion. After the color balance though, I apply a display gamma of 2.2 to the image. At first I applied the display gamma after all the other edits, but then I realized that it makes the shadows adjustments affect a much larger area of the image than they should since in the linear image most of the things that our eyes would regard as midtones are regarded as shadow as discussed in the literature review. After the gamma adjustment comes the intensity edits. I apply the blacks and whites, after which come the density and contrast. After those come the shadows and highlights. This order allows some whites or blacks recovery after density adjustments. The white and black recovery after the density adjustment is not fully functional, since due to the nature of the curves being bound to only the are between 0 and 1, every adjustment has to clip at 0 and 1 and not leave any data behind that goes over those values, but you only rarely have to do such big adjustments to the images as you would notice. This could be a future improvement point, but that might mean finding new curves or not to use curves at all for the adjustments.

Sharpening

Sharpening is only applied at the final export, or to the small sharpening preview image and thus it is separate from all the other edits. Sharpening is applied as a basic unsharp mask. What an unsharp mask is, is that you apply gaussian blur to an image, then subtract that blurred image from the original image and the result is called the unsharp mask. The sharpening happens, when you add that mask to the original image (Gonzalez & Woods, 2018). Implementing the unsharp masking behavior was quite straightforward, since I already had the necessary image algorithms created. When you combine multiple box blurs together, you can approximate a gaussian blur (Kovesi, 2010). So I just had to create an algorithm that performs my box blur algorithm multiple times and after that creates the mask by a simple subtraction and adds the mask to the original image by simple add math operation. The final algorithm is as follows:

- 1. Copy the image for blurring
- 2. Perform multiple box blurs to emulate gaussian blur
- 3. Create the mask by subtracting the blurred image from the original for every pixel
- 4. For every pixel in the mask, multiply it by the sharpening amount
- 5. For every pixel, add the mask to the original image

The radius used for the blur sample controls whether the sharpening affects smaller or bigger features and the sharpening amount controls how much of the mask is applied and how much the image is sharpened.

Color Management

One final important factor in the conversion and editing process is to make sure that what the user sees is actually what they get when exporting the image. Thus color management is required with icc profiles as described in the background research. I chose to use the LittleCMS library for implementing color management. In VeryNeg, I decided to convert the incoming image to AdobeRGB as the “working” space since it’s a bit wider gamut than sRGB. The application detects if the image contains an embedded icc profile and then converts it to AdobeRGB. If the image does not contain a profile, I decided to assume that the colors are in sRGB. Then when the user edits the image, the application converts the preview to the display profile that is used by the operating system. This way the user actually sees how the final colors will look like. During export, the user can choose whether to convert the image to sRGB or to just use AdobeRGB. When scanning negatives with an Epson flatbed scanner, I tried to use the Epson scanning software to not include a color profile or to convert the colors from the scanner’s icc profile to AdobeRGB and the difference is clearly visible in image 35. With proper color management the colors are more accurate and should better represent the actual colors as recorded by the film.

3.2.2 Tool Building

I will now proceed to reflect on the process of designing and creating an application and what were the main takeaways from the whole tool design and build process

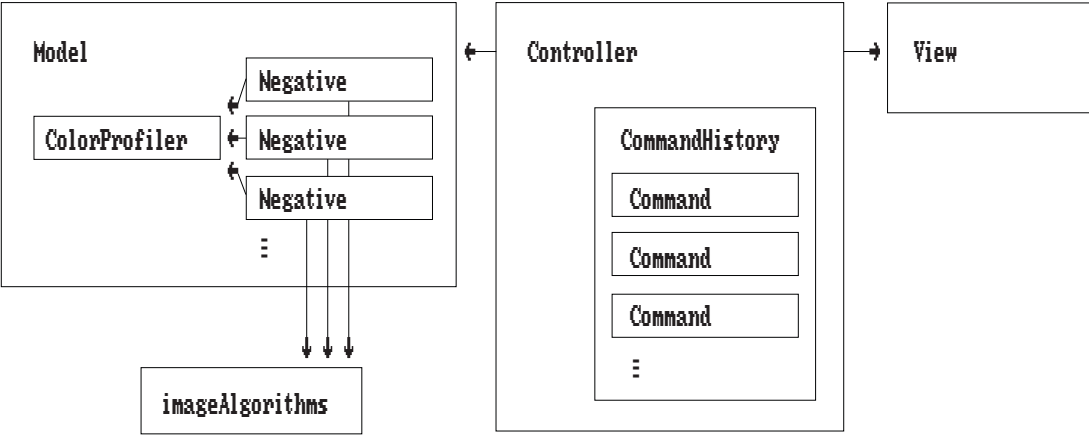


[Image 35] Left: A scan scanned without color management, assumed sRGB when imported to VeryNeg, converted to AdobeRGB and back to sRGB when exporting. Right: A scan with color management, converted during scanning from Epson profile to AdobeRGB. Kept as Adobe RGB in VeryNeg, converted to sRGB when exporting.

Architecture

On the basic level, the architecture of the application follows the popular Model, View Controller pattern (Shim, 2025). A visualization of the architecture is shown in image 36. The Model is code that handles the core logic of the application, handling image opening and saving, image processing and editing. The View contains code that tells, how the user interface should look like and where things should be and what size they are, etc. The Controller handles input from the user and tells the model to change and

[Image 36] Graph of the architecture showing the objects and relationships between them



updates the view to match the data from the model. It works as a mediator between the two. Separating the application this way allows to develop the logic of the application separately of the user interface and the input functionalities and also allows to change those if needed. What this practically means is that only the View and Controller are aware of the existence of SFML and RmlUi and they handle everything to do with the window or input events.

The View specifies in which order things are rendered and in which position using the SFML library. It also uses the RmlUi library to draw the user interface as specified in the RML and RCSS files.

The Controller receives preview images and edit data from the Model and forwards them to the view. It converts the preview images from raw color values to SFML textures and sprites that can be shown using SFML. It controls events and also assigns specific Commands to the GUI elements that tell the model to do something if the user interacts with the GUI. It also controls the main program loop and opens up file-save or file-open dialog boxes from the operating system using a Portable File Dialog library. The Controller owns an object called CommandHistory, that allows it to keep a history of Commands and so allows undo and redo actions.

The Model represents the actual data of the application. It owns Negative objects and handles their creation or deletion and order. The Model owns an object called ColorProfiler that is responsible for converting images from one color profile to another with LittleCMS to keep the application color managed. The Negative objects owned by the Model then use this ColorProfiler

Negatives represent the negative image that is opened by the application. A Negative opens an image and manages all the operations on that image. It manages smaller working and preview images for

the real-time GUI, It performs the actual negative conversion algorithm and also applies intensity, color and sharpening edits to the image. All of those operations are performed non-destructively. All of the conversion and edit data is stored and retrieved from a JSON file that is saved next to the original image file. This allows the user to save the edits performed by the application for other sessions. The image input and output and format and size edits are performed with OpenImageIO. The image conversion and edits are defined by separate image algorithms written by me that the Negative calls. The negative converts the image to a working color space on import and a chosen color space on export by using the Models ColorProfiler. It also converts it to the display Color space when generating a preview for the GUI.

The image algorithms are defined in separate files. These contain mathematical functions for the basic intensity transformations or more complex algorithms for sampling or blurring or sharpening etc. The algorithms apply the operations for every pixel in the image, depending on the algorithm they either do it one by one or can use parallelism to process multiple pixels at once.

Designing the User Interface

Since the main point of the application was to be mainly tailored to my workflow, I got to choose how to organize the user interface. I didn't use that much thought in pondering about radical designs and the final user interface is maybe a bit boring but it was designed with functionality as a first priority. The decision on how to organize everything came quite naturally. The main layout is heavily inspired by Adobe Lightroom. This was a conscious decision since I am so used to the default layout of Lightroom from my professional image editing routines as a photographer. From the early stages of the demo, I already had the image on the left and the buttons and sliders on the right. This is directly from how Lightroom's UI is organized (Adobe, 2026b) and feels natural to me since I am right handed. If I was left handed I wonder if I would have preferred to have the controls on the left? On the bottom is the "film roll" to navigate between images, again inspired by Lightroom (Adobe, 2026b). The application and the UI is not only inspired by Lightroom but I was also heavily impressed and inspired by the software used to operate the old Fuji Frontier professional lab scanner that I covered lightly in chapter 3. I decided to take a sort of a hybrid approach between the way Fuji Frontier approaches editing and what I am used to and prefer from Adobe Lightroom. Like the Frontier, I decided to have the ability to tweak all of the three color channels (linusandhiscamera, 2023; Verbeeck, 2025) unlike in Lightroom which has a white balance

slider (Adobe, 2026b). The Frontier itself doesn't let the user directly adjust shadows, blacks, whites and highlights, but instead relies on tone-adjust profiles (linusandhiscamera, 2023; Verbeeck, 2025). I wanted to have the full manual ability but I also liked the simplicity of the profiles so I included presets inspired by the Frontier that adjust the sliders for you. Since this is a custom application built by me for me I think that it's an interesting hybrid between minimalism but also some professional options that might not be there if it was a plug and play application which in a way it is. I wanted to have the ability to be fast and to have it fully automatic, but I also wanted to be able to manually adjust many of the conversion parameters like the sampling or gamma, if required. This also explains why I wanted to be able to have full control over the intensity edits. Since I wanted this to speed up my workflow, the addition of holding and applying the settings was also essential.

Project Structure and Code Cleanliness

I have never done a software project this large before. For this reason I faced new unexpected aspects of software development. For most of my small programs, I have a map of the program in my head so that I know where everything is located and where to go to edit something. As a programmer, one should keep their code clean, well structured and organized and named. I saw this in action when at some point VeryNegs codebase grew so large that I had trouble having it all in my head to comprehend. Even though I had organized the code well, I sometimes struggled and got lost in the code, having to think for a substantial amount of time about what I was supposed to edit and where it was. What helped in this was to keep naming consistent so I could use the search functions to find things and also if I had to change something to use the tools that automatically change the name in every part of the code that uses it. In some cases it also helped to further split the code more into sub functions or subclasses. Now, I feel like the project could be organized better and especially the way the user interaction and button callbacks are handled could probably be written in a smaller footprint. The model-view-controller architecture should probably also be split into smaller submodules since now the individual classes grow really big. I often had to balance between making things work or to think how to write this cleanly and sometimes the faster dirtier approach won.

Artificial Intelligence

When writing about the development process, it would be insufficient without talking about Artificial Intelligence and large language models or LLMS. I would first like to say that the application was not made with the practice of "vibe coding", meaning that AI does all of the actual programming based on prompts. Most of the code was written by my own human hands. There are some parts that are written by ChatGPT since either in this scope, I knew I didn't have the time to dive deep into the matter to do it myself, such as converting from a float pixel to 8bit pixel and clipping it at 0 and 255. Or repetitive tasks such as filling functions that would be almost copy pasting from another one with slight tweaks. Sometimes I was lazy and didn't want to spend time thinking about how the boundaries of filters radius should be calculated or to translate the command "rotate clockwise" to an EXIF orientation tag code. These were simple things that would still have taken a lot of time and it was faster to use AI for this. But the main part of how I used AI was quick and interactive consulting, ideation, as a search engine or to answer questions about how a library works or how to do some specific thing in C++. I also used it to find errors and for debugging. I think this is where AI shines. It's useful for people who might not be fully confident in coding or undertaking a project where they have a lot of new stuff to learn. I would ask a lot of questions on how some image formats work, how some functions work, how things are done usually and ideas for implementation or to analyze my code. I would argue that if you don't let the AI do all of the work for you but use it as a tool to learn and to receive answers, you can learn a lot.

This however brings us to the flip side of using AI. Even though at this point ChatGPT for example knows a lot and can search the internet, it still often gives false information and hallucinates. And it can be really confident in insisting that it's right. I found this out multiple times when asking about the library SFML. ChatGPT would always use the wrong version or to just come up with functions that don't actually exist. This was probably because it just didn't have the data when it was trained. And this happened often and with different libraries. When using it in debugging, it can also try to look for problems where there are none and to ignore the actual problem. Once I was sure that I just had a small typo or error somewhere and chatGPT was sure it was something bigger. Eventually after an hour I found the problem myself and it was that I just had width and height mixed up in one place and an LLM didn't notice it before me. Now I realize that I would have maybe gotten more accurate answers from editor integrated AI tools but I opted not to use them due to the subscription costs.

5.

Comparison of Conversion Results

*[Image 37] A scanned negative converted and corrected by
using VeryNeg.*

In order to assess the functionality of VeryNeg and to shed more light on the issue of whether there is a correct way to convert a film negative, I wanted to edit a set of photos using the different software tools that I compared earlier in the background research. I chose random images I have shot during the past 10 years. The sample includes many different film stocks and different exposures, properly exposed, underexposed and overexposed.

The tools used in this comparison are: VeryNeg, Negative Lab Pro, FilmLab, Darktable, Negadoctor, Filmomat SmartConvert, Negfix8 and a lab scan and a lab print.

Some photos may miss lab scans and prints that were not available or photoshopped from software that had limitations on their trial periods. The Negatives were scanned using an Epson V370 flatbed scanner in 16bit TIFF and the colors were converted from Epson Standard profile to Adobe RGB when scanning.

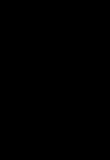
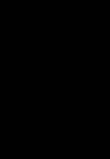
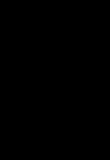
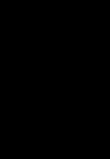
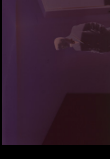
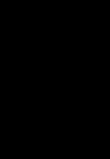
When editing the images, they were edited in isolation from the other tools. I only had one tool open at a time. This meant that I had to make decisions on each image based on how the tools behaved and showed the image and how it guided my decisions. Some images might be darker or lighter, since when using the tool, that's what my intuition told me when editing the image in the tool. In this way, the comparison does not try to exactly match white balance and exposure situations perfectly, but rather, it also reflects on how the tool affects one's workflow at the moment. It also shows how every time we edit a negative, even if it's the same person, the results might look different. Since I have individually edited each photo it's also not a direct comparison of how accurate the automatic conversion of the tool is.

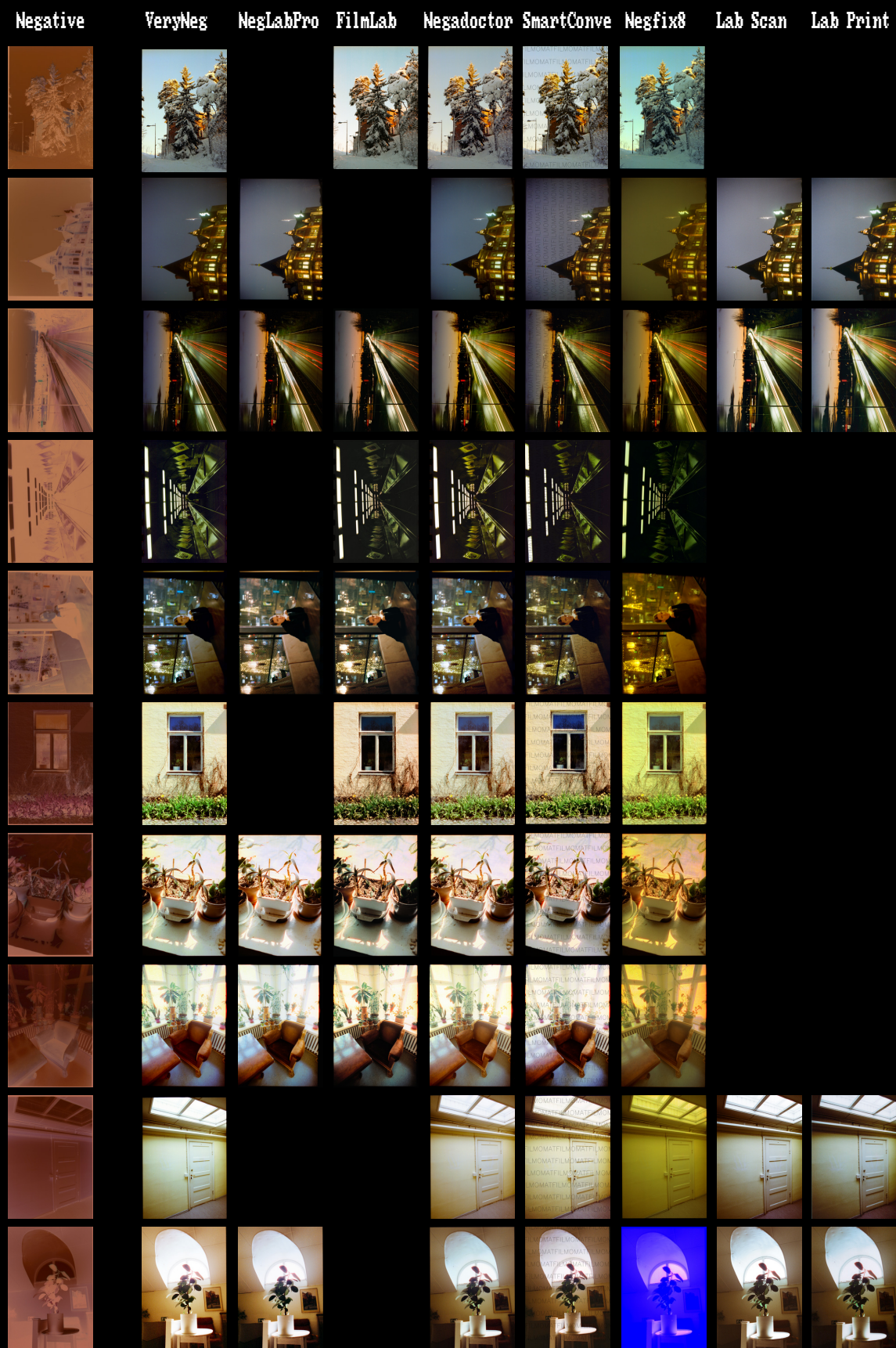
I will be studying the results through both lenses of technical image processing and of artistic interpretation. And the aspects of the results in helping me look at them through those lenses are the following:

- ▮ The overall look and feeling. How pleased am I with the results?
- ▮ Recovery of under and overexposed images. Is it possible and with what kind of results?
- ▮ White and black point color correctness. Is there a color cast in the shadows or highlights?
- ▮ Tonality control. How much of the highlights and shadows and dynamic range can you control?
- ▮ White balance. Can you achieve a color correct image in the midtones?
- ▮ Print emulation look or a look up table. Does the tool apply a specific look or remap colors?

Next I present the conversion results in image 38 in the following pages. On the left is the original negative, with interpretations of it by the different software on the right in that row. Each column shows results of a specific application or source. After the conversions, I present a table comparing the aspects mentioned above (table 2). After that I present my reflections on the conversions.

[Image 38, this and next pages]: Comparison of all the conversions from the different software, VeryNeg and a film lab.

Negative	VeryNeg	NegLabPro	FilmLab	Negadoctor	SmartConve	Negfix8	Lab Scan	Lab Print
								
								
								
								
								
								
								
								
								
								



Negative



VeryNeg



Negative Lab Pro



FilmLab



Darktable Negadoctor



SmartConvert



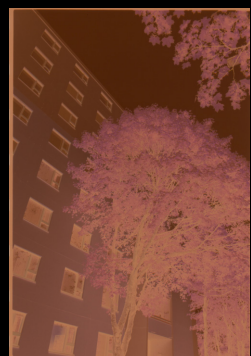
Negfix8



Lab Scan

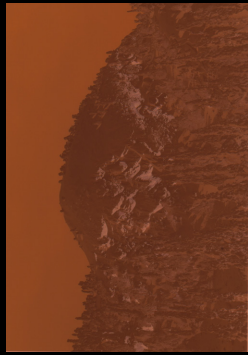


Lab Print



Few selected enlarged negatives

Negative



VeryNeg



Negative Lab Pro



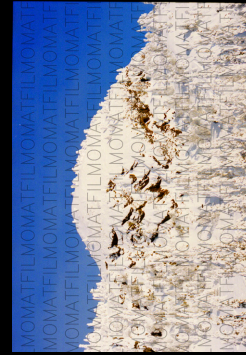
FilmLab



Darktable Negadoctor



SmartConvert



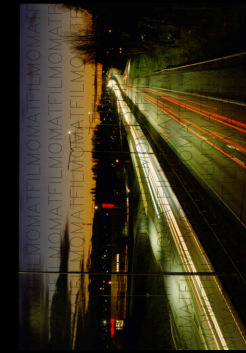
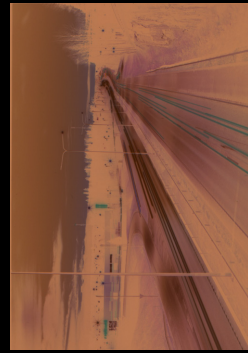
Negfix8



Lab Scan



Lab Print



Few selected enlarged negatives

Appli- cation/ source	VeryNeg	Negative Lab Pro	FilmLab	Dark- table Nega- doctor	Smart- Convert	Negfix8	Lab Scan	Lab Print
Overall look and feel	Clean and neutral. Maybe too boring	Clean and neutral with a little look or interpretation. Pleasing	Strong look that sometimes works, sometimes not	Neutral and clean with little feeling of paper in the tonality. Maybe a bit boring	Quite neutral but a little stuffy and not quite clean results	A failed conversion most of the time, some successes	Clean and pleasing results. A bit on the colder side for me with a little styling or look applied	Really close to a lab scan but a little more saturation and stronger look from the paper
Under and over exposure recovery	Very good	Very good	Not good	Very good	Not good	None	Should be good, but in these cases just okay	Should be good, but in these cases just okay
White and black point color correctness	Very Good	Very Good	Good	Good	Sometimes color casts introduced	Really variable and generally bad	Good. Little green tint in shadows sometimes	Good. Little green tint in shadows sometime. Slight warm tint to the whites
White balance	Good overall performance	Good and option to edit shadow and highlight balance	Okay, really sensitive	Good with two point correction for shadows and highlights	Okay, too sensitive sometimes	Bad	Good, maybe best for hard correction situations	Good, maybe best for hard correction situations
Tonality control.	Very good	Good	Not good	Okay	None	None	Not enough data, probably very good	Not enough data, probably very good
Print emulation look or a LUT	None	Option to have none or to have a strong one or custom ones	Strong look in both tones and colors with warm highlights and cold shadows	A little something in the tones emulating the paper, but nothing visible in the colors	Not much, may also be due to color casts	None	Does have a look applied, probably adaptive to the negative and based on LUT:s	Same as lab scan, but a bit stronger look from the paper

As a first takeaway, the majority of the results are surprisingly close to each other. I am quite surprised with VeryNeg and in my opinion it's quite in a competitive position. The colors are neutral and accurate and the image can be edited a lot, for example to recover underexposed photos.

My other favourites, results wise, from the bunch were Negative Lab Pro and Darktable. With Darktable I would have liked more options to edit the results, such as more sophisticated tonality editing and saturation. When I tried to edit these outside the negadoctor module, I couldn't get them to work since they were always applied before the conversion into the negative image. With further investigation there is probably a way to overcome this problem but I didn't find it quickly. But, overall, I really liked the results and the two step white balance for highlights and shadows works especially well to correct difficult situations.

Negative Lab Pro gave the best automatic conversion out of the box. It also delivered quite consistently good images and I can definitely see why people like it. I like the results too. The settings options are complex, but in the end it was nice to have the ability to adjust the color balance of shadows, mids and highlights separately to get rid of color casts if there were any left.

FilmLab has a really strong look to it. And I can see how it is trying to emulate the printing papers results, if we compare it to the lab prints. But I found this look maybe too much. Often the baked in look felt a bit too cold for me. But in some images the look worked. FilmLab also really struggles with underexposed photos lacking the ability to edit them to look better.

Things that immediately stand out from the conversion is that we can see that Negfix8 really struggled. For some images it did fine, but for others it completely lost it. But it's understandable since it's just a script for the conversion and there aren't any options to edit anything. We can clearly see that they would need more post-processing in an image editor.

Another one that didn't quite give the results that I liked, was Filmomat. Like I said, they say that the application doesn't give any look to the images, but I don't like the overall results as much as the others and I feel like it does add a subtle look to the images. I felt like I lacked control when editing. The images had too strong a contrast or saturation by default and it was hard to get the color balance right

sometimes. There are also some fading and color casts present. But luckily most of these faults are quite small, but I still wasn't that impressed with SmartConverts results.

I included scanned prints when I could in the comparison. These aren't the best for comparisons since the printer isn't calibrated, even though I did use color management. I didn't change the color balance and many prints seem too blue. The only editing I did was to make sure the white and black points were the darkest and brightest. But if I look at the prints with my eyes, I can see that the paper does affect the result a little. It's a little warmer and slightly greener maybe with a little bit of fading of the blacks. There is also the problem that they are not true darkroom prints. Nowadays, prints are first scanned and corrected and then printed. They are still printed to the traditional photosensitive paper so we can see if the paper has any effect, but they are still digitally processed and not fully analog. We can see this in that the prints and scans are really similar.

Even though I did all the edits separately, I find it interesting that the results don't differ that much. Not even from the lab scans. I feel like from this we can derive that there is a general direction of how a negative conversion should look like and so it can be said that there is a technically correct conversion, but then within that correctness there is a large amount of possibilities and interpretations. And we need to remember that most of these images were edited by me so another person's results might differ much more.

[Table 2] Comparison between the different conversion results from the different sources.

6.

Conclusion

6.1 Summary

In this thesis I identified and addressed three problems: lack of knowledge on how design tools work under the hood, lack of agency in the creation of design tools and the over-reliance on Adobe, and lack of knowledge and agency in film negative conversion. I investigated the topic through three lenses: technical image processing, artistic interpretation in image editing and toolmaking and agency. The research was performed through practice based applied research by developing a bespoke and independent software tool VeryNeg for the conversion and editing of scanned film negatives. The goal was to increase our creative agency by knowing and customizing how our tools work. The methods used were iterative prototyping and experimentation supported by documentation and reflection of the process. This was supported by a literature review and case studies and comparative analysis.

In the literature review, I provided a thorough understanding of film and its properties in relation to scanning and conversion. I lifted the shroud on the inner workings of digital images and digital image editing algorithms. Then, I combined both of those themes when researching sources for negative scanning and digital conversion, trying to find and understand the algorithms used.

Based on these findings, in the production part, by practical experimentation I produced knowledge of tool building by automating the conversion process and by creating a free and open source standalone application VeryNeg to perform the conversion. Developing VeryNeg was done in two phases: demo and release development. During the development process, I experimented with different conversion methods to find a solution that works for me. Developing the tool was an experiment to try to prove that it is possible for a designer to make their own design tools and then to reflect on the process.

Finally, I performed a comparative analysis on the conversion results from VeryNeg, different paid and free software and a local film lab.

These methods were chosen to answer the following research question and sub-questions:

How to build a bespoke digital image editing tool for automating scanned film negative conversion?

- How does digital image editing work on the software level?
- How to develop a personal tool for my photography practice?
- How to correctly convert a film negative, what does that mean or is there a correct way?

I will next proceed to offer my suggestions to answers for these questions based on the methods used.

6.2 Findings

How to build a bespoke digital image editing tool for automating scanned film negative conversion?

This main question was split into three sub-questions that together provide an answer:

How does digital image editing work on the software level?

One way to approach digital image editing on the software level is by thinking in terms of a curve between 0 and 1 and how that curve would affect the image and then trying to find a correct mathematical function that produces that curve. To perform edits on an image with code is then applying this function to each pixel. Filtering is another fundamental image editing method. It is used for blurring and sharpening. A filter not only considers the individual pixel but takes into consideration also the neighbouring pixels' placement. Another useful and important image editing function is the levels tool which is achieved with linear normalization.

For automatic white balance, a good and simple option is the gray world algorithm, that compares the image's average color to a target neutral gray and provides scaling factors for each channel.

Image bit depth is an important consideration in editing: to allow the extensive edits required for a negative conversion, the image should be stored inside the editing application with a high bit depth. Gamma and color management with ICC profiles are other important considerations with digital images to make sure that tonality and colors stay consistent between applications.

How to develop a personal tool for my photography practice?

The best technologies for me to develop a bespoke tool for my practice were the combination of a C++ backend with OpenImageIO and SFML libraries. C++ was chosen for speed and the available libraries and my familiarity with it. The chosen way for me to make a user interface with the C++ library was RmlUI with its HTML and CSS based way to create a UI, since I am most familiar with web technologies.

Due to my familiarity with Adobe Lightroom and Photoshop, I emulated aspects of them in the layout and UI of VeryNeg. This was evident with including sliders on the right side's edit bar and a

bottom placed film roll and how some edits behave, for example the density slider, which is inspired by Photoshop's brightness adjustment. I also enjoy the UI of Fuji's front panel and incorporated color balance edits and tonality presets inspired by it. Automation is essential for my workflow with the option to do quite specific manual adjustments. Other requirements were the option to copy/hold settings, and an auto white balance to be a little warmer than usual since that's how I like it to be.

For the architecture, a basic Model, View, Controller model helps me to organize and comprehend the development process better. The practice of making a quick demo to reflect on before starting the actual proper development is a working workflow for my tool building. Time management and a general plan and structure for the project is essential as the codebase grows.

AI can be a good helper in tool building. It can be used as a search engine, for ideation, debugging and as a general helper to increase efficiency and to get instant feedback from. It comes with its downsides of hallucination and making it act as if it knows something when it doesn't, which proved especially hard for debugging.

How to correctly convert a film negative, what does that mean or is there a correct way?

I presented two general directions to convert a negative and made my own algorithm by combining things from both of them. Basic considerations in the conversion are: Consider the gamma of the scan and perform inversion in linear space. Take into account the orange film mask, consider the characteristic curves, balance the black and white points and apply further color correction to the midtones.

You can convert an image in different ways but still obtain almost the same image that is just as usable and that a more complex conversion might not be required. This is evident in how a "simple" conversion surprisingly yields almost the same results as a more complex conversion following the methods outlined by Tuijn (1996).

Film conversion is always an interpretation. There is a general direction of a correct conversion, but it's not an absolute truth and can have a large amount of interpretations.

These interpretations come from how a film negative is a "raw" image. Every positive image is an interpretation of the negative that is affected by almost every step of the scanning and conversion pipeline. Each person and/or tool interprets a negative in different ways.

6.3 Reflection

The main goal of this thesis was the ambitious task to develop VeryNeg and to see if it is possible. Now that the application is developed, it can be concluded that it is possible. The application also stands as a proof for many answers to the research questions.

VeryNeg's goals were to be usable and to work as part of a workflow. I will be using VeryNeg in my workflow. It did achieve the goal of producing usable results and it does provide independence from paying for labs or being tied to Adobe or other tools and thus increases my agency. In the same way, it does save me money. In addition, when I share VeryNeg to others in an open source principle I will help increase the agency of others in addition to my own.

The reason I chose technical image processing as one lens to approach this research was, as stated before, that I feel like there isn't very much talking on design generally on what actually happens under the hood of applications. The literature review and the practical process of iteration and experimentation provided a strong foundation to show some of the inside workings of an image editing application. VeryNeg is a proof that those examples work in practice.

I understand that many designers might not be interested in knowing what happens on the code level of tools since it is very technical and might be hard to understand. But in my opinion, there is value in trying to understand our tools better inside. Even if we try a little, it increases our agency to understand the tools and help us to use them better and to choose the correct tool for the job. I for sure did gain much more agency. I now understand how other tools such as Photoshop work a little better as well.

Turning our attention to the more specific technicalities of film conversion, as I emphasised in the introduction, I have had trouble in the past in finding definite methods for the conversion and there appears to be a lack of detailed explanations on how to convert a film negative properly. A lot of this is very technical, but it also involves some interpretation and for that reason I chose the lens of artistic image interpretation in addition to the technicalities. With the help of the literature review and practical experimentation, I now understand it a little better, but there is still no definite answer to the conversion. Maybe there doesn't have to be one. I cannot give a concluding answer on how the conversion happens, but I have provided a few different methods that seem to work and that are probably used to a variable degree in film conversion software and that I hope might be enlightening to someone interested in film conversion.

My findings from the literature review, experimentation with VeryNeg and the comparison of results

from different conversions show that the conversion itself and the question on correct conversion walks a weird path between two aspects. On one hand we have quite restrictive mathematical functions, algorithms and characteristic curves and on the other we have individual preferences and artistic vision. The conversion can look vastly different but still be counted as correct, but then I feel like one can quickly see that something is not correct if some specific thing in the conversion is wrong. An example in this would be if the black or white point is not aligned correctly causing for example yellow whites or a blue tint in the blacks. But then again, even this could be used as a way of artistic expression. And especially for blacks we might be used to having a greenish tint in them from the results of lab scanners that we are used to. So film negative conversion is a really interesting balance between correctness and interpretation.

If we consider the lens of tool making and agency It would be awesome if every designer could tailor their tools to their practice or to create bespoke tools. As I stated above, VeryNeg does increase my agency. However, based on my experience with developing VeryNeg, still for the general designer without the knowledge about programming, this would be impossible to do. So the practice of building one's own tools is not very accessible. AI could maybe help and has helped me a lot but I still know what almost every line of code does and I understand it and that is quite essential for a bigger application like VeryNeg, that's still very small in the scale of multi purpose design applications. Some of the algorithms and formulas started to go really deep and required a lot of time for me to understand and I am still a beginner in software development. Developing applications is hard, especially if one wants to share the app. Even if creating an application is still hard, I hope that my research on the topic of programming an application as a designer has sparked interest and shown hope to aspiring designers that are interested in programming and are brave enough to take on the task of creating an application. It can be done!

As noted by Ingo in his book on open source (2005), software is not material and there is no limitation to copying it and there are no production costs associated with the act of producing more copies. Thus the idea of open source is that it should be shared and that's how open source thrives. (Ingo, 2005) Even though this tool was meant to be bespoke for me, in a way, I still thought about how others would use this tool. I naturally want to share and show the tool to others to use as well and to contribute to their options to increase their creative agency. This posed the interesting question of how much it unconsciously affected some of my decisions and does that mean that maybe the app is not fully bespoke to me?

Another thought on how much agency I really have is how my experience with Adobe's tools affected my decisions when making VeryNeg. I am so used to Photoshop and Lightroom that they have had a huge influence on me and have shaped the way I approach photo editing. There is a reason why VeryNeg reminds you of Lightroom, since that's what I've been using professionally for my digital photography and that's what I am used to. I also chose to use adobeRGB for the working color profile since it's known to many and for me and what I might use in Photoshop. So I wonder is VeryNeg in this way fully independent in the end?

With these final reflections we can turn our attention to the limitations of this thesis

Limitations

The study of image editing on the code level is interesting, but this thesis has only scratched the surface. Image processing is a discipline of its own and this thesis can only provide a few examples of it. The same can be said of color spaces and calibration. I feel like the research could have gone a bit further into detail regarding those but that wasn't possible since the project is already quite big and more general. There could be a separate project focusing only on color correct workflows or digital image editing.

I could have had more applications to compare but since the research has grown a bit too large already, I could only include a few. The thesis also had to abandon a more general study of tools from the literature review.

Right now VeryNeg isn't quite ready to be shared yet and currently only works on a Mac. It still has bugs. There are also many features I could add. The code itself should be cleaned also and many things could be done more efficiently. The development also took longer than anticipated. Even though I anticipated it to take longer, it still took longer than the anticipated amount and it was hard to say when to stop and say that I have enough for now.

For further investigation on film negative conversion, there still exists the problem that many software and scanners are commercial black boxes and don't expose the algorithms used for researching. It was also a challenge to find open descriptions or academic papers on the subject. Those were then often hard to understand.

This research was largely based on my preferences of editing and how I judge if a conversion is pleasing or not. In contrast to this, an empirical study of photographers and image editors and how their interpretation results would differ or not differ from each other would be interesting.

This thesis also focuses on a medium that's future feels uncertain. How relevant will film remain?

Will film live on or die at some point? In a way, it's funny trying to produce a modern application for an old practice and trying to recover information from software from over 10 or 20 years ago. I personally think that film will live on but hope that it won't go too crazy in prices and thus be out of reach in the future.

6.4 Future

During this research I have had many ideas of other possible derived subjects for research. A natural progression from this research would be to apply knowledge from this research to research on film emulation in digital images. It could research and create methods for film emulation, maybe a tool even.

Even though here we are trying to achieve a "correct" image, it could be interesting to see how an image editor using math and random different graphs could give new artistic possibilities to image editing. Combining math and art in an image editor, where you can directly apply a mathematical function or algorithm to the image along with conventional image editing methods. I haven't looked further into it but I think it's possible even with Photoshop or Gimp but not that often used.

From the general view of tool building, it would be interesting to develop the idea of a tool builder or an AI tool builder further. This research investigated tool building in the niche of film negative conversion and editing, but a more general research on the practice and idea of making tools would be interesting as well. AI could be studied from the perspective of learning based or AI based learning from edits of converted negatives to that way figure the average "correct way" or a specific style from a person to help them edit in their style.

VeryNeg can be and will be developed further in the future. Even though the conversion algorithm implemented in the current version performed fine for all the test samples, further safety could be built into the algorithm. What if the white point of the image is actually a neon green light? The current version also lacks the option to edit black and white negatives. VeryNeg could become an all inclusive film negative conversion application, adding the option of editing digital camera raw scans, better white balance algorithms, more auto options and adaptive white balance inspired by professional film scanners.

For now though, VeryNeg is enough for me to actually scan the negatives that I have hidden away in my drawer and to show those images to the world. I achieved some independence from film labs and other software and I can now go out and shoot more film photos, knowing that they have a smaller chance of gathering dust in my drawer.





References

- Academy Software Foundation. (2026). *OpenImageIO* (Version 3.1.12.0) [Computer software]. <https://sites.google.com/site/openimageio/home>
- Adobe. (2026a). *Adobe Photoshop* (Version 27.1.0) [Computer software].
- Adobe. (2026b). *Adobe Lightroom Classic* (15.0.1) [Computer software].
- Alchemy Color. (2024). *A scientific method for inverting color negatives* [Video]. YouTube. <https://www.youtube.com/watch?v=0YC6YzmXmD0>
- Blender. (2026). *Blender* (Version 5.1.2) [Computer software]. Blender Foundation.
- Buchler A. (2023). *Scanning color negative film*. <https://abpy.github.io/2023/08/20/color-neg.html>
- Burke, A. (2019). *Manual inversion of color negative film*. <https://www.alexburkephoto.com/blog/>
- Captured by Sam. (2024). *I scanned my film with 4 different scanners* [Video]. YouTube. <https://www.youtube.com/watch?v=cw8zFa8heyc>
- Case, D. (2001). *Film technology in post-production* (2nd ed.). Focal Press.
- Darktable developers. (2026) *Darktable* (Version 5.4.1)[Computer Software]. <https://www.darktable.org/>
- Day M. (2024). *How I convert my film negatives* [Video]. YouTube. <https://www.youtube.com/watch?v=3rvVplUQdWQ>
- Develop and Fix, inc.. (2025) *FilmLab* (Version 3.4) [Computer Software]. <https://www.filmlabapp.com/>
- Dorsky, E. (2024). *Why is color negative film orange?* <https://observablehq.com/@dorskyee/understanding-color-film>
- Filmmaker IQ. (2018). *Diving into dynamic range* [Video]. YouTube. <https://www.youtube.com/watch?v=2sshGdMgJxQ>
- Filmomat. (n.d.) *SmartConvert Demo* (Version 3.4) [Computer Software]. <https://www.filmomat.eu/smartconvert>
- Fisher, R., Perkins, S., Walker, A. & Wolfart, E. (2003). *Contrast stretching*. HIPR2, University of Edinburgh. <https://homepages.inf.ed.ac.uk/rbf/HIPR2/stretch.htm>
- Foley, J. D., van Dam, A., Feiner, S. K., & Hughes, J. F. (1996). *Computer graphics: Principles and practice* (2nd ed.). Addison-Wesley.
- Fujifilm. (n.d.-a). *Film simulation*. <https://www.fujifilm-x.com/global/products/film-simulation/>
- Fujifilm. (n.d.-b). *SP-500 film scanner instruction manual*. <https://www.scribd.com/document/564378100/>
- Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing* (4th ed.). Pearson. <https://www.cl72.org/090imagePLib/books/Gonzales%2CWoods-Digital.Image.Processing.4th.Edition.pdf>
- Grey, T. (2004). *Color confidence*. SYBEX.
- Hirsch, R. (2015). *Exploring color photography: From film to pixels* (6th ed.). Focal Press.
- Ingo, H. (2005). *Avoin elämä: Näin toimii open source*. Otamedia.
- JaZ99. (2013). *Negfix8* (Version 1.3) [Computer software]. <https://github.com/chrishunt/negfix8/tree/master>
- Kodak. (2006). *Basic photographic sensitometry workbook*. Kodak. <https://www.kodak.com/content/products-brochures/Film/Basic-Photographic-Sensitometry-Workbook.pdf>
- Kodak. (2007). *The essential reference guide for filmmakers*. Kodak. <https://www.kodak.com/content/products-brochures/Film/kodak-essential-reference-guide-for-filmmakers.pdf>
- Kodak. (n.d.). *Shot on film*. <https://www.kodak.com/en/motion/page/shot-on-film/>
- Koraks Tinkers. (2023). *Upside down – How color film and paper are fundamentally different*. <https://tinker.koraks.nl/photography/upside-down-how-color-film-and-paper-are-fundamentally-different/>
- Koskinen, S. (2025). *Computational color constancy using spectral domain*. <https://urn.fi/URN:ISBN:978-952-03-3735-3>
- Kovesi, P. (2010). *Fast Gaussian smoothing*. In *Proceedings of the 2010 International Conference on Digital Image Computing: Techniques and Applications (DICTA)* (pp. 121–125). IEEE. <https://www.peterkovesi.com/papers/FastGaussianSmoothing.pdf>
- Langford, M. (2000). *Basic photography* (7th ed.). Focal Press.
- Lin, Q., & Tretter, D. (2004). *Digital processing of scanned negatives* (HPL-97-16R1). Hewlett-Packard Laboratories. https://www.researchgate.net/publication/220865339_Digital_Processing_of_Scanned_Negatives
- linusandhiscamera. (2023) *My \$10,000 Film Scanner* [Video]. YouTube. <https://www.youtube.com/watch?v=r7vOuArDnxU>
- Maudet, N. (2017). *Designing design tools* (Doctoral dissertation, Université Paris-Saclay). <https://theses.hal.science/tel-01827014v2>
- McDougall, K. (2024). *The scanner I always dreamed of owning* [Video]. YouTube. https://www.youtube.com/watch?v=KfuUJ_NOTT8
- Molina-Pérez, D., & Rojas-López, A. G. (2024). *Resolving Contrast and Detail Trade-Offs in Image Processing with Multi-Objective Optimization*. *Mathematical and Computational Applications*, 29(6), 104. <https://doi.org/10.3390/mca29060104>
- Nate Photographic. (2024). *Negative Lab Pro* (Version 3.1) [Computer Software]. <https://www.negative-labpro.com/>
- Noriega, L., Morovic, J., Lempp, W., MacDonald, L. (2001). *Colour characterisation of a digital cine film scanner*. In *Proceedings of the Color Imaging Conference (CIC 2001)*, 9(1), Article art00044. <https://doi.org/10.2352/CIC.2001.9.1.art00044>
- OpenEXR. (n.d.). *Technical introduction to OpenEXR*. <https://openexr.com/en/latest/TechnicalIntroduction.html>
- PARVEC. (2025). *Every 35mm film color science explained – Part 1* [Video]. YouTube. <https://www.youtube.com/watch?v=AVFTtE-8C94>
- Poynton, C. (2018). *Colour appearance issues in digital video, HD/UHD, and D-cinema* (Doctoral dissertation, Simon Fraser University). <https://www.poynton.ca/pdf/Poynton-2018-PhD.pdf>

Qian, Y., Kämäräinen, J.-K., Nikkanen, J., & Matas, J. (2019). *On finding gray pixels*. arXiv. <https://doi.org/10.48550/arXiv.1901.03198>

RetroPixel. (2018). *Converting Negatives Into Positives [FOR FREE | Without Any Plugins]* [Video]. YouTube. <https://www.youtube.com/watch?v=vHnw8hZdd18>

The RmlUi Team. (2026). *RmlUi* (Version 6.2) [Computer software]. <https://github.com/mikke89/RmlUi>

Safelight Berlin. (2024). *Frontier vs. Noritsu scanner comparison* [Video]. YouTube. <https://www.youtube.com/watch?v=h1Ah0Om7SEU>

Saguer, M. (2026). *Little CMS* (Version 2.19) [Computer software]. <https://www.littlecms.com/>

Serteep, H. J. (2024). *Subject review: White balance techniques of digital image*. In *Global Journal of Engineering and Technology Advances*, 20(01), 179–185. <https://doi.org/10.30574/gjeta.2024.20.1.0129>

SFML Developers. (2026). *Simple Fast Media Library* (Version 3.1.0) [Computer software]. <https://www.sfml-dev.org/>

Sharma, A. (2018). *Understanding color management*. John Wiley & Sons, Incorporated.

Shim, J. (2025). *The architecture of web-based graphic editors*. <https://dev.to/feconf/the-architecture-of-web-based-graphic-editors-and-7-design-patterns-part-1-2l43>

Solomon, C., & Breckon, T. (2011). *Fundamentals of digital image processing: A practical approach with examples in MATLAB*. John Wiley & Sons.

Stone, J. (2016). *Darkroom dynamics*. Routledge.

Sweeting, K. (2021.). *How to scan film on a Fuji Frontier SP-3000* [Video]. YouTube. <https://www.youtube.com/watch?v=LFMJZeIUNhI>

Taylor, J. (1983). *Colour printing in practice*. Element.

The Cine Network. (2021). *Advanced emulsion* [Video]. YouTube. https://www.youtube.com/watch?v=I4_7tW-cx1I

The MathWorks, Inc. (n.d.). *Comparison of auto white balance algorithms*. <https://www.mathworks.com/help/images/comparison-of-auto-white-balance-algorithms.html>

mathworks.com/help/images/comparison-of-auto-white-balance-algorithms.html

Santo, T. (2024). *Color Negative Film Conversion - The Easy Way!* [Video]. YouTube. <https://www.youtube.com/watch?v=aZTgHudKaIO>

Tran, A. (2019). *How to Edit Negative Scans and keep True Film Colours* [Video]. YouTube. <https://www.youtube.com/watch?v=mMOSUqgcJow>

Tuijn, C. (1996). *Scanning color negatives*. Proceedings of the Color Imaging Conference, 4(1), Article 00010. <https://doi.org/10.2352/CIC.1996.4.1.art00010>

Vandebroek, C. (2026). *Frontier vs Noritsu: Why film scans look different – and how to get the best results*. Optik Oldschool. <https://www.optik-oldschool.com/blogs/blog/why-we-prefer-frontier-scans-for-opticolour200>

Verbeeck, W. (2025). *I bought my dream film scanner* [Video]. YouTube. <https://www.youtube.com/watch?v=D1UrzdE8YLo>

Verbeeck, W. (2023). *Printing film negatives in my darkroom* [Video]. YouTube. <https://www.youtube.com/watch?v=58lK5Gs5JYo>

Walker, M. E. (2006). *Complete guide to digital color correction*. Lark Books.

Zeidenberg, M. (1996). *Kill the king: Monopolies, alliances, and litigation in the American computer industry* (Master’s thesis, University of Wisconsin–Madison). <https://www.mattzeidenberg.com/publication/zeidenberg-kill-1996/zeidenberg-kill-1996.pdf>

7.1 Image references

[Images 1,2,5,16,21,26,29,31,37,38] Koponen, J. (2019-2024) *Scanned film photographs* [Photographs].

[Image 3] Fujifilm. (n.d.). *Film Structure* [Illustration from a product information brochure]. Product Information Bulletin Fujicolor C200. Japan: Fujifilm Corporation. 5. https://asset.fujifilm.com/master/emea/files/2020-10/98c3d-5087c253f51c132a5d46059f131/films_c200_datasheet_01.pdf

[Images 4,8,36] Koponen, J. (2026). [Infographics].

[Images 6,7,9,10,13,32,33,34] Koponen, J. (2026). *Graphs of mathematical functions* [Graphs created with Desmos graphing calculator: <https://www.desmos.com/calculator>].

[Image 11] Captured by Sam. (2024). *I Scanned My Film With 4 Different Scanners* [Still frame from video]. YouTube. <https://www.youtube.com/watch?v=cw8zFa8heyc>

[Image 12] Kodak. (2016). *Kodak Gold 200 Film Curves* [Illustration from a product information brochure]. 2016. Kodak Gold 200 Film Technical Data. New York: Kodak Alaris Inc. 4. https://business.kodakmoments.com/sites/default/files/files/resources/E7022_Gold_200.pdf

[Image 14] Koponen, J. (2026). *Lin & Tretters negative conversion pipeline* [Infographic], interpreted from: Lin, Q., & Tretter, D. (2004). Digital processing of scanned negatives (HPL-97-16R1). Hewlett-Packard Laboratories. https://www.researchgate.net/publication/220865339_Digital_Processing_of_Scanned_Negatives

[Image 15] Koponen, J. (2026). *Visualization of Tuijn’s conversion method* [Infographic], interpreted from: Tuijn, C. (1996). Scanning color negatives. Proceedings of the Color Imaging Conference, 4(1), Article 00010. <https://doi.org/10.2352/CIC.1996.4.1.art00010>

[Image 17] Nate Photographic & Adobe. (2026). *Negative Lab Pro UI* [Screenshots of Negative Lab Pro in Adobe Lightroom taken by Koponen, J. featuring photos taken by Koponen, J.].

[Image 18] Develop and Fix, inc.. (2026). *FilmLab UI* [Screenshots of FilmLab taken by Koponen, J. featuring photos taken by Koponen, J.].

[Image 19] Filmomat. (2026). *SmartConvert UI* [Screenshot of SmartConvert taken by Koponen, J. featuring photos taken by Koponen, J.].

[Image 20] JaZ99. (2026). *Negfix8 usage from the command line* [Screenshot of Negfix8 in the command line taken by Koponen, J.]

[Image 22] Darktable developers. (2026). *Darktable UI*. [Screenshots of Darktable taken by Koponen, J. featuring photos taken by Koponen, J.].

[Image 23] Darktable developers. (2026). *Darktable’s Negadoctor module UI*. [Screenshots of Darktable taken by Koponen, J.].

[Image 24, top] Gulabi Photo. (2021). *Tone Adjustment submenu* [Screenshot]. Gulabi. https://images.squarespace-cdn.com/content/v1/5cfe3d458dc4ed00010dc-d00/1617536144428-9CQM1QXGECJ9AFMH-QWHA/Screenshot_fuji-s2-jodie-foster_2021-04-02_13%3A32%3A17.png?format=2500w

[Image 24, bottom] Gulabi Photo. (2021). *That time Ronnie James Dio visited the lab and did some scanning*. [Screenshot]. Gulabi. <https://images.squarespace-cdn.com/content/v1/5cfe3d458dc4ed00010dc-d00/1617538419322-2935PXLVATAJPHTER-A9T/Untitled-1.gif?format=2500w>

[Image 25] Fujifilm. (2005). *Digital Minilab Frontier 570*. [Photograph from sales brochure]. 2005. Digital Minilab Frontier 550/570/590. Japan: Fuji Photo Film co., Ltd., 1. <https://www.minilab.ca/dat/files/200.pdf>

[Image 27] Koponen, J. (2026). *VeryNeg demo UI* [Screenshots taken by Koponen, J. featuring photos taken by Koponen, J.]

[Image 28] Koponen, J. (2026). *VeryNeg UI* [Screenshots taken by Koponen, J. featuring photos taken by Koponen, J.]

[Image 35] Koponen, S. (1998). *Joonatan with dad* [Photograph].

[All other illustrative images by Koponen, J.].

