Fall 12-2022

COLOR RESOLVED CHERENKOV IMAGING ALLOWS FOR DIFFERENTIAL SIGNAL DETECTION IN BLOOD AND MELANIN CONTENT

Vihan A. Wickramasinghe
Dartmouth College, vihan.wickramasinghe.th@dartmouth.edu

COLOR RESOLVED CHERENKOV IMAGING ALLOWS FOR DIFFERENTIAL SIGNAL DETECTION IN BLOOD AND MELANIN CONTENT

A Thesis
Submitted to the Faculty
in partial fulfillment of the requirements for the
degree of

Master of Science

in

Engineering Sciences

by VIHAN WICKRAMASINGHE

Thayer School of Engineering
Guarini School of Graduate and Advanced Studies
Hanover, New Hampshire

September/2022

Examining Committee:

Chair________________________

(Brian W. Pogue)

Member______________________

(David Gladstone)

Member______________________

(Petr Bruza)

____

F. Jon Kull, Ph.D.

Dean of the Guarini School of Graduate and Advanced Studies

ABSTRACT

Cherenkov imaging in radiation therapy allows a video display of the irradiation beam on the patient's tissue, for visualization of the treatment. High energy radiation from a linear accelerator (Linac) results in the production of spectrally-continuous broadband light inside tissue due to the Cherenkov effect; this light is then attenuated by tissue features from transport and exits from the delivery site. Progress with the development of color Cherenkov imaging has opened the possibility for some level of spectroscopic imaging of the light-tissue interaction and interpretation of the specific nature of the tissue being irradiated. Generally, there is a linear relationship between Cherenkov emission and dose in a homogenous medium; however human tissue has multiple factors of scatter and absorption that result in the distortion of this linear relationship. This project investigated what color Cherenkov imaging could be used for, in the situation of tissue with different levels of pigmentation present in skin and/or different levels of hemoglobin present inside the tissue. A custom-developed time-gated three-channel intensified camera was used to image the Red Green and Blue (RGB) Cherenkov emission from tissue phantoms that had synthetic epidermal layers and blood. The hypothesis was that RGB color Cherenkov imaging would allow for the detection of signals that varied uniquely in these channels in response to changes in blood content or melanin content, because of their different absorption spectra in the RGB channels. Oxy-hemoglobin in the blood is highly absorbing in the blue & green, but not as much in the red, whereas the melanin is highly absorbing across the channels, falling slightly from blue through green and red. The results showed that these spectral absorption differences did indeed lead to different amounts of exiting light, predominantly in the red wavelength band, where melanin has a higher relative absorption than blood. This observation leads to the provision for future color distortion corrections, and interpretation of more accurate Cherenkov imaging via color-based modeling or correction for dose quantification. Based on this work, it is possible to separate the effects of attenuation from skin color or blood volume based upon the colors seen in the Cherenkov images, as these are emissions that are specific to the patient.

Table of Contents

List of Figures

List of tables:

Acknowledgments:

i. I want to express my indebtedness and appreciation to my advisor Dr. Brian Pogue for his leadership, patience, and support through hurdles of global uncertainty and troubled times. I am thankful for your guiding hand and flexibility in this project and for helping me understand a wonderful field better.

ii. I would like to thank Dr. Rongxiao Zhang as well for his kind intervention in my project from time to time. Thank you for providing empathetic and supportive guidance through challenging technical and personal hurdles.

iii. I would like to thank Dr. David Gladstone for his brilliant support in both academic and administrative matters. I am fortunate to have had you drive and push for the success of my project through thick and thin.

iv. I would like to thank Dr. Petr Bruza for his advice as well as technical wizardry and prowess in making a novel and unique camera, without which this study wouldn’t be possible. Thank you for your tireless work.

v. I would like to thank DHMC for allowing my research to be conducted safely and without worry. The staff provided a comfortable environment for me and my colleagues. Thank you for helping advance research in the eradication of cancer.

vi. I would like to thank my family for their sacrifice, love and faith. I would especially like to thank my wife Miya Wickramasinghe for keeping me grounded despite the world tumbling around me. When I felt myself slip underwater, you’d always catch my hand. Your unconditional love and support are why I attribute my success to you.

vii. Lastly, I want to thank Dr. Michael Walters, Dr. Ty Kretser and Dr. Ken Podolak from SUNY Plattsburgh for believing in me and being my family through my early college years. As educators, you did the right thing and nurtured me to be better in every way, and held me to high standards. You gave me a ground to stand on and a sky to reach for. Thank you for believing in this small-town boy.

Introduction

The time-gated tri-channel intensified camera used in this study is a proprietary device that is unique and one of a kind, designed by Dr. Petr Bruza at Dartmouth College. It was crafted to image Cherenkov radiation emission in three separate channels, specifically red, blue, and green. At the time of writing, there has yet to be any published independent validation of its performance with regard to confirming a differential Cherenkov signal response with regard to changing blood and melanin content. The purpose of this project was to show that color Cherenkov imaging would allow the detection of signals that varied differently in response to changes in blood content or melanin content, and compare and contrast this variable response. Measurements of the mean Cherenkov signal in each channel were done in-house crafted blood solutions and synthetic epidermal layers, with varying concentrations. Analysis was done using CDose and MATLAB.

1.1 Radiotherapy

A key goal in this project is to provide practical insight into better point-of-care treatment via improved clinician guidance in the field of radiotherapy through visualization of dose and using the detection of differential signal response. Radiotherapy is used in the majority of cancer patients, where the radiation dose is maximized to the target volume, while minimizing dose to the surrounding normal tissues, with minimal exposure of personnel and adequate patient monitoring at determining the end result of the treatment[1]. For the management of this tool to proceed smoothly, a major aspect is proper monitoring of the delivered radiation dose to tissue. In this setting, proper quality control is needed to reduce uncertainties and errors in dosimetry, treatment planning, treatment delivery, etc., raising tumor control rates as well as reducing complication and recurrence rates. This is described as the regulatory process through which the actual quality performance is measured, compared with existing standards, and the actions necessary to keep or regain conformance with the standards, being one part of overall quality assurance, with a primary goal of checking that quality requirements are met and adjusting and correcting performance should the requirements be found not to have been met[1]. As such, ideally, radiotherapy needs to provide consistent and safe dose delivery, accounting for dose distribution, patient alignment, and day-to-day anatomy changes. At the time of this writing, no available technique can quickly and readily provide information regarding the dose distribution with respect to patient's anatomy in real-time using stand-alone tools[2]. However, Dartmouth's Optics in Medicine lab has pioneered the implementation of a phenomenon known as Cherenkov emission as a tool in photon and electron beam radiation therapy. This technique was recently advanced into a color-sensing Cherenkov camera[3]. . In this project, by visualizing dose through the revolutionary process of Cherenkov imaging, improved quality control can be achieved through the recognition and correction of color-based factors that affect Cherenkov radiation.

1.2 Cherenkov Radiation

Cherenkov radiation is the phenomenon observed when a charged particle, in this case, an electron, passes through a dielectric medium traveling at a speed greater than the phase velocity of light, thus resulting in an electromagnetic shockwave emitted and commonly observed as a blue glow. The signal is seen as blue in water or air but is actually broadband and across the entire UV/visible/IR spectrum. This effect will be better demonstrated with visuals.

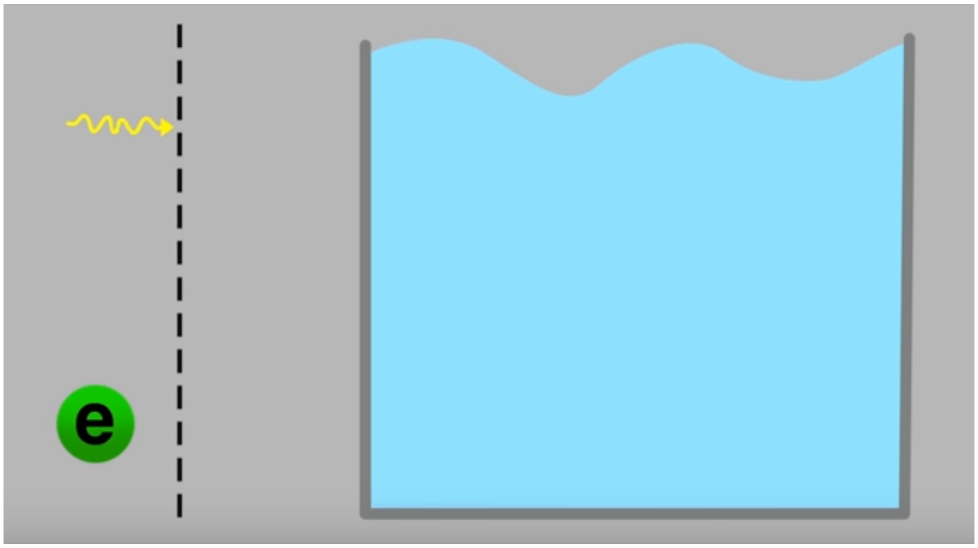

Figure 1 Charged particle (an electron) and photon shown before entering a tank of water.

In Fig 1 above[21], a charged particle and photon are shown to be about to enter a tank of water. The charged particle is slower than light in air and is a fraction behind the photon. Upon entry, due to slowing down in the dielectric medium, light will refract and bend, while the charged particle has a possibility of continuing on with a velocity higher than light (due to this slowdown).

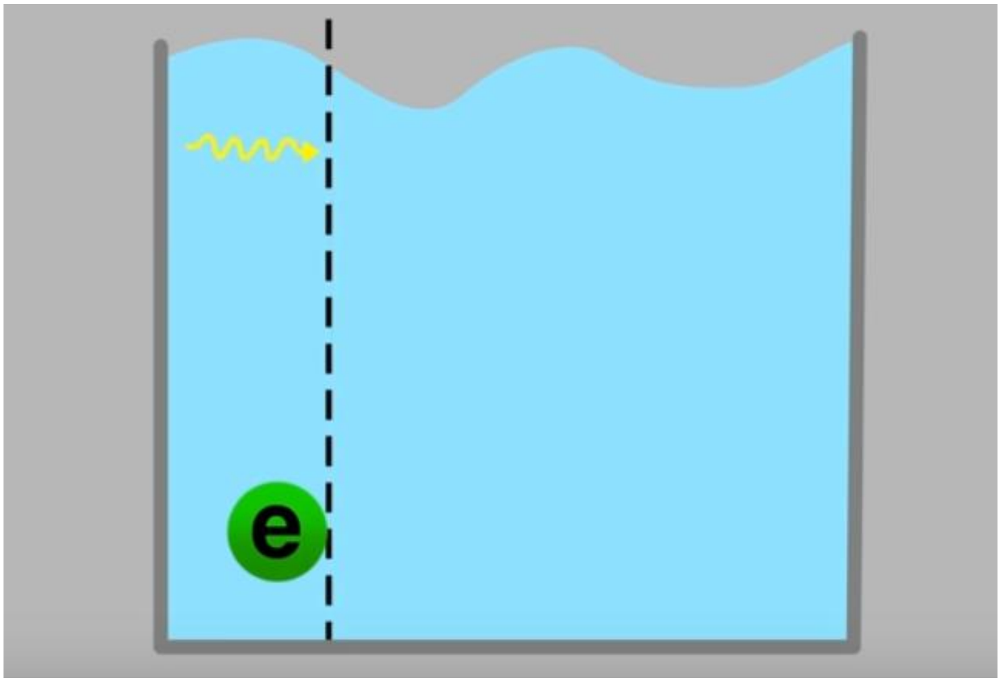

Figure 2 Electron and light particle in tank of water, shown to have different speeds. The charged particle is now faster than light.

During the motion of the charged particle through the medium, it interacts with the medium exciting atomic electrons or ionizing atoms along the way. The electric field of the charged particle disrupts the electrons and those disruptions cause a cascade. Because the charged particle is also, in the context of the dielectric medium, traveling faster than light, these cascades or electromagnetic waves will begin to overlap and constructively interfere, as shown in figure 3 below[22].

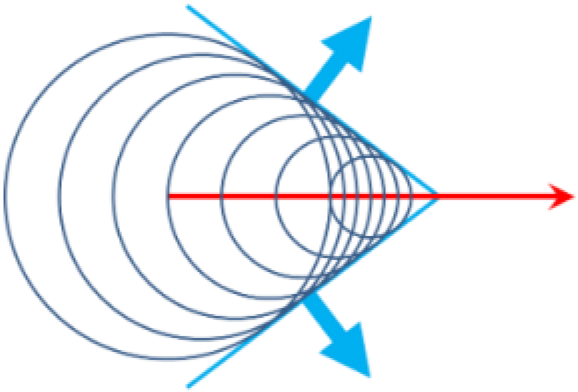

Figure 3 Constructively interfering waveform due to charged particle being faster than light in dielectric medium.

This phenomenon is similar to the sonic boom effect, where mechanical soundwaves overlap constructively when an object travels faster than the speed of sound. Similarly, when electromagnetic waves constructively interfere, the energy of the photon increases and the wavelength shortens. The refractive index of a material depends on the optical frequency or wavelength, a dependency known as chromatic dispersion. Figure 4 below[24,25] illustrates this relationship.

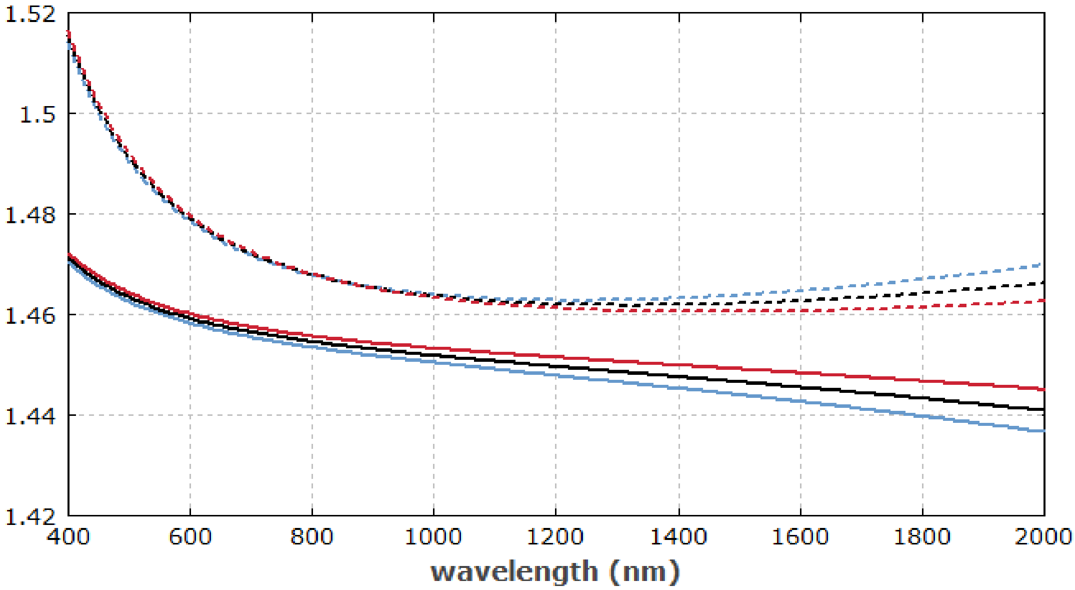

Figure 4 Refractive index (solid lines) and group index (dotted lines) of silica versus wavelength at temperatures of 0 °C (blue), 100 °C (black) and 200 °C (red). The plots are based on data from M. Medhat et al., J. Opt. A: Pure Appl. Opt. 4, 174 (2002).

This is important given that as the wavelength approaches the x-ray range most materials' index of refraction will approach 1. Consequently, this means a photon's wavelength will shorten until it is able to travel through the medium faster than the particle is traveling, stopping the constructive interference for that photon as it travels out as Cherenkov radiation[23]. For a given initial electron energy spectrum, the emitted Cherenkov signal generated is directly proportional to the deposited dose. The practical outlet of this observation is that the signal might be used as a means of quantitatively mapping radiation dose in humans.

The exploration of this physical phenomenon within a biological therapeutic setting is a major aspect of this study.

1.3 Cherenkov Imaging Restrictions

While Cherenkov imaging is a fascinating tool, it does come with a few restrictions that need to be studied both at a fundamental phenomenon level as well as to potentially overcome some of the limitations for practical use. The origins can be modeled by Monte Carlo simulation of the radiation interaction cascade, where the only major factors influencing the intensity generation are the local index of refraction and the energy of the charged particles. However, exiting Cherenkov signals are also affected by attenuation from the intrinsic tissue optical properties of the patient, with Monte Carlo simulations estimating tissue absorption and scattering events to contribute up to 45% variation in the detected light[26,27]. Studying the differences between individual tissues in the context of Cherenkov emission intensity can prove to be a valuable diagnostic tool. Visualizing the interaction of dose with tissue was only a recent devlopment[5] but it can be leveraged to further showcase differential Cherenkov signal response.

To accomplish this task, a proprietary in-house tri-channel camera fitted with a time-gated image intensifier per channel (red, green, blue) was used to capture cumulative images of Cherenkov emission from tissue phantom samples irradiated with x-rays. Previous studies followed suit in this matter[3] successfully, based on the premise of gating radiation therapy LINAC pulses. Specifically, time-gating allowed for the detection of low-intensity Cherenkov emission signal above background ambient light levels[3,13] and therefore the study and analysis of emission intensities. This imaging setup helped explore the hypothesis that multi-spectrum color Cherenkov imaging allowed for differential signal detection in blood and melanin content, and thus provide practical insight into better point-of-care treatment via improved clinician guidance.

Materials and Methods

1.4 Color Cherenkov camera

A color Cherenkov camera that captures red, green, and blue (RGB) wavelength channels separately was used in this project. Acquisition was time-gated to the LINAC, capturing Cherenkov emission only during radiation pulses, thereby removing background ambient light. This setup is shown in Fig 1 (a). Image acquisition and processing were largely in line with prior work by Dr. Alexander[3] (Alexander et al. Light: Science & Applications (2021)10:226 Page 6 of 7) with changes being only the MATLAB processing, where mean intensities for each individual channel were obtained from consistent ROIs on frame-averaged image stacks. To simulate tissue, synthetic epidermal layers 100 μm (+/- 5 μm) thick were fabricated, with varying biological concentrations of melanin (0.0018, 0.0038, 0.0076, 0.0114, 0.019, 0.027, 0.045 and 0.072 mg/ml). The epidermal layers were placed on top of 2-cm thick bulk tissue phantoms made of silicone with flesh-colored pigment. Average optical properties (absorption coefficient, μ_a, and reduced scattering coefficient, μ_s’) of the phantoms were determined to confirm tissue-like optical behavior for a range of human skin colors. Respectively, blood solutions with varying biological concentrations of 0.5%, 1.0%, 1.5%, 2.0%, 2.5%, 3.0%, 3.5% bovine whole blood were concocted and poured into blacked out petri dishes. During LINAC irradiation of the phantoms, images were captured for each color channel, and post-processing extracted average RGB Cherenkov intensities versus simulated skin melanin concentration.

The RGB color Cherenkov camera, as seen in Fig 5 (b), was composed of three independent intensified CMOS (iCMOS) cameras (C-Dose, DoseOptics LLC, Lebanon NH USA) housed in a three-tube color video camera assembly (JVC, Yokohama, Japan). The red channel was outfitted with a red-sensitive intensifier and red filter, the blue and green channels were blue-green sensitive intensifiers, and appropriate color filters for each. Each camera was remotely triggered to stray X-rays, allowing for synchronization with and gating to the linear accelerator pulses; the schematic is shown in Fig 5 (d). The beam splitter assembly of the video camera, consisting of RGB dichroic beam splitters and bandpass filters, allowed for incoming Cherenkov light signals to be redirected according to wavelength to the appropriate camera channel resulting in three raw image stacks for each acquisition, shown in Fig 5 (c). The camera was equipped with a 10–100 mm, f/1.6 zoom lens (JVC, Yokohama, Japan)[3]

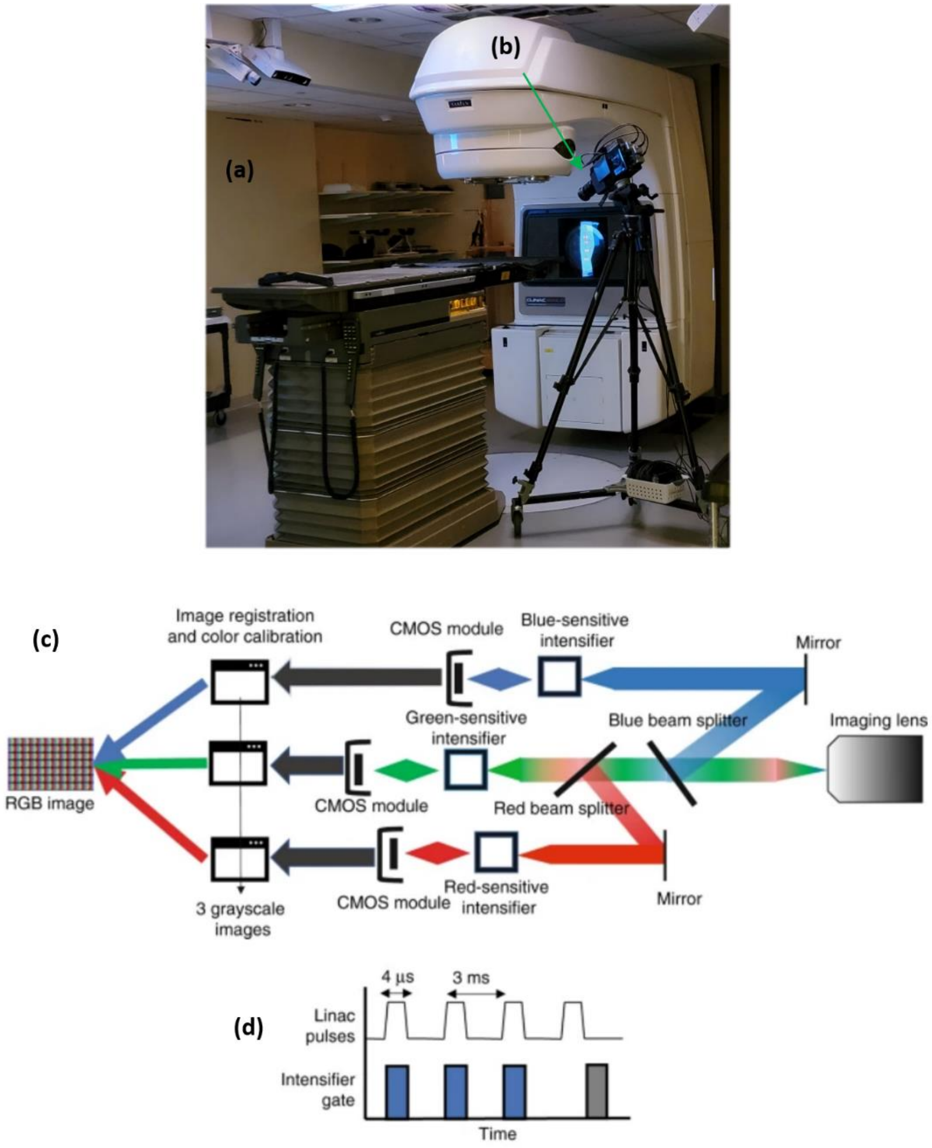

Figure 5 (a) Acquisition setup, using C-Dose software for on-site image processing, LINAC for beam delivery and (b) three-channel time-gated intensified cameras for RGB Cherenkov capture (arrow). Reused with permission from Alexander et al. LSA 2021, (c) shows a detailed schematic

of custom camera[10]. (d) is a representation of necessary time gating and synchronization to LINAC pulses[10].

1.5 Camera and Cherenkov Spectra

The absorption spectra of melanin[10,12], hemoglobin, and oxyhemoglobin [3] were graphed with the spectrum of Cherenkov light as a means of comparison to show their relative wavelength magnitudes. This is shown in Fig 6 (a) along with the Cherenkov emission spectrum, where a thick blue line representing Cherenkov stands as marker to which all else is compared to.

The spectral sensitivity of the imaging system was evaluated. Individual camera detection spectra for the three channels (red, green, blue) were characterized to verify the efficacy and reliability of the filters in tracking and capturing individual channel intensities from experiment tissue samples within expected bands for red, green, and blue wavelengths. The camera filter and intensifier photocathode work together to influence these detectable wavelengths emitted from the TLS, which are from 380 nm to 720 nm with a step of 20 nm, as graphed in Fig 6 (b). This was done using a manual tunable light source (TLS)(Optometrics Manual TLS[18], Optometrics Manufacturing, Ayer, MA) and the tri-color camera, shown in Fig 6 (c). The TLS maximized throughput in the visible region of the spectrum using a filament of a 20W tungsten halogen lamp, with a spectral energy between 360 nm and 2000 nm[17]. The light which output through a small slit was imaged in close range with the tri-color camera using CDose software, with images processed in MATLAB (version X, Natick, MA) and signal intensity graphed as in Fig 6 (b)

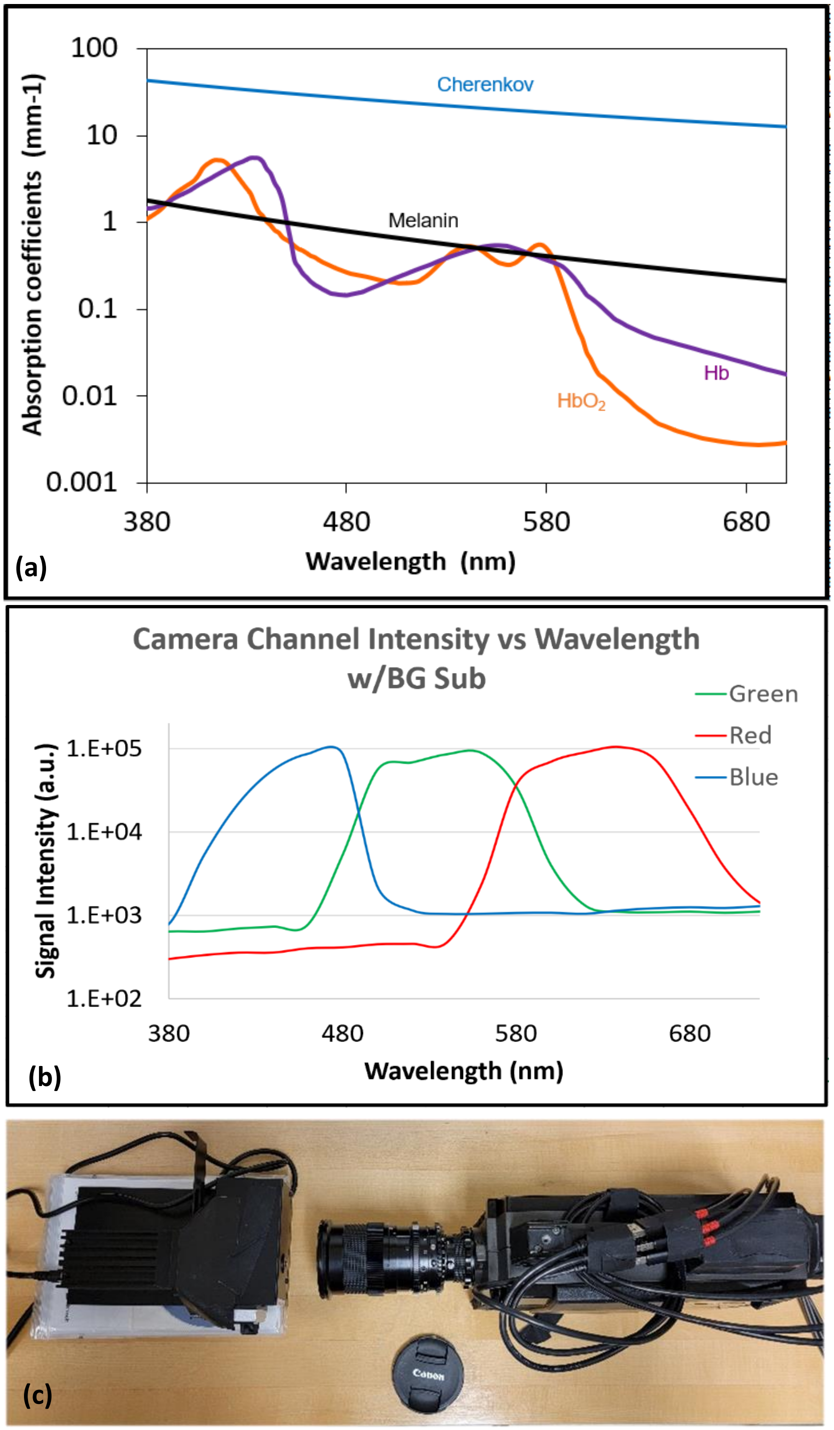

Figure 6 (a) Absorption spectra of melanin, hemoglobin (Hb) and oxyhemoglobin (HbO2) are shown on the same graph as an arbitrarily scaled emission spectra of Cherenkov. In (b) the red, green and blue background subtracted (BG sub) sensitivities of the filtered color Cherenkov camera are shown. (c) A tunable light source was used to calibrate the camera with narrowband monochromatic light, for the data in (b).

1.6 Pigmentation Layer Phantoms

Synthetic epidermal layers of 0.1mm thickness were fabricated based on a previously described method[14] with adjustments for the expected coverage of skin colors specific to this purpose. This was done by dissolving 1 g of gelatin powder (Type A porcine powder, 300 g Bloom, Sigma-Aldrich, St. Louis, MO) and 0.5 g glycerol (49781, ≥98% purity, Sigma-Aldric, , St. Louis, MO) in 10 mL of distilled water. 0.01%–0.1% glutaraldehyde (G5882, ≥99.5% purity, Sigma-Aldrich, St. Louis, MO) was then added, along with varying concentrations of synthetic melanin (0.0018, 0.0038, 0.0076, 0.0114, 0.019, 0.027, 0.045 and 0.072 mg/mL, which came in the form of small crystals and was manually crushed to a fine powder. The resulting solution was heated evenly with a 1200-W microwave (Kenmore Elite #405.74229310, Kenmoore, NY, NY) for 5 s to allow for an even mixture. The solution was poured onto plastic weigh boat with a 5.80 cm base diameter and then placed into a vacuum chamber to remove air bubbles and distortions. These were then dried for 48 h at 21 °C in a fume hood to acquire thin, permanent pliable layers.

Gelatin powder (mg)	Glycerol (mg)	Distilled Water (mL)	Melanin (mg)	Melanin Concentration (mg/mL)
0.5	5.0	10.0	0.5	0.0018
1.0	5.0	10.0	1.0	0.0038
1.5	5.0	10.0	3.0	0.0114
2.0	5.0	10.0	5.0	0.0189
2.5	5.0	10.0	7.2	0.0273
3.0	5.0	10.0	12.0	0.0452
3.5	5.0	10.0	19.0	0.0719

Table 1 For a given sample with a radius of 2.9 cm and volume of 264.2 mm^3, individual solution components of gelatin powder (mg), glycerol (mg), distilled water (mL) and synthetic melanin (mg) are shown, with biologically representative melanin concentration on the far right. Glutaraldehyde was added as a single drop as an antimicrobial per sample.

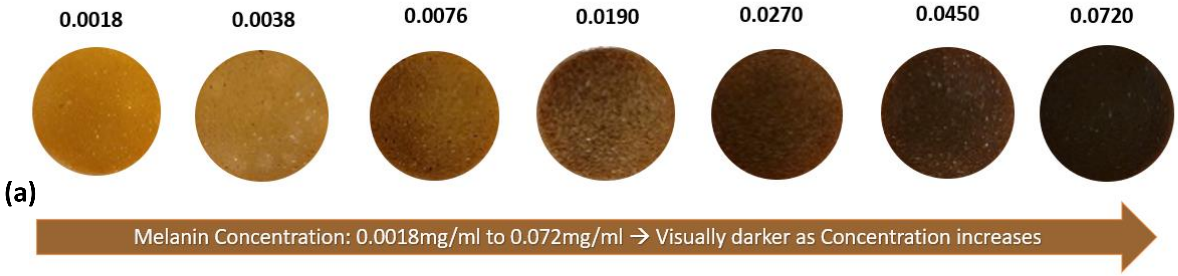
0.0018
0.0038
0.0076
0.0190
0.0270
0.0450
0.0720
(a)
Melanin Concentration: 0.0018mg/ml to 0.072mg/ml → Visually darker as Concentration increases

Figure 7 (a) True color view of epidermal layers created in their respective dishes with increasing melanin concentration from left to right. A vacuum chamber was used to degas the samples when curing, to remove trapped air bubbles that can cause distortion.

1.7 Blood Phantoms

Blood solutions of volume 100ml were created (Fig 8) using bovine whole blood (Lampire biological laboratories, whole blood in Na-EDTA, Donor, Catalog #7200809 500ml), phosphate buffered saline solution (1x, Cytiva 0.0067M PO_4, Lot NO. 02478 500ml), and intralipid (20% emulsion phospholipid stabilized soybean oil, Lot# MKCF1870 100ml). The solution was mixed with concentrations of 1% intralipid (5ml since 20% emulsion), 0.5%, 1.0%, 1.5%, 2.0%, 2.5%, 3.0%, 3.5% bovine whole blood and remaining volume being phosphate buffered solution. A 0.5% solution for example would be 5ml intralipid, 0.5ml bovine whole blood and 94.5ml phosphate buffered solution. The solutions were crafted and poured into petri dishes ((Fig 6 (b)) (Corning 100mmx20mm style cell culture dish, Ref 430167) which had been blacked out (Krylon fusion all-in-one matte black paint and primer spray) an hour before LINAC experiments were conducted.

Whole Bovine Blood Volume (ml)	Intralipid Volume (ml)	Phosphate Buffer Solution Volume (ml)	Blood Concentration (%)
0.5	5	94.5	0.5
1.0	5	94	1.0
1.5	5	93.5	1.5
2.0	5	93	2.0
2.5	5	92.5	2.5
3.0	5	92	3.0
3.5	5	91.5	3.5

Table 2 For a total solution volume of 100ml, individual solution components of whole bovine blood volume (ml), Intralipid volume (ml) and phosphate buffer

solution volume (ml) are shown, with total solution concentration given as a percentage on the far right.

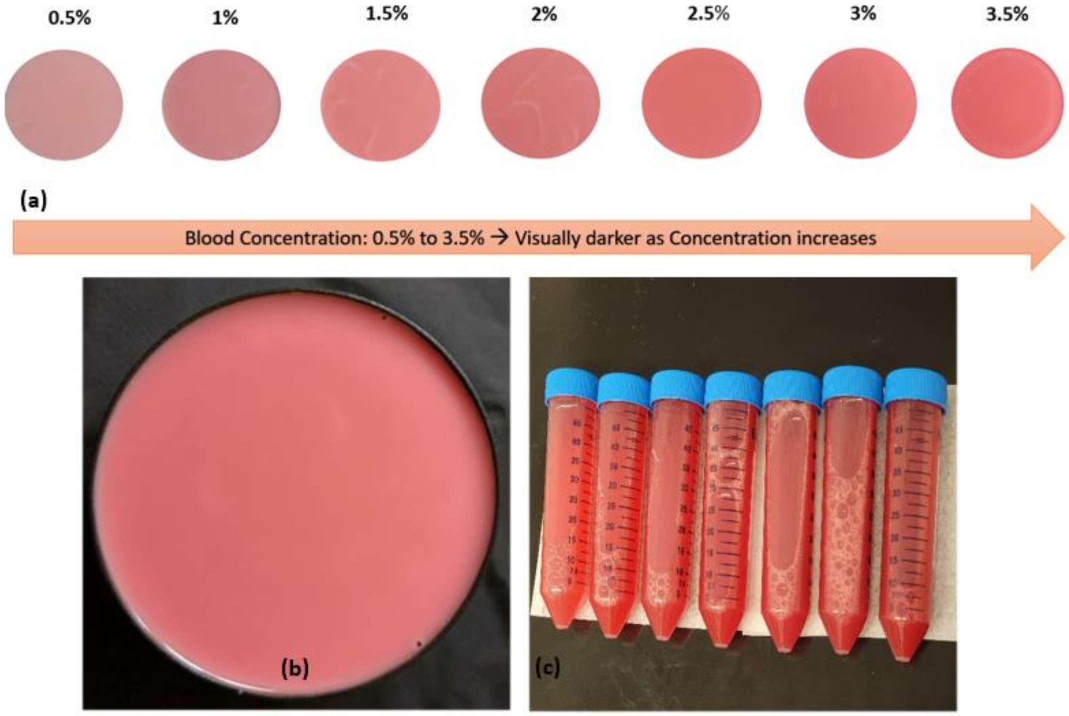
0.5%
1%
1.5%
2%
2.5%
3%
3.5%
(a)
Blood Concentration: 0.5% to 3.5% → Visually darker as Concentration increases
(b)
(c)

Figure 8 (a) True color images of solutions created in their respective dishes with increasing blood concentration from left to right (b) example of single solution dish in blacked-out petri container (c) prepared bovine whole blood and phosphate-buffered saline, that were later combined with the intralipid.

Results

2.1 Layer Validation

The impact of melanin concentration on the optical properties (absorption coefficient, μ_a, and reduced scattering coefficient, μ_s’) of the epidermal layers was quantified using a validated, reflectance geometry spatial frequency domain imaging system and software (Reflect RS, Modulim, Irvine, CA). SFDI separates the effects of scattering and absorption and can be used to estimate the concentrations of chromophores in the tissue. The technique works by shining different patterns light on the tissue, recording a video of the remitted light, and processing the movie acquired[24,25]. Optical properties were quantified at each of 8 wavelengths (471, 526, 591, 621, 659, 691, 731, and 851 nm) using 5 spatial projection frequencies (0.00, 0.05, 0.10, 0.15, and 0.20 mm−1). The epidermal layers were placed on top of 2-cm thick bulk tissue phantom made of silicone with flesh-colored pigment during this procedure. Results are summarized in Fig 9 and Fig 10. Optical property measurements also confirmed that the layers exhibited optical properties relevant to an array of human skin colors.

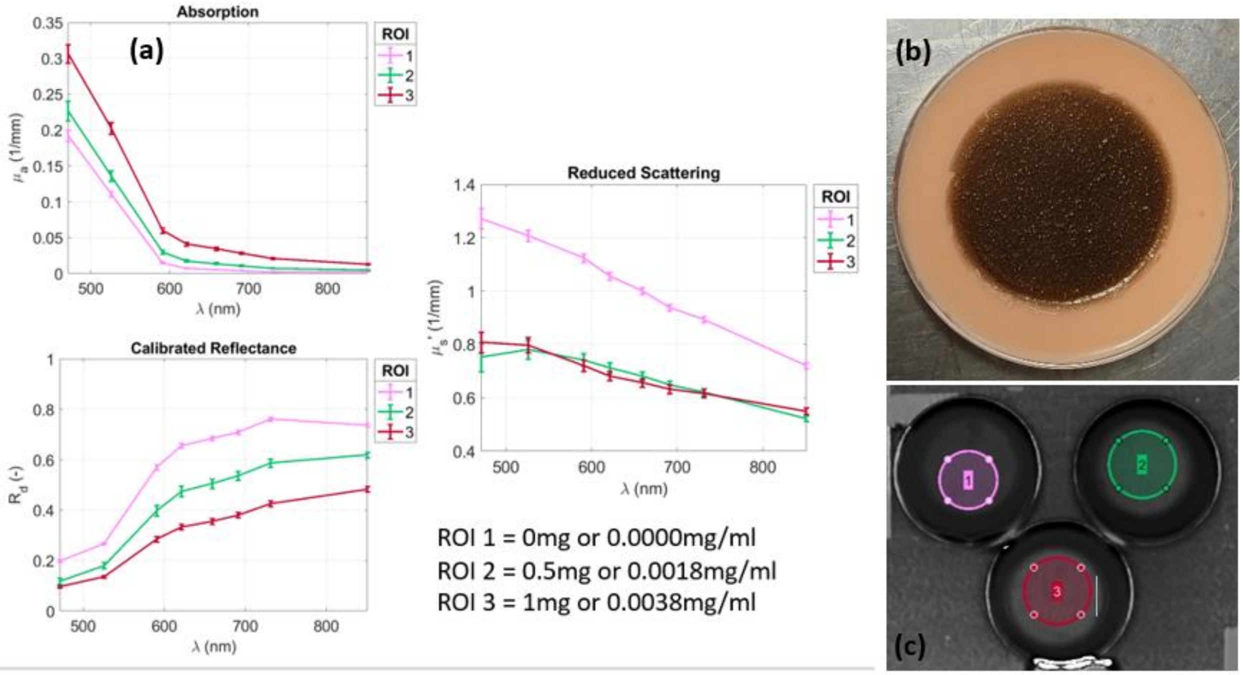

Figure 9 (a) SFDI processed results for absorption coefficient, μ_a, reduced scattering coefficient, μ_s', and calibrated reflectance R_d, shown here with 3 synthetic skin layers. These will be compiled in Fig 4 (b) Synthetic skin on silicone phantom prepped for scanning using SFDI (c) Also shown is a POV from SFDI with the pink ROI (1) being 0mg/ml, green ROI (2) being 0.0018mg/ml and the red ROI (3) being 0.0038mg/ml.

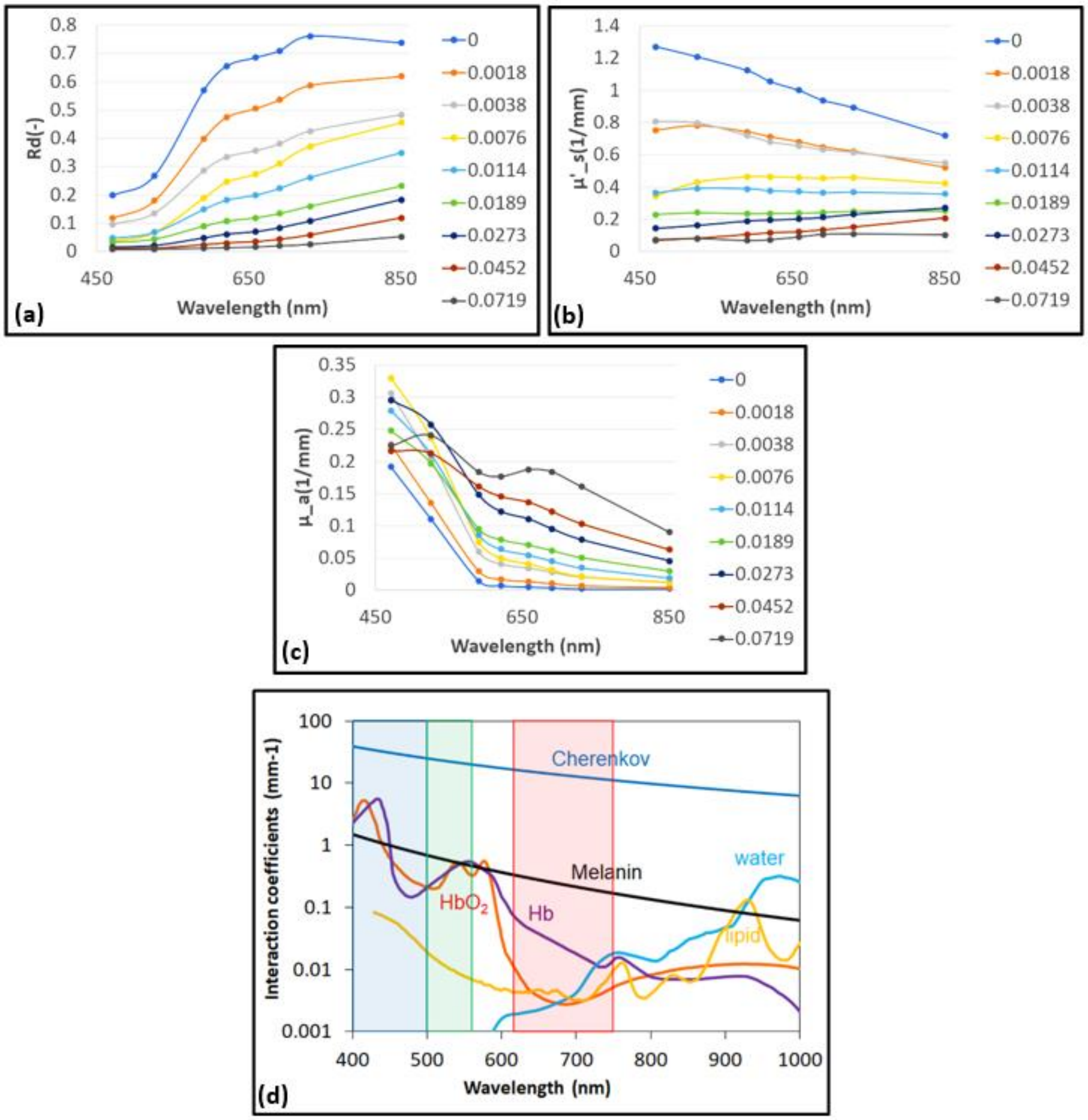

Figure 10 (a) Calibrated reflectance average (b) Reduced scattering averages (c) Absorption averages, all with concentration values in units of mg/ml (d) Known haemoglobin and melanin absorbances in relation to wavelength spectrum[6,7,8] that further help validate the efficacy and viability of the layers created.

2.2 Cherenkov images

With LINAC irradiation and image acquisition completed, the data were processed to output Cherenkov images for melanin and blood concentration variations as shown in Fig 11. This was compared and compiled with respective white light images to showcase the attenuation of image intensity in the region of interest.

For melanin, as shown in Fig 11 a, the attenuation in Cherenkov emission from the tissue phantoms—seen in as a dark circular shape in the middle of the bulk silicon phantom—shows even and tight decrease as concentration increases, without one particular color vastly differentiating itself in intensity from another. This is in line with melanin spectral properties showing no anomalous preference to a particular wavelength (unlike blood) and decreasing in absorption units with increasing wavelength as noted in known literature[6,7,8] as well as being confirmed via SFDI optical property validation in Fig 9 and Fig 10. For purposes of visualization, the blue channel needed individualized windowing and leveling and thus a separate color bar is show.

With regard to blood, as shown in Fig 11 b, the attenuation in Cherenkov emission from the blood samples show a varying and individualized decrease as concentration increases, with one particular color, red, greatly distinguishing itself in mean signal intensity from green and blue, a phenomenon which is further highlighted in Fig 12 where a color Cherenkov camera was used in a previous Pogue Lab study[3] to capture Cherenkov emission from blood. This is in line with blood spectral properties showing preference to a particular wavelength—haemoglobin bound to oxygen absorbs blue-green light, reflecting red-orange light[16]—appearing red and varying non-uniformly in absorption units with increasing wavelength as noted in known literature[6,7,8]. For purposes of visualization, as before, the blue channel needed individualized windowing and leveling and thus a separate color bar is shown.

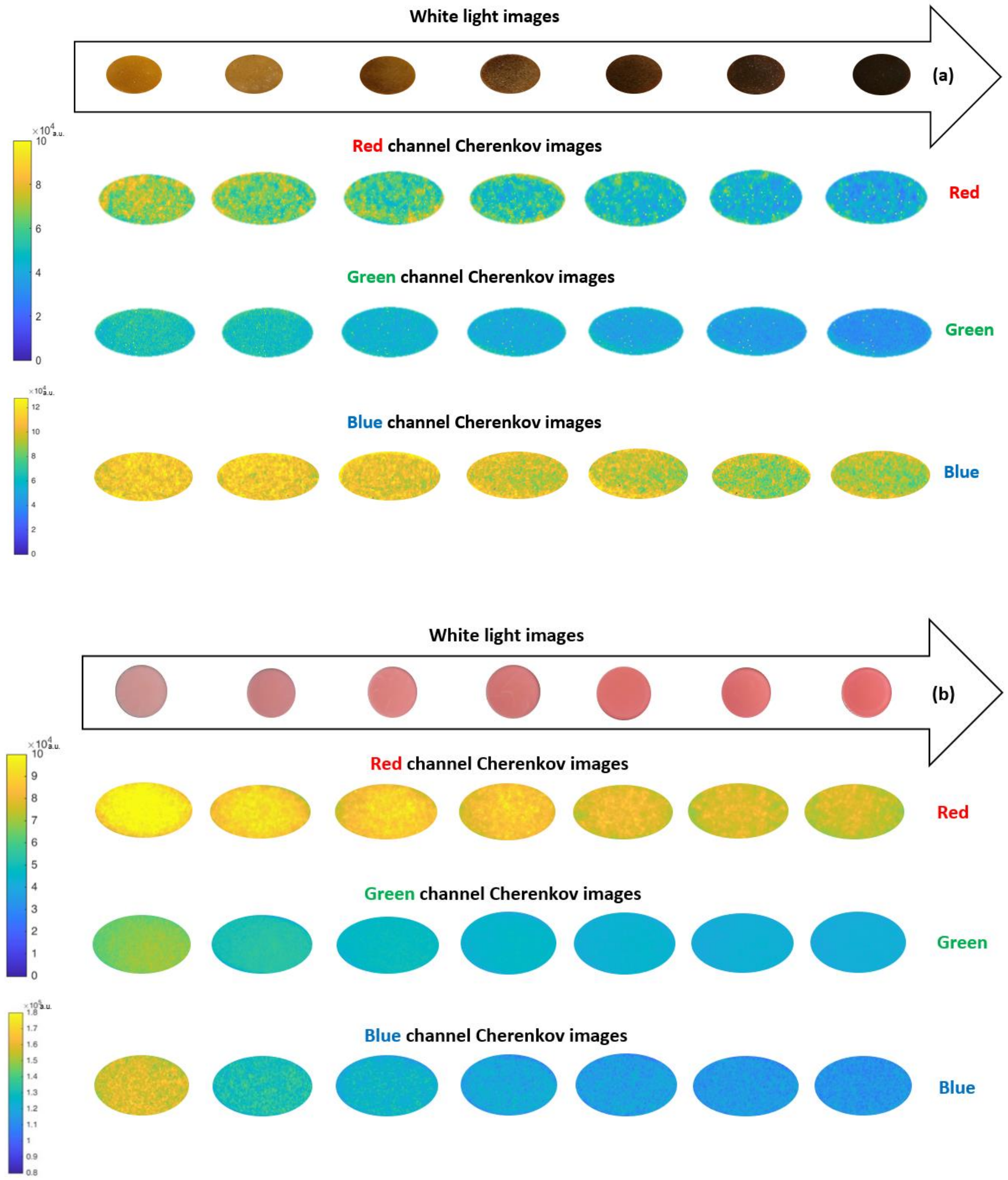

Figure 11 (a) White light images of melanin tissue phantoms, side by side, going from low absorption to high absorption. Images of the Cherenkov emission images in each of red, green and blue wavelength bands are shown. Concentrations of melanin are 0.0018, 0.0038, 0.0076, 0.0114, 0.019, 0.027, 0.045 and 0.072 mg/ml respectively, left to right. (b) White light images of blood phantoms, side by side, going from low absorption to high absorption. Images of

the Cherenkov emission images in each of red, green and blue wavelength bands are shown. Concentrations of blood are 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5 % respectively, left to right.

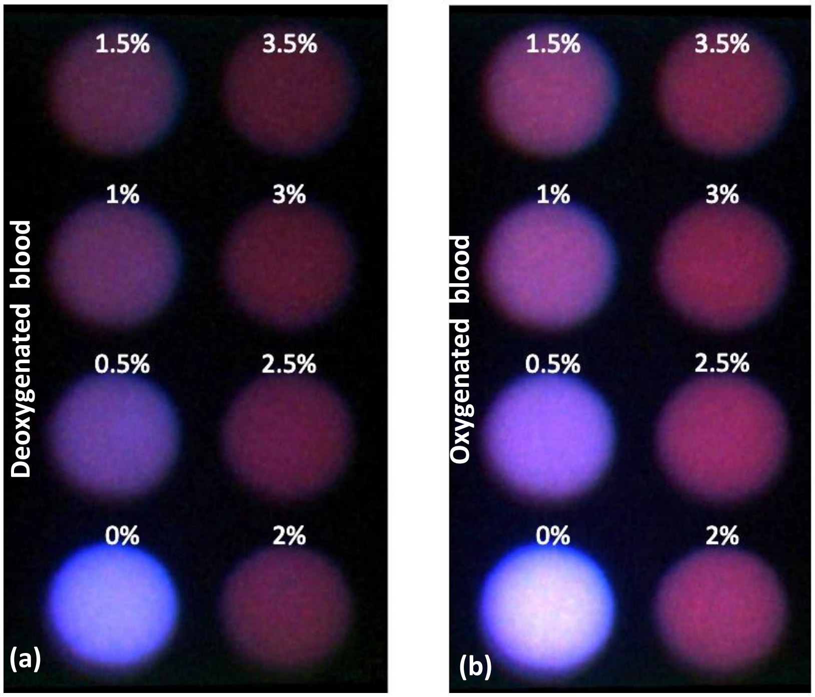

Figure 12 (a) Blood-based phantoms imaged with the color Cherenkov camera showing greater attenuation of Cherenkov light as concentration increases. On the left (a) is deoxygenated blood, and on the right (b) is oxygenated blood[3]. (Alexander et al. Light: Science & Applications (2021)10:226))

2.3 Differential signal response

Cherenkov images from each of the individual channels—gathered using CDose software and processed in MATLAB—resulted in the output of signal intensities across various concentrations in melanin and whole bovine blood, as shown in Fig 13. Data gathered show an attenuation response for both melanin and blood. For melanin, as per visual aid of Fig 11 (a) and (b), the quantitative response graphed in Fig 13 (a) shows all channels following lower emission intensity with increasing pigment concentration. Comparing this response with the reference chart in Fig 10 (a), the trend seen in our results follows what is expected for Cherenkov emission and melanin concentration. Conversely, the whole bovine blood quantification (Fig 13 (b)), with visual aid from Fig 11 a and b, shows a differential response from the three channels. The red channel Cherenkov signal intensity attenuates to a much lesser extent, while the green and blue channels follow a tight diminution together to levels lower than red as a whole. The increased absorption of blue and green by hemoglobin[16] results in an increased signal in the red channel. Furthermore, Fig 6 (a) corroborates this observation; oxygenated hemoglobin exhibits two peaks where blue and green light wavelengths are represented. This phenomenon of different amounts of exiting light in the red, green, and blue bands in color Cherenkov imaging provides quantitative and qualitative confirmation of detection of signals that varied differently with changes in blood or melanin content. Further color Cherenkov imaging of skin pigmentation phantoms is seen in Fig 15. 3Gy of dose was given via 6MV photon and 6MeV electron beams to the seven phantoms to show the limits of visual analysis of Cherenkov color imaging. With both beam energies, pigmentation past 0.0076mg/ml is seen to be extremely difficult to distinguish from the background—effectively no observable Cherenkov emission. This observation also further illustrates the need for correcting Cherenkov signal attenuation due to biological tissue factors so that the emission can be seen as visually independent from patient-specific attenuating influences.

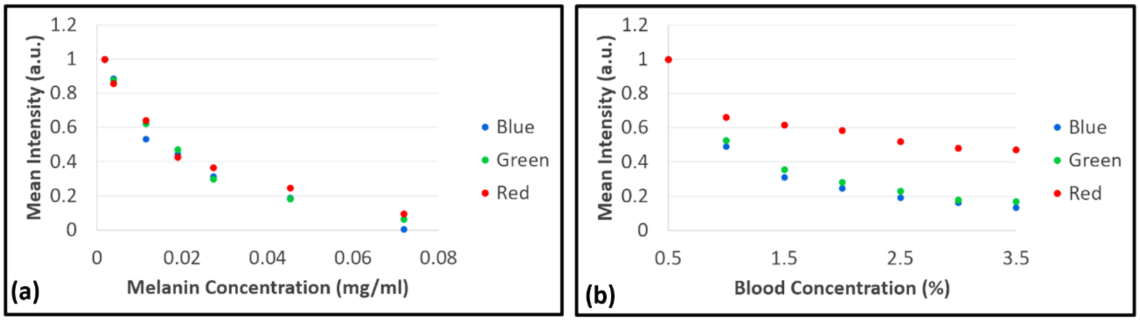

Figure 13: (a) Melanin concentration in mg/ml and (b) blood concentration as a percentage, both measured against mean Cherenkov signal intensity across individual red, green and blue channels.

2.4 Differential visual response

In addition to a variance in Cherenkov signal response as seen in Fig 11, this shows that pigmentation increase is visually different from the hemoglobin increases by the colors seen. The phenomenon is depicted in Fig 14 below. It was found to be interesting because it is optically representative of real Cherenkov emission differential and acts as another visual verification of quantitative data.

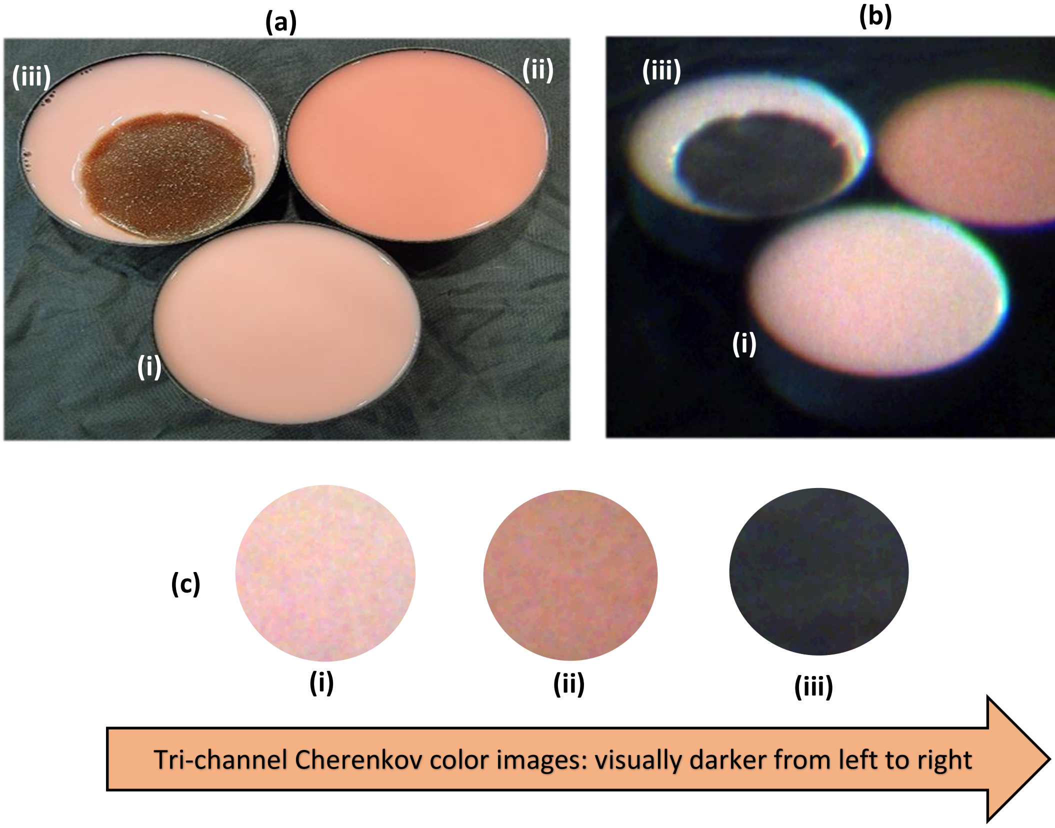
(a)
(iii)
(ii)
(i)
(b)
(iii)
(ii)
(i)
(c)
(i)
(ii)
(iii)
Tri-channel Cherenkov color images: visually darker from left to right

Figure 14: Visual display of 3 phantoms chosen to show variation in blood (i) vs (ii) and variation with overlying pigment layer (i) vs (iii). The standard color photograph of the tissue phantoms is shown in (a) with ambient room lighting, and the Cherenkov color image is shown in (b) taken in a darkened room. The true color values extracted from the RGB values are shown in (c). Here (i) is 1% intralipid & 1% blood, (ii) is 1% intralipid & 3% blood, and (iii) is a 1% intralipid & 1% blood phantom with a 0.0270 mg//mL pigmentation layer on top.

Further color Cherenkov imaging of skin pigmentation phantoms is seen in Fig 10. 300MU of dose was given via 6MeV and 6MeV beams to the seven phantoms to show the limits of visual analysis of Cherenkov color imaging. With both beam energies, pigmentation past 0.0076mg/ml is seen to be extremely difficult to distinguish from the background—which behaves as an indicator of zero Cherenkov emission—and is thus indicative of an interpretive threshold for visualizing dose via Cherenkov color imaging. This answers a central question of the study i.e. how much attenuation occurs from skin color, when doing Cherenkov imaging, and what the relevant limits are. The finding also further showcases the need for correcting Cherenkov signal attenuation as a result of biological tissue factors so that the emission can be seen as visually independent from that of attenuating influences.

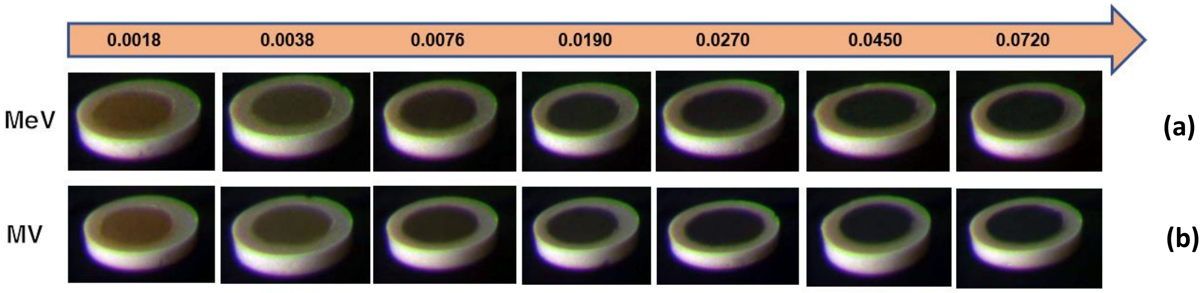

Figure 15 Visual display of seven phantoms chosen to show variation in melanin content. The standard color photograph of the tissue phantoms is shown in Fig 4 with ambient room lighting, and the Cherenkov color image is shown here in (a) and (b) taken in a darkened room. (a) Represents color Cherenkov images taken with MeV beam, and (b) with MV beam. The values

depicted above the images are in units of mg/ml to represent biological melanin concentration in each pigmentation phantom.

Discussion

This study examined the expansion from prior work on Cherenkov imaging[3,19,20] but with the use of a three-channel RGB Cherenkov emission camera, to illustrate how imaging in color may provide a visual separation of tissue attenuation effects. The hypothesis driving this work was that differential RGB color Cherenkov emission levels would result from variations in the most dominant biological tissue absorption features, such as blood concentration within tissue and melanin concentration in the skin. Experimental measurements shown in Fig 11, 14, and 15 visually demonstrate that this is true. Furthermore, these changes in blood or melanin concentrations result in distinct visual changes in the RGB outputs values of the Cherenkov color emission. These findings support the idea that color or spectral imaging of Cherenkov might provide an experimental methodology for separation of biological attenuation of the intensity from the physical generation of Cherenkov with dose deposition. The goal would ideally be to use the Cherenkov intensity as an indicator of the dose delivered in the tissue, independent of the blood volume within it or the skin color, by using color correction.

While spectroscopic imaging of Cherenkov would likely provide a better quantitative measure of the color changes, the utility and convention of imaging in three RGB channels is ubiquitous today. The images provide a visual cue to the users of Cherenkov imaging about the biological origins of what is being seen. It is possible that corrected Cherenkov images could be displayed, side by side with information about the melanin or blood levels. Oxygenation studies have been examined in a previous paper[3], but in most normal tissues blood oxygenation is nearly maximal in the surface tissue layers, so here the focus was maintained on oxygenated blood.

Perhaps one of the most central questions from this work is: how much attenuation occurs from skin color when acquiring Cherenkov images? The attenuation depends upon the concentration of melanin in the skin, and examining the data in Fig 14(a) and the images in Fig 15(a) and (b) provide the answer to this. The skin colors imaged in the phantom correspond to the range of human skin colors expected, with the darkest having an extremely high melanin level, and in the data shows a 90% to 100% reduction in Cherenkov emission in all three color channels, with blue and green being the least emitted. It may not be possible to perform Cherenkov color imaging in people with the darkest skin tone because of the extreme emission attenuation, visually shown in Fig 15. It should be noted that despite this, imaging in the next lower melanin level was possible with attenuation just at the 75-80% level. An increase in camera or image gain might be employed to overcome this.

A major advantage to the setup in this work is the capacity to quickly and efficiently gather images showing contrast between changing biological factor concentration (blood and melanin here) with a portable setup. This can be performed in any clinic following a standard procedure. Despite requiring background light suppression via LINAC pulse time gating (Figure 5), this can be used to great effect via dynamic real-time acquisition visualization with a wireless setup, presenting a natural and realistic interaction of Cherenkov emission response. A main disadvantage to this method is the proprietary nature of the complex camera, which as a whole is not mass-produced and is the only one known to exist. Additionally, the image processing and analysis can be a lot more demanding as opposed to conventional monochrome Cherenkov imaging[3] due to the presence of three separate channels to consistently calibrate, frame stack and process.

Future work would include automation of image processing as well as further investigation of correction factors for variation in blood and melanin concentration changes in patients, as well as correcting for lighting conditions and camera setup to ensure the least possible signal-to-noise ratio in fully color resolved Cherenkov image output. The work could also be expanded to study the variances in RGB color Cherenkov imaging colors, comparing entrance and exit beams as well as MV and MeV energies. This would be interesting since the apparent color may likely change because the buildup curve is different for these two types of radiation, and hence spectral attenuations would likely be different. Patient studies could be performed on a large-scale clinical level, with a focus on the impact of how differential Cherenkov signal response contributes to Cherenkov dose readouts and guide better, more accurate patient surface dosimetry. This would require the manufacturing of an equal or better camera setup if it is to be done in multiple locations, and involve a careful step-by-step acquisition procedure with consistency in room and camera setup. Such future work would be done with a greater diversity of patients as opposed to previous studies done with mainly lower melanin pigmentation as well as varying levels of blood concentration or oxygenation in accordance with any present afflictions or ailments resulting in perturbation from a basal state. Further work could examine the context of other biological factors such as lipids and localized blood in vasculature. A whole skin model could potentially be used to further this goal, with tunable sectors to monitor the effects of changing variables, allowing for a major expansion to this work on blood and melanin alone.

Conclusion

This study quantitatively and visually showed how the imaging of biological tissue using RGB color Cherenkov imaging could provide independent information on intensity variations from pigmented skin or hemoglobin level variations in the material, with this information being independent of the total dose. The differential signal response seen in blood versus melanin show that it would be possible to differentiate the attenuation effects of the two spectrally. There is a possibility to correct Cherenkov intensity for the attenuation of one of these biological factors. However, an important observation is also that in the very darkest color phantoms, it seems as though there may be insufficient blue Cherenkov light emitted to gain a reliable signal without large improvements to the light capture approach. Still, red wavelengths were sufficient in all skin color phantoms, albeit with a near 90% reduction in the darkest skin tones. Further focus on spectral distortion corrections for Cherenkov intensity changes might be used in quantitative patient dosimetric imaging from Cherenkov intensity.

Disclosures

Brian Pogue, Petr Bruza, Savannah Decker disclose financial involvement in DoseOptics LLC, a company manufacturing Cherenkov imaging cameras for radiation therapy.

Acknowledgments

This work has been supported by NIH research grant R01 EB023909 and shared service resources from the Dartmouth Cancer Center provided by NIH grant P30 CA023108.

Code, Data, and Materials Availability

The data produced in this report are available to interested readers upon reasonable request to the corresponding author. No original software was produced in this effort.

References:

1. Thwaites, D. I., Mijnheer, B. J., & Mills, J. A. (2005). Chapter 12 quality assurance of external beam radiotherapy

2. Yogo, K., Matsushita, A., Tatsuno, Y. *et al.* Imaging Cherenkov emission for quality assurance of high-dose-rate brachytherapy. *Sci Rep* **10,** 3572 (2020). https://doi.org/10.1038/s41598-020-60519-z

3. Alexander, D.A., Nomezine, A., Jarvis, L.A. *et al.* Color Cherenkov imaging of clinical radiation therapy. *Light Sci Appl* **10,** 226 (2021). https://doi.org/10.1038/s41377-021-00660-0

4. Glaser, A. K. et al. Optical dosimetry of radiotherapy beams using Cherenkov radiation: the relationship between light emission and dose. *Phys. Med. Biol.* **59**, 3789–3811 (2014).

5. radiation therapy in real time. *Int. J. Radiat. Oncol. Biol. Phys.* **89**, 615–622 (2014).

6. Igarashi, Takanori & Nishino, Ko & Nayar, Shree. (2007). The Appearance of Human Skin. Foundations and Trends in Computer Graphics and Vision. 3. 1-95. 10.1561/0600000013.

7. Mhawes, Ahmed & Hanaa, A & Hasan, & Mona, Jammel. (2017). The use of Nd:YAG laser in treatment of superficial vascular and pigmentary lesions استخدام ليزر Nd:YAG 15. الوحمات الوعائية السطحية والصبغية لمعالجة.

8. Jennifer Riesz, Joel Gilmore, Paul Meredith, Quantitative Scattering of Melanin Solutions, Biophysical Journal, Volume 90, Issue 11, 2006, Pages 4137-4144, ISSN 0006-3495, https://doi.org/10.1529/biophysj.105.075713.

9. Wagner, Robert & Byrum, K. & Sanchez, Mayly & Vaniachine, Alexandre & Siegmund, Oswald & Otte, Nepomuk & Ramberg, Erik & Hall, Jeter & Buckley, James. (2009). The Next Generation of Photo-Detectors for Particle Astrophysics. 10.2172/956926.

10. Melanosome Absorption Coefficient. (2022). Omlc.org. https://omlc.org/spectra/melanin/mua.html

11. SL Jacques, DJ McAuliffe. The melanosome: threshold temperature for explosive vaporization and internal absorption coefficient during pulsed laser irradiation. Photochem. Photobiol. 53:769-775, 1991

12. SL Jacques, RD Glickman, JA Schwartz. Internal absorption coefficient and threshold for pulsed laser disruption of melanosomes isolated from retinal pigment epithelium. SPIE Proceedings 2681:468-477, 1996.

13. Ashraf, M. R. et al. Technical note: time-gating to medical linear accelerator pulses: stray radiation detector. *Med. Phys.* **46**, 1044–1048 (2019).

14. Chen, A. I., Balter, M. L., Chen, M. I., Gross, D., Alam, S. K., Maguire, T. J., & Yarmush, M. L. (2016). Multilayered tissue mimicking skin and vessel

phantoms with tunable mechanical, optical, and acoustic properties. *Medical physics*, *43*(6), 3117–3131. https://doi.org/10.1118/1.4951729

15. "Spatial Frequency Domain Imaging." Harvard.edu, 2022, scholar.harvard.edu/ndurr/pages/sfdi.

16. Bremmer, R. H., de Bruin, K. G., van Gemert, M. J. C., van Leeuwen, T. G., & Aalders, M. C. G. (2012). Forensic quest for age determination of bloodstains. Forensic Science International, 216(1-3), 1–11. https://doi.org/10.1016/j.forsciint.2011.07.027

17. Optometrics Tunable Light Sources | Optometrics Knowledge Base. (2022, March 9). Retrieved June 9, 2022, from Optometrics website: https://www.optometrics.com/knowledge-base/tunable-light-sources/

18. Manual Tunable Light Sources Archives - Optometrics. (2022). Retrieved June 9, 2022, from Optometrics website: https://www.optometrics.com/product-category/tunable-light-sources/manual-tunable-light-sources/

19. Zhang, R. et al. Cherenkoscopy based patient positioning validation and movement tracking during post-lumpectomy whole breast radiation therapy. Phys. Med. Biol. 60, L1–L14 (2015)

20. Hachadorian, R. L. et al. Imaging radiation dose in breast radiotherapy by X-ray CT calibration of Cherenkov light. Nat. Commun. 11, 2298 (2020).

21. Fermilab. (2018). How does Cerenkov radiation work? [YouTube Video]. Retrieved from https://www.youtube.com/watch?v=Yjx0BSXa0Ks

22. Water Cherenkov detector – ESSnuSB. (2020). Retrieved June 23, 2022, from Essnusb.eu website: https://essnusb.eu/glossary/water-cherenkov-detector/

23. Etd | Characterization of the Standard Imaging Exradin W2 scintillator detector for small field dosimetry | ID: 0r967453b | Digital Collections. (2021). Retrieved June 23, 2022, from Ohsu.edu website: https://scholararchive.ohsu.edu/concern/etds/0r967453b

24. Dr. Rüdiger Paschotta. (2022, June 3). Refractive Index. Retrieved June 23, 2022, from Rp-photonics.com website: https://www.rp-photonics.com/refractive_index.html

25. J. C. Owens, "Optical refractive index of air: dependence on pressure, temperature and composition", Appl. Opt. 6 (1), 51 (1967), doi:10.1364/AO.6.000051

26. Zhang R, Glaser AK, Andreozzi J, Jiang S, Jarvis LA, Gladstone DJ, Pogue BW. Beam and tissue factors affecting Cherenkov image intensity for quantitative entrance and exit dosimetry on human tissue. J Biophotonics. 2017 May;10(5):645-656. doi: 10.1002/jbio.201500344. Epub 2016 Aug 10. PMID: 27507213; PMCID: PMC5529250.

27. Hachadorian R, Bruza P, Jermyn M, Mazhar A, Cuccia D, Jarvis L, Gladstone D, Pogue B. Correcting Cherenkov light attenuation in tissue using spatial frequency domain imaging for quantitative surface dosimetry during whole breast radiation therapy. J Biomed Opt. 2018 Nov;24(7):1-10. doi: 10.1117/1.JBO.24.7.071609. PMID: 30415511; PMCID: PMC6228320.

28. Finding Your Fitzpatrick Skin Type - Injectables. (2019, May 3). Retrieved August 11, 2022, from Injectables website: https://injectables.com.au/finding-fitzpatrick-skin-type/

Appendix: Color Correction Method

Color correction was done on output image stacks using Photopea’s auto color correct (https://www.photopea.com/) .

The ROI is selected and under the Image tab, auto color was selected.

Appendix: Code

```
% iCMOS color calibration %Petr Bruza/Daniel Alexander

%% Specify Working Folders

WorkingFolder=''; % Specify working folder here
addpath(genpath(WorkingFolder));
addpath(genpath('')); % Load any external Matlab functions you need
(read_dovi, for example)
cd(WorkingFolder)

data_path = ''; % path to your data
addpath(genpath(data_path));

%% Specify Tiff Header Parameters

tagstruct.Photometric     = Tiff.Photometric.MinIsBlack;
tagstruct.BitsPerSample   = 32;
tagstruct.SamplesPerPixel = 1;
tagstruct.PlanarConfiguration = Tiff.PlanarConfiguration.Chunky;
tagstruct.Software        = 'MATLAB';
tagstruct.SampleFormat    = 3; %uInt1 signed int2 IEEEFP 3
tagstruct.Compression = 1;
%tagstruct.ResolutionUnit = 3; %2 inch 3 cm
%tagstruct.XResolution = 0.05;
%tagstruct.YResolution = 0.05;


%% Read Raw Color Checkerboard Images

current_folder = fullfile(''); % specific data folder with checkerboard images
filetext = fileread(fullfile(data_path, current_folder,'settings.ini'));

% match serial numbers ----------
n0 = str2double(regexp(filetext, '(?<=Camera0_Serial_Number=[^0-9]*)[0-
9]*\.?[0-9]+', 'match'));
n1 = str2double(regexp(filetext, '(?<=Camera1_Serial_Number=[^0-9]*)[0-
9]*\.?[0-9]+', 'match'));
```

```
n2 = str2double(regexp(filetext, '(?<=Camera2_Serial_Number=[^0-9]*)[0-
9]*\.?[0-9]+', 'match'));
colorcalib_order = [n0, n1, n2];

% read images ----------

for cam = 1:3
    tic
    data_colorcalib =
double(read_dovi(fullfile(data_path,current_folder,['meas_s0_cam',num2str(cam-
1),'.dovi'])));
    toc
    colorcalib_mean(:,:,cam) = rot90(mean(data_colorcalib,3),2);
    clear data_calib
end

%% Prep Variables for Co-registration

% filter if needed ---------------------
% inreg_img = medfilt3(colorcalib_mean,[3,3,1]);

% Prep Vars ---------------------
inreg_img = colorcalib_mean;
reg_img = zeros(size(inreg_img));
reg_img(:,:,1) = inreg_img(:,:,1); % don't need to warp first image
color_calib_tforms=struct(); % prep tform struct

%% Register with Control Point registration tool, first green vs red

cpselect(2*mat2gray(inreg_img(:,:,1)),2*mat2gray(inreg_img(:,:,2)),'WAIT',
false);

%% Generate TForm for green vs red and check

% Warp --------------------------
movingPoints1 = movingPoints; fixedPoints1 = fixedPoints;
%clear movingPoints fixedPoints
color_calib_tforms.tform_1 = fitgeotrans(fixedPoints1, movingPoints1,
'polynomial',2);
reg_img(:,:,2) = imwarp(inreg_img(:,:,2), color_calib_tforms.tform_1,
'OutputView', imref2d(size(inreg_img(:,:,1))));

% Check Reg ---------------------------
figure(1)
imshowpair(mat2gray(reg_img(:,:,1)), mat2gray(reg_img(:,:,2)));

%% Now Channel 3 vs 1+2

cpselect(mat2gray(reg_img(:,:,2)+reg_img(:,:,1)),
3*mat2gray(2*inreg_img(:,:,3)), 'WAIT', false)

%% Generate second TForm

% Warp --------------------------
movingPoints2 = movingPoints; fixedPoints2 = fixedPoints;
```

```
clear movingPoints fixedPoints
color_calib_tforms.tform_2 = fitgeotrans(fixedPoints2, movingPoints2,
'polynomial',2);
reg_img(:,:,3) = imwarp(inreg_img(:,:,3), color_calib_tforms.tform_2,
'OutputView', imref2d(size(inreg_img(:,:,1))));

% Check Reg ---------------------------
figure(2)
imshowpair(mat2gray(reg_img(:,:,2)+reg_img(:,:,1)),
3*mat2gray(reg_img(:,:,3)));


%% Read in color checkerboard reference image

checkr24 = rot90(imread(fullfile(WorkingFolder, 'checkr24.bmp')),3);

%% Register iCMOS images agaisnt reference BMP image of color checkerboard

cpselect(mat2gray(reg_img(:,:,2)+reg_img(:,:,1)), checkr24, 'WAIT', false)

%% Generate Color BMP tform

movingPoints_checkr = movingPoints; fixedPoints_checkr = fixedPoints;
color_calib_tforms.tform_checkr = fitgeotrans(movingPoints_checkr,
fixedPoints_checkr, 'polynomial',2);

%% Transform iCMOS images to BMP for calibration and check color reg

% Warp --------------------------------
reg_checkr_imgG = imwarp(reg_img(:,:,1), color_calib_tforms.tform_checkr,
'OutputView', imref2d(size(checkr24(:,:,1))));
reg_checkr_imgR = imwarp(reg_img(:,:,2), color_calib_tforms.tform_checkr,
'OutputView', imref2d(size(checkr24(:,:,1))));
reg_checkr_imgB = imwarp(reg_img(:,:,3), color_calib_tforms.tform_checkr,
'OutputView', imref2d(size(checkr24(:,:,1))));
reg_checkr_img(:,:,1) = reg_checkr_imgR;
reg_checkr_img(:,:,2) = reg_checkr_imgG;
reg_checkr_img(:,:,3) = reg_checkr_imgB;

% Check Reg --------------------------
figure(3)
imshowpair(mat2gray(reg_checkr_img(:,:,1)), checkr24(:,:,1));


%% Save tforms

save(fullfile(WorkingFolder, 'color_calib_tforms.mat'), 'color_calib_tforms');

%% If tforms already generated, use this

% load(fullfile(WorkingFolder, 'color_calib_tforms.mat'));
% reg_img = zeros(size(inreg_img));
% reg_img(:,:,1) = inreg_img(:,:,1);
% reg_img(:,:,2) = imwarp(inreg_img(:,:,2), color_calib_tforms.tform_1,
'OutputView', imref2d(size(inreg_img(:,:,1))));
```

```
% reg_img(:,:,3) = imwarp(inreg_img(:,:,3), color_calib_tforms.tform_2,
'OutputView', imref2d(size(inreg_img(:,:,1))));
% checkr24 = rot90(imread(fullfile(WorkingFolder, 'checkr24.bmp')),3);
% reg_checkr_imgG = imwarp(reg_img(:,:,1), color_calib_tforms.tform_checkr,
'OutputView', imref2d(size(checkr24(:,:,1))));
% reg_checkr_imgR = imwarp(reg_img(:,:,2), color_calib_tforms.tform_checkr,
'OutputView', imref2d(size(checkr24(:,:,1))));
% reg_checkr_imgB = imwarp(reg_img(:,:,3), color_calib_tforms.tform_checkr,
'OutputView', imref2d(size(checkr24(:,:,1))));
%
% imwrite(reg_checkr_imgG./max(reg_checkr_imgG(:)), 'reg_checkr_imgG.tif');
% imwrite(reg_checkr_imgR./max(reg_checkr_imgR(:)), 'reg_checkr_imgR.tif');
% imwrite(reg_checkr_imgB./max(reg_checkr_imgB(:)), 'reg_checkr_imgB.tif');
%% Create ROIs for color calibration

checkr_roi_center_x = repmat(round(linspace(45,240,3)),5,1);
checkr_roi_center_y = repmat(round(linspace(40,366,5))',1,3);
clear checkr_roi_center
checkr_roi_center(:,:,1) = checkr_roi_center_x;
checkr_roi_center(:,:,2) = checkr_roi_center_y;

checkr_roi_center = rot90(checkr_roi_center ,3);
checkr_roi_center(:,:,2) = flip(checkr_roi_center(:,:,2) ,2);

%% Extract pairs of RGB data (iCMOS, BMP reference), each composed of three 8-
bit values
clear rgb_pairs
i = 1;
for m = 1:size(checkr_roi_center,1)
    for n = 1:size(checkr_roi_center,2)
        y = checkr_roi_center(m,n,2);
        x = checkr_roi_center(m,n,1);
        rgb_pairs(1,i,1) = mean(mean(checkr24(x-10:x+10,y-10:y+10,1))); %first
layer checkr, second layer meas
        rgb_pairs(2,i,1) = mean(mean(checkr24(x-10:x+10,y-10:y+10,2)));
        rgb_pairs(3,i,1) = mean(mean(checkr24(x-10:x+10,y-10:y+10,3)));
        rgb_pairs(1,i,2) = mean(mean(reg_checkr_imgR(x-10:x+10,y-10:y+10)));
%first layer checkr, second layer meas
        rgb_pairs(2,i,2) = mean(mean(reg_checkr_imgG(x-10:x+10,y-10:y+10)));
        rgb_pairs(3,i,2) = mean(mean(reg_checkr_imgB(x-10:x+10,y-10:y+10)));
        i = i+1;
    end
end

%% Plot iCMOS vs BMP RGB values

figure(4)
for i=1:3
    scatter(squeeze(rgb_pairs(i,:,1)),squeeze(rgb_pairs(i,:,2))); hold on;
end
% savefig(gcf,'calib_points.fig')
%% Fit function to relate RGB values of iCMOS to BMP
polydeg=1;
color_p = zeros(polydeg+1,3); % array for 1st deg poly; for deg n, need matrix
dims (n+1)x3
```

```
color_p(:,1) =
polyfit(squeeze(rgb_pairs(1,:,2)),squeeze(rgb_pairs(1,:,1)),polydeg);
color_p(:,2) =
polyfit(squeeze(rgb_pairs(2,:,2)),squeeze(rgb_pairs(2,:,1)),polydeg);
color_p(:,3) =
polyfit(squeeze(rgb_pairs(3,:,2)),squeeze(rgb_pairs(3,:,1)),polydeg);

%% Check Fit
xax = (0:1:4000)';
yax = zeros(numel(xax), 3);
for i=1:3
    yax(:, i) = color_p(1,i)*xax + color_p(2,i);
end

% check fit -----------------------
figure(5)
colors=linspecer(3);
for i=1:3
    plot(xax,yax(:,i), '-', 'Color', colors(i,:)); hold on;
    scatter(squeeze(rgb_pairs(i,:,2)),squeeze(rgb_pairs(i,:,1)), 'o',
'MarkerEdgeColor', colors(i,:)); hold on;
end
xlim([0 1500])
ylim([0 400])
% savefig(gcf,'calib_points_fit.fig')
%% Save calib coefficients

save(fullfile(WorkingFolder, 'color_p.mat'), 'color_p');

%% Calibrate and see
clear calib_test
calib_test(:,:,1) = color_p(1,1).*reg_checkr_imgR;
calib_test(:,:,2) = color_p(1,2).*reg_checkr_imgG;
calib_test(:,:,3) = color_p(1,3).*reg_checkr_imgB;
figure;
imagesc(uint8(255.*calib_test./max(max(max(calib_test)))));

%% Save Tiffs
cstruct.Photometric     = Tiff.Photometric.RGB;
cstruct.BitsPerSample   = 8;
cstruct.SamplesPerPixel = 3;
cstruct.PlanarConfiguration = Tiff.PlanarConfiguration.Chunky;
cstruct.Software        = 'MATLAB';
cstruct.SampleFormat    = 1; %uInt1 signed int2 IEEEFP 3
cstruct.Compression = 1;
save_buffer = uint8(255.*calib_test./max(max(max(calib_test))));
cstruct.ImageLength     = size(save_buffer,1);
cstruct.ImageWidth      = size(save_buffer,2);
cstruct.RowsPerStrip    = size(save_buffer,1);
outputFileName = fullfile(WorkingFolder, 'cam_color_checkr.tif');
outputTiff = Tiff(outputFileName, 'w');
outputTiff.setTag(cstruct)
outputTiff.write(save_buffer);
outputTiff.close();
```

```
% iCMOS color rconstruction
% P-Bruza_2020

clear;clc;
%% Specify Working Folders

WorkingFolder='E:\Vihan'; % Specifiy working folder here
addpath(genpath(WorkingFolder));
addpath(genpath('E:\Matlabfunctions')); % Load any external matlab functions
you need (read_dovi, for example)
cd(WorkingFolder)

data_path = 'E:\Color Run 1\data\cherenkov\Stain Large'; % path to your data
addpath(genpath(data_path));

%% Specify Tiff Header Parameters

tagstruct.Photometric     = Tiff.Photometric.MinIsBlack;
tagstruct.BitsPerSample   = 32;
tagstruct.SamplesPerPixel = 1;
tagstruct.PlanarConfiguration = Tiff.PlanarConfiguration.Chunky;
tagstruct.Software        = 'MATLAB';
tagstruct.SampleFormat    = 3; %uInt1 signed int2 IEEEFP 3
tagstruct.Compression = 1;
%tagstruct.ResolutionUnit = 3; %2 inch 3 cm
%tagstruct.XResolution = 0.05;
%tagstruct.YResolution = 0.05;


%% Read Raw Color Checkerboard Images

current_folder = fullfile('2021-12-15 19-01-03-032'); % specific data folder
filetext = fileread(fullfile(data_path, current_folder,'settings.ini'));

% match serial numbers ----------
n0 = str2double(regexp(filetext, '(?<=Camera0_Serial_Number=[^0-9]*)[0-
9]*\.?[0-9]+', 'match'));
n1 = str2double(regexp(filetext, '(?<=Camera1_Serial_Number=[^0-9]*)[0-
9]*\.?[0-9]+', 'match'));
n2 = str2double(regexp(filetext, '(?<=Camera2_Serial_Number=[^0-9]*)[0-
9]*\.?[0-9]+', 'match'));
colorcalib_order = [n0, n1, n2];

% read images ----------
for cam = 1:3
    disp(['Reading in data from cam ', num2str(cam-1), '...'])
    data_colorcalib =
double(read_dovi(fullfile(data_path,current_folder,['meas_s0_cam',num2str(cam-
1),'.dovi'])));

    disp(['Median filtering data from cam ', num2str(cam-1), '...'])
    parfor i=1:size(data_colorcalib, 3) % this loop is spatial median
filtering each frame
        data_colorcalib(:,:,i) = medfilt2(data_colorcalib(:,:,i), [5,5]);
    end
```

```
     data_colorcalib = medfilt1(data_colorcalib, 5, [], 3); % this line is
temporal median filtering each pixel
    %If need to crop
    %data_colorcalib = data_colorcalib_raw(:,160:1599+160,:);
    colorcalib_mean(:,:,cam)  = flip(mean(data_colorcalib,3),1);

    clear data_colorcalib
end

%% Load in registration tforms

load('color_calib_tforms.mat'); % load tform.mat file (whatever it's called)

%% Load in color calibration

load('color_p.mat'); % load color_p.mat file (whatever it's called)


%%

reg_img = zeros(size(colorcalib_mean));

reg_imgG = colorcalib_mean(:,:,1);
reg_imgR = imwarp(colorcalib_mean(:,:,2), color_calib_tforms.tform_1,
'OutputView', imref2d(size(reg_imgG)));
reg_imgB = imwarp(colorcalib_mean(:,:,3), color_calib_tforms.tform_2,
'OutputView', imref2d(size(reg_imgG)));

% reassign and crop

reg_img(:,:,1) = reg_imgR;
reg_img(:,:,2) = reg_imgG;
reg_img(:,:,3) = reg_imgB;

reg_img = reg_img(150:end, :, :);

%% Read in Raw DOVI and Imcontrast for ImageJ mean RGB Value Check
I = sum(read_dovi(['E:\Color Run 1\data\cherenkov\Stain Large\2021-12-15 18-
54-27-221\meas_s1_cam2.dovi']), 3);
f1 = figure;
imagesc(I);
axis image;
colorbar;
caxis([0, 100000]); % set M to be some number that looks appropriate

%% Color recon

calib_img(:,:,1) = color_p(1,1).*reg_img(:,:,1) + color_p(2,1);
calib_img(:,:,2) = color_p(1,2).*reg_img(:,:,2) + color_p(2,2);
calib_img(:,:,3) = color_p(1,3).*reg_img(:,:,3) + color_p(2,3);
figure;
imagesc(uint8(255.*calib_img./max(max(max(calib_img)))));

% Select region of interest in image
```

```
h = drawpolygon()
m = uint8(createMask(h));
im_m = uint8(255.*calib_img./max(max(max(calib_img)))).*m;

%Get average R, G, B intensity of light for plotting
R_Array = im_m(:, :, 1);
avg_R = mean2(R_Array);
G_Array = im_m(:, :, 2);
avg_G = mean2(G_Array);
B_Array = im_m(:, :, 3);
avg_B = mean2(B_Array);

% Rongxiao Zhang, 2022
% Script to correct optical aberrations from tri colour camera image output
from color_recon_new.m script

% Reference the output image from color recon script (have to finish running
% color_recon_new.m script first
im_final = uint8(255.*calib_img./max(max(max(calib_img))));
%Display this final image to look for possible corrections/shifting
imshow(im_final);
axis equal;
axis tight;
%% Example shift, can be edited to add another channel or change rotation
degrees or coordinates as needed

% Blue channel shift
% Correcting by rotation and cropping, channel 1, from 943 to 876 coordinates
by 2.5 degree rotation
im_tmp = imrotate(circshift(squeeze(im_final(:,:,1)),943-876,2),2.5,'crop');
im_shift = im_final;
im_shift(:,:,1) = im_tmp;

% Red channel shift
% Correcting by rotation and cropping, channel 3, from 970 to 949 coordinates
by 3 degree rotation
im_tmp = imrotate(circshift(squeeze(im_final(:,:,3)),-(970-930),2),1,'crop');
im_shift(:,:,3) = im_tmp;

%Green channel shift
% im_tmp = imrotate(circshift(squeeze(im_final(:,:,2)),-(974-
960),2),0,'crop');
% im_shift(:,:,2) = im_tmp;

%Show shifted and corrected image
imshow(im_shift);
axis equal;
axis tight;

function data = read_dovi(file_name, zrange)

% Copyright DoseOptics LLC 2017
%
% Reads DOVI data format into Matlab
% @param file_name - File name/location of DOVI file (e.g. 'test.dovi')
% @param zrange - [Optional] Range of slices to load (e.g. [1 40])
% @param data - Output data in Matlab


data = [];

% check inputs
if (length(file_name) < 5)
    return;
end
if (exist(fullfile(pwd, file_name), 'file'))
    file_name = fullfile(pwd, file_name);
end

% load header (dovi)
fs = fopen(file_name);
contents = textscan(fs,'%s');
fclose(fs);
x = 800;
y = 600;
z = 1;
compressed = 0;
cmp_size = 0;
ucmp_size = 0;
scalar_bytes = 2;
block = 0;
for i=1:length(contents{1})
    if (length(contents{1}{i}) > 5)
        if (contents{1}{i}(1:5) == 'dims0')
            x = str2num(contents{1}{i}(7:end));
        end
        if (contents{1}{i}(1:5) == 'dims1')
            y = str2num(contents{1}{i}(7:end));
        end
        if (contents{1}{i}(1:5) == 'dims2')
            z = str2num(contents{1}{i}(7:end));
        end
        if (contents{1}{i}(1:5) == 'block')
            block = str2num(contents{1}{i}(7:end));
        end
        if (length(contents{1}{i}) > 11)
            if (contents{1}{i}(1:11) == 'compressed=')
                compressed = str2num(contents{1}{i}(12));
            end
        end
        if (length(contents{1}{i}) > 15)
            if (contents{1}{i}(1:15) == 'compressed_size')
                cmp_size = str2num(contents{1}{i}(17:end));
            end
        end
        if (length(contents{1}{i}) > 17)
            if (contents{1}{i}(1:17) == 'uncompressed_size')
                ucmp_size = str2num(contents{1}{i}(19:end));
            end
        end
        if (length(contents{1}{i}) > 12)
            if (contents{1}{i}(1:12) == 'scalar_bytes')
                scalar_bytes = str2num(contents{1}{i}(14:end));
            end
        end
    end
end

fn_new = [file_name(1:end-5) '.raw'];
if (compressed)
    fn_new = [file_name(1:end-5) '.rawu'];
    if (~exist(fn_new, 'file'))
        % decompress
        disp('Uncompressing...');
        systemcall = ['"' which('uncompress_dovi.exe') '" "' file_name(1:end-
5) '" ' ...
            num2str(block) ' ' num2str(x) ' ' num2str(y) ' ' num2str(z) ' '
...
            num2str(scalar_bytes) ' ' num2str(cmp_size)];
        system(systemcall);
    end
end

if (nargin < 2)
    zrange = [1 z];
end

% load data
if (~exist(fn_new, 'file'))
    return;
end
fs = fopen(fn_new);
for i=1:zrange(1) - 1
    fread(fs, [x,y], 'uint16');
end
data = fread(fs, [x,y*(zrange(2) - zrange(1) + 1)], 'uint16');
fclose(fs);
data = uint16(reshape(data, x, y, zrange(2) - zrange(1) + 1));
data = permute(data, [2 1 3]);

% cleanup
% if (compressed)
%     delete(fn_new);
% end


%% Read in Raw DOVI and Imcontrast for mean RGB Value Check
I = sum(read_dovi(('E:\Color Run 1\data\cherenkov\8_23_2022\2022-08-23 18-11-
57-122\meas_s1_cam0.dovi'), 3));
f1 = figure;
imagesc(I);
axis image;
colorbar;
caxis([0, 100000]); % set M to be some number that looks appropriate
```

```
% S. Streeter
% 6/18/2020

% Original Modulim code from R. Hachadorian 8/29/2019, provided by Modulim

% If you forget "out name" aka "processing tag", then go into ../PROCESSED/
% and then look in the first .txt file and slide all the way to the end.
% The "outname" will be there. This is for the *** line below.

% This code is for Alberto Ruiz's resin printed phantoms project

% NOTE: % The OPTIONS variable provides a way of performing additional
% adjustments/processing specific to one dataset. If there are not options
% to add, this variable will be zero.

clc; clear all; close all;

% Add dependency for colormap
addpath('.\pmkmp');

% Generate out(put) data structure, customize ROIs (if CALC == 0, skip this
% step and load MAT file with this info)
CALC = 1;

% Set the initial size of interactive ROI shapes
RECSZ  = 65; % Length of rectangular ROI (starts as a square) in pixels
RADIUS = 50; % Radius of circular ROI in pixels

% Set the data path

%--------------------------------------------------------------------------
% Vihan's first set of silicone tissue-like phantoms
%pth = '.\211215\';
%fn1 = 'Sample1_2021.12.15.08.30.10';
%fn2 = 'vihan2';
%OPTIONS = 0;
%--------------------------------------------------------------------------
% Vihan's second set of silicone tissue-like phantoms - thinner layers
% SET 1/3
%pth = '.\Thinner Layers\';
%fn1 = 'vihan_2022.02.21.07.38.50';
%fn2 = 'vihan2'; % % <-- same as before
%OPTIONS = 0;
%--------------------------------------------------------------------------
% Vihan's second set of silicone tissue-like phantoms - thinner layers
% SET 2/3
pth = '.\Thinner Layers\';
fn1 = 'vihan_2022.02.21.07.38.50';
fn2 = 'vihan2'; % % <-- same as before
OPTIONS = 0;
%--------------------------------------------------------------------------

% Baseline output directory
pth_out = sprintf('%s%s\\OUTPUT\\',pth,fn1);
if exist(pth_out, 'dir')
```

```
else
    mkdir(pth_out)
end

% Read in Reflect RS data
readoptions.TissueName      = fn1;
readoptions.outname         = fn2;      % <-- ***
readoptions.DataDirectory   = pth;
readoptions.OutputDirectory = pth;
readoptions.Measurement     = 'visnir'; % Specifies wavelength regime 'nir' or
'visnir'
out = ReadReflectRSData(readoptions);   % Loads data <-- output structure here

% Continue with all steps if MAT file isn't generated yet
if CALC == 1

    % Display example image, count ROIs
    temp = out.op_maps(:,:,1,1);
    f0 = figure('Position',[100 100 1200 1200],'Color',[1 1 1]);
    imshow(temp,[0 0.5]);
    axis off;
    axis image;

    % Input number of circular ROIs
    % Credit: https://www.mathworks.com/matlabcentral/answers/49942-how-to-
limit-input-to-a-whole-interger
    flag = 0; fprintf('\n');
    while flag == 0
        Nc = input('Enter number of circular ROIs: ','s');
        try
            Nc = str2double(Nc);
            if fix(Nc) - Nc == 0
                flag = 1;
            else
                disp('Only integer numbers are allowed!');
            end
        catch
            disp('Only integer numbers are allowed!');
        end
    end

    % Input number of rectangular ROIs
    flag = 0;
    while flag == 0
        Nr = input('Enter number of rectangular ROIs: ','s');
        try
            Nr = str2double(Nr);
            if fix(Nr) - Nr == 0
                flag = 1;
            else
                disp('Only integer numbers are allowed!');
            end
        catch
            disp('Only integer numbers are allowed!');
        end
    end

    % Create color for each ROI - use fun and new colormaps
    Nroi = Nc + Nr;
    try % This try-catch statement handles the situation when there's only one
ROI to analyze
        colors    = pmkmp(Nroi,'Swtth');
    catch
        colors = 'blue';
    end

    % Save output absorption and reduced scattering maps (all units of 1/mm)
    % Dimensions of out.op_maps below: (1: x, 2: y, 3: wavelength, 4: 1=mua or
2=musp)
    data_mua = out.op_maps(:,:,:,1);
    data_musp = out.op_maps(:,:,:,2);

    % Pull out wavelengths, the array for which corresponds to the third index
    % of the "data" variables above
    wvlns = out.parameters.wavelengths;

    % Pull out RGB image - it doesn't look good due to binning, I think,
    % but it gives you enough for context, even to quantify RGB channel
    % values if desired
    RGB = out.color;

    % Check if OPTIONS exist for this dataset
    if OPTIONS == 1

    end

    % Save RGB image
    f0 = figure('Position',[100 100 800 800],'Color',[1 1 1]);
    imshow(RGB); axis off;
    saveas(f0,sprintf('%sRGB.png',pth_out));

    % Display example map, adjust ROIs
    I = data_mua(:,:,1);
    f1 = figure('Position',[100 100 800 800],'Color',[1 1 1]);
    imshow(I); title('Hit ''Return'' to continue. Helpful tip: Maximize
figure!');
    axis off;
    label_no = 1;
    if Nc > 0
        for i = 1:Nc
            rois_circles{i} =
images.roi.Circle(gca,'Center',[round(size(temp,2)/2)
round(size(temp,1)/2)],'Radius',RADIUS,'Color',colors(i,:));
            rois_circles{i}.Label  = sprintf('%i',label_no); label_no =
label_no + 1;
        end
    end
    if Nr > 0
        for i = 1:Nr
```

```
            rois_recs{i} = 
images.roi.Rectangle(gca,'Position',[round(size(temp,2)/2) 
round(size(temp,1)/2) RECSZ RECSZ],'Color',colors(i,:));
            rois_recs{i}.Label  = sprintf('%i',label_no); label_no = label_no 
+ 1;
        end
    end

    % Now wait until RETURN key is pressed - gives user time to move/adjust 
ROIs
    % Credit: https://www.mathworks.com/matlabcentral/answers/251319-how-can-
i-pause-until-the-return-key-is-pressed
    fprintf('\nAdjust ROIs, then hit RETURN key to continue...\n');
    currkey = 0;
    % Do not move on until enter key is pressed
    while currkey ~= 1
        pause; % Wait for a keypress
        currkey = get(gcf,'CurrentKey');
        if strcmp(currkey, 'return')
            currkey = 1;
        else
            currkey = 0;
        end
    end

    % Save ROI image
    title('');
    saveas(f1,[pth_out 'ROI_map.png']);

    % Create mask - number each ROI based on label_no value
    mask = zeros(size(data_mua(:,:,1)));
    if Nc > 0
        for i = 1:Nc
            mask = mask + rois_circles{i}.createMask*i;
        end
    end
    if Nr > 0
        for i = 1:Nr
            mask = mask + rois_recs{i}.createMask*(i+Nc);
        end
    end

    % Save coded mask
    f2 = figure('Position',[100 100 800 800],'Color',[1 1 1]);
    imshow(mask,[]);
    axis off;
    saveas(f2,[pth_out 'ROI_mask.png']);

    % Now duplicate mask for ROI averaging
    mask = repmat(mask,[1 1 length(wvlns)]);

    % Added 2022/02/21 by Vihan and Sam
    % Get calibrated reflectance values
    Rd = out.planar_Rd;
    Rd = Rd(:,:,1:end-1); % Drop the 971 wavelength, now Rd has the same # of
wavelengths as the opt. props. matrices

    % Average each ROI with respect to wavelength, also quantify standard
deviation and median
    Nroi = Nc + Nr;
    avg_mua  = zeros(length(wvlns),Nroi);
    avg_musp = zeros(length(wvlns),Nroi);
    avg_Rd   = zeros(length(wvlns),Nroi);
    std_Rd   = zeros(length(wvlns),Nroi);
    std_mua  = zeros(length(wvlns),Nroi);
    std_musp = zeros(length(wvlns),Nroi);
    med_mua  = zeros(length(wvlns),Nroi);
    med_musp = zeros(length(wvlns),Nroi);
    for j = 1:Nroi

        % Must set all zero-valued elements to NaN, so they aren't included in
statistics
        temp_mask = mask == j;
        temp_convert = double(temp_mask);
        temp_convert(temp_convert==0) = nan;

        temp_mua  = temp_convert .* data_mua;
        temp_musp = temp_convert .* data_musp;
        temp_Rd   = temp_convert .* Rd;

        for i = 1:length(wvlns)

            % Absorption
            slice = temp_mua(:,:,i);
            avg_mua(i,j) = nanmean(slice(:));
            std_mua(i,j) = nanstd(slice(:));
            med_mua(i,j) = nanmedian(slice(:));

            % Reduced scattering
            slice = temp_musp(:,:,i);
            avg_musp(i,j) = nanmean(slice(:));
            std_musp(i,j) = nanstd(slice(:));
            med_musp(i,j) = nanmedian(slice(:));

            % Calibrated reflectance
            slice = temp_Rd(:,:,i);
            avg_Rd(i,j) = nanmean(slice(:));
            std_Rd(i,j) = nanstd(slice(:));

        end
    end

    % Save processed optical property maps and corresponding mask matrix
    save([pth_out
'processed_data.mat'],'data_mua','data_musp','avg_mua','avg_musp','avg_Rd','st
d_mua','std_musp','std_Rd','med_mua','med_musp','wvlns','Nc','Nr');

else
```

```
    try
        load([pth_out 'processed_data.mat']);
        Nroi = Nc + Nr;
        try % This try-catch statement handles the situation when there's only
one ROI to analyze
            colors    = pmkmp(Nroi,'Swtth');
        catch
            colors = 'blue';
        end
    catch
        fprintf('\nWARNING: CALC = 0, but there is no processed MAT file
found! Set CALC = 1, then rerun!\n');
        return
    end

end % End of CALC == 0 or 1
%---------------------------------------------------------------------------

% Now generate plots
FSZ = 18;

% Reduced scattering map
f3 = figure('Position',[100 100 700 500],'Color',[1 1 1]);
for i = 1:Nroi
    h(i) =
errorbar(wvlns,avg_musp(:,i),std_musp(:,i),'Color',colors(i,:),'LineWidth',2);
hold on;
    l{i} = sprintf('%i',i);
end
grid on;
xlim([min(wvlns) max(wvlns)])
leg = legend(h,l,'Location','northeastoutside');
htitle = get(leg,'Title');
set(htitle,'String','ROI')
xlabel('\lambda (nm)');
ylabel('\mu_s'' (1/mm)');
title('Reduced Scattering');
if Nroi < 10
    leg = legend(h,l,'Location','northeastoutside');
    htitle = get(leg,'Title');
    set(htitle,'String','ROI');
    set(findall(gcf,'-property','FontSize'),'FontSize',FSZ);
else
    set(findall(gcf,'-property','FontSize'),'FontSize',FSZ);
    leg = legend(h,l,'Location','northeastoutside','FontSize',FSZ-8);
    htitle = get(leg,'Title');
    set(htitle,'String','ROI');
    leg.NumColumns = 2;
end
saveas(f3,sprintf('%sPlot_Average_OneStd_musp.fig',pth_out));
saveas(f3,sprintf('%sPlot_Average_OneStd_musp.png',pth_out));

% Absorption map
f4 = figure('Position',[100 100 700 500],'Color',[1 1 1]);
for i = 1:Nroi
```

```
    h(i) =
errorbar(wvlns,avg_mua(:,i),std_mua(:,i),'Color',colors(i,:),'LineWidth',2);
hold on;
    l{i} = sprintf('%i',i);
end
grid on;
xlim([min(wvlns) max(wvlns)])
leg = legend(h,l,'Location','northeastoutside');
htitle = get(leg,'Title');
set(htitle,'String','ROI #')
xlabel('\lambda (nm)');
ylabel('\mu_a (1/mm)');
title('Absorption');
if Nroi < 10
    leg = legend(h,l,'Location','northeastoutside');
    htitle = get(leg,'Title');
    set(htitle,'String','ROI');
    set(findall(gcf,'-property','FontSize'),'FontSize',FSZ);
else
    set(findall(gcf,'-property','FontSize'),'FontSize',FSZ);
    leg = legend(h,l,'Location','northeastoutside','FontSize',FSZ-8);
    htitle = get(leg,'Title');
    set(htitle,'String','ROI');
    leg.NumColumns = 2;
end
saveas(f4,sprintf('%sPlot_Average_OneStd_mua.fig',pth_out));
saveas(f4,sprintf('%sPlot_Average_OneStd_mua.png',pth_out));

% Calibrated reflectance map
f5 = figure('Position',[100 100 700 500],'Color',[1 1 1]);
for i = 1:Nroi
    h(i) =
errorbar(wvlns,avg_Rd(:,i),std_Rd(:,i),'Color',colors(i,:),'LineWidth',2);
hold on;
    l{i} = sprintf('%i',i);
end
grid on;
xlim([min(wvlns) max(wvlns)])
leg = legend(h,l,'Location','northeastoutside');
htitle = get(leg,'Title');
set(htitle,'String','ROI #')
xlabel('\lambda (nm)');
ylabel('R_d (-)');
title('Calibrated Reflectance');
if Nroi < 10
    leg = legend(h,l,'Location','northeastoutside');
    htitle = get(leg,'Title');
    set(htitle,'String','ROI');
    set(findall(gcf,'-property','FontSize'),'FontSize',FSZ);
else
    set(findall(gcf,'-property','FontSize'),'FontSize',FSZ);
    leg = legend(h,l,'Location','northeastoutside','FontSize',FSZ-8);
    htitle = get(leg,'Title');
    set(htitle,'String','ROI');
    leg.NumColumns = 2;
```

```
end
ylim([0 1]);
saveas(f5,sprintf('%sPlot_Average_OneStd_Rd.fig',pth_out));
saveas(f5,sprintf('%sPlot_Average_OneStd_Rd.png',pth_out));

%--------------------------------------------------------------------------

% If you wish to VISUALIZE separate images for mua and musp w.r.t. all
% wavelengths, uncomment the code below. IMPORTANT: Note that MATLAB does
% not have a way to visualize single or double precision data in image
% form. In other words, the images that are saved below will NOT be
% absolute values of optical properties. They will only provide a relative
% and qualitative depiction of the optical property maps. This depiction is
% good for identifying artifacts, for instance.

% Create output image directories
pth_out_abso = sprintf('%s/optical_prop_map_mua/',pth_out);
pth_out_musp = sprintf('%s/optical_prop_map_musp/',pth_out);
if exist(pth_out_abso, 'dir')
else
    mkdir(pth_out_abso)
end
if exist(pth_out_musp, 'dir')
else
    mkdir(pth_out_musp)
end

% Output images
for i = 1:length(wvlns)
    imwrite(data_mua(:,:,i),sprintf('%smua_%inm.png',pth_out_abso,wvlns(i)));

imwrite(data_musp(:,:,i),sprintf('%smusp_%inm.png',pth_out_musp,wvlns(i)));
end

%--------------------------------------------------------------------------

% Commandline summary for each ROI
fprintf('Wavelength range:   %i <-> %i nm\n',wvlns(1),wvlns(end));
fprintf('(Stats given for these wavelength limits)\n');
for i = 1:Nroi
    fprintf('ROI No. %i:\n',i);
    fprintf('   (Average +/- std. deviation)\n');
    fprintf('   Reduced scattering: %.4f+/-%.4f <-> %.4f+/-%.4f
1/mm\n',avg_musp(1,i),std_musp(1,i),avg_musp(end,i),std_musp(end,i));
    fprintf('   Absorption:         %.4f+/-%.4f <-> %.4f+/-%.4f
1/mm\n',avg_mua(1,i),std_mua(1,i),avg_mua(end,i),std_mua(end,i));
    fprintf('   (Median)\n');
    fprintf('   Reduced scattering: %.4f <-> %.4f
1/mm\n',med_musp(1,i),med_musp(end,i));
    fprintf('   Absorption:         %.4f <-> %.4f
1/mm\n',med_mua(1,i),med_mua(end,i));
end

% Write summary stats to text file
fileID = fopen(sprintf('%sStats_Summary.txt',pth_out),'w');
```

```
fprintf(fileID,'Wavelength range:   %i <-> %i nm\n',wvlns(1),wvlns(end));
fprintf(fileID,'(Stats given for these wavelength limits)\n');
for i = 1:Nroi
    fprintf(fileID,'ROI No. %i:\n',i);
    fprintf(fileID,'   (Average +/- std. deviation)\n');
    fprintf(fileID,'   Reduced scattering: %.4f+/-%.4f <-> %.4f+/-%.4f 
1/mm\n',avg_musp(1,i),std_musp(1,i),avg_musp(end,i),std_musp(end,i));
    fprintf(fileID,'   Absorption:         %.4f+/-%.4f <-> %.4f+/-%.4f 
1/mm\n',avg_mua(1,i),std_mua(1,i),avg_mua(end,i),std_mua(end,i));
    fprintf(fileID,'   (Median)\n');
    fprintf(fileID,'   Reduced scattering: %.4f <-> %.4f 
1/mm\n',med_musp(1,i),med_musp(end,i));
    fprintf(fileID,'   Absorption:         %.4f <-> %.4f 
1/mm\n',med_mua(1,i),med_mua(end,i));
end

%---------------------------------------------------------------------------

% Header row for CSV file
header = cell([1 Nroi+1]);
header{1} = 'Wavelength (nm)';
for i = 1:Nroi
    header{i+1} = sprintf('ROI %i',i);
end

% Create CSV
filename = fullfile(pth_out, 'Stats.csv');
[fid, msg] = fopen(filename, 'wt');
if fid < 0
  error('Could not open file "%s" because "%s"', fid, msg);
end

% Create fprintf printing format - the same for all rows of stats
print_str = '%i,'; % <-- needed for wavelength
for i = 1:Nroi
    print_str = [print_str '%.6f,']; % <-- depends on number of ROIs
end
print_str = [print_str '\n'];

% Write blocks of stats to CSV
CSV_block(fid,header,'ABSORPTION AVERAGES (1/mm)\n',print_str,wvlns,avg_mua);
CSV_block(fid,header,'ABSORPTION MEDIANS (1/mm)\n',print_str,wvlns,med_mua);
CSV_block(fid,header,'ABSORPTION STANDARD DEVIATIONS 
(1/mm)\n',print_str,wvlns,std_mua);
CSV_block(fid,header,'REDUCED SCATTERING AVERAGES 
(1/mm)\n',print_str,wvlns,avg_musp);
CSV_block(fid,header,'REDUCED SCATTERING MEDIANS 
(1/mm)\n',print_str,wvlns,med_musp);
CSV_block(fid,header,'REDUCED SCATTERING STANDARD DEVIATIONS 
(1/mm)\n',print_str,wvlns,std_musp);
CSV_block(fid,header,'CALIBRATED REFLECTANCE AVG (-
)\n',print_str,wvlns,avg_Rd);
CSV_block(fid,header,'CALIBRATED REFLECTANCE STD (-
)\n',print_str,wvlns,std_Rd);
fclose(fid);
```

```
%---------------------------------------------------------------------------

function CSV_block(fileID,header,title_str,print_str,wavelengths,data)

    fprintf(fileID,title_str);
    for i = 1:length(header)
        fprintf(fileID,'%s,',header{i});
    end
    fprintf(fileID,'\n');
    for i = 1:length(wavelengths)
        fprintf(fileID,print_str,wavelengths(i),data(i,:));
    end
    fprintf(fileID,'\n\n');

end

   function out = ReadReflectRSData(readoptions)
% -------------------------------------------------------------------------
% Reads in Reflect RS data
% -------------------------------------------------------------------------
% -------------------------------------------------------------------------
% Inputs:
% readoptions: various inputs to read data
% -------------------------------------------------------------------------
% Outputs:

% Assign % directory information
dir.basedir=readoptions.DataDirectory;
dir.procdir=readoptions.OutputDirectory;
basename=readoptions.TissueName;

% Load acquisition and analysis xml options
% parameters = ReadRSParametersFromXML([dir.basedir basename '\' basename  '_'
'parameters.xml' ]);
% if isfield(readoptions,'SeperateReps')
%     Seperatereps=readoptions.SeperateReps;
% else,
%     Seperatereps=1;
% end;
% if Seperatereps
% analysis_parameters=ReadRSOptionsFromXML([dir.procdir basename '\' basename
'-0_' readoptions.outname '_options.xml' ]);
% else
% analysis_parameters=ReadRSOptionsFromXML([dir.procdir basename '\' basename
'_' readoptions.outname '_options.xml' ]);
% end;

MeasurementType=readoptions.Measurement;

if MeasurementType(1:3)=='ski'
    parameters.wavelengths=[471,526,621,731,851];
    parameters.freqs=[0,0.15];
    parameters.planar_wavelengths=[971];
    analysis_parameters.model='skin'
```

```
elseif MeasurementType(1:3)=='vis'
    parameters.wavelengths=[471,526,591,621,659,691,731,851]
    parameters.freqs=[0,0.05,0.1,0.15,0.2]
    parameters.planar_wavelengths=[471,526,591,621,659,691,731,851,971];
    analysis_parameters.model='homogenous';
elseif MeasurementType(1:3)=='nir'
    parameters.wavelengths=[659,691,731,851];
    parameters.freqs=[0,0.05,0.1,0.15,0.2];
    parameters.planar_wavelengths=[659,691,731,851,971];
    analysis_parameters.model='homogenous';
end;

parameters.CamBinSize=2;
parameters.phases=[0,120,240];
parameters.xlength=696;
parameters.ylength=520;


% freqs = parameters.freqs;
phases = parameters.phases;
np = length(phases);
wavelengths = parameters.wavelengths;
nWv = length(wavelengths);
if isfield(parameters,'planar_wavelengths')
    planar_wavelengths = parameters.planar_wavelengths;
    nWv_planar = length(planar_wavelengths);
else,
end;

% Load spatial freqeuncy information
frequencies=parameters.freqs;
nfx = length(frequencies);
if isfield(parameters,'profile_freqs')
    profile_frequencies = parameters.profile_freqs;
    nprofilefx = length(profile_frequencies);
end

% Default options
readoptions.Rd=1;
readoptions.chrom=1;
readoptions.planar_Rd=1;
readoptions.color=1;
readoptions.mask=1;

% Use model to identify number of chromophores
if isfield(analysis_parameters,'model')
    if analysis_parameters.model(1:4)=='skin'
        parameters.chroms={'Mel','HbT_1','HbT_2','A','StO_2'};
        parameters.chrom_limits=[0,0.1;0,0.1;0,0.15;0,4;0,1];
        readoptions.op=0;
    else
        parameters.chroms={'HbO','HbR'};
        parameters.chrom_limits=[0,100;0,100];
        readoptions.op=1;
    end;
```

```
end;
nchrom=length(parameters.chroms);

% Define image dimensions
pixelWidth=parameters.xlength;
pixelHeight=parameters.ylength;
nPixels = pixelWidth*pixelHeight;
out.parameters=parameters;
outname=readoptions.outname;

% Assign directories for reading data
if isfield(dir,'basedir')
    basedir=dir.basedir;
end;
if isfield(dir,'procdir')
    procdir=dir.procdir;
end;

% Standard loading parameters
repnums=0;



%% Read data
if isfield(readoptions,'SeperateReps')
    Seperatereps=readoptions.SeperateReps;
else,
    Seperatereps=1;
end;

if Seperatereps==0
    repnums=-1;
end;

if isfield(readoptions,'raw')
    if readoptions.raw
        out.raw=zeros(pixelHeight,pixelWidth,nWv,nfx,np);

        disp(['Reading Raw Data: ' basename]);
        filename_raw = [basedir basename '\' basename '-' int2str(repnums)
'_raw'];
        [fid_raw, message] = fopen(filename_raw, 'rb');
        for wvidx = 1:nWv
            for fidx=1:nfx
                for pidx=1:np
                    rawtemp = fread(fid_raw,[pixelWidth
pixelHeight],'float')';
                    out.raw(:,:,wvidx,fidx,pidx)=rawtemp;
                end;
            end;
        end
        fclose(fid_raw);
    end;
end;
```

```
if isfield(readoptions,'planar_raw')
    if readoptions.planar_raw
        out.planar_raw=zeros(pixelHeight,pixelWidth,nWv_planar,2);

        disp(['Reading Raw Data: ' basename]);
        filename_praw = [basedir basename '\' basename '-' int2str(repnums)
'_planar'];
        [fid_praw, message] = fopen(filename_praw, 'rb');
        for wvidx = 1:nWv_planar
            for k=1:2
                prawtemp = fread(fid_praw,[pixelWidth pixelHeight],'float')';
                out.planar_raw(:,:,wvidx,k)=prawtemp;
            end
        end;
        fclose(fid_praw);
    end;
end;

if isfield(readoptions,'op')
    if readoptions.op
        out.op_maps=zeros(pixelHeight,pixelWidth,nWv,2);

        disp(['Reading Absorption Rep: ' basename]);
        filename_mua = [procdir basename '\' basename '-' int2str(repnums) '_'
outname '_mua'];
        [fid_mua, message] = fopen(filename_mua, 'rb');
        for wvidx = 1:nWv
            muadatatemp = fread(fid_mua,[pixelWidth pixelHeight],'float')';
            out.op_maps(:,:,wvidx,1)=muadatatemp;
        end
        fclose(fid_mua);

        disp(['Reading Scattering Rep: ' basename]);
        filename_mus = [procdir basename '\' basename '-' int2str(repnums) '_'
outname '_musp'];
        [fid_mus, message] = fopen(filename_mus, 'rb');
        for wvidx = 1:nWv
            musdatatemp = fread(fid_mua,[pixelWidth pixelHeight],'float')';
            out.op_maps(:,:,wvidx,2)=musdatatemp;
        end
        fclose(fid_mus);
    end;
end;

if isfield(readoptions,'demod')
    if readoptions.demod

        disp(['Reading Demod Data: ' basename ]);
        filename_demod = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_demod'];
        [fid_demod message] = fopen(filename_demod, 'rb');
        filename_demod
        for k=1:nWv
            for j=1:nfx
```

```
                demod(:,:,j,k) = fread(fid_demod,[pixelWidth
pixelHeight],'float')';
            end;
        end;
        out.demod=demod;

        fclose(fid_demod);
    end
end;

if isfield(readoptions,'Rd')
    if readoptions.Rd
        disp(['Reading Reflectance Data: ' basename]);
        if repnums==-1
        filename_rd = [procdir basename '\' basename '_' outname
'_calibrated_rd'];
        else
        filename_rd = [procdir basename '\' basename '-' int2str(repnums) '_'
outname '_calibrated_rd'];
        end
        [fid_rd message] = fopen(filename_rd, 'rb');
        for wvidx = 1:nWv
            for fidx=1:nfx
                rddatatemp = fread(fid_rd,[pixelWidth pixelHeight],'float')';
                out.Rd(:,:,fidx,wvidx)=rddatatemp;
            end;
        end;
        fclose(fid_rd);
    end
end;

if isfield(readoptions,'height')
    if readoptions.height
        if repnums==-1;
            disp(['Reading Height Data ' basename ]);
            filename_h = [procdir basename '\' basename '_' outname
'_height'];
        else
            disp(['Reading Height Data: ' basename ', Rep: '
int2str(repnums)]);
            filename_h = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_height'];
        end;
        [fid_h message] = fopen(filename_h, 'rb');
        height = fread(fid_h,'float')';
        height = permute(reshape(height,[pixelWidth pixelHeight 2]),[2 1 3]);
        out.height=height;

        fclose(fid_h);
    end
end;

if isfield(readoptions,'chrom')
    if readoptions.chrom
```

```
        disp(['Reading Chromophore Data: ' basename ]);
        filename_chrom = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_chrom'];
        [fid_chrom message] = fopen(filename_chrom, 'rb');

        for k=1:nchrom
            chrom(:,:,k) = fread(fid_chrom,[pixelWidth pixelHeight],'float')';
        end;
        out.chrom=chrom;

        fclose(fid_chrom);
    end
end;

if isfield(readoptions,'planar_Rd')
    if readoptions.planar_Rd
        disp(['Reading Planar Reflectance: ' basename]);
        if repnums
                filename_rd = [procdir basename '\' basename '_' outname
'_planar_calibrated_rd'];
        else
                filename_rd = [procdir basename '\' basename '-'
int2str(repnums) '_' outname '_planar_calibrated_rd'];
        end;
        [fid_rd message] = fopen(filename_rd, 'rb');
        for wvidx = 1:nWv_planar
            rddatatemp = fread(fid_rd,[pixelWidth pixelHeight],'float')';
            out.planar_Rd(:,:,wvidx)=rddatatemp;
        end;
        fclose(fid_rd);
    end
end;

if isfield(readoptions,'physio')
    nphysio=2;
    if readoptions.physio

        disp(['Reading Physio Data: ' basename ]);
        filename_physio = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_physio'];
        [fid_physio message] = fopen(filename_physio, 'rb');

        for k=1:nphysio
            physio(:,:,k) = fread(fid_physio,[pixelWidth
pixelHeight],'float')';
        end;
        out.physio=physio;

        fclose(fid_physio);
    end
end;

if isfield(readoptions,'mask')
    if readoptions.mask
```

```
        disp(['Reading Mask Data: ' basename ]);
        filename_mask = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_mask'];
        [fid_mask message] = fopen(filename_mask, 'rb');

        mask(:,:) = fread(fid_mask,[pixelWidth pixelHeight],'float')';

        out.mask=mask;

        fclose(fid_mask);
    end
end;

if isfield(readoptions,'ab')
    if readoptions.ab

        disp(['Reading scattering AB: ' basename ]);
        filename_ab = [procdir basename '\' basename '-' int2str(repnums) '_'
outname '_ab'];
        [fid_ab message] = fopen(filename_ab, 'rb');

        for i=1:2
            ab(:,:,i) = fread(fid_ab,[pixelWidth pixelHeight],'float')';
            out.ab(:,:,i)=ab(:,:,i);
        end

        for yi=1:size(ab,2)
            temp=squeeze(ab(:,yi,:));
            if isnan(temp)
            else,

forward_musp(:,yi,:)=powerLawNorm(squeeze(ab(:,yi,:)),planar_wavelengths,800);
            end;
        end;
        out.forward_musp=forward_musp;

        fclose(fid_ab);
    end
end;

if isfield(readoptions,'color')
    if readoptions.color

        disp(['Reading Color Image: ' basename ]);
        filename_color = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_rgb.bmp'];
        % To do: Add color co-registration for each system
        colorimage = imread(filename_color);
        cexes=184:603; cwhys=123:693;
        colorimage_crop(:,:,:)=flipdim(colorimage(cexes,cwhys,:),1);
        out.color=colorimage_crop;

    end
end;
```

```
if isfield(readoptions,'colorreg')
    if readoptions.colorreg

        disp(['Reading Color Image: ' basename ]);
        filename_color = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_reg.bmp'];
        % To do: Add color co-registration for each system
        colorimage = imread(filename_color);
        out.color=colorimage;

    end
end;

if isfield(readoptions,'align')
    if readoptions.align
        disp(['Reading Alignment Slide: ' basename ]);
        filename_a = [basedir basename '\' basename '-' int2str(repnums)
'_align'];
        [fid_a message] = fopen(filename_a, 'rb');

        align = fread(fid_a,[pixelWidth pixelHeight],'float')';
        out.align=align;

        fclose(fid_a);
    end
end;

if isfield(readoptions,'laser')
    if readoptions.laser
        disp(['Reading Laser Slide ' basename]);
        filename_a = [basedir basename '\' basename '-' int2str(repnums)
'_laser'];
        [fid_a message] = fopen(filename_a, 'rb');

        laser = fread(fid_a,[pixelWidth pixelHeight],'float')';
        out.laser=laser;

        fclose(fid_a);
    end
end;

if isfield(readoptions,'planar_mua')
    if readoptions.planar_mua % Need to make this read planar wavelengths.
        disp(['Reading Absorption Rep: ' basename]);
        filename_mua = [procdir basename '\' basename '-' int2str(repnums) '_'
outname '_planar_mua'];
        [fid_mua, message] = fopen(filename_mua, 'rb');
        for wvidx = 1:nWv_planar;
            muadatatemp = fread(fid_mua,[pixelWidth pixelHeight],'float')';
            out.planar_op_maps(:,:,wvidx,1)=muadatatemp;
        end
        fclose(fid_mua);
    end;
end;
```

```
if isfield(readoptions,'planar_chrom')
    nchrom=2;
    if readoptions.planar_chrom

        disp(['Reading Height Data ' basename ]);
        filename_chrom = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_planar_chrom'];
        [fid_chrom message] = fopen(filename_chrom, 'rb');

        for k=1:nchrom
            chrom(:,:,k) = fread(fid_chrom,[pixelWidth pixelHeight],'float')';
        end;
        out.planar_chrom=chrom;

        fclose(fid_chrom);
    end
end;

if isfield(readoptions,'planar_physio')
    nphysio=2;
    if readoptions.planar_physio

        disp(['Reading Physio Data ' basename ]);
        filename_physio = [procdir basename '\' basename '-' int2str(repnums)
'_' outname '_planar_physio'];
        [fid_physio message] = fopen(filename_physio, 'rb');

        for k=1:nphysio
            physio(:,:,k) = fread(fid_physio,[pixelWidth
pixelHeight],'float')';
        end;
        out.planar_physio=physio;

        fclose(fid_physio);
    end
end;


if isfield(readoptions,'calibrated_position')
    nposition=3.*nprofilefx;
    if readoptions.calibrated_position

        disp(['Reading Calibrated Position ' basename ]);
        filename_position = [procdir basename '\' basename '-'
int2str(repnums) '_' outname '_calibrated_position'];
        [fid_position message] = fopen(filename_position, 'rb');

        for k=1:nposition
            position(:,:,k) = fread(fid_position,[pixelWidth
pixelHeight],'float')';
        end;
        out.position=position;

        fclose(fid_position);
    end
end;


% function lineStyles = linspecer(N)
% This function creates an Nx3 array of N [R B G] colors
% These can be used to plot lots of lines with distinguishable and nice
% looking colors.
%
% lineStyles = linspecer(N);  makes N colors for you to use: lineStyles(ii,:)
%
% colormap(linspecer); set your colormap to have easily distinguishable
%                      colors and a pleasing aesthetic
%
% lineStyles = linspecer(N,'qualitative'); forces the colors to all be
distinguishable (up to 12)
% lineStyles = linspecer(N,'sequential'); forces the colors to vary along a
spectrum
%
% % Examples demonstrating the colors.
%
% LINE COLORS
% N=6;
% X = linspace(0,pi*3,1000);
% Y = bsxfun(@(x,n)sin(x+2*n*pi/N), X.', 1:N);
% C = linspecer(N);
% axes('NextPlot','replacechildren', 'ColorOrder',C);
% plot(X,Y,'linewidth',5)
% ylim([-1.1 1.1]);
%
% SIMPLER LINE COLOR EXAMPLE
% N = 6; X = linspace(0,pi*3,1000);
% C = linspecer(N)
% hold off;
% for ii=1:N
%     Y = sin(X+2*ii*pi/N);
%     plot(X,Y,'color',C(ii,:),'linewidth',3);
%     hold on;
% end
%
% COLORMAP EXAMPLE
% A = rand(15);
% figure; imagesc(A); % default colormap
% figure; imagesc(A); colormap(linspecer); % linspecer colormap
%
%   See also NDHIST, NHIST, PLOT, COLORMAP, 43700-cubehelix-colormaps
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% by Jonathan Lansey, March 2009-2013 – Lansey at gmail.com               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%% credits and where the function came from
% The colors are largely taken from:
% http://colorbrewer2.org and Cynthia Brewer, Mark Harrower and The
Pennsylvania State University
%
```

```
%
% She studied this from a phsychometric perspective and crafted the colors
% beautifully.
%
% I made choices from the many there to decide the nicest once for plotting
% lines in Matlab. I also made a small change to one of the colors I
% thought was a bit too bright. In addition some interpolation is going on
% for the sequential line styles.
%
%
%%

function lineStyles=linspecer(N,varargin)

if nargin==0 % return a colormap
    lineStyles = linspecer(128);
    return;
end

if ischar(N)
    lineStyles = linspecer(128,N);
    return;
end

if N<=0 % its empty, nothing else to do here
    lineStyles=[];
    return;
end

% interperet varagin
qualFlag = 0;
colorblindFlag = 0;

if ~isempty(varargin)>0 % you set a parameter?
    switch lower(varargin{1})
        case {'qualitative','qua'}
            if N>12 % go home, you just can't get this.
                warning('qualitiative is not possible for greater than 12
items, please reconsider');
            else
                if N>9
                    warning(['Default may be nicer for ' num2str(N) ' for
clearer colors use: whitebg(''black''); ']);
                end
            end
            qualFlag = 1;
        case {'sequential','seq'}
            lineStyles = colorm(N);
            return;
        case {'white','whitefade'}
            lineStyles = whiteFade(N);return;
        case 'red'
            lineStyles = whiteFade(N,'red');return;
        case 'blue'
            lineStyles = whiteFade(N,'blue');return;
        case 'green'
            lineStyles = whiteFade(N,'green');return;
        case {'gray','grey'}
            lineStyles = whiteFade(N,'gray');return;
        case {'colorblind'}
            colorblindFlag = 1;
        otherwise
            warning(['parameter ''' varargin{1} ''' not recognized']);
    end
end
% *.95
% predefine some colormaps
  set3 = colorBrew2mat({[141, 211, 199];[ 255, 237, 111];[ 190, 186, 218];[
251, 128, 114];[ 128, 177, 211];[ 253, 180, 98];[ 179, 222, 105];[ 188, 128,
189];[ 217, 217, 217];[ 204, 235, 197];[ 252, 205, 229];[ 255, 255, 179]}');
set1JL = brighten(colorBrew2mat({[228, 26, 28];[ 55, 126, 184]; [ 77, 175,
74];[ 255, 127, 0];[ 255, 237, 111]*.85;[ 166, 86, 40];[ 247, 129, 191];[ 153,
153, 153];[ 152, 78, 163]}'));
set1 = brighten(colorBrew2mat({[ 55, 126, 184]*.85;[228, 26, 28];[ 77, 175,
74];[ 255, 127, 0];[ 152, 78, 163]}),.8);

% colorblindSet = {[215,25,28];[253,174,97];[171,217,233];[44,123,182]};
colorblindSet = {[215,25,28];[253,174,97];[171,217,233]*.8;[44,123,182]*.8};

set3 = dim(set3,.93);

if colorblindFlag
    switch N
        %     sorry about this line folks. kind of legacy here because I used
to
        %     use individual 1x3 cells instead of nx3 arrays
        case 4
            lineStyles = colorBrew2mat(colorblindSet);
        otherwise
            colorblindFlag = false;
            warning('sorry unsupported colorblind set for this number, using
regular types');
    end
end
if ~colorblindFlag
    switch N
        case 1
            lineStyles = { [  55, 126, 184]/255};
        case {2, 3, 4, 5 }
            lineStyles = set1(1:N);
        case {6 , 7, 8, 9}
            lineStyles = set1JL(1:N)';
        case {10, 11, 12}
            if qualFlag % force qualitative graphs
                lineStyles = set3(1:N)';
            else % 10 is a good number to start with the sequential ones.
                lineStyles = cmap2linspecer(colorm(N));
            end
        otherwise % any old case where I need a quick job done.
            lineStyles = cmap2linspecer(colorm(N));
    end
end
lineStyles = cell2mat(lineStyles);

end

% extra functions
function varIn = colorBrew2mat(varIn)
for ii=1:length(varIn) % just divide by 255
    varIn{ii}=varIn{ii}/255;
end
end

function varIn = brighten(varIn,varargin) % increase the brightness

if isempty(varargin),
    frac = .9;
else
    frac = varargin{1};
end

for ii=1:length(varIn)
    varIn{ii}=varIn{ii}*frac+(1-frac);
end
end

function varIn = dim(varIn,f)
    for ii=1:length(varIn)
        varIn{ii} = f*varIn{ii};
    end
end

function vOut = cmap2linspecer(vIn) % changes the format from a double array
to a cell array with the right format
vOut = cell(size(vIn,1),1);
for ii=1:size(vIn,1)
    vOut{ii} = vIn(ii,:);
end
end
%%
% colorm returns a colormap which is really good for creating informative
% heatmap style figures.
% No particular color stands out and it doesn't do too badly for colorblind
people either.
% It works by interpolating the data from the
% 'spectral' setting on http://colorbrewer2.org/ set to 11 colors
% It is modified a little to make the brightest yellow a little less bright.
function cmap = colorm(varargin)
n = 100;
if ~isempty(varargin)
    n = varargin{1};
end

if n==1
    cmap =  [0.2005    0.5593    0.7380];
    return;
end
if n==2
     cmap =  [0.2005    0.5593    0.7380;
              0.9684    0.4799    0.2723];
          return;
end

frac=.95; % Slight modification from colorbrewer here to make the yellows in
the center just a bit darker
cmapp = [158, 1, 66; 213, 62, 79; 244, 109, 67; 253, 174, 97; 254, 224, 139;
255*frac, 255*frac, 191*frac; 230, 245, 152; 171, 221, 164; 102, 194, 165; 50,
136, 189; 94, 79, 162];
x = linspace(1,n,size(cmapp,1));
xi = 1:n;
cmap = zeros(n,3);
for ii=1:3
    cmap(:,ii) = pchip(x,cmapp(:,ii),xi);
end
cmap = flipud(cmap/255);
end

function cmap = whiteFade(varargin)
n = 100;
if nargin>0
    n = varargin{1};
end

thisColor = 'blue';

if nargin>1
    thisColor = varargin{2};
end
switch thisColor
    case {'gray','grey'}
        cmapp =
[255,255,255;240,240,240;217,217,217;189,189,189;150,150,150;115,115,115;82,82
,82;37,37,37;0,0,0];
    case 'green'
        cmapp =
[247,252,245;229,245,224;199,233,192;161,217,155;116,196,118;65,171,93;35,139,
69;0,109,44;0,68,27];
    case 'blue'
        cmapp =
[247,251,255;222,235,247;198,219,239;158,202,225;107,174,214;66,146,198;33,113
,181;8,81,156;8,48,107];
    case 'red'
        cmapp =
[255,245,240;254,224,210;252,187,161;252,146,114;251,106,74;239,59,44;203,24,2
9;165,15,21;103,0,13];
    otherwise
        warning(['sorry your color argument ' thisColor ' was not
recognized']);
end
```

```
cmap = interpomap(n,cmapp);
end

% Eat a approximate colormap, then interpolate the rest of it up.
function cmap = interpomap(n,cmapp)
    x = linspace(1,n,size(cmapp,1));
    xi = 1:n;
    cmap = zeros(n,3);
    for ii=1:3
        cmap(:,ii) = pchip(x,cmapp(:,ii),xi);
    end
    cmap = (cmap/255); % flipud??
end



% This function displays the c-dose description for every data folder in the
path
% Written by Dan Alexander, 2019

function display_desc(pathname) %path to study folder (should contain all the
acquisition folders within)

    folders = dir(pathname);
    folders = folders(3:end); % first two entries are meaningless

    for i=1:numel(folders)
        folder_contents = dir(fullfile(folders(i).folder, folders(i).name)); %
check contents of this folder
        m = {folder_contents.name}; % make cell array of content names
        if sum(strcmp('settings.ini', m)) % check if folder has a settings.ini
file
            info = ini2struct(fullfile(folders(i).folder, folders(i).name,
'settings.ini')); % get contents of settings.ini file
            desc = info.general.description; % get the description
            disp([folders(i).name,':  ', desc]) % display folder and
description
        end
    end
end


function Result = ini2struct(FileName)
%==========================================================================
%  Author: Andriy Nych ( nych.andriy@gmail.com )
% Version:        733341.4155741782200
%==========================================================================
%
% INI = ini2struct(FileName)
%
% This function parses INI file FileName and returns it as a structure with
% section names and keys as fields.
%
% Sections from INI file are returned as fields of INI structure.
% Each fiels (section of INI file) in turn is structure.
% It's fields are variables from the corresponding section of the INI file.
%
% If INI file contains "oprhan" variables at the beginning, they will be
% added as fields to INI structure.
%
% Lines starting with ';' and '#' are ignored (comments).
%
% See example below for more information.
%
% Usually, INI files allow to put spaces and numbers in section names
% without restrictions as long as section name is between '[' and ']'.
% It makes people crazy to convert them to valid Matlab variables.
% For this purpose Matlab provides GENVARNAME function, which does
%  "Construct a valid MATLAB variable name from a given candidate".
% See 'help genvarname' for more information.
%
% The INI2STRUCT function uses the GENVARNAME to convert strange INI
% file string into valid Matlab field names.
%
% [ test.ini ]~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%
%     SectionlessVar1=Oops
%     SectionlessVar2=I did it again ;o)
%     [Application]
%     Title = Cool program
%     LastDir = c:\Far\Far\Away
%     NumberOFSections = 2
%     [1st section]
%     param1 = val1
%     Param 2 = Val 2
%     [Section #2]
%     param1 = val1
%     Param 2 = Val 2
%
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%
% The function converts this INI file it to the following structure:
%
% [ MatLab session (R2006b) ]~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%  >> INI = ini2struct('test.ini');
%  >> disp(INI)
%         sectionlessvar1: 'Oops'
%         sectionlessvar2: 'I did it again ;o)'
%             application: [1x1 struct]
%             x1stSection: [1x1 struct]
%            section0x232: [1x1 struct]
%
%  >> disp(INI.application)
%                    title: 'Cool program'
%                  lastdir: 'c:\Far\Far\Away'
%         numberofsections: '2'
%
%  >> disp(INI.x1stSection)
%         param1: 'val1'
%         param2: 'Val 2'
```

```
%
%  >> disp(INI.section0x232)
%         param1: 'val1'
%         param2: 'Val 2'
%
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%
% NOTE.
%
WhatToDoWithMyVeryCoolSectionAndVariableNamesInIniFileMyVeryCoolProgramWrites?
% GENVARNAME also does the following:
%   "Any string that exceeds NAMELENGTHMAX is truncated". (doc genvarname)
% Period.
%
% =========================================================================
Result = [];                            % we have to return something
CurrMainField = '';                     % it will be used later
f = fopen(FileName,'r');                % open file
while ~feof(f)                          % and read until it ends
    s = strtrim(fgetl(f));              % Remove any leading/trailing spaces
    if isempty(s)
        continue;
    end;
    if (s(1)==';')                      % ';' start comment lines
        continue;
    end;
    if (s(1)=='#')                      % '#' start comment lines
        continue;
    end;
    if ( s(1)=='[' ) && (s(end)==']' )
        % We found section
        CurrMainField = genvarname(lower(s(2:end-1)));
        Result.(CurrMainField) = [];    % Create field in Result
    else
        % ??? This is not a section start
        [par,val] = strtok(s, '=');
        val = CleanValue(val);
        if ~isempty(CurrMainField)
            % But we found section before and have to fill it
            Result.(CurrMainField).(lower(genvarname(par))) = val;
        else
            % No sections found before. Orphan value
            Result.(lower(genvarname(par))) = val;
        end
    end
end
fclose(f);
return;

function res = CleanValue(s)
res = strtrim(s);
if strcmpi(res(1),'=')
    res(1)=[];
end
res = strtrim(res);
```

```
return;
```