**The Feasibility and Demonstration of a Method Using a Digital Camcorder or Camera for Recording of Projected 8 mm Film**

by Gary Dyrkacz,

March 12, 2015

Table of Contents

[SECTION 1. INTRODUCTION 2](#_Toc413404295)

[1A. Description of the Film to Video Problem Using a Camcorder 2](#_Toc413404296)

[1B. I am not after the best quality video, but I am cheap and I have a camcorder. Can I get an okay video without becoming a programmer? 5](#_Toc413404297)

[2B. Determination of the camcorder visual frame time and the vertical blanking times 10](#_Toc413404298)

[2C. Leader Frames Data Reduction 10](#_Toc413404299)

[2C.1. General information on using ImageJ and protocols on software command or function use 10](#_Toc413404300)

[2D. Isolating the Projector Shutter Image Shape 13](#_Toc413404301)

[2D.1. Determining the visual frame and vertical blanking timings 13](#_Toc413404302)

[2D.2. Determining the camcorder visual frame row scan time 13](#_Toc413404303)

[2D.3 Image generation within a frame 16](#_Toc413404304)

[2D.4. The luminance profile of the image of a moving object (bar) 18](#_Toc413404305)

[2D.5. Between frame (interframe or Vertical Blanking) time 20](#_Toc413404306)

[2E. Vertical Object Movement Characteristics 24](#_Toc413404307)

[2E.1. Vertical Calculation of Moving Object Distortion 25](#_Toc413404308)

[2E.2. Calculating the position of the bar on successive frames 28](#_Toc413404309)

[2F. Exposure calculations 29](#_Toc413404310)

[2F.1. Leading edge 30](#_Toc413404311)

[2F.2. Trailing edge 30](#_Toc413404312)

[SECTION 3. FILM PROCESSING 31](#_Toc413404313)

[3A. Deflicker process that can be used with the concept of targeted compositing 31](#_Toc413404314)

[3A. Developing the Projector Shutter Image (PSI) Luminance Distribution Curves 33](#_Toc413404315)

[3A.1. Transferring film to camcorder – recording the film 34](#_Toc413404316)

[3A.2. Correcting optical vignetting 36](#_Toc413404317)

[3A.3. Extracting the PSI profile image 46](#_Toc413404318)

[3B. Analyzing the leader data 49](#_Toc413404319)

[3B. 1 Extracting the row based luminance variations and base PSI profile 50](#_Toc413404320)

[3B.2. Smoothing and modeling the maxima and minima data 51](#_Toc413404321)

[3B.3 Extracting and compiling the base PSI 55](#_Toc413404322)

[3C. Enhancement of the PSI and frame luminance curves - possible alternate method to determine PSI profile and luminance curves 59](#_Toc413404323)

[3D. Calculating the velocity of the PSI and the shutter width on the visual frame 61](#_Toc413404324)

[3D.1. Calculating the position of the PSI on successive frames 63](#_Toc413404325)

[3D.2. Manually determining the center position of a PSI 67](#_Toc413404326)

[3D.3. Template recognition 67](#_Toc413404327)

[SECTION 4. PSI REMOVAL BASED ON DIFFERENCE PROFILE BETWEEN SUCCESSIVE FRAMES 71](#_Toc413404328)

[4A. Removal of the PSI from real scene film footage 76](#_Toc413404329)

[4B. Final operations 81](#_Toc413404330)

[4C. Limitations 82](#_Toc413404331)

[APPENDIX 85](#_Toc413404332)

[COMBINED ROUTINE STEPS TO CONVERT AN .MTS FILE WITH RECORDED 8MM FILM TO A DIGITAL VIDEO 85](#_Toc413404333)

[Copyright Information 87](#_Toc413404334)

# ****SECTION 1. INTRODUCTION****

The basis of this investigation was a notion for an enhanced process to eliminate the inevitable flicker when a camcorder is used to record 8mm or super 8mm film that is projected onto a wall or screen. Using a camcorder to record film is one “homemade" or do-it-yourself method sometimes considered to convert film to digital media.

Ultimately, removing the flicker succeeded, but not quite as initially envisioned. The final process largely, but not completely, removes the shutter shadow from film. This is a detailed report describing how the results were achieved, along with the and analysis files and code scripts codes to arrive at a solution. As a personal project, a secondary goal was to expand my knowledge of aspects of video and video processing, and my expertise of computer language platforms. In that respect, I learned way more than I thought I would.

The detail provided here will be excruciating painful, except to those deeply interested in the mechanics of the process, or bent on extending it. In many respects, this report is like a scientific article, except with more detail than typical journal editors like to see. Called it "dumbed down", with the hope that non experts can still get some useful information out of it. Conversely, experts in the area will be saying to themselves, "we all know that", while the less expert may say, "that is interesting". If anyone else wishes to tackle such a project, or is just curious what is involved and the mechanics of analysis, the detailed effort here may provide a basis for improved directions.

## 1A. Description of the Film to Video Problem Using a Camcorder

Attempts to use most consumer or prosumer camcorders to record projected film off a wall or screen, are usually accompanied by a dark-light flickering that appears on the final digital video, unless you have a variable speed projector available. This flicker is caused by the shadow of the projector’s shutter on each of the camcorder’s film frames, which is due to a frequency mismatch between the camcorder recording and the film projecting rates. The recording frame rate of most consumer or prosumer camcorders is typically 24 or 30 fps, while 8 mm film is projected at a rate of 18 fps.

When faced with the results from using a camcorder to record a projector image, many people just decide it is easier to send the film to a digital transfer company, where the company should be doing frame by frame transfer. In theory, the results should be of higher quality, especially when it comes to removing flicker. Color reproduction and vignetting are additional artifacts that need to be considered against a company's quality assurance policies. Commercially, and even for some hobbyists, specialized telecine hardware is used to sync the film frame rate with the capture hardware/software; this is the most time efficient approach, and done well, creates a product with the least amount of distortion and artifacts. There is no disputing this. However, the process can be expensive for good to high quality transfers, especially when a lot of film is involved.

There are many ways that have been described to reduce the flicker problem. There are not a lot of detailed perspectives on the problem. One of the best, by Bruno Peter Hennek disappeared from the internet.

As an intellectual project to challenge the high cost hardware transfer method, the potential for direct digital camcorder transfer from projected film to camcorder was examined. The basic premise, which is entirely personal and not without its righteous detractors, is that processing time on a personal computer (PC) and personal time is cheap. The final result demonstrated here will produce a video with substantial reduction in shutter shadowing. The process involves identifying the position of the shutter image on a video frame, subtracting the projector shutter shadow from the frame, and further correcting the image to reduce artifacts.

**A red flag about the processes described here:** The software-scripts-analyses used here are not turnkey methods. Anyone who hopes with a few mouse clicks, to obtain a reasonable quality transfer from video recording of projected film, should read on only for amusement. (but take a look at section 1B first.) The methods consist of manual, often tedious methods, to obtain required information, which is then coupled with automated scripts that might require some modification to meet specific needs. Script interpretation and modification can require a comfortable knowledge of Python/Jython or Excel Visual Basic for Applications (vba). Any specific problems or, in most cases, even changes to the folders used to store data or scripts will force the user into modifying the script code. I suspect the techniques described are beyond the average person’s knowledge and in some cases abilities. What can be found here, are the concepts and skeleton of a workable methodology, as the basis for sophisticated turnkey software. If you struggle using a spreadsheet program like Excel, have zero programming experience of any kind, and get confused and frustrated using simple video or image editors, the processes discussed and employed here will be daunting. The mindset for this project must be that you are embarking on an exotic intellectual adventure. I hope I succeeded in describing the process in sufficient detail to alleviate most of the intellectual, and some of the time consuming pain, to achieve decent results, but the tasks will not be without their frustrations and demands. If you are an adventurous or creative type and not afraid of some tedious work…*read on*.

This investigation quickly became a lengthy excursion into many unfamiliar software tools and image concepts with significant learning curves. For image analysis and image processing, finding and learning to use the right software tools increased the time on the project dramatically. This was all done on the cheap. Except for Sony Vegas Movie Suite, Microsoft Excel and CorelDraw Suite X5, all other software was open source. Personally, I had to learn one new programming language and relearn enough Java and C++ to interpret API or script files to understand what various filters or scripts were doing, and enough python/jython programming to create my own scripts. Thankfully, for the mechanical equipment needed to obtain data on the camcorder recording process characteristics, I already had the tools, materials and skills available. As a reference to the computer hardware used for this work: The final processing system consisted of a homebuilt computer system with a Z87 motherboard and Intel i5 cpu over-clocked to 4.5 GHz, a fast graphics card, 1 TB drives, and 16 GB of moderately fast DDR3 RAM. However, initially only 6 GB memory was used on a slower second generation Intel processor system. Adequate drive space is critical for the files used and generated here; uncompressed avi files will have upwards of 40 GB of data for even one video, and more than one avi may be needed per movie conversion.

Source code and examples are provided with this document, but there are three programming languages involved in using a total of five software packages ("apps" in the new language) and multiple modules or scripts. Excel Visual Basic for Applications (vba macros), python, and ImageJ script codes, were all used to record, collate and manipulate information and video images. Video editing packages, such as Sony Movie Studio and VirtualDub were used to edit and convert video. In addition, the complexity involved in producing a quality transfer product touches upon realms of camcorder operation that are probably rarely explored by the average camcorder user.

Admittedly, some of the included Excel files will take head scratching to decipher all the details of what they are doing and how sheets may be connected. Some of these Excel files represent very important steps in the development of certain parameters or analysis of output information that everyone attempting the process will need. Despite both *[info]* sheets describing some details and localized text boxes attempting to explain what is being done or found, the workbooks are not in a perfectly easy to follow, nor completely explained format. In many cases, referring to this document may also help understand the purposes and results of a workbook. Some worksheets are no more than artifacts of failed attempts to solve an issue, which is important to show why another path was taken. Many workbooks contain macros that automate handling complex calculations. Security options may pop up when the files are opened. None of the macros are password protected. If you have qualms about macro code security, you may review the macros yourself before enabling.

Expert programmers who make it past the disclaimer and view the scripts or macros, will quickly discover I am not an expert coder. I stumbled for many hours to write some of the scripts used here, and some could definitely be improved by experts, or translated to compiled code. In more than a few cases, I was just content to find a way to get a process to work, *efficient code be damned*! Scripts may contain artifacts of cumulative efforts and in some cases, there are sections commented out that may point to script enhancements or directions, that were not implemented for some reason. With a lot more work, several of the separate ImageJ scripts could be combined into a single script, with dialogs of checkboxes to trigger the desired output. At this stage, leaving scripts as separate stages of processing, permitted more flexibility in control and debugging.

On the internet, quite a few discussions can be found with people raving about using a camcorder to achieve excellent film to video transfer with a camcorder, and others flatly saying, they tried, and the results were awful. I believe this investigation has given me a self-educated understanding of these contrasting opinion. Both camps were right. Of course, some people have higher standards of quality than others, but beyond that, how a person does the recording is very important. The most likely cause of the nearly opposite opinions is the shutter time used for the recording and whether the camcorder was running in programmed mode or not. I will have a lot more to say about this and the averaging process after discussing the mechanics of integrating camcorder videos of film.

When I started the project to transfer our super 8 mm film to digital media using a camcorder, I found that the flicker problem had been addressed with a variety of open source or proprietary deflicker filters. However, I found that the results from the open source filters were not very effective. In most of my test transfers, the results were only slightly better than the original, and in some cases, worse. In retrospect, the reason turned out to be a combination of my naivety on what controlled flicker and the limitations of the deflicker filters, which depended on their intended use. For those filters for which I could view the source code, the deflicker process is based on some sort of total frame luminance average or weighted averaging process across a number of successive frames to “average out” light/dark contrast on successive frames. This process works well when the projector shutter shadow (or other light variance) basically covers the entire frame, but that was not what is found with a CMOS camcorder unless recording at very long exposures - 1/30s or longer. Unless long shutter times are used, the shadow is localized to specific regions on each video frame, and does not uniformly affect the entire frame luminance.

# 1B. I am not after the best quality video, but I am cheap and I have a camcorder. Can I get an okay video without becoming a programmer?

The answer to the section question is maybe; the results depend on your personal expectations of video quality. You will still have to learn something about the image analysis software called ImageJ and how to set up your camcorder properly, but the quick answer is - yes. Skip to Section 5.

**SECTION 2. MODELING**

With the failure of the existing filter methods, I began to wonder if there was a more straightforward approach to remove flicker. The process that seemed to have potential was to track where the projector shutter image will appear on a frame and target only that specific region for luminance correction. To get to that stage is a two part discussion. The first part is to understand the dynamics of the camcorder recording process and its relationship to the annoying periodic shutter image effect on every video frame. The second part is to use that information to find a suitable method to track and remove the shutter shadow on each video frame. The first part will get quite technical and mathematical. Although much of the mathematics can be glossed over, there are rudimentary aspects of the interaction of film and camcorder recording that are important to understand how and why the shadow removal process developed here works.

In my early film to camcorder video transfer attempts, I did not understand the three interrelated mechanical causes of the problem. First, I set the camcorder’s exposure mode to Programmed mode, which adjusts both the lens aperture and the shutter speed to produce a pleasing frame exposure. This affected both the shadow luminance and the width of the shutter image on the camcorder image frames. The fault was due to a naïve new camcorder user. The shutter speed needed to be manually controlled. There were discussions on this issue on various internet forums related to direct digital recording of film off screens, but this issue was muddled by some very bad advice in popular internet “how to” articles on direct recording of film with camcorders, with unspecified conditions.

Analyzing individual frames from a film sequence, and after a heck of a lot of reading, I realized that I had overlooked several critical elements of the problem. A camcorder does not instantly stop recording a frame at 1/30 of a second and within microseconds start recording another frame, there is a time interval or vertical blanking interval between the camcorder frames. In at least some image sensors, this vertical blanking is used to buffer the shutter width or shutter time interval. Because of the digital nature of video and how the images are managed and edited in the video file, there is no direct physical manifestation of this delay; there is no frame line as observed with projected film (no “bar” appears), unless the scene action is quite fast, and you are looking at individual frames when editing.

In addition, this timing introduced me to a new complication that is common in most consumer and prosumer camcorders and cameras using CMOS image sensors. The effect is known as the rolling shutter effect. There are various schemes used by different sensor manufacturers for resetting, integrating, and reading out the luminance or intensity values for each pixel (really a light sensitive electronic circuit element). A common scheme with CMOS image sensors is to reset one row of sensors - pixels - at a time, allow all the pixels in the row to accumulate photons for a certain delay time related to the shutter speed, read out the values of the row of pixels, and restart the process on the next row. The operations are done sequentially on a CMOS image sensor pixel array from top to bottom, one row of pixels at a time, with the pixels in a row read out from left to right. Thus, not all CMOS pixels in the frame turn on simultaneously, accumulate light, and are then subsequently read out. (A CCD image sensor chip typically does do this.) Whatever the CMOS scheme, the critical point is that it takes a finite amount of time to read out either a row or column of pixel elements. If an object moves horizontally or vertically very fast across the CMOS sensor active field, because of the sequential nature of the light collection and read out, the object’s image recorded at the top of the optical sensor chip cannot be in the same position as it is recorded at the bottom of the active optical area. A vertically positioned object moving horizontally from left to right across the camcorder visual frame results in an image of the object canted at an angle from the vertical. Whether the object moves or the camera moves does not matter, the effect will be the same. For a vertically moving object, the result is either a vertical compression or stretching of the image of the object.

There are many excellent examples of rolling shutter distortion that can be seen by searching the internet. The images in Figure 1 show two successive camcorder frames that recorded a vertical black bar moving from left to right in this work. (We will see later that the camcorder was moving and not the bar, but it is the relative motion that is important.)

Several other CMOS image sensor characteristics enter into the rolling shutter effect. Generally, they are known as blanking intervals. There are two blanking intervals that impact moving objects on the frame. One is horizontal blanking, which is part of the makeup of every extended row of pixels. Horizontal blanking is used to balance and fill in time around the active frame to usually produce a constant row time. Vertical blanking is the time to move from the last row of a frame to the top of the next recording frame. The vertical blanking is analogous to the frame line in film, but this blanking is variable in time. A number of image sensor functions occur during these blanking intervals, and the time slice they represent is variable depending on many other image sensor parameters that are controlled indirectly by the user or the sensor operation itself. Generally, these detailed operations do not directly concern us (nor do I completely understand them). However, we need to take blanking intervals into account as part of understanding how the frequency of the camcorders recording process and the film frame rate interact and cause the changes in luminance that we observe on the camcorder. In the image above, the impact of the blanking interval is evident by comparing the bottom of the black bar on the upper image with the top of the bar image on the lower frame. The "jump" in the horizontal position is a measure of the blanking interval time.

Figure . Two successive frames of a horizontally moving bar on white background.

With this general discussion of the issues of trying to directly record film to camcorder, we are in a better position to address the flicker problem. We need to find a method that takes into account the rolling shutter effect distortion on the projector shutter image and the relationship between when a camcorder turns on its pixels and the shutter image arrives at the camcorder visual frame.

The problem is how to get at the information on how the image exposure time sequence (rolling shutter effect) and the blanking intervals work together, to produce the series of shutter shadow images we see on the camcorder frames. These two camcorder parameters are often embodied in what is called the row scan velocity. This velocity information is not usually publically available from the camcorder or camera manufacturer, and in addition, the functions controlling the vertical blanking between frames are not given. This was true for the Canon Vixia.

Camcorder test clips of projected 8mm film were difficult to analyze , because of all sorts of uncontrolled scene changes. In recording film, there was also severe non symmetrical vignetting at the edges of the projected film making it hard to get good baseline information. Thus, a standardized system was needed that could simulate the movement of an object (band) across the video frame at constant and known velocity. There are a variety of ways to determine the row scan time and vertical blanking of a camcorder. One method is strobe measurements; a second is motion analysis of objects moving at known speeds and distances. Both strobe and motion analysis can be used for blanking times, but the strobe needs to be able to operate at very fairly low rates to capture the blanking times. For expense reasons, I chose to use a mechanical system to obtain the scan and blank times. These two parameters are critical in compensating for a variety of motion artifacts. There are many other ways one can think of to achieve the information, but they all will require knowing the relative velocity between camcorder and object, and both the lateral and camcorder to object distances.

There are several constraints that need to be placed on whatever method we use to obtain the scan and blank times. We need to keep the moving object reasonably in focus across the entire camcorder field of view. To ultimately compensate the exposure for the projector shutter image (PSI), we need to know the relationship between the camcorder shutter speed and blurring of the PSI on the camcorder frame. I felt there were two simple ways to find the scan and blanking times: a belt system to move an object across the camcorder field, or have the camcorder move relative to the field.

The problem was also constrained by not wanting to spending a great deal of money to achieve a stable well defined object velocity. So what should move, the camera with an object on a stationary background, or a stationary camera with a moving object against a stationary background? An advantage of a stationary camera against a stationary background is that the exposure will likely remain more uniform, because it is mostly constant. The disadvantage is that it requires some kind of movement system where the object remains relatively flat across the view frame, such as a continuous belt arrangement. However, a moving belt likely will be more prone to vibrations and harmonics unless special precautions are taken. I had some scrap nylon strapping from a crate, but it was not very wide. Moreover, it turned out I was going to need over a 12” drive pulley on even the fastest constant speed motor to get a reasonable moving rate at a reasonable distance from the camera. I was concerned about the weight I might end up with and the problem of attaching a straight edge to the moving belt, which of course had to stay on as the belt turned around the pulleys. In the end, I decided it was mechanically simpler to have the camera turn and the straight edge static.

In my pile of accumulated bits and pieces of stuff waiting for a project, I had several Bodine geared electric motors. These motors are reversible and use a capacitor as part of the circuit. As hand-me-downs, the motor circuit arrangement was unknown. A long internet search finally located some analogous information for similar Bodine models.

With some initial rough calculations, I found a 14 rpm motor would serve my purposes very well. I built a small case for the motor and wired the motor circuit with an on/off push switch and a toggle switch to reverse the motor. The motor circuit set up is shown in the figure.

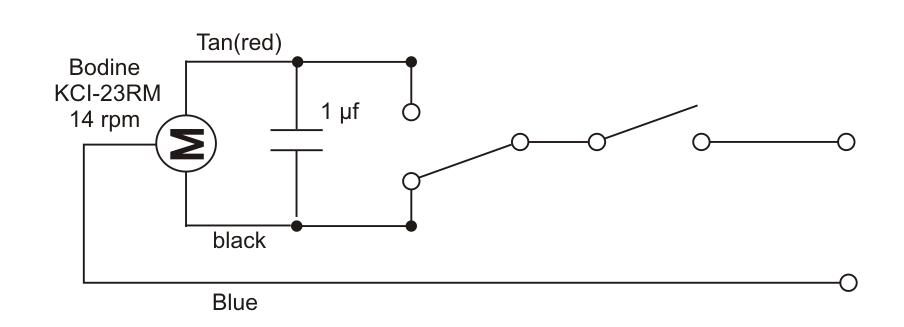


Figure . Wiring diagram for Bodine motors.

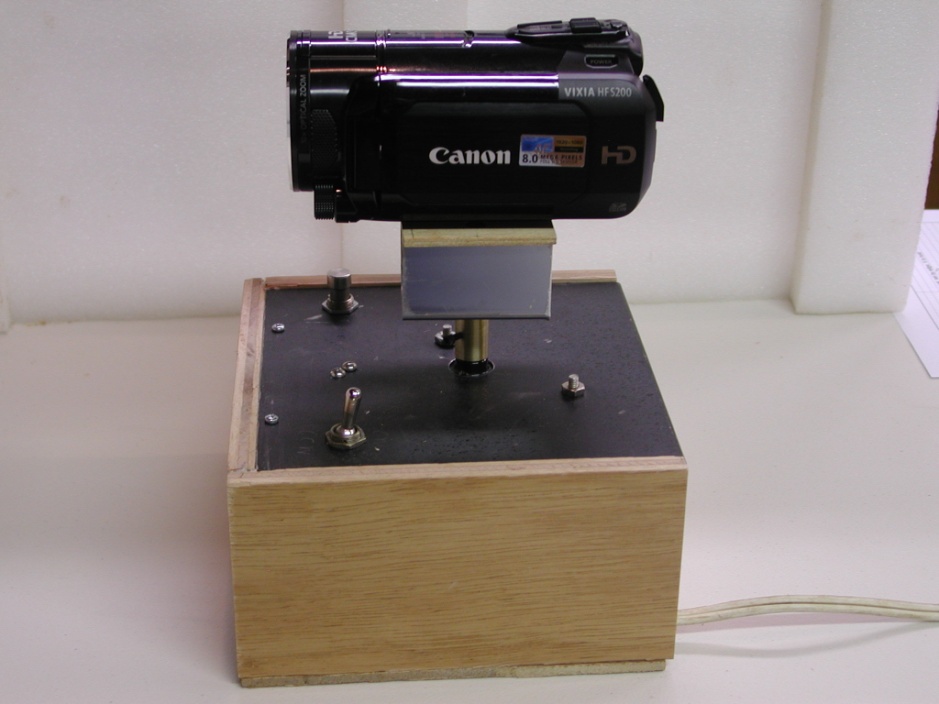
Using my lathe, I made a coupling for the motor shaft to which I could also screw a piece of U channel. The camera was then mounted to this channel. (The U channel was a piece of scrap cutoff from the frame of a sliding shower door.) Through the magic of an angle grinder, the channel was trimmed flat on the open side to mount the camcorder holding plate. The channel was then drilled and mounted to the coupling shaft. Across the top of the channel, a 1/8” lite ply plate was attached. This plate was drilled with a ¼” hole to attach the camcorder using ¼-20 nylon bolt (so as to not scratch my shiny new camcorder). Figure 3 is a photo of the rig complete and ready to go. Everything except the camcorder is from some sort of scrap material I had around.

Figure . Device to mount and rotate Canon Vixia camcorder.

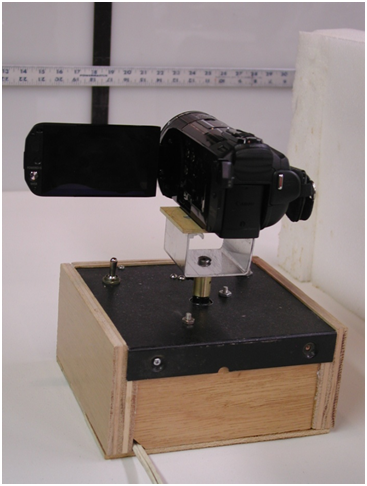
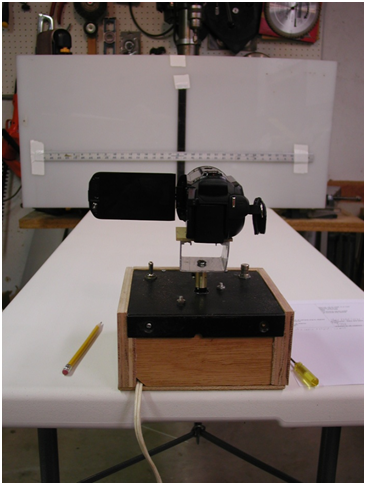


Figure 4 shows the set up in my workshop. The backdrop is a sheet of ¼” white translucent plastic, resting on the table. The black object is scrap anodized aluminum strip from a cannibalized home audio center. Both the black strip and the metal ruler where mounted normal to the camera lens axis and normal to the table upon which everything rested. Lighting consisted of three banks of fluorescent lights above the camera on either side and a third fluorescent light at the back. An incandescent lamp was also situated a couple of feet in front of the object panel and about five feet above the table. Thus, lighting was reasonably uniform.

Figure 4. Camcorder and screen set up (inside).

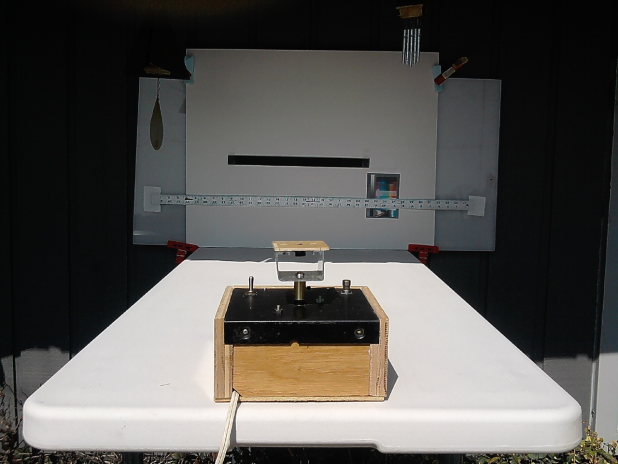
In the end, this indoor lighting arrangement turned out to be a bad idea. Despite the uniform lighting, the well known banding artifact caused by fluorescent lights 60 Hz off/on cycle, caused severe issues with the bar luminance distributions. Although some of the data were useful, the errors introduced were unacceptable. Instead, the arrangement was moved outside on a sunny day with a north-south alignment and the screen in full unfiltered sun. The recordings were done at high noon (sun directly overhead) to limit any shadow affects. However, this turned out to be too much sun against a white background, and the aperture could not compensate at the slower time settings. The assembly had to be moved under the eve of the house to provide a bit of shade. This turned out to work quite well, with a highly uniform light. Figure 5 shows the final set up - minus the camcorder, which was used to take the picture) that provided the data viewed here. Note that there was a modification of the object field as well at this point. A white, fine texture poster board was attached to the screen, and a color scanner card was included off to the right. The poster board was needed because the white plastic backdrop exhibited to too many spot reflections in daylight. (The scanner card turned out to be useless.) The poster board was the only cost of this entire project - $1.06 with tax.

Figure 5. Camcorder holder and screen set up (outside).

Video was taken with the motor running both clockwise and counter clockwise. There were two sets of data generated at 1/250th, 1/100th, 1/60th, 1/30th and 1/15th s shutter speeds with the following sets of recording conditions:

|  |  |  |
| --- | --- | --- |
| Camcorder in normal vertical position | rotation horizontal cw and ccw (drive motor shaft axis vertical) | black bar in vertical position |
| camcorder mounted sideways | rotation horizontal (drive motor shaft axis vertical; camcorder held down with hold down straps and lens axis shimmed to be parallel to table) | black bar in horizontal position |

The camcorder was placed in manual mode for the experiments, to lock certain functions across the experimental runs and provide consistent results. Two critical parameters were the shutter speed and the focus. The focus was set manually to the black bar object using the Canon Vixia’s automatic peaking function and locked for the recordings. The camcorder has the usual three exposure modes, i.e., shutter priority (Tv), aperture priority, (Av), or programmed (P). Shutter priority was used for all runs. In all cases, the exposure was checked to ensure the camcorder did not exceed the maximum or maximum lens aperture.

Knowing the motor rotation accurately is important as a totally independent timing mechanism. What we finally need to determine is the moving object's velocity in pixels per second. This information will be the basis to calibrate all the timings and scan rates for the camcorder.

Three parameters dictated how fast an object scans across the visual frame, the rpm of the motor, the distance of the CMOS sensor from the target, and the width of the field the camcorder scans. From these values, the velocity in pixels per second of a bar at a fixed distance from the camcorder can be determined from:

This equation is based on calculating the inches per second (related to the circumference of the circle at target distance) that the bar is moving, and the pixels per inch (at target distance of the visual frame field). The latter relationship is the reason for the metal ruler shown in the images, which is at the same distance as the bar. The value in inches that the frame is covering was simply read off the visual frame when the camcorder was not turning. The following Table displays the values for the particular setup used in the results obtained from the outdoor measurements and the calculated from this equation.

|  |  |
| --- | --- |
| **Moving bar parameters and velocity of bar** | |
| Frame width in pixels (HD mode) | 1920 |
| Frame width in inches | 14.56 |
| Camcorder to bar distance (in.) | 41.75 |
| rpm | 14.22 |
| vb (pixels/s) | 8196.91 |

The rpm was based on using a stop watch for 28 rotations.

## 2B. Determination of the camcorder visual frame time and the vertical blanking times

The camcorder was run strictly from the battery for all recordings. Each run consisted of the following sequence: Set the shutter value, center the black bar near frame center, set motor rotation to clockwise, start recording to provide a static baseline, announce the conditions, turn on the drive motor and record 10 revolutions, announcing each revolution number as the bar entered the field of view. When the revolutions where done, stop the drive motor, reset the camcorder to center the black bar in the frame, change the rotation direction to counter clockwise, and repeat the recording sequence.

Data reduction was a tedious, labor-intensive process; it consisted of several stages to reduce the output to a manageable size. No single software program conveniently or efficiently handled all the reduction operations. Two image editing programs and Excel were involved in data reduction, and analysis. Sony Vegas Studio was used to initially reduce the frame data to a manageable size and convert the camcorder frames to \*.png image files. ImageJ, was used to analyze the separate frame images. All the information extracted from the image analyses was passed to and processed using Excel with several specialized visual basic for application (vba) macros.

Each shutter speed run consisted of approximately a minute of recording time, which was around 2000 video frames over five to ten rotations. Because the camera was simultaneously rotating and recording 360 deg., only a small fraction of the frames recorded the black bar on white background target - roughly 6% of the video. At most, based on the 14 rpm of the drive motor, for each rotation a sequence of six frames covered this area. In addition, a couple of frames before rotation was started, established the baseline or static information on the vertical bar size and distance from the screen. To ensure the drive motor and camcorder had reached stability, only rotation frame sequences between rotations two to five were used, so sequences of three to six frames were used for each shutter speed and rotation direction.

## 2C. Leader Frames Data Reduction

To reduce the raw camcorder data to a manageable size, a two stage data reduction was used. All the recordings at various shutter speeds were initially loaded into Sony Vegas Studio. First stage editing consisted of putting in text frames to bracket the shutter speed sequences as markers to provide a clear index when the individual frames were rendered to image files.) Only the useful rotation sequences of roughly six frames outlined above for each shutter speed were kept; all remaining rotation frames were edited out. This left a file with text frames, baseline (static image) sequences, and shutter speed frame sequences that recorded coming only into and out of the object field. This edited sequence was saved as a Sony project file. From this project file, a second project file was generated which only left those frames which recorded the full image of the black bar, and a single text image frame and two single frames of the static bar frames for clockwise and counter clockwise rotation at each shutter speed. All audio was stripped from this project. This project was then rendered as \*.png files, to ensure lossless compression. The total number of files to analyze at this point was 122.

Sony Vegas Studio is a flexible and capable video editor program, but is not an image analysis program; it cannot analyze each frame to find the position of the bar or effectively analyze the bar image itself, hence the reason for converting the images to .png files.

### 2C.1. General information on using ImageJ and protocols on software command or function use

Image analysis and video manipulations were done with a very powerful and flexible open source image analysis program - ImageJ, also known as Fiji. This freeware program is primarily used for biological, medical, and astronomical image analysis measurements; it has many plugins and user contributed scripts that extend basic image analysis functions to make the basic app very powerful. The software continues to evolve, and was ideal for the tasks needed for the frame analysis. Quite a few scripts are used here to simplify many of the tasks. Of course, this meant conquering another code learning curve. Fortunately, there are many tutorials and macro examples in ImageJ that made the task just a little easier. Also, the basic interface of ImageJ is quite intuitive.

In many cases, the details needed to run certain commands or functions in software, are described here as a sequence of menu commands. The protocol for representing the sequence of menu bar commands or functions is to use a "|" as a separator ("*I*" in italics) between commands. In most cases, the vertical bar can be interpreted as "then click on". Files are indicated by italics, but italics are not used exclusively for this function. If discussing Excel spreadsheets (worksheets) brackets, *"[]"*, are used to signify the sheet name.

Installing ImageJ/Fiji is straightforward. Install Fiji instead of just ImageJ. Fiji is just ImageJ on steroids, and Fiji is used in the final process developed here to filter the film sequences. The interface of Fiji is exactly like that of ImageJ; the differences are in how scripts can be generated and the interface to Java. I will use the terms Fiji and ImageJ interchangeably, but in any event, Fiji was used. Once ImageJ has been loaded onto a system, for the primary use here, scripts are used to automate analysis. ImageJ allows several language interfaces to be used for scripts, including its own built-in language, which is similar to JavaScript. The script extensions denote the type of script that will be run. For instance *.txt* or *.ijm* for Fiji's internal language, or *.py* for python. Most of the scripts are based on python (actually jython). In retrospect, java is really the preferred language, despite its much stricter object definition and syntax requirements, because most of the plugins and basic ImageJ functions are written in Java. There just are not a lot of python examples available. However, switching to Java would have required an even longer time investment to relearn and improve my Java expertise. For python scripts, there are a couple of golden nugget sites of Fiji python examples, primarily by A. Cardona.

http://fiji.sc/Jython\_Scripting

http://fiji.sc/wiki/index.php/Jython\_Scripting

http://albert.rierol.net/jython\_imagej\_examples.html, and especially

http://www.ini.uzh.ch/~acardona/fiji-tutorial/

Often, it was necessary to go to the ImageJ java API for the function or plugin to fully understand what the function requires as input and provides as output.

Python does not have the performance speed of C/C++ or Java, but it has an extensive set of libraries available, many of which are also available to jython, and python is fairly easy to learn, with a forgiving, streamlined syntax. Jython also provides a conduit to many of the underlying java libraries (if you understand them). The bad news is that jython is quite a few versions behind the latest python release, and not all python libraries are available. At first, scripts were created and debugged using the open source IDE, *IEP*, but this was set to only generate python ver. 3.4 scripts, which had to be then converted to 2.7 scripts to be incorporated into a final Fiji - ImageJ script; this two stage process got old fast, and subsequently development and editing were done directly in the ImageJ editor. At times, *Notepad++* was used to generate initial scripts.

#### 2C.1.1. A few words on directory set up and scripts

Most ImageJ scripts were python/jython scripts and are barebones; they do not make extensive use of dialog windows to obtain parameters or files. Thus, the scripts may require user modifications to fit your preferred directory structure so that data files are where scripts expect them to be. As an aid to what folder structure was used in the development system, below is the file structure used, starting from a primary directory on my F drive - *F:\Canon*:

|  |  |  |
| --- | --- | --- |
| **folder** | **Sub folder** | **Usage** |
| *F:\Canon* |  |  |
|  | *avi\_in* | Holds the Hotspot filtered final avi files that need to have the PSI filtered out. Also must contain the associated csv file that holds the position of the PSI in each avi frame. |
|  | *avi\_out* | Output folder that holds the final PSI filtered avi files at various stages and the intermediate csv file generated after PSI removal. |
|  | *bkgrds* | Holds the final filter image .bmp file and .png files used in developing the filter image. |
|  | *camcorder rotating tests* | Holds files and sub folders of movies and images used in understanding camcorder timings. (This directory is not included in the online file set.) |
|  | *imagedump* | Contains intermediate png files used as a virtual stack to develop intermediate and final PSI filtered avi files. Each new avi processed will overwrite files in this folder. |
|  | *ImageJ Stuff* | Contains many Excel analysis files, all macro or python script files. Served as output folder for several csv files, such as *profiles.csv and aver\_stack.csv. Also contained* other general information files. |
|  | *leaders\_avi* | Holds the leader video sequences extracted from the videos of the films. |
|  | *Overlays* | Contained extraneous filter files used to remove vignetting using VirtualDub Hotspot filter. |
|  | *templating\_files* | Contains critical files related to templates used to define and compensate for PSI:   * L*umCorr.csv*, which contains the correction coefficients for the row based luminance variations. * *PSI\_norm\_1358.csv,* which is the normalized PSI two widths wide. * *PSI\_2frame\_diff\_1358.csv*, which is the difference of two offset PSIs that would appear on adjacent frames. |
|  | *R Stuff* | Contains scripts, input and output csv files used with the statistical software package"R" for various operations, especially LOESS smoothing of complex data. |
|  | *Virtualdub stuff* | Contains initializing scripts that automatically set up, the conditions for filtering and output for operations such as black masking around useful video images, and removing projector vignetting effects. |

It is not absolutely necessary that this folder structure be used, but most scripts are "hardwired" to input or output certain files with this folder structure. Thus, using a different folder structure will require most scripts be edited to change the folder and/or filenames. This hardwiring expedites the myriad of debugging and code rewrite operations necessary to produce operational scripts, which permitted skipping the extra window clicks to acknowledge where the folders and files are.

Even though there is no extensive use of dialogs for user input, the python scripts are not written so they can be run from a command line. Hence, there is a performance hit due to screen updating, which slows processing down. This becomes particularly an issue in the processing script to filter out the PSI, which may take many hours of processing for even a two minute movie sequence (like overnight!). In some cases, it was not clear how to avoid some image to screen operations (at least from my neophyte perspective).

Another disclaimer: The scripts developed here may often not be the most efficiently written code. In some cases, the reason is my neophyte programming abilities. However, the many failed paths to remove the projector shadow required writing, rewriting, and tweaking most of the scripts many times; this took its toll on exploring the most efficient code for an operation. (Basically, I was often satisfied to just get a working script that did the job correctly, never quite sure it was the final code.) The scripts are generally loaded (or overloaded) with comments, and therefore readers will generally be spared discussion here on detailed operations within each script.

#### 2C.1.2. Loading script files into ImageJ

There are a number of ways to handle ImageJ scripts, but the following method is reasonable, and allows easy review of a script before running. To run an ImageJ script, the following process is recommended, (unless you are an ImageJ expert):

1. Start Fiji or ImageJ
2. Go to *File|New|Script..|*. A two frame script editor window will appear. The upper frame is for writing or editing scripts. The lower frame has two functions: show the output of "print" commands and to display error messages. A button on the top right of the lower frame will show either "Show Output" or "Show Errors". The button toggles between the two modes.
3. Click *File|Open..* (or *Open Recent*) and find the appropriate *.py*, *.ijm*, or *.txt* script, which should then appear in the upper frame.
4. Check parameters, file names, and directories in the script file that may need to be changed for your specific case.
5. Run the script by clicking on the *Run* button on the top left of the bottom frame.
6. Any output is determined by the script function. Generally, the macros produce either avi uncompressed video files or csv files, or both.

If a script fails to run, the bottom frame in the script window will provide information on what line the script failed. In many cases, information is also printed concerning the run.

## 2D. Isolating the Projector Shutter Image Shape

### 2D.1. Determining the visual frame and vertical blanking timings

From the horizontally moving bar video frames, or more correctly, the *.png* files generated from the videos, several critical pieces of information, such as the scan time per row, or rows per second that the camcorder acquires an image, and the vertical blanking time can be found. As the preliminary observations noted, the scan rate must be a constant for each row of the video frame and is independent of the velocity of the bar.

As already explained, a CMOS sensor sequentially integrates photons for each pixel on a row by row basis, and then reads out the information after some predetermined time relationship. The result is a shift in the average horizontal position of the moving object. The magnitude or time of the shift per row can be determined from the shift. The shift is normally small, so to establish an accurate value it should be measured over the widest possible number of rows. Naturally this is from the top and bottom of the image.

### 2D.2. Determining the camcorder visual frame row scan time

In principle, it is not necessary to use the average of the bar image to find the row scan rate. Any distinctive feature is sufficient in this regard, as long as it can be reliably determined. However, for a large number of frames to analyze, automating the data reduction is useful and often necessary. Trying to isolate by computation, what the eye perceives as a distinctive feature, is not always so easy with automatic processing. As it turns out, averaging a visible band, particularly at the fast shutter times, can provide quite good accuracy.

The basic analysis of the vertical bar images consisted of using the built in particle analysis plugin in ImageJ. The macro code to do this is **:** *Find\_Top\_Bottom\_Bars\_Binary.txt* in folder */ImageJ Stuff/*.

2D.2.1. Using Find\_Top\_Bottom\_Bars\_Binary.txt*:*

Before starting the macro, the type of measurements needs to be set. Use the *Analyze|Set Measurements* plugin to record the proper information with the following settings:



Figure . ImageJ Set Measurements window.

The script does the following operations: loads an image file, duplicates it, crops the top of the image to a rectangle 20 pixels in size vertically, by 1920 pixel horizontally, converts it to an 8 bit image, creates a binary image (only black and white), reverses the b/w image luminances, then runs the particle size analysis plugin to obtain the centroids (YM and XM), then does the same for the bottom of the image. The reason for the luminance reversal step is because the particle analyzer plugin default is to analyze white on a black background, not black on a white background.

When the macro starts it will open a file window. The first time it opens, it expects the user to click on the folder that contains ***only*** a sequence of image files. It will then ask for the output folder to save the Excel file. The output file name is hardwired into the script (the name can be changed in the macro) which places the file in the user designated output folder. An example result file can be found in *F:/ImageJ Stuff*/*topbotresults2.xls.* This macro code runs in batch mode. The Results table will open and results are added as the analysis proceeds. The centroid data is saved directly into the Excel file. The YM values are related to the positions of the bar.

The script generated some 3600 entries; this was more than 10 times greater than expected. The raw data was transferred to a new file, *F:\Canon\topbotresultsoutside.xlsm [IJResults1]* Some artifacts were obvious, such as the title slides between shutter changes, others were due to a mismatch in what was considered a continuous “particle” by the particle analyzer plugin.

Removing these artifacts was done using the area measurements recorded by the ImageJ particle analysis script macro. The areas are a measure of the width of the bar band on the image and the vertical pixel range. The vertical region pixel count was always set at 20 pixels. The band areas for even the stationary bands was a minimum of 2600 sq. pixels. A simple Excel macro, *RemoveExtraneous,* was generated to remove all the low area bands (*see [Filtered]*)*.* A conservative value of 1000 was chosen as cutoff and this resulted in only 265 bands. However, the 1/15th s images were not well handled by the macro, because of the low luminance of the imaged bar. There were many values that clearly were artifacts. In addition, in order to integrate over such a long time, the camcorder uses a frame doubling process, to achieve the expected integration time. The effect of this is that the camcorder ends up with two duplicate image frames. This further complicated the low shutter speed data.

It also turned out the relative alignment of the camcorder and the vertical black bar was not perfect, due to a slight amount of play between the motor shaft and the coupling, and a very small mismatch between the camera position and the vertical bar that was not apparent when setting up the relative angle, and could be accessed using the baseline static frames that were part of the frames recorded. This effect was quite small, and ranged from 8 to 30 pixel differences between the bottom and top of the video frame, but this offset was not considered negligible.

Another correction was needed to accurately compare the position of the moving bar from frame to frame. As already mentioned, the bar positions were determined from a region spanning the image horizontally (1920 pixels) by 20 pixels vertically at the top and bottom of the frame. The reason for the 20 pixel width was to ensure a reasonably low error on the measurements. However, the recorded centroid positions of the rectangular region do not represent the 0 and 1080 positions of the image. It was necessary to correct for this. The two point formula for a linear line was rearranged to:

where Eqn 1

X and y are the pixel coordinates in the horizontal and vertical direction, respectively. Subscripts t and b refer to the top and bottom (x,y) coordinate values respectively; d is a correction factor, which takes into account the slight off-vertical shift between the camera vertical axis and the vertical bar vertical axis, and is the horizontal (x) difference in pixels between the position of the bar measured at the top and bottom of static bar images. The correction factor d was found to be 34.67 pixels for this particular run from the static images obtained just after the shutter time was changed, or the camcorder rotation changed. Compared to the vertical resolution of 1080 pixels, this is a relatively small change, but it does affect the result, because the rolling shutter pixel displacement for the top and bottom band position is also small.

The rolling shutter shift corrected to an x at y = 0 and x at y = 1080 reference was calculated directly as:

Eqn 2

(The value of 1080 in the denominator reflects the centroid of the rectangle for the bottom value is still only 20 pixels.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rolling Shutter Shifts of a bar moving at 8196.91 pixels/s** | | | | | |
| **Shutter speed; s** | **time; s** | **intraframe Top/Bot shift (pixels)** | **interframe blanking shift (pixels)** | **intraframe Top/Bot shift (pixels)** | **interframe blanking shift (pixels)** |
|  |  | **CW** | | **CCW** | |
| 1/250 | 0.004 | -140.84 | -150.15 | 139.14 | 150.20 |
| 1/100 | 0.01 | -139.48 | -151.07 | 139.31 | 150.25 |
| 1/60 | 0.01667 | -139.78 | -151.55 | 140.24 | 151.37 |
| 1/30 | 0.03333 | -138.36 | -152.98 | 106.18 | 148.99 |
| 1/15 | 0.06667 | - | - | - | - |
| averages |  | **-139.61** | **-151.44** | **139.56** | **150.20** |
| std dev. |  | 1.18 | 1.18 | 0.97 | 0.97 |
| time; s |  | -0.017033 | -0.018475 | 0.017026 | 0.018324 |
| 1/time |  | -58.71 | -54.13 | 58.73 | 54.57 |

In the table, "intraframe" refers to the shift of the bar within the visual frame from the top to the bottom of the frame (0-1079 pixels); "interframe" refers to bar shifts between visual frames.

Also note the rolling shutter shifts from top to bottom for the visual frames, or bottom of one visual frame and the top of the next frame are constant for rotation in either direction. The time to move through the visual frame or the vertical blanking region differ by a slight amount. Note also that the time to actually acquire a visual image is slightly greater than 1/60 s. The nominal times derived are just outside the std. dev, but are within the 99% confidence limit.

Although we do not know the size of the vertical blanking region, we do know the height of the visual frame. Dividing the average time to acquire the entire frame (0.01703 s) by the number of rows, 1080, gives us the time for each row to be reset, integrate photons, stop integrating and be read out. This value is 1.577x10-5 s, or 15.77 µs/row.

### 2D.3 Image generation within a frame

The preceding information is important for understanding how a periodic motion will end up displayed on the camcorder, and determining when and where a bar image (or any moving image) will appear in a visual frame. Equally important is how the image of any periodic object, such as the black projector shutter image, or black bar in the test case, will look on the frame. We are now concerned with how much the movement of the bar impacts the desired video image.

I have already shown a couple of actual images of the bars in Figure 1. Figure 7 displays the corresponding luminance profiles obtained from horizontal cross sections of the video frame as obtained using the Plot Profile plugin in Fiji. [To get the plot profile, for a vertical transect or region of interest (ROI), first do *Edit|Options|Plot Profile Options|Vertical profile*. Set the ROI using a rectangle or line, and do: *Analyze|Plot Profile.*

Figure . Moving vertical bar luminance profiles 255 RGB units is white; 0 RGB units is black.

The overlaid distributions have been plotted so that they approximately center over each other for better comparison. The intensity axis is a typical way to represent the overall intensity as the average of the three RGB color channels. In the present case, the averaged values are fairly close to grey scale, with the red channel dominating by approximately 5 – 10%.

These plots contain some critical information on understanding how the bar image and its effect on the exposure and frame is related to the shutter speed, which we will need to extract. The best way to look at these profiles is to consider them as inverted peak distributions. Later, we will invert this profile to a more common positive peak for more common visual representation.

#### 2D.3.1. Understanding how shutter time and motion affect the image of a moving bar on video frame

Note the complex behavior in the actual exposure plot of the bar in Figure 7. The minimum reversed peak luminance values appear to be a constant for some shutter times and then suddenly increase at longer shutter times. Although our main concern with film to digital media transfer depends not on changes to pixels within a row, but between vertical rows of pixels, a discussion on why this occurs at this time is important for understanding the changes between rows and visual frames.

We wish to find a simple model to predict the shape and exposure profile of the images of the bar moving horizontally across the camcorder frame. We first need to understand a bit about how the CMOS chip picks up light and reads out the luminance values. Two observations are important to understanding the mechanics and ultimately the predictive model we will use.

1. For the Canon Vixia, no difference in the width of the bar images was discernible when the bar moved from left to right, or right to left with change in shutter speed. This suggests that the real shutter delay time is very fast relative to the row to row interval time; this is consistent with the way many CMOS sensors work.
2. The linear distortion of the bar across the entire visual frame and the constant width of the bar image from top to bottom of the frame suggests that whatever happens in a row, happens in the entire row at once, or very fast relative to exposure time. For practical purposes, we therefore can assume that an entire row of pixels is exposed at the same time, and read out after the shutter delay time is reached.

Observation 1 is important because it assures us that the observed bar image width is coupled only to the bar velocity and not to the row scan rate. Observation 2 is also important both in this respect, and as an indication we can find the row scan rate independent of the bar velocity.

The schematic below ( Fig. 8) describes the system to be analyzed. A bar produces an image of length (PL - P0) on a sensor chip, which is subsequently translated into a video frame. The bar is initially at position P0 when the frame exposure begins. When the shutter delay or exposure time ts has elapsed, the bar is at position Pt. The position of the bar (object) depends on the row exposure time and the velocity, vb of the moving bar, which is assumed to be horizontal.

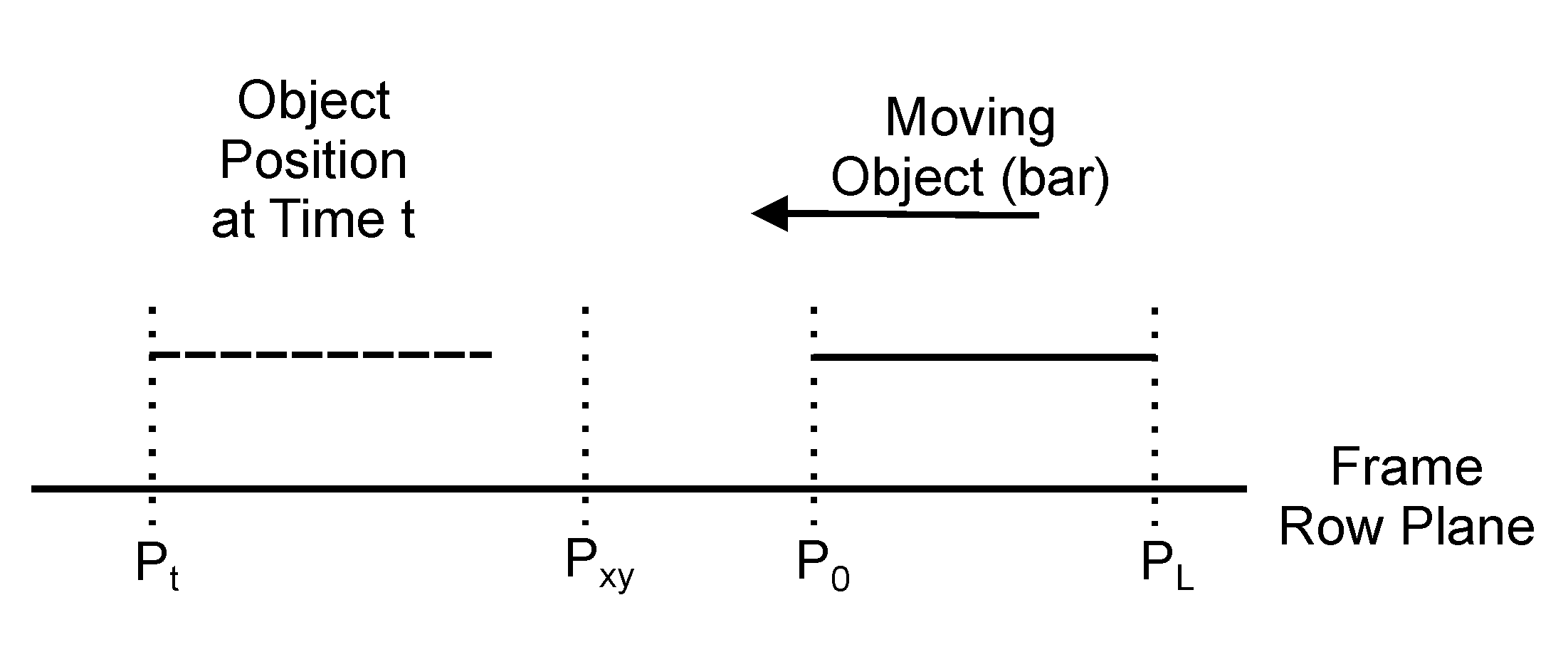


Figure . Model representation of bar moving across image plane.

The final position of the bar object from the starting position P0 at t = 0 is given by Eqn 3,

Eqn 3

k is a factor to relate the nominal shutter speed to the true shutter delay time. The number of pixels affected by the bar, i.e. the width of the bar image on the frame is,

Eqn 4

Eqn 5

or

Eqn. 6

Eqn 6 is simply the sum of the bar length and the number of pixels that the bar moved in the shutter time. This equation is important for several reasons; we will return to it later, after discussing the reasons for the shape of the bar image.

### 2D.4. The luminance profile of the image of a moving object (bar)

With the simple scheme presented in the previous section, we can understand the luminance profiles that we observed in Figure 7. There is no straightforward way to represent the truncated triangle distribution as a simple equation. Thus, we build up a picture of the image of a uniformly dark or black bar moving across a row of pixels, by sub dividing the exposure profile into three regions, the leading edge of the inverted peak, the flat peak, and the trailing edge of the peak distribution. Refer to Figure 8 and Figure 9 to help understand the exposure profiles.

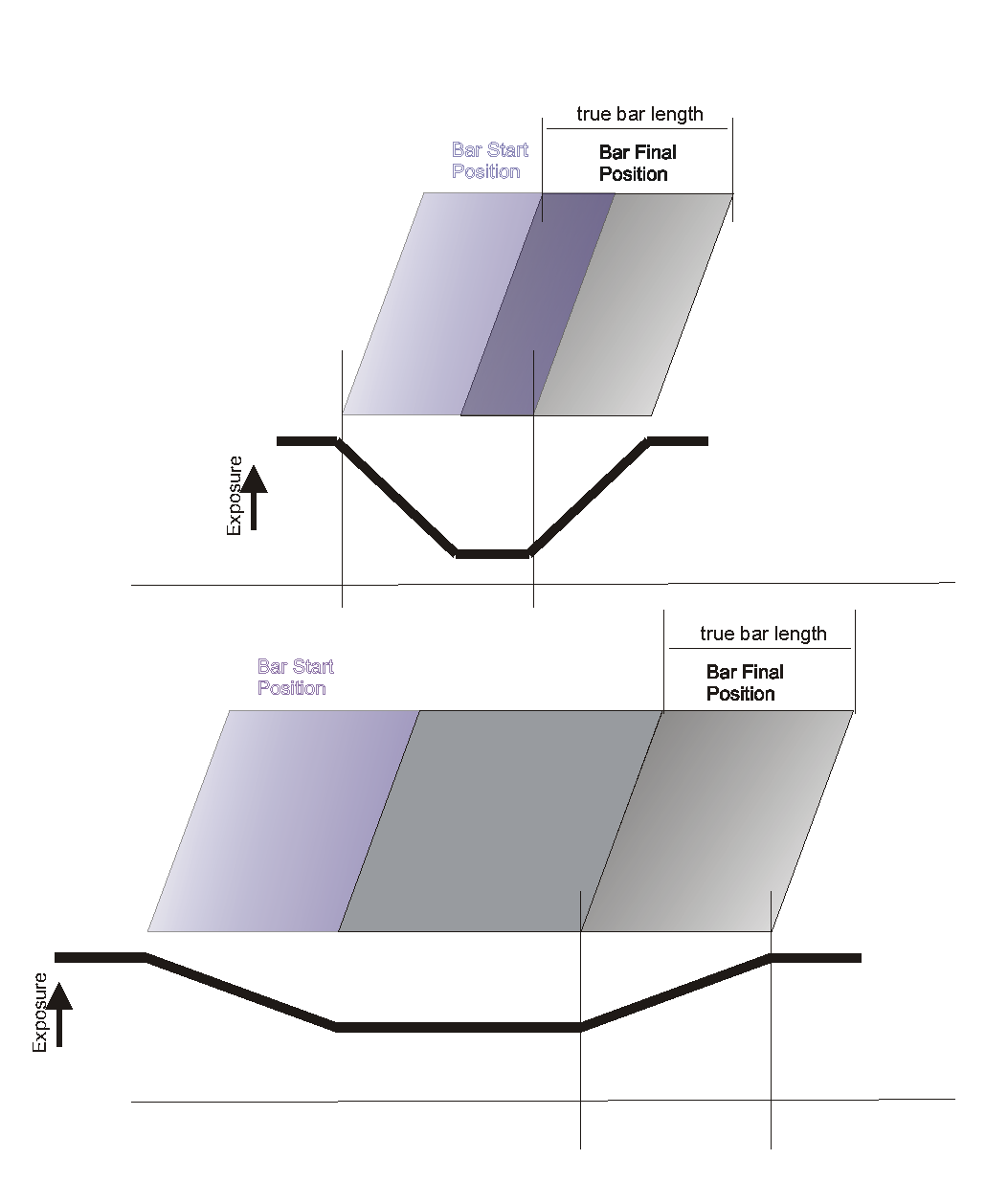
In Fig. 9, the bar starts in the rightmost position and moves to the left. At infinitely short shutter time, the image of the bar on the frame is essentially the image width of the bar. As the exposure time increases and the bar moves to the left, pixels at the right side of the bar that were initially covered by the moving bar become exposed to light. (Keep in mind that the higher the intensity or luminance value, the less dark and more white the pixel will be. The further the bar moves to the left (time increases) the more photons these early exposed “trailing” pixels receive, the higher will be their luminance value, and the closer they will be to the luminance of any pixels never under the influence of the bar. Since the velocity of the bar is a constant, the integrating exposure of each CMOS element or pixel is directly proportional to the time of exposure. When the bar image has moved on the frame so that the original right edge of the bar is at the leftmost original starting position of the bar image, all the pixels that were under the bar now are receiving the ambient light exposure. However, they still retain the loss of light from being under the black bar for a certain amount of time, until the right bar edge passed them. The plot of the exposure as a result of the movement of the bar is linear with the time exposure curve. The leading or left edge of the moving bar has the same effect on the pixels, but reversed. Since these pixels are now being successively covered by the bar, those pixels covered earliest by the bar will show a progressively greater light intensity decrease or exposure from the background exposure.

Figure . Change in luminance as bar moves across image plane

The flat or horizontal bottom portion of the exposure peak is more complicated to describe. As the bar moves, the pixels under the leading edge of the bar are still covered by the bar image until the trailing edge of the bar passes them. The pixels are not accumulating photons, and so retain an exposure value that they had just before they were covered by the bar; this static exposure continues until the bar has finally moved past the pixel position. Thus, we observe two changes as the exposure time (shutter speed) changes. At low exposure times (fast shutter speeds), but (with high aperture) the flat portion is nearly a stop action image of the moving bar, and approaches the static width of the bar. As the exposure increases, the amount of time that the once covered pixels spend without light becomes less and less, fewer and fewer pixels remain unaffected (in the dark shadow of the bar. The flat top representing these pixels not receiving light continues to shrink as we increase exposure time until the flat portion completely disappears. After the bar has moved past what was the t = 0 original position of the leading edge of the bar, the exposure for subsequent pixels in the row is simply proportional to the velocity of the bar across the frame, and the length of the bar. The result is a broad flat peak that has a nearly constant exposure value, but at a reduced luminance level from having been under the bar. The zero point at which the width of the flat portion of the peak (maximum dark) occurs is given by ktsvb = (PL - P0). Before that point the pixel range of the flat portion of the object’s exposure profile is given by (PL - P0) - 2ktsvb, and ktsvb - (PL - P0) after that point. This phenomenon takes some thinking to fully understand.

In Figure 9, the top drawing represents the case where a bar is moving across a frame from left to right. The shutter time is this case is short enough that the starting and final bar image positions overlap. To determine the original starting position of the bar in a frame requires finding the pixel value at the vertical lines. Similarly, for the final position, we would measure the right edge of the exposure profile and the right edge of the flat (minimum peak exposure). However, if the shutter time is sufficiently long, then we must take the measurements as shown in the bottom drawing. Knowing when to do this can be realized by noting when the bar position exceeds the static image size in pixels, or as we will see later, when the difference between the minimum and maximum exposure values drop.

Based on this view, we can extract several very important pieces of information from any image. The first has already been described by Equation 6. For shutter time regimes where the width of the shutter profile slopped sides are constant from one shutter speed to the next, the difference between the beginning and end of any sloped portion is a direct measure of the width of the bar in pixels. For cases where successive shutter speeds do not show a constant side slope, the bar width is given by the difference between the edge and the opposite flat peak corner. If we identify the various “breaks” in the truncated triangle as left edge, left peak (where flat portion begins), right peak, and right edge, then in the former case, the size of the bar is (left edge – left peak) or (right edge – right peak); in the latter case, the bar width is (left peak – left edge) or (right edge - right peak).

The second important piece of information we can extract is that by measuring and plotting the total width of the bar image at different shutter times, from Eqn 6 we can determine both the bar image size (in pixels) at the shutter time ts = 0. The slope of the equation is the velocity in pixels that the bar is moving. Note that the constant “k” will actually be folded into ts and cannot be directly calculated from this data. A plot of the data obtained for a vertically oriented bar moving over the visual frame is show in Fig. 10:

Figure . PSI base width change with shutter time.

The data clearly follows a straight line relationship, with the slope value representing kvb, (or really just vb, because we have no way of separating out k) and the intercept, representing the static width value of the bar on the visual frame. The problem is that neither value is in line with the values derived directly from the geometry of the camcorder setup or the image analysis. From the rational speed and geometry of the system, the value of vb was expected to be 8196.9 pixels/s, and the static width of the bar was 191.3 pixels as determined from the Profile Plot plugin of ImageJ. The intercept value of 157.3 found from Fgure 10 more closely resembles the image width of 135 pixels obtained by eye off the raw image. Thus, the slope of the line is too steep, and the entire linear curve should be shifted upwards by 34 pixels.

One reason for these difference can be seen in the composite photo below of a .png close up image of a static bar image aligned with the corresponding luminance plot profile. (The actual region used for the plot profile was much narrower than shown in the image to avoid artifacts from the slight tilt of the bar.)

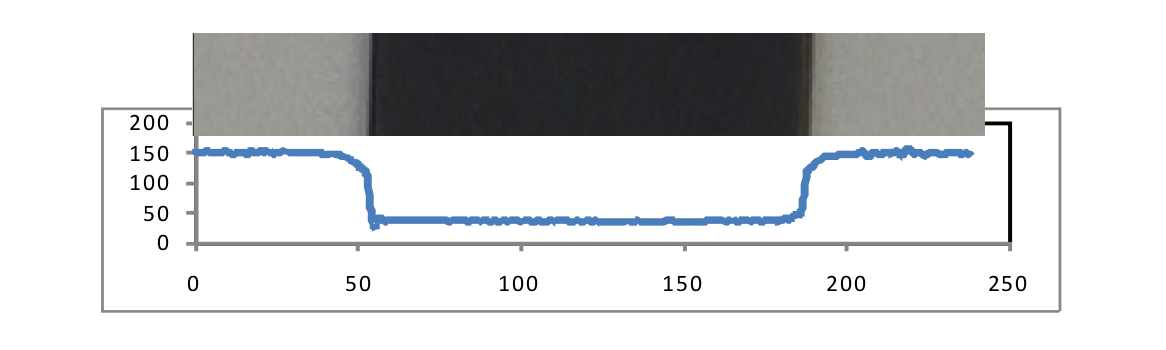
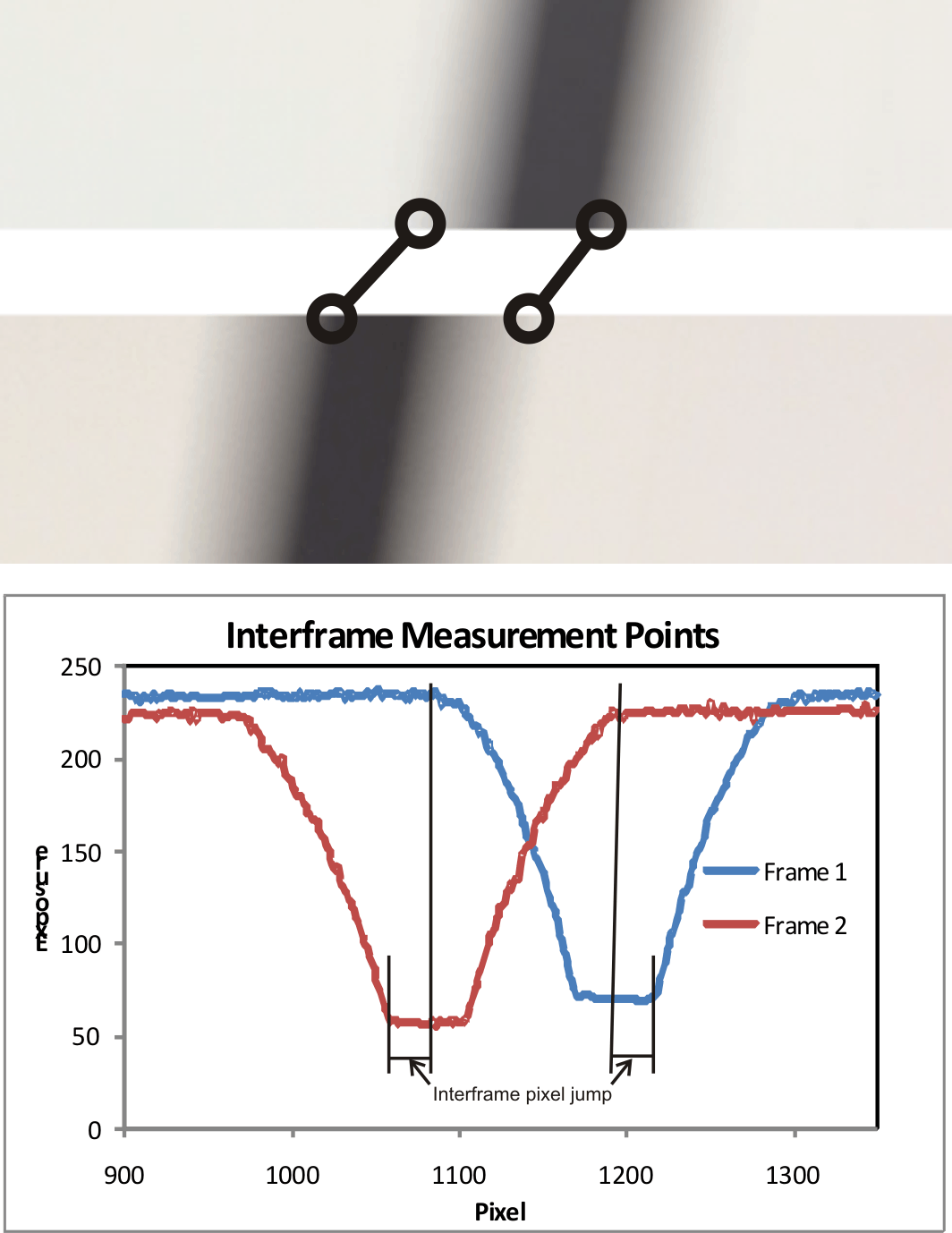


Figure . Image section of static bar and corresponding luminance profile.

Note the extended drop off of the image luminance at either edge of the bar. This is just barely evident in the bar image itself. There is a clear bleed over into adjacent background pixels. This may due to optical or electronic effects. To define a cutoff point to eliminate this problem works only for static cases as above. In general, it is not a good policy, when the overall difference between the background luminance values and the bar image values changes as a function of shutter speed. What is important to keep in mind is that all profile plots will lead to a bar image width that is wider than true image width of the object.

### 2D.5. Between frame (interframe or Vertical Blanking) time

 As the work proceeded, and the relationships in the previous section became clear, it also became clear the interframe relationship may not as straightforward as the intraframe relationships. In some respects, this section is a diversion from the main goal, but is important in realizing where measurements should or should not be taken when determining the interframe blanking parameters.

At first glance, vertical blanking between frames seems just as easy to obtain from the horizontal bar motion data as the top row to bottom row pixel shift in a single visual video frame (active frame in CMOS lingo), but the visual display is deceptive. We have to decide just what part of the bar image we will measure to monitor the change in horizontal positions between frames. Should it be an edge or the center? The difference (in pixels) between two successive bar images is not straightforward. The darkest area we see never truly represents the bar itself. it is a product of the length of the bar, its velocity across the image field, and the exposure time as Eqn 6 predicts.

A real example of the type of interframe images that were used and their corresponding luminance profiles is shown in Fig. 12. The top image shows a portion of the bottom of a frame and below it the corresponding portion of the next frame for a 1/100 s exposure. Camcorder rotation was clockwise, i.e., the image of the black bar moves from right to left. The figure below the images is the corresponding exposure distribution curves of the last row of pixels (y=1079) from the top image, and the first row of pixels (y=0) from the bottom image. Coupled with the discussion

Figure . Peak marker assignments. Top image represents bottom of one frame and top of next frame. Bottom is corresponding luminance profiles of the images.

in the previous section, the combination shows the pixel positions denoting the starting and ending position of the bar end up. Note that the pixels positions are applicable only to clockwise rotation of the camcorder. When the rotation is counterclockwise, the measurements are reversed.

Although it is possible to develop a script to automate the process of finding the appropriate corner pixel positions between frames, the time invested in developing a script using derivative analysis was considered to be far greater than the time to just manually read off the necessary values using ImageJ. A manual decomposition of the data in each frame was required. The ImageJ analysis consisted of generating the plot profile of either the top or bottom row of pixels, and use the Point Selector function (using multipoint selection) to tag and generate the measurements, which were then copied to an Excel spreadsheet.

Again, because the black bar was not perfectly normal to the frame vertical axis, a correction factor was applied to compensate for the difference between the top and bottom row of pixels. The formula was,

Eqn 7

The indices [1,1080] and [2,0] refer to the bottom row of the first frame (1), and the top value of the successive frame (2).

With this method for measuring the gap between visual frames, the vertical blanking time is not a constant; it depends on the shutter speed, but it is also not a monotonic relationship with shutter speed. The plot below shows the data for both the clockwise and counterclockwise rotation.

The mirror symmetry between the two curves, suggests we are dealing with two linear equations to describe the 250th to 30th second shutter speed range for each direction. Note that this plot does not take into account the direction of the bar velocity, only its magnitude. Hence, the reason for a mirror image rather than a parallel relationship between the CW and CCW data.

Figure . Clockwise and counter-clockwise differences in pixel widths for different shutter times.

The real surprise was that the lack of symmetry between the CW and CCW data pixel jumps between successive frames. We expected the pixel jump to be the same from frame to frame, regardless of the direction the bar moved, but the absolute value of the pixel jumps is larger for the CCW data then the CW data. (The curves are shifted.) This is very difficult to reconcile with any kind of simple model. The observation suggests that information is still being acquired when the valid frame time is over. The collection of this information is done in a specific direction, which runs with or against the image of the moving object. The result is that the image velocity appears retarded when the object is moving against the chip scan, and expanded when the object moves with the chip scanning process. If this were not true, then we are faced with suggesting the camcorder itself monitors motion and alters its data acquisition based on that direction.

The camcorder operation is more complicated than initially realized, because the sequence of events is not as linear as the model suggests. Without understanding the specific details of the Vixia CMOS chip, the general sequence of developing an image for many image sensor chips with a rolling shutter follow a similar pattern. In many image sensor chips, the exposure time for each pixel on the chip is related to the shutter width, whose value is measured in rows. There are two triggers or processor register values that define the shutter width. The first activates a sequential scan process to reset the pixels, and begin accumulating photons. The second trigger stops the integration and dumps the column values of a row of pixels to output. Thus, multiple rows may be accumulating photons at the same time, but the readout process is usually bound by a constant row time. The row time is the time to set up the process, dump the value of pixels, start and stop the accumulation, plus other delay and operation functions.

The problem is that without understanding the mechanism for this shift it is difficult to understand some details of the camcorder blanking and dark row operations.

In the end, I could not find a simple model that back engineered what was going on. To describe, rather than understand the timing process leading to this asymmetric behavior, a reduction process was used that was part logic based on expected chip dynamics, and part brute force mathematical model building. I first noticed that the equation for the difference between PB, the bar pixel position of the bottom row of the visual frame n, and PT, the bar pixel position of the top row of the visual frame n+1 is related by,

Eqn 8

When Eqn 8 is solved for ti and plotted against the shutter time two parallel horizontal lines were produced, (excluding the 1/30s data, which clearly represents a change in the way the camcorder is processing an image). It was reasoned that an additional time term, tx also dependent on vb, was needed to adjust the lines to produce a single plot. The resulting equation was then,

Eqn 9

From this equation, tx was adjusted until ti was constant for all six shutter time measurements for CW and CCW operation. The values found, were: ts = 0.0170 s and tx = 1.03 x 10-6 s (1.03 µs). By rearranging Eqn 9 to,

Eqn 10

the set of PB – PT values experimentally obtained were reproduced within an acceptable accuracy. Figure 14 displays the derived values and the experimental values is shown below.

Figure . Comparison of difference values for various shutter time between calculated and experimental pixel widths.

The question is what do ti and vbtx really represent? ti is likely related to the vertical blank time. The similarity of this value to the row time is noted, but there is no good reason for assuming it is the same. vbtx is not clear at all; it may represent nothing more than an extra parameter that happens to allow a good fit to the observed data, with no physical interpretation, or it could represent the sum of other time related camcorder function terms.

Remember that this information came from measuring the image position between successive frames when the bar moved horizontally and represented measuring the same bar edge in both visual frames. If instead we measure the same peak image position for the bottom and top rows, disregarding which image edge of the bar really represents which true edge of the bar i.e., we measure the left or right image edges, or left or right peak image edges, or as was done, the midpoint of each peak, we find the pixel difference between successive visual frames is constant and does not depend on the shutter time at all. The reason for this appears to be that in the former case, we have deliberately involved a consideration of the shutter time in the manipulation of the data. This shutter time is dependent on the optical chip shutter width and shutter delay times. The result is that the first case is closer to examining the chip reset, collection, and read out and time buffering processes. In that sense, the former analysis method is telling us a lot more about the real chip dynamics. The problem is interpreting this information is a difficult reverse engineering problem.

The upshot from this section is that for our simple goal of predicting the position of the bar on each visual frame, measuring this additional edge to proper edge information serves only to complicate the analysis. We are much better off just using the same part of the bar image to record the pixel difference, or the average position of the image at each shutter speed. The entire reason for what amounts to a digression from the main goal is as a warning to avoid confusion in considering what should be measured for different shutter speeds.

## 2E. Vertical Object Movement Characteristics

From the horizontal moving bar images, I.e., with a vertically placed bar moving horizontally across the video frame, we learned several important facts. The rolling shutter deflection was constant for all shutter speeds, within experimental error; there was no dependence on shutter speed. There was also no measurable change in the width of the image when the object moves from left to right or right to left. However, the flicker problem to solve is motion in the vertical direction, to represent the effect of a projector shutter on the visual frame.

Vertical object movement in the active image frame is not expected to have the same characteristics as the horizontal movement case. Images of any object moving up or down along the vertical frame axis exhibit a distinct expansion or compression of their width. Rows are being read in a time comparable to observable changes in the moving object’s position, and in a predetermined top to bottom direction, while the object is moving with or against this recording process. We can observe the asymmetric nature of this effect when looking at vertical exposure slices through a video frame when a horizontal bar is moving vertically. The exposure distributions are narrower when the bar moves up, as the camcorder scans rows downward.

The designation of CW or CCW as references to the bar motion are used and can be confusing when applied to applied to the vertical measurements. Clockwise rotation is defined as the camcorder mounted and shimmed on its right side to the motor mount, and the motor and the entire rotating box assembly turned on its side so that the camera rotates on the left side of the mounting rig as viewed from behind the camcorder. In this configuration, clockwise motion sweeps the bar image vertically from top to bottom across the camcorder frame. Thus, CW measurements are analogous to the direction a movie projector shutter would appear to move on the camcorder.

Two sets of successive frames in Fig. 15 illustrate the CW an CCW rotation effects. In the photo set on the left, the bar image is moving down the frame. In the set on the left, the bar is moving up. The grey level differences are due to the camera metering and aperture system being unable to respond quickly enough to the changing light conditions, because the area above the bar target was dark grey, while the table upon which everything was mounted was bright white; the camcorder swept this entire area in a 360 degree rotation. Thus, for the CW images, the aperture is automatically stopped down quite a bit more than in the CCW case. The ruler used to keep track of the width of the image and monitor distances can also be seen in the lower right photo. Note that there is visible compression of the CCW images compared to the CW.

Figure . Clockwise and counter-clockwise vertical scanning of stationary horizontal bar with moving camcorder.

We are now faced with modeling where the bar’s image will appear on successive frames.

### 2E.1. Vertical Calculation of Moving Object Distortion

To describe the rolling shutter effects in the vertical direction, we have to take into account a system with two rates. One rate is the velocity of the object, vb, in pixels/sec moving from top to bottom of the frame, and the second rate, vr, is the row scan rate of the camcorder scanning sequentially down the rows of a video frame. We already have measured both of these values. What we want to know is how these two competing rates quantitatively affect the shape of the distribution and the exposure. The information developed from the analysis of the horizontally moving bar already provided us with the scan rate and the velocity of the moving object. We also know the starting position for both the rows and the object.

For a simplified consideration of the system, we will examine the overall effect of the different rates on the distortion or smearing of the image on the video frame. Qualitatively, we expect that if the bar is moving in the opposite direction to the rows being read out, then we should see a compression distortion of the image object, i.e, the total range of rows of pixels will be lower than the static object image. If the object is moving with the row scanning process, i.e. from top to bottom, we expect a expansion or lengthening of the objects image on the video frame. To find the magnitude of this effect given the starting edge points of the bar, we need to find the points these edges will first meet an exposing row. The time for this to occur when the two rates are in the same direction can be found by equating the equations for the time that the two rates of travel will meet at the same pixel row,

Eqn 11

where P0 is the row starting position of the bar at time = 0, when the frame is activated and scanning begins; Pr0 is the row the camcorder begins the scanning process, which is just the top row, Pr0 = 0. Solving for t,

Eqn 12

t is the time that a pixel row position P0,representing some point on the continuously moving bar’s image when exposure starts, will be overtaken by the camcorder circuitry process that resets, exposes to light and reads out the pixel values. As a simple analogy, this is the same kind of problem as the common algebra example of a race between two runners to find the time when they will meet when each starts from a different position with different running speeds. The final position of the bar on the frame can then be found from:

Eqn 13

Note that P0 is any part of the bar image we consistently measure. It could be an edge, peak edge, or average value. We can also calculate the position of any other point as well. Using this equation, for both the left (PO) and trailing edge position (PL) of the bar, we find,

Eqn 14

where Pl is the full width of the image of the bar on the film frame. Although this equation allows us to see how the image bar can change, there is an important piece missing from this equation. (PL – P0) is not simply the static image of the bar, but is the image of the bar at a certain shutter speed. Thus, (PL – P0) is really the value of Lmax in equation 6. Combining Eqn 13 and 14, the expected distortion is:

Eqn 15

Or, from the definition of Lmax, the image width of a moving bar at any shutter speed ts is given by.

Eqn 16

where now (PL – P0) is the width in pixels of the static image of the bar. Note that vb will be either positive or negative, depending on whether the bar is moving down the frame {clockwise camcorder rotation, CW) or up the frame (counterclockwise camcorder rotation, CCW).

Eqn 16 can be rewritten in the form:

Eqn 17

Plotting the observed width of the bar image against the shutter time for CW or CCW rotation should result in linear equations. (The constant k has been dropped at this point, because we have no independent measure of it, and it therefore becomes part of the recorded shutter speed value.) In addition, from the intercept value and knowledge of the width of the static bar image, we can determine [1 + vb/(vr + vb)] and with the slope value we can then determine vb  independently. The plot below displays the vertical bar image widths as a function of shutter time. The data for both CW and CCW is indeed nicely linear within experimental error.

Figure . Bar width Image as a function of shutter time.

The width of the bar derived from measurements of the luminance profiles from static images was 191.3 for the bar mounted horizontally and 193.4 for the bar mounted vertically, or an average width of 192.4.

If instead of specifying the bar width (in pixels), we can calculate the bar width using the scan velocity, vr, and bar velocity, vb, we derived from the geometry and rotation information of the moving system, the bar image size is 188.9 and 175.6 for the CW and CCW cases, respectively, with the average being 182.3. This can be compared against the 192.4 pixel average bar image width found from the measurements of the stationary bar. So, although the average bar velocity is quite a bit higher than the expected value, the bar image size is just within the precision limits of the measurements.

However, everything is still not quite as clear as it seems. There is an issue comparing this image size information from a horizontal bar moving vertically, with the value found from a vertical bar moving horizontally, when the moving bar image widths are plotted against shutter speed. Unlike the case of a horizontal bar moving vertically, there is no evidence for a rolling shutter distortion of the bar image width for CW or CCW movement in the case where a vertical bar moves horizontally (only the overall bar position is affected). As shown in the luminance profile section, the bar width was found to be 157.3 pixels wide. This is in stark contrast to the profile measurements of the static bar image width before each rotation run, which produced a value of 191.3 pixels.

A third measure of the bar image width comes from measuring the difference between the profile edge and the appropriate (flat) peak value for a horizontally moving bar as indicated in the luminance profile section. The average of all the measurements in this case provides a bar width of 144.6 pixels.

In another contrast, the edge to peak values of the horizontal bar moving vertically were 201.2 pixels for the CW measurements and 144.5 for the CCW measurements. Although the two values should not be the same because of the rolling shutter effect, the CCW measurement is the same as the horizontal measured width, which indicates these measurements are showing a drift to a higher bar image width that must be between 201.2 and 144.5 pixels, and is probably close to 173 pixels.

The following table may help to put all these numbers in better perspective.

|  |  |
| --- | --- |
| **Table xx. Variation in Bar image pixel widths with type of measurement** | |
| **bar Image width derived from:** | **bar image widths in pixels** |
| ***static Measurements of bar image width:*** | |
| By eye directly from image (using ImageJ line function) | vert. bar = 135.0; horiz. bar = 135.0; |
| From luminance profile (Plot profile function in ImageJ) | vert. bar = 193.4; horiz. bar = 191.3; aver. Is 192.4 |
| ***measurements from dynamic data:*** | |
| Direct: from vertical bar moving horizontally profiles, measured as difference between baseline edge and appropriate peak point. | Vert. bar = 144.6 |
| Indirect: Intercept from plot of vert. bar full peak profile widths vs shutter time | Vert. bar = 157.3 |
| Indirect: From horiz. bar moving vertically, plotting full peak profile widths vs shutter time, and eqn 2 with vb = 8197 and vr = 63420 | Bar moving down frame = 188.9  Bar moving up frame = 175.6 |
| Indirect from horiz. bar moving vertically. from profile edge to flat peak differences, and eqn 2 with vb = 8197 and vr = 63420 | Bar moving down frame = 175.2  Bar moving up frame = 126.1 |

To summarize the Table, bar widths derived from examining full bar image widths whether from a moving horizontal or vertical bar are lower than static measurements of the luminance profiles, but higher than the static image measurements by eye. No perfectly clear explanation was evident to completely reconcile all the differences between these measurements. As mentioned in the last section, the luminance profiles will be higher than the “true” bar image width. It would appear that the dynamic measurements are measuring widths in between the extremes. (There is no reason why the last measurement is well out of range of the widths read directly off the image. Although tempting to suggest that maybe the problem lies in our values of vb and vr, as measurements based on reliable measurements of geometry and rotation speed, these values are believed to be much more trustworthy than the luminance profile width values, especially against the previous discussion of issues with image width measurements. One possible reason for the “in-between” character of the data is that the luminances due to bleed over around the main bar are lost in the dynamic data because of CMOS sensor light collection efficiency issues. The result of this is that the effective bar image width approaches that which is easily observed by eye. Another factor may be that in the vertical measurement, the camcorder did have some difficulty responding to the wide range of scene luminances as the camcorder rotated.

Although the discrepancies pointed out are important for a complete understanding of the dynamics of the system, for the goal intended here, it is not absolutely necessary to understand the problem. What is necessary is that whatever values we use with Eqn 2 must reliably predict both the width of the bar image at any shutter speed, and subsequently the position of the bar image as a function of time.

Equation 17 is a very important relationship; it will be used to determine the velocity of the projector shutter and the profile width, because we cannot directly determine the number from static images. The result will be a phenomenological value for projector shutter width and relative projector shutter speed, which will likely be somewhat larger than the true shutter width.

### 2E.2. Calculating the position of the bar on successive frames

This simple analysis provides a starting point for understanding the visual distortion that a rolling shutter produces, but tracking the position of the bar’s image on successive frames is a bit more complex. It is not just related to the vertical blanking time, but also must take into account the relative difference between the row scan rate and the object’s velocity, the final position when the first frame exposure ends, and the shutter speed dependency.

Just as in the case of the horizontal data, this is a case of too much information to reconcile in a reverse engineered model. Thus, average or overall parameters of the image were measured rather than the more detailed peak structural points. Just as in the case of the horizontal image reduction, ImageJ was used to measure the average position of successive video frames. The data in this case were sparse compared to the horizontal information, because generally only two or three successive frames contained sequential complete width images of the bar. In the case of the 1/30s data the information content, only one CW image set was found to be useful because of a substantially broadened image of the bar.

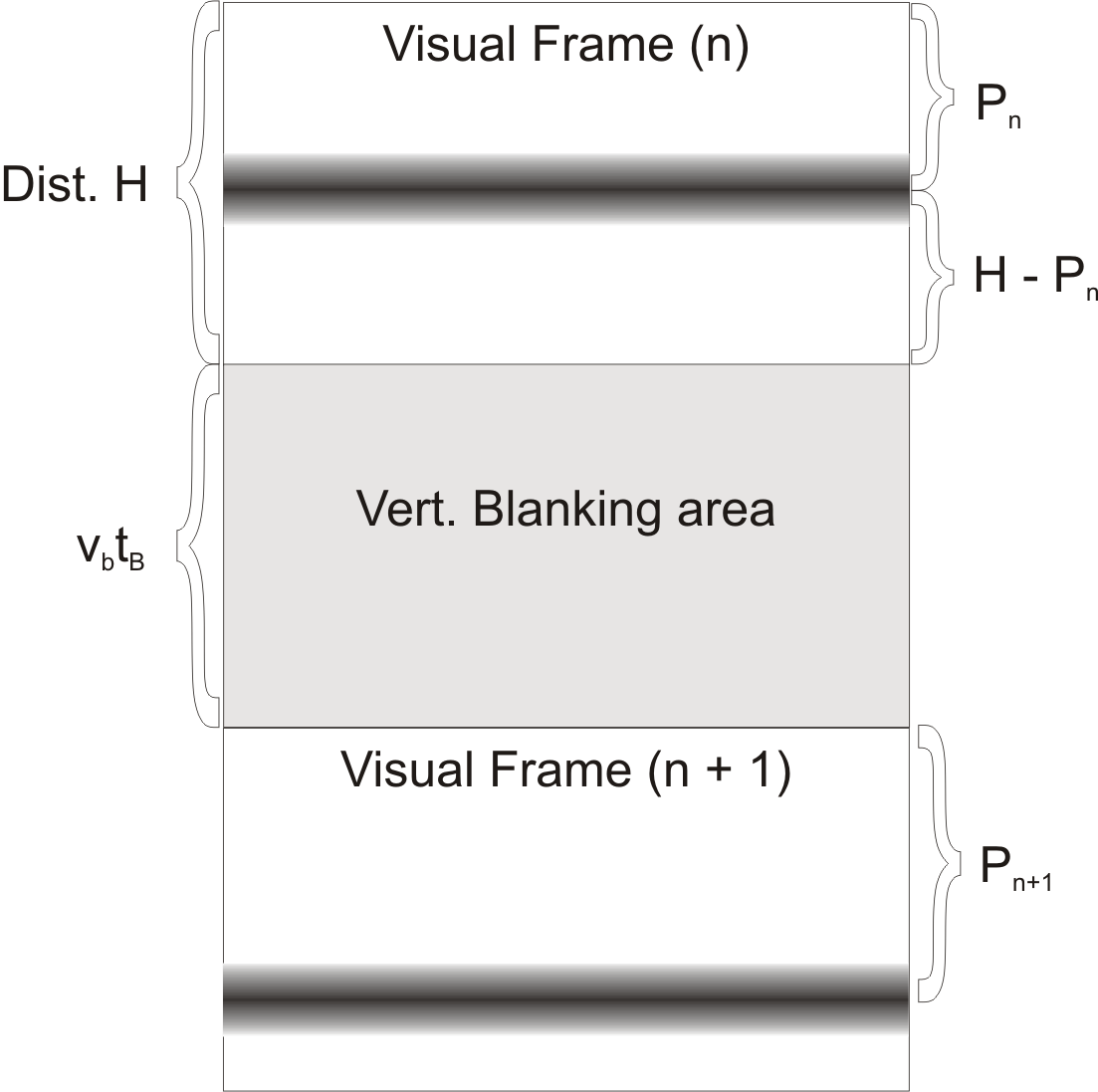
The diagram at the left may help understand the logic of how the equation for determining the position of an object (bar) on successive visual frames was established. In this case, Pn and Pn+1 are the position of a point on the bar. These points are shown as being taken at the midpoint of bar, but could be any point on the bar we choose as long as we choose the point consistently between successive visual frames. As already discussed, as long as we use the same relative pixel position on the bar image, the vertical blanking time, tB, is a constant for all shutter times greater than 1/30 s; vbtB is the number of rows (vertical pixels) that the bar will move.

Figure . Schematic representation of visual and blancking areas that make up one "frame".

The following table will be useful to show how the equation to predict successive bar positions on a visual frame was built up from the diagram representation.

|  |  |  |
| --- | --- | --- |
| **Eqn. #** | **Value** | **Equation** |
|  | For 1080 > Pn > 0 (Pn in visual frame region) |  |
| 1 | Time to get from bar position Pn to end of frame n | (H-Pn)tr |
| 2 | Time to get from end of frame n to top of frame n + 1 | tB = 0.0184 |
| 3 | time to get to opening of frame n + 1 | (H-Pn)tr + tB |
| 4 | Time to get to the final bar position, Pn+1 from when frame n + 1 recording starts | [(H-Pn)tr + tB]/(vr-vb) (with Pr0 = 0) |
| 5 | Total time for transition to image on n + 1 frame | (H-Pn)tr + tB r + [(H-Pn)tr + tB]/(vr-vb) |
| 6 | Rows moved between original Pn position on frame n and Pn+1 position of frame n + 1 | vbt = vb[(H-Pn)tr + tB r + [(H-Pn)tr + tB]/(vr-vb) |
| 7 | Row position on frame n + 1, where t is total time for transition (Eqn. 5) | Pn + vbt |
| 8 | Final row position on frame n + 1 starting from Pn | Pn+1 = Pn + vb[(H-Pn)tr + tB ] + [(H-Pn)tr + tB]/(vr-vb) |

Note that Eqn 4 is derived from an analogous equations used for examining the rates of row travel and the velocity of the bar, but with Po = (H-Pn)tr + tB which is the nominal position of the bar when frame (n + 1) opens.

The final equation that predicts the position of the bar on frame (n + 1) as listed as Eqn 8 in the Table, in more familiar algebraic form is:

eqn 18

Note that this equation only applies to the case of a simple moving object. A similar, but different equation will be used for the actual prediction of the PSI.

## 2F. Exposure calculations

The fraction of light attenuated after the ktsvb = (PL - P0), break point depends only on the length of the bar and how fast the bar moves. The maximum exposure attenuation is given by:

Eqn 19

Emax is the maximum exposure in the shutter time ts, E0 is the luminance of an unaffected video frame, such as the mean luminance value of the background; Eb is the static luminance of the (not moving) bar . The fractional portion of the relationship is the image width of the static object image compared to the entire region or pixels that the moving bar effects. All pixel values must be between E0 and Eb, but a moving object can be no greater that Emax in luminance. (see *testprofile.xlsm* for calculation of plot). This formula is also useful with respect to understanding the impact of the vb parameter. If the projector speeds up, (the PSI width on a frame is greater), the fraction is lower, which reduces the PSI luminance of the bar in the frame, which increases Emax of the bar. This is important when we automate the process for removing the PSI, because we need to know the width of the PSI on the fly and how to treat the redistribution of luminances from a standardized PSI distribution provided at a specific width.

For ktsvb < (PL - P0) the peak exposure values are approximately constant over the pixel range from P0 to [(PL - P0) - ktsvb]  - P0  = PL - ktsvb and the exposure is Eb. (Approximately constant because the value may depend on how fast the camera aperture changes with respect to changes in light as the bar moves across the frame row.)

For ktsvb > (PL - P0) the peak values are approximately constant starting at point Pt + (PL - P0) = PL - ktsvb

(2P0 - PL - ktsvb ) and ending at P0, the exposure is defined by:

or

Eqn 20

The exposure curves for each of the cases before or after the point where ktsvb = (PL - P0), will be different.

### 2F.1. Leading edge

For the case when ktsvb > (PL - P0), a point Pxy at the leading edge of the peak, i.e., in range Pt to Pt + (PL – P0) = PL - ktsvb, (peak front) the exposure is given by

Eqn 21

For the case when ktsvb < (PL - P0), the exposure of a point Pxy is given by,

Eqn 22

### 2F.2. Trailing edge

For Pxy at the trailing edge or negative slope of the peak, the exposure again depends on which side of the zero point at ktsvb = (PL - P0) the peak occurs.

If ktsvb > (PL - P0) then exposure in the range P0 to PL is.

Eqn 23

If ktsvb < (PL - P0) the exposure in range PL - ktsvb to PL  is

Eqn 24

Although this set of equations and restrictions looks intimidating, the derivation is a fairly straightforward examination of the model. So what does all this do for us?

Figure 18 is a plot of the four shutter times using the real values found for the Canon Vixia FS 200 from Eqn 18 and the exposure equations 20, 22, and 24.

Figure . Theoretical Exposure and Shape profiles at various shutter speeds.

Note the similarity of these theoretical curves to the actual derived set in Figure 7. There are differences, especially in the values of the baselines. The luminance or brightness values range between 0 and 255. So variations must fit within this range. This suggests we will need to deal with relative fractions of changes and not just absolute areas. If I start with a grey background with a black bar the area cannot be the same, since I am starting at a much higher value. The question is how to represent a relative area. Essentially, the light integration limits change.

At this point, we have been able to rationalize the shape and most of the dynamics of the projector shutter image on video frames and between video frames. The task now turns to use this background information with real projector shutter information to develop a usable luminance profile for the projector shutter image (PSI)

# SECTION 3. FILM PROCESSING

To put the size of the problem in processing film in perspective, 8mm film size specs are 4.01mm/frame. A typical 50 ft roll of film then contain ~3600 frames. However, when recorded using a camcorder running at 30 frames/s, we end up processing ~6000 digital frames, or more precisely digital images.

## 3A. Deflicker process that can be used with the concept of targeted compositing

Before starting it is useful to note a difference between masking and compositing techniques. Masks typically are binary color images that are applied to a video stream. Masks themselves may be a video stream, but the important point is that they deal with only two colors, usually one of which will be defined as transparent when the mask is applied as a filter. Compositing is the mixing of two or more images that can have variations in colors intensities and transparency. Both masks and especially compositing will be used here in post processing of 8 mm film to digital media.

There are several ways to use the preceding information to reduce flicker by directly targeting where the PSI is predicted to be on each visual frame. Once the preferred shutter speed is chosen:

1. Develop a series of full visual frame images of the PSI at fine intervals from the bottom to top of the visual frame from a series of camcorder frames that display only the PSI profile against no or a white backgrounds. From this series, apply the image using compositing techniques that closest matches the calculated position on the visual frame to be corrected.
2. Generate a complete image profile of the PSI on a flat neutral background. Apply only the portion of this image profile that corresponds directly to the portion of the image on the camcorder visual frame. This method requires filtering out uneven visual frame illumination.
3. Avoid any type of PSI image generation and base PSI correction on a mathematical representation of the PSI, which is applied pixel by pixel starting at the calculated position of the PSI on the camcorder frame correction.

Method 1 has the advantage that it directly uses available image information with a minimum amount of frame luminance corrections to reduce flicker. It serves two purposes simultaneously, it will both reduce flicker and reduce vignetting. However, it was not used. At first glance, the number of image files needed should equal the height of the camcorder resolution, which in this case is 1080. That is a huge task and data storage issue to create that many images. In practice, far fewer image files are likely needed. The width of the bands and the luminance difference levels to be corrected are sufficiently low that 108 or less images would likely be needed. Only the most discerning eye would be able to notice any correction problems due to off PSI centered corrections. (Of course, the magnitude of the problem will depend on the shutter speed used.) However, from a digital processing point of view this would be expected to be very cpu intensive. Each vignette image file still has to be created, and during filter correction, each image file would have to be loaded to alter all the camcorder frames with the similar position, the image unloaded, and the next one brought in. It is likely a standardized method for generating the image files could be built once all the luminance variations are understood for all the PSI positions, but developing this filter in addition to the deflicker filter was considered too much effort for the level of this work. In retrospect, this method also suffers from an issue regarding contrast in the image that will be addressed later in this document.

Method 2 requires only one PSI filter image to be created as an image of the projector shutter shadow. Horizontal sections of this PSI file corresponding to the vertical portion visible on the camcorder frame are applied to the camcorder frame. Because of complex vignetting effects, this “idealized” PSI image file must be applied to frames that have had vignetting corrected. Thus, this mode of correction requires two files. An image file to correct vignetting and the PSI image file. Because it is nearly impossible to totally eliminate vignetting from the entire frame, this method will still exhibit some sort of artifacts related to compositing the PSI image.

Method 3 is really an extension of Method 2. In order to obtain the luminance equation representing the PSI, it is necessary to be able to have a low noise vertical cross section of the PSI image. This cross section can then be converted to a mathematical function using various curve fitting routines.

Which method to use? The choice was based on what would be the least computationally intensive method and the easiest for an intrepid person willing to try the method to use, without the mathematical background or lack of memory of mathematical techniques. Method 2 was considered the most likely to be useful for general use.

Another critical variable in how effective we can build a targeted deflicker filter is the shutter speed chosen for recording all the 8 mm film. Fig. 19 displays single frames from each of the shutter speed runs. Note that the 1/500 data is underexposed; the open aperture limit was exceeded. All the rest of the sequences were within the proper exposure. The first three frames clearly show the dark PSI. The 1/60th s frame shows only the leading edge of the peak. The top third of the frame is the white leader. Although it is only just visible, the 1/30th s frame shows a bright band at the bottom representing the transition between successive PSI images on the visual frame. The 1/15th s frame effectively shows no distinctive PSI image at all.

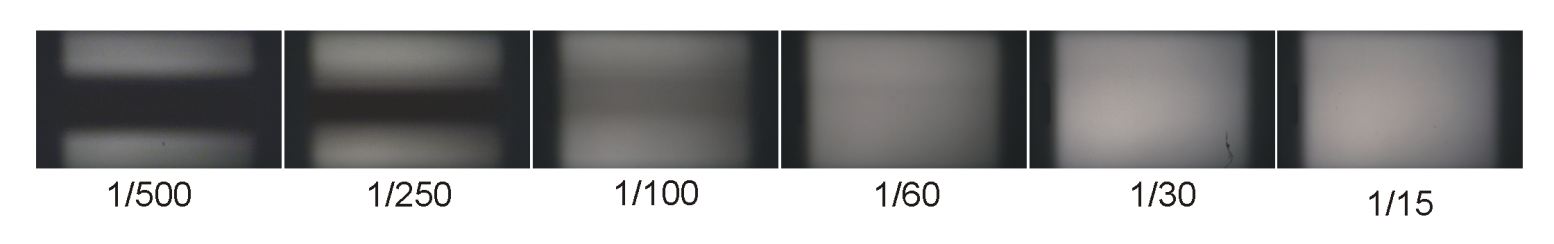


Figure . Single frame images of PSI recorded at various shutter speeds.

There are a number of competing factors which affect the choice of shutter speed:

1. Ensure there is sufficient information content to recover most of the scene detail under the PSI. Clearly, the 1/500 and the 1/250 data do not meet this criteria.
2. At long shutter times, the contrast between the PSI and the underlying scene is minimal, which is a good thing, because the amount of correction to be applied is less. However, the long shutter time also means that fast motion will be badly smeared.
3. For the filter to work effectively, we need to be able to identify where the PSI is on the frame. Although in principle all that is needed is a single point of the PSI on any frame in order to predict the position on all other frames, reality would be that minor variations in projector speed or camcorder recording would lead to a need to provide indexing points along the film timeline to adjust the position of where to expect the PSI on each frame.
4. A shutter speed must be high enough so that the risk of exceeding the camcorder aperture is not exceeded.

Two shutter speeds come close to nearly meet all these criteria. The 1/100 s and the 1/60 s shutter speeds. However, the PSI for the 1/60 s shutter speed does not allow the full PSI luminance curve to be observed. The information on peak position would have to come solely from the peak edges, which is a somewhat less accurate than judging the center of the PSI. This ability to identity where a peak is on a video frame will be needed as a registration marker for gauging and correcting the anticipated PSI position against the true position. Thus, 1/100th s shutter time was chosen as more appropriately meeting all the criteria for the exposure.

## 3A. Developing the Projector Shutter Image (PSI) Luminance Distribution Curves

Now that we have chosen the method to apply to removing the PSI, we have several issues to consider and control before we can routinely apply a correction to remove the PSI from the camcorder frames:

1. The width of the PSI at various camcorder shutter times to determine optimal recording shutter speed
2. The optical vignetting due to the original film camera and the projector
3. The shape of the PSI at various camcorder shutter times
4. The cycle time between appearances of the PSI

Again, we need to simplify or control these aspects of the recording and process to establish the best set of correction parameters and conditions under which to record the film images for further processing. Trying to establish these from a film with real scenes is very difficult. Much of what we need to understand is best learned when there is no scene. There are a couple of ways this might be done. One way would be to shoot a portion of a film against a uniformly lit bright white background. Unfortunately, most film probably was never taken with this idea in mind, and the camera may no longer be around. In addition, ensuring a uniform light field is not going to be easy for most people. The next best alternative is to run the projector with no film, and record the clear white image on the screen. (David Graft, <http://neuron2.net/hotspot/hotspot.html>) This was not possible with my Bell and Howell 487 projector, which has a drop down mask when no film is in the projector or the motor is not running. The mask prevents the film from being burned by the projector bulb. The next best alternative is to use the white leader at the beginning of a film to produce a uniform white image from which the optical light changes in the visual frame can be taken. This method will not compensate for any camera optics vignetting, but camera vignetting is likely much less then the projector vignetting (at least true of my equipment). Of course, the problem with the film leader is that it typically is not very long, usually about three feet, so we do not get an extensive sequence of camcorder frame images to work with. The last method of correcting the luminance variation is to create an artificial image that represents the vignetting using a vector or image editing program (See the D. Graft reference above for how he suggests doing this.) For this work, using the white leader was considered the most expedient method for deciphering the PSI luminance distribution and position information. From this information, all four of the above factors could be determined.

### 3A.1. Transferring film to camcorder – recording the film

It is finally time to establish the recording process for the film. The reason for some elements of this process will not become totally clear until later, but the process must start with obtaining a consistent set of recordings to manage. The following rules for recording should apply.

1. The finer grain the screen to project an image, the better. If it is possible to use some sort of telecine type of setup to record the film, all the better.
2. A room that minimizes extraneous light sources should be used for recording.
3. If possible, set up the distance between the screen and the camcorder so the camcorder does not have to be zoomed. (However, see rule 4.) Alternatively, If at all possible, all recording should be done at the same time with all the equipment set in the same way. Do not turn off the camcorder until the recording process is finished. The reason is that when the camcorder is turned off, the zoom is likely to return to its original neutral settings when turned back on. This If the camcorder must be shut down, some means to establish the same camcorder-projector-screen positions and camcorder zoom setting of the camcorder to screen must be done, such as registration marks on the screen, so that the zoom can be reset to the same conditions as previously used. If this is not done, a completely new, independent analysis of the white leader data might be required to determine the PSI width and cycle information for the next set of recordings.
4. Minimize vibration as much as possible. Placing the camcorder on a tripod so it remains isolated from the projector is a good practice.
5. The camcorder lens axis should be coincident with the projector’s lens axis as much as possible. The camcorder should be as close to the projector as possible to avoid distortion when recording from the screen. (Fine for silent films, but may be a problem for movie cameras with audio. This may entail a separate recording for sound.) A distance of at least four to six feet from the screen is good.
6. The camcorder should be zoomed in sufficiently or placed so that that some portion of the top and bottom image will likely be lost. How much should be eliminated is determined by the vignetting effect at the top and bottom edges of the film. Often the light fall off is rapid and clearly visible. Eliminating a portion of this region now will substantially reduce artifacts in the vignetting correction that will be applied to the recording. The amount of information lost, in most cases, will not likely make or break the quality of the image anyway.
7. No post editing of the camcorder video can be done to remove or add frames. If pre-editing is desired, then each video segment remaining before and after a removed sequence must be saved as a separate file.

#### 3A.1.1. White film leader recording and data processing

Figure 20 shows the simple set up used to obtain both the leader and projector image recordings. The camcorder is mounted and leveled horizontally on a tripod as close as possible to the projector so that the projector and camcorder lens axes are at the same height to minimize vertical distortion. There will be some horizontal distortion, but in retrospect it was minor for this case. The center to center lens offset between the camcorder and projector was 10 3/8” . The distance from the projector lens to the screen was 60”. The screen at the far end onto which the film is projected was the same as was used for the outdoor measurements – a white, fine texture poster board mounted to a ¼” thick plastic sheet. The projected image size under these conditions was 11.5” x 15.5”

Figure . Camcorder and 8mm projector set up for transferring film to HD digital video.

Parameters for the Canon Vixia HF200 camcorder:

* Memory configuration: Class 10 16 GB SDHC card in slot A and 8 GB Class 6 Eye-Fi card in slot B.
* Set camcorder switch to Manual mode
* From Menu on screen key:
  + Under the film icon: rec mode->mxp 24 mps
  + Frame rate -> pf30
  + Focus assistance-> on (using peak focus mode)
* From Func on screen key:
  + Rec Programs-> Tv-> 1/100 s (or whatever shutter speed)
  + Mic level->manual-> 0
  + Set zoom so that vignette at the top and bottom edges are mostly off the screen to reduce major vignetting.
  + Manual focus preset: Tap the on screen eye-like icon to activate the manual focus screen, ensure peak focus is engaged. Run a trial film and tap to to set the focus. Touch the “X” to close the manual screen and lock the focus setting. The peak focus lines should always register either the edge of the frame, or objects on the film.

On the Vixia, what the small touch view screen sees is what you get, but even zooming the image past the top and bottom of the frame was not always as effective as I hoped.

The problem is that if the camcorder is turned off, the zoom function resets to normal position. The Vixia has no readout of the zoom factor, so it could not be reset to exactly the same position. There can be a lot of fooling around to reproduce the exact same zoom factor. Variation in the zoom setting affects the effectiveness of removing the vignette and affects the PSI width.

Typically six 50ft super 8mm reels were recorded to the A card. Each file was ~ 590 MB +/-20 MB at 1910x1080 resolution. The six files were then copied to the B Eye-Fi card, which was set to automatically upload the files to a computer. The upload was a slow process, but did have the advantage that there was no connecting and disconnecting from the USB port to disturb the camcorder setup, and no wires to trip over. (Transfer was to a PC desktop system with plenty of available drive space to handle the files. Once the upload was completed, the files were erased from both cards and the process repeated, until all the films had been recorded.

It was not possible to finish all the recordings and uploads in one day. The zoom was set to the same position as best it could be, but there was some variation.

To record the projected film, the camcorder was set to manual operation at maximum resolution and progressive frame recording as indicated in the settings. The camcorder zoom was set so the projector image filled the camcorder visual frame at the top and bottom. The projector image was then horizontally centered on the camcorder visual frame. Because of the aspect ratio difference between the film and camcorder, there were black regions on either side of the projected image. Often before a film was recorded an initial portion was run to make sure the position and focus were set properly. The film was then run backward to start the final recording focus. This was especially useful for recording just the white leader, because the leader was around three feet long, which is only a few seconds of recording time.

For some early leader work, the camcorder video sequences at different shutter speeds were combined into a single video project using Sony Vegas Movie Studio and text frames added between the sequences. The properties of the output format were set to progressive frame, 1920 x 1080 images and 30 frames per second. (Any video editor that will allow output to picture image files with these conditions is fine, Vegas was the normal editor used for most work.) For the leader work the output was converted to individual \*.png files covering 1/500, 1/250, 1/100, 1/60, 130, and 1/15 second camcorder shutter speeds.

The PSI image sequence shown previously as a comparison of the images for different shutter speeds represents single frames from each if the different shutter speed sequences. As already noted, based on this information, the standard shutter speed chosen for this work was 1/100 s.

***VERY IMPORTANT: THE ONLY EDITING ALLOWED ON THE DIGITIZED FILM IS REMOVING FRAMES FROM EITHER END OF THE VIDEO. NO FRAMES, NO MATTER HOW BAD, SHOULD BE EDITIED FROM THE MIDDLE OF THE RECORDED VIDEO STREAM.***

The reason for this is because we will track the position of the PSI on each frame and expect the sequence to be continuous. Editing will cause the PSI centers position to be wrong. Post editing certainly can be done once the PSI correcting process has finished.

### 3A.2. Correcting optical vignetting

The next task is to correct for vignetting, which are dark areas at the edges and corners of the images. This is absolutely necessary, because we assume a PSI template with 2D uniform luminance values across the video frame. My projector had no provision for aligning the bulb with the optical axis, and therefore it was necessary to compensate for vingetting using an image filtering technique. A filter mask was created that uses a single representation of the PSI profile and the profile will be applied across the entire horizontal film frame. If we do not have a reasonably flat horizontal luminance field, the filter will not optimally remove the effect of the PSI.

In the world of filters, there are a number of ways to combine or composite images. Some very basic filter operations for compositing images are multiplicative, additive, alpha source, and subtractive. (Alpha refers to the transparency channel.)

The following equations are fairly standard for defining what these compositing methods do, where source = src is the “mask” or top layer; destination = dst is the image to be corrected or acted upon.

**Alpha Blending:** (*src* ×α + (*dst* × (1 - α); α is the amount of transparency, with range between 0 and 1.

**Additive Compositing:**  *src* × as + *dst* × ad; as and ad are mixing factors. If they both equal one, this is a straight addition of the src and dst file values. (However, limits cannot exceed 0 or 255 in the rgb system.

**Multiplicative Compositing:** *src* × 0 + *dst* × src

Many programmers seem to use these terms in the fashion defined above, but you may need to be sure. Also note that masking and compositing are related to each other, but typically a mask allows only two “colors” – such as black (rgb, 0) and white (rgb, 255). The filter process needed here is compositing, and the image generated to remove the effect of vignetting is referred to as the filter image, rather than a mask.

Additive compositing, used with an inverted source or mask file is the most appropriate. In such a case, black (a 0 rgb value) will not add any further luminance (brightness) to the dst file, while values greater than 0 will lighten the dst pixel.

There are a number of methods to remove or reduce the vignette. In a perfect world, the projector will not have a drop down shutter mask to prevent the film from being burned, and the camcorder can thus directly record the vignette pattern off a white surface. In a less perfect world, there exists a piece of film that consists of recording nothing more than a white surface. Beyond these approaches, the process of creating a filter gets more complex. The most desperate method is to artificially represent the vignette using a generated gradient image of some sort, usually as part of the software package for video or image editing. Building up from scratch is tedious and may work well in the hands of an expert, especially with a vignette that exhibits a symmetrical image. Alternatively, a filter image can be built up piecemeal from sections of visual frames not affected by the PSI, using an image editing program to splice the sections together; this method is very laborious and requires extensive smoothing operations at the splice junctions; it is a last resort approach.

In principle, a simple method is to concentrate on the recorded images of the white leader at 1/15 s or lower shutter speed. At this shutter speed, the PSI band is so broad that it covers over 3600 pixels, which is far more than the vertical frame resolution of 1080 pixels. This represents an averaging process over three or four passes of the PSI image over the video frame. The result should be a fairly good representation of vignetting. Alternatively, if insufficient low shutter speed information is not available or there appear to be different luminance distortion between different shutter speeds, then an image of the white leader background at faster shutter speeds can be used with an averaging process. As it turned out, this latter case ultimately was used for the final film processing, despite the advantage of the low shutter speed method.

The easy and direct way of isolating the leaders from a longer .mts file that can be directly used in ImageJ is the following procedure:

If you have not already done so, download and install VirtualDub. Even if your system is using a 64 bit operating system download and install the VirtualDub 32 bit version. Some filters, such as the Hotspot filter, will not work in VirtualDub 64. The 32 bit version of VirtualDub will work in 64 bit Windows 8.1 and the Hotspot filter works.

With VirtualDub installed, download and install the latest codec package "ffdshow tryout" to you system. Also, find the 32 bit DirectShow driver plugin for VirtualDub, so that you can directly open the AVCHD files, such the native camcorder format for the Vixia, which was .mts (see <http://forums.virtualdub.org/index.php?act=ST&f=7&t=15093>). On my 64 bit Windows 8.1 system, I kept getting an error message, "unable to set file on media detector...". Just why this error comes up is not clear; a number of plugins were downloaded to fix the problem. As too often happens, one or more of the plugins worked, but which one I am not sure. At any rate, make sure the following plugins are in your Virtualdub plugins folder: FFInputDriver (place the dll folder and the plugin in the plugin32 folder. Also, if the plugin ffvdub.vdf is not in your plugin32 folder, search for it (likely will be in the ffdshow folder somewhere and copy it to the VirtualDub plugin32 folder. With the Directshow plugin and maybe the other pluginsinstalled, the video files could directly be opened in VirtualDub. After installing the plugins recommended above, just try opening the *.mts* file normally. However, you will find alternate methods for opening *.*mts files on the internet that discuss a strange two stage process. First, to search for an .mts file, use *file type “All Files”.* Click on the file so that it appears in the File Name textbox, then open up the *File Type* textbox and choose the *DirectShow* type option before opening the mts file.

Another tactic, would be to first convert the *.mts* file to a HD progressive scan avi file using your favorite movie editing software; you should then be able to open the avi file in Virtualdub.

Several VirtualDub filter plugins will be needed, specifically, the Hotspot filter and the Image Layering filter. It is suggested you download a package of VirtualDub plugins, such as at <http://codecpack.co/download/VirtualDub-Filter-Pack.html> (use the installer version for convenience), If using the individual zip files extract the contents to the VirtualDub plugins folder.

Unzip the compressed files directly into the VirtualDub 32 plugin folder.

Once an .mts file is loaded into VirtualDub, there are a several ways to isolate a sequence. Scroll to the start frame of the stream to be extracted, using the slider on the bottom of the VirtualDub window. Click the left half arrow marker. Go to the end of the desired sequence and click the right half arrow marker. (For more information, there are numerous internet tutorials on this.) Check if *Video|Full Processing Mode* is engaged*.* Check *Video|Compression…* is set to *Uncompressed RGB/YCbCr*. Assuming there is no audio with the movie, save drive space by preventing audio output by clicking on the *Audio|No Audio*. Finally, use *File|Save as avi…* to save the highlighted sequence to a folder.

An indirect method is to first render the video as an uncompressed .avi file with other editing software that can handle the .mts format.

As an example, the video file can be brought into Sony Vegas and the following actions performed:

1. Remove the audio track: |Right click on the audio track*|Group|Remove from|*click Delete key
2. Trim the video at the front and back as needed in Sony Vegas Movie Studio and render as an avi file:

To render the file as avi:

1. *File|Render as*:
2. Provide a file name dump to C:\Canon\avi\_in such as *leader bkgrd 100th\_1 (to \_3)*, click Custom… and use following settings:

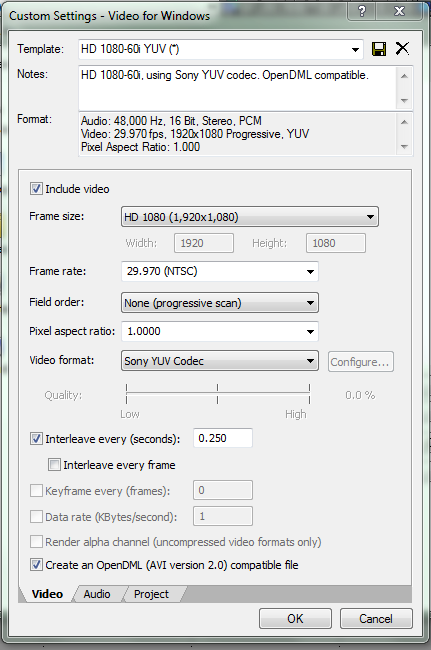


Figure . Render setting in Sony Vegas Movie Studio.

Click on *Audio* tab on the bottom and make sure the “*Include audio*” button is unchecked.

Click on the Project Tab and makes sure video quality is set to “*Best*”.

The leader videos displayed a gradual color shift from yellowish-grey to nearly pure grey that occurred over almost 100 frames. This was not corrected. If the color shift is unacceptable the first 80 frames or so would have to be eliminated to reduce the color shift. Of course, this means averaging only 80 to 120 frames.

ImageJ refused to read any of the avi rendering formats that are available in Sony Vegas Movie Studio. The errors were either an unrecognized compression or that there was more than one frame chunk per block. It was necessary to convert each Sony edited avi file in VirtualDub and resave the file with the default VirtualDub options (appending a \_x to the file name, where x was the file number to further distinguish the original and final). This VirtualDub avi files could then be opened in ImageJ. Before saving the avi file, turn off the audio in VirtualDub (Audio|No audio). Thus, the direct method described first is preferred.

If using a very low shutter speed, it is possible to use most of the individual video frames as the starting point for the filter image. The only issue is to ensure that a frame that shows a bright “valley” between PSIs is not chosen. A better method, which can be used with any leader sequence at any shutter speed, is to average the entire leader video sequence to obtain the filter. The only caveat to the averaging process is to have sufficient leader frames to completely smooth out the PSI images, for instance, averaging over 100 frames. The averaging process has the advantage of also eliminating local dust spots. Using ImageJ, the averaging process is straightforward. Once the avi file was open in ImageJ, either as an image stack or virtual stack, the average frame intensities for the entire stack could be obtained using *Image|Stacks|Z-project…|Average Intensity*.

Three separate leader sequences were used to generate three separate average intensity image files. The average image was saved to a new folder, e.g.,“bckgrd” as a .png file, with the last character in the file name being a sequence number. Finally, these three image files were brought into ImageJ using either *File|Image Sequence,* to obtain the folder name, if they were the only files in the folder or opened separately in ImageJ and converted to a stack using the *Image|Stacks|Images to stack* function. The three files were averaged using *Image|Stacks|Z-project|Average Intensity* again. This repetition of averaging may have been overkill, but it did result in a smooth image with no dust artifacts. Of course, if the vignette varies a great deal from video to video, each video will require its own background correction image.

Deciding on a background image is not necessarily without unexpected results. For instance, initial exploratory work using 1/15th sec videos to generate the filter image worked very well. However, when generating the final filter image that was to be used to correct vignetting for most of the recorded films, the 1/15 s image file unexpectedly did not have the same vignette pattern as all the rest of the film video. The hotspot was shifted down by over 500 rows. Of course, it was impossible to get good vignette reduction with this filter image version. Why this suddenly occurred was unknown, but such odd behavior must be kept in mind as the process proceeds. This kind of problem highlights the need for the user to not treat the entire process as a cookbook , faultless method. In the end, the 1//100th leaders from the recorded films were used to generate the filter image.

Residual noise was further reduced using the ImageJ Gaussian Blur filter (*Process|Filters|Gaussian Blurr* with a sigma of 3.0, and the file saved as a .png file. Depending on whether you want to preserve or alter the color temperature using the vignette, you can continue with this color version or convert it to a greyscale image *(Image|Type..|8 bit)*

At this point we have the single image that will be converted to the filter image. Below is a false color image of the filter image that enhances the grey level variation to better expose the vignette problem; the darker the magenta color, the more light intensity fall off. The darkest right and left "bars" are due to the different aspect ratios of film versus the HD camcorder settings. The vignette pattern is clearly complex.



Figure . False color image of vignette pattern of projector.

With this information in hand, generating a mask to compensate or filter out vignetting was possible.

Unfortunately, my version of Sony Vegas Movie Studio Suite only allowed alpha source changes and multiplicative compositing. (Sony Vegas Pro versions have more compositing possibilities for combining image and video files.) These two compositing modes cannot perform the necessary operations. However, the Hotspot filter for VirtualDub ((David Graft, <http://neuron2.net/hotspot/hotspot.html>) was especially written for removing the vignetting caused by projections. This filter has both a multiplicative and additive mode. The additive mode is a bit altered from the above definitions. After examining the source code, the equation that describes the operation that is performed on each pixel for every video frame is:

target = dst + dst x src x (Multiplier/40000) + src2 x (Adder/100000) Eqn 24

Target is the final output pixel or collectively, the final altered video frame; dst is the original uncorrected video frame pixel; src is the inverted image file that represents the vignette geometry as a 24 bit bmp file loaded into the filter. The Multiplier and Adder are values (0-1000) derived from the Hotspot filter window sliders. The equation is shown here because we will use it as a method to predict and monitor the goodness of correction and find out what initial slider values to start with.

The process of converting the raw filter image to an inverted filter to reduce vignetting is tedious. To develop the vignette image from the average filter image, it is necessary to determine the maximum and minimum luminance values on what is considered the usable portion of the video frame that will be corrected on the image. We then transform that range into an inverted image that will have at the maximum value (hotspot) a value of zero and at the minimum rgb value (coldspot), the difference between the maximum and minimum.

The first part is straightforward: A rectangular region of interest (roi) of the usable portion of the image is set. From viewing the film, this region is bounded by X,240-1680 (1440 wide);Y,0-1080. The Y range that was used in the final frames was somewhat smaller, but this is acceptable.

As an example, in the false color image displayed above, the hot spot value was found to be 135 rgb units at (x,y: 620, 570) and the coldspot value was estimated at (x,y: 1700, 1080) with an rgb value of 47. Thus, the range of variation from hotspot to coldspot is 135 to 135 – 47 = 88. On the inverted filter image to be created this will translate to 135 -> 0 hotspot value and a coldspot value of 88.

The inverting and rescaling of the filter image to the final filter image was done in CorelDraw. It is likely that this can be done in ImageJ, but I was not able to find a suitable combination of functions to mimic what could be done in CorelDraw. Other image editors such as Photoshop or Gimp will have similar capabilities to CorelDraw.

The detailed process steps to develop the source (filter/mask) file:

|  |  |
| --- | --- |
| **Step** | **How to create and validate the filter file:** |
| 1 | Find the useful x and y ranges that will be the final visual field of a typical the video frame. This operation is a tradeoff between eliminating the darkest edges that still contain some useful visual information, but also create a very steep luminance gradient that is very hard to correct for without causing general color distortions on the more important regions of the video frame. It is better to lose some of the video frame edge than try and recover the luminance information from really dark areas. (if you are uncertain about how much to leave out, you can decide what is appropriate by iteratively using the regression method below with and without the edge information.) |
| 2 | Isolate where the maximum and minimum pixel values are on the video frame. There a many ways to do this, but a simple, effective method uses ImageJ. use the *Image|Adjust|Color Threshold…* function with the *black background* box checked, or use Image|Adjust|Threshold, if working with greyscale image*.* The sliders are moved until the hotspot region is isolated to a small set of points, from which a guess at a reasonable hotspot coordinate can be established. When the point is isolated, either manually note the position. or tag it with the point tool and use *Analysis|Measure* to print out the hotspot position.  For the coldspot position note where the vignette gradient is the highest (darkest) and follow that line to what will be considered the edge of the film image that retains enough luminance information to reconstruct the image. Uncheck the *black background* box and repeat the process above to isolate the coldspot.  There will often be several close spots with the same hotspot maximum. You do not need to exactly find the hotspot or coldspot, but you should be reasonably close. This information will be used for determining the starting values for the multiplicative and additive sliders on the Hotspot filter. |
| 3 | In ImageJ, switch to the line tool and shoot a line from the maximum luminance point (hotspot) to the minimum point. Use *Analyze|Plot Profile* to generate the pixel values along the curve. Copy the profile data to an Excel spreadsheet. |
| **4** | ***Creating the filter image***  In many of the steps below, specific software operation descriptions are provided for my own ease of use, and reproducibility; similar functions exist in most image editing software, such as Photoshop, Gimp or VirtualDub, but how you get to the option and what they are called might vary. This was even true for different versions of Corel Photo-Paint. |
| 5 | I was unable to figure out how to use ImageJ to normalize the values of the filter, but I suspect this should be possible. Instead, the cleaned file generated in the previous step was brought into Photo-Paint and the following sequence of transformations applied. Note that different versions of Photo-Paint (using X5) may have the function under different menu tools, but the operation is essentially the same:  **Set the info docker.** *Window|Dockers|Info.* This will allow you to run the cursor over the hotspot and coldspot as you change the contrast inputs and outputs to see if the process is working at least for these two points.  **Invert the image colors:** *Image|Transform|Invert*  **Convert the image into a grayscale image**:  *Image|Convert to…|Greyscale 16* .  **Shift the grayscale to a rgb 0 base color:** *|Adjust|Contrast Enhancement...* (In Gimp, this is called the Contrast Tool.) This filter has four inputs adjusted by two sliders, each with two tabs. The purpose is to shift the grayscale luminance values to end up with zero at the pixel representing the hotspot (maximum brightness pixel) and minimum brightness pixel near the coldspot value found in Steps 2 and 3 above. As an example, the maximum and minimum rgb values in the cleaned 1/100 s image file were found to be 135 and 47, respectively. The difference between these two values is 88.  Thus, the range to achieve is a hotspot of 0 and a coldspot of 88. All four inputs were adjusted to produce this range in an iterative process of changing the values and noting whether the hotspot pixel (around (x,y: 620,570) and the coldspot pixel (around x,y:1700,1080) were at values of 0 and 88, respectively (using the *View|Image Info* tool in Corel Photo-Paint). I was not able to find a simple relationship to predict what the sliders should be set at; this is a trial and error process that can take some time to get right. An actual view of the contrast enhancement final settings are shown below.   |  |  | | --- | --- | |  |  | |
| 6 | When satisfied with the fit in Step 5, the final negative image is converted back to a 24 bit image using *Image|Color Mode|RGB 24 bit.*  Next, the Gaussian blur filter with a radius of five was used four times consecutively to smooth out the image rgb values. The blur filter is a very important step, especially for Hotspot filter use. Apparently the multiplier factor of the Hotspot filter can easily magnify small differences between nearby pixels leading to a mottled coloration of the image. This was especially evident where the vignette was most pronounced, which suggests the multiplier factor is the culprit.  Finally, the result is saved as a .bmp file. The file *must* be saved as a .bmp file, and represents the source or mask to use for the Hotspot filter, or in ImageJ. An example of what the filter image file might look like is shown below.    Filename: *Projector Leader smoth invrt cntrst enhcd 24 bit 091611\_000888 102011.BMP*  Of course, the less the vignette effect the more uniform grey this filter image would be. |
| 7 | ***Checking the goodness of fit***  There are two methods to check the ability of the newly created filter image to reduce the vignette. One is straightforward and uses ImageJ, the other uses the Hotspot filter in VirtualDub32.  ***ImageJ check***  Load one of the original leader avi files into ImageJ as a virtual stack. Also, open the new filter image. Click *Process|Calculator Plus…|(Choose) Add function*. Do not change the constants. Note the two files in the textboxes. If the only files open are the two mentioned, these will already be set in the boxes. Clicking Ok, every image in the avi file will be filtered. To examine the goodness of the correction, use the line tool to run a horizontal line across the frame. Click *Analyze|Plot Profile* to see the horizontal profile. Click the “Live” button. As you scroll through the avi images with the mouse wheel or arrow keys, the profile plot will update itself. You can also move the profile line to different vertical positions to get an idea of the effectiveness of the filter at different positions. The maximum variation noted in most of the frames was less than 20 rgb values. If the values are well beyond this, it is necessary to tweak the contrast enhancement values a bit to see if a better fit can be achieved. |
| 8 | ***Calculating the Multiplier and Additive Slider Values and expected goodness of fit for the Hotspot.***  The following multivariate regression analysis is not strictly necessary, but can be used to predict the set of multiplicative and additive values to use as starting points for the Hotspot filter sliders. Depending on the complexity of the vignette pattern, and the mathematical model behind linear regression models, this process may or may not work well. Note this procedure requires that the Analysis Pak plugin be available in Excel. If you have made it this far, you probably are very familiar with the package anyway. Skip to step 13, if you wish to bypass the regression calculations. |
| 9 | Open the newly created inverted filter file and the original averaged leader frames image in ImageJ and run an analysis line from the hotspot to the coldspot on each image. Use *Analysis|Profile Plot…* and copy the data into the same Excel file. (Any profile line can be used, if it seems more appropriate to use to provide a better predictor of goodness of fit.) Just make sure the line used starts and stops at the same pixel positions for the two image files. |
| 10 | You can use all the pixel column data, i.e., horizontal (x) pixel position, original 1/100th averaged image file, and final filter image file, but it may be quicker to use a subset of the points, such as every 20th point. If the starting row of the data in Excel was row 21 and the pixel X position was in the first column then the formula, =OFFSET( $A$21,(ROW(A21)-21)\*20,0), placed in some other column and swept down that column would print every 20th row. To get the pixel data, sweep this formula across several columns and replace “A” in the second term in the formula with the column letter with the desired data column letter. The final data point to sweep down the column should be only to the point chosen as the useful video frame field. There will be far too much bias in the resulting calculation otherwise. |
| 11 | To test the predictive values to a first order extent, a multivariate regression analysis is used based on a summary of the equations used in the Hotspot filter code already discussed. The equation to minimize or regress is:  Pf- Po = Po\*(Multiplier/40000)\*Filter + (Adder/100000)\*Filter^2    Pf is the hotspot luminance value on the original image file to which all points will be corrected; this value is determined from ImageJ by duplicating (*Image|Duplicate…* ) individual avi frames that clearly show the PSI valley (brightest area) in the hotspot region, converting the image to 16 bit (*Image|Type|16-*bit) and reading the rgb value at the hotspot coordinate by setting an roi around the hotspot area and using *Analysis|Measure* to provide the minimum rgb value in the identified hotspot region. Averaging several such points from different frames provides the maximum luminance value. Po is the pixel color value of the original averaged image at an x,y coordinate, to which we will apply the correction; Filter is the corresponding pixel luminance value of the generated inverted filter image file at the same or nearly same position as the averaged image. Multiplier and Adder are the coefficients to find and hopefully will represent the starting slider values in the HotSpot VirtualDub filter. |
| 12 | To do the regression analysis, the above equation is split into three terms. In addition, to the x (horizontal) position, original pixel image profile, and filter profile columns representing the track from the hotspot to the coldspot, a column of Pf-Po values (Y values) is generated,. In the next two adjacent columns, the value of Po\*Filter/40000 and (Filter^2)/100000 are calculated, respectively, where Filter is the rgb value of the corresponding inverted filter image profile row. The Excel Analysis Pack R*egression* plugin was then used to determine the values of the additive and multiplicative sliders. To do a multivariate regression analysis, the y values are set to the (Pf-Po) values range; the x values are ranges of the two adjacent columns just described. Choose the options to force the constant term to zero, print out the residuals, and to plot the variables to see the goodness of fit. Once the regression is performed. the two factors, Multiplier and Adder, can directly be read off as the coefficient of variable x1 and x2 along with the fitting statistics. |
| 13 | The final step is to use the mask against some real video data. If not already on your system, download and install VirtualDub as described earlier. The Hotspot filter will not work in VirtualDub 64, thus if your system uses a 64 bit operating system, also download and install VirtualDub 32. The 32 bit version will work in Windows 7-8.1 64 bit and the Hotspot filter also works.  Although the *.mts* file can first be converted to an avi file, it is an unnecessary step. To directly use the *.mts* file in VirtualDub, just try opening the *.mts* file normally, if you have heeded the earlier directions to installing VirtualDub, and the extra drivers and plugins to load *.mts* files. Alternatively, it may be necessary to first set *file type* to *“All Files”.* Click on the *.mts* file to open so that it appears in the File Name textbox and then open up the *File Type* textbox and choose the *DirectShow* type option. Clicking Ok should now opening the *.mts* file in VirtualDub.  Next, click on *Video|Filters…* and choose the *Hotspot* filter. Set the bmp filter file, for instance, in my case, *Canon/Overlays/Projector Leader smoth invrt cntrst enhcd 24 bit 091611\_000888 102011.BMP.*  To scan the sliders while visualizing the best fit, choose an image in the video stream and click *show preview* in the Hotspot filter window. Shrink the image preview to whatever shape/size you want and move the sliders; the image will change to show the effect of the filter.  An avi file consisting of only the 1/100 s white leader was opened in VirtualDub and the Hotspot filter applied with the sliders values indicated above. One must be careful adjusting the sliders because the black left and right areas can throw off the eye’s ability to find the best values. In addition, it was noted that once the filter was applied, the edges of the PSI where not quite as sharp as the initially seemed. This means that when the final black and white mask is chosen to restrict the video viewing area, the area would have to be even a bit more restricted than found in the early steps of this procedure.  Set the following parameters in *VirtualDub:*  *Video|Full Processing Mode*;  *Video|Compression...|Uncompressed RGB/YCbCr),*  *Options|Performance -* if you have the memory available, see if you can set the sliders to higher buffer numbers than the default;  *Options|Preferences|Process Priority to Highest* (for fastest conversion rates);  *Audio|No Audio*.  The script *HotSpotsetup.vdscript* can be loaded (*File|Load Processing Settings...)* after or before the video file, and automaticall*y* set the above parameters.  There are other parameters, such as frames to process that you may wish to set.  Finally, click *File|Save as avi...,* and save the file with a suitable file name, I usually appended "corr" somewhere in the file name. When the processing starts. uncheck the *Show input* and *Show output* checkboxes in the status window; this will help speed up processing. It will take ~8 minutes to complete the filtering on a typical 50 ft roll of super 8 film transferred to video. |

Using this regression method, the starting multiplier and adder coefficients for the Hotspot filter had an r2 value was 0.94, which at first glance is a moderate correlation. The Multiplier and Adder values identified were 492 and 270. However, these values were not acceptable in practice. The values that were finally chosen by moving the sliders and observing the changes to several different preview frames were 551 and 691. The Multiplier is not too far off, but the Adder is way off. On different videos, this regression methods was good to mediocre in predicting adequate Multiplier and Adder values. The reason for this has to do with the limitations of linear regression analysis and the shape of the vignette curves, and will not be discussed further.

One of the best methods for optimizing the slider settings was to observe a leader PSI on several frames and adjust the sliders until the bands had nearly the same intensity horizontally at every row. Because of noise in the pixels, even this method is not foolproof, but still provides a more uniform frame luminance than using no vignette correction.

The example image on the left below shows an uncorrected leader frame and on the right is the corresponding Hotspot filter corrected image. The edge vignetting has improved markedly, but there is an obvious dark “wisp” artifact in the middle of the image.

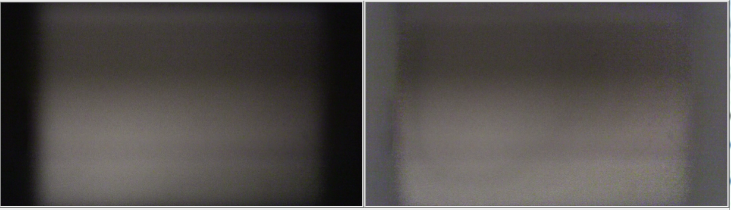


Figure . Comparison of video frames uncorrected (left) and corrected (right) for projector vignette.

Below is a profile of the corresponding horizontal slice at row 606 for each of these frames; this was picked as a worst case where the artifact exhibits a low rgb value. Not all videos were as bad as this.

Figure . Horizontal profiles of frames uncorrected and corrected for vignette effect.

Clearly, the filter substantially reduced the vignette. The region between the vertical bars at 270 and 1700 represent the limits that were initially chosen as the potential video frame size that might be achieved. The difference in rgb value between the center artifact at x = 1077 and the brightest pixel at x = 612 is 13 rgb units. So despite what the image shows, the actual difference is fairly small. From a practical point of view of a constantly moving image, the artifact may be even less obvious. However, the right limit of 1700 shows a difference of 27. This means we will likely need to mask at a right position somewhat less than 1700. However, the steep curve means we will not lose much of the image field; a limit of 1650 will drop the difference to nearly the same value as the center rgb difference.

An alternate method in VirtualDub is to use the “Layer” filter. This filter can directly add the .bmp filter image to each frame of an .avi or .mts stream, but does not offer a way to adjust brightness across the frame. Which filter to use depends on the quality of the initial filter. In this specific case, it turned out that the Hotspot filter provided a somewhat more uniform overall horizontal vignette removal. The original filter still showed residual vingetting on the right side. The rgb differences is the latter case were still under 20 rgb, but the Hotspot correction seemed more pleasing in the current case.

ImageJ was not used for vignette correction of the film sequence. The method failed on a typical avi from a 50 ft roll of 8mm film because of stack memory problems. (In retrospect, the reason it failed is a naive ImageJ script writer at the time, this filter process was developed. ImageJ can be used if the avi file is opened as a virtual stack with a script to to the correction. However, VirtualDub would still be much faster at filtering, because it is using a compiled plugin filter.

For routine use of the Hotspot filter in VirtualDub:

1. Start VirtualDub. To use VirtualDub at this stage, you should have already added all the filters and plugins as outlined in installing VirtualDub as explained previously.
2. Load the script *Canon/virtualdub stuff/Hotspotsetup.vdscript,* then load the *.mts* file.

Go to *Video|Filters* . When the parameter window opens, browse to the vignette filter *.bmp* image file *(e.g.,) Canon/Overlays/Projector Leader smoth invrt cntrst enhcd 24 bit 091611\_000888 102011.BMP*

1. Change the sliders to your defined multiplier and additive parameters that you have optimized in the preceding process. Click Ok.
2. Set the following parameters in *VirtualDub: Video|Full Processing Mode*; *Options|Performance;* if you have the memory available, you may wish to set the sliders to higher buffer numbers than the default; set *Options|Preferences|Process Priority to Highest* (for fastest conversion rates); *Audio|No Audio*. There are other parameters, such as start and end frames to process that you may wish to set using *Video/Select Range..*. Optionally, you may wish to save your settings to use later. Click *File|Save Processing Settings...* The *File|Load Processing Settings* can be used to bring the saved setting back at any time.
3. Click *File|Save as avi...,* and save the filtered avi file with a suitable file name, I usually appended "corr" somewhere in the original file name.
4. When processing starts, uncheck the *Show input* and *Show output* checkboxes in the status window; this will speed up processing a bit. Filtering will take ~9 minutes to complete for ~6000 frames. The avi file size at this point will be as high as 40 GB.

### 3A.3. Extracting the PSI profile image

Once the distortion due to vignetting is reduced on the leader frames, the next operation is to extract the PSI horizontal profile image. From the first views of the PSI image near center frame it was discovered that the PSI for the 1/100th s shutter speed was wider than the visual frame. Thus it was necessary to build up the image from separate images presenting different regions of the PSI. The real use of the Solver method is to refine the PSI profile and luminance corrections.

After removing the vignette using the previous section process, for each leader avi the first 20 frames were not used. Setting the limits was done as discussed earlier by setting the left and right markers in VirtualDub.

There are two process that can be used to determine the PSI profile and a subsequent frame luminance correction factor that will be discussed later. The first that will be described was originally how the PSI profile was manually ecked out of the leader frames. The second process is based on a complex regression method using Excel's Solver function to predict the shaped of the PSI and the luminance corrections. However, the manual method was not a waste of time. Without a profile and luminance correction curves to use as seed parameters to Solver, a reasonable solution may or may not have been achieved. The problem arises because the Solver method requires an array of where the PSI centers fall on a frame, which in turn is derived from a frame by frame analysis that fits a PSI profile to each frame, to determine the PSI center. If the PSI profile for a different projector differs substantially from the one here, the center will be shifted, or not even found, and the Solver method will fail. Basically, beware the automation process as a first method for solving a PSI profile and luminance correction factors. The Solver method originally intent was to improve the PSI filtering process.

#### 3A.1.1. Manually determining the PSI profile

The data from the vignette corrected avi files will be manually transferred to an Excel workbook. The workbook is: *leader\_profiles base PSI data.xlsm*; The format of the worksheets is specific. There are two classes of worksheets. *[corr\_#\_#]* are where the set of leader frame profiles obtained from each avi file will go. [*#\_# meas*] are the worksheets where the alignment points will go. The number of worksheets should be reduced or expanded to fit the sets of avi profiles analyzed, no more - no less, or the macro that reorganizes the data will not work properly.

This workbook contains a macro that will automatically color the measured points, which will be used to align the profiles. ImageJ was used to extract the profiles for the PSI from this new file using the script - *Get profiles of stack slices.py*. This script automatically compiles all or a subset of the frame profiles and places the data in a file - *profiles.csv*. The data can then be transferred to *[corr\_#\_#]* worksheets of the *leader\_profiles base PSI data.xlsm*. The automated script operation is straightforward. A vertical profile of the video frame is extracted in a user defined rectangular ROI (region of interest). The rectangle dimensions typically used were a horizontal range from 240 to 1680 and a full vertical range of 0 to 1080. The profile in the region is then an average of all the horizontal pixels from the left to the right edge of the ROI, as a function of 0-1080 rows (frame height) for a high definition image.

Using the *Get profiles of stack slices.py:*

1. Open up the Excel file *leader\_profiles base PSI data.xlsm.*
2. Clear the data in the *[corr\_#\_#]* sheets.
3. Start ImageJ
4. Click *File|New|Script|*
5. In the new window click *File|Open* - migrate to the scripts containing directory (e.g., /*ImageJ Stuff/*)
6. Click on *Get profiles of stack slices.py*
7. Check line 34 to set the drive path parameter (default is "F" drive). The drives/folders options in lines lines 43-59 may need to be modified for your particular case. (Several flash drives and computers were used in the development process, which is why there are several choices.)
8. Check parameters that may be varied starting on lines 63-76. It is almost certain that the values for the starting and ending leader frames to analyze will need to be changed in lines 73, startf, and line 74, endf.
9. To start the script, click the *[Run]* button at the bottom of the script window. You will be prompted for an avi file, the default folder that opens is “avi\_in”. The analysis process will take less than a minute to run a couple of hundred leader files.
10. While still in Excel, open up the *profiles.csv* file.
11. Copy all the data from *profiles.csv* to one of the *[corr\_#\_#]* worksheets.
12. Close the *profiles.csv* file.
13. Repeat this process until all the AVI files leader frames that you wish to use are in the workbook, adding [corr\_#\_#] worksheets as needed, or deleting those not used.
14. Save the workbook.

Once the profiles are in Excel, to build up a final template of the PSI, alignment points need to be established for a subset of the profiles. These alignment points are whatever distinct points can be easily resolved from the each PSI profile. The points will be used to overlay each PSI for averaging to a final PSI template shape. Six alignment points were used. Four points were where the left and right sides of the PSI meet the baseline, and where the left and right sides of the PSI meet the flat peak portion. Two more points were added because they represented usually very distinct, sharp features on the PSI; the edges of some sort of secondary peak imposed on the main PSI profile, just to the left of the first flat peak point and the second just to the left of the right edge point. These will be referred to as the left and right glitch points. Not all projectors may exhibit such distinctive characteristics. At a minimum, and when clearly defined on the video frame, the four PSI points are necessary.

[The sequence below is detailed and should be taken as suggestive of a sequence to obtain necessary information. Work styles differ from person to person. To some extent, the sequence could be automated, but the amount of effort to do so was not considered worth the time for this project. Expect this process to take the better part of a day – it is one of the most tedious of the entire video filtering process.

Using ImageJ and Excel to obtain and record PSI information:

1. Open the *leader\_profiles base PSI data.xlsm* file. For each avi file to be analyzed make sure the number of *[corr\_#\_#]* and [*#\_# meas*] worksheets match exactly the number of avi frames to be analyzed. Delete and extra worksheets. On each of the [*#\_# meas*] clear all the data in columns C to F. Reset the first column “s#”to just “s”; having the correct slice information in column A is critical.
2. Open ImageJ
3. Ensure that *Edit|Options…|Profile Plot Options…|Vertical* button is checked.
4. Remove all checked boxes related to the Results window in: *Analyze|Set Measurements…* This will only print the x,y values of the points chosen.
5. *One time only, at start of operation*: Check *Edit|Options…|Input/Output|Copy Column Headers*
6. Open an avi file for which the profiles were read into a *[corr\_#\_#] worksheet*.
7. Establish the rectangular region (ROI) to be analyzed using the Rectangular Selection Tool. Make sure it covers the y range from 0 to 1080 (or whatever maximum resolution.) The horizontal region, should be whatever was established as optimal to reduce vignette distortion, e.g., 240 and 1680 in the examples here. You can either adjust the ROI on the image, or set the information using *Edit|Selection|Specify...*
8. Click *Analyze|Plot Profile* to obtain the vertical image profile.
9. Switch to the *Multipoint Selector Tool* (this may be set to point or multipoint - right click to change function).
10. On the profile plot, click whatever is visible of the six specific points on the PSI in the following order (from left to right of the plotted image);
    1. point where the left side of the PSI profile meets the baseline
    2. left sharp edge point of the left glitch.
    3. point where the left peak side meets the left flat peak
    4. point where the right flat peak point joins the right side of the peak
    5. sharp edge point of the right glitch.
    6. point where the right side of the PSI meets the baseline

Anywhere from two to five these points may be clearly well defined on a profile. Even though points near the edges of the video frame may look well resolved, because of the filtering process and edge distortion, use them only in grave desperation. Choosing these points requires a best guesstimate in some cases, but it is an important step to building an accurate profile shape. In some cases, the best way of picking a point is to imagine two lines drawn through the points where the regions intersect. Choose that point. Often this point will not be on the actual plot line, just be consistent from frame to frame.

1. Click *Analyze|Measure* to obtain the point information. In the Results window, highlight the rows and click *Edit|Copy* and copy the information to the [*#\_# meas*] worksheet that matches the *[corr\_#\_#]* worksheet for the same avi file, ***aligning the points correctly to fit which of the six possible points were collected***. (***The points may wrap around***.) (If this is not the first transfer, then turn off the *Copy headers* trigger in ImageJ.)
2. Go back to ImageJ and delete the information in the Results window for the next set of data.
3. Repeat this process for all the frames to analyze in the avi, and for every avi. How much data to take is not easy to estimate. More is always better, but it takes time. The example used four different leader avis with around 15 to 20 images analyzed in each avi. However, acceptable PSI profiles were generated with as little as 30 frames.
4. Once the data entry has been completed. Run the macro “GatherData”. This macro will add to each *[#\_# meas]* worksheet, the profiles listed in column A of the worksheet, and selectively color code each of the alignment points that were analyzed and recorded on the worksheet.
5. All the profiles analyzed and colorcoded on the individual *[#\_# meas]* sheets are also collected automatically on the *[All\_Profiles]* worksheet. This final set of data will be copied into *projector film leader reduction.xlsm* for final analysis.

While compiling this information, several issues were noted.

1. For whatever reason, early video frames exhibited bright or dark values that exceeded later frames, as if the camcorder had not stabilized the exposure with the aperture, or the projector bulb luminance had not settled in. It was decided to not use early frame PSI profiles. Only video frames (> 20) where used to establish the peak shapes (*see [hi frame profiles]* worksheet).
2. There is always some subjectivity in such an analysis, expect that you may see up to +/- 20 row differences between repeated measurements.

## 3B. Analyzing the leader data

See the Excel file - *projector film leader reduction.xlsm* for the calculations

The plot below displays a partial collection of PSIs video frames of a leader. Note the rise and fall of the maximum and minimum values as a function of the PSI position on the frame. The pattern is consistent from one video sequence to another and even reproducible within a single video as the overlapping PSIs show.

Figure . Film leader profiles from a series of frames.

Thus, what we observe is a row and PSI position based luminance that distorts a PSI itself. Several artifacts likely contribute to this PSI position dependent luminance distortion:. One is the inability to completely remove the vignette effects, another is that the camcorder aperture may be adjusting to produce a correctly exposed frame. Whether it is necessary to compensate for the distortion is a value judgment. It first depends on how well the vignette removal works. For instance, in the case shown, if we ignore the left and right rapid edge luminance drop offs, then the maximum variation is only 8 rgb units. In some early testing, the distortion was much higher. A prudent approach would try and account for this distortion to provide the best possible film conversion. The cost is time, both in devising an effective compensation equation, and in computer processing time, because every frame will require a luminance adjusting calculation. The effect can be large enough that it may not be reasonable to ignore it or simply average it out over the entire frame.

The specific approach to resolve the problem was to consider the PSI profiles as the product of two dynamic components. One component is the truncated triangle (or inverted triangle-like feature that is the PSI described in preceding sections; this component is independent of any position on the video frame. Superimposed on this idealized or base PSI profile is a second dynamic luminance distortion component, which is dependent on the position of the PSI in the video frame. It is the combination of these two components that provides the observed PSI profile for any single video frame, which is what needs to be filtered from each frame.

The task of defining these two components is split into two parts: 1.) Extracting the distortion component, so 2.) We can extract the essential or base profile from the PSI data.

### 3B. 1 Extracting the row based luminance variations and base PSI profile

This process is complex. We have some general idea of the theoretical shape of the PSI we expect to see, but we need to extract the precise PSI shape from the recorded PSI profiles. To do this, we first define the row dependent distortion producing component.

In Figure 25, the PSI curves exhibit a nearly continuous envelope of maximum and minimum values across the entire frame because of the close overlap of the individual PSI’s. We simply find the maximum and minimum values of all the PSIs represented at that particular row. We do this for all 1080 rows, which provides a large data set to find a fitting equation that represents the distortion effect *([hi frame profiles]* worksheet in *“projector film leader tests.xlsm”* represents this process*) .*

The original suite of profiles in the Excel file *leader\_profiles base PSI data.xlsm* were transferred to another Excel workbook *leader\_profiles summary.xlsm* sheet by sheet. The maximum and minimum values for each row were found for each leader set, and these minimum and maximum values were in turn averaged. From this final profile aligned worksheet, the maximum and minimum values for each row across all the data sets were extracted and plotted. This can be done manually in Excel by simply finding the maximum and minimum across all the profile columns. However, the task has been automated to an ImageJ script, which will locate the maximum and minimum at every row across all the frames of the original avi file. Load *[Get luminance points from avi stack.py]* in Fiji/ImageJ*.* The start and stop frames to use can be adjusted by setting the limits manually before execution. The maximum and minimum values for each row are printed to the file: [*Luminace\_pts.csv]*. The csv file can be opened in Excel and plotted to see the curves.

The resulting curves for the minima and maxima as a function of row are shown below.

Figure . Luminance profiles of distortion of PSI frames compile from many successive frames.

There is a potential problem with this method because the maximum or minimum values could represent outlier frames and not represent a reasonable value. This was examined using a specialized macro (minmax) in the *leader\_profiles summary.xlsm* workbook. This macro averaged the three maximum values or the three minimum values of for every row for each of four profile suites. The number to choose must not be excessive, because if a region has a sparse number of profiles, the maxima or minima value could be weighted to a lower or higher value respectively. The averaged maxima and minima data are also displayed on the above plot. Although the plots are similar there are some interesting variations. The averaged maxima are lower than the non averaged and the minima are lower than the non averaged; this may indicate exactly the problem of outliers that was mentioned above. The signal to noise appears to be lower in the averaged data, because we are likely not at the mercy of a single profile dictating the maxima or minima across all the profiles. Curiously, a tri-lobed like pattern is more pronounced in the averaged data. The reason for this oscillatory behavior is not really clear.   
Which data to use? The decision was made to go with the average minima and maxima data set, with the hope that it more clearly would fit the videos. The final averaged data was transferred to a new worksheet and saved as PSIminmax.csv.

### 3B.2. Smoothing and modeling the maxima and minima data

In any kind of peak deconvolution or peak fitting operation, smoothing the data before modeling is a typical operation. The data is relatively smooth already, but clearly needs further smoothing. We have two choices: We can smooth the data and try to fit it to some type of modeling equation, or we can jump right to the modeling, which will likely smooth the data anyway. One observation immediately flags the sophistication that will be necessary to fit the data to a model. The especially sharp drop offs on the left and right edge preclude simple regression equations found in Excel for use as models. A more sophisticated model will be needed for a good fit.

Smoothing can be done in many ways. One good choice for the kind of data here is exponential smoothing. This can be done in Excel using the Data Analysis package function “Exponential Smoothing. However, the smoothing in this specific case was done in a freeware software package called “R”, which is a very powerful suite of statistical functions. R has an interactive interface, and can operate from scripts as well. If you are familiar with Matlab, or SAS, you will find R easy to use. Of course, finding the right functions/packages to use is always an issue. There are many tutorials on R, and multitudes of examples of scripts and functionality. Nevertheless, using R represents another software learning curve, and worse, many of the function packages are written by expert mathematicians and statisticians for other expert mathematicians and statisticians. As usual, some sort of code will be provided as an example of how it was used here. Typically, scripts were written in a code editing program (Notepad++) and copied whole or piecemeal into the R interface from where they could be debugged or run. Very rarely was a script run in batch mode.

Of the plethora of regression based smoothing methods available in R, the LOESS method was used. LOESS is a local regression method that applies local weighting to the points around a point to smooth the data. A span factor provides more or less smoothing to the points. Canon*/R stuff/PSI\_min\_max\_loess.r* was the script used in this case. The final data was output to *luminance loess fit.csv*. The data is plotted in the minimum and maximum plot above, but the fit is so close that the lines cannot be distinguished at the resolution of the plot.

The next problem is to fit a mathematical model to the luminance data. There are many ways to model this data. As already suggested, the shape of the data clearly precluded the simplest curve fitting methods. One class of methods that can be used is spline fitting. However, I considered the results of spline models (such as the R package “earth” as being computationally a bit messy to program into the later functions needed to remove the PSI, because of how they use ranges to fit the data. The approach that seemed most appropriate was to fit the data to a series of exponential curves. Despite the power of R for data analysis, I was unable to discern a suitable package or set of functions that had all the capabilities needed. Such a package probably exists in R, but is so convoluted in specialized math language, that I could not recognize it as appropriate. I also was not of a mind to build an ad hoc script, which would undoubtedly lack flexibility.

Regression analysis with Gaussian or Lorenzian curves or mixtures of the two curve types is a common method for peak deconvolution and peak fitting. A complex curve or peak can often be represented as a sum of a series of N exponentials. The particular form of the equation we will use is:

Eqn 25

Am is the height or intensity of specific peak m; xp is the position of the individual peak; σ in this specific case, is the full width at half maximum height (FWHM), which is one way to represent the width of a peak. The factor 4ln(2) arises because the measure of the peak width is based on the FWHM, rather than the standard deviation – σ (basically redefining σ from its usual meaning).

There are many commercial packages, which given a set of starting parameters, will attempt to fit the observed data. However, these packages were not available to me, or were not worth the cost.

However, there is a nifty Excel freeware program (see <http://www.chem.qmul.ac.uk/software/eXPFit.htm>) called exPFit that was developed by Roger Nix to do deconvolution of X-ray photoelectron spectroscopy (XPS) data. This is a great little program once you understand what it is doing. The program can handle up to 10 exponential curves and can vary the proportion of Gaussian to Lorentzian character of the distributions, and allows skewed distributions, such that the left and right side of the distributions from the xp value, can have separate widths. The workbook uses Excel's Data Analysis Solver function.

The input data needed to be massaged a bit before fitting. First, it was necessary to normalize each data set to a baseline of zero. This was done by subtracting the maxima or minima luminance values from the lowest value of the maximum or minimum curves. From the plot above, for the maxima data, this can be seen to be the first point. In addition, a (x,y:1081:0) point was added at the end of the data set. A minor drawback for the current task is that eXPFit only handles 1000 points. This was easily overcome by choosing every second point in the PSI profile. (To do this very simply, use the following Excel formula applied to the full data set: *=OFFSET($B11,1\*(ROWS(G$11:G11)-1),0),* where column B is the column to pick every second value starting at row 11),and 1\* is the number n to skip minus 1 (n-1). This formula is placed in G11 and copied down the cells until all the B row cells are read (which in this case will occupy only half the cells. This reduced set of data was then brought into the ExPFit workbook on the *[Data]* worksheet. The columns to the right were then swept down to match the data size in the first two columns; this is important for the next step.

To minimize the later mathematical complexity, only Gaussian symmetrical curves were allowed. This was achieved by setting the LFWHM and RFWHM cells to 1.0 for all distributions. Similarly, no baseline corrections were allowed. In addition, in the changing cells textbox only the height, peak width and peak position were allowed to vary for each peak. To do this the Solver function must be edited by changing the “by changing..” textbox, which references the cells to hold constant. In addition, to the above changes, exPFit requires starting height, peak position, and FWHM information for each peak on the *[Ft]* worksheet. A time consuming, but extremely useful operations, is that as the peaks and their defining parameters are added and manipulated based on intuition, the plot on *[Fit]* updates to show how the sum of the Gaussian curves fits the data. It is not necessary to find an exact fit; that is the function of the entire workbook. However, the closer the initial parameters, the faster and more accurately Solver will find an acceptable fit. [Solver should be run repeatedly until it no longer shows a change; the workbook attached here has a slight modification in that cell B24 on *[Ft]*, displays the r2 correlation coefficient which helps determine when Solver is no longer having any effect. An important consideration is to minimize as much as possible the number of peaks used to fit the data. Certainly, the more peaks added, the better will be the overall fit, but the more peaks added, the slower the process of removing the PSI from frames, because every frame will require calculating and summing each exponential to determine the luminance correction. Fewer terms means fewer cpu cycles. Just by looking at the data in the luminance plot, the minimum set of exponentials would appear to be five. However, a really close examination shows that seven peaks would be a much closer fit.

After a number of tries, with a number of peaks and parameter values, the best compromise was five exponentials. The plot below shows the fits for the luminance corrections as a function of the row.

The way this is plotted better highlights the differences between the observed and fitted data. The differences between the curves, except at the extreme limits, are under 1 rgb value. As an example, the final set of fitting parameters for this data were:

See, *Exponential Fit five peak for luminance minima.xlsm* and *Exponential Fit five peak for luminance maxima.xlsm* for the data. Because the final curves were originally rescaled or normalized to a zero baseline, it was necessary to rescale back to the original values, using the minimum values originally subtracted from the curves.

The purpose of fitting the luminance data is to back out its impact on the PSI shape as observed in the previous plot, hopefully, leaving the “true” or normalized fragment of the PSI in a visual frame. To back out the luminance artifacts, each leader frame row is rescaled according to:

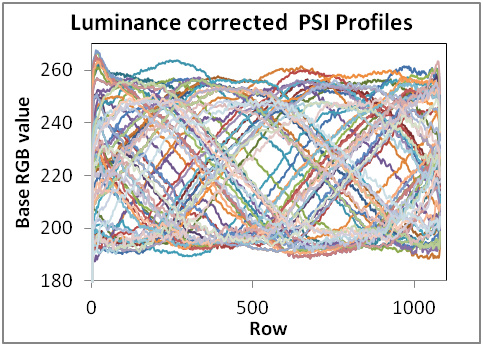
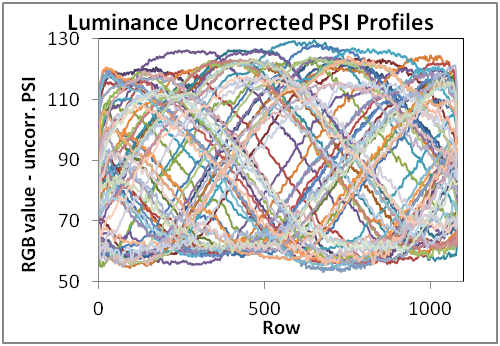
Ir is the new scaled or normalized rgb value at row *r*; ; Io is the original rgb value of the row; In is the new baseline value, which should be the highest rgb value (white= 255); Imax,r and Imin,r are derived from the exponential fitting equations (eqn 10 )for the maximum and minimum limits at row r, found from the suite of PSIs; (Imax,0 – Imin,0) represents the range of luminance that a PSI is expected to have (not the frame luminance max. or min.). The value is interpreted as a constant, and represents what the expected difference will be on any frame between the true rgb value and that masked by the PSI. For the present case, this value was chosen as the average of the difference between the maximum and minimum rgb value of the final LOESS smoothed maximum and minimum luminance curve data- in this case, 61.32 rgb units (see *leader\_profiles summary.xlsm [smoothed\_lumin]*); m in the lm,x term is either *“max”* or *“min”*; x is either “o” or “r”, xp is the row of the peak value, and σ is the full width of the PSI. The 4ln(2) term arises because the exponential model uses the full width at half maximum (FWHM) to represent the peak widths rather than the standard deviation. (The use of FWHM is common in scientific applications.)

Equation 26 looks complicated, but it is derived from simple consideration of fractional proportions. We can choose the new baseline value In to be whatever we wish. The value was chosen to be 255 for reasons that will be clear later.

Each base PSI was then extracted using the above equations in an Excel macro. (See *projector film leader reduction.xlsm* - macro *PSINormalizer*). To do this, the profiles from *leader\_profiles base PSI data.xlsm[All Profiles]* were transferred to *projector film leader reduction.xlsm[filtered Profiles]* and*[Hi Frame Profiles].* The original difference between the two worksheetswas the removal of clear outlier profiles, so that *[Hi Frame Profiles]* represented a more refined set of profiles. (Before deleting the existing data make sure you know where the profile header starts and transfer to exactly the same place.) Once the transfer is completed, the macro *PSINormalizer* can be invoked.

*PSINormalizer* will remove the effect of the row dependent luminance leaving only the undistorted PSI profiles in *[Pk\_Normalized].* Because of the exponentials to be calculated with the large number of points, this is a very inefficient macro. If a large number of profiles are processed (like 70), come back in an hour, even on a relatively fast computer. Excel vba is not an efficient language for this kind of calculation, but it does not have to be run often. Below shows the comparison of the data before and after luminance correction data.

Figure . Collection of PSI profiles before (left) and (right) after luminance correction.



There is a close similarity between the row luminance corrected and uncorrected data, but the corrected data displays less variation as it should.

### 3B.3 Extracting and compiling the base PSI

#### 3B.3.1. Aligning the peak information to obtain the complete peak profile

Once the row dependent luminance variation has been removed from each profile, we finally are in a position to obtain the undistorted PSI profile. First, the individual PSI profiles must be aligned so that they can be averaged to obtain the full width and shape of the fully normalized PSI. When the base PSI is coupled with the reverse luminance correction, the effect of a PSI on a frame can be filtered from a frame. Because the PSI profile width is larger than the 1080 rows in the video frame at 1/100th second, alignment is a piecemeal process of aligning different portions of the PSI profile observed on the leader video frames to build the complete profile.

For each profile, it is necessary to color code the row values representing some subset of the six registration points that were manually identified and recorded on the *leader\_profiles base PSI data.xlsm* workbook. The color choices are not arbitrary. The *projector film leader reduction.xlsm [Hi Frame Profiles]* worksheet shows a table of colored cells at the top, which corresponds to the colors of various registration points. Because this color scheme is also used in *leader\_profiles base PSI data.xlsm* and by several macros in both workbooks, it is not a good idea to change the color scheme unless done in both workbooks, and using the same cells. If the process so far has been faithfully followed and the *GatherData* macro in *leader\_profiles base PSI data.xlsm* used, the correct points will have already been color coded on the worksheets.

There are two similar methods that were developed to align the values; they differ only in the points used to align the peaks. The first aligns each profile to the left and right points representing the flat portion of the PSI peak. Without any other sharp features in the profiles, these points are considered the most accurate to obtain, because where the peak side meets the flat peak top has the sharpest change in slope.

If doing this manually (not recommended) on a new spreadsheet, the first column should be a pixel range that exceeds the value of the expected width of the PSI, in this case a minimum of 2 x 1400.

The operation, whether manual or based on the automated methods, proceeds as follows: First, all profiles indicating a left peak point color, but not preceded by a right peak point, were aligned to the same arbitrary worksheet row. For the right peak points, the right peak point row indicies were tabulated for each of the profiles that exhibited the now aligned left peak points and a right peak point. These right peak point index positions were then averaged. All the remaining profiles that contained a right peak point (and may have contained a left peak point, and may have had a left peak point at a higher row position number) where aligned at this average right peak point index. An Excel macro, *projector film leader reductiontests.xlsm* macro *Peak4PointAligner,* automatically handles this alignment task. Read the macro header for details on where the macro expects to find the data and where the final aligned plots will be found.

The second method, which was considered the most accurate way to align the profiles, was based on the presence of additional “glitch” peaks superimposed on the much broader PSI profiles. If you look closely at the PSI image sequence shown before, one of the glitch or notch peaks is just visible, above the main flat (darkest) portion of the PSI. Both of these features were relatively narrow and their position consistent from frame to frame. Such sharp feature changes are very convenient for aligning the profiles. These were the points used for the final alignment in a manner similar to the description for using the flat peak edge points for alignment. Both methods are described, because it is uncertain whether such sharp features are common to all projectors. See *projector film leader reduction.xlsm* macro *Peak2PointAligner* and read the introduction to the macro to understand expected inputs and outputs using the glitch peaks. Not all the profiles may end up being involved; a few did not exhibit glitch peaks that could be reliably located. The macro will place the aligned data in *[temp\_aligned].*

The aligned PSI profiles should go through a final alignment review based on the color coded peak points measured earlier. This is done by generating a scatter plot of the data on *[temp\_aligned].* Although the automated process aligned the data, there usually was final tweaking to adjust the positions or eliminate PSI profiles that appear as outliers. The major reason for outliers is incorrectly assigning the points from the ImageJ Results window into the proper registration position on the *leader\_profiles base PSI data.xlsm*. Thus, it may be necessary to go back and reanalyze some profiles, and repeat the above processes to achieve proper alignment. If there are many profiles analyzed, it may be simpler to just remove the non aligned profiles in *[temp\_aligned]*. A scatter plot in Excel 2007 can handle up to 255 series, which should be more than enough.

An additional manual data massaging must be done on *[temp\_aligned]* profiles at this point. Unless the luminance correction equations are perfect, most profiles will exhibit residual rapid extreme rgb spikes at the edges of the frames. Each profile needs to have these removed. This process can take some time and usually requires removal of the first and last 4 – 10 points of a profile. Without doing this the final average PSI profile will be distorted and much noisier.

Once all the profiles were aligned and the end point dealt with, the data can be averaged for every row across all the columns. The Excel macro, *AveragePSIColumns* in *projector film leader reduction.xlsm* handles this task.In doing this, an issue regarding potential residual peak distortion issues was ignored. The “flat” portion of PSI profiles near the left edge of the frame often show a tendency to have a higher left peak point than a right peak point, and the converse was true of flat profile portions near the right edge. The effect is not large, usually no more than a couple of rgb units, but the magnitude certainly depends on how effective the row luminance correction process is. Another confounding factor is that the peaks can change because of projector speed changes. However, the effect of this is to somewhat smear out the final base PSI profile.

#### 3B.3.2. Reducing the noise of the averaged PSI

The next operation is to reduce the noise of the averaged PSI profile. First, the averaged profile was examined to decide what final range should be used. These were the end points where clearly no averaging artifacts could be noticed. This data between these two points was then copied to *[PSI Smooth]* in *projector film leader reduction.xlsm*. Two standard methods were tried to smooth the data: LOESS, (locally weighted scatter plot smoothing) and the Excel built-in Data Analysis Pak Exponential Fitting function with a 0.91 damping factor. Both methods provided essentially the same smoothing. The exponential fit data was generally used because it used the built-in Excel Data Analysis function and was thus a bit more convenient than transferring the data to use the LOESS routine in R. The final PSI profile is shown below. The average peak is the average of the experimental data, and the “exp…” data is the exponentially smoothed data. Compared to how we have been considering the shape of a PSI profile up to now, the PSI peak here is reversed from that sense; the peak is represented by the lowest rgb values, not the highest. This representation is true to the way the PSI profile will be analyzed and removed.

Figure . PSI extracted and smoothed from suite of manually adjusted PSI sections.

The final task was to isolate one PSI width from this smoothed data. Based on the data shown in the figure, the PSI spans about one and a third PSI profiles. We want to isolate one complete PSI from this. Although the points could be isolated by searching for the maxima in the plot, a more definitive process was to take the derivative of the data (simply subtracting one row value from the previous row) and note the values where the derivative crossed the zero axis. From these limits, the width and final shape of the base PSI could be obtained. The range determined was then transferred to *projector film leader reduction.xlsm[ PSI].*

A final manipulation of the curve is necessary. We will ultimately generate a two cycle PSI profile vector. The endpoints, e.g, the left and right maxima of the PSI, were displaced by a small amount, the rgb values were 254.4 and 253.3, respectively. To reduce the effect of this sharp transition between two cycles, it was necessary to smooth the transition using the first and last five points of the profile; this was done using a linear interpolation between the ten points. In addition, a final baseline subtraction was done to force the maximum value of the PSI to 255.

As mentioned, a single cycle representation of a PSI is not what is used. The reason is that the PSI must be sampled across a range equal to the vertical resolution of the frame, whether the PSI profile on the frame starts at the beginning or end of a PSI profile. Therefore, a two cycle PSI vector is needed. Also, the way the profile removal process was devised, the range for the two cycles *must* start with the midpoint of the base PSI as row zero. *The starting and ending points are not arbitrary and must be in this form. The analysis scripts expect the profile to be in this same pattern.* Thus, the PSI that is ultimately stored for use by the PSI filter is the following:

Figure . Final normalized double width PSI profile used as standard PSI profile normalized to 255 RGB.

The final action is to copy the 2 cycle PSI profile to a new workbook and save it as a csv file with two columns of data (row and rgb value) with no header. The file name is *PSI\_norm\_1358.csv*. This file name can be changed, but only if the scripts discussed below are modified to recognize a different name.

For strictly display purposes to compare with some of the earlier discussion to invert the single cycle curve, the PSI profile can be inverted by subtracted from 255. The final result representing the inverted single PSI base cycle is shown below.

Figure . Single PSI cycle represented as a positive profile (reversed for viewing only.)

The full width of this profile is 1358 rows, which is beyond the maximum height resolution of the 1080 high resolution vertical value of visual frame. Remarkably, this width was found to be exactly the same between even different methods for luminance correction, and even different leader recording sessions. Note that there is a certain amount of asymmetry in the peak. The glitch peaks are evident, although the left glitch point is less sharp then was clear in most individual PSI, due to the averaging and smoothing process. The differences are only 1-2 rgb values and will not be evident in the PSI removal stage.

Some early considerations of the approach to filtering the PSI from frames and the final method of optimizing the PSI profile need a mathematical representation of the PSI profile, even though from a computing perspective, the PSI will be loaded and used as a vector. Three fitting models were developed: 1. A linear line model, where the PSI curve was modeled using three straight lines. 2. A fit using Fourier Analysis to deconvolute the PSI into a series of frequencies to model the PSI curve. 3. An exponential curve fit which relied on a series of Gaussian curves to fit the data, using once again the ExPFit Excel. The latter two could be made to fit the curve to a very high degree. In the Fourier case, 30 terms were needed to get a good fit and in the Gaussian curve fit, six symmetrical Gaussian (no Lorentizian components) distributions were able to produce a high fidelity mathematical representation of the experimental PSI profile. Just for comparisons sake, below is the PSI with just three straight lines representing the sides and top of the distribution. This is shown below with lines based on the four points identified as the left and right peak edges and the left and right flat peak points as indicated in the previous section.

In the worst case, the variation is about 22% deviation. However, that only represents a difference of 2 rgb units. It was not immediately clear whether this would be sufficient to still observe flickering, but the issue was not investigated further.

## 3C. Enhancement of the PSI and frame luminance curves - alternate semi-automated method to determine PSI profile and luminance curves

In principle, the reverse process, to regenerate the row dependent distorted PSI luminance profiles given the theoretical PSI distribution and the row to correct is,

Eqn 28

by simple rearrangement of the original equation 26. Initially, this equation was used to apply the luminance correction to every PSI filter template, which in turn was used to remove the PSI shadow from each frame. However, some consistent luminance artifacts on frames were noticed that suggested the preceding equation was not sufficiently removing the PSI. This prompted a second approach to determine the best combination of PSI profile and a frame luminance correction, using a regression optimization strategy.

The modified approach to find the PSI and luminance profiles used the manually determined PSI profile and a modified luminance correction equation to regressively find optimum profiles. If any projector and camcorder produce similar profile characteristics to those identified here, there is a chance that the cumbersome manual method of determining the PSI and luminance corrections can be avoided. Success or failure will hinge on how similar the PSI profile derived here works to find a PSI on a frame (see later), and second, the vertical blanking time of the camcorder, which dictates the frame to frame timing cycle, which in turn affects the critical differential PSI shape template (again, see later). However, the universality of this approach remains untested.

With the apparent failure of the above equation. a somewhat arbitrary equation was used to finally correct the PSI for frame level distortion and is defined as:

Eqn 29

The last factor is simply the average of the luminance differences. This equation had the computational advantage that it only needed to be calculated once for all frames, because there was no row level dependency on the PSI luminance value as in the previous equation.

The optimization process started from the already established exponential versions of the manually derived luminance curves and PSI from the leader frames. (*See F:\Canon\PSI and Lumnance Curves via Solver.xlsm*) . Excel's Solver function was used for the optimization. A user defined function was created which used the sets of exponentials that defined the PSI, and the luminance maximum and luminance Minimum curves previously mentioned. The 48 exponential parameters that were initially defined as the parameters were chosen as adjustable parameters in Solver , six exponentials to describe the PSI, and five each to describe the Luminance Max/Min curves. Solver adjusting the parameters, to create a new PSI profile and new minimum and maximum luminance frame curves to remove the effect of the PSI shadow from each frame. Solver expects a simple scalar value to monitor goodness of fit. The output value chosen to minimize was the sum of the standard deviations of all rows across all frames in a suite of leader frames (variance could also have been used).

Solver is a powerful optimizer, but it is slow, particularly with the extensive manipulations to calculate the modified PSI. (The functions being optimized are nonlinear.) To help reduce the optimization, the optimization was done in a stepwise fashion, first with 10 frames and finally with 18 frames. Still, we are talking about four to six hours of calculation on a moderately fast, over-clocked computer to arrive at a solution. R packages, such as optimx, are expected to be a more efficient , but convenience for what was expected to be a onetime calculation won out over the considerable time to write and debug another R script.

The final PSI profile and the luminance curves compared to those previously determined is shown below.

Figure . Solver optimized PSI normalized to 255 rgb

Figure . Frame luminance maximum and minimum envelopes (Eqn 29) optimized using Solver

The PSI curve differs little from the manually established version, but the Luminance Correction is very different.

Below is an example of the final exponential parameters that describe the PSI, and Luminance Maximum and Minimum Curves.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Exponential parameters for PSI and Luminance | | | | | | |
| ***PSI exponential parameters*** | | |  |  |  |  |
| **Peak No.** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Position** | 297.22 | 343.91 | 144.65 | 501.49 | 211.13 | 600.71 |
| **Height** | 0.83 | 64.80 | 6.54 | 0.97 | 6.19 | 1.41 |
| **LFWHM** | 17.49 | 344.41 | 75.98 | 44.97 | 64.36 | 16.08 |
|  |  |  |  |  |  |  |
| **Luminance maxima and minima Gaussian parameters** | | | | |  |  |
| ***Maximum parameters*** | | | | | |  |
| **Position** | -4.12 | 24.07 | 443.92 | 809.68 | 1031.39 |  |
| **Height** | -15.34 | 4.29 | -3.06 | 109.93 | 6.75 |  |
| **LFWHM** | 39.29 | 211.03 | 360.93 | 383.92 | 79.90 |  |
| **Minimum parameters** | | | | | |  |
| **Position** | -68.59 | 140.22 | 310.08 | 557.25 | 1079.42 |  |
| **Height** | -121.23 | -0.68 | -2.01 | 22.09 | 5.64 |  |
| **LFWHM** | 95.53 | 22.31 | 275.15 | 128.56 | 240.58 |  |

## 3D. Calculating the velocity of the PSI and the shutter width on the visual frame

The original goal of his work was to develop a method to predict the position of the PSI on any frame. However, after much effort, the conclusion was that this goal was not attainable in a real world analysis. As the analysis of the white leader data progressed, serious discrepancies arose in predicting the position of a PSI, even within this small set of frames. Using a variety of statistical methods, plots of the residuals between calculated and observed values always exhibited regimes defined by abrupt jumps or transitions. Ultimately, the discrepancies were traced to two assumptions: One, that the PSI width is a constant, and second, that the PSI width and frame to frame time are known with a very high degree of accuracy, e.g. to more than four significant figures. The original concept was to determine the position of PSI by successively adding a constant PSI width value, but if either the vertical blanking time or the PSI observed width is off by a small amount, there will be a rapid accumulation of these small errors into large shifts in calculated position. As a simple example, if the PSI width is off by just one row per thousand, this means the predicted PSI value will be off by 600 rows after only 300 frames, which is only 10 seconds of movie time. That is more than a half frame shift! In addition, without special equipment, the projector speed is not likely so finely controlled that we can assume the frame rate, which also affects the PSI width (and thus the frequency) will be constant over thousands of frames. Such errors totally negate a successive predictive calculation.

As a tutorial on the considerations involved in predicting the position of PSI centers on frames, the effort expended in describing the dynamics of the moving shutter image may be instructive. Some of the concepts discussed here are important for understanding some of the calculations used later; skipping this section may be appropriate for some, targeting the conclusions as quickly as possible, but for those trying to understand later image manipulations, this section will be very important.

As we found for the bar, especially the scan rate and the velocity of the PSI are needed to allow accurate calculation of the PSI position on a frame at any time. Earlier, Equation 2 was developed that predicts the image width a moving bar based on the scan rate, the bar velocity, and the static width of the bar on the visual frame. Unlike the moving bar case, we do not have any independent means to measure the true or static PSI width. In addition, as Equation 2 and Table x show, even if we are able to measure the edge to peak distances, these widths will still be distorted due to the rolling shutter effect and we must correct for that. (Since we are monitoring the shutter as it moves with the row scan direction of the camcorder, we expect the PSI width to be larger than the true width.) One way around the rolling shutter effect is to turn the camcorder on its side so that the PSI appears as a vertical bar moving horizontally across the camcorder frame. We found that in the case of a vertical bar moving horizontally, the shutter width was the same in either left to right or right to left motion. However, this method is not easy to set up. Adjusting the frame size to match the horizontal case can be tricky. In addition, this still does not produce the true image of the shutter width, but only eliminates the rolling shutter distortion. There still is the issue of what is the proper width measurement. The data may also be suspect, because it does not represent the actual conditions we need to achieve the goal of removing the PSI from each frame.

There is a way to get the appropriate data from normal recording of the film leader images. The PSI widths are measured as a function of the shutter speed as we have done for the vertical bar. This is quite simply done from a short series of recordings of the film leader at different shutter speeds. Equation 2 is then used to extract both vs and a “virtual” shutter width. The only practical issue is the tedious data reduction process to measure the image widths.

From equation 2, if we know the PSI widths at any two points, we have a system of two equations with two unknowns, which can then be solved for vs and the PSI width. Note that if we set up two equations with the measured PSI image widths and shutter times and subtract the two equations the intercept term containing (PL – P0) drops out, leaving us with an equation with vs as the only unknown. Combining the two equations as suggested, produces the general equation set:

Eqn 30

where

Eqn 31

The indexes 1 and 2 refer to any two measurements of the PSI width, P, at two different shutter speeds ts. The film leader was recorded at shutter speeds, 1/500th s, 1/250th s, and 1/100th s. As already discussed in defining the PSI curve, the data at 1/100th s was obtained as a composite of several curves, because it extended out to 1358 rows, well beyond the 1080 vertical resolution of the visual frame. Thus, the shutter speed was extended to a shorter time in order to have at least a couple of points that could be directly measured from a single visual frame.

Figure . Projector shutter Image total width as a function of shutter time.

The linear equation fitting the three points is also shown on the plot. From this equation, using any combination of two points, the above equation for calculating vs, and the row scan rate, vr, derived from the moving bar image (63,420 pixels/s), vs was calculated to be 34,805 pixels/s.

From equation 2, the intercept of the line through the points is given by:

Eqn 32

Rearranging the equation, we can isolate the virtual PSI width as:

Eqn 33

Plugging in the values of the intercept from the plot and the values velocities, results in a PSI width of 221.0 pixels. Assuming the measurements of the projector shutter width is subject to the same problems we saw with the bar measurements, this value represents an upper limit to the true shutter width value. However, this value is what will be used for any predictions of the moving shutter.

Compared to the moving bar results, the very high velocity of the shutter over the visual frame leads to a very large amount of smearing of the shutter image compared to the bar’s image, even though the true shutter width appears to be near the same relative size as bar’s image.

### 3D.1. Calculating the position of the PSI on successive frames

The system for determining the position of a PSI on a frame is quite different than the way we determined how to find the position of a moving bar. In the PSI case, we have a periodic function and successive images of the PSI represent new instances of the shutter being projected. As we will see, because we have established the shape of the PSI and its frequency from a moving reference frame, many of the issues involved in calculating the position of a moving bar are no longer necessary.

There is a ground rule that must be set before we can move into the calculations. All positions for PSIs will be based on measuring them at the midpoint of the flat top portion of a PSI; this protocol is chosen as the most easily identifiable feature, despite its fairly broad contour. The latter feature does have the drawback of a higher inaccuracy in determining the PSI center. For any frame n, the midpoints will be considered as Pn.

The positions of our directly observed peaks are measured with respect to the frame number, but this is not convenient for managing and manipulating the positions. Instead, we look at the PSI positions as a continuous time stream with visual and blank frame markers superimposed on a continuous time stream. Because we know that the row position is proportional to time, we can use a streaming row position to identify any PSI peak. Starting with a PSI in frame n (Pn), where on the xth visual frame will the other PSIs be observed? On a continuous stream basis, the starting row of any frame x is given by,

Eqn 34

where tB  is the vertical blank time between frames and vr is the row scan velocity. B is thus the virtual number of rows in the blanking region. This equation has also introduced a different view of what constitutes a camcorder frame. Previously, we concentrated on what we could see in the visual frame alone, but this is not appropriate for discussing the effect of a PSI on the visual frame. PSI’s that are positioned “off” the visual frame in the vertical blanking region will impact the rows in the visual frame. To consider the two regions as completely unrelated is not appropriate. Now the term “frame” will refer to the combination of the visual frame and the succeeding (in time or rows) vertical blanking region. The reason for the choice of order arises because we always start measuring PSIs on a visual frame or the next PSI in the sequence.

The position of the end of frame x is,

Eqn 36

Similarly, the frame based value of a PSI, Pn, that we normally measure, can be converted to a stream position P’n using,

Eqn 37

The camcorder frame rate and the PSI motion rate are two independent data streams that merge into the composite data stream we observe. To work with these two independent streams, we need to find a relationship between the frame index and the PSI index to predict where a PSI will appear on a frame. The PSIs are a periodic function with a peak to peak wavelength of w rows, which starts at Pn where n is the first frame of the film sequence. The kth PSI position, Pk, is equal to,

Eqn 38

on a streaming basis. For any frame n, this must also be equal to P’n,

Eqn 39

Eqn 40

or,

Eqn 41

This equation provides a relationship between the frame number n and the index of a PSI on the frame. k is strictly not an integer, but is converted to an integer by stripping away the fractional part.

The reverse process, which is important for converting the kth PSI to frame position, is given by,

Eqn 42

where no is the PSI position on the first frame. The basis for this equation is that the starting index, k, for the stream value of Pk need not coincide with the frame stream value, P’n. We typically will count the first PSI as k = 0.

We can also establish the number of PSI peaks between two frames, from the first equation,

Eqn 43

or more conveniently,

Eqn 44

The use of rows to monitor the continuous steam has a disadvantage that we end up dealing with very large numbers very quickly, as the number of frames increases. Another formulation avoids this. We have two periodic processes. The camcorder frame generation and the PSI images. We are interested in where the periodic maxima of the PSIs fall on the time axis of the periodic camcorder stream. The maxima for the PSI can be described by a simple cosine function with unit amplitude,

Eqn 45

where fPSI is the frequency (cycles/s) of the PSIs and k is the PSI period or cycle number. For this relationship to be true for any peak, must be equal to zero. Thus, the PSI peaks occur at times:

Eqn 46

We do not really care what shape the PSI takes on, only that it has a known period and that some point, e.g., the midpoint of a PSE cycle, can be identified in a movie sequence. Given the frame rate frequency (frames/s) fc of the camcorder, the peak times of the camcorder correspond to the position or cycle number Ck = fct. Substituting the above equation for t,

Eqn 47

Because a PSI peak will not start at the first row of a video frame we have to add a phase factor φ that represents where the first PSI peak starts relative to the first frame we analyze. Ck will be a real number; the integer portion represents the kth frequency cycle, in our case, the frame number, and a fractional part that tells us where within the kth camcorder cycle the PSI peak falls. The running row position is given by Ck(H+B). Because we have defined a frame as H+B, the fraction of a cycle from H/(H+B) represents a PSI falling within a visual frame. The phase difference φ or starting position of the first identified PSI peak must also be converted to a time fraction of the frame size and is given by φ = Po/ fc (H+B). fc is equal to the inverse of the camcorder cycle, which is 1/(H+B), and from the previous derived equations fPSI is = 1/w, where w is the width of the PSI band. Putting this all together,

Eqn 48

or

Ck are the PSI positions converted to a frame fraction based on the sum of the frame height and blanking region rows. The frame index n is just the integer portion of Ck and the fractional part represents the position of a PSI within frame n,

Eqn 50

Any PSI measured on a frame can be converted to the corresponding frame fraction using,

Eqn 51

As in the previous case we can calculate the number of PSIs between frames using,

Eqn 52

or we can estimate the PSI width,

Eqn 53

The problem with all these equations is that they assume a constant PSI width. If the width varies then the k values will vary and (k) x (w) which determines the position of a PSI center, is unreliable after a number of frames. We can get around this problem, by independently providing PSI center information on some interval across the video sequence to constantly reaffirm or adjust the PSI width to still allow a successive predictive calculation. We can estimate the number of PSI peaks between frames if we know an approximate range for the PSI width (and hope it does not vary dramatically). An idea of how this works can be demonstrated considering the simplest case of finding the k value between successive frames. It turns out the number of PSI peaks between video frames given the established width can either be one or two peaks. If we take the measured difference between the centers of two PSIs on a successive frame set, and divide the result by either one or two, whichever result is closer to the expected PSI width is the appropriate PSI width for that interval. In this specific case of successive frames, the more reasonable PSI width is easy to recognize, because the difference will be close to a factor of two. It turns out that there will always be a range of two whole numbers of PSI peaks that are possible for any frame interval chosen. However, there is a limit to the number of frame intervals over which we can reliably distinguish between the two sets of number of PSIs considered against the potential error range of measuring the PSI width. The maximum interval depends on the stability of the projector, but is likely to be around only four to five frames. What that means is that in a standard 50 ft roll of film containing some 6200 frames, something like 1400 manually determined PSI positions would need to be fed into the calculation to keep the successive approach on track; clearly, this is impractical.

The entire project reached a crossroads at this point. Necessity drove the decision to explore methods to automatically or semi-automatically determine the PSI positions from the visual frame data, as a pattern recognition technique. Developing this process had a profound impact on the entire direction of the project.

### 3D.2. Manually determining the center position of a PSI

As discussed in the previous section, we are constrained by the accuracy of the PSI width and the mechanical variation of the projector, so that establishing an a priori PSI position is not a reasonable option. I will discuss the manual method for obtaining the center position of a PSI, because the automated method follows the same fundamental pattern. This process was used only for initial explorations of limited frame ranges.

Manually recording the frame number and position of PSIs at intervals by analyzing each frame by eye is a reliable, but very inefficient method. This process itself is not foolproof, but it has the advantage of the nearly instant analytical power our visual capacity and high speed analytical brain to decide between good data and bad data.

Any software program that shows the frame number and the position of a cursor within the frame will work to extract the peak data. In some cases, to be sure of the PSI position, moving back and forth between frames may help estimate just where a PSI appears on a frame. If a higher precision is desired or the ability to more conveniently record the information is more important, then ImageJ is highly recommended. However, this is only reliable for frames where the luminances and colors of the scene are nearly uniform in original intensity and color, such as film leaders, or sky shots. ImageJ offers the option of quickly isolating a vertical slice, plotting the rgb luminance profile and from the profile plot, obtain a fairly accurate value. However, ImageJ can only open \*.avi files, but not \*.mts files or mp4 files directly.

To use ImageJ, the video file must first be converted to uncompressed .avi format, using whatever video software is available; there is currently no plugin in ImageJ to read .mts. The avi file is then opened in ImageJ using *File|Import…|avi…..* A slice (frame) is chosen. A value for the center of a PSI can be guestimated by eye. However, a more reliable method is to use the standard Plot Profile plugin to provide more accurate data. To do this, a rectangle mask is applied, covering an appropriate area of the video frame; typically this area was the full height, and the width constrained to the column range, 240 – 1680. This eliminated the dark sides of the film image, which results from the mismatch between film dimensions and HD dimensions. As usual, if the Plot Profile method is used, first set the vertical profile measuring option: *[Edit|Options...|Plot Profile Options|Vertical Profile]. Analyze|Measure…* option was then invoked to collect the data, which can then be transferred to a spreadsheet. Note that the Analyze|Measure… plugin does not record the frame position. However, if the single point tool was used on the image itself, then the frame number as the “Slice” was recorded using Analyze|Measure...

Even under the best circumstances, e.g. film leaders with generally uniform rgb values, the precision of picking a PSI center was typically within a ±20 pixel range. Thus, we have a limit of ±20 rows as a target to ensure goodness of fit. In normal scene footage, the precision is much worse. Even in a moderately uniform color video, the best one can do is more like ± 80 pixel range, and the errors only get worse from there.

### 3D.3. Template recognition

To process the enormous amount of data needed to filter out effect of a PSI on a video frame, some kind of automated method must be used to identify a peak center. This falls into the regime of pattern recognition techniques. The class of computer pattern recognition techniques that best fits the current need is template recognition. The concept is straightforward: A template of an object or pattern is created and this pattern is moved and rotated around an image attempting to find a match. These are cpu hungry methods.

There are all sorts of statistical methods to determine the best fit and identify the position or characteristics of an object. A common method is based on Pearson’s correlation coefficient given by the formula below:

Eqn 54

A reworked version of the correlation coefficient equation which streamlines the cpu calculation time a bit is (<http://fiji.lbl.gov/mediawiki/phase3/index.php/Integral_Image_Filters>) :


 r_{XY} = \frac{n\sum_{i=1}^nx_iy_i - \sum_{i=1}^nx_i\sum_{i=1}^ny_i}{\sqrt{n\sum_{i=1}^nx_i^2 - \left(\sum_{i=1}^nx_i\right)^2}\sqrt{n\sum_{i=1}^ny_i^2 - \left(\sum_{i=1}^ny_i\right)^2}}


A version that avoids the slow square root function is


r_{XY}^2 = \frac{a^2}{\left(n\sum_{i=1}^nx_i^2 - \left(\sum_{i=1}^nx_i\right)^2\right)\left(n\sum_{i=1}^ny_i^2 - \left(\sum_{i=1}^ny_i\right)^2\right)}


with


a = n\sum_{i=1}^nx_iy_i - \sum_{i=1}^nx_i\sum_{i=1}^ny_i\quad\text{and}\quad{}\sgn(r_{XY}) = \sgn(a)


The latter version eliminates the calculation of the averages, which requires cycle consuming division operations.

This is still a computationally intensive fitting process. Consider that at full resolution, because of the luminance distortion of the PSI as a function of row position on the visual frame, we must create not just one template, but 1358 templates each containing 1080 (xi) points; each column template represents a shift of one row along the PSI profile adjusted for the luminance factor. Each of these templates needs to be compared to each of the 1920 columns of the visual frame. We have (1920 x 1080 x 1358) values to correlate. Some compromises are necessary to manage the problem for handling on personal computers.

The good news is that unlike most pattern recognition problems, where the x,y position, the angle, and sometimes the size difference between the template and image object need to be considered, because the PSI occupies the entire horizontal recording frame, we can convert a two dimensional problem into a one dimensional vector problem. This substantially reduces the calculation overhead to a 1 x 1080 x 1358 problem. However, we still will have some problems related to the width of the image as was mentioned previously.

Also, there are really only two sets of differences that we need to calculate in the correlation coefficient equation, one for x and one for y, which are then used in the four terms. The summation terms of the template PSI (x values) can be pre-calculated for each template, stored and reused as needed.

A further reduction in calculation time is made by reducing the 1080 vector size of each template column using a scaling factor to take every nth row. For an image frame, if we start with every 10th row, we will be correlating against 108 points in the image. This reduces both the columns and rows to be analyzed in the template by 10. The number 10 at this point is an example, but as will be shown later, test spacings of 10, 20, and 30 rows, showed that even 30 row intervals may be adequate to find the peak center. The most appropriate interval may have to vary, depending how well the correlations coefficients are able to ferret out the PSI frequency variations from the image. In this specific example, we have now reduced the calculation problem to (1x108x120) per frame.

Further reduction of the correlation time was attempted under the impression that edge effects might cause analysis problems anyway, and could be reduced by using a subset of the visual frame rows. However, this did not turn out to be true in practice. When the top and bottom 60 rows of a 1080 height image were eliminated, the standard deviations of the positions compared to the experimentally measured values were a bit worse. The reasons for this were not investigated. It was also noted that the full height calculation only marginally improved the calculation time anyway. Accuracy of the prediction was considered to outweigh the small computation time improvement.

As mentioned previously, we benefit enormously in processing time, because the PSI covers the entire width of a frame. If we are careful about aligning the projector and camcorder to produce a horizontal band across the entire image, we can average each horizontal row to get a single 1D column vector of row values. Just as important is that this averaging process further helps by averaging horizontally (frame width) all the scene luminances, which will help draw out the luminance differences caused by the PSI. (We will ignore the fact that we have gamma encoded information, because both the template and the images are recorded from the same device.)

If it were not for the video frame luminance distortion of the base PSI, the comparison could be even faster. All we would need to do is compare a single vector consisting of a subset of the image profile rows at some pre-defined interval, and compare it at steps equivalent to the interval size along the base profile vector. Some effort was made to examine this; the leader data suggested that it might be possible to avoid the pre-generation of position corrected luminance values, but it remained to be seen whether this could be applied to real video scenes.

There are other methods of template recognition that might be faster, such as the Viola-Jones recognition technique and various extensions, but none of these were tried.

*More compromises in developing the recognition process*

The choice of software package was another variable to consider. A full process of finding the PSI positions consists of the following two stages and sub steps:

Stage 1:

1. **Generate a PSI profile csv file containing 2 PSI cycles, and a 2 cycle difference file based on a standard PSI width and the full frame size, which includes the visual frame and the blanking frame.** This is handled using Excel as discussed in previous sections.
2. **Generate the row dependent luminance equations to account for the luminance variation as a function of frame row.**
3. ***Generate pattern recognition template of PSI profiles.*** An initial template of luminance corrected profiles is necessary to compare against each frame to locate the PSI center on every video frame. This is essentially a two dimensional matrix consisting of the number of columns equal to the normalized PSI width and each column containing 1080 rows (for HD). Each column represents a shift of one row along the PSI. The first row in each column is luminance corrected according to the luminance correction equations for row 0 in a frame. Similarly, the second row in the columns is corrected against the luminance correction for row 1 in the frame. Also, a template may not represent every row, but a uniform subset of rows at some interval.
4. ***Compare each video frame at some interval between rows against the luminance corrected PSI to locate the PSI center positions on a frame.***  The inputs to this process are the rgb values that define the PSI difference profile in step 1, and a difference image created from adjacent video frames as a vertical greyscale luminance profile. The difference image profile is then compared to every column of the template and the Pearson coefficient calculated for each comparison. The best correlation is determined, from which the PSI center is established for that frame. This process is repeated for every frame. The PSI center results are then output as a csv file which is associated with the corresponding avi file. Along with the centers information, the nominal PSI width between the two frames is established; the output rows also contain the row centers adjusted for the 0-400 row anomaly and the moving average width. This information will be used to establish an average PSI width over some number of frames.

A clear inaccuracy in this process is that the generated matrix template is based on the assumption of a constant PSI width, which may not always be true. We then use that template to find the position of the PSI on the camcorder frames. This is a cart-before-the-horse dilemma. We developed a normalized or base profile of the PSI with a defined width from the white leader information, but we do not really know the exact width of a PSI on every image. That variation is what we hope to account for. A second inaccuracy is that we will find a PSI position based on a self-imposed subset of the data. These compromises will lower the correlation coefficient and may lead to some systematic inaccuracy in calculated PSI center positions. What we can get away with regarding the limits of this process can only be based on the observed final results.

Stage 2 represents the heart of the PSI filtering operation. The general steps below are integrated into a single program:

1. ***Determine the PSI widths that will best accommodate the data.*** The PSI center differences between successive frames are not only a function of mechanical frequency variations in the projector, but inaccuracies in the pattern recognition process. The variation turns out to be periodic and thus the PSI widths need to be smoothed out. In addition, the PS width data is sorted into regimes to reduce calculating templates as much as possible.
2. ***Generate an interpolated or extrapolated PSI profile for each width regime.***
   1. The base or “true” PSI is modified to fit the new PSI profile width, The equation used to compress or expand the PSI profile is,

Eqn 55

where w is the width of either the original PSI distribution, i.e., that calculated through the process described above, or the desired new width in rows of the PSI; r is the row number. This equation simply scales the entire distribution to a new range. This new row scale is then renormalized to discrete integer rows over the new width. A simple linear interpolation of the new points to integer positions is done using the standard method: I=I1+(r-r1)(I2-I1)/(r2-r1), where r represents the x value to interpolate and I represents the interpolated luminance result. The indices represent the points bracketing the r value.

* 1. The compressed or expanded PSI profile is then corrected for the frame luminance variations using Eqn 29,

to produce the true PSI profile for comparison with real image data.

1. ***Create a filter image of the new PSI profile.*** The 1080 element PSI profile vector values are inverted to provide an additive mask. The vector is then converted to a filter image of the same size as the frame to be filtered.
2. ***Filter out the PSI on each frame.*** The corresponding filter image is added to each frame to filter out the impact of the PSI on the image.
3. ***Save the corrected frames as an avi file.***

Both Stage 1 and particularly Stage 2 turn out to be very cpu intensive. Although template generation was first modeled in Excel, it became quickly clear that Excel could not be used to generate templates on a routine basis. The time to compute a single template, even with all the simplifications described above, took over three minutes of computer time. (The PSI\_Template\_Builder macro in Projector Film Leader Tests.xlsm still contains the code.) Stage 2 clearly needed to be handled by either writing a C++ procedure, or handled in ImageJ. The familiarity with ImageJ software was a big plus. However, a simple ImageJ macro would not efficiently handle some of the other tasks needed. ImageJ scripting in another programming language was needed to handle these other tasks.

The problem was resolved by dividing the above computations into two separate scripts using Python or rather the Jython interpreter between java and python to generate the scripts to handle the template and filtering operations.

Two python language based ImageJ scripts initially handled finding the PSI center positions, and correcting each frame. PSI\_avi\_Leader\_Template\_Processor.py (This script file has been superseded by another version below and is only described here for historical continuity.):

The script output a csv file that contained the PSI center, and PSI interval distance data.

A second ImageJ script file, PSI\_remover.py, takes the csv file and the original avi file, generates a template file for each frame based on the PSI center for that frame. subtracts it from the frame data, and generates a PSI corrected avi file.

Debugging the various scripts and investigating the best factors was done using the set of 183 camcorder film leader frames.

As discussed previously, attempts to use single PSI profiles to determine the PSI center met with dismal results. The analytical results that confirmed the inherent problems with the original scheme will not be presented, because they only would serve to even further confound the reader.

# SECTION 4. PSI REMOVAL BASED ON DIFFERENCE PROFILE BETWEEN SUCCESSIVE FRAMES

The original concept of removing a PSI from video frames by finding the PSI on each video frame failed because of the difficulty in distinguishing the relatively broad and low luminance PSI shadow on real image scenes. Scenes often have similar horizontal dark and light regions, which overwhelmed the correlation coefficient method for accurately determining a PSI center. At the point of nearly admitting defeat, the idea of differential method of detecting the PSI centers gelled.

If the difference image is calculated between two successive frames, the new image represents changes between moving objects. A large portion of the scene is unlikely to change dramatically over this step of 1/18th second. With respect to PSIs, we will see the impact of two separate PSIs on this difference image. The subtraction process will remove or lower much of the confounding detail that makes PSI identification difficult on individual frames. The major luminance changes left are due to the impact of two or more PSIs and any quick action between the two frames. This simple extension of the original idea dramatically affects the ability to pick out PSI Centers. In this case, the comparison template used to compare with the image and derive the correlation coefficient is a differential PSI template. In this respect, this is related to a derivative plot. The following plot compares the two template versions.

Figure . Comparison of original two cycle PSI and the difference relationship expected for width 1358.

The differences are unremarkable, except that the rgb values are now shifted, and the cycle size of the differential template is different. In practice, it is not the images that are directly subtracted, but the image profiles, using the standard ROI xy coordinates (240,1 1680,1080) for frame n and n+1. The reason for using the profiles is that there is no such thing as a negative RGB value, so image subtraction cuts off any values outside 0 - 255.

Figures 35-37 show some correlations of 129 frames of real film to video.

Figure . PSI widths found using differential PSI correlations. The variations are a rough measure of how well the PSI center has been located.

Figure . PSI widths compared to the Pearson Correlation Coefficients.

Figure . Comparison of the PSI widths with the corresponding center found on the frame using the PCC method.

The PSI widths are a measure of how effectively the PSI centers on each frame have been identified. We do not expect large adjacent PSI widths, so a very large variation in peak widths suggest major recognition problems. Figure 35 shows what appears to be a periodic variation, but compared to single frame analysis (not shown) this is much less variation. Fig. 36 demonstrates how the Pearson correlation coefficients (pcc) varied with the width. The r2 values are all quite high, but the data exhibit two regimes, at two distinct PSI widths, around 1310 and around 1255. Despite the improvement in center positions, the results indicate we still have some sort of systematic error using the frame derivatives. By itself, this data in Figure 36 is not very useful. Figures 35, and especially Figure 37, provide the explanation for the two correlation regimes. In Figure 37, centers below 400 rows exhibited a consistent pattern of low PSI widths. The expected value of the PSI width based on the average width is 1295. However, from a statistical analysis using a confidence limit of 99% to decide which points must be kept, we can reject values outside the range 1288 - 1301 with some hope they are spurious. There is a possibility of a small trend to lower width values at higher frames, so it may be prudent to take a more conservative approach in a rejection interval. We can certainly eliminate the points below 1280 and those above 1330 with some degree of confidence we are not overly biasing the data. With this approach, the average PSI width over the 127 leader frames was 1304.

We need to do something to fix the very low values. There are many ways to do this. One is to assume a standard width to adjust any center nominally found from the pcc method using a standardized PSI width, which we can base on the average known value. A somewhat better method is to use the previous and next frame center values (as long as they are not within the same 0-400 region) and revaluate the low width frame center. The latter further has the advantage that it takes into account slow changes in the PSI width due to film speed changes. Eqn 56 is used to calculate the anticipated PSI center:

Eqn 56

where F is the frame number of the nth frame, P is the PSI center, w is the full frame width H+B, and J is the number of PSIs between the two frame centers. This equation is a straightforward derivation from the type of considerations already expressed in previous sections. It may seem confusing, because it has been reduced to its simplest form. The middle term without Jn in the numerator, is the average PSI jump from Fn-1 to Fn+1.

The final solution to reduce scatter in the PSI centers was a combination of two processes. First, the PSI centers in the 0-400 row region where interpolated using the adjacent PSI widths over the n+1 and n-1 frames as in Eqn 56, and second, a moving average process was applied to these modified values.

The following three plots interpret some of the final results of all these corrections on a set of 120 leader frames :

Figure . PSI widths corrected for row 0-400 variance and subjected to moving average smoothing.

At first glance, the data in Figure 38 may not seem much better than the original in Figure 35, but the scales are very different; the range of variation is much reduced.

Figure . Mean luminance with centers corrected for less than 400 variation and subjected to moving average smoothing as function of leader frame number.

Figure 39 plots the mean frame luminance as a function of frame number, showing that this film leader sequence exhibited a luminance change as a function of time or frame. (The region of interest (ROI) plotted is bounded by the (x,y) rectangular coords( 240, 0;1680,1080), avoiding averaging in the black side areas.) Additionally, a regular oscillating pattern of light and dark frames is evident. The pattern was confirmed by autocorrelation, but the interpretation is complex with cycles of either three or four frames.

Figure . Frame luminance as function of PSI center.

Figure 40 shows how the average frame luminance varies with the PSI center. Although there is quite a bit a scatter, the pattern is a trend from lower than average values when the PSI center is at the top of a frame, and higher values as it shows up lower down in the frame. The relationship of general frame luminance as a function of PSI center is not directly related to a type of correction that we likely have explicitly corrected for previously; it seems to amount to a cross correlation term between PSI center position and overall frame luminance. The decision was made not to pursue this level of correction, because of the extreme amount of work involved to address this issue, without evidence of a producing a dramatic benefit. Particularly, the analysis was expected to be complex to ferret out whether the variation is uniform across a frame or is the result of a luminance gradient across the frame. Another major reason for not investigating further is, in some earlier efforts to remove the PSI from real video scenes, another factor (explained below) would preclude the worth of another correction. However, the deviation was fit to a 4th order equation, and the value output in the csv file containing the PSI centers and width information, in case they were needed later. In addition, a plot of maximum luminance against the PSI center showed either a very small correction. or

The corrections applied to the PSI centers and PSI widths, may be due to artifacts of how the differential frame process calculates the best fit to find the PSI centers on frames and the running PSI widths between frames. A differential template must be created from a basic PSI profile, as displayed in Figure 34. This template is defined from a PSI of a width developed from the manual or regression optimized PSI profile. The "standard" width in the case here PSI is 1358 rows. The previous discussion on the avi example, alluded to widths that are smaller than this on the average. Creating a template based on the standard PSI then does not accurately reflect this PSI difference. However, there is a chicken and egg dilemma in this process. To get the PSI width difference from each frame, we need to know where the PSI center is on the frame, but we cannot find the PSI centers accurately without an accurate PSI width. We get stuck in a circular argument and we must start somewhere, with some assumptions. In principle, a regression process could be used, in which we "approximate" both the PSI widths and PSI centers with an initial pass through, and then repeat the process one or more times using a PSI profile and corresponding difference profile adjusted for the new widths. This leads to a huge computing task, which is already time consuming. In practice, this iteration was not used. Instead, the 1358 width and corresponding difference profile was used for the correlation comparisons. The script does contain code to generate a new difference template, but that process alters only the difference PSI profile directly. and does not generate a new profile from a width corrected PSI profile. Whether the two methods are mathematically equivalent and therefore valid, was not worked out. In defense of the current process, a new PSI profile and difference file were created with a video file which indicated an average PSI width of 1256 rows. Examining the centers produced from this distribution against those produced with a1358 version, produced *exactly* the same set of centers. An interesting observation is that the regression optimized PSI developed from these smaller than standard PSI widths, also ended up with the same width as the standard PSI. An optimist would say that this just reflects that our laboriously identified PSI width is indeed the true width. The pessimist might say the result may be due to an interaction that the luminance corrections compensate for, or some complex anti-correlating factors that just happen to cancel each other, or inherent limitations we have unknowingly forced on the regression process.

All of these basic operations to establish a set of reliable PSI centers and widths were handled by a python ImageJ script, *PSI\_avi\_Difference\_Template\_Processor2.py*, which located the PSI centers, corrected the 0-400 row centers, and obtained the moving average PSI widths.

Use of *PSI\_avi\_Difference\_Template\_Processor2.py* :

Input files and folders required (also see the earlier comments regarding folder structure:

* An avi file with the video corrected for projector vignette. ***PRE-EDITING OF A VIDEO IS ONLY ALLOWED AT THE START OR END OF THE VIDEO. IF FRAMES OTHER THAN AT THE BEGINNING AND END ARE EDITED OUT, YOU WILL LOSE IMAGE QUALITY FOR UP TO 10 FRAMES ON EITHER SIDE OF THE EDIT.***
* A general difference template representing the difference between two adjacent PSI frames. Obtaining this file uses two Excel files. The first is *F:\Canon\projector film leader reduction.xlsm, sheet [PSI\_4\_cycle], column "F".* Generation of the base PSI profile data has been described previously. This Excel file contains the manual method of obtaining the PSI profile. In addition, as previously described the PSI in this file was optimized using the file *F:\Canon\PSI and Lumnance Curves via Solver.xlsm.* The final ouput from this latter file was then transferred to the former *[PSI\_4\_cycle]*sheet as the final result, the PSI difference profile found, and the resulting data finally stored in *Canon/templating\_files/* *PSI\_2frame\_diff\_1358.csv.*
* Ensure line around 54, *DriveOption = "F"* ,points to correct drive volume letter.
* Check lines 60 to 65 point to correct folder paths as necessary.
* Script will prompt for avi file name.
* Processing time is around 50 frames/min for PSI removal and the lengthy process of rebuilding of the images into an avi file. (The latter will depend on your drive transfer rates.)
* The output file containing the recognized PSI centers for each frame, along with the corresponding moving average PSI widths, plus some other information that is not relevant to the current operations, will have the file name form *Cntrs\_OriginalAviFileName .csv*. This file will be placed in the same folder as the avi file, which is */avi\_in/*. It must always be there.

Parameters to check:

* To skip some number of initial frames change "nstart" in line 72. Frame counting starts at 1.
* Line 73 picks the number of rows to skip between successive points. 10 means use every 10th row out of 1080 or generate a template containing 108 rows to use in the pattern recognition.
* A Slicecutter value >1 analyzes only a subset of the avi slices at the interval set by Slicecutter; it is *only* useful for debugging and data validation with manually observed center data. Slicecutter must be 1 for all real video sequences.

## 4A. Removal of the PSI from real scene film footage

Early tests to remove the effect of the PSI from the leader frames were quite successful. However, when real scene footage was used, it quickly became evident that PSI removal was more complex. The large range of color and light intensity in real film added more degrees of freedom to the problem, which was not readily apparent from the much narrower range of leader luminances.

One element, not apparent from the leader data, was a washed out look to the areas with the highest degree of correction, i.e., PSI subtraction. Second, enhanced pixel color noise developed in the areas bounded by the highest amount of correction to remove the effect of the PSI. The first issue was easier to resolve than the second.

Attacking the first issue, which is related to contrast required a more complicated approach than simply subtracting the PSI profile from each frame. Resolving this issue relied on a combined PSI normalized fraction profile and contrast adjustment approach.

Regarding the contrast problem, the reason for the washed out appearance was that simply subtracting the average luminance is not appropriate. The individual r,g, and b pixels, rather than the average luminance of a pixel, need to be modified based on the fraction of grey level change desired. Simple subtraction worked for the leader frames, because of the relatively low range of luminances. However, it was also noticed that beyond the necessity of fractional change of each pixel, the response of the individual pixels with different luminance values, also was nonlinear (probably related to the gamma factor).

The final form of the correction used to adjust each color channel for each pixel is:

Eqn. 57

where pf(r,g,b) and po(r,g,b) represent r,g, or b pixel channel luminance values for the final and original pixel; fpsi is a straightforward normalization of the original PSI based on the average luminance values to a 0-1 normalized PSI profile fraction equivalent. The maximum correction we need is at the maximum of the PSI curve, with corresponding lower levels of correction away from the PSI center. fL(r,g,b) is a luminance contrast correction factor, and represents a function that changes depending on the original channel luminance. This factor has no limits and can range from negative to positive values; in general, the values fell within the range -0.1 to 1.6.

The only parameter we do not already know in Eqn. 57 is the contrast correction factor. What we need is to determine how the r,g,b luminance values change under the influence of the PSI. This requires comparing individual pixels when the influence of the PSI on a scene is a maximum and a minimum in the same region of a scene. Defining this factor required working with scene frame data directly. Three restrictions, or necessary conditions, dictated the choice of region to analyze.

1. Find a set of two frames where the PSI center, exhibits a maximum luminance, which is mirrored by a minimum luminance at a nearby frame.
2. The region chosen at or near the PSI center should contain a broad range of pixel luminances.
3. The frames chosen exhibited either no scene shift, or there was a predictable pixel shift, so the exact relationship between pixels on the two frames could be determined.

The first and second criteria were the easiest to manage. However, in regards to the first criteria, it is nearly impossible to find nearby frames where the maximum and minimum luminances occur at *exactly* the same row. Typically, it was necessary to choose frames where the row difference was within a 30 row range. From the PSI profile this difference can contribute a maximum of 0.2% luminance variation at the PSI center. The third restriction was generally very difficult to resolve. Early attempts to note any scene pixel shifts were based on rapidly shifting between the frames and noting shifts along frame edges, or using an roi in a region with sharp features and looking for shifts. However, this was only applicable to a first order . Attempting to zoom in on individual pixels at the very high resolution of the images, made determining a small pixel scene shift all but useless. Signal to noise problems contributed to complicating identification of pixel scene shifts due to the low luminances due to the PSI, which affects the efficiency of light correction by the CMOS elements. In addition, the wide difference in luminances between the two frames at the PSI center further confused shift identification. The combined result of all these problems was a very large spread in luminance values that made data interpretation very difficult.

What saved the day was a serendipitous discovery that at the end of one film sequence, the film had moved through the projector drive sprockets and had stopped on a single frame, the projector protective screen that helps avoid film burn had come down, but the shutter was still strobbing on this static film frame. It was the perfect system for determining the correction factor and inherent error spread, despite the scene itself having mostly green hues in it. The noise level in the data dropped substantially using this data, but was still higher than anticipated, especially at low luminance values. The most likely reason for the residual variation is believed to be the CMOS sensor light sensitivity. In the end, this end of film data alone was analyzed to produce the luminance contrast factor, fL The physical process to obtain the data is provided below.

Final Process using ImageJ/FIJI:

1. Find set of close together frames where a PSI center on either frame falls on or near maximum and minimum PSI luminance values and there is no pixel shift or measurable pixel shift between the two frames.
2. Set a desired roi region that has range of rgb values desired (often broad range) on first frame.
3. Duplicate range of frames with Image|Duplicate... example: range 722-724; leave stack duplicate checked.
4. On new roi stack, use (+) key to expand roi to the point that individual pixels can be seen and possibly identify feature clearly (very hard).
5. Set a 1 pixel high by n pixel wide roi on this set.
6. Duplicate the roi for first frame. Image|Duplicate with range = 1 and duplicate stack checkbox unchecked. Result will be a 1xn pixel array image.
7. Expand the image with (+) key. roi the entire 1xn row of pixels.
8. Go to Analyze|Plot Profile, copy the profile, and paste the profile into a worksheet in Excel; this is the mean luminance pixel profile.
9. Go to Image|Split Channels. Three new images appear with red, green, or blue labels added to image name.
10. Expand each channel as with primary 1xn image and get full image roi (1x n pixels), use Analyze|Profile Plot and copy the channel profile to the Excel worksheet. Clear the profile plot and expanded channel data.
11. Close the r,g,b images and the single image used to generate the channels.
12. Repeat for each channel.

See *F:\Canon\ImageJ Stuff\PixelAdjuster\_Analysls.xlsx* for data reduction and final contrast factor set.

At this point, the data represents the maximum and minimum luminance range we can expect when the PSI is at a minimum or maximum at this PSI center. As equation 57 is written, we need to change the luminance values to a factor that is multiplied by the original luminance value of the pixel. fL is simply defined as the difference between the minimum and maximum luminance at the PSI center, divided by the luminance at the PSI center: (lum. at PSI min. - lum. at PSI max.)/(lum. at PSI max.). Thus, the luminance contrast factor is the fraction change in luminance given the channel luminance of the PSI affected pixel. It was found that the r,g, b, or mean fractional factors paralleled each other, with constant shifts between the curves. Thus, color related differences were not apparent for the factors, and we can use the mean luminances to correlate with the luminance factor fL.

Next, the experimentally determined fL values needed to be correlated to the PSI mean luminances. This was a three stage process: First, the still quite scattered experimental contrast factor data were fit to the mean r,g,b values at the PSI centers using the R LOESS curve fitting routine. The resulting smoothed LOESS luminance contrast factors became the starting values for the next optimization stage. The second stage used these rough correction factors in the rgb range 22-171 as 150 adjustable parameters to an Excel Solver minimization. This range represented the full range of rgb data available. The sum of the absolute value of the difference between the calculated and experimental po(mean rgb) according to eqn. 57 was used as the variance parameter to minimize. The final factor data followed the expected trend, but still exhibited scatter. These latter optimized contrast factors, with endpoints added to complete a 0-255 rgb range, were again subjected to a LOESS fit. As both a check and a further refinement of the factors, the difference between these values and the experimental data was determined and found to fit a third order trend line, which was then used to correct the last LOESS set of factors; this correction amounted to only a few percent change in any single values.

The final luminance contrast factor set with the original data and the raw Solver solution set are displayed in Figure 41.

Figure 1. Luminance contrast factors as a function of the mean rgb value of a pixel.

The second problem of increasing noise in the low light areas was reduced by using smoothing. The high level of noise exhibited in the corrected data is most likely due to two factors: First, the PSI's large effect of reducing the scene luminance values means that the signal to noise level for each CMOS sensor at these lower values will be higher; this will be especially true around the PSI center. Second, a pixel is corrected by multiplying the r,g, or b value by a fairly large factor. This factor "amplifies" the inherent noise difference, resulting in noticeably more shade noise. The smoothing operation code simply uses the typical 3x3 averaging process, but uses it multiple times, depending on the relative position on the PSI. At the PSI center, which is where the maximum luminance alteration occurs, the smoothing operation is repeated more times than at regions further away from the center. (To save processing time, regions, rather than individual rows are smoothed.) The graded approach depends on the distance from the PSI center, and reflects the observation that the noise level is definitely related to how severe the PSI correction needs to be.

The final code used to eliminate the PSI was generated with several options. The moving average values of the interpolated PSI centers are by default used to remove the PSI bands, as well as the 0 - 400 rows modified PSI centers, but it is possible to use a standard user defined width. The smoothing operation described above is an option, because it reduces the resolution compared to just removing the PSI, and increases the processing time. Unfortunately, the need to deal with each individual pixel's three component channels severely compromises processing time. For a 50ft roll of 8 mm or super 8 mm film, expect about 10 hours to process the entire roll. Clearly, this is an overnight processing task.

Below is a summary of all the corrections applied to reduce the PSI on each frame:

1. The fundamental PSI profile itself, generated from *PSI\_avi\_Difference\_Template\_Processor2.py*, and saved in a csv file as a normalized 0-1 profile in standard width format (1358 for the present case).
2. An option to correct for changes in projector speed changes by altering the PSI with. The options are to use a moving average PSI width, a user defined width, or the standard PSI width (moving average is set by default)
3. Correction for luminance contrast variations, loaded from a file LumCorr.csv that contains 255 values representing factors that are multiplied by the mean PSI luminance of a pixel and added to the each original channel value of the pixel.
4. An optional smoothing process that reduces in a graded manner, the noise of the PSI reduced image.

Using *PSI\_remover.py*

* Input files and folders required (also see the earlier comments regarding folder structure):
* Input files:
  + An avi file.
  + A corresponding PSI centers file generated from *PSI\_avi\_Difference\_Template\_Processor2.py* (see above) for the avi file with name *Cntrs\_avifilename.csv*.
  + The PSI profile csv file *PSI\_norm\_1358.csv* in */templating\_files/*
  + A luminance correction file, *lumcorr.csv* in */templating\_files/* generated from *F:\Canon\PSI and Lumnance Curves via Solver.xlsm* , which in turn uses leader frame profiles generated by *Get Profiles of Stack Slices.py* and corresponding centers data for the frames of the avi file. (The profiles script only asks for the name of the avi file with the leader data in it and generates a profiles.csv file in the */ImageJ Stuff/* folder. Each column represents a frame row profile over the film ROI with the usual bounds.)
* Start ImageJ
* Go to *Files|New|Script* and open file *PSI\_remover.py*
* Change line around 34, *DriveOption = "F",* and change to correct drive volume letter where folders are.
* Check lines 110-133 and correct folder paths as necessary.
* lines 54-78 contain various triggers and parameters which may need to be changed. DoSmoothing, by default is set to True; it invokes the smoothing routine.
* Save script file after changes.
* Click *Run* at bottom of script window
* Script will prompt for an avi file name.
* Images will appear and disappear as the PSI is removed from the frames; to an extent, the process result can be seen. Be sure to have plenty of drive space to store intermediate high definition png files.
* Rate of processing is ~ 10 frames/min.
* When finished, the output files generated is called *stack6.avi* and is placed in the */avi\_out/* folder. Any prevous *stack2.avi* entry will be overwritten without warning.
* The script will also copy the original centers file to a new name. If the default avi output file name is *stack2.avi*, the new csv centers output file will be named *Cntrs\_stack2.avi*. You may change the name of the *stack2.avi* file in the script, and the centers file will use the new file name. The original centers csv file is not deleted. This copied csv file has no specific use now, but was used in early attempts to deal with the contrast issue.

As a check on the process and provide some idea of how effective the process using PSI\_remover.py is, leader data was used. The results below vertical profiles of four separate sequential original leader frames and the corresponding frames after PSI removal.

Figure . Final vertical luminance profile (red) compared to original frame with uncorrected PSI.

Although the removal process has substantially reduced the effect of the PSI, the removal is not perfect. The glitch peaks are not always removed. Averaging each row on the corrected frames across 118 leader frames showed a standard deviation of 5.36 at the 1σ level except at the top and bottom rows of the frames. with the conclusion that there may be a small complex relationship between luminance and PSI position. (see F:\Canon\ImageJ Stuff\leader profiles stack6 110514.xlsx). However, this deviation partly represents rather large variations because of vignetting at the top and bottom of frames. The standard deviation from row 100-1035 was 5.14. At the 2σ level, assuming a normal distribution, we have 10.72 rgb units. This variation is larger than hoped for and is noticeable in the leader frames, however, generally the variation is over the entire frame and therefore not as prominent in real scene frames. Furthermore, in light of the discussion below the variation was deemed too small to invest in further correction strategies.

As already mentioned. the leaders represent a rather restricted range of luminances compared to real scenes. Comparing variations in these two quite different cases is not a directly translatable discussion. From manual measurements and qualitative observations of various luminance and color ranges, the adjusted PSI centers luminances appeared about 8% to bright when the original luminances where in the range of greater than 150. However, several attempts to alter the luminance correction factors to reduce this variation where not successful. In darker regions, the PSI removal appeared better. Potentially, a major problem which may have been at play here is that resolution cannot ever be totally preserved because luminance information must be lost in the darkest and lightest regions due to CMOS sensor noise, or CMOS sensor saturation. The only way to recover some of this information is by averaging adjacent frames.

## 4B. Final operations

Although the preceding efforts were successful at removing most of the PSI, the leader data and the previous discussion acknowledge that residual luminance artifacts up to around 10% were still present. The result is still flickering video , but at a less distracting level than on the original frames. To some, this flicker may be either acceptable or desirable to suggest vintage film footage. The residual artifacts can be removed by using a final ImageJ script which combines images on a moving average basis. *PSI\_avi\_Average\_Processor\_v3.py*, combines images based on a user setting of the number of images to combine. There is no weighting factor used in this process, other than the moving average process.

Averaging frames is akin to reducing the resolution of the image and increases smearing and ghosting effects, so only enough frames should be averaged to reduce flicker to an acceptable level. What was found in general is that removing the PSI from each frame in the manner described here, allows no frames or fewer frames to be averaged to produce a flickerless video, than if we just averaged the frames and did no PSI luminance correction at all. Using a two frame moving average nearly completely removes the flicker, and a three frame moving average completely removes flicker, albeit with a concomitant increase in ghosting, especially in frames with fast movements.

Using *PSI\_avi\_Average\_Processor\_v3.py*:

1. Start ImageJ.
2. Go to *File|New|Scripts* and open the python script file in whatever directory was used (e.g., */ImageJ Stuff/*).
3. Around line 68 in the script change: *DriveOption = "F"* to the drive letter on which the files reside. (If you have decided not to use the original directories you will need to find and change all the directory information as well.) Save the script.
4. Click the Run button below the script window.
5. The script will ask for the input avi file. A message may appear that files in the folder */imagedump/* need to be removed. Remove all files and folders from i*magedump* and restart the script.
6. A window will pop up asking for the number of frames to average, two will produce an acceptable image with a slight flicker and three will eliminate the flicker. The trade off of increasing the number of frames to average is a pronounced ghosting or "stuttering" of frames with large motion changes from frame to frame.
7. Click *Ok* and a new message will pop up indicating that the script will be running despite the appearance it is not. Click *Ok on this window*. To see if script is actually running, click on the ImageJ menu bar itself to bring it forward. The status area below the menu should be rapidly changing. Only a static image will be observed on the monitor.
8. Images are processed at a rate of ~65 frame/min.
9. When complete a message will appear asking if you wish to delete the images. The best answer is to affirm deletion of the images, but doing so will also delete the *stack3.avi* on the screen, because it is based on a virtual avi file and uses the images stored on the drive rather than RAM memory.)
10. *stack3.avi* in */avi\_out/* will contain the final process images. Previous data will be overwritten, so it is advisable to copy and change the name of this file to a new name.

## 4C. Limitations and Future Directions

After converting a series of 28 8 and super 8 mm films to video, a few limitations or issues were noted. For most films with good exposures and moderate action, the results are quite good, if the directions are followed. However, the many steps to improve the video quality, usually based on averaged properties, can introduce their own set of artifacts. In most cases, the final result is still far better than the original camcorder video of the film. There was one case where the cure was worse than the disease. Very dark, underexposed scenes, ended up reversing the PSI black bar to an overexposed white bar, with an even worse result than the original camcorder video of the film. In these cases, the only recourse was not to go through the bother of using the PSI remover app and just use *PSI\_avi\_Average\_Processor v2.py* to average over at least three frames.

A second issue is the critical care necessary to remove the vignette due to non uniform illumination by the projector lamp. Overly aggressive application of the mask to remove the vignette, can lead to over exposure of the edges of the images; the result will be an annoying edge flicker. A bit of underexposure is preferred to overexposure. The artifact introduced is likely introduced because the PSI shadow is assumed to be approximately 60 rgb units. The vignette correction alters this assumption, but is not corrected for in the existing removal process. An underexposure can then be rectified to some extent using the Video FX|Levels|Brighten filter in Sony Vegas Movie Studio, or something similar in other video editor software.

From the perspective of my future input, I am exhausted after three years off and on work on this project. Moreover, I have achieved the goal I set out to do for my own film conversion. I doubt I will be offering any major improvements; other projects are pulling my interest in completely different directions. There are elements where I believe more work could lead to improvements. I believe the pattern recognition process to find the PSI centers worked our quite well, and I suspect needs little or no improvement. However, my intuition nags me that the PSI subtraction process and contrast manipulation can be improved, but needs the hand of experts in image luminance methods to improve the process. Of course, pulling together the codes more efficiently is also an additional big issue.

**SECTION 5. THE I-DON'T-CARE-ABOUT-BEST-QUALITY,-BUT-I-AM-CHEAP-AND-IMPATIENT-AND-STILL-WANT-TO-PRODUCE-AN-"OKAY"-VIDEO-OF- MY-FILMS METHOD.**

This section does not absolve you from reading some sections of this report, and downloading and setting up ImageJ/Fiji. Read carefully all of Sections 2C.1 and 3A.1. From the zip file, you will want to download at a minimum the *PSI\_avi\_Average\_Processor v2.py* script, and place it in the appropriate directory. To minimize grief, it is best to just copy all the files in the folder structure in the zip file; you can then delete those that do not matter. You must set your camcorder or camera video to TV mode and record at 1/100 a second or higher exposure. Once you have recorded your videos, and converted the video to an uncompressed avi file, using Virtualdub, you will use the script discussed in Section 4B to reduce the flicker from your videos. All 4B does is average a user-defined number of images to reduce flicker. First, work with only a small subset of frames, 50 - 100 should be sufficient to get a feel for what the outcome will look like, and you will not feel like you have grown older waiting for processing to complete. Start with a frame averaging of 3. Also, make sure you have plenty of free drive space available - at least 150 GB.

What are the tradeoffs in just using the averaging method versus the long drawn out projector shadow removal process? The videos developed from just the averaging process will be darker and require post editing to increase brightness and probably contrast in the final video. More frames must be averaged to get to the same amount of residual flicker found in the videos subjected to the shadow removal process. This means image resolution suffers more. The end result will be greater smearing and ghosting, i.e., loss of resolution, that will especially be visible in fast action movement sequences, compared to the frame by frame shadow removal process.

Keep in mind there is an inverse relationship between recording film at higher exposure times, such as 1/60th second, which reduces the projector shadow, and the number of frames that need to be averaged. The greater the exposure time, the fewer frames need to be averaged, but the effect on quality of the video is inescapable - you will lose resolution because of action smearing. The averaging method does nothing to remove the vignette that most projectors produce because of non uniform lighting across the projected film. (If you wished to go the extra five miles of work to remove the vignette before the averaging process, see Section 3.2)

**SECTION 6. EDITING A FINAL AVI VIDEO IN VIRTUALDUB AND SONY VEGAS MOVIE STUDIO, AND STORING VIDEOS**

A necessary step is the masking of the video to remove edge effects and remove the grey sides on each frame due to the different HD and 8mm aspect ratios. If you are using Pro versions of movie editing software, creating a mask is likely easy. With at least Sony Vegas Movie Studio up to version 12, there is no simple border masking filter and setting up a mask is more involved (see below). A better approach, although not without its own set of twists, is to use VirtualDub to apply the mask. VirtualDub has a great filter which makes simple border masking easy - *Border Control 2.34*. However, I found the default VirtualDub settings will cause an unacceptable periodic stutter in the final uncompressed avi video. To remove this stutter required some adjustments to VirtualDub. First make sure you have a codec package such as ffdshow installed; this should really be done right after installing VirutalDub. In ffdshow, go to *Preferences|Compression, ffdshow|Configure* and choose HuffYUV encoder, not the mjpeg encoder. In VirtualDub, check the compression type: go to Video|Compression. If not selected click on "ffdshow Video Codec", then click Configure to make sure the HuffYUV is selected. Close the Compression windows. Next, go to *Video|Filters|Add...* and locate *Border Control 2.34.* If it is not in the filter set you will have to find a VirtualDub plugin package that has the filter. Also, in VirtualDub make sure you set *Video|Full Processing Mode* and set *Audio|No audio*. You can make life a bit simpler for routine use by saving the current settings, use *File|Save Processing Settings*, you can then load the settings when starting VirtualDub.

Several helpful *.vdscript* files can be found in the *Canon\virtualdub stuff\* folder; *BlackMaskFilter.vdscript* loads and applies the above options. Run the script when you open VirtualDub. However, when the video is opened or the filter script loaded, for reasons not clear, you may need to open up the Border Control filter, *(Video|Filters...|,* open the filter,click Ok and then click Ok again to get back to the main screen. The mask will automatically change from a non black mask to a black mask.

Of course. the mask set up depends on how much of the HD frame was used. In my case, the border widths were set to 240 pixels at the left and right sides, and 45 pixels at the top and bottom. VirtualDub processes the file very fast.

I cannot speak to other movie editing software, but if you use Sony Vegas Movie Studio, at least up to version 12, you are not done with hand wringing setup. Vegas will not natively read *Huffyuv* compressed avi files, even though Media Player and other players will recognize and play the compressed files. You really do want to do at least lossless compression at this stage. The avi file size will be substantially reduced in the range of a factor of five by using lossless compression.

To get Sony Vegas Movie Studio V11 to read the lossless compressed huffyuv file, go to the already installed *ffdshow*, run the VFW (video for windows) app, click on *codecs* in the right window, locate the *huffyuv* entry and change it from disabled to enabled, and save the change. Sony Vegas will then be able to open up the compressed avi file for editing in Vegas. (I profess no expertise or understanding of this codec stuff, which remains mostly magical to me, other than to say that from what I have read, *ffdshow* is sort of an intermediate translator.) I can say that after doing the above, Sony Vegas Movie studio opened up the *Huffyuv* compressed video for editing with no problems.

If you prefer to avoid the border masking using VirtualDub and wish to instead use Sony Vegas Studio or other commercial software instead, the process could be messy. You may need to use multiple video channels. There is no simple mask function in Vegas Movie Studio, only the much more expensive Pro version. However, one kludgy way of creating the necessary mask is the following: Load the uncompressed **a**vi video into Vegas Studio. Click on the Text channel, go to the *Media Generation* tab and drag and drop a Solid Color background on the Text Channel. Edit the color so it is black. Right click on the Text Channel and pick the crop/pan/zoom function. Adjust the numbers and black bar so that it covers the left 240 pixels of the frame. Create a duplicate text channel from the modified one; make sure it is not a child of the previous text channel or the video. Again go to crop/pan/zoom and manipulate the bar to the right side of the image to cover the right 240 pixels. Repeat the process for the top and bottom masks. Extend the four text channels to the full video length. The preview should now indicate the image properly masked to just the film size. A way to reuse the mask again is to just create the mask as above, without a film video channel, and save as a Vegas Project. Sony Vegas Movie Studio allows more than one instance to be open at a time. Open a second instance of the program, load the film avi, switch to the mask project, click on any of the far left side channel descriptions, and click Ctrl-A to copy all the channels. Switch to the film video file, click on the left side and *Edit|Copy* will add all the channels from the mask project. You can then adjust the length of the mask channels to the size of the video file.

Regarding storing the avi files: As mentioned previously, ImageJ only works with uncompressed avi files and even a couple of minutes of 8mm film are 30-40 GB files. The original .mts file is a relatively small file, on the order of 600 MB. Even though this file would require going through the entire process again, it is a reasonable to store this file, if you hope that in the future a much better converter becomes available (not from me!). For quicker access, the original vignette corrected avi file, and the final file could be stored. If the final file is compressed as suggested in the sequence discussed for adding the mask in VirtualDub, then the file sizes are on the order of 6-8 GB, which is a easier to store to media such as a blu-ray disk. The original file can be compressed using VirtualDub ( Avisynth is also popular), but must be restored to a noncompressed avi file using VirtualDub or other utility to reuse in ImageJ. In VirtualDub, use *Video|Compression...|Uncompressed* to generate the uncompressed file. A bit more compression can be accomplished if the file in converted to an mp4. which will create a compressed file around 1/20 the original uncompressed file size.

I preferred to render the files as *.mp4* files.

# APPENDIX

## COMBINED ROUTINE STEPS TO CONVERT AN .MTS FILE WITH RECORDED 8MM FILM TO A DIGITAL VIDEO

*The following outline is a summary of the routine process sequence to produce digital video output from a camcorder recorded film. To use in the sequence of programs and scripts below, the following must already be generated and present as described in detail in the main body of this article:*

* *The original folder tree as copied, or every script updated with new folder and filenames as necessary.*
* *Digital video of the projected film as an HD (1920x1080) .mts or other HD H263 format.*
* *A .bmp file of the color reversed image of the projector illumination vignette of the film.*
* *A PSI profile csv file, named PSI\_norm\_1358.csv in Canon/templating\_files/*
* *A PSI Difference template file named PSI\_2frame\_diff\_1358.csv in Canon/templating\_files/*
* *lumcorr.csv* in *Canon/templating\_files/*
* *(The latter three file names may be changed as long as the names have also been changed in all the appropriate ImageJ scripts.*

Once over the semi-manual methods for producing a vignette mask and the PSI shadow profile, the use of the ImageJ scripts easily drops in to a rote pattern.

1. Use VirtualDub to correct the vignette for a group of *.mts* videos (I liked to work in groups of four) and place the results in *Canon\avi\_in\*.
2. Switch to *ImageJ*.
3. Run *PSI\_avi\_Difference\_Template\_Processor2.py*  on the vignette corrected avi in *Canon\avi\_in\*; a new file - with the prefix *Cntrs\_* and the suffix *.csv* will appear in *\avi\_in\;* the avi file is unmodified.
4. *Run PSI\_Remover.py* on the same corrected avi, which generates the output avi file - *Canon\avi\_out*\*stack2.avi*.
5. If the video is acceptable, delete the roughly 6000 image files in *Canon\imagedump\.*
6. Run the *PSI\_avi\_Average\_Processor2.py* script on the *Canon\avi\_out*\*stack2.avi*., which generates the file - *Canon\avi\_out*\*stack3.avi*. Acknowledge deleting the *\imagedump\* frame images.
7. Rune VirtualDub, (usually by loading the black mask filter script to mask the useless parts of the video frame.
8. Take the output avi and use your favorite movie editor to create your final video.
9. Repeat steps 2 - 6 until all four files have been processed, then restart the entire sequence on the next batch of *.mts* files.

*Expect that your computer will be generally unusable while the scripts below are running.*

*Using the Hotspot filter in VirtualDub to remove the projector vignette and convert the .mts file to .avi:*

1. Start *VirtualDub*. As explained previously, you should have already struggled to add all the filters and plugins to open *.mts* files as outlined in installing *VirtualDub*.
2. Load the *.mts* file.
3. Go to *Video|Filters* and *Add..* the *Hotspot* filter. When the Hotspot parameter window opens, browse to the vignette *.bmp* image file you created elsewhere. Change the sliders to your defined multiplier and additive parameters which you have already optimized. Click Ok.
4. Set the following parameters in *VirtualDub: Video|Full Processing Mode*; *Options|Performance...*, if you have the memory available, set the sliders to higher buffer numbers than the default; set *Options|Preferences|Process Priority to Highest* (for fastest conversion rates); set *Audio|No Audio*. There are other parameters, such as start and end frames to process that you may wish to set using *Video/Select Range..*. Optionally, you may wish to save your settings to use later. Click *File|Save Processing Settings...* *File|Load Processing Settings...* can be used to bring the saved setting back at any time.
5. Click *File|Save as avi...,* and save the filtered avi file with a suitable file name in the *Canon/avi\_in/* folder*.* I usually appended "corr" somewhere in the original file name.
6. When processing starts, uncheck the *Show input* and *Show output* checkboxes in the status window; this will speed up processing a bit. Removing the vignette on all frames will take ~9 minutes to complete for ~6000 frames. The avi file size at this point will be as high as 40 GB.

Use of *PSI\_avi\_Difference\_Template\_Processor2.py* :

Input files and folders required (also see the earlier comments regarding folder structure:

1. Start ImageJ
2. Go to File|New|Script and load the script *PSI\_avi\_Difference\_Template\_Processor2.py* from */ImageJ Stuff/*
3. Click *Run* on lower frame*.* Script will prompt for avi file name that should be in *Canon|avi\_in|*.
4. The images being processed will flash on the screen.
5. The output file containing the recognized PSI centers for each frame, along with the corresponding moving average PSI widths, plus some other information that is not relevant to the current operations, will have the file name form *Cntrs\_OriginalAviFileName .csv*. This file will be placed in the same folder as the avi file, which is *Canon/avi\_in/*. It must always be there.

Parameters to check:

To skip some number of initial frames change "nstart" in line 72 of the script. Frame counting starts at 1.

Using *PSI\_Remover.py*

* Input files and folders required (also see the earlier comments regarding folder structure):
* Input files:
  + An avi file.
  + A corresponding PSI centers file generated from *PSI\_avi\_Difference\_Template\_Processor2.py* (see above) for the avi file with name *Cntrs\_avifilename.csv*.
  + The PSI profile csv file *PSI\_norm\_1358.csv* in */templating\_files/*
  + A luminance correction file, *lumcorr.csv* in */templating\_files/* generated from *F:\Canon\PSI and Lumnance Curves via Solver.xlsm* , which in turn uses leader frame profiles generated by *Get Profiles of Stack Slices.py* and corresponding centers data for the frames of the avi file. (The profiles script only asks for the name of the avi file with the leader data in it and generates a profiles.csv file in the */ImageJ Stuff/* folder. Each column represents a frame row profile over the film ROI with the usual bounds.)
* Start ImageJ
* Go to *Files|New|Script* and open file *PSI\_remover.py*
* Change line around 91, *DriveOption = "F",* to correct drive volume letter where folders have been downloaded.
* Check lines 210-244 and correct folder paths as necessary.
* lines 54-79 contain various triggers and parameters which may need to be changed. DoSmoothing, by default is set to True; it invokes the smoothing routine.
* Save script file after changes.
* Click *Run* at bottom of script window
* Script will prompt for an avi file name.
* Images will appear and disappear as the PSI is removed from the frames; to an extent, the process result can be seen. Be sure to have plenty of drive space to store intermediate high definition png files.
* Rate of processing is ~ 10 frames/min.
* When finished, the output file generated is called *stack2.avi* and is placed in the */avi\_out/* folder. **Any prevous stack2.avi entry will be overwritten without warning.**
* The script will also copy the original centers file to a new name. If the default avi output file name is *stack2.avi*, the new csv centers output file will be named *Cntrs\_stack2.avi*. You may change the name of the *stack2.avi* file in the script, and the centers file will use the new file name. The original centers csv file is not deleted. This copied csv file has no specific use now, but was used in early attempts to deal with the contrast issue.

Using *PSI\_avi\_Average\_Processor2.py*:

1. Start ImageJ.
2. Go to *File|New|Scripts* and open the python script file in whatever directory was used (e.g., */ImageJ Stuff/*).
3. Around line 68 in the script change: *DriveOption = "F"* to the drive letter on which the files reside. (If you have decided not to use the original directories you will need to find and change all the directory information as well.) Save the script.
4. Click the Run button below the script window.
5. The script will ask for the input avi file. Typically, this will be stack2.ave, which is the output from *PSI\_remover.py*, unless you have renamed the file. A message may appear that files in the folder */imagedump/* need to be removed. Remove all files and folders from /*imagedump/* and restart the script.
6. A window will pop up asking for the number of frames to average, two will produce an acceptable image with a slight flicker and three will eliminate the flicker. The trade off of increasing the number of frames to average is a pronounced ghosting or "stuttering" of frames with large motion changes from frame to frame.
7. Click *Ok* and a new message will pop up indicating that the script will be running despite the appearance it is not. Click *Ok on this window*. To see if script is actually running, click on the ImageJ menu bar itself to bring it forward. The status area below the menu should be rapidly changing. Only a static image will be observed on the monitor.
8. Images are processed at a rate of ~65 frame/min.
9. When complete a message will appear asking if you wish to delete the images. The best answer is to affirm deletion of the images, but doing so will also delete the stack3.avi on the screen, because it is based on a virtual avi file and uses the images stored on the drive rather than RAM memory.)
10. *stack3.avi* in */avi\_out/* will contain the final process images. Previous data will be overwritten, so it is advisable to copy and change the name of this file to a new name immediately, if this is your final file.

You can expect the total cpu time for these operation to be on the order of 24 hours.

# Copyright Information

*Copyright March 7, 2015. All Rights Reserved. The use of the concepts described here and implemented in the macros, scripts, and Excel files referred to and/or accompanying this document are allowed for non commercial use only.*

*Contact the author Gary Dyrkacz at dyrgcmn@comcast.net for other uses.*