Creating an open source color management lab

Shankhya Debnath*

Regional Institute of Printing Technology

* This laboratory would not have been set up without the tireless efforts put into the movement that is FOSS. I am immensely grateful towards the contributors and creators of ArgyllCMS, LittleCMS, LittleArgyllGUI, CoCa, LProf, Python and the entire open source community who values the concept of free sharing of knowledge more than commercialization.

Table of Contents

Creating an open source color management lab 1
Shankhya Debnath
1 Why this lab was set up? 5
2 How this laboratory came into existence? 6
3 Pre-Requisites 12
4 Using the reflective scanner as densitometer [23] 13
5 Creating the scanner profile 14
5.1 Preparing the IT 8.7/2 scanner calibration test target 14
Documentation from LProf [37] 17
Notes on Profile Parameters 19
Evaluating scanner profile quality using Profile Checker 20
Preferences 20
Profile Identification 21
Monitor Tab 21
Setting Gamma 23
Using the Install Reference File Dialog 29
6 Argyll CMS [40] 30
6.1 Installing Argyll in Windows 30
Unpacking the .zip archive 30
Making the tools accessible 31
6.2 Copyright, Licensing Trade Mark 31
6.3 Main Tools and the command line 33
6.4 The Argyll Workflow 34
6.5 Main Tools by Category 34
Calibrating devices 34
Creating test targets for profiling or print calibration 34
Obtaining test results for profiling or print calibration 34
Creating Device Profiles 36
Creating Device Link Profiles 36
Converting colors or applying print calibration 36
Color Tweaking tools 36
Creating gamut views 37
Diagnostic and test tools 37
Other Tools 37
File formats that Argyll uses [90] 37
6.6 Related technical information 38
Calibration vs. Characterization 38
Fluorescent Whitener Additive Compensation (FWA Compensation) . 39
About ICC profiles and Gamut Mapping 43
About display monitor settings and targets 46
Crushed Display Blacks 48

7 Scenarios possible with Argyll 50
7.1 Display profiling 50
7.2 Profiling Scanners and other input devices such as cameras 50
Types of test charts 50
Taking readings from a scanner or camera 50
Creating a scanner or camera input profile 53
7.3 Profiling printers 54
Creating a print profile test chart 54
Printing a print profile test chart 56
Reading a print test chart using an instrument 57
Reading a print test chart using a scanner or camera (NOT recommended) 57
Creating a printer profile 58
Choosing a black generation curve (and other CMYK printer options) 59
Overriding the ink limit 64
7.4 Calibrating Printers 65
Calibrated print workflows 65
Creating a print calibration test chart 66
Creating a printer calibration 67
Using a printer calibration 68
How profile ink limits are handled when calibration is being used ... 69
7.5 Linking Profiles 69
7.6 Image dependent gamut mapping using device links 70
7.7 Soft Proofing Link 71
7.8 Transforming colorspaces of raster files 71
8 LittleArgyllGUI 71
8.1 Installation 72
8.2 Setup 72
8.3 The working folder and base name 73
8.4 Usage 73
9 Profiling using CoCa 74
9.1 Usage 75
Target Image 75
Visual Check 76
Reference File 76
Corrections 76
Tweaks 77
Select Target Type 77
Profile Information 77
Options - Algorithm 77
Options - Quality 77
9.2 Creating the Profile 77
9.3 Using the Profile 77
10 PyCharm Documentation 78
10.1 Standalone Installation of PyCharm on Windows OS 78

10.2 Open/Create a project in PyCharm . . . 79
Why do I need a project? . . . 79
Look around . . . 82
Customize your environment . . . 83
Code with smart assistance . . . 85
Generate some code . . . 86
Find your way through . . . 86
Run, debug and test . . . 87
Keep your source code under Version Control (VCS) . . . 88
11 Installing the colour-science package [8] . . . 89
12 Installing OpenCV [95] . . . 90
13 Installing NumPy [96] . . . 91
14 Some ideas to try out . . . 91
15 Conclusions . . . 91

Creating an open source color management lab

1 Why this lab was set up?

A priority in industrial print production is to get the right color first time and every time. Consequently, methodologies in process and product control and their standardization is a continuous effort. Some of the well established standardized protocols for industrial print process control include ISO 12647, GRACol, SWOP, G7 and more recently ISO/PAS 15339 (CRPC). An increasingly large number of printers today in this country are adopting such standardized practices in their production workflows. At the core of such standards exists the idea to manage colors across devices and platforms from a variety of sources. Notwithstanding mandates from print-buyers, reproducing and bringing in color accuracy and constancy in all devices plays a major role in determining the business that printers are engaged in. Hence, implementing standards is no longer limited to a select few but has gained enormous momentum in the recent years in India. This is an indication that expertise in implementing such workflows in actual production environments is of considerable value for prospective employers. As one of the long standing institutes serving the human resources requirements in the printing industry across the nation, it is only right that the institute works in developing and honing such expertise among its students for enhanced opportunities among prospective employers nationally.

The primary challenge in communicating color across varied devices and platforms stems from the fact that color is reproduced differently based on reproduction mechanisms of individual devices. In order to maintain constancy of color appearances across input and output devices, device spaces are normally transformed to independent or connecting spaces. However, such conversions are extremely non-linear and this requires either modelling all intrinsic parameters of the device or using a sufficiently large training data. Many printers as output devices print continuous-tone images by converting them into discreet spatial regions, known as halftone dots. The halftone dots comprises some part of the digital pixels in a manner where both the regions on output and input devices match visually [1]. For a close visual match, the halftone dots are varied in size or frequency or both. Murray-Davies [2] and Neugebauer [3] models are based on the premise that the overall reflectance of the halftone system is some weighted-function of individual reflectance of the dots and the substrate for single colorant printing and different colorants and substrate for multi-color printing, respectively. However, owing to scattering of light through paper and its subsequent absorption by the ink dot causes an apparent increase in the dot size, which is known as optical dot gain. Apart from this, the nip pressure between the cylinders through which the substrate passes causes the dot to increase in size, this is known as mechanical dot gain. Some corrections are required in the reflectance prediction models as mentioned before, so that the model accounts for such non-linearity. The first approach of characterization, is often called a model-based technique for printer characterization. All physical modelling approaches described in this article considers only halftone printers. Continuous-tone printers however are characterized based on Kubelka-Munk model, where tonal gradation is created using variable colorant

concentration. Physical models are created based on device parameters. Models for halftone printers maybe defined [1] by either expressing reflectance as a function of halftone dot-area or by expressing colors as a function of spectral reflectance. Physical models are computationally efficient as they require characterizing devices using relatively smaller number of training data, but efficiency of the model depends heavily on the effective derivation of the model parameters. In certain cases measuring comparatively limited number of patches from multi-colorant ramps maybe sufficient for identifying and computing model parameters.

Numerical models on the other hand considers the device to be a black-box [4], whose dependencies are unknown. With this assumption, numerical methods involve making a sufficiently large, known training data which is printed through the device to be characterized, measured and a characterization function is derived in the form of three-dimensional look-up tables (LUTs). A suitable interpolation technique is then devised to transform test data. For instance, such models might require a standard IT 8.7/3 test chart having some hundreds of patches or even more, for measuring colorimetric training data and creating the model. Efficiency of the method can be computed by calculating distance between predicted and measured data.

One bottleneck however in terms of implementing color managed workflows at the educational institute level is the dissimilarity in scale of infrastructure at offer in the industry and that at the institute level. One of the primary reasons amongst many is the fact that the purpose of both the above mentioned entities are quite different. On one hand when the industry is catering to a volume of few millions impressions per month, the same can not be said for an educational institute. A proof of concept approach in development of infrastructure at the institute level appears to be a win-win situation.

2 How this laboratory came into existence?

Any industrial color managed workflow requires both extensive almost state-of-art hardware and software infrastructure. However, under the given circumstances, it was not possible to develop this massive infrastructure that essentially has no commercial return. So the problem at our end was actually a ”Scaling-Down” issue. For instance, a full-blown multicolor offset machine maybe substituted with a desktop variant of an inkjet printer. The other equipment available with me is a Windows 10 based desktop PC and a faltbed reflective type scanner.This is in no way to suggest that they are to be equated. That these printers are fundamentally different is common knowledge. It only means, that both can print multicolor artworks in discrete forms to give an illusion of a contone image.

With the printer available, it is time to search for profile creators and measurement devices. The idea to create a color managed workflow for a printer kept lingering in my mind for the past two years. Failed attempts were initally made to procure large format industrial grade digital printers with retrofitted color profiler and measurement devices. It was my understanding that premium and proprietary software were mandatory for managing colors and creating bespoke workflows. Personally, early on, I would like to state my ignorance on the impact that creators like Graeme Gill [5],

Marti Maria [6], Thomas Mansencal [7] and many other opens source authors would have in developing this laboratory and help us and this institute. In fact, it was fairly recently that I started learning Python and scratch the surface of what was possible using it. As I have very little to no knowledge at computer science or coding, the learning curve for me is fairly steep in any such exercise. At a point when I thought numerical color computation was not possible without professional and proprietary software, it was then I stumbled upon the colour-science package [8]. It was an eye-opener and incredible to use. Since then, I have been using the package along with the Numpy, OpenCV, Matplotlib and many more in PyCharm's Community Edition IDE [9] with Conda [10] as the interpreter to solve all the computational color based problems.

I would like to get distracted a little at this point. There exists an idea, and quite rightly to some extent, that any serious work be it in research or in the industry mandates the requirement for proprietary software (which is always a closely guarded and might I say highly priced proposition). It was during my University days when we used some proprietary software for numerical computational work related to image processing. One of the many advantages for a non-computer science people like me is the fact that such systems include a lot of ready-made solutions and functions to common problems that ensures ease of use for learners like me. However, I do not have access to such software in my laboratory at the time of writing this document. And honestly I had left all hope for continuing any such work in the future in absence of proprietary systems. This was due to my own ignorance in many ways. At that point I did not know who Donald Knuth [11] or Richard Stallman [12] or Linus Torvalds [13] was and the massive impact that there work would have in developing this laboratory in the days to come. Based on my interactions with many and anecdotal evidence, there is a general lack of information or even a basic awareness on the open source movement among a large section of the University goers in this nation today. I myself am a prime example of that statistic. The idea that proprietary systems can offer things that is not possible with open source is a wrong notion. Although I agree that the range of functionality offered by software from multi-billion dollar corporations have an edge in terms of "in-built" functionality over the OSS systems (as mentioned earlier), yet with each passing day my belief in the OSS "community" is shattering this notion as well. It was at this point somewhere in 2018 that I happened to attend a short term course organized by NIT Durgapur [14] on Open Source Software. This turned out to be a watershed for me and my journey in developing this laboratory that soon followed. I learnt about the people mentioned above and many more along with the contributions they have made in creating and nurturing a philosophy that is the Open Source Movement. I think Richard Stallman summarized the core belief and idea for such initiative in the perfect way possible when he said: Think free as in free speech, not free beer" [15]. The very idea that software made by people, for the people and of the people trickled down from this belief itself. The Free Software Foundation reiterates this belief while saying, "Free software" means software that respects users' freedom and community. Roughly, it means that the users have the freedom to run, copy, distribute, study, change and improve the software. Thus, "free software" is a matter of liberty, not price. To understand the concept, you should think of "free" as in "free speech," not as in "free beer". We sometimes call it "libre software," borrow-

ing the French or Spanish word for "free" as in freedom, to show we do not mean the software is gratis" [16]. While attending this course, I learnt (read "learnt" as barely scratched the surface) about LaTeX, Python, Scipy and many more such systems. The idea that there exists this ocean of "free" systems and knowledge driven by a passionate and truly committed and universal "community" was incredible. I came back with ideas bubbling in my head on how to implement and use such systems to the fullest capabilities at the institute. Albeit, I must admit that it took some time to actually start using the software (as is for anything that is new for me), but then again I think it was a very personal experience and the learning curve is definitely going to vary with each individual. When I look back today, it seems that this was the point which changed everything really and seeded the idea for developing this laboratory.

With all the computational firepower at hand, the problem that still persisted was the fact I did not have any solution to the original problem in developing color profiles and using them to create a standardized print-workflow for our desktop inkjet printer. It was during one such afternoons of searching the web incessantly that I came across the ArgyllCMS [5] and much later on the LittleCMS [6] portals. Initially I was struck with disbelief at the thought that someone would actually take the trouble and put in so much effort and hardwork in creating such a critical system that I would consider to be a radical development in color image processing domain.

With this new and gargantuan ammunition in my quiver I set upon reading the innumerable number of scenarios in which Argyll can work. But then again, I was met with a challenge, you see, as I already mentioned earlier, I am a novice at understanding computer languages and their operations. And incidentally, the entire ArgyllCMS is command driven, which I know is not that hard for many but was Greek to me at the beginning. I think Graeme had thought of people just like me, who would like to use his system but are obtuse enough to not understand the command lines, and had linked some good Samaritans' work in developing GUI for the Argyll system. I immediately availed the opportunity to install LittleArgyllGUI [17] along with CoCa [18] and LProf [19] (which although is based on LittleCMS platform, on which I shall come to later).

With the software system in place, all I had to look into was the measurement devices. And frankly, almost none of the devices that were mentioned in the various use-cases for ArgyllCMS were available at my laboratory at that point. So the next obvious challenge for me was to go on a journey to either procure a spectrophotometer/colorimeter or find a work-around to it. Incidentally, a couple of years back I had bought an app called Colorimeter [20] and used it regularly again as a proof-of-concept on how a colorimeter would detect colors and provide their spectral characteristics as a result. Although a paper has also been published [21] using this app to detect changes in color, yet I had my inhibitions in using it for measurement purposes. My primary concern was that the app used the phone camera for measurement, which might lead to variations in result based on a number of parameters like illuminant, measurement distance, etc. Hence using it was not an option for managing colors in an already non-ideal situation where I was make doing most of the stuff required for an actual print standardization. It was again time to search for options. While going through the possible scenarios as listed in the Argyll portal, I came across what Graeme calls using a "poor man's colorimeter". It essentially means to create a custom ICC profile for

my scanner and scan test charts created by the Argyll `-targen` functionality. Some previous work had already been done to use an image scanner as a colorimeter here [22], along with some work on how it can also be used as a densitometer [23]. [1] And thankfully enough I did have a decent flatbed reflective type scanner at my disposal. So now all I had to do was to scan one of the many test charts as mentioned in the Argyll documentation in order to create a scanner profile, that can be used to convert colors so as to obtain their precise colorimetric measurements in spite of the absence of an actual colorimeter.

It seemed at this point that I was finally ready to take the full advantage of the open source systems presented to me. But then came upon the realization that I had reached a point which has no work-around. And that I would most certainly need a hard copy of the IT 8.7/2 or ColorChecker or any other such test chart for scanning and creating the scanner profile. It was then, that I thought of asking for help from some of the people I knew from my previous stint at DIC India Ltd. At one of the trial runs of an ink at Gujarat,I had met Das Damodaran [25] who is an experienced professional in the print standardization business. Of the many people I reached out to on LinkedIn to inquire if they had any of the mentioned test charts, he was the only one to revert. However, he had a different test target designed for printer calibration based on ECI 2002, which he kindly sent out to me. Unfortunately, due to size constraints and other factors, I could not use it for creating the scanner profile. So the search was on and I kept on looking for such a test chart. It went on for quite a few months but I could not come up with anything. I did eventually contact Wolf Faust [26] who provides IT test targets at affordable costs. Here too I failed to get results as I did not have a PayPal account. And my attempts to get the chart went on. Wolf also hosts the reference files in `.txt` files to all batches of the test targets available. I downloaded one such file for reflective copies and starred at it aimlessly, thinking about if there were other methods to obtain the test targets. Quite a few days later, when I was still thinking about whether I could work around the problem of not finding an IT test target, it suddenly hit me.

The original methodology required to create profiles for my scanner was to obtain an IT 8.7/2 or ColorChecker test target (original hardcopy printed on Kodak or Fujifilm photopaper), scan the target, import it into the LProf system and simply generate the profile by selecting the reference file based on the batch number as written on the target. To elaborate a little ICC Profile saves the reference or standard data which is supposed to be printed on the test target and the actual color values of the patches that are printed. These two datasets saved in matrices help to calculate how a printer will reproduce a given color. They basically provide a basis for some form of interpolation that can be done to obtain all other colors that do not exist in the mentioned matrices.

[1] In regards to the Colorimeter app, I shall implore readers of this document and my students to try out something. As Colorimeter app is premium costing about $1, hence you may also try out the free version of the same here [24]. You will have one IT 8.7/2 test target whose precise colorimetric evaluation (from instrumental measurements) is available in our laboratory. Try to measure the colorimetric values for those patches using the Colorimeter app and calculate the standard deviation and correlation coefficients for the two sets of data obtained from two different sources. It shall indeed be a very interesting study.

Now in absence of the original hardcopy test target how was I supposed to create the scanner profile! Especially when there are fixed reference data against each batch of the test targets. While thinking about other alternative methodologies and right on the verge of giving up on the project (once again!) for setting up the laboratory with almost nothing with me, I thought of something. Why not create my own IT 8.7/2 test target with any one the reference data given and use it as a basis on which to ensure that the colors for each patch in the test target remains "nearly" the same. I can't stress on "nearly" enough. So the idea was to create my own test target that would bear a passing resemblance with the original reference data (in my case, I chose R991201.txt as the basis such that patches in the reference that is supposed to appear "green", "blue", etc, appears "green", "blue" and so on in my test target as well.

Once the new test target was ready, I went to the local copier shop to get it printed. The electrophotographic color large-format printer with retro-fitted color profiler and management suite seemed to do a pretty terrible job at printing the target and rendered them almost useless with banding (one may read my old paper on detecting such banding appearances here [27]) appearing and quite evident in almost all the color patches in the test target [2]. With no luck with a state-of-art color managed digital printer, I turned to another copier shop with not so state-of-art infrastructure but a modest Epson L130 inkjet photoprinter and an unnamed photopaper brand which were ultimately used for printing my test target. And the result was "good" (if you press me, I can not at this time define the "good"ness of the printed reproduction, and really I don't need any metrics at this stage to qualify my printed test target to cross some quantifiable and allowable threshold. It was just that the print looked free from banding to a great extent and the colors seemed to not have any artefacts that would hinder my goal). Now that I had the printed test target, my next job would be to use colorimetric measurements from this to create a new reference file that I would use in the LProf system to create a profile for my scanner.

Having said, thought and done all this, my biggest challenge came next. You see, I believe that "one often meets his destiny on the road he took to avoid it" (in fact it was my first bio on Facebook when I started way back in 2007, which I have changed for better or worse now thinking that using profound quotes which at that time I did not fully understand was lame). My belief was reiterated by the fact that something that I avoided and dreaded so much so that I had worked around it all the while, had come back and how! The astute readers of this manual might have already figured out what I am referring to quite a while back, and might be smiling at my poor luck at this time! So let me elaborate where the colorimeter that I am trying to avoid to measure colors with and creating my own version of "poor man's colorimeter", check-mated me. I had printed a test target with the hope that I would create my own set of IT target with new references obtained from my chart. But how do I get the new reference then? For this there is no work-around and I **NEED** a spectrophotometer to measure the tristimulus values for each of the 264 patches. Once I have this data, I can make my new_reference.txt file to be used for creating the scanner profile in LProf.

[2] Begs the question as to why such costly instruments led to poor reproduction quality! The inquisitive amongst the readers of this manual may ponder and realize that this is exactly why I am trying to implement a color managed print production laboratory in the first place.

At around the same time, when I was speaking with one of my colleagues who happens to be a Lecturer in the Department of Photogragphy, Pallab Roy and sharing my frustrations while creating this laboratory, something interesting came up. I was casually referring to the fact that I have no hardcopy test targets like IT 8.7/2, Macbeth ColorChecker and so on, when he came to the rescue, saying that he did have a 24-patch Macbeth ColorChecker. I was ecstatic thinking that now I do not need the spectrophotometer I dreaded searching for. But as luck would have it, LProf does not support any other test target apart from IT 8.7, so I turned to CoCa, which does support almost all the test targets Argyll is capable of handling but my letter size scanner isn't designed to "scan" the ColorChecker (which the more perceptive of the readers already know is meant for calibrating cameras and not scanners, yet principally they represent the same idea i.e., they capture images). My journey continued while I kept on searching for someone who would let me use their spectrophotometer for measuring my test patches.

With all that has gone wrong till this time, I thought to myself may be there is a place where I can get this measurement thing done without any further poor luck. I wasn't wrong this time. I visited my alma mater, Department of Printing Engineering at Jadavpur University and met with my long time project mentors Prof Kanai Chandra Pal [28] and Prof Arun Kiran Pal [29], to whom I presented my entire ordeal and asked for help. They very graciously granted me permission for measuring the the tristimulus values for all the 246 patches in my test target using their X-Rite SpectroEye spectrophotometer. It would be amiss if I don't acknowledge the contribution of my long time friend and now colleague as well as my Head of Department at the institute Krishnendu Halder [30], who helped me in gathering and collating the data. We went back the next day too to measure the L*a*b* values as well. The readers might say, why the hassle! Just transform your tristimulus values to L*a*b* space or L*u*v* space. Yes, I would have done so, yet I went the next day to do something which might seem futile or redundant, but at the time it was just for good measure. Maybe I was just worried with all that went wrong till now I wasn't confident enough that my transformation code would correctly address the issue of color space transformation. There was a practical problem in the process as well. As I had mentioned before that the "good"ness of the test target though was apparently evident from visualizing the test patches, yet while measuring we realized that the we could not really rely on the apparent "good"ness of the print wherein we could observe a ΔE variation within the same patch (from one area to the other) for a substantial number of color patches of around $0.1 - 0.2$. This was another reason why we went the next day to measure the L*a*b* values. I knew we couldn't blindly trust the tristimulus data and the standard deviation in the measurement was on the higher side. So apart from good measure, I made a conscious decision for taking another set of colorimetric measurements which will give me the advantage of choosing the data that minimizes ΔE variations while creating the scanner profile. This would also help in creating a more acceptable reference data that might enhance the quality of the "poor man's colorimeter".

With the final piece of the puzzle sorted out, it was now time to start the actual work for implementing a color managed desktop printing workflow. The gathered data will be now used for making our new_reference.txt file. This reference file will be used

in LProf for creating the scanner profile, that would in turn be saved in the working folder for Argyll to be used in the later sections when the we would need the scanner to be used as a colorimter.

To summarize our work to manage colors will have the following components:

- A scanner that will also work as a colorimeter. The idea is any color scanned using the scanner profile will bear a considerable accuracy to what might be its actual colorimeteric value.
- With the "poor man's colorimeter" ready to operate, the next task in hand will be to create a profile for our desktop inkjet printer. This will be done by first, printing a test target generated in LittleArgyll GUI, followed by printing it using our printer in question and finally measuring the colorimetric values of the patches using our colorimeter. Actually, I don't have to measure each of the patches indvidually, as Argyll takes care of that by taking the entire test target as an image input using the `scanin` tool. This is then followed by creating the printer profile using the `colprof` tool.
- Once the printer profile is created, Argyll also offers the functionality to calibrate the profiled printer to tweak the appearance as close to the original as possible.

The following sections will deal in the technical discussions on how to use the individual systems in order to manage colors and create visual parity between originals and their reproduction using our desktop printer. However, it should be noted here that as on date of writing this manual, I do not have a measurement instrument for monitor calibration [3]. In absence of this it is not possible for accurately calibrating our laboratory monitor before we go for printing. Hence the quality of our soft-proofs may not be reliable enough. The comparison of our reproduction will only be possible when we view our print against the soft copy of the original using the scanner profile as the monitor profile or by some other means which I has still haven't figured out yet. But like every misstep noted earlier, this too can be overcome. From the point when it was a dormant idea in my mind till today, both I and my laboratory have come a long way and better still we have a longer journey ahead, which is full of newer problems and revelations that is likely to teach me and in turn my students a lot.

3 Pre-Requisites

I would like to state that this laboratory is a part of the final year elective course. Hence this is not meant for those selecting machine production as their elective. So I have assumed that students with a strong interest in the pre-press focus area will be choosing this as their elective, and as such they should have a fair bit of priori knowledge on the following domains:

- Digital image reproduction methodologies for print viz., halftoning, dithering techniques
- Fundamental concepts in pre-press operations

[3] I do have the LProf's approximated version of instrument-less display calibration method, which I shall employ to claibrate the display.

- Pre-flighting operations
- Print production workflows
- Page Description Languages
- Colorimetry and color spaces
- Densitometry and Spectrophotmetry
- Color management: Device dependent and Device independent models
- Basic numerical and scientific computation using Python

I strongly recommend using the various cited documents, websites, etc in this manual to understand the working procedures of various systems and software at use in this laboratory before you start doing the experiments.

4 Using the reflective scanner as densitometer [23]

The idea of using my scanner as an instrument for measuring optical densitometer first came when I stumbled upon [**negfix**] site. I could not find the name of the author for the site, the only information I have is his username: JaZ99wro who is a professional photographer. The basic premise is to scan the image in question and obtain the ratio for the amount of light that is incident on the surface and the amount that is reflective off of it. The detailed procedure can be summarized below.

The relation to calculate optical density is :

$$D = log_{10}(I_0/I)$$

Here, I_0 represents the total amount (or intensity) that is incident on the surface of the substrate. And I represents that which is reflected from the surface. But it is to be noted that the scanner must work in linear BW mode (gamma=1.00) without any modifications to the data. The measurement procedure is as follows:

- Scan your film exactly as described above, save the output to the tiff file (one should choose file formats that support lossless compression, this is going to be a recurring theme).
- Open this file in any image editor, then check the mean RGB value of the measured area. You may use Gimp (download from here [31]) or ImageJ (download from here [32]) This is your I value.
- Calculate the density. If you are working with 16-bit values, the I_0 is 65535, in 8-bit mode it is 255. Working in 16-bit mode is highly recommended.

The reader may also refer to the wonderful website and the resources developed by Bruce Lindbloom here [33]. Once the calculations are done, there is a neat way to check if your values hold true. To check it out, you need to refer to our IT 8.7/2 test target and our new reference file. You can convert L* value to the density, using the relation

$$D = log(1/\beta)$$

You might refer Prof Roy S Berns book [34] to get a fair amount of idea on the background of this relation. The previously refered formula with intensity of light is

analogous to the one mentioned in this para. The β referred to here represents the luminance or luminance factor that is again similar to transmittance factor. The mathematics in this case can be a bit confusing due to multiple terminologies, yet if you think intuitively, I hope you will be able to see how this relation can be used to our advantage. Now to give you a brief background, let me refer to one of the cornerstone and foundational books in Colorimetry [35]. I will summarize the formulae here,

$$L^* = 116f(Y/Y_n) - 16$$
$$If((L^* + 16)/116)^3 > 0.008856, \beta = log_{10}(1/((L^* + 16)/116)^3$$
$$else \quad \beta = log_{10}(903.292/L^*)$$

Once you have calculated the luminance factor, compute the optical density. This optical density that you have obtained is what you should have obtained under ideal conditions and measurement devices. However, earlier you have calculated the optical density for the same patch using the intensity of light. The difference between the two may then be calculated. I have also provided some images and data as was obtained by [23] below.

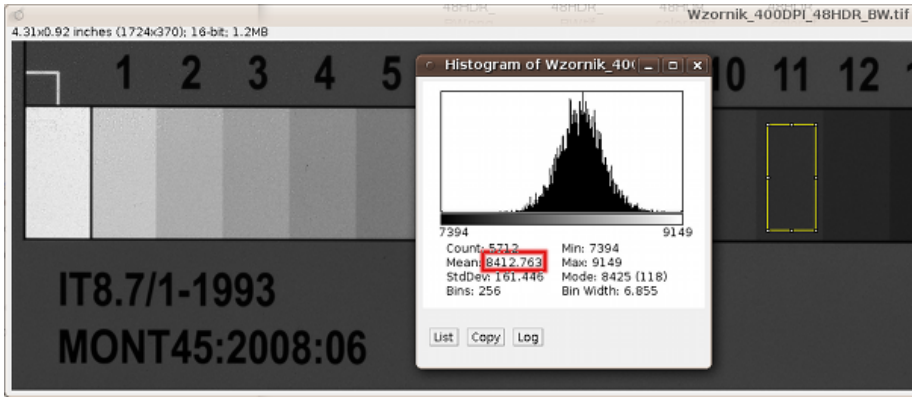

Fig. 1. Finding the mean RGB count

So now we have a device that can help in analyzing whether the measured L* value obtained from the ”poor man’s colorimeter” is equal to the calculated value. By analyzing this, you can understand the percentage error that your ”colorimeter” has while measuring color patches.

5 Creating the scanner profile

5.1 Preparing the IT 8.7/2 scanner calibration test target

As already mentioned earlier, that I have made my own test target for profiling the scanner. This is not a very difficult job. All you need is an object based image editing

Density testing

Patch name	L* value	Patch density	RGB value	Measured density	Error	%Error
Dmin	86.65	0.159	41912	0.193	0.034	21.29%
GS01	80.47	0.240	36317	0.255	0.015	6.35%
GS02	75.64	0.307	31744	0.314	0.007	2.20%
GS03	71.83	0.362	28496	0.361	-0.002	-0.46%
GS04	67.72	0.425	25212	0.414	-0.011	-2.58%
GS05	63.74	0.488	21994	0.473	-0.015	-3.09%
GS06	59.37	0.562	18719	0.543	-0.019	-3.29%
GS07	55.16	0.637	15675	0.620	-0.016	-2.56%
GS08	51.58	0.704	13417	0.688	-0.016	-2.28%
GS09	48.37	0.767	11707	0.747	-0.020	-2.63%
GS10	44.88	0.840	9919	0.819	-0.021	-2.48%
GS11	41.6	0.912	8413	0.891	-0.022	-2.36%
GS12	38.15	0.993	6934	0.975	-0.018	-1.81%
GS13	34.62	1.080	5681	1.061	-0.019	-1.78%
GS14	30.66	1.187	4437	1.168	-0.018	-1.52%
GS15	26.56	1.306	3368	1.288	-0.018	-1.39%
GS16	22.71	1.430	2518	1.414	-0.015	-1.08%
GS17	18.6	1.576	1805	1.559	-0.017	-1.08%
GS18	14.71	1.732	1261	1.715	-0.017	-0.96%
GS19	10.58	1.920	826	1.899	-0.021	-1.10%
GS20	5.56	2.211	474	2.140	-0.071	-3.21%
GS21	3.13	2.460	305	2.331	-0.129	-5.24%
GS22	1.76	2.710	231	2.452	-0.258	-9.53%
Dmax	1.1	2.914	195	2.526	-0.389	-13.34%

Fig. 2. Calculated and measured density values

tool. You may use InDesign or Inkscape (which may be downloaded from here [36]), depending on what you have in your system. However, I strongly recommend using Inkscape. As you are not working with the standard 5" X 7" reflective copy of the target, hence you need not worry about the size of the target. I will suggest though you be mindful of the following considerations while designing your target, consider the size of the scanner that you have in the laboratory, because it will determine the maximum size of the pages that you can feed as input to the scanner. And secondly, also ensure that the size of each patch be sufficient enough to allow for the spectrophotometric or colorimteric reading, meaning that the head of the instrument should fit exactly inside the patch without taking any other color in the patche's vicinity. If you are using InDesign however, you will have some form of ease while choosing your color for each patch, because InDesign lets you modify the L*a*b* values of the patches. So plan carefully before you start designing. Use the grid option, reduce the borders of each rectangular patch to essentially zero. I think I do not need to deliberate on how to design the test target in any greater detail as you have a fair bit of idea on designing from courses like Typesetting & Composition and Print Design. Once the design is complete, simply export your file into eps or pdf formats, but ensure at the same time that InDesign does not force any color management into your file. And there you have your test target ready to be printed.

But its a bit tricky if you are using Inkscape. It allow you to use either HSL or RGB or CMYK format. I think the best option in this case would be to choose the RGB values. Now be careful here, think about what you are about to do. Refer to Bruce's documents here [33]. You need to use the conversion matrix (inverse matrix) as you need to convert from XYZ space to RGB space. The XYZ data is mentioned in the

reference file R991201.txt. Use it as a starting point. You would also need some other information, which RGB space is your documents currently in? Is it sRGB, AdobeRGB or something else. The inverse matrix will change depending on it. The reference illuminant will also have to considered while you do the transformation. So while you are likely to get the transformation matrices from Bruce's portal or any other textbook, but you need to consider that it is also essential to consider for chromatic adaptation. In order to convert the tristimulus values from source to destination on varying the illuminant,

$$\begin{bmatrix} X_D \\ Y_D \\ Z_D \end{bmatrix} = \begin{bmatrix} M \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \end{bmatrix} \tag{1}$$

You may use Bruce's online calculator for selecting the transformation metrix. The reference file was prepared with D50 as the illuminant. For instance, if you need to calculate the sRGB values from the tristiumulus colors, then you would first need to convert the tristimulus value from your reference file into the new tristiumulus values with D65 as the illuminant and only then select the XYZ to RGB inverse transformation matrix given in the site. This has to be followed by the inverse transformation from XYZ to RGB transformation. This can be done by

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} M \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2}$$

I suggest that you write a small function in PyCharm to implement this procedure.

The colour-science library does eliminate the need for all this though. The function $colour.XYZ\_to\_RGB$ can be used directly convert from the former to later and you can mention the illuminant required for the transformation. $colour.XYZ\_to\_RGB$ $(XYZ, illuminant\_XYZ, illuminant\_RGB, XYZ\_to\_RGB\_$ $matrix, chromatic\_adaptation\_transform =' CAT02', cctf\_encoding = None, **$ $kwargs)$

For example:

Albeit this might seem to be a redundant process, especially when the process would be much more easier and apparently efficient if doing in InDesign. But the purpose of this exercise is not to make you do an elaborate procedure just for the sake of it. However, I would like to reiterate that by doing this you would get a fairly good idea on color space transformation and it will also help you in familiarizing with the colour-science package and the various concepts like linear transformation, chromatic adaptation, and much more.

After getting this target printed, you need to measure the the colorimetric values for each of the 246 patches using a spectrophotometer (which at the time of writting this I don't have, hence a practical method will be to use the newreference.text file that I have made using the spectrophotometric measurements of the patches from my test target). Once you have theses measurements (both XYZ and L*a*b* values) prepare your reference file that has to be fed into the LProf system. Select the necessary

```
>>> XYZ = np.array([0.21638819, 0.12570000, 0.03847493])
>>> illuminant_XYZ = np.array([0.34570, 0.35850])
>>> illuminant_RGB = np.array([0.31270, 0.32900])
>>> chromatic_adaptation_transform = 'Bradford'
>>> XYZ_to_RGB_matrix = np.array(
...     [[3.24062548, -1.53720797, -0.49862860],
...      [-0.96893071, 1.87575606, 0.04151752],
...      [0.05571012, -0.20402105, 1.05699594]]
... )
>>> XYZ_to_RGB(XYZ, illuminant_XYZ, illuminant_RGB, XYZ_to_RGB_matrix,
...            chromatic_adaptation_transform)
array([ 0.4559557...,  0.0303970...,  0.0408724...])
```

Fig. 3. colour.XYZ_to_RGB

pre-conditions in the system and create the scanner profile (use .icm as the file extension to the profile you have created). You should also use the in-built profile inspector offered by LProf to evaluate the quality of your profile. Observe the ΔE value and RMS errors to assess the ”good”ness of the profile created. Take necessary actions to improve the same. For instance, select the correct Gamma, input and output profiles for your reference data, and so on.

Now you have the scanner.icm profile in the C:\Windows\system32 \spool \drivers\color folder in case you are using Windows OS. Copy it and paste it in the working folder of your LittleArgyll GUI, so that Argyll has access to it while creating the printer profile.

Documentation from LProf [37] Camera and scanner profiles are built by using a thing known as ”IT8 target”. IT8 targets are just a photo with a set of color patches. There are known colorimetric measurements for each of these patches. There is also a standard way to specify these measurements, the IT8/CGATS file format. Each IT8 target comes with a IT8/CGATS file that contains the colormetric measurements for that target.

IT8/CGATS is a way to encode data that is both, human and machine readable. You can open any IT8 reference file with a text editor to see it. You can even modify the contents. Since it is a standard, most color management systems will accept it including LPROF. A IT8 reference file contains measurements for a set of color patches. The measurement of each patch can be the device colorant values (the RGB or CMYK in 0..255 range) the tristimulous values (XYZ or Lab) and some statistic data.

Step 1: Adjust controls of scanner First you should be aware that getting a good capture of the IT8.7 image is 80% to 90% of getting a good profile.

Bit depth: You will need 24 bits per pixel at least but using 48 bit images is best.

Gamma: On most scanners you can select the gamma to be used for scanning the image. In general you should use a gamma between 2.2 and 3.0. A Gamma 2.2 has the additional benefit of being close to the sRGB gamma, and this means the uncorrected Image will ”look nice” on an ”average” monitor. It is also near to perceptual gamma.

Gamma 2.4 has the additional benefit of being closest to perceptual space, and this is a very good reason to use this value. Less that 2.2 (and of course the infamous 1.0) can generate huge loss of detail in shadows, only to give a slight bettering of highlights. Don't use this unless your are using 16 bits per sample, and even in such case, don't do it unless you know what are you doing! Gammas around 2.4 are best for flat bed scanners and film scanners with limited dynamic range. With high dynamic range film scanners values closer to 3.0 may be best. Hutch Color, for example, recommends a gamma of 2.8 for high dynamic range scanners. But for flat bed scanners more than 2.4 (up to 3.0) looses some highlight detail with no gains in shadow detail.

Turn off all color management, color enhancing and tweaking on scanner driver. Set brightness contrast and hue (if available) to default values. Make sure that all controls are locked down and that the scanner software is not doing any automatic adjustments. These setting are supposed be your working scanning mode, so set all of them to reasonable defaults.

IMPORTANT: The profile will only be valid with the settings used to capture the IT8.7 target!

Step 2: Scan the IT8 target Store it in PNG, TIFF, BMP on depending of supported formats of your platform. DON'T use JPEG! Its lousy nature makes it unsuitable for this purpose. You could use JPEG on final images, but not in the profiling process.

Step 3: Extract the RGB values for each patch Without help, this could be a huge task. There are over 200 patches in a typical IT8.7 target (some have as many as 288) and taking each sample by hand would be tedious at best. With the help of the LPROF this is an easy task. Before we can proceed the reference file for the target must be installed. Please see Install Reference File Dialog for details. Start LPROF and on the Camera/Scanner Profiler tab press the "Load Image" button. Select the image of target you scanned in step 2.

I will use the scandmo.png image located in the "data/pics" directory as a sample. This is a Kodak film target, scanned with a scanner that adds a huge magenta cast. Don't worry, the profile will turn this one into a nice looking image and the magenta cast will help demonstrate how effective profiles generated with LPROF are. The reference file for this IT8.7 target (e3199608.TXT) is also located in the "data/pics" directory. Since this is a Kodak target in the Install Reference File Dialog select the "IT8.7 19+3 (Kodak) column picker" template when installing the reference file. To "measure" the RGB values of the target image you using the mouse to place the pick template over the patches. You can do this by clicking on each corner of the of the target image. As each corner is selected with the mouse you will see a mark placed on the corner.

When all four corners are marked the picker template will appear superimposed on the target image. Assure all green rectangles are inside the patches. Failure to do so will result in a wrong profile! You can adjust the locations of the corner marks to fine tune the location of the picker template. You can also adjust the relative size of "hot areas" by setting the % in "safe frame" on "Preferences" tab. A smaller safe value results in a larger "hot" zone. Normally there is no need to touch anything. Once you were happy with the location of the patch grabber zones you now have the correct reference sheet of the target, which were given by Kodak, and the RGB values obtained by our scanner. Now we can compare these to create a profile.

Step 4: Create the Profile From this point on it is quite easy. First we have to specify the profile filename. Press the "..." button placed on right of "output profile file", to select the location where the profile will be saved. On Linux/Unix systems this will default to $HOME/.color/icc. On Windows systems this will default to the Windows profile directory (on most systems c:32. But you can override this. The file name for the new profile could be "scandmo.icm". You can now fill some info to help locate the profile latter. Press the "Profile Identification" button, and fill in the fields. You are now ready to generate the profile. Press the "Create Profile" button to launch the profiler process. After a few seconds, you will have a small profile for your scanner. You can inspect the profile in some detail by pressing the Profile Checker button. If you do you will see the the demo profile has the a dE (amount of error) that is about 1.24, which is reasonable. Now, we can reduce the error level, but it will take more time to generate the profile. To do this press the "Parameters Parameters" button, and select on "Resolution" group "33 points". Then press the check box labeled "local convergence analysis". Press the "OK" button to save these settings and then the "Create Profile" button. Now the profile generation process will take a while, but the obtained profile is much improved... average dE is about 0.7 and the peak value is as small as 3!!! Again you can use the "Profile Checker" to inspect your profile. In the Profile Checker have a look at the Shaper TRC curve in the curves tab. Notice that the green curve is significantly different from the red and blue curves. Since green is the complement of magenta this is exactly what you would expect for a device that was producing images with a magenta cast.

Step 5: Proofing the Profile Let's check how this profile affects the original Image. In the "Preferences" tab set the "input profile" to the newly created scanner profile, and "monitor profile" should be set to either a custom profile that was created using the monitor profiler (see next section) or a generic profile such as sRGB. In the "Preferences" tab check the "color manage display" check box and then go to the "Camera/Scanner Profile" tab to see the results. The girl's Image is dark, this is normal. However, the rest of patches are hugely modified, the gray scale should appear smooth and without any cast, as well as all magenta tint should disappear.

Notes on Profile Parameters Resolution (CLUT points) This set of radio buttons are used to select how many point will be included in the profiles Color Look Up Table (CLUT). Select smaller values to make small but less accurate profiles. Select larger values to create larger more accurate profiles.

Profile Verbosity These radio button allow you to control how much additional information is stored in the profile. Selecting "Only required tags" will store only information that is absolutely needed in the profile. This makes for smaller profiles. "Store additional tags" stores additional information but does not store measurement information. "Verbose, store anything" will include everything that can be included in the profile including measurement information that is useful to the Profile Checker.

Extrapolation Checking this will cause the use of the local convergence analysis algorithm when creating camera or scanner profiles. This is more accurate but also takes longer to process. On fast modern hardware using this with 33 CLUT points it

will take about a half minute to create the profile but it will result in the most accurate possible profile.

Cromatic adaption viewing condition The default setting is Linear Bradford which is the standard setting recommended by the ICC. Use CIECAM97s to have more control over your profiles. This is particularly useful for monitor profiles since it allows you to compensate for viewing conditions.

Evaluating scanner profile quality using Profile Checker Info Tab The info tab contains basic information about the profile. This includes:

- Identification information entered in the ”Profile Identification” dialog
- The profile white point
- The profile media white point
- The profile primaries
- The profile red, green and blue gamma
- The target reference file
- The measurement sheet file

CIE Diagram Tab This tab contains a CIE or chromaticity diagram. A CIE diagram is a representation of all of the colors that a person with normal vision can see. This is represented by the colored sail shaped area. In addition you will see a triangle that is superimposed on the diagram outlined in white. This triangle represents that outer boundries of the color space of the device that is characterized by the profile being inspected. This is called the device gamut.

In addition there are black dots and yellow lines on the diagram. Each black dot represents one of the measurement points that was used to create this profile. The yellow line represnsents the amount that each point is corrected by the profile and the direction of the correction.

dE Report Tab dE is short for delta E which is a standard way to measure color errors. On ”dE report tab” you will see the error statistics. (unless you forgot to set the profile verbosity to ”store anything”). This report shows how close the the color will be once the profile is applies to the image. A delta E of 1.0 is right at the treshold of human perseption. That is under perfect lighting conditions when comparing two color patches that are different by 1.0 delta E just over half of those looking that those patches side by side will correctly identify them as not being the exact same color. Think of it as the colormeteric equilivalant of a decibel. It is fairly typical to get average delta E of less than 1.0 using profiles created with LPROF.

Curves Tab The curves tab shows the correction curves applied to each color channel by the profile. When the device being profiled has good color balance the reb, green and blue curves will lay on top of each other. But if the color balance of the device is off significantly the curves will seperate from each other. Also as the gamma of the devices response changes that amount of curvature in these correction curves will change.

Preferences Monitor Profile Using this combo box you can select what monitor profile will be used by LPROF for proofing images and also to color manage the display

in the Profile Checker dialog. Using the button labeled "..." you can select the directory that contains your monitor and other profiles. The combo box drop down will list all profiles in that directory that are tagged as display profiles. Note that it uses tag data from the profile and does NOT list the file names of the profile.

Input Profile To select the input profile that will be used for proofing use this part of the dialog. In most cases this will be used to check profiles you are creating for devices like cameras and scanners. It operates in the same way as the item above but does not limit itself to only display class profiles.

Color Manage Display This is used to turn soft-proofing on and off. To proof a scanner or camera profile select the profile for that device in the input profile item and check this item. When you go back to the Camera/Scanner Profiler tab the target image will be color managed using a transform from the input profile to the monitor profile selected in this dialog.

Install Reference File Press the button to open the "Install Reference File" dialog. This dialog is used to install a reference file.

Safe Frame This is used to set the percentage of each color patch on the target will NOT be used by the picker template. The smaller this number is the larger the "hot" zones for each patch will be. A larger "hot" zone will sample a larger area from each patch but will also make it harder to prevent the "hot" from spilling over into a near by patch. The default values is 65% but values between 50% and 75% can be used. In most cases the default value is best.

Output Colorspace This is used to control what kind of values will be output when an IT8.7 target is sampled. In most cases you will want to leave that set to the default value which is RGB - Pick from image.

Profile Identification Model Used to identify the device and device settings that this profile is for. As an example for a camera profile for a Nikon D70 camera the value in this field might be something like "Nikon D70 - direct sun" This field is commonly used by other software for profile identification. It is highly recommended that this be set to a user meaningful value.

Manufacturer Used to identify the manufacturer of the profile. This field is also used by other software to display information about the profile. You should set this to some value that identifies the profile as yours.

Copyright and Comments Place copyright information in the copyright field and any comments you want to include in the profile in the comments field.

Monitor Tab Create a Monitor Profile LPROF can work in two different modes to generate a monitor profile. The "advanced" mode, that gives accurate profiles by using hardware devices like X-Rite DP92 or EyeOne, and the "simple" mode, that allow to you build reasonably accurate profiles without any additional hardware.

Building a monitor profile: a coarse approximation Let's build our first monitor profile to check how all this stuff works. We don't need great accuracy at this point. We want to calibrate coarsely, just to visually check if all is going OK. First start LPROF. Since we need only a coarse profile on the "Monitor Profiler" tab select the radio button labeled: "I want to build a coarse profile, giving approximate values". This

is selected by default. We do not need any hardware measurements at this point only a few visual adjustments. Press the button labeled: ”Enter monitor values”. You will see the monitor values dialog.

Step 1: Set contrast In order to obtain maximum performance, we need to adjust the physical monitor controls. Set contrast to near maximum, (yes, near maximum. This is the gain of monitor and we need as much gain as monitor can deliver).

Step 2: Set Gamma and Black Point Now select the ”Set Gamma and Black Point” button. This dialog will help you to set both the gamma and black point for your monitor.

Step 4: Set White Point For information on setting monitor White Point see Rough Monitor Values.

Step 5: Set Primaries Next adjustment is setting the primaries. For information on setting Monitor Primaries values see Rough Monitor Values.

Setting Monitor White Point Some monitors refer to white point as ”temperature”. Most monitors come from the factory with the white point set to D93 in order to get more brightness, unfortunately this restricts the gamut and adds a large quantity of blue, so my advice would be to select something lower. If you can switch temperature of your monitor, D65 (near 6500K) could be a good choice for multipurpose use. For proofing devices, D50 is almost a must, but it adds too much yellow for a unadapted use such as using non-color aware software. D50 requires a dim or dark room to give the best results. If you don’t know which temperature your monitor has, select D65 or D93 but most modern monitors allow users to set the color temperature so try to find where this setting can be changed and set it to the best value you can locate. On our fist approach both D65 and D93 will give reasonable results. If your monitor has any other white point, you can select ”User defined”, the last option. You can then select your particular white point as a temperature in Kelvin degrees. Don’t use CIE illuminants. They are intended for workspace construction.

Setting Primaries For the primaries, select the default ITU/R BT.709 if you are profiling a CRT monitor. There are primaries for Samsung TFT and LCD. The other primaries are for building workspaces. If you don’t know the primaries, don’t worry. Use the default ITU/R BT.709. There are 6 or 7 CRT tube manufacturers in the world, and all them does use this set.

Step 6: Make Final Settings and Create Profile Now you can use the ”Profile Identification” dialog to fill in the info about your monitor. Although this is not strictly required, it could prevent a lot of confusion when you have more than a couple of profiles. Now we are ready to generate our profile. We must specify the output filename just like you did when creating a scanner/camera profile. For the name, we could use ”CoarseMonitor.icm” So, type in the name and then proceed to generate the profile by pressing the ”Create Profile” button. If all is OK, the status console will show ”Profile DONE!” Let’s now check how well the profile is working by eye. Do this is basically the same way you proofed your scanner profile above.

Building a Monitor Profile: Using a Hardware Measurement Device To measure your monitor, you need to create a IT8 sheet containing XYZ measurements for several RGB combinations for your monitor. A sample sheet of this kind can be found in ”data/pics” directory. The file is called ”sRGB.IT8” and it measures an ideal sRGB

monitor. Use this file as a template for creating your own monitor measurement sheet. You need to change the XYZ values to those read by your measurement device. Don't put any Lab value, just RGB and XYZ. Please keep the RGB values the sRGB.IT8 has. Optionally, you can use more patches to assure consistent color reproduction, but make sure to include at least the ones in sRGB.IT8. In the "data/pics" directory you will also find "monitor patches.tiff". This file contains color patches that correspond to the values used in sRGB.IT8 and it can be used to measure the RGB values from your monitor. Note that a large number of gray patches are needed. You can also add as many as you wish, with more gray patches, you will get better linearization control curves. You must specify at least 16 gray patches.

Now we will try this procedure, if you do not have a measurement hardware device you can use sRGB.IT8. We are going to generate a sRGB profile, by using the sRGB.IT8 measurement sheet. Go to the "Monitor Profiler" tab. Now, we are going to use it in advanced mode. Select the radio button labeled: "I want to build accurate profile from measurement sheet". Then press the "..." button located at right of the edit box labeled "Measurement Sheet". Select in "data/pics" directory the file called "sRGB.IT8". In the "Output profile file", select a new file name for your profile. Call it "my_sRGB.icm". I will use this sample to present another utility, the profile checker. Since the profile checker can understand advanced tags, we want to include all information in the profile, so select the "Profile Parameters" button, and make sure to select "Verbose, store anything" in the "profile verbosity level" group. Now use the "Profile Indetification" button and fill the info fields if you want (again, this is recommended) and press then press the "Create Profile" button. A new "my_srgb.icm" profile will be created. We will check now how accurate this profile is. Start the profile checker dialog selecting the "Profile Checker" button. The profile checker is a tool for inspecting some of profiles internals. In our case, we want to check the accuracy.

Setting Gamma This is by far the most important setting when making a rough monitor profile. You can link the red, green and blue channels together and adjust them all to the same value or they can be unlinked and individually adjusted.

On the dialog are two charts. One is small and the other is large. These charts will assist you with setting both the gamma and the monitor black point. These charts are both used with permition from Norman Koren. Please see the Norman Koren Monitor Calibration Web Page for more details on these charts.

Brief note on Gamma from [38]
Gamma describes the nonlinear relationship between the pixel levels in your computer and the luminance of your monitor (the light energy it emits) or the reflectance of your prints. The equation is,

$$Brightness = C * DigitalValue^{\Gamma} + BlackLevel$$

C is set by the monitor Contrast control. Value is the pixel level normalized to a maximum of 1. For an 8 bit monitor with pixel levels 0 - 255, value = (pixel level)/255. Black level is set by the (misnamed) monitor Brightness control. The relationship is linear if

gamma = 1. The chart on the right illustrates the relationship for gamma = 1, 1.5, 1.8 and 2.2 with C = 1 and black level = 0.

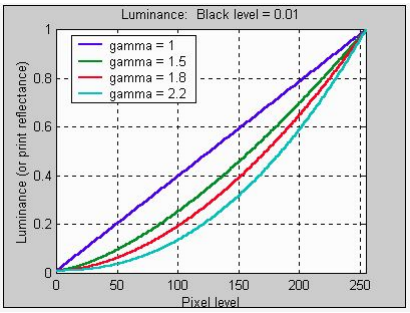

Fig. 4. Gamma

Gamma affects middle tones; it has no effect on black or white. If gamma is set too high, middle tones appear too dark. Conversely, if it's set too low, middle tones appear too light. [Note: Film is different. Gamma increases with development time; highlights are strongly affected.] Gamma, as defined above, is also called display gamma– the product of monitor's native gamma and video card lookup table (LUT) gamma. (Most video cards have LUTs.) As we shall see, it is closely related to film gamma, which is the average slope of the film response curve.

Black level is the monitor luminance or print reflectance for value = pixel level = 0; i.e., it is the deepest black in the monitor or print. It is a constant that includes the effects of viewing flare (stray light). In good monitor viewing environments it can be very small, less than 0.01, relative to a normalized maximum Luminance of 1. It's also around 0.01 for high quality prints (higher for mediocre paper/ink combinations). Sometimes black level appears inside the exponent, but it makes little difference since it's a constant[4].

The small chart is a fixed gamma 2.2 only chart. The large chart will directly read out the gamma on a sliding scale. When the small chart is a consistant color all the way across the gamma sliders have been correctly adjusted. The large chart will show

[4] The simplified, ideal equation for the sRGB color space (the standard color space of Windows and the Web; gamma = 2.2) is $y = x^{2.2}$, where y is the luminance and x is the normalized pixel level. But the correct equation from the sRGB standard is $y = x/12.92$ for $x <= 0.03928$; $y = ((0.055 + x)/1.055)2.4$ otherwise. The curves for the two equations are very close, as illustrated here. The correct curve is linear below x = 0.03928, y = 0.003035. This corresponds to a pixel level of 10 for images with a bit depth of 8, where the maximum pixel level is 255. For the ideal curve at pixel level = 10, y = 0.0008 (much lower). The difference is even more striking at pixel level = 5 (x = 0.0196): y = 0.00152 for the correct equation; y = 0.000175 for the ideal curve. This has consequences for setting the black level. [Obscure note: the two links above give different values of x for the boundary between the linear and exponential curves: 0.03928 and 0.04045. Annoying, but the equations are nearly identical because the slopes of the two curves are very close around x = 0.04.]

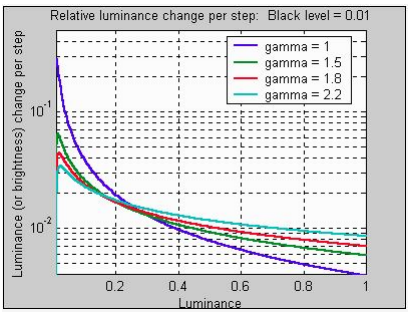

Fig. 5. Gamma

a band that has a consistant color that corresponds to the currently set gamma. In addition, the black level segment of the large chart will add in setting monitor black point. These charts allow users to make very accurate and repeatable gamma adjustments.

For best results make these adjustments in a darkened room and maximize the gamma dialog. Start out with the color channels linked and adjust the slider until the small chart is all one shade of gray. Then adjust your monitors ”brightness” control (this is really the black point control) until the just visible bars in the black level part of the large chart are visible across from the point marked 2.2 on the large chart. Depending on how much you adjusted the black level you will likely have to go back and adjust the gamma slider again to get the gamma to be 2.2 again. You may have to repeat this process several times to get both black point and gamma correct. At that point you can unlink the color channels and adjust the gamma of each color channel. On many monitors this may not be necessary. You should obtain a typical value of 2.4-2.6. The profiler will give you a profile that corrects this to 2.2 which is the recommended gamma for monitors.

Why Gamma? The eye doesn’t respond linearly to light; it responds to relative brightness or luminance differences. The smallest luminance difference the eye can distinguish in bright light (Delta L) is expressed by the Weber-Fechner law,

$$\Delta L/L = 0.01$$

Most video cards display 8 bits per color (256 levels), even if you store and edit in 16 bits per color channel. With gamma = 1, the relative luminance difference at the highest luminance levels would be less than 0.004 (1/256)– much less than the eye can distinguish, but it increases rapidly for lower levels. In dark areas it can be large enough to cause perceptible banding between levels. This can be corrected by applying a gamma curve, as illustrated in the graph on the right, which shows the relative luminance difference between pixel levels for gamma = 1, 1.5, 1.8, and 2.2. The relative difference is most consistent for gamma = 2.2. It remains under 0.01 at high brightness levels, but it is lower than gamma = 1 for luminances under 0.2. Relative differences are not displayed uniformly when luminance is plotted on a linear scale, but they are on logarithmic scales: relative differences such as doubling or halving the luminance

(changing it by one exposure zone) occupy the same distance, independently of the absolute level.

A deeper insight into the meaning of gamma can be gained by looking at a logarithmic plot of Luminance vs. Pixel level. If we take the logarithm of both sides of the luminance equation, above, and neglect black level (set it to zero), the equation becomes $log(Luminance) = log(C * value^{\Delta}) = log(C) + \Delta * log(value)$. In a logarithmic plot, gamma becomes the slope of a straight line, as illustrated on the left for gamma = 1 and 2.2. Now compare this plot to photographic paper, on the right.

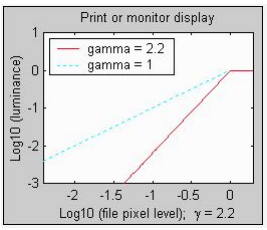

Fig. 6. Gamma

For photographic film and paper, gamma is defined at the average slope of the response curve in its linear region. As we can see in this diagram for Kodak Polymax photographic paper, higher contrast corresponds to a steeper slope– higher gamma. In comparing these two plots, note that the independent variable, Log10(Exposure) corresponds to Log10(Pixel level), while Log10(Luminance) corresponds to -Density. (Both slopes are positive because the paper is a negative material.) One unit on a Log10 scale (such as Density) equals 3.32 exposure zones (f-stops); one exposure zone equals 0.301 Density units.

Comparing these two plots should make it clear that gamma in film is essentially the same as gamma in monitors: it is the slope of the characteristic curve that relates Density (-Log10(Luminance)) to the independent variable. In other words, **Gamma is contrast**.

The Contrast control on monitors and television sets is actually brightness, and the Brightness control is, as we've seen, black level. The nomenclature is confusing but deeply entrenched. In television sets operating in typical viewing conditions, where high ambient light limits the visible dynamic range, the Contrast control affects the apparent contrast. The native gamma of monitors– the relationship between grid voltage and luminance– is typically around 2.5, though it can vary considerably. This is well above any of the display standards, so you must be aware of gamma and correct it.

A display gamma of 2.2 is the de facto standard for the Windows operating system and the Internet-standard sRGB color space. The standard for Macintosh and prepress file interchange is 1.8. Video cameras have gammas of approximately 0.45– the inverse of 2.2. The viewing or system gamma is the product of the gammas of all the devices in the system– the image acquisition device (film+scanner or digital camera),

color lookup table (LUT), and monitor. System gamma is typically between 1.1 and 1.5. Viewing flare and other factor make images look flat at system gamma = 1.0.

Gamma and black level chart

The chart below enables you to set the black level (brightness) and estimate display gamma over a range of 1 to 3 with precision better than 0.1. The gamma pattern is on the left; the black level pattern is on the right. Before using the chart, the monitor should be turned for on at least 15 minutes (30 preferred). For flat screen (LCD) monitors, Screen resolution (right-click on the wallpaper, Properties, Settings) should be set to the monitor's native resolution.

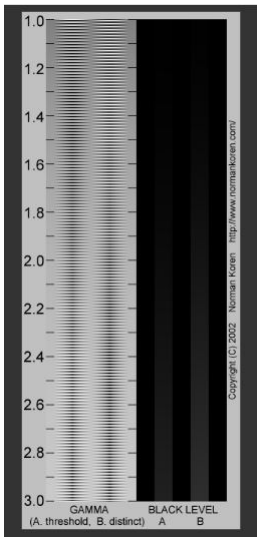

Fig. 7. Gamma

Gamma is estimated by locating the position where the average luminance across the gamma pattern is constant. The corresponding gamma is shown on the left. You should be far enough from your monitor so the line pattern is not clearly visible. The example below shows what to look for. The solid areas are calculated from the equation,

$$pixellevel = 255 * luminance(1/\Delta); luminance = 0.5$$

This chart features gradual density changes along horizontal scan lines (thus eliminating risetime problems). It allows more precise gamma estimation than most traditional charts. I encourage you to download it and check it occasionally.Your monitor's gamma should be 2.2 or 1.8.

2.2 is recommended for Windows, the Internet sRGB color space, and the popular Adobe RGB (1998) color space. 1.8 is the standard for older Macintosh systems and prepress file interchange (Mac users, see note below). I aim for gamma = 2.2. Most laptop

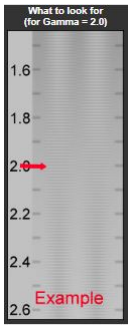

Fig. 8. Gamma

LCD screens are poorly suited for critical image editing because gamma is extremely sensitive to viewing angle.

Black level (brightness) Your monitor's brightness control (which should actually be called black level) can be adjusted using the mostly black pattern on the right side of the chart. This pattern contains two dark gray vertical bars, A and B, which increase in luminance with increasing gamma. (If you can't see them, your black level is way low.) The left bar (A) should be just above the threshold of visibility opposite your chosen gamma (2.2 or 1.8)– it should be invisible where gamma is lower by about 0.3. The right bar (B) should be distinctly visible: brighter than (A), but still very dark. There is considerable interaction between the brightness and gamma settings– increasing brightness decreases gamma– so you may have to go back and forth two or three times. There is less interaction between Contrast and gamma. The vertical bars correspond to normalized luminances of 0.002 and 0.006 at the specified gamma.

This chart is only for monitors; it doesn't work on printed media. Black White and color calibration/test charts are available for calibrating printers. Alternative gamma charts can be found on pages by Pete Andrews or Hans Brettel. Background to calibration contains specialized gamma charts with three luminance levels (0.25, 0.5, and 2/3) and three colors (RGB), useful for diagnosing monitor problems.

Monitor test patterns The patterns on the right represent an ultimate test of monitor quality and calibration. If your monitor is functioning properly and calibrated to gamma = 2.2 or 1.8, the corresponding pattern will appear smooth neutral gray when viewed from a distance. Any waviness, irregularity, or color banding indicates incorrect monitor calibration or poor performance. You will see irregularities if the black level (brightness control) is set too low, causing dark areas clip, if the monitor saturates in bright areas (a sign of old age), or if your monitor is malfunctioning in any way. The optimum black level setting is typically a little higher than the lowest level where smoothness can be achieved. This setting should be consistent with the gamma chart instructions, above.

QuickGamma [39] Eberhard Werle of Laatzen, Germany (near Hannover) has written a excellent gamma calibration utility called QuickGamma 2.0 that runs on all ver-

sions of Windows starting with 98. (Monica is a comparable Linux program.) The 437 kB installation program, QuickGammaV2EN.exe (new version with bug fix, October 2004) can be downloaded by clicking here. Running it installs QuickGamma.exe (the main program), QuickGammaLoader.exe (the program that loads the LUT at Windows start), QuickGamma.chm (the help file), and two files for easy uninstallation (using the Windows Control Panel). The program is also available from Eberhard's QuickGamma site. German and French versions can be accessed by clicking on the appropriate flag.

The right portion of the QuickGamma dialog box is shown on the right. The left portion, which consists of my gamma chart, is used to calibrate gamma and to adjust black level, which interacts with gamma. When you first load it, the gamma doesn't change. You have to click on one of the spin buttons (the up or down-arrows to the right of the Gamma box) to load QuickGamma's settings in the LUT. The number in the Gamma box increases as displayed gamma (on the chart) decreases: it indicates the monitor's native (uncorrected) gamma when the monitor is corrected to display gamma = 2.2. Version 2.0 has an option for individually adjusting gamma for Red, Green, and Blue. Unlike the color management packages (above), QuickGamma doesn't create a monitor profile. It stores the LUT values in the Windows Registry. Check the Run QuickGammaLoader at Windows Startup box if you'd like the calibration values to be loaded whenever you reboot the computer.

Fig. 9. Quick Gamma

This is an outstanding little program. I strongly recommend it if you don't have a calibrator. If your monitor is healthy it can achieve excellent calibration, which you can verify with the Monitor test pattern.

Using the Install Reference File Dialog This simple dialog allows you to install a target reference file and tell LPROF which template file goes with this target. Using the IT8 Target combo click on the button labeled "..." to select the directory where

the new target reference file is located. After the directory is selected the drop down combo box will have a list of target reference files that are located in that directory. These will not be listed by file name but rather the data listed in the drop down will have come from key fields in the reference file and will identify the target type, serial number and manufacturer. In most case the actual IT8.7 target will also have the serial number some place on the face of the target. Normally this will be located in the lower right hand corner of the target.

After selecting the target reference file that goes with your target select the correct pick template for your target from the "Pick template" drop down. At that point select OK to "install" the reference file. The reference file will be copied into the LPROF configuration directory (this directory is hidden) and will keep track of which template goes with this target. When you select a target reference file in the Camera/Scanner Profiler tab LPROF will automatically select the pick template that you selected when that target was installed.

If you made a mistake and installed a reference file with the wrong pick template you can install the reference file again and LPROF will detect that this file was previously installed and will only update the template.

6 Argyll CMS [40]

ArgyllCMS (current version is 2.1.2 (14th January 2020)) is an ICC compatible color management system, available as Open Source. It supports accurate ICC profile creation for scanners, cameras and film recorders, and calibration and profiling of displays and RGB CMYK printers. Device Link can be created with a wide variety of advanced options, including specialized Video calibration standards and 3dLuts. Spectral sample data is supported, allowing a selection of illuminants observer types, and paper fluorescent whitener additive compensation. Profiles can also incorporate source specific gamut mappings for perceptual and saturation intents. Gamut mapping and profile linking uses the CIECAM02 appearance model, a unique gamut mapping algorithm, and a wide selection of rendering intents. It also includes code for the fastest portable 8 bit raster color conversion engine available anywhere, as well as support for fast, fully accurate 16 bit conversion. Device color gamuts can also be viewed and compared with a modern Web browser using X3DOM . Comprehensive documentation is provided for each major tool, and a general guide to using the tools for typical color management tasks is also available. A mailing list provides support for more advanced usage.

Argyll is a collection of source code that compiles into a set of command line tools, licensed under an AGLP license. Argyll also includes a general purpose ICC V2 profile format access library, icclib, and a general purpose CGATS file format I/O library.

6.1 Installing Argyll in Windows

Unpacking the .zip archive You will need to unzip the downloaded file in the location you have chosen to hold the executable files (perhaps somewhere like \bin). I would NOT put them in \Program Files, since spaces in directory or file names and

command line environments are a very bad mix, and will cause you endless problems. The archive will create a top level directory Argyll_VX.X.X, where X.X.X is the version number, and the executables will be in Argyll_VX.X.X \bin.

Making the tools accessible You should also configure your %PATH% environment variable to give access to the executables from your command line environment.

For Windows 8, 8.1 10, look in Desktop - Settings - Control Panel - System And Security - System - Advanced System Settings - Environment Variables

You want to add the directory you've chosen to your %PATH%, which is a System Variable. Normally you would add this to the end of the current setting, separated by a ';'.

So if the current value of PATH is "%SystemRoot%\system32;%SystemRoot%" and you unpacked Argyll version 1.8.0 in d:\bin \, then you would modify PATH to be "%SystemRoot% \system32;%SystemRoot%;d:\bin \Argyll_V1.8.0 \bin", - i.e. you append the path to the Argyll binaries to your PATH, separated by the ';' character. The change will take effect when you start a new command shell, which you start from Start Menu-¿Accessories-¿Command Prompt, or Start Menu-¿Programs-¿Accessories-¿Command Prompt. You can check that the environment variable has been set by running the command "echo %PATH%" in the command shell.

The .zip file also contains several useful reference files (such as scanner chart recognition templates, sample illumination spectrum etc.) in the ref sub-directory, all the current documentation in a doc sub-directory, and instrument USB drivers in the usb directory.

NOTE: Vista 64/Windows 7, 8,8.1 & 10 64 bit and Beep prompt: Microsoft (in its infinite wisdom) has removed the built in speaker driver, and now relies on systems having a sound card and speakers's installed and turned on to hear system beeps. So if you're wondering where the beeps have gone when using `chartread`, now you know.

6.2 Copyright, Licensing Trade Mark

Most of the source code and provided executable files are copyrighted works, licensed under the Affero GNU Version 3 license, and therefore they (or works derived from them) can't be copied, sold or made available to users interacting with them remotely through a computer network, without providing the source code. Nothing other than your agreement and compliance with the Affero GNU License grants you permission to use, modify or distribute ArgyllCMS source code, executables or its derivative works. You could be sued for copyright infringement if you use or distribute ArgyllCMS without a valid license. The Affero GNU license prohibits extending these tools (i.e. by combining them with other programs or scripts that make use of, depend on, or work with the ArgyllCMS code) and distributing them, unless all the elements of the extensions are also made available under a GPL compatible license. It is permissible to provide ArgyllCMS tools with other non GPL components if the elements of the package are not related, such that the packaging is mere aggregation. For all the gory details, please read the accompanying license. Please note that if you wish to incorporate or make use of the code in commercial and closed source products, that you will

need to negotiate a commercial license to do so. Many portions of the ArgyllCMS code are very technically specialized and took a great deal of expertise and time to develop, and licensing cost will reflect this. Successfully negotiating a commercial license is not certain, so it is strongly advised that commercial products making use of ArgyllCMS not be developed until such a licensing agreement is in place. Note that unlike many commercial ICC profiling tools, the profiles created using ArgyllCMS, are not subject to any claims or restrictions of ArgyllCMS's author(s), but are assumed to be the copyright property of the person who gathers the characterization data, and causes the profiles to be created. The ArgyllCMS is Copyright 1995 - 2016 Graeme W. Gill, and is made available under the terms of the Affero GNU General Public License Version 3, as detailed in the License.txt file. Documentation is licensed under the terms of the GNU Free Documentation License, Version 1.3. The author asserts his moral rights over this material in relationship to the attribution and integrity of these works. In particular, if these works are modified in a way that materially changes their functionality, then the modified works should be renamed in a way that clearly distinguishes them from "Argyll" or "ArgyllCMS" so that the effects of such changes do not reflect on the original works integrity or the original authors reputation. A subset of files (those that are related to the color instrument drivers, and are collected together into the instlib.zip archive by the spectro/instlib.ksh script + xicc/ccmx.h and xicc/ccmx.c) are licensed under the General Public License Version 2 or later, as detailed in the License2.txt file. Portions of the ColorHug instrument library (spectro/colorhug.[ch]) are Copyright 2011, Richard Hughes, and is licensed under the General Public License Version 2 or later, as detailed in the License2.txt file.

The tool spectro/spec2cie.c is Copyright 2005 Gerhard Fuernkranz, and is made available under the terms of the GNU General Public License Version 2 or later, and is licensed here under the Version 3 license, as detailed in the License3.txt file. The Win32 USB library libusb-win32 kernel drivers are included in this distribution in the usb/driver and usb/bin directories, and are copyright Stephan Meyer and Travis Robinson, and are licensed under the GNU Version 2 or later (the drivers, services, installer). See usb/driver/License.txt, libusbw/COPYING_LGPL.txt and libusbw/COPYING_GPL.txt for details. Additional terms noted on the website are "This license combination explicitly allows the use of this library in commercial, non-Open-Source applications." The icc library in icc/, the CGATS library in cgats/, the jcnf library in jcnf/, the files spectro/xdg_bds.*, spectro/aglob.* and the ucmm library in ucmm/ are Copyright 1995 - 2015 Graeme W. Gill, and available according to the "MIT" permissive free software license granted in the License4.txt file, and the licenses at the top of ucmm/ucmm.c and jcnf/jcnf.c. The yajl library in yajl/ is Copyright (c) 2007-2014, Lloyd Hilaiel (Email: me@lloyd.io) and is used under an ISC permissive free software license granted in the yajl/COPYING files. The yajl library has been repackaged and modified slightly to adds some features and for packaging and build convenience. The TIFF library included in this distribution for convenience, has its own copyright and license detailed in tiff/COPYRIGHT (an "MIT"/"BSD" like permissive free software license). The Independent JPEG Group's JPEG library included in this distribution for convenience, has its own copyright and license detailed in jpg/README (an "MIT"/"BSD" like permis-

sive free software license). Executables that include JPEG format support are based in part on the work of the Independent JPEG Group.

xicc/iccjpeg.h and xicc/iccjpeg.c are from lcms and they are Copyright (c) 1998-2010 Marti Maria Saguer and is licensed under an "MIT"/"BSD" like permissive free software license. See the top of the iccjpeg.c file for the detailed copyright and licensing conditions. The mongoose web server software is Copyright (c) 2004-2011 Sergey Lyubka, and is licensed under an "MIT" permissive free software license. The axTLS library is Copyright (c) 2008, Cameron Rich, and the license is detailed in ccast/axTLS/LICENSE file (an "MIT"/"BSD" like permissive free software license). It has been modified to permit multiple threads to use it, but is not used for any security sensitive purpose, but is used purely to enable communication with the ChromeCast in a portable fashion. The X3DOM x3dom.css and x3dom.js files are Copyright (C) 2009 X3DOM and licensed dual "MIT" permissive free software and "GPL" license. See plot/X3DOM_LICENSE.txt. "ArgyllCMS" is a trade mark. It is permissible to refer to copies or derivatives of this software as being the same as ArgyllCMS if they are materially unchanged, and retain all the functionality provided by the software made available at www.argyllcms.com. Modified versions of this software that are materially changed or have missing functionality must be clearly marked as such, so as not to to be confused with ArgyllCMS.

6.3 Main Tools and the command line

These are all command line ("DOS" shell) tools, and each tool require appropriate options to be set, followed by filename arguments. Sometimes the filenames will have to include the usual extensions, sometimes they are implicit. To get a brief listing of the possible arguments and usage of any of the tools, run it with just an "-?" argument, i.e. targen -? (or some other unrecognized flag, if the "?" character is treated specially in your shell, i.e. try "–" on OS X zsh).

Note that in general the arguments consist of possible flags or options followed by file path+name arguments. All arguments need to be separated by whitespace. (If you need to specify a string with embedded white space, double quote the string). A flag consists of a dash attached to a single letter, the letter identifying the flag, and is usually case sensitive. An option is a flag that has an associated parameter or parameters. The parameter can be separated from the flag by white space, or may come directly after the flag. So if a tool has a usage that looks like this:

```
tool -?
usage: tool [options]    infile outfile
-v    Verbose mode
-d n    Choose a depth 0-4
-r    Use a random depth
-f [nn]    Use full range. nn optional range 0 - 100.
-M    Manual
infile    Input file
outfile    Output file
```

then there are 5 flags/options, and two filename arguments. Notice that square braces [] denote optional items. The first flag/option is a flag. The second is an option that has a numerical argument in the range 0 to 4. The third is a flag. the fourth is an option with an optional argument. The fourth is a flag. The flags and options can generally be in any order, but must be before the file name arguments. (For a few special tools you actually specify a sequence of flags and files where the flags apply just to the following file.) So example invocations may look like:
tool -v testin testout
tool -d3 -M testin1 testout2
tool -f infile outfile
tool -f 45 infile outfil
tool -d 3 -f67 infile outfile
In order to make use of the tools, it is necessary to keep track of where various files are, and what they are called. There are many possible ways of doing this. One way is to put each source profile and all its associated files (test charts, spectrometer values etc.) in one set of directories for each source profile type. Similarly the device profiles could be stored in a hierarchy of directories ordered by device type, media, resolution, device mode etc. Naturally you will want to set your $PATH so that you can run the tools from whichever directory you are in, as well as specify any necessary directory paths for file arguments so that the tools are able to open them. Note that there are two ways the Argyll tools deal with filename extensions. In one you supply the extension (ie. you supply the whole file name), so the extension is up to you. In the other (used where one name is used for input and output files, or where there are multiple output files), the program adds the extension. In the documentation this should be indicated by calling it a ”base name”.

6.4 The Argyll Workflow

6.5 Main Tools by Category

Calibrating devices
`dispcal` [41] Adjust, calibrate and profile a display.
`printcal` [42] Create a printer calibration .cal file from a .ti3 data file.

Creating test targets for profiling or print calibration
`targen` [43] Generate a profiling test target values .ti1 file.
`filmtarg` [44] Create film recorder TIFF files from Argyll .ti1 file.
`printtarg` [45] Create a PS, EPS or TIFF file containing test patch values, ready for printing.

Obtaining test results for profiling or print calibration
`chartread` [46] Read a test chart using an instrument to create a .ti3 data file.
`dispread` [47] Test and read colorimetric values from a display.

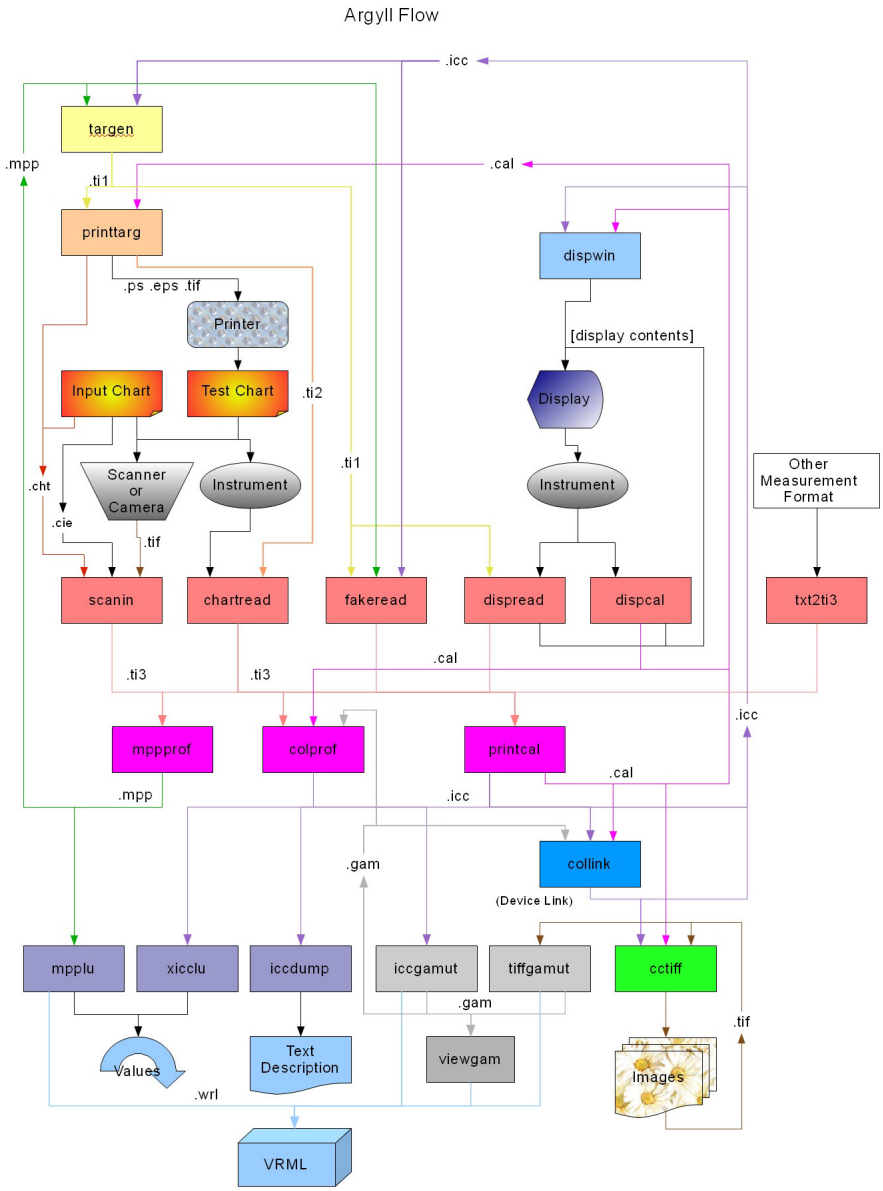

Fig. 10. ArgyllCMS Workflow

`filmread` [48] Read film colorimetric values using a SpectroScanT (Deprecated ?)
`scanin` [49] Convert a TIFF image of a test chart into .ti3 device values.
`illumread` [50] Use an instrument to measure an illuminant spectrum, and esti-

mate its UV content.
`fakeread` [51] Fake the reading of a device using an ICC or MPP profile.
`synthread` [52] Fake the reading of a device using a synthetic device model.
`cb2ti3` [53] Convert Colorblind format CMY/RGB test chart into Argyll .ti3 CGATS format.
`kodak2ti3` [54] Convert Kodak Colorflow format CMYK test chart into Argyll .ti3 CGATS format.
`txt2ti3` [55] Convert Gretag/Logo/X-Rite/Barbieri or other format RGB or CMYK test chart results into Argyll .ti3 CGATS format.
`cxf2ti3` [56] Convert X-Rite CxF3 format reference and test charts into .cie or Argyll .ti3 CGATS format.
`ls2ti3` [57] Convert LightSpace format RGB .bcs test chart results into Argyll .ti3 CGATS format.
`fake`CMY [58] Create a fake Argyll .ti3 CMY data file from a CMYK profile, as a basis of creating a CMY to CMYK separation.
`average` [59] Average or Merge two or more measurement data files, or average patches within a single file.

Creating Device Profiles
`colprof` [60] Create an ICC profile from the .ti3 test data.
`mppprof` [61] Create a Model Printer Profile (MPP) from the .ti3 test data.
`revfix` [62] Regenerate a device profiles B2A table data by inverting the A2B table.

Creating Device Link Profiles
`collink` [63] Link two device ICC profiles to create a device link profile.

Converting colors or applying print calibration
`cctiff` [64] Color convert a TIFF or JPEG file using a sequence of ICC device, device link, abstract profiles and calibration files.
`applycal` [65] Apply calibration curves to an ICC profile.
`icclu` [66] Lookup individual color values through any ICC profile table.
`xicclu` [67] Lookup individual color values forward or inverted though an ICC profile or CAL table.
`mpplu` [68] Lookup individual color values though an MPP profile. Also create MPP gamut files/views.
`greytiff` [69] Convert a TIFF file to monochrome using an ICC device profile.

Color Tweaking tools
`refine` [70] Creates an abstract profile from two chart readings, useful for refining proofing profiles.

Creating gamut views
`iccgamut` [71] Create a gamut file or VRML file of the color gamut of an ICC profile.
`tiffgamut` [72] Create a gamut file or VRML file of the color gamut of a TIFF or JPEG image.
`viewgam` [73] Convert one or more gamuts into a VRML 3D visualization file. Compute an intersection.

Diagnostic and test tools
`iccdump` [74] Dump the contents of an ICC profile as text.
`profcheck` [75] Check an ICC profile against .ti3 test chart data, create pruned .ti3 file.
`invprofcheck` [76] Check ICC forward against inverse lookup.
`splitsti3` [77] Split a CGATS file (ie. a .ti3) into two parts randomly to verify profiling.
`timage` [78] Create TIFF test images.
`mppcheck` [79] Check an MPP profile against .ti3 test chart data.
`spotread` [80] Use an instrument to read a single spot color value.
`colverify` [81] Verify matching of CIE in two CGATS/.ti3 files (also view differences as VRML)
`synthcal` [82] Create a synthetic input, display or output calibration (.cal)file.

Other Tools
`ccxxmake` [83] Use a Spectrometer to create a Colorimeter Correction Matrix (CCMX) or a Colorimeter Calibration Spectral Set (CCSS) for a particular display.
`extracticc` [84] Extract an embedded ICC profile from a TIFF or JPEG file.
`extractttag` [85] Extract a text tag (ie. CGATS .ti3 data or CAL) from an ICC profile.
`dispwin` [86] Install or uninstall display profile, set display calibration from profile or .cal file, test displace and dispwin access to a display.
`oeminst` [87] Install Instrument manufacturers files for the Spyder 2, EDR or CCSS calibration files for i1d3 or Spyder 4 or 5, CCMX files for colorimeters.
`specplot` [88] Plot a spectrum (.sp, .cmf, .ccss) and calculate CCT and VCT.
`spec2cie` [89] Convert spectral .ti3 or .sp readings into CIE XYZ and D50 L*a*b* readings. Apply FWA, plot spectrums. Convert to/from XRGA standard.

File formats that Argyll uses [90]
Argyll uses a number of file formats for its operation, some that are external standards, and some that are unique to Argyll. `.ti1` Device test values
`.ti2` Device test values chart layout
`.cal` Device calibration information.
`.cht` Test chart recognition template.

.gam 3D gamut surface description
.sp Illuminant spectral description
.cmf Color Matching Functions
.ccmx Colorimeter Correction Matrix
.ccss Colorimeter Calibration Spectral Set
CGATS Standard text based data exchange format
ICC International Color Consortium profile format
MPP Model device profile format
TIFF Tag Image File Format raster files.
JPEG Joint Photographic Experts Group, JPEG File Interchange Format raster files.
ucmm Unix micro Color Management Module convention and configuration file format and Profile Locations.
VRML Virtual Reality Modelling Language 3D file format.
X3D Open standards file format to represent 3D scenes using XML.
X3DOM Open-source framework and runtime for 3D graphics on the Web.

6.6 Related technical information

Calibration vs. Characterization
What is Calibration ?
"Calibration" is a short hand Graphic Arts term for adjusting a devices behavior to meet calibration targets. Calibration is the process of modifying the color behavior of a device. This is typically done using two mechanisms:
1) Changing controls or internal settings that it has. 2) Applying curves to its color channels.

The idea of calibration is to put a device is a defined state with regard to its color response. Often this is used as a day to day means of maintaining reproducible behavior. Calibration is often the most practical way of setting parameters such as white point and brightness of displays. Typically calibration will be stored in device or systems specific file formats that record the device settings and/or per channel calibration curves.

What is Characterization ?
Characterization (or profiling) is recording the way a device reproduces or responds to color. Typically the result is stored in a device ICC profile. Such a profile does not in itself modify color in any way. What it does is allow a system such as a CMM (Color Management Module) or color aware application to modify color when combined with another device profile. Only by knowing the characteristics of two devices or colorspaces, can a way of transferring color from one device representation to another be achieved.

Note that a characterization (profile) will only be valid for a device if it is in the same state of calibration as it was when it was characterized.
What about display calibration and profiles ?
In the case of display profiles there is some additional confusion because often the calibration information is stored in the profile for convenience. By convention it is

stored in a tag called the 'vcgt' tag. Although it is stored in the profile, none of the normal ICC based tools or applications are aware of it, or do anything with it, it is just "along for the ride". Similarly, typical display calibration tools and applications will not be aware of, or do anything with the ICC characterization (profile) information.

Fluorescent Whitener Additive Compensation (FWA Compensation)

Introduction

To make paper look "whiter" without increasing the cost of production, paper manufactures often employ a couple of different techniques. One technique is to add "shading agents" to the paper, that absorb a little of the middle wavelengths, thereby changing the color of the paper to be a little less green. By far the most powerful way of making the paper appear more white is to add Fluorescent Whitener Additive (FWA, or Optical Brightening Agents - OBA) to the paper. This is basically a fluorescent material that absorbs light at Ultra Violet (U.V.) wavelengths, and re-emits it at a slightly longer blue wavelengths. Subjectively something that appears more blue, is regarded as being "whiter".

For more technical treatment of this topic, please refer to this excellent paper here [91].

Fluorescence

Fluorescent materials absorb light radiation at one wavelength, and then almost instantaneously re-emit some of that energy at a longer wavelength. Typical FWA absorbs wavelengths in the U.V. between about 300 and 400 nm, and re-emit it between 400 and 460nm. The visual effect of FWA depends on the amount of it present in the paper, and the amount of U.V. in the illumination compared to the level of normal, visible light. Generally better quality papers have lower levels of whitening agents, and cheaper papers more.

Reflection Models and Spectro-colorimetry

The way a spectrometer measures the effect of ink on paper, depends on a model of how an illuminant is reflected by the ink and the paper. Typically a spectrometer instrument illuminates the sample with a known illumination, often a incandescent tungsten lamp having a color temperature of 2800 degrees Kelvin. It measures the amount of light reflected by the sample at each wavelength, and then converts that to spectral reflectance value between 0 and 100% by dividing by it's measurement illuminant's intensity at each wavelength. When it comes time to use that measurement to create an ICC profile, the intensity of the assumed viewing illumination at each wavelength (typically D50 for standard ICC profiles) is then multiplied by the reflectance at each wavelength, and the overall spectral reflectance is in this way converted into CIE tri-stimulus values using an observer model.

So while the instrument measures with one type of light (type A, or a white LED), it returns a measurement as if it had been measured under a different kind of light (D50) by making use of a simple model of light reflection off the media.

Notice that a key assumption of this simple model is that the light that impinges on the sample at a given wavelength is reflected back at exactly the same wavelength

at a diminished intensity. Notice also that any sort of fluorescent material (such as FWA) breaks this model, since fluorescent materials emit light a different wavelengths to which they absorb it. So the color measurements do not accurately portray the appearance of the media when FWA is present. A more complicated bi-spectral measurement (2 dimensional spectral reflectance) is actually needed to fully characterize fluorescent materials.

What Argyll's FWA compensation does

The FWA compensation function in Argyll improve on this simple model of spectral reflection by taking into account the action of FWA. To do this, it needs to measure the amount and nature of the FWA in the media, and then have enough information about the viewing environment to model how that FWA will behave.

To be able to measure the level of FWA in the media, the instrument needs to be able to "see" the FWA in action, so the instrument needs to be illuminating the samples with some level of U.V. Typically all instruments do this, unless they have been fitted with a filter that filters out any U.V. illumination (so called "UV cut" instruments), or use an illumination source such as a "white" LED that doesn't emit any U.V. Such UV excluded instruments are not suitable for use with FWA compensation. The effects of FWA are modeled spectrally, so a spectral reading instrument is also required.

Argyll can compute a model for the effects of FWA given the media's spectral characteristics, and the illuminations spectral characteristic, which must include the levels of U.V. in the illuminant. Given these two things, Argyll can calculate how much effect the FWA will have on the light being reflected and emitted by the media under the intended illumination.

Ideally the level of FWA would be measured by comparing the paper spectrum with and without U.V. present in the instruments illumination. Because not all instruments allow these two measurements to be done without some sort of manual intervention, Argyll avoids the need for an FWA inactive (UV cut) or extra UV (UV LED) measurement by employing a heuristic to estimate the FWA inactive spectrum from the spectrum of the paper with FWA active. Being a heuristic, it can sometimes be fooled by certain paper colors into estimating more or less FWA content than is actual present. The heuristic works best with high quality papers with an essentially flat non-FWA enhanced spectrum. Papers with colored tints or particularly off white appearance may not work well with FWA compensation, unless the instrument has the capability of measuring with two different levels of UV.

Note that typically in Argyll, if a viewing illuminant is specified, then it is used for computing the appearance under that illumination (CIE XYZ values), and if FWA compensation is used, then that same illuminant will be assumed for the simulated measurement illuminant. This results in measurements that better reflects the appearance as the media as if it was being viewed under that illuminant, FWA effects and all.

It is possible to also simulate the measurement of a media under one illuminant, while then computing the tristimulus values as if being viewed under a different illuminant, but this scenario is only really useful for reproducing standardized measurement conditions such as ISO 13655:2009 M0, M1 and M2, and is less useful than the normal

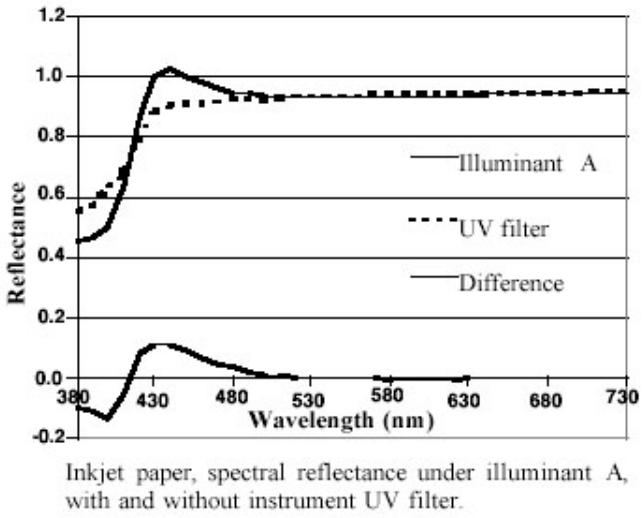

Fig. 11. Spectral characteristics with and without UV filter

FWA compensation scenario in modelling real world situations. The paper written by Graeme can be found here [92].

Using FWA Compensation with proofing

The most common situation for employing FWA compensation, is in proofing. This is when you have one printing device, the target (say a printing press), and wish to emulate the behaviour of it with a different device, the proofer (say an inkjet printer). The aim is to be able to put both prints next to each other in a viewing booth, and have them look identical. Typically the printing process, the inks, and the media will be different between the target device and the proofer. The aim of applying color profiling is to compensate for these differences. Since the printing process can only darken a white media, the selection of the proofing stock is critical. Ideally it should be exactly the same color as the target, or if not possible, lighter, so that the proofer can tint the proofing media to match the target. If the two media had identical levels and types of FWA in them, then there would be no need to use FWA compensation, since the appearance of the media would match under any viewing condition. Typically though, the levels and types of FWA are different between the target paper and the proofing paper. A limitation imposed by tri-stimulus colorimetry is that the differences between the two media, inks and FWA can only be compensated for perfectly, under a fixed and known illuminant.

By allowing Argyll to model the effects of FWA for both the source profile (the target device), and the destination profile (the proofing device), the effects can be accounted for, modeled accurately, and incorporated in the profiles, so that a subsequent transformation from source to destination device spaces using absolute colorimetric intent, achieves a (hopefully) perfect colorimetric reproduction. Since this is a closed system, where both the source and destination profiles are made for each other, non-

standard parameters such as illuminant and observer models can be used, as long as they are the same for both profiles. For proofing, FWA should be applied identically to both profiles, by specifying the same illuminant, and (optionally) the same observer model.

[In practice it is possible to compensate for the color shift that results in viewing the media under non-D50 illumination or using a non 1931_2 deg observer, or allowing for FWA effects without severe incompatibility because all rendering intents except absolute rendering normalize to the media color, rendering the media white as white, even though the absolute values are not measured using a D50 illuminant.]

Using FWA compensation for single, general use profiles

For creating ICC profiles that will be interchanged with other unknown ICC profiles, or used with non-print source or destination profiles, there is less flexibility, since ICC profiles by convention assume that all media is being viewed under D50 illumination. The implication of this is that to be fully interchangeable, it's not really possible to make the profile for your actual viewing environment. Note that the D50 values that are calculated without FWA compensation do not actually reflect the appearance of a media under real D50, because they fail to take into account the different levels of FWA activity between the illumination using by the instrument to measure the media, and real D50. To allow for this and actually meet the letter of the ICC specifications, FWA compensation should ideally be used when building a interchangeable ICC profile, by selecting the D50 illuminant, and the 1931_2 observer model (ISO 13655:2009 M1). Note however that this is likely to make profiles less interchangeable rather than more, since few if any other profiles will represent the appearance under real D50, since few if any instruments use a real D50 illuminant that will trigger the correct level of FWA response, and few if any other packages will compensate for the differences in FWA activity between the instrument illuminant used and real D50 (ie. most instruments are actually returning ISO 13655:2009 M0 measurements).

Similarly, the effects of viewing the media in an environment with a UV filter fitted over the D50 illuminant can be simulated by using FWA compensation with the D50M2 illuminant, and the 1931_2 observer, thereby simulating the results one would get if the media had been measured with a "UV cut" type instrument, although such profiles are not technically ICC compatible.

Measuring the illuminant

For FWA compensation to work well, it is necessary to know what the spectral shape of the illuminant used for viewing is. While many instruments provide an illuminant measurement capability over the visible spectrum, for FWA compensation it is desirable to know the Ultra Violet (UV) component of the illuminant. Few color instruments are capable of reading to such short wavelengths though (the JETI specbos 1211 is an exception). Argyll provides an indirect way of estimating the UV component of an illuminant using its illumread utility. Using illumread in combination with FWA compensation is the recommended approach to modelling real world appearance of paper containing FWA.

FWA myths

Amongst the user (and to some degree) vendor community, there is a widely held belief that the solution to fluorescent whitener affecting color profiles is to simply use a UV filter fitted instrument. Exactly what the origin of the legend is, is hard to tell. Possibly it is a misinterpretation of the ANSI CGATS.5-1993 Annex B recommendations for measuring the impact of fluorescent effects, a translation of some of paper whiteness measurement standards into the color profiling world, or possibly in some common situations, if the viewing environment is very poor in UV, then adding a UV filter to the tungsten instrument illuminant makes for a better instrument/viewing illuminant match. There seems to be no scientific or practical basis for believing that a UV filter fitted instrument magically makes all FWA induced problems go away.

Instrument UV filters

Note that to be able to measure the FWA in the paper, the instrument has to be able to trigger Fluorescence, which it cannot do if it is fitted with a UV filter, or uses a light source that emits no UV (e.g. a white LED). So UV excluded instruments are not suitable for use with FWA compensation.

About ICC profiles and Gamut Mapping

How ICC profiles support different intents cLUT (Color Lookup Table) based ICC profiles support multiple intents by having a table for each intent. In a typical device cLUT profile, there are up to 6 cLUT's, three for input (AtoB tables, that convert from device space to PCS (Profile connection space)), and three for output (BtoA tables, that convert from PCS to device space). The tables allow the use of different color transforms, each transform being tailored for a different effect:

AtoB0, BtoA0: Perceptual
AtoB1, BtoA1: Colorimetric
AtoB2, BtoA2: Saturation

The colorimetric intent is meant to convey the exact device color behaviour, without any gamut mapping. Typically it is used to store the devices behaviour (characterization), and is also used where exact color reproduction is required, such as for proofing. The Colorimetric tables double up for both relative colorimetric and absolute colorimetric with the application of a white point restoration.

The Perceptual and Saturation tables are meant to contain gamut mapping combined with the device characterization. The allowance for this in both the AtoB direction, as well as the BtoA direction permits a profile to gamut map from the device gamut to some intermediate gamut, and then from the intermediate gamut to the device gamut.

[Note that Shaper/Matrix profiles are always Colorimetric intent, since there is only a single transformation, and it does not have the necessary flexibility to accommodate gamut mapping.]

ICC Version 2 behaviour

Apart from defining the general purpose of the different tables, the ICC Version 2 specification doesn't specify exactly how they are to achieve this, so it is up to the profile

maker to make a choice in this regard. There is no common gamut boundary specified for the PCS, and such an approach limits the achievable intents in any case (see ICC Version 4 behaviour for an explanation why).

What I've chosen to do with Argyll profiles, is to make all the AtoB tables the same as colorimetric. This means that the conversion used for the source profile is always colorimetric, and also means that the source gamut seen by the destination profile is the source colorspace gamut. This means that the gamut mapping is done solely in the BtoA tables, and that their task is to map the source colorspace gamut to the destination colorspace gamut. So to construct the perceptual and saturation intent mapping tables, a source profile or source gamut needs to be specified, so that a gamut mapping can be constructed.

The advantages of this approach is that the behaviour is precisely defined, a full range of gamut mapping options is available, and compatibility with matrix profiles (which do not have gamut mapping transforms) and other foreign profiles can be assured, by simply using such profiles as colorimetric sources. The main disadvantage is that the gamut mapping will only operate exactly as intended when the profile is linked with the source profile it was setup for. This is really a fundamental limitation of the idea of having pre-computed gamut mapping color transforms, that the ICC profile format was intended to support.

Some non-Argyll profiles have gamut mapping transforms in their Perceptual and Saturation A2B tables, and this means that the apparent gamut of a source through these tables may be different to the actual device gamut. To accommodate using these profiles with CMM's (Color Management Modules) that do not permit the separate choice of intent tables for the source and destination profiles, Argyll will by default use the gamut defined by the source profile perceptual table to create the gamut mapping of the destination perceptual table, and the source saturation table to make the destination saturation table. Note that this can affect the exact nature of the gamut mapping, the distortion of the source gamut changing the apparent relationship between it and the destination gamut - see "ICC Version 4 behavior" for an illustration of the kind of changes this causes. [This default can be overridden though using the colprof -nP and -nS flags.]

ICC Version 4 behaviour

Argyll does not currently support ICC V4.

By default, ICC Version 4 profile operates similarly to the ICC V2 profile in regard to gamut mapping, with the exception that a minimally specified reference medium and reference viewing conditions are introduced for perceptual (and presumably saturation) tables, allowing at least the luminance range to have a well defined behavior when mixing and matching the perceptual A2B and B2A tables of different profiles. A slight adjustment was made to the permitted tag contents, to allow things like Display profiles to contain the full range of AtoB and BtoA tables, so that they could also be gamut mapped. An optional part of ICCV4, introduces a more comprehensively specified Profile Reference Medium Gamut (PRMG) as an intermediate gamut boundary between the source colorspace, and the destination colorspace. If this option is used, then an additional tag in the ICCV4 profile indicates that this is the case. This then solves the problem of the gamut mapping having to know the source and destination

gamuts to operate. Instead, the gamut mapping is split into two parts, the first where the source gamut to RMG is done by the AtoB tables, and then the RMG to destination gamut is done by the BtoA tables. Profiles can therefore be mix and matches, while retaining true gamut mapping.

This approach has a number of drawbacks though. One is that the colors get gamut mapped twice. Gamut mapping is sometimes not very precise, and the geometry of the transforms may not cancel out, especially since different profile vendors may choose different algorithms in their gamut mapping. By ”cancel out”, I mean that even if you were linking the same source colorspace to the same destination colorspace, the gamut may be expanded (say) in the process of mapping to the PRMG, and then compressed again in mapping from the RMG to the device space, and these expansions and compressions may not quite match. Given that the PRMG is a relatively large gamut, larger than many real devices actual behavior, this sort of expansion and re-compression will be the normal thing.

The chief drawback, is that only one (non colorimetric) intent can really be supported, that of saturation.

The typically expected behavior of perceptual intent gamut mapping, is to compress any areas of the source gamut that lie outside the destination gamut, but for areas that fall within the destination gamut, change them as little as possible, consistent with keeping smooth and proportional with respect to the compressed colors. This preserves the source ”look” as much as possible, while ensuring that out of gamut colors are smoothly brought within the destination gamut.

Typical behavior of a saturation intent, is (at least), to not only compress out of gamut source colors to fit within the destination, but to expand any source boundary that falls within the destination gamut outwards match the destination gamut. Some practical saturation gamut mappings may go further than this, and expand a little beyond the destination gamut to ensure fully saturated boundary colors, and also enhance the saturation of all colors mapped through it.

By mapping the source gamut to the RMG in the A2B, all information about what areas of the source gamut are inside or outside of the destination gamut are lost, so the destination gamut mapping can not known which colors may be left unchanged, and which really need compressing. All it can do is map the RMG to match the destination gamut, thereby effecting a saturation style intent.

If the source was not expanded out to fill the RMG in some area by the A2B, then the resulting output will be over compressed and end up looking dull, because the B2A table has no choice but assume that there may be colors that do fill the RMG.

Once again, this is all a fundamental limitation of using pre-computed gamut mappings. The only effective way of overcoming such limitations is to move to a more active color management architecture, in which gamut mappings are computed at link time, to accommodate the actual source and destination gamuts.

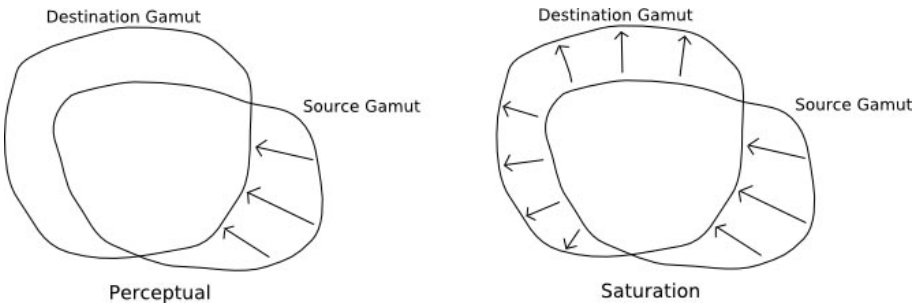

Fig. 12. Gamut Mapping

About display monitor settings and targets
Setting monitor controls and target behaviour for monitor calibration boils down to two things:

- What is the equipment capable of without introducing side effects ?
- What are you trying to do ?

There are three reasons you may want to adjust display settings and set calibration targets:

- You want to change how non-color managed applications appear.
- You want to change basic behaviour of the display that the profile based color management doesn't usually change, such as white point and brightness.
- You want to improve the behaviour of the device so that the normal profile based color management does a better job of controlling the display.

You can make adjustments to a display using it's controls and/or the video card LUTs. Generally the former are more powerful and have less side effects. There can be exceptions though, for instance LCD's have no native contrast control capability, only brightness, so contrast is usually faked by manipulating the lookup curves, which can introduce side effects. The same applies to white point control on an LCD (unless it has R/G/B LED back lighting). So generally LCD displays are much less flexible than CRT displays in targeting some non-native colorspace without introducing side effects, so it is generally best to set all the LCD controls except back lighting brightness to their default settings.

Brightness depends on what you are trying to do. If you are trying to do soft proofing for instance, you will have some brightness level in mind dictated by the hard proofing booth you are comparing to, or the ambient brightens level. For good color judgement and low fatigue it's desirable that the display brightness roughly match that of the ambient lighting.

In terms of what you are trying to do, it comes down to what colorspace you want the display to be, and how far from native for that display it is. A CRT can be reasonably flexible in the behaviour it can be given without side effects, an LCD less so. If you want to minimize artefacts on an LCD you want to set the contrast and white points to their

native values (ie. where the monitor is not manipulating the digital signal levels). It may not be easy to figure out what this is. In this scenario you would probably only want calibration to set the transfer characteristic and neutral axis, and leave the white point native.

For typical MSWindows/Linux this would probably be the typical CRT transfer ("gamma") curve, or a gamma of about 2.4. For OS X it would probably be a gamma of 1.8 for versions 10.5 or earlier, or 2.4 for 10.6 or latter ("Snow Leopard").

The nominal white point of a display is D65 (set by Television standards), and an LCD's native white point is somewhere near there, but this is dictated by their backlight color. The CRT's will give maximum brightness with a much higher white point (9000K or so), but this can be reduced with fewer side effects (just reduced brightness) using typical CRT controls.

If you have specific requirements (trying to do soft proofing) then you may want to target a specific white point and brightness, and be prepared to compromise other aspects of the display to achieve this. By all means use the controls to move the display in the direction you want to go, and then use the calibration curves to get there. If you are moving far from native (especially on an LCD) you may find the side effects unacceptable though.

About Gamma

When calibrating display devices, the notion of "gamma value" quickly becomes a topic for discussion. Various numbers are often bandied about as if they have a well known and accepted meaning, but it turns out that gamma values are not a very precise way of specifying real world device behavior at all.

A "gamma" curve is typically thought of as an ideal power curve, but no real world device has the necessary zero output at zero input to be able to match such a curve, and in general a display may not exactly reproduce an idealized power curve shape at all. The consequence of this is that there are countless ways of matching a real world curve with the ideal gamma power one, and each different method of matching will result in a different notional gamma value.

Argyll's approximate specification and reading is simply the gamma of the ideal curve that matches the real 50% stimulus relative-to-white output level. I think this is a reasonable (robust and simple) approximation, because it matches the overall impression of brightness for an image. A more sophisticated approximation that could be adopted would be to locate the idea power curve that minimizes the total delta E of some collection of test values, but there are still many details that the final result will depend on, such as what distribution of test values should be used, what delta E measure should be used, and how can a delta E be computed if the colorimetric behavior of the device is not known ? Some approaches do things such as minimize the sum of the squares of the output value discrepancy for linearly sampled input values, and while this is mathematically elegant, it is hard to justify the choice of device space as the metric.

There are many other ways in which it could be done, and any such approximation may have a quite different numerical value, even though the visual result is very similar. This is because the numerical power value is very sensitive to what's happening

near zero, the very point that is non-ideal. Consider the sRGB curve for instance. It's technically composed of a power curve segment with a power of 2.4, but when combined with its linear segment near zero, has an overall curve best approximated by a power curve of gamma 2.2. Matching the 50% stimulus would result in yet another slightly different approximation value of about 2.224. All these different gamma values represent curves that are very visually similar.

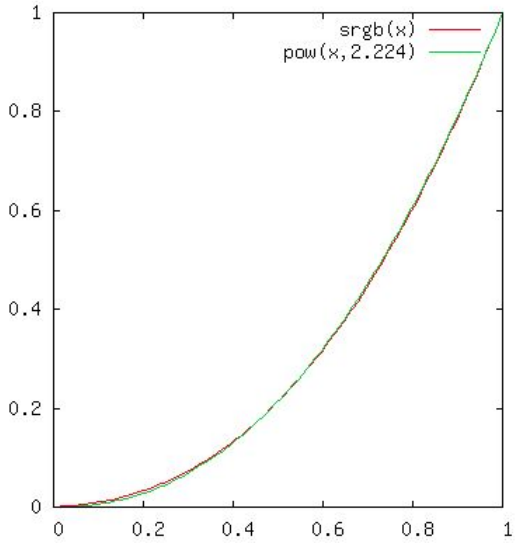

Fig. 13. Plot of sRGB curve vs. power of 2.224

The result of this ambiguity about what gamma values mean when applied to real world curves, is that it shouldn't be expected that there are going to be good matches between various gamma numbers, even for curves that are very visually similar, unless the precise method of matching the ideal gamma curve to the real world curve is known.

Crushed Display Blacks

Often people create a display profile, and then notice that when they try and display some images using the profile, that the darkest blacks in the image all get crushed into the black of the display. Why does this happen ?

There are many reasons this may happen, but here is a common one:

The image has blacks that are darker than the black of the display, and the color management intent being used clips out of gamut colors. So all the blacks that are darker than the display black get mapped to the display black. To avoid this, some sort of Gamut Mapping that maps that black of the source image to the black of the display, while preserving the distinction between all the rest of the colors needs to be used.

Some popular synthetic colorspaces have a perfect (and unrealistic) zero black, for instance sRGB and AdobeRGB. Real world display profiles have non-zero blacks, so transforming between these two using a colorimetric intent will clip the blacks, and loose the shadow details.

What performs gamut mapping ?
Typically there are only two mechanisms available to perform gamut mapping. The main one is a pre-cooked (static) gamut mapping built into cLUT type ICC profiles. The second is an on-the-fly (dynamic) gamut mapping performed by the CMM (Color Management Module). A limited form of the latter is Adobe BPC (Black point compensation), which is also available sometimes with applications or systems that use lcms. (Little cms).
How do I fix it ?
There are two ways of avoiding the black crush. One is to turn on BPC if it is available in the system you are using. Sometimes it may only be available for certain intents.

The second way of fixing it is to create your display profile with appropriate gamut mapping, and make sure that it gets used.

Shaper/Matrix type ICC profiles do not support gamut mapping, since there is only one transformation in them and it does not have the necessary flexibility to incorporate gamut mapping. Shaper/Matrix profiles are always colorimetric intent. So it is necessary to create a cLUT based Display profile if gamut mapping is to be incorporated into the profile. (Note that not all systems accept cLUT based Display profiles). Creating cLUT profiles that incorporate appropriate gamut mapping depends on the profile creation tools, and not all tools give adequate control over gamut mapping to reliably fix this problem.

OK, so how do I fix it using Argyll ?
You can usually fix this problem using Argyll by simply creating a cLUT based profile (the default), and telling colprof what the source colorspace is going to be.

i.e. say your source images are in sRGB space, then:

```
colprof -v -S sRGB.icm -D ”My Display”
MyDisplayProfile
```

[It’s usually safer to use the sRGB profile provided by Argyll than use an sRGB profile of unknown origin. Find it in the ref directory.]

This will create 3 separate B2A cLUT tables, one for colorimetric intent, one for Perceptual intent, and one for Saturation intent. Both Perceptual and Saturation tables will have appropriate gamut mapping for a source colorspace of sRGB. So it is just a matter of making sure that either Perceptual or Saturation intent is used when making use of the display profile.

7 Scenarios possible with Argyll

7.1 Display profiling

As on the date of writing this manual, I do not have any instrument for measuring patches for display profiling. Hence I would urge you to refer LProf's version of a instrument less display profiling strategy mentioned above.

7.2 Profiling Scanners and other input devices such as cameras

I prefer using the LProf's GUI based scanner profiling strategy instead of command driven Argyll's method. As on this date of writing this manual, LittleArgyll GUI does not support scanner calibration. CoCa GUI does support multiple scanner profiling test targets and reference values. But I could not get them to work. I shall refer to CoCa's documentation at a later stage in this manual.

Types of test charts The most common and popular test chart for scanner profiling is the IT8.7/2 chart. This is a standard format chart generally reproduced on photographic film, containing about 264 test patches. An accessible and affordable source of such targets is Wolf Faust a www.coloraid.de. Another source is LaserSoft www.silverfast.com.

- The Kodak Q-60 Color Input Target is also a typical example.
- A very simple chart that is widely available is the Macbeth ColorChecker chart, although it contains only 24 patches and therefore is probably not ideal for creating profiles
- Other popular charts are the X-Rite/GretagMacbeth ColorChecker DC and ColorChecker SG charts
- A chart provided for camera profiling is the X-Rite ColorCheckerPassport
- The GretagMacbeth Eye-One Pro Scan Target 1.4 can also be used
- Also supported is the HutchColor HCT
- Christophe Métairie's Digital TargeT 003, Christophe Métairie's Digital Target - 4 , and Christophe Métairie's Digital Target - 7 and Christophe Métairie's Digital Target 2019, Christophe Métairie's Digital Target Studio Edition
- The LaserSoft Imaging DCPro Target
- and the LaserSoft Imaging ISO12641-2 reflective target
- and the LaserSoft Imaging ISO12641-2 transparency target: in three parts
- The Datacolor SpyderCheckr
- The Datacolor SpyderCheckr24
- One of the QPcard's, QPcard 201, & QPcard 202

Taking readings from a scanner or camera The test chart you are using needs to be placed on the scanner, and the scanner needs to be configured to a suitable state, and restored to that same state when used subsequently with the resulting profile. For a camera, the chart needs to be lit in a controlled and even manner using the light source that will be used for subsequent photographs, and should be shot so as to minimise any geometric distortion, although the scanin -p flag may be used to compensate for

some degree of distortion. As with any color profiling task, it is important to setup a known and repeatable image processing flow, to ensure that the resulting profile will be usable.

The chart should be captured and saved to a TIFF format file. I will assume the resulting file is called scanner.tif. The raster file need only be roughly cropped so as to contain the test chart (including the charts edges).

The second step is to extract the RGB values from the scanner.tif file, and match then to the reference CIE values. To locate the patch values in the scan, the scanin tool needs to be given a template .cht file that describes the features of the chart, and how the test patches are labeled. Also needed is a file containing the CIE values for each of the patches in the chart, which is typically supplied with the chart, available from the manufacturers web site, or has been measured using a spectrometer.

For an IT8.7/2 chart, this is the ref/it8.cht file supplied with Argyll, and the manufacturer will will supply an individual or batch average file along with the chart containing this information, or downloadable from their web site. For instance, Kodak Q60 target reference files are here. NOTE that the reference file for the IT8.7/2 chart supplied with Monaco EZcolor can be obtained by unzipping the .mrf file. (You may have to make a copy of the file with a .zip extension to do this.)

For the ColorChecker 24 patch chart, the ref/ColorChecker.cht file should be used, and there is also a ref/ColorChecker.cie file provided that is based on the manufacturers reference values for the chart. You can also create your own reference file using an instrument and chartread, making use of the chart reference file ref/ColorChecker.ti2: chartread -n ColorChecker.ti2 Note that due to the small number of patches, a profile created from such a chart is not likely to be very detailed.

For the ColorChecker DC chart, the ref/ColorCheckerDC.cht file should be used, and there will be a ColorCheckerDC reference file supplied by X-Rite/GretagMacbeth with the chart.

The ColorChecker SG is relatively expensive, but is preferred by many people because (like the ColorChecker and ColorCheckerDC) its colors are composed of multiple different pigments, giving it reflective spectra that are more representative of the real world, unlike many other charts that are created out of combination of 3 or 4 colorants. A limited CIE reference file is available from X-Rite here, but it is not in the usual CGATS format. To convert it to a CIE reference file useful for scanin, you will need to edit the X-Rite file using a plain text editor, first deleting everything before the line starting with ”A1” and everything after ”N10”, then prepending this header, and appending this footer, making sure there are no blank lines inserted in the process. Name the resulting file ColorCheckerSG.cie. There are reports that X-Rite have experimented with different ink formulations for certain patches, so the above reference may not be as accurate as desired, and it is preferable to measure your own chart using a spectrometer, if you have the capability. If you do happen to have access to a more comprehensive instrument measurement of the ColorChecker SG, or you have measured it yourself using chart reading software other than ArgyllCMS, then you may need to convert the reference information from spectral only ColorCheckerSG.txt file to CIE value ColorCheckerSG.cie reference file, follow the following steps: txt2ti3 ColorCheckerSG.txt ColorCheckerSG spec2cie ColorCheckerSG.ti3 ColorCheckerSG.cie

For the full ColortChecker Passport chart, the ref/ColorCheckerPassport.cht file should be used, or if just the 24 patches corresponding to the original ColorChecker are in the shot, the ref/ColorCheckerHalfPassport.cht should be used. A user has kindly provided their measured values for this chart, and they are available in ref/Color CheckerPassport.cie and ref/ColorCheckerHalfPassport.cie respectively.

For the Eye-One Pro Scan Target 1.4 chart, the ref/i1_RGB_Scan_1.4.cht file should be used, and as there is no reference file accompanying this chart, the chart needs to be read with an instrument (usually the Eye-One Pro). This can be done using chartread, making use of the chart reference file ref/i1_RGB_Scan_1.4.ti2: chartread -n i1_RGB_Scan_1.4 and then rename the resulting i1_RGB_Scan_1.4.ti3 file to i1_RGB $_Scan\_1.4.cie$

For the HutchColor HCT chart, the ref/Hutchcolor.cht file should be used, and the reference .txt file downloaded from the HutchColor website.

For the Christophe Métairie's Digital TargeT 003 chart with 285 patches, the ref/CMP _DT_003.cht file should be used, and the cie reference files come with the chart.

For the Christophe Métairie's Digital Target-4 chart with 570 patches, the ref/CMP _Digital_Target-4.cht file should be used, and the cie reference files come with the chart.

For the Christophe Métairie's Digital Target-7 chart with 570 patches, the ref/CMP _Digital_Target-7.cht file should be used, and the spectral .txt file provided with the chart should be converted to a cie reference file: txt2ti3 DT7_XXXXX_Spectral.txt temp spec2cie temp.ti3 DT7_XXXXX.cie

For the Christophe Métairie's Digital Target 2019 with 522 patches, the ref/CMP _Digital_Target_2019.cht file should be used, and the spectral .txt file provided with the chart should be converted to a cie reference file: txt2ti3 "DT SPECT 451.txt" temp spec2cie temp.ti3 DT_SPECT_451.cie

For the Christophe Métairie's Digital Target Studio Edition chart with 988 patches, the ref/CMP _Digital_Target_Studio.cht file should be used, and the spectral .txt file provided with the chart should be converted to a cie reference file: txt2ti3 "CMP DT-SE 100 Spectral.txt" temp spec2cie temp.ti3 CMP_DT-SE_100_Spectral.cie

For the LaserSoft DCPro chart, the ref/LaserSoftDCPro.cht file should be used, and reference .txt file downloaded from the Silverfast website.

For the LaserSoft ISO 12641-2 reflective chart, the ref/ISO12641 _2_1.cht file should be used, and reference .CxF file downloaded from the Silverfast website, and the .CxF file converted to the cie reference file: cxf2ti3 Rnnnnnn.cxf Rnnnnnn

The LaserSoft ISO 12641-2 transmissive chart comes in three parts, and the ref/ ISO12641_3_1.cht, ref/ISO12641_3_2.cht and ref/ISO12641_3_2.cht recognition files should be used, and the reference .CxF file downloaded from the Silverfast website, and the .CxF file converted to the cie reference file: cxf2ti3 Ennnnnn.cxf Ennnnnn After creating three corresponding .ti3 files using scanin, the files should be combined using average: average -m scan1.ti3 scan2.ti3 scan3.ti3 combined.ti3

For the Datacolor SpyderCheckr, the ref/SpyderChecker.cht file should be used, and a reference ref/SpyderChecker.cie file made from measuring a sample chart is also

available. Alternately you could create your own reference file by transcribing the values on the Datacolor website.

For the Datacolor SpyderCheckr24, the ref/SpyderChecker24.cht file should be used, and a reference ref/SpyderChecker24.cie file made from measuring a sample chart is also available. Alternately you could create your own reference file by transcribing the values on the Datacolor website.

For the QPCard 201, the ref/QPcard_201.cht file should be used, and a reference ref/QPcard_201.cie file made from measuring a sample chart is also available.

For the QPCard 202, the ref/QPcard_202.cht file should be used, and a reference ref/QPcard_202.cie file made from measuring a sample chart is also available.

For any other type of chart, a chart recognition template file will need to be created (this is beyond the scope of the current documentation, although see the .cht_format documentation).

To create the scanner .ti3 file, run the scanin tool as follows (assuming an IT8 chart is being used):

scanin -v scanner.tif It8.cht It8ref.txt

"It8ref.txt" or "It8ref.cie" is assumed to be the name of the CIE reference file supplied by the chart manufacturer. The resulting file will be named "scanner.ti3".

scanin will process 16 bit per component .tiff files, which (if the scanner is capable of creating such files), may improve the quality of the profile.

If you have any doubts about the correctness of the chart recognition, or the subsequent profile's delta E report is unusual, then use the scanin diagnostic flags -dipn and examine the diag.tif diagnostic file, to make sure that the patches are identified and aligned correctly. If you have problems getting good automatic alignment, then consider doing a manual alignment by locating the fiducial marks on your scan, and feeding them into scanin -F parameters. The fiducial marks should be listed in a clockwise direction starting at the top left.

Creating a scanner or camera input profile Similar to a display profile, an input profile can be either a shaper/matrix or LUT based profile. Well behaved input devices will probably give the best results with a shaper/matrix profile, and this may also be the best choice if your test chart has a small or unevenly distributed set of test patchs (ie. the IT8.7.2). If a shaper/matrix profile is a poor fit, consider using a LUT type profile.

When creating a LUT type profile, there is the choice of XYZ or L*a*b* PCS (Device independent, Profile Connection Space). Often for input devices, it is better to choose the XYZ PCS, as this may be a better fit given that input devices are usually close to being linearly additive in behaviour.

If the purpose of the input profile is to use it as a substitute for a colorimeter, then the -ua flag should be used to force Absolute Colorimetric intent, and avoid clipping colors above the test chart white point. Unless the shaper/matrix type profile is a very good fit, it is probably advisable to use a LUT type profile in this situation.

To create a matrix/shaper profile, the following suffices:

colprof -v -D"Scanner A" -qm -as scanner

For an XYZ PCS LUT based profile then the following would be used:

colprof -v -D"Scanner A" -qm -ax scanner

For the purposes of a poor mans colorimeter, the following would generally be used:

colprof -v -D"Scanner A" -qm -ax -ua scanner

Make sure you check the delta E report at the end of the profile creation, to see if the sample data and profile is behaving reasonably. Depending on the type of device, and the consistency of the readings, average errors of 5 or less, and maximum errors of 15 or less would normally be expected. If errors are grossly higher than this, then this is an indication that something is seriously wrong with the device measurement, or profile creation.

If profiling a camera in RAW mode, then there may be some advantage in creating a pure matrix only profile, in which it is assumed that the camera response is completely linear. This may reduce extrapolation artefacts. If setting the white point will be done in some application, then it may also be an advantage to use the -u flag and avoid setting the white point to that of the profile chart:

colprof -v -D"Camera" -qm -am -u scanner

7.3 Profiling printers

The overall process is to create a set of device measurement target values, print them out, measure them, and then create an ICC profile from the measurements. If the printer is an RGB based printer, then the process is only slightly more complicated than profiling a display. If the printer is CMYK based, then some additional parameters are required to set the total ink limit (TAC) and black generation curve.

Creating a print profile test chart The first step in profiling any output device, is to create a set of device colorspace test values. The important parameters needed are:

- What colorspace does the device use ?
- How many test patches do I want to use/what paper size do I want to use ?
- What instrument am I going to use to read the patches ?
- If it is a CMYK device, what is the total ink limit ?
- What information do I already have about how the device behaves ?

Most printers running through simple drivers will appear as if they are RGB devices. In fact there is no such thing as a real RGB printer, since printers use white media and the colorant must subtract from the light reflected on it to create color, but the printer itself turns the incoming RGB into the native print colorspace, so for this reason we will tell targen to use the "Print RGB" colorspace, so that it knows that it's really a subtractive media. Other drivers will drive a printer more directly, and will expect a CMYK profile. [Currently Argyll is not capable of creating an ICC profile for devices with more colorants than CMYK. When this capability is introduced, it will by creating an additional separation profile which then allows the printer to be treated as a CMY or CMYK printer.] One way of telling what sort of profile is expected for your device is to examine an existing profile for that device using iccdump.

The number of test patches will depend somewhat on what quality profile you want to make, how well behaved the printer is, as well as the effort needed to read the

number of test values. Generally it is convenient to fill a certain paper size with the maximum number of test values that will fit.

At a minimum, for an ”RGB” device, a few hundred values are needed (400-1000). For high quality CMYK profiles, 1000-3000 is not an unreasonable number of patches.

To assist the determination of test patch values, it can help to have a rough idea of how the device behaves, so that the device test point locations can be pre-conditioned. This could be in the form of an ICC profile of a similar device, or a lower quality, or previous profile for that particular device. If one were going to make a very high quality Lut based profile, then it might be worthwhile to make up a smaller, preliminary shaper/matrix profile using a few hundred test points, before embarking on testing the device with several thousand.

The documentation for the targen tool lists a table of paper sizes and number of patches for typical situations.

For a CMYK device, a total ink limit usually needs to be specified. Sometimes a device will have a maximum total ink limit set by its manufacturer or operator, and some CMYK systems (such as chemical proofing systems) don’t have any limit. Typical printing devices such as Xerographic printers, inkjet printers and printing presses will have a limit. The exact procedure for determining an ink limit is outside the scope of this document, but one way of going about this might be to generate some small (say a few hundred patches) with targen pritntarg with different total ink limits, and printing them out, making the ink limit as large as possible without striking problems that are caused by too much ink.

Generally one wants to use the maximum possible amount of ink to maximize the gamut available on the device. For most CMYK devices, an ink limit between 200 and 400 is usual, but and ink limit of 250% or over is generally desirable for reasonably dense blacks and dark saturated colors. And ink limit of less than 200% will begin to compromise the fully saturated gamut, as secondary colors (ie combinations of any two primary colorants) will not be able to reach full strength.

Once an ink limit is used in printing the characterization test chart for a device, it becomes a critical parameter in knowing what the characterized gamut of the device is. If after printing the test chart, a greater ink limit were to be used, the the software would effectively be extrapolating the device behaviour at total ink levels beyond that used in the test chart, leading to inaccuracies.

Generally in Argyll, the ink limit is established when creating the test chart values, and then carried through the profile making process automatically. Once the profile has been made however, the ink limit is no longer recorded, and you, the user, will have to keep track of it if the ICC profile is used in any program than needs to know the usable gamut of the device.

Lets consider two devices in our examples, ”PrinterA” which is an ”RGB” device, and ”PrinterB” which is CMYK, and has a target ink limit of 250

The simplest approach is to make a set of test values that is independent of the characteristics of the particular device:

```
targen -v -d2 -f1053 PrinterA
targen -v -d4 -l260 -f1053 PrinterB
```

The number of patches chosen here happens to be right for an A4 paper size being read using a Spectroscan instrument. See the table in the targen documentation for some other suggested numbers.

If there is a preliminary or previous profile called ”OldPrinterA” available, and we want to try creating a ”pre-conditioned” set of test values that will more efficiently sample the device response, then the following would achieve this:

targen -v -d2 -f1053 -c OldPrinterA PrinterA

targen -v -d4 -l260 -f1053 -c OldPrinterB PrinterB

The output of targen will be the file PrinterA.ti1 and PrinterB.ti1 respectively, containing the device space test values, as well as expected CIE values used for chart recognition purposes.

Printing a print profile test chart The next step is turn the test values in to a PostScript or TIFF raster test file that can printed on the device. The basic information that needs to be supplied is the type of instrument that will be used to read the patches, as well as the paper size it is to be formatted for.

For an X-Rite DTP41, the following would be typical:

printtarg -v -i41 -pA4 PrinterA

For a Gretag Eye-One Pro, the following would be typical:

printtarg -v -ii1 -pA4 PrinterA

For using with a scanner as a colorimeter, the Gretag Spectroscan layout is suitable, but the -s flag should be used so as to generate a layout suitable for scan recognition, as well as generating the scan recognition template files. (You probably want to use less patches with targen, when using the printtarg -s flag, e.g. 1026 patches for an A4R page, etc.) The following would be typical:

printtarg -v -s -iSS -pA4R PrinterA

printtarg reads the PrinterA.ti1 file, creates a PrinterA.ti2 file containing the layout information as well as the device values and expected CIE values, as well as a PrinterA.ps file containing the test chart. If the -s flag is used, one or more PrinterA.cht files is created to allow the scanin program to recognize the chart.

To create TIFF raster files rather than PostScript, use the -t flag.

GSview is a good program to use to check what the PostScript file will look like, without actually printing it out. You could also use Photoshop or ImageMagick for this purpose.

The last step is to print the chart out.

Using a suitable PostScript or raster file printing program, downloader, print the chart. If you are not using a TIFF test chart, and you do not have a PostScript capable printer, then an interpreter like GhostScript or even Photoshop could be used to rasterize the file into something that can be printed. Note that it is important that the PostScript interpreter or TIFF printing application and printer configuration is setup for a device profiling run, and that any sort of color conversion of color correction be turned off so that the device values in the PostScript or TIFF file are sent directly to the device. If the device has a calibration system, then it would be usual to have setup and calibrated the device before starting the profiling run, and to apply calibration to the chart values. If Photoshop was to be used, then either the chart needs to be a single

page, or separate .eps or .tiff files for each page should be used, so that they can be converted and printed one at a time (see the -e and -t flags).

Reading a print test chart using an instrument Once the test chart has been printed, the color of the patches needs to be read using a suitable instrument.

Several different instruments are currently supported, some that need to be used patch by patch, some read a strip at a time, and some read a sheet at a time. See instruments for a current list.

The instrument needs to be connected to your computer before running the chartread command. Both serial port and USB connected Instruments are supported. A serial port to USB adapter might have to be used if your computer doesn't have any serial ports, and you have a serial interface connected instrument.

If you run chartread so as to print out its usage message (ie. by using a -? or –flags), then it will list any identified serial ports or USB connected instruments, and their corresponding number for the -c option. By default, chartread will try to connect to the first available USB instrument, or an instrument on the first serial port.

The only arguments required is to specify the basename of the .ti2 file. If a non-default serial port is to be used, then the -c option would also be specified.

e.g. for a Spectroscan on the second port:

chartread -c2 PrinterA

For a DTP41 to the default serial port:

chartread PrinterA

chartread will interactively prompt you through the process of reading each sheet or strip. See chartread for more details on the responses for each type of instrument. Continue with Creating a printer profile.

Reading a print test chart using a scanner or camera (NOT recommended) Argyll supports using a scanner or even a camera as a substitute for a colorimeter. Note though that this rarely gives good results. A scanner or camera is no replacement for a color measurement instrument.

The main problems of the scanner-as-colorimeter approach are:

- The spectral interaction of the scanner test chart and printer test chart with the scanner spectral response can cause color errors (i.e. a scanner or camera typically has quite different spectral sensitivities than a human observer).
- Spectral differences caused by different black amounts in the print test chart can cause color errors.
- The scanner reference chart gamut may be much smaller than the printers gamut, making the scanner profile too inaccurate to be useful.
- The scanner dynamic range and/or precision may not match the printers or what is required for a good profile.

As well as some of the above, a camera may not be suitable if it automatically adjusts exposure or white point when taking a picture, and this behavior cannot be disabled.

The end result is often a profile that has a noticeable color cast, compared to a profile created using a colorimeter or spectrometer.

It is assumed that you have created a scanner or camera profile following the procedure outline above. For best possible results it is advisable to both profile the scanner or camera, and use it in scanning the printed test chart, in as ”raw” mode as possible (i.e. using 16 bits per component images, if the scanner or camera is capable of doing so; not setting white or black points, using a fixed exposure etc.). It is generally advisable to create a LUT type input profile, and use the -ua flag to avoid clipping scanned value whiter than the input calibration chart.

Scan or photograph your printer chart (or charts) on the scanner or camera previously profiled. The scanner or camera must be configured and used exactly the same as it was when it was profiled.

I will assume the resulting scan/photo input file is called PrinterB.tif (or PrinterB1.tif, PrinterB2.tif etc. in the case of multiple charts). As with profiling the scanner or camera, the raster file need only be roughly cropped so as to contain the test chart.

The scanner recognition files created when printtarg was run is assumed to be called PrinterB.cht. Using the scanner profile created previously (assumed to be called scanner.icm), the printer test chart scan patches are converted to CIE values using the scanin tool:

scanin -v -c PrinterB.tif PrinterB.cht scanner.icm PrinterB

If there were multiple test chart pages, the results would be accumulated page by page using the -ca option, ie., if there were 3 pages:

scanin -v -c PrinterB1.tif PrinterB1.cht scanner.icm PrinterB scanin -v -ca PrinterB2.tif PrinterB2.cht scanner.icm PrinterB scanin -v -ca PrinterB3.tif PrinterB3.cht scanner.icm PrinterB

Now that the PrinterB.ti3 data has been obtained, the profile continue in the next section with Creating a printer profile.

If you have any doubts about the correctness of the chart recognition, or the subsequent profile’s delta E report is unusual, then use the scanin diagnostic flags -dipn and examine the diag.tif diagnostic file.

Creating a printer profile Creating an RGB based printing profile is very similar to creating a display device profile. For a CMYK printer, some additional information is needed to set the black generation.

Where the resulting profile will be used conventionally (ie. using collink -s, or cctiff or most other ”dumb” CMMs) it is important to specify that gamut mapping should be computed for the output (B2A) perceptual and saturation tables. This is done by specifying a device profile as the parameter to the colprof -S flag. When you intend to create a ”general use” profile, it can be a good technique to specify the source gamut as the opposite type of profile to that being created, i.e. if a printer profile is being created, specify a display profile (e.g. sRGB) as the source gamut. If a display profile is being created, then specify a printer profile as the source (e.g. Figra, SWOP etc.). When linking to the profile you have created this way as the output profile, then use perceptual intent if the source is the opposite type, and relative colorimetric if it is the same type.

”Opposite type of profile” refers to the native gamut of the device, and what its fundamental nature is, additive or subtractive. An emissive display will have additive primaries (R, G B), while a reflective print, will have subtractive primaries (C, M, Y possibly others), irrespective of what colorspace the printer is driven in (a printer might present an RGB interface, but internally this will be converted to CMY, and it will have a CMY type of gamut). Because of the complimentary nature of additive and subtractive device primary colorants, these types of devices have the most different gamuts, and hence need the most gamut mapping to convert from one colorspace to the other.

Note that specifying a very large gamut colorspace as the source gamut (i.e. ProPhoto etc.) is probably NOT what you want to do, since unless the source images have a similar very large gamut to that of the colorspace, they will end up getting over compressed and come out looking dull. Instead use a source profile that has a gamut more representative of the images gamut, or you should provide a gamut using the the -g parameter.

If you are creating a profile for a specific purpose, intending to link it to a specific input profile, then you will get the best results by specifying that source profile as the source gamut.

If a profile is only going to be used as an input profile, or is going to be used with a ”smart” CMM (e.g. collink -g or -G), then it can save considerable processing time and space if the -b flag is used, and the -S flag not used.

For an RGB printer intended to print RGB originals, the following might be a typical profile usage:

colprof -v -D”Printer A” -qm -S sRGB.icm -cmt -dpp PrinterA

or if you intent to print from Fogra, SWOP or other standard CMYK style originals:

colprof -v -D”Printer A” -qm -S fogra39l.icm -cmt -dpp PrinterA

If you know what colorspace your originals are in, use that as the argument to -S.

If your viewing environment for the display and print doesn’t match the ones implied by the -cmt and -dpp options, leave them out, and evaluate what, if any appearance transformation is appropriate for your environment at a later stage.

A fallback to using a specific source profile/gamut is to use a general compression percentage as a gamut mapping:

colprof -v -D”Printer A” -qm -S 20 -cmt -dpp PrinterA

Make sure you check the delta E report at the end of the profile creation, to see if the sample data and profile is behaving reasonably. Depending on the type of device, and the consistency of the readings, average errors of 5 or less, and maximum errors of 15 or less would normally be expected. If errors are grossly higher than this, then this is an indication that something is seriously wrong with the device measurement, or profile creation.

Choosing a black generation curve (and other CMYK printer options) For a CMYK printer, it would be normal to specify the type of black generation, either as something simple, or as a specific curve. The documentation in colprof for the details of the options.

Note that making a good choice of black generation curve can affect things such as: how robust neutrals are given printer drift or changes in viewing lighting, how visible screening is, and how smooth looking the B2A conversion is.

For instance, maximizing the level of K will mean that the neutral colors are composed of greater amounts of Black ink, and black ink retains its neutral appearance irrespective of printer behavior or the spectrum of the illuminant used to view the print. On the other hand, output which is dominantly from one of the color channels will tend to emphasize the screening pattern and any unevenness (banding etc.) of that channel, and the black channel in particular has the highest visibility. So in practice, some balance between the levels of the four channels is probably best, with more K if the screening is fine and a robust neutral balance is important, or less K if the screening is more visible and neutral balance is less critical. The levels of K at the edges of the gamut of the device will be fixed by the nature of the ink combinations that maximize the gamut (ie. typically zero ink for light chromatic colors, some combination for dark colors, and a high level of black for very dark near neutrals), and it is also usually important to set a curve that smoothly transitions to the K values at the gamut edges. Dramatic changes in K imply equally dramatic changes in CMY, and these abrupt transitions will reveal the limited precision and detail that can be captured in a lookup table based profile, often resulting in a ”bumpy” looking output.

If you want to experiment with the various black generation parameters, then it might be a good idea to create a preliminary profile (using -ql -b -no, -ni and no -S), and then used xicclu to explore the effect of the parameters.

For instance, say we have our CMYK .ti3 file PrinterB.ti3. First we make a preliminary profile called PrinterBt:

copy PrinterB.ti3 PrinterBt.ti3 (Use ”cp” on Linux or OSX of course.) colprof -v -qm -b -cmt -dpp PrinterBt

Then see what the minimum black level down the neutral axis can be. Note that we need to also set any ink limits we’ve decided on as well (coloprof defaulting to 10% less than the value recorded in the .ti3 file). In this example the test chart has a 300% total ink limit, and we’ve decided to use 290%:

xicclu -g -kz -l290 -fif -ir PrinterBt.icm

Which might be a graph something like this:

Note how the minimum black is zero up to 93% of the white-¿black L* curve, and then jumps up to 87%. This is because we’ve reached the total ink limit, and K then has to be substituted for CMY, to keep the total under the total ink limit.

Then let’s see what the maximum black level down the neutral axis can be:

xicclu -g -kx -l290 -fif -ir PrinterBt.icm

Which might be a graph something like this:

Note how the CMY values are fairly low up to 93% of the white-¿black L* curve (the low levels of CMY are helping set the neutral color), and then they jump up. This is because we’ve reach the point where black on it’s own, isn’t as dark as the color that can be achieved using CMY and K. Because the K has a dominant effect on the hue of the black, the levels of CMY are often fairly volatile in this region.

Any K curve we specify must lie between the black curves of the above two graphs.

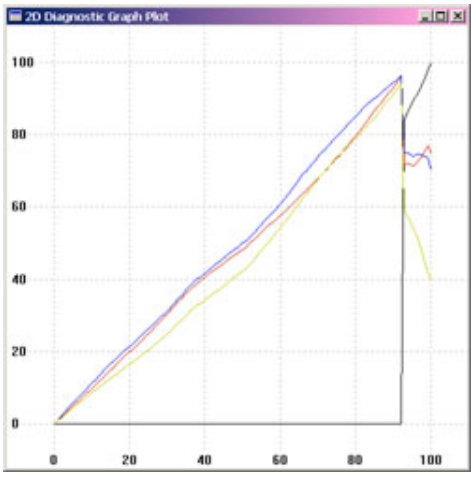

Fig. 14.

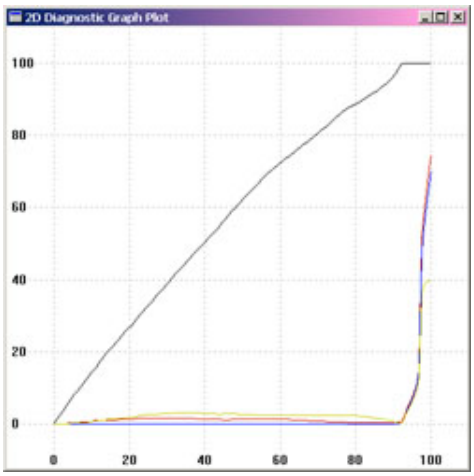

Fig. 15.

Let's say we'd like to chose a moderate black curve, one that aims for about equal levels of CMY and K. We should also aim for it to be fairly smooth, since this will minimize visual artefacts caused by the limited fidelity that profile LUT tables are able to represent inside the profile.

For minimum discontinuities we should aim for the curve to finish at the point it has to reach to satisfy the total ink limit at 87% curve and 93% black. For a first try we can simply set a straight line to that point:

xicclu -g -kp 0 0 .93 .87 1.0 -l290 -fif -ir PrinterBt.icm

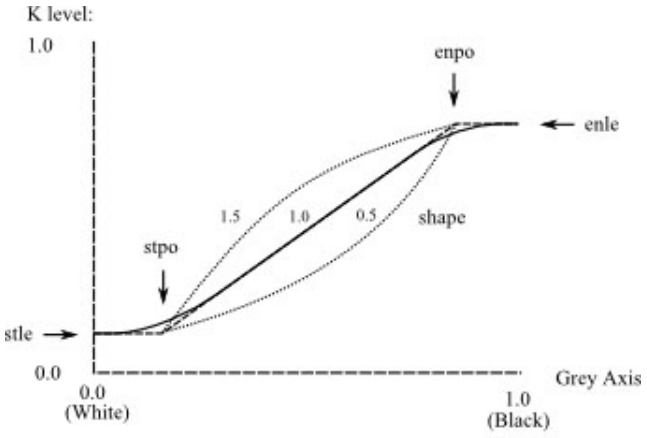

Fig. 16. -k parameters in order: stle, stpo, enpo, enle, shape

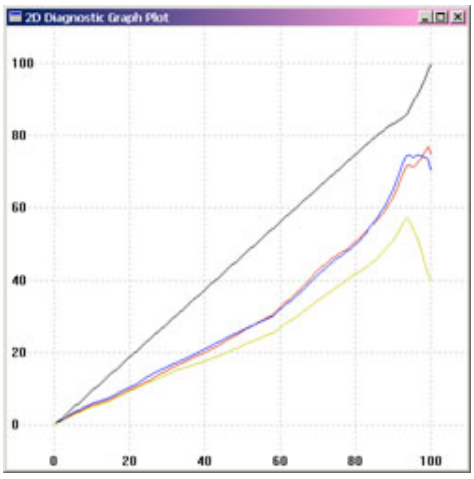

Fig. 17.

The black ”curve” hits the 93%/87% mark well, but is a bit too far above CMY, so we’ll try making the black curve concave:

xicclu -g -kp 0 0 .93 .87 0.65 -l290 -fif -ir PrinterBt.icm

This looks just about perfect, so the the curve parameters can now be used to generate our real profile:

colprof -v -D”Printer B” -qm -kp 0 0 .93 .87 0.65 -S sRGB.icm -cmt -dpp PrinterB

and the resulting B2A table black curve can be checked using xicclu:

xicclu -g -fb -ir PrinterB.icm

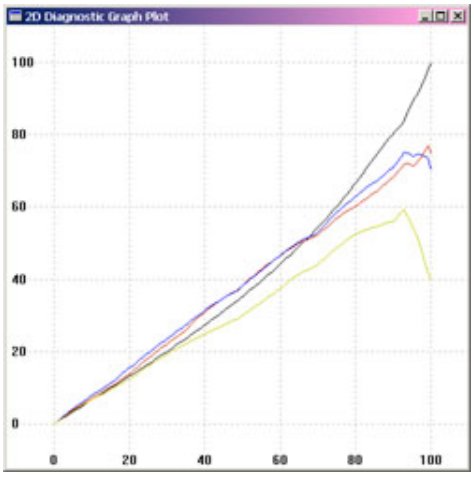

Fig. 18.

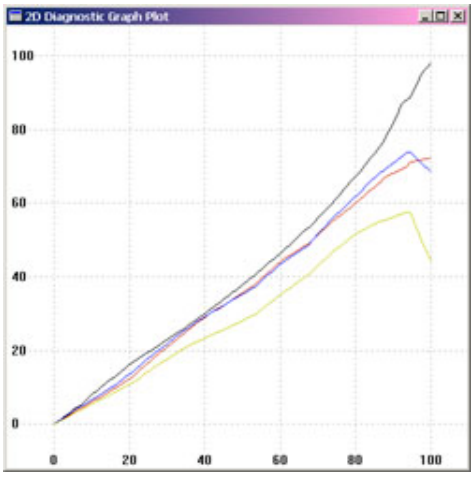

Fig. 19.

A smoothed zero black inking
xicclu -g -kp 0 .7 .93 .87 1.0 -l290 -fif -ir PrinterBt.icm

A low black inking
xicclu -g -kp 0 0 .93 .87 0.15 -l290 -fif -ir PrinterBt.icm

A high black inking
xicclu -g -kp 0 0 .93 .87 1.2 -l290 -fif -ir PrinterBt.icm

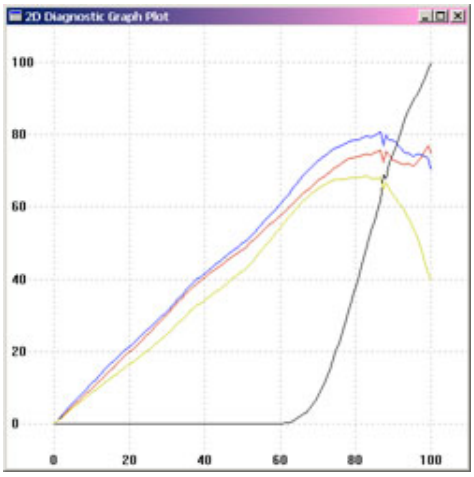

Fig. 20.

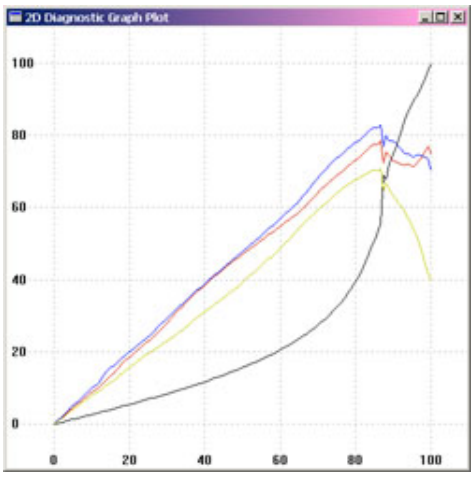

Fig. 21.

Overriding the ink limit Normally the total ink limit will be read from the PrinterB.ti3 file, and will be set at a level 10% lower than the number used in creating the test chart values using targen -l. If you want to override this with a lower limit, then use the -l flag.

colprof -v -D”Printer B” -qm -S sRGB.icm -cmt -dpp -kr -l290 PrinterB

Make sure you check the delta E report at the end of the profile creation, to see if the profile is behaving reasonably.

One way of checking that your ink limit is not too high, is to use ”xicc -fif -ia” to check, by setting different ink limits using the -l option, feeding Lab = 0 0 0 into it,

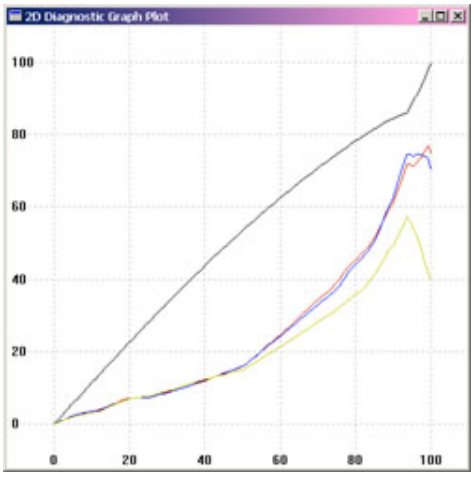

Fig. 22.

and checking the resulting black point. Starting with the ink limit used with targen for the test chart, reduce it until the black point starts to be affected. If it is immediately affected by any reduction in the ink limit, then the black point may be improved by increasing the ink limit used to generate the test chart and then re-print and re-measuring it, assuming other aspects such as wetness, smudging, spreading or drying time are not an issue.

7.4 Calibrating Printers

Profiling creates a description of how a device behaves, while calibration on the other hand is intended to change how a device behaves. Argyll has the ability to create per-channel device space calibration curves for print devices, that can then be used to improve the behavior of of the device, making a subsequent profile fit the device more easily and also allow day to day correction of device drift without resorting to a full re-profile.

NOTE: Because calibration adds yet another layer to the way color is processed, it is recommended that it not be attempted until the normal profiling workflow is established, understood and verified.

Calibrated print workflows There are two main workflows that printer calibration curves can be applied to:

Workflow with native calibration capability

Firstly the printer itself may have the capability of using per channel calibration curves. In this situation, the calibration process will be largely independent of profiling. Firstly the printer is configured to have both its color management and calibration disabled (the latter perhaps achieved by loading linear calibration curves), and a print calibration test chart that consists of per channel color wedges is printed. The calibration

chart is read and the resulting .ti3 file converted into calibration curves by processing it using printcal. The calibration is then installed into the printer. Subsequent profiling will be performed on the calibrated printer (ie. the profile test chart will have the calibration curves applied to it by the printer, and the resulting ICC profile will represent the behavior of the calibrated printer.)

Workflow without native calibration capability

The second workflow is one in which the printer has no calibration capability itself. In this situation, the calibration process will have to be applied using the ICC color management tools, so careful coordination with profiling is needed. Firstly the printer is configured to have its color management disabled, and a print calibration test chart that consists of per channel color wedges is printed. The calibration chart is converted into calibration curves by reading it and then processing the resultant .ti3 using printcal,. During the subsequent profiling, the calibration curves will need to be applied to the profile test chart in the process of using printtarg. Once the the profile has been created, then in subsequent printing the calibration curves will need to be applied to an image being printed either explicitly when using cctiff to apply color profiles and calibration, OR by creating a version of the profile that has had the calibration curves incorporated into it using the applycal tool. The latter is useful when some CMM (color management module) other than cctiff is being used.

Once calibration aim targets for a particular device and mode (screening, paper etc.) have been established, then the printer can be re-calibrated at any time to bring its per channel behavior back into line if it drifts, and the new calibration curves can be installed into the printer, or re-incorporated into the profile.

Creating a print calibration test chart The first step is to create a print calibration test chart. Since calibration only creates per-channel curves, only single channel step wedges are required for the chart. The main choice is the number of steps in each wedge. For simple fast calibrations perhaps as few as 20 steps per channel may be enough, but for a better quality of calibration something like 50 or more steps would be a better choice.

Let's consider two devices in our examples, "PrinterA" which is an "RGB" printer device, and "PrinterB" which is CMYK. In fact there is no such thing as a real RGB printer, since printers use white media and the colorant must subtract from the light reflected on it to create color, but the printer itself turns the incoming RGB into the native print colorspace, so for this reason we are careful to tell targen to use the "Print RGB" colorspace, so that it knows to create step wedges from media white to full colorant values.

For instance, to create a 50 steps per channel calibration test chart for our RGB and CMYK devices, the following would be sufficient:

targen -v -d2 -s50 -e3 -f0 PrinterA_c

targen -v -d4 -s50 -e4 -f0 PrinterB_c

For an outline of how to then print and read the resulting test chart, see Printing a print profile test chart, and Reading a print test chart using an instrument. Note that the printer must be in an un-profiled and un-calibrated mode when doing this print.

Having done this, there will be a PrinterA.ti3 or PrinterB.ti3 file containing the step wedge calibration chart readings.

NOTE that if you are calibrating a raw printer driver, and there is considerable dot gain, then you may want to use the -p parameter to adjust the test chart point distribution to spread them more evenly in perceptual space, giving more accurate control over the calibration. Typically this will be a value greater than one for a device that has dot gain, e.g. values of 1.5, 2.0 or 2.5 might be good places to start. You can do a preliminary calibration and use the verbose output of printcal to recommend a suitable value for -p.

Creating a printer calibration The printcal tool turns a calibration chart .ti3 file into a .cal file. It has three main operating modes:- Initial calibration, Re-Calibration, and Verification. (A fourth mode, "Imitation" is very like Initial Calibration, but is used for establishing a calibration target that a similar printer can attempt to imitate.)

The distinction between Initial Calibration and Re-Calibration is that in the initial calibration we establish the "aim points" or response we want out of the printer after calibration. There are three basic parameters to set this for each channel: Maximum level, minimum level, and curve shape.

By default the maximum level will be set using a heuristic which attempts to pick the point when there is diminishing returns for applying more colorant. This can be overridden using the -x percent option, where represents the choice of channel this will be applied to. The parameter is the percentage of device maximum.

The minimum level defaults to 0, but can be overridden using the -n deltaE option. A minimum of 0 means that zero colorant will correspond to the natural media color, but it may be desirable to set a non-pure media color using calibration for the purposes of emulating some other media. The parameter is in Delta E units.

The curve shape defaults to being perceptually uniform, which means that even steps of calibrated device value result in perceptually even color steps. In some situations it may be desirable to alter this curve (for instance when non color managed output needs to be sent to the calibrated printer), and a simple curve shape target can be set using the -t percent parameter. This affects the output value at 50% input value, and represents the percentage of perceptual output. By default it is 50% perceptual output for 50% device input.

Once a device has been calibrated, it can be re-calibrated to the same aim target.

Verification uses a calibration test chart printed through the calibration, and compares the achieved response to the aim target.

The simplest possible way of creating the PrinterA.cal file is:

printcal -i PrinterA_c

For more detailed information, you can add the -v and -p flags:

printcal -v -p -i PrinterB_c

(You will need to select the plot window and hit a key to advance past each plot).

For re-calibration, the name of the previous calibration file will need to be supplied, and a new calibration file will be created:

printcal -v -p -r PrinterB_c_old PrinterB_c_new

Various aim points are normally set automatically by printcal, but these can be overridden using the -x, -n and -t options. e.g. say we wanted to set the maximum ink for Cyan to 80% and Black to 95%, we might use:

printcal -v -p -i -xc 80 -xk 95 PrinterB_c

Using a printer calibration The resulting calibration curves can be used with the following other Argyll tools:
`printtarg` To apply calibration to a profile test chart, and/or to have it included in .ti3 file.
`cctiff` To apply color management and calibration to an image file.
`applycal` To incorporate calibration into an ICC profile.
`chartread` To override the calibration assumed when reading a profile chart.

In a workflow with native calibration capability, the calibration curves would be used with printarg during subsequent profiling so that any ink limit calculations will reflect final device values, while not otherwise using the calibration within the ICC workflow:

printtarg -v -ii1 -pA4 -I PrinterA_c.cal PrinterA

This will cause the .ti2 and resulting .ti3 and ICC profiles to contain the calibration curves, allowing all the tools to be able to compute final device value ink limits. The calibration curves must also of course be installed into the printer. The means to do this is currently outside the scope of Argyll (ie. either the print system needs to be able to understand Argyll CAL format files, or some tool will be needed to convert Argyll CAL files into the printer calibration format).

In a workflow without native calibration capability, the calibration curves would be used with printarg to apply the calibration to the test patch samples during subsequent profiling, as well as embedding it in the resulting .ti3 to allow all the tools to be able to compute final device value ink limits:

printtarg -v -ii1 -pA4 -K PrinterA_c.cal PrinterA

To apply calibration to an ICC profile, so that a calibration unaware CMM can be used:

applycal PrinterA.cal PrinterA.icm PrinterA_cal.icm

To apply color management and calibration to a raster image:

cctiff Source.icm PrinterA.icm PrinterA_c.cal infile.tif outfile.tif

or

cctiff Source.icm PrinterA_c.icm infile.tif outfile.tif

[Note that cctiff will also process JPEG raster images.]

Another useful tool is synthcal, that allows creating linear or synthetic calibration files for disabling calibration or testing. Similarly, fakeread also supports applying calibration curves and embedding them in the resulting .ti3 file

If you want to create a pre-conditioning profile for use with targen -c, then use the PrinterA.icm profile, NOT PrinterA_c.icm that has calibration curves applied.

How profile ink limits are handled when calibration is being used Even though the profiling process is carried out on top of the linearized device, and the profiling is generally unaware of the underlying non-linearized device values, an exception is made in the calculation of ink limits during profiling. This is made possible by including the calibration curves in the profile charts .ti2 and subsequent .ti3 file and resulting ICC profile 'targ' text tag, by way of the printtarg -I or -K options. This is done on the assumption that the physical quantity of ink is what's important in setting the ink limit, and that the underlying non-linearized device values represent such a physical quantity.

7.5 Linking Profiles

Two device profiles can be linked together to create a device link profile, than encapsulates a particular device to device transform. Often this step is not necessary, as many systems and tools will link two device profiles "on the fly", but creating a device link profile gives you the option of using "smart CMM" techniques, such as true gamut mapping, improved inverse transform accuracy, tailored black generation and ink limiting.

The overall process is to link the input space and output space profiles using collink, creating a device to device link profile. The device to device link profile can then be used by cctiff (or other ICC device profile capable tools), to color correct a raster files.

Three examples will be given here, showing the three different modes than collink supports.

In simple mode, the two profiles are linked together in a similar fashion to other CMMs simply using the forward and backwards color transforms defined by the profiles. Any gamut mapping is determined by the content of the tables within the two profiles, together with the particular intent chosen. Typically the same intent will be used for both the source and destination profile:

collink -v -qm -s -ip -op SouceProfile.icm DestinationProfile.icm Source2Destination.icm

In gamut mapping mode, the pre-computed intent mappings inside the profiles are not used, but instead the gamut mapping between source and destination is tailored to the specific gamuts of the two profiles, and the intent parameter supplied to collink. Additionally, source and destination viewing conditions should be provided, to allow the color appearance space conversion to work as intended. The colorimetric B2A table in the destination profile is used, and this will determine any black generation and ink limiting:

collink -v -qm -g -ip -cmt -dpp MonitorSouceProfile.icm DestinationProfile.icm Source2Destination.icm

[If your viewing environment for the display and print doesn't match the ones implied by the -cmt and -dpp options, leave them out, and evaluate what, if any appearance transformation is appropriate for your environment at a later stage.]

In inverse output table gamut mapping mode, the pre-computed intent mappings inside the profiles are not used, but instead the gamut mapping between source and destination is tailored to the specific gamuts of the two profiles, and the intent parameter supplied to collink. In addition, the B2A table is not used in the destination profile,

but the A2B table is instead inverted, leading to improved transform accuracy, and in CMYK devices, allowing the ink limiting and black generation parameters to be set:

For a CLUT table based RGB printer destination profile, the following would be appropriate:

collink -v -qm -G -ip -cmt -dpp MonitorSouceProfile.icm RGBDestinationProfile.icm Source2Destination.icm

For a CMYK profile, the total ink limit needs to be specified (a typical value being 10% less than the value used in creating the device test chart), and the type of black generation also needs to be specified:

collink -v -qm -G -ip -cmt -dpp -l250 -kr MonitorSouceProfile.icm CMYKDestinationProfile.icm Source2Destination.icm

Note that you should set the source (-c) and destination (-d) viewing conditions for the type of device the profile represents, and the conditions under which it will be viewed.

7.6 Image dependent gamut mapping using device links

When images are stored in large gamut colorspaces (such as. L*a*b*, ProPhoto, scRGB etc.), then using the colorspace gamut as the source gamut for gamut mapping is generally a bad idea, as it leads to overly compressed and dull images. The correct approach is to use a source gamut that represents the gamut of the images themselves. This can be created using tiffgamut, and an example workflow is as follows:

tiffgamut -f80 -pj -cmt ProPhoto.icm image.tif

collink -v -qh -G image.gam -ip -cmt -dpp ProPhoto.icm RGBDestinationProfile.icm Source2Destination.icm

cctiff Source2Destination.icm image.tif printfile.tif

The printfile.tif is then send to the printer without color management, (i.e. in the same way the printer characterization test chart was printed), since it is in the printers native colorspace.

You can adjust how conservatively the image gamut is preserved using the tiffgamut -f parameter. Omitting it or using a larger value (up to 100) preserves the color gradations of even the lesser used colors, at the cost of compressing the gamut more. Using a smaller value will preserve the saturation of the most popular colors, at the cost of not preserving the color gradations of less popular colors.

You can create a gamut that covers a set of source images by providing more than one image file name to tiffgamut. This may be more efficient for a group of related images, and ensures that colors are transformed in exactly the same way for all of the images.

An alternative generating a gamut for a specific set of images, is to use a general smaller gamut definition (i.e. the sRGB profile), or a gamut that represents the typical range of colors you wish to preserve.

The arguments to collink should be appropriate for the output device type - see the collink examples in the above section.

7.7 Soft Proofing Link

Often it is desirable to get an idea what a particular devices output will look like using a different device. Typically this might be trying to evaluate print output using a display. Often it is sufficient to use an absolute or relative colorimetric transform from the print device space to the display space, but while these provide a colorimetric preview of the result, they do not take into account the subjective appearance differences due to the different device conditions. It can therefore be useful to create a soft proof appearance transform using collink:

collink -v -qm -G -ila -cpp -dmt -t250 CMYKDestinationProfile.icm MonitorProfile.icm SoftProof.icm

We use the Luminance matched appearance intent, to preserve the subjective apperance of the target device, which takes into account the viewing conditions and assumes adaptation to the differences in the luminence range, but otherwise not attempting to compress or change the gamut.

If your viewing environment for the display and print doesn't match the ones implied by the -cpp and -dmt options, then either leave them out or substitute values that do match your environment.

7.8 Transforming colorspaces of raster files

Although a device profile or device link profile may be useful with other programs and systems, Argyll provides the tool cctiff for directly applying a device to device transform to a TIFF or JPEG raster file. The cctiff tool is capable of linking an arbitrary sequence of device profiles, device links, abstract profiles and calibration curves. Each device profile can be preceded by the -i option to indicate the intent that should be used. Both 8 and 16 bit per component files can be handled, and up to 8 color channels. The color transform is optimized to perform the overall transformation rapidly.

If a device link is to be used, the following is a typical example:

cctiff Source2Destination.icm infile.tif outfile.tif or cctiff Source2Destination.icm infile.jpg outfile.jpg

If a source and destination profile are to be used, the following would be a typical example:

cctiff -ip SourceProfile.icm -ip DestinationProfile.icm infile.tif outfile.tif or cctiff -ip SourceProfile.icm -ip DestinationProfile.icm infile.jpg outfile.jpg

Summary of the workflow

Russel Cotrell from LittleArgyll GUI has beautifully summed up the workflow for color managing a desktop printer in 23. The highlighted files are saved for future use.

The files printer.old.icm and printer_c.old.cal are the previous instances of these files, renamed so updated versions can be created.

8 LittleArgyllGUI

The LittleArgyll GUI has been developed by Russel Cotrell. It is a Graphical User Interface for Argyll CMS users not comfortable with the command driven usage of the

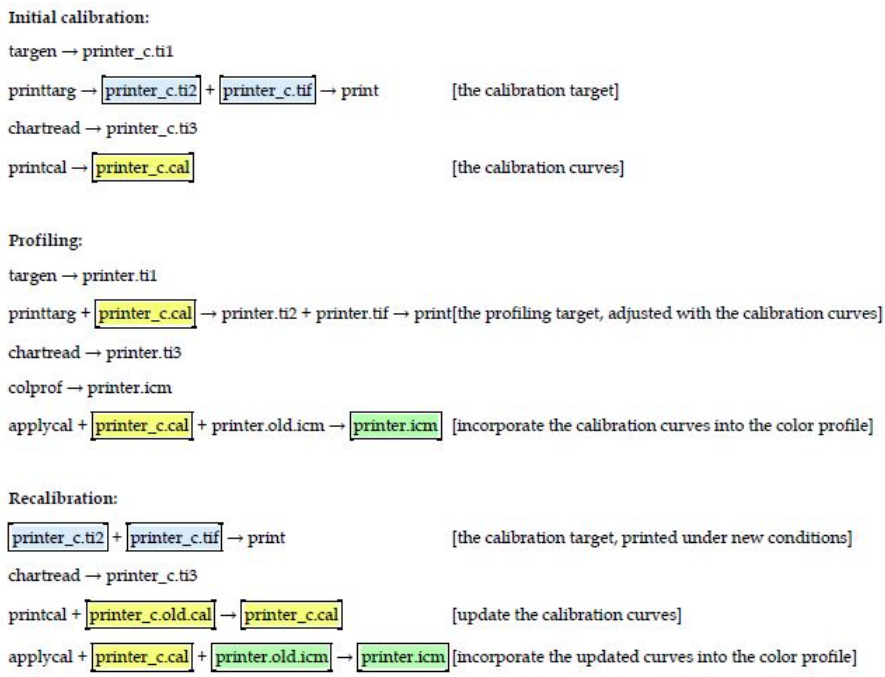

Fig. 23. A Desktop Printer Calibration Workflow with ArgyllCMS

Argyll. The Little Argyll GUI uses open and save dialogs and other graphic elements to simplify the creation and use of the ArgyllCMS command line. It also saves the selected parameters and command lines under the working file name for future use.

The program is cross-platform, running under Windows, Linux, and Mac OS X. (The latter has a number of issues; see below.)

The program was written for a monitor and desktop RGB printer workflow. Not all options are included in the GUI elements; but the command line fields are editable, and any of the applicable flags and options may be typed in manually and saved.

8.1 Installation

The LittleArgyllGUI binary should be in its own folder together with its doc folder.

Install ArgyllCMS.

Familiarize yourself with the ArgyllCMS documentation, particularly Scenarios.html and the help documents for the individual programs.

8.2 Setup

On the Setup tab, select the path to the ArgyllCMS/bin folder, unless (1) the ArgyllCMS binaries are installed in a location already specified by the PATH environment variable,

such as the Linux /bin directory; or (2) you have manually added ArgyllCMS/bin to PATH.

Linux only: enter the name of your preferred terminal emulator, such as mate-terminal, gnome-terminal, xfce4-terminal, konsole, etc.

Adjust the font height if necessary. (Linux distros have different default font sizes, for example.)

These parameters, the current working folder, and other information about the program such as the window size, are saved to an .ini file with the program binary, and will be reloaded every time it opens.

8.3 The working folder and base name

Create a working folder to contain the project files such as the patch set information, calibration files, and the resulting color profile.

In the program, select the above folder and enter the base name for the project, such as MyPrinter, as the file name. When returning to the project, you may select any file with the base name, such as MyPrinter.ini. (It is not a good idea to use spaces in the folder or file names, but they will work because all such user-provided information is enclosed in quotation marks. The latter are actually necessary for the operation of the overwrite warnings.)

Once you have selected the working folder, you may Alt-Click or Command-Click the save file icon to open it.

All the rest of the parameters and command lines will be saved to an .ini file with the base name in the working folder, and may be reloaded by selecting it as above.

To use similar command lines in a new project, copy the .ini file to a new working folder. You may rename the .ini if desired; you will just have to regenerate the command lines with the new base name.

8.4 Usage

The general flow parallels the Monitor calibration, Monitor profiling, and Printer profiling tabs. I placed Printer calibration last, partly because I am ambivalent about the concept, and partly because the first steps are the same as in profiling.

Follow the scenarios (Scenarios.html) and individual program help files in the ArgyllCMS documentation. The program's form fields are accompanied by the relevant command line flags for easy reference. Examples are provided for some of the individual programs. In general, select or enter the relevant parameters (some are optional; others may not be visible depending on other options), press Generate, edit the command line if necessary, then press Run. The terminal will open and run as usual.

The resulting files such as the patch set information and resulting color profile will all be created in the working folder.

The reversed color fields are the actual command lines, and may be copied (click the copy icon) and pasted directly into the terminal if desired.

I would also like implore the readers to go through the RGB to XYZ color space converters that Russell has made here [93]. The data and the formulae has been referred from Bruce Lindbloom's portal.

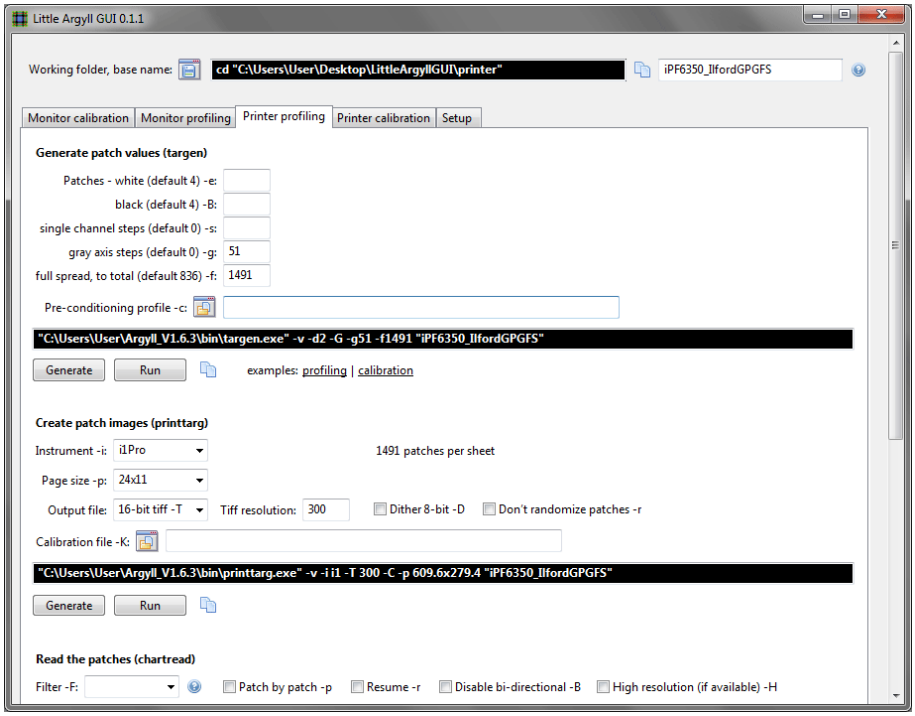

Fig. 24. LittleArgyll GUI

9 Profiling using CoCa

CoCa is an open source, mainly MS Windows appplication, designed to create ICC color profiles for digital capture devices, like scanners and cameras. It utilises Argyll open source color library, which is included in CoCa installation package. CoCa supports a variety of calibration targets, including:
Xrite ColorChecker Classic (24 patches) - www.xrite.com
Xrite ColorChecker Digital SG (140 patches) - www.xrite.com
Xrite ColorChecker Passport - www.xrite.com
Christophe Métairie's Checker - www.cmp-color.fr
Christophe Métairie's Digital TargeT 003 - www.cmp-color.fr
Christophe Métairie's Digital Target-3 - www.cmp-color.fr
Christophe Métairie's Digital Target-4 - www.cmp-color.fr
HutchColor HCT - www.hutchcolor.com/hct.htm
LaserSoft DCPro - www.silverfast.com/show/dc-targets/en.html
IT8.7 - available from many vendors. Inexpensive targets are available from www.targets.coloraid.de or www.hugorodriguez.com
QPcard 201 - www.qpcard.se
QPcard 202 - www.qpcard.se
SpyderCHECKR - www.datacolor.com

SpyderCHECKR 24 - www.datacolor.com

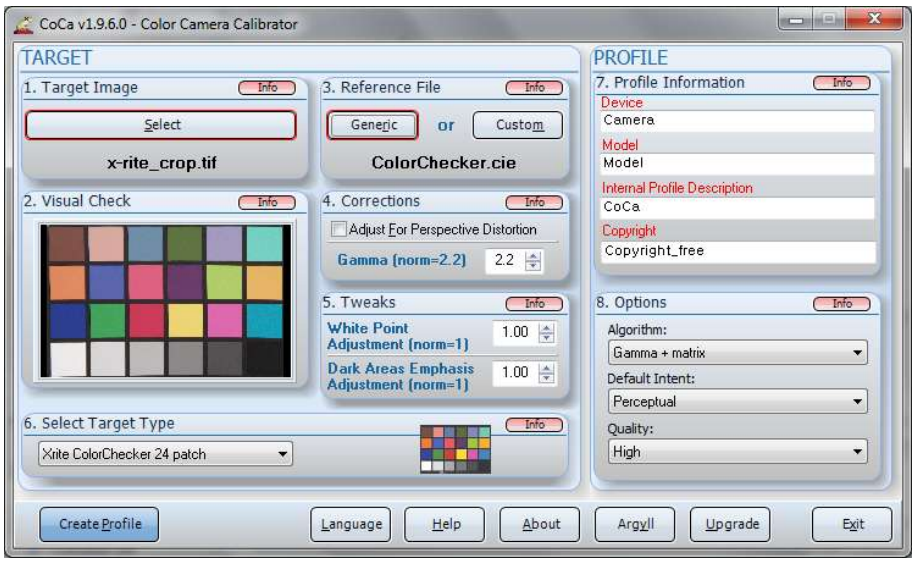

Fig. 25. CoCa GUI

For creation of profiles, all targets require the accompanying reference file. Usually, the reference file is supplied together with the target by the vendor. For ColorChecker, QPcard 201 and QPCard 202, generic reference files may be used and are included within CoCa. QPCard 203 has been included as well, working on the assumption that it has the same values as 202 (but this has not been confirmed yet). Some other reference files are available for download from vendor's web sites.

9.1 Usage

CoCa is simple to use and to help with the ICC profile creation process it provides numbered steps, that follow the logical path of building up a profile. CoCa has not been tested with all supported targets but it works very well with many of them.

Described below are the program functions.

Target Image
To create an ICC profile, special colour target is required (from vendors mentioned elsewhere on this page) that needs to be captured with a digital camera or scanner. To ensure high quality capture, suitable for colour calibration please observe the following:

- All automatic functions in the capture device and/or software should be disabled (e.g. White Balance, Exposure).
- Image should be saved (or converted to, in a standard way) as an uncompressed TIFF, 8-bit or 16-bit per channel RGB.
- Check the histogram of the target image for clipping using an image editor (like Adobe Photoshop). The scan should not reach an R, G or B value of 0 (for darkest tones) or 255 (for lightest tones). For 16-bit images it should not reach 0 or 65535. If it does, try changing the exposure setting of the camera or in the scanner driver. Usualy the range of 10-245 (about 2600-63000 in 16-bit mode) should be suitable for producing colour profiles. This range is ideal with the coloraid.de IT8 targets (for digital scanners). IT8 targets from some manufacturers often have a smaller density range and thus might need an even smaller RGB range in order to avoid clipping in actual image scans. There may be some experimentation required for the best capture of other targets.
- Sometimes the color quality can be improved by gamma correcting the image in the software before making/applying the profile. If your target image is extremly dark with hardly any detail visible in the shadows, try changing the camera exposure (setting of the scanner driver) and if that fails, try increasing the gamma correction until shadows appear decently normal. This hint is especialy recommended to users that require a gamma correction larger than 4 for shadows to appear normal in brightness. The gamma correction must than be applied to all captured images before the profile is applied.

CoCa allows you to use an already captured target image (click on ”Select” button) or to capture a new image (”Capture”), providing that the capturing device supports a TWAIN driver.

Visual Check
In this section, it is possible to visually check the positioning and cropping of the captured image. Clicking on the image would diplay a larger preview with a rough histogram window.

Reference File
Reference file is necessary for creation of a colour profile. Usually, they are provided by a vendor together with a target. Xrite ColorChecker and QPcard 201 may be used with generic references that are provided with CoCa. To use generic files, click on the ”Generic” button. However, higher quality profiles are usually created with references that were produced specifically for the particular target. Click ”Custom” to open and use such files.

Corrections
If your captured target image has a perspective distortion, this option would try to automatically correct it. Default Gamma setting is 2.2. It should not be changed for standard 24-bit images. If the image has linear Gamma (e.g. RAW files converted with dcraw linear option 16-bit/channel), Gamma 1.0 should be selected.

Tweaks
By using the slider in this section, it is possible to adjust the White Point of the profile. Values higher than 1.0 will create a profile that produces lighter images, and values below 1.0 will produce darker images.

Select Target Type
It is important to tell CoCa what type of target it should expect as it cannot detect it automatically from the image. If the wrong type is selected, CoCa either will be unable to create a profile or the resulting profile will be invalid. Small image next to the selection list will help you to determine the right target type.

Profile Information
This is user dependent information that needs to be correctly adjusted. In particular, ”Internal Profile Description” field is important as some programs (like Adobe Photoshop) use it to identify the profile.

Options - Algorithm
Here it is possible to select appropriate algorithm to calculate the profile. Some experimentation may be required when trying to determine the best algorithm for a particular device/target. According to some colour management practitioners, ”Gamma+matrix” is the most suitable for many digital cameras.

Options - Quality Various profile quality levels can be selected in this section. Higher quality would result in a better profile but the profile creation will be longer and the profile itself will be larger (bigger file size). The Argyll documentation specifies that ”Ultra High” quality should almost never be used.

9.2 Creating the Profile

Click the ”Create Profile” button. Providing that all options were correctly selected, CoCa would start calculating the profile. Current progress will be visible inside (usually) black command-line boxes. After profile has been created, CoCa will ask you to save it.

9.3 Using the Profile

To use the created profile, the scanner’s capture software would need to have an option of selecting the default scanning ICC profile. Once selected, the profile would be automatically applied during every capture. You would need to select the created profile’s name in the device’s software.

With scanners and cameras

Alternatively, after images have been captured, they need to be opened in an image manipulation program like Adobe Photoshop, GIMP or Paint Shop Pro.Use the ”Assign Profile” option to apply the profile.

Remember, an ICC colour profile is valid only for the device that it has been created for. In case of digital cameras, the lighting has to be always the same as it was when the profile was created.

ICC profiles are not perfect (in some cases when the original material you are capturing is very different to the material used in production of a target, the results may be quite inaccurate), but at the moment they are the best mechanism available to ensure high fidelity colour in digital images.

10 PyCharm Documentation

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated together to create a convenient environment for productive Python, web, and data science development.

PyCharm is available in three editions: Professional, Community, and Educational (Edu). The Community and Edu editions are open-source projects and they are free, but they have less features. PyCharm Edu provides courses and helps you learn programming with Python.

Supported languages

To start developing in Python with PyCharm you need to download and install Python from python.org depending on your platform.

PyCharm supports the following versions of Python:
Python 2: version 2.7

Python 3: from the version 3.5 up to the version 3.8

In addition, in the Professional edition, one can develop Django, Flask and Pyramid applications. Also, it fully supports HTML (including HTML5), CSS, JavaScript, and XML: these languages are bundled in the IDE via plugins and are switched on for you by default. Support for the other languages and frameworks can also be added via plugins (go to Settings — Plugins or PyCharm — Preferences — Plugins for macOS users, to find out more or set them up during the first IDE launch).

10.1 Standalone Installation of PyCharm on Windows OS

- Download the installer here [94]. To verify the integrity of the installer, use the SHA checksum linked from the Download page.
- Run the installer and follow the wizard steps. Mind the following options in the installation wizard.

  - 64-bit launcher: Adds a launching icon to the Desktop.
  - Open Folder as Project: Adds an option to the folder context menu that will allow opening the selected directory as a PyCharm project.
  - .py: Establishes an association with Python files to open them in PyCharm.

- Add launchers dir to the PATH: Allows running this PyCharm instance from the Console without specifying the path to it.

– When you run PyCharm for the first time, or after you have upgraded it from a previous version, some steps are required to complete the installation, customize your instance and start working with the IDE.

10.2 Open/Create a project in PyCharm

Why do I need a project? Everything you do in PyCharm is done within the context of a project. It serves as a basis for coding assistance, bulk refactoring, coding style consistency, and so on.

You have three options to start working on a project inside the IDE:

Open an existing project Begin by opening one of your existing projects stored on your computer. You can select one in the list of the recent projects on the Welcome screen or click Open as in Fig.26 Otherwise, you can create a project for your existing

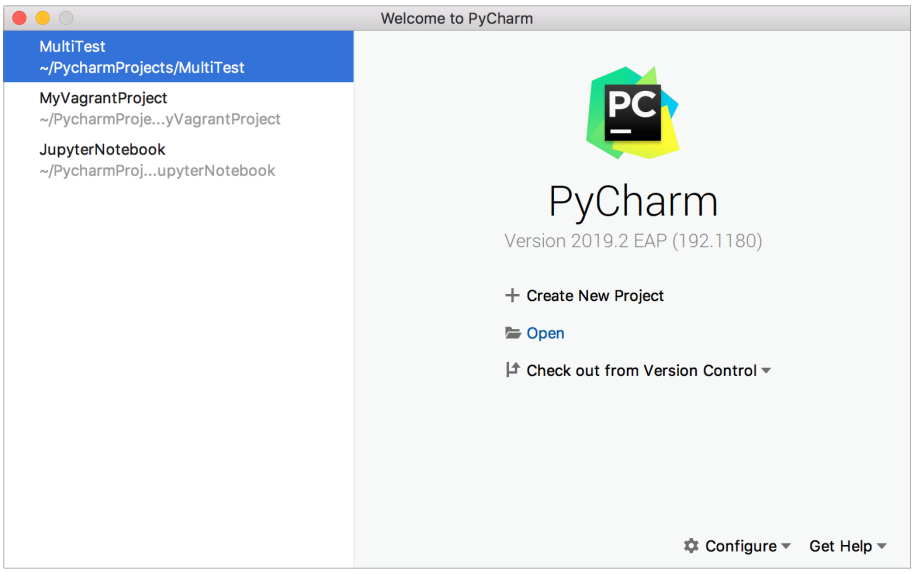

Fig. 26.

source files. Select the command Open on the File menu, and specify the directory where the sources exist. PyCharm will then create a project from your sources for you. Refer to the section Importing Project from Existing Source Code for details.

Check out an existing project from Version Control
You can also download sources from a VCS storage or repository. Choose Git (GitHub), Mercurial, Subversion, Perforce (supported in Professional edition only), and then enter your credentials to access the storage.

Then, enter a path to the sources and clone the repository to the local host as in Fig.27.

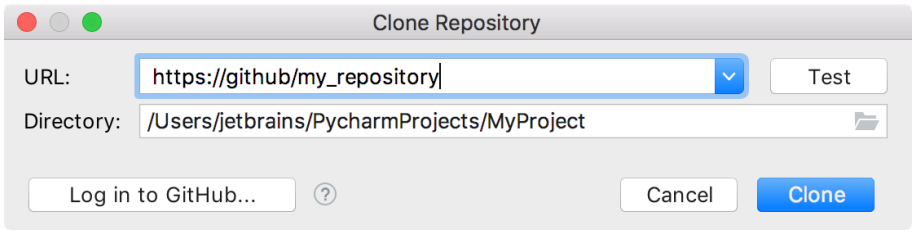

Fig. 27.

Create a project from scratch

- To create a project, do one of the following:
  - From the main menu, choose File — New Project
  - On the Welcome screen, click Create New Project

  New Project dialog opens.
- In the New Project dialog as in Fig. 28, specify the project name and its location. The dialog may differ depending on the PyCharm edition.
- Next, click Expand the node to expand the Project Interpreter node, and select the new environment or existing interpreter, by clicking the corresponding radio-button. The following steps depend on your choice:
  - New environment using: if this option has been selected, choose the tool to be used to create a virtual environment. To do that, click the list and choose Virtualenv, Pipenv, or Conda.
    Next, specify the location and base interpreter of the new virtual environment. If necessary, click the Inherit global site-packages and Make available to all projects check boxes.
    When you configure a project Python interpreter, you need to specify the path to the Python executable in your system. So, before configuring a project interpreter, you need to ensure that you've downloaded Python and installed it in your system and you're aware of a path to it. You can create several project interpreters based on the same Python executable. This is helpful when you need to create different virtual environments for developing different types of applications. For example, you can create one virtual environment based on Python 3.6 to develop Django applications and another virtual environment based on the same Python 3.6 to work with scientific libraries.
  - Existing interpreter: if this option has been selected, choose the desired interpreter from the list, or (if the desired interpreter is not found), click Open and choose the interpreter. See Configure a Python interpreter for details.

– The new project will be created as in Fig. 29. By this time, the editor area is grey as you don't have any files in your project. Create a Python file to start coding your script.

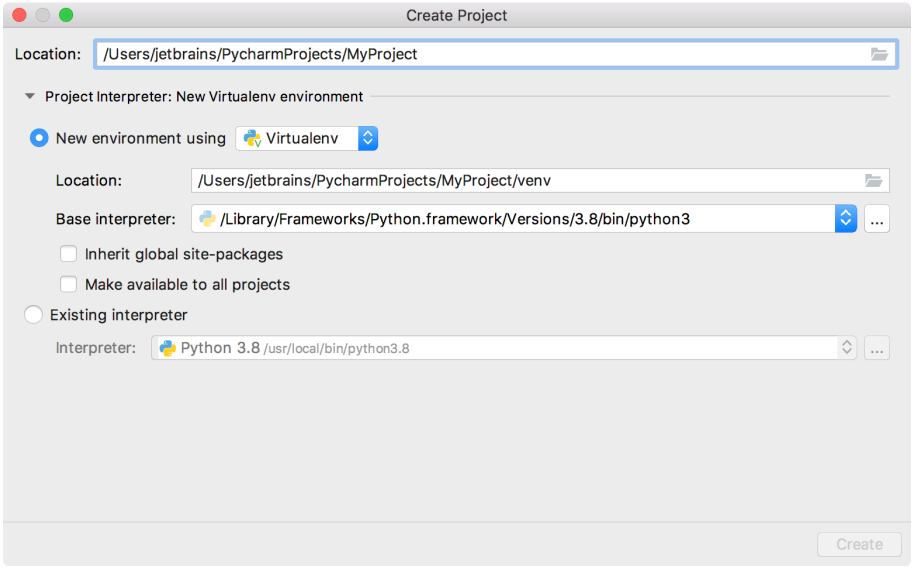

Fig. 28.

Creating a Python file

Select the project root in the Project tool window, then select File — New ... from the main menu or press Alt+Insert. The choose the option Python file from the popup, and then type the new filename.

PyCharm creates a new Python file and opens it for editing as in Fig. 30.

Configure project interpreter

When creating a new project you need to add and configure a project interpreter.

When you configure a project Python interpreter, you need to specify the path to the Python executable in your system. So, before configuring a project interpreter, you need to ensure that you've downloaded Python and installed it in your system and you're aware of a path to it. You can create several project interpreters based on the same Python executable. This is helpful when you need to create different virtual environments for developing different types of applications. For example, you can create one virtual environment based on Python 3.6 to develop Django applications and another virtual environment based on the same Python 3.6 to work with scientific libraries.

In the Settings/Preferences dialog Ctrl+Alt+S, click "Settings" in Project Interpreter, then click Add project interpreter, and choose Add .... Depending on your project specifics you can select:

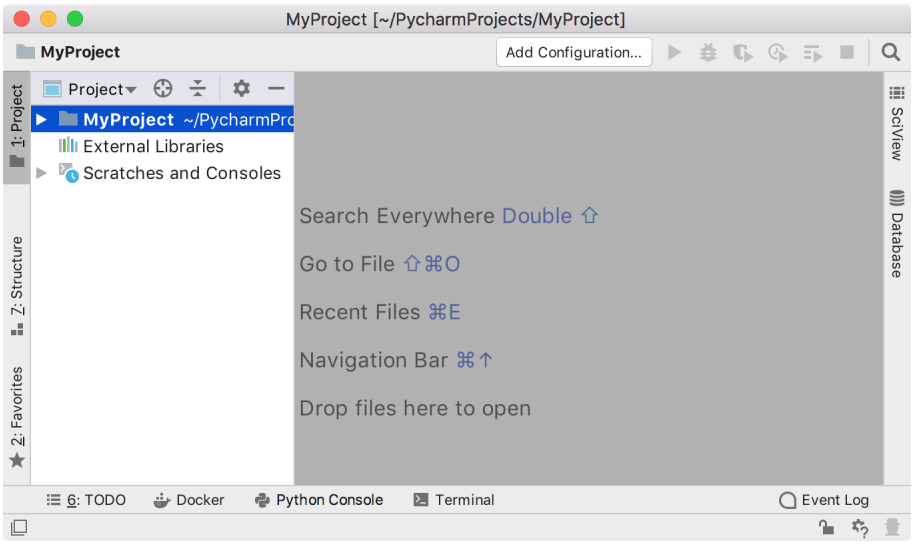

Fig. 29.

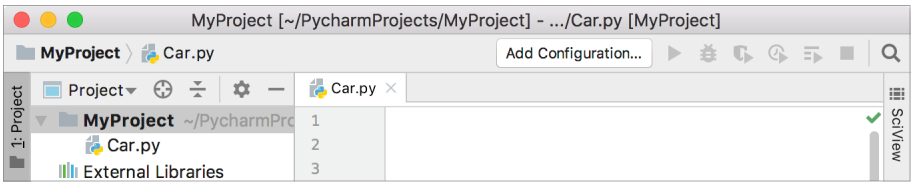

Fig. 30.

- System Interpreter
- Virtualenv Environments: Virtualenv, Pipenv, and Conda.

Look around When you launch PyCharm for the very first time, or when there are no open projects, you see the Welcome screen. It gives you the main entry points into the IDE: creating or opening a project, checking out a project from version control, viewing documentation, and configuring the IDE.

When a project is opened, you see the main window divided into several logical areas. Let's take a moment to see the key UI elements here as in Fig. 31

- Project tool window on the left side displays your project files.
- Editor on the right side, where you actually write your code. It has tabs for easy navigation between open files.
- Navigation bar above the editor additionally allows you to quickly run and debug your application as well as do the basic VCS actions.

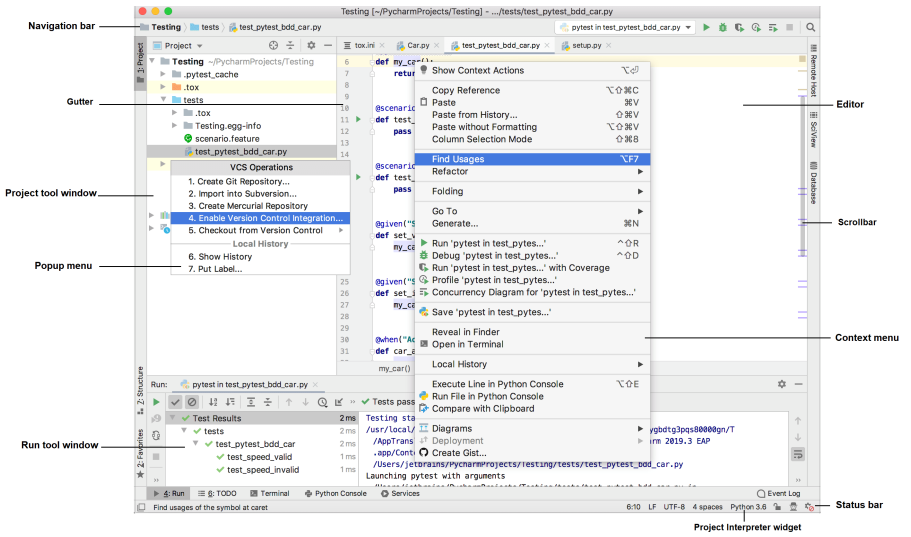

Fig. 31.

- Gutter, the vertical stripe next to the editor, shows the breakpoints you have, and provides a convenient way to navigate through the code hierarchy like going to definition/declaration. It also shows line numbers and per-line VCS history.
- Scrollbar, on the right side of the editor. PyCharm constantly monitors the quality of your code and always shows the results of its code inspections in the gutter: errors, warnings, and so on. The indicator in the top right-hand corner shows the overall status of code inspections for the entire file.
- Tool windows are specialized windows attached to the bottom and sides of the workspace and provide access to typical tasks such as project management, source code search and navigation, integration with version control systems, and so on.
- The status bar indicates the status of your project and the entire IDE, and shows various warnings and information messages like file encoding, line separator, inspection profile, and so on. It also provides quick access to the project interpreter settings.

Customize your environment Feel free to tweak the IDE so it suits your needs perfectly and is as helpful and comfortable as it can be. Go to File — Settings (PyCharm — Preferences for macOS users) to see the list of available customization options.

Appearance

The first thing to fine-tune is the general ”look and feel.” Go to File — Settings — Appearance and Behavior — Appearance (PyCharm — Preferences — Appearance and Behavior — Appearance for macOS users) to select the IDE theme: the default light theme, or Darcula if you prefer a darker setting.

Editor

The many pages available under File — Settings — Editor (PyCharm — Preferences — Editor for macOS users) help you adjust every aspect of the editor's behavior. A lot of options are available here, from general settings (like Drag'n'Drop enabling, scrolling configuration, and so on.), to color configuration for each available language and use case, to tabs and code folding settings, to code completion behavior and even postfix templates.

Code style

Code style can be defined for each language under File — Settings — Editor — Code Style (PyCharm — Preferences — Editor — Code Style for macOS users). You can also create and save your own coding style scheme as in Fig.32.

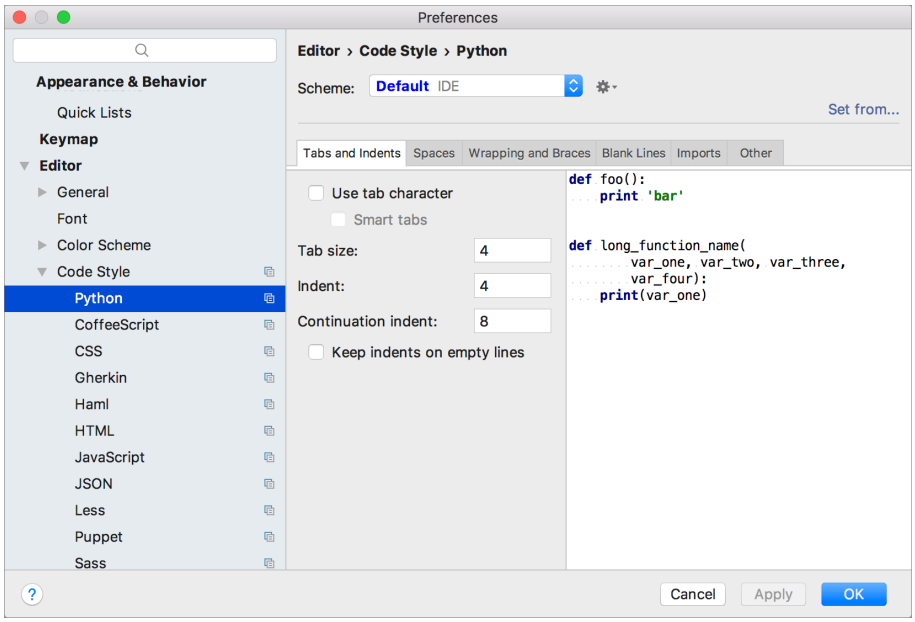

Fig. 32.

Keymap

PyCharm uses the keyboard-centric approach, meaning that nearly all actions possible in the IDE are mapped to keyboard shortcuts.

The set of keyboard shortcuts you work with is one of your most intimate habits — your fingers "remember" certain combinations of keys, and changing this habit is easier said than done. PyCharm supplies you with a default keymap (choose Help — Keymap Reference from the main menu) making your coding really productive and

convenient. However, you can always change it going to File — Settings — Keymap (PyCharm — Preferences — Keymap for macOS users).

There are also some pre-defined keymaps (like Emacs, Visual Studio, Eclipse, NetBeans and so on), and you can also create your own keymap based on an existing one.

If you feel most productive with vi/Vim, an emulation mode will give you the best of both worlds. Enable the IdeaVim plugin in the IDE and select the vim keymap.

Refer to the section Configuring keyboard shortcuts for details.

Code with smart assistance PyCharm takes care of the routine so that you can focus on the important. Use the following coding capabilities to create error-free applications without wasting precious time.

Code Completion Code completion is a great time-saver, regardless of the type of file you're working with.

Basic completion works as you type and completes any name instantly.

Smart type completion analyzes the context you're currently working in and offers more accurate suggestions based on that analysis. Refer Fig. 33

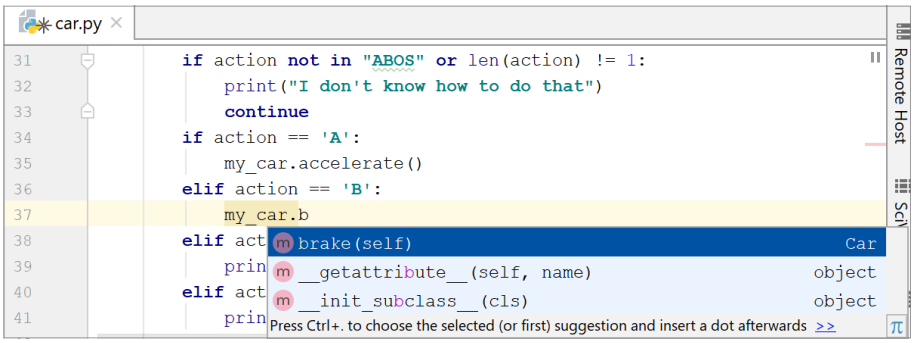

Fig. 33.

Intention Actions

PyCharm keeps an eye on what you are currently doing and makes smart suggestions, called intention actions, to save more of your time. Indicated with a lightbulb, intention actions let you apply automatic changes to code that is correct (in contrast to code inspections that provides quick-fixes for code that may be incorrect). Did you forget to add some parameters and field initializers to the constructor? Not a problem with PyCharm. Click the lightbulb (or press Alt+Enter) and select one of the suggested options as shown in Fig. 34.

Keep your code neat PyCharm monitors your code and tries to keep it accurate and clean. It detects potential errors and problems and suggests quick-fixes for them.

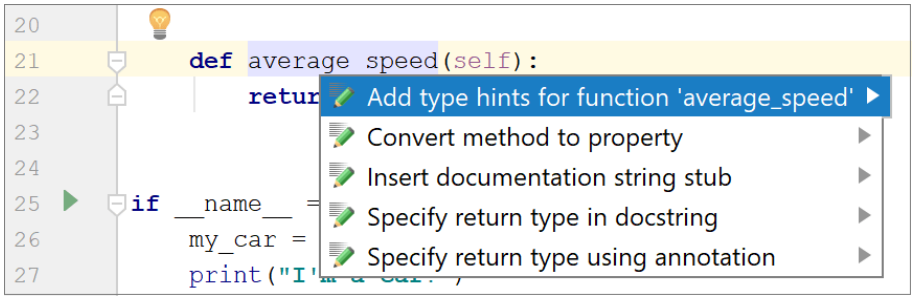

Fig. 34.

Every time the IDE finds unused code, an endless loop, and many other things that likely require your attention, you'll see a lightbulb. Click it, or press Alt+Enter, to apply a fix.

Generate some code Writing code can be a lot easier and quicker when you use the code generation options available in PyCharm. The Code — Generate menu Alt+Insert will help you with creating symbols from usage, as well as suggest overriding/implementing some functions:

Use live templates (choose Code — Insert Live Template or press Ctrl+J) to produce the entire code constructs. You can explore the available ready-to-use live templates In the Settings/Preferences dialog Ctrl+Alt+S (Settings — Editor — Live templates)

If you see that you are lacking something especially important for your development, extend this set of templates with your own. Also, consider quickly surrounding your code with complete constructs (choose Code — Surround With or press Ctrl+Alt+T.

Find your way through When your project is big, or when you have to work with someone else's code, it's vital to be able to quickly find what you are looking for and dig into the code. This is why PyCharm comes with a set of navigation and search features that help you find your way through any code no matter how tangled it is.

Basic Search With these search facilities, you can find and replace any fragment of code both in the currently opened file Ctrl+F, or in an entire project Ctrl+Shift+F.

Search for usages
To find where a particular symbol is used, PyCharm suggests full-scale search via Find Usages Alt+F7

Project navigation
You can tell a lot just looking at your File Structure, with its imports or call hierarchies.

Navigate through the timeline Remembering all your activity in the project, Py-

Charm can easily navigate you to the Recent Files Ctrl+E or Recently Changed Files Shift+Alt+C.

To go through the history of changes, try using Back/Forward navigation (Ctrl+Alt+Left/ Ctrl+Alt+Right) and/or go to last edit location Ctrl+Shift+Backspace.

Search Everywhere If you have a general idea of what you're looking for, you can always locate the corresponding element using one of the existing navigation features. But what if you really want to look for something in every nook and cranny? The answer is to use Search Everywhere!

To try it, click the magnifying glass button in the upper right-hand corner of the window, or invoke it with Double Shift (press Shift twice) as shown in Fig. 35.

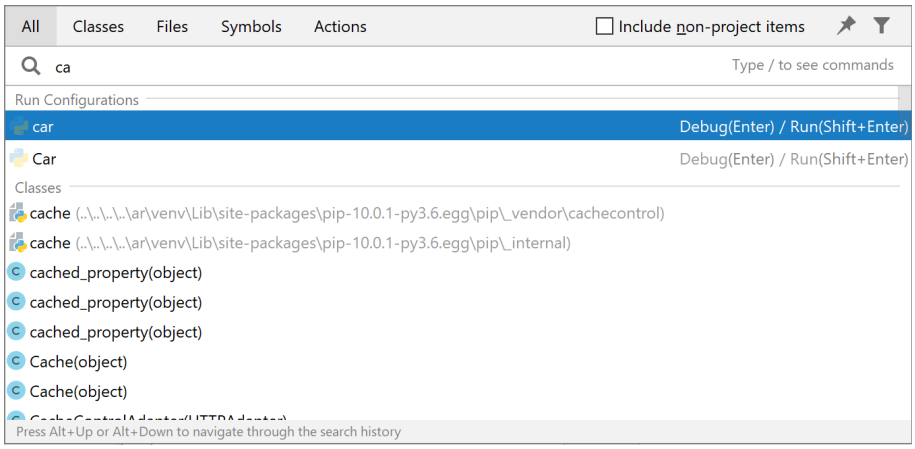

Fig. 35.

Run, debug and test Now when you've played with the code and discovered what you can do with it, it's time to run, debug and test your app.

Run

The easiest way to run an application is to right-click its background in the editor, and then choose Run ¡name¿ from the context menu in Fig.36.

Run Configuration When you perform run, debug, or test operations with PyCharm, you always start a process based on one of the existing run/debug configurations, using its parameters.

When you run your application for the very first time, PyCharm automatically creates the temporary Run/Debug configuration. You can modify it to specify or alter the default parameters and save it as a permanent Run/Debug configuration.

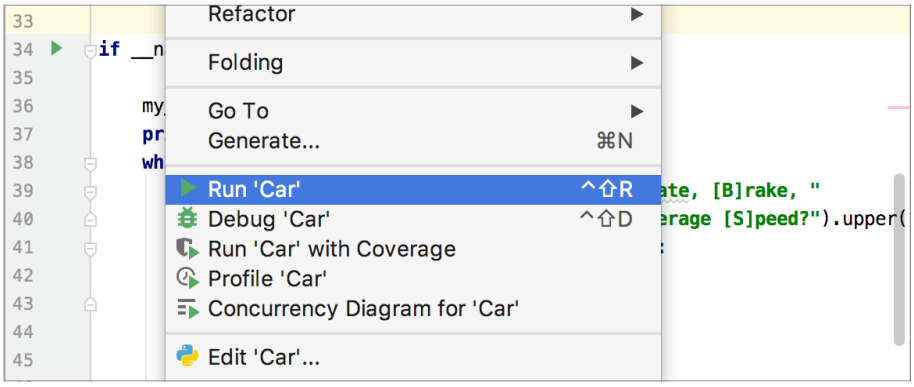

Fig. 36.

Open the Run/Debug Configurations dialog Run — Edit Configurations to see all the available options. For example, if you want to run some script before/after the build phase, you can do this easily by creating an external tool as in Fig. 37.

Debug

Does your application stumble on a runtime error? To find out what's causing it, you will have to do some debugging. PyCharm supports the debugger on all platforms as in Fig.38

Debugging starts with placing breakpoints at which program execution will be suspended, so you can explore program data. Just click the gutter of the line where you want the breakpoint to appear.

To start debugging your application, press Shift+F9. Then go through the program execution step by step (see the available options in the Run menu or in the Debug tool window), evaluate any arbitrary expression, add watches and manually set values for the variables.

Test

It is a good idea to test your applications, and PyCharm helps doing it as simple as possible.

With PyCharm, you can:

Create tests

Create special testing run/debug configurations.

Run and debug tests right from the IDE, using the testing run/debug configurations.

And, finally, the most important thing - you can explore test results in the test runner tab of the Run tool window as in Fig. 39.

Keep your source code under Version Control (VCS) If you are keeping your source code under version control, you will be glad to know that PyCharm integrates with many popular version control systems: Git (or GitHub), Mercurial, Perforce. The VCS menu gives you a clue about what commands are available. For example, you can

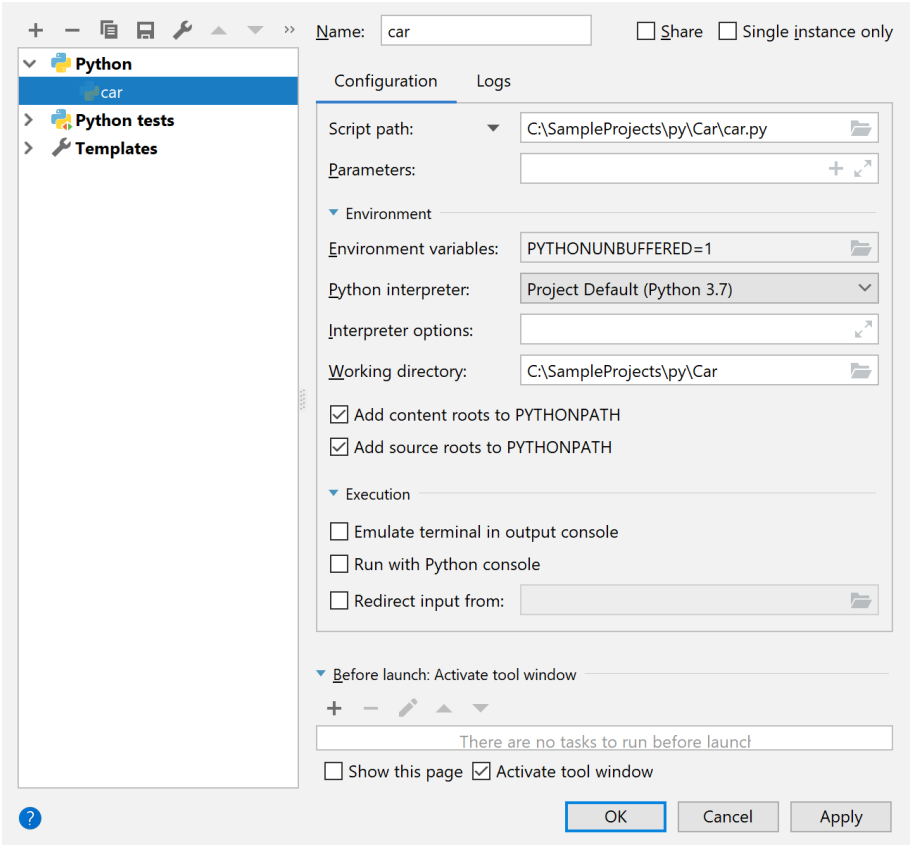

Fig. 37.

see the changes you've made, commit them, create changelists and much more from the Local Changes view: VCS — Show Changes (or just press Alt+9). Also find some VCS basic commands in the Navigation bar above the editor.

11 Installing the colour-science package [8]

Colour can be easily installed from the Python Package Index by issuing this command in a shell:

```
pip install colour-science
```

The optional features dependencies are installed as follows:

```
pip install 'colour-science[optional]'
```

The optional features dependencies are installed as follows:

```
pip install 'colour-science[optional]'
```

The development dependencies are installed as follows:

```
pip install 'colour-science[development]'
```

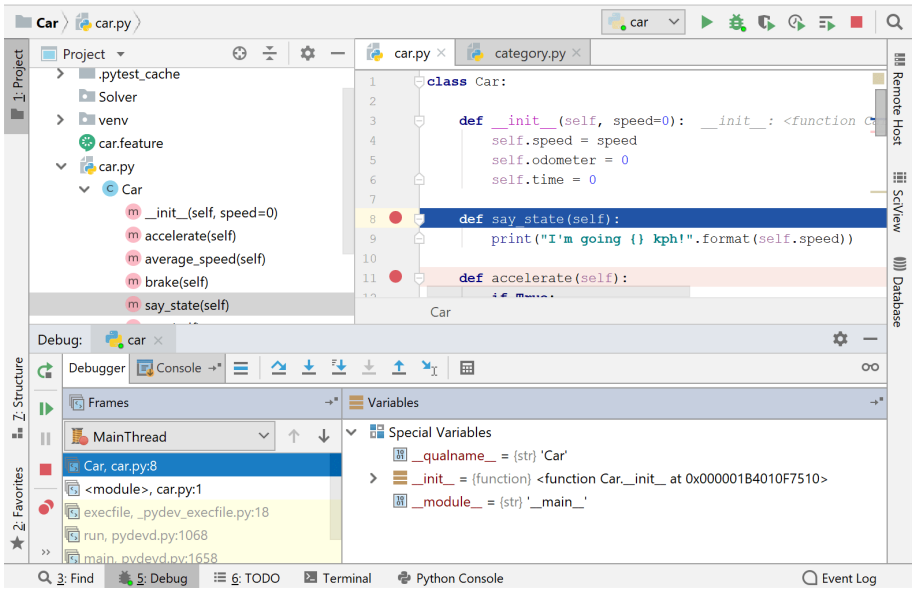

Fig. 38.

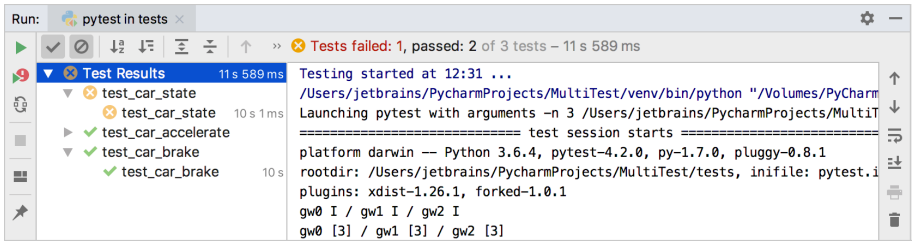

Fig. 39.

The figures plotting dependencies are installed as follows:

```
pip install 'colour-science[plotting]'
```

12 Installing OpenCV [95]

- If you have previous/other manually installed (= not installed via pip) version of OpenCV installed (e.g. cv2 module in the root of Python's site-packages), remove it before installation to avoid conflicts.
- Select the correct package for your environment. There are four different packages and you should select only one of them. Do not install multiple different packages in the same environment. There is no plugin architecture: all the packages use the same namespace (cv2). If you installed multiple different packages in the same environment, uninstall them all with pip uninstall and reinstall only one package.

– Packages for standard desktop environments (Windows, macOS, almost any GNU/Linux distribution).
`run pip install opencv-python` if you need only main modules.
`run pip install opencv-contrib-python` if you need both main and contrib modules (check extra modules listing from OpenCV documentation)

13 Installing NumPy [96]

It provides:a powerful N-dimensional array object, sophisticated (broadcasting) functions, tools for integrating C/C++ and Fortran code useful linear algebra, Fourier transform, and random number capabilities and much more.

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

All NumPy wheels distributed on PyPI are BSD licensed.

Use the following at the terminal to install:

```
pip install numpy
```

14 Some ideas to try out

– Try to approximately calibrate the monitor display (without using any instrument) such that soft-proofs may have colorimetric fidelity
– Try to measure the optical density of the 24 grayscale patches in the IT 8.7/2 target and validate your data using the ΔE values from your newreference.txt file
– Try to measure the XYZ values of any color patch that you may find on a printed substrate using the scanner as colorimeter
– And then of course, for a completely new set-up with scanner, display and printer, profile the devices and calibrate them so that the soft proof and print matches "visually" with ΔE not exceeding 2.0

15 Conclusions

As I end this document, I would like to converge my thoughts into some insights that I have gained during the whole process. My belief in the "community" of open source developers have only been strengthened over the years. While I was still learning Python and invariably finding some blocks down the road, I used to seek help from Thomas Mansencal [7] and others from the community of developers, there was not a single time that they did not respond. Try to understand the core idea behind my saying this. They could very easily have avoided answering me (remember I am not a contributor towards the community), but they chose to take the higher moral ground and be inclusive. This is something to reckon with. My key takeaway from this exercise apart from the academic and research stuff would be that generally in life we should all try to be part of a "community" that cares about each other, comes to the help

and rescue each other at the time of need without really calculating the individual gain while doing so. Secondly, it also helped me broaden my horizons on what was possible and what can be achieved using open source systems. It helped me realize my dream for creating this laboratory in spite of all the obvious infrastructural lacuna's that I wasn't able to overcome. This journey has been a constant uphill battle with an obstacle popping out almost every bit of the way. I myself struggle every day to push the boundaries of what can be achieved with what little I have in this laboratory [5]. So the journey has been a source of constant revelations and one through which I get to learn everyday.

But let me at this point also say that when I say "my dream" was realized it in turn means that maybe this might help some of my students experience the process of standardizing print workflows first hand which they might translate into a successful professional outcome in the years to come. There success would be success of the "community" that helped them realize this. And even if they don't ever use or need to implement standardized protocols for print workflows at their workplace, I would be content if I could successfully rub off some of my own enthusiasm onto them. I do not know if it would be of any value to them but I am an optimist, I can certainly hope that it would lead them to create good, great things and be a valuable contributor to the community at large.

References

[1] Phil et. al. Green. *Colour Engineering*. Chichester, England: John Wiley Sons Ltd, 2002.

[2] Henry R Kang. *Computational Color Technology*. Bellingham, Washington: SPIE - The International Society for Optical Engineering, 2006.

[3] Stephen et al Westland. *Computational Colour Science using MATLAB*. Chichester, England: John Wiley Sons Ltd, 2012.

[4] Minghui et al Xia. *INTERNATIONAL CONGRESS ON ADVANCES IN NONIMPACT PRINTING TECHNOLOGIES*. 1999, pp. 700–716.

[5] Graeme Gill. *ArgyllCMS Homepage*. URL: `https://www.argyllcms.com/`.

[6] Marti Maria. *Little CMS*. URL: `http://www.littlecms.com/`.

[7] Thomas Mansencal. *Colour-Science Package*. URL: `https://www.thomasmansencal.com/`.

[8] Thomas Mansencal. *Colour-Sience Manual*. URL: `https://colour.readthedocs.io/en/develop/manual.html`.

[9] JetBrains. *PyCharm Community Edition*. URL: `https://www.jetbrains.com/pycharm/download/other.html`.

[10] Anaconda. *Anaconda Distribution for Python*. URL: `https://www.anaconda.com/distribution/`.

[5] Even at the time of writing this, I am trying to write a code that would automatically segment each of the color patch in my test targets, but to no success till now. I contacted Thomas and he had some help ready for me, which I am trying out to find a solution to my problem

[11] Wikipedia. *Donald Knuth*. URL: https://en.wikipedia.org/wiki/DonaldKnuth.
[12] Wikipedia. *Richard Stallman*. URL: https://en.wikipedia.org/wiki/RichardStallman.
[13] Wikipedia. *Linus Torvals*. URL: https://en.wikipedia.org/wiki/LinusTorvalds.
[14] NIT Dgp. *National Institute of Technology Durgapur*. URL: https://nitdgp.ac.in/.
[15] Lawrence Lessig. *"Free, as in beer"*. URL: https://www.wired.com/2006/09/free-as-in-beer/.
[16] GNU Operating System. *"What is free software?* URL: https://www.gnu.org/philosophy/free-sw.html.
[17] Russel Cotrell. *LittleArgyll CMS GUI*. URL: http://www.russellcottrell.com/photo/index.asp.
[18] Andrzej Stawowczyk. *Digitization of Heritage Materials*. URL: http://www.dohm.com.au/.
[19] Marti Maria et al. *LCMS Profiler*. URL: http://lprof.sourceforge.net/.
[20] Lab Tools. *Colorimeter Android App*. URL: https://play.google.com/store/apps/details?id=com.colorimeter&hl=enIN.
[21] R.et al. Ravindranath. "Smart app-based on-field colorimetric quantification of mercury via analyte-induced enhancement of the photocatalytic activity of TiO_2 Aunanospheres". In: *Analytical and bioanalytical chemistry* 410.18 (2018), pp. 4555–4564.
[22] J. E et. al. Farrell. "How to turn your scanner into a colorimeter". In: *INTERNATIONAL CONGRESS ON ADVANCES IN NONIMPACT PRINTING TECHNOLOGIES*. 1994, pp. 579–580.
[23] JaZ99wro. *How to use a scanner as a densitometer*. URL: https://sites.google.com/site/negfix8pl/scandens.
[24] Lab Tools. *Color Analysis App for Android*. URL: https://play.google.com/store/apps/details?id=com.color.coloration&hl=enIN.
[25] Das Damodaran. *Das on LinkedIn*. URL: https://www.linkedin.com/in/das-damodaran-9722b45b/.
[26] Wolf Faust. *Coloraid*. URL: http://www.targets.coloraid.de/.
[27] Shankhya Debnath and Jitamitra Bagchi. "Analysing Banding Features for Classifying Print Processes using Artificial Neural Networks". In: *NIP Digital Fabrication Conference 2014*. Philadelphia, Pennsylvania, 2014, pp. 281–288.
[28] Prof Kanai Chandra Paul. *Dept of Printing Engg, Jadavpur University, Faculty Homepage*. URL: http://www.jaduniv.edu.in/profile.php?uid=862.
[29] Prof Arun Kiran Pal. *Dept of Printing Engg, Jadavpur University, Faculty Homepage*. URL: http://www.jaduniv.edu.in/profile.php?uid=581.

[30] Krishnendu Halder. *https://sites.google.com/view/krishnenduhalder*. URL: http://www.jaduniv.edu.in/profile.php?uid=862.
[31] The GIMP Team. *Open Source Image Editing Software GIMP*. URL: https://www.gimp.org/downloads/.
[32] NIH US. *Open Source Image Editing Software ImageJ*. URL: https://imagej.nih.gov/ij/download.html.
[33] Bruce Lindbloom. *Color Space Transformation*. URL: http://www.brucelindbloom.com/index.html.
[34] Roy S Berns. *Billmeyer and Saltzman's Principles of Color Technology*. Hoboken, NJ: John Wiley Sons Inc., 2019.
[35] Gunther Wyszecki and W.S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Hoboken, NJ: John Wiley Sons Inc., 2000.
[36] The Inkscape Project. *Open Source Scalable Vector Graphics Editing Software Inkscape*. URL: https://inkscape.org/release/inkscape-0.92.4/.
[37] Marti Maria et. al. *LProf Documentation*. URL: http://lprof.sourceforge.net/help/lprof-help.html.
[38] Norman Koren. *Gamma and black level website*. URL: http://www.normankoren.com/makingfineprints1A.html#Monitorsetup.
[39] Eberhard Werle. *Quick Gamma*. URL: http://quickgamma.de/indexen.html.
[40] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/ArgyllDoc.html.
[41] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/dispcal.html.
[42] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/printcal.html.
[43] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/targen.html.
[44] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/filmtarg.html.
[45] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/printtarg.html.
[46] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/chartread.html.
[47] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/dispread.html.
[48] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/filmread.html.
[49] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/scanin.html.
[50] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/illumread.html.
[51] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/fakeread.html.

[52] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/synthread.html.
[53] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/cb2ti3.html.
[54] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/kodak2ti3.html.
[55] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/txt2ti3.html.
[56] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/cxf2ti3.html.
[57] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/ls2ti3.html.
[58] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/fakeCMY.html.
[59] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/average.html.
[60] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/colprof.html.
[61] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/mppprof.html.
[62] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/revfix.html.
[63] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/collink.html.
[64] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/cctiff.html.
[65] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/applycal.html.
[66] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/iccluhtml.
[67] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/xicclu.html.
[68] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/mpplu.html.
[69] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/greytiff.html.
[70] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/refine.html.
[71] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/iccgamut.html.
[72] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/tiffgamut.html.
[73] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/viewgam.html.

[74] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/iccdump.html.
[75] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/profcheck.html.
[76] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/invprofcheck.html.
[77] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/splitsti3.html.
[78] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/timage.html.
[79] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/mppcheck.html.
[80] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/spotread.html.
[81] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/colverify.html.
[82] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/synthcal.html.
[83] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/ccxxmake.html.
[84] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/extracticc.html.
[85] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/extractttag.html.
[86] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/dispwin.html.
[87] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/oeminst.html.
[88] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/specplot.html.
[89] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/spec2cie.html.
[90] Graeme Gill. *ArgyllCMS Documentation*. URL: http://www.argyllcms.com/doc/FileFormats.html#.ti1.
[91] Axiphos GmbH. "Reviewing the concept of paper brightness". In: *Axiphos GmbH Axiphos GmbH Marketing, Trading and Consulting*. 2011.
[92] Graeme W. Gill. "A Practical Approach to Measuring and Modelling Paper Fluorescence for Improved Colorimetric Characterisation of Printing Processes". In: *Color and Imaging Conference, Vol. 2003. No. 1. Society for Imaging Science and Technology*. 2003.
[93] Russell Cotrell. *LittleArgyll GUI*. URL: http://www.russellcottrell.com/photo/matrixCalculator.htm.
[94] JetBrains. *PyCharm Download*. URL: https://www.jetbrains.com/pycharm/download/.

[95] opencv. *OpenCV dor Python*. URL: https://pypi.org/project/opencv-python/.
[96] SciPy. *NumPy*. URL: https://numpy.org/.