NTNU – Trondheim
Norwegian University of
Science and Technology

Imaging of biological samples using UV light – determination of fluorescence yield

Torbjørn Vala

Electronics System Design and Innovation
Submission date: July 2014
Supervisor: Lise Lyngsnes Randeberg, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Abstract

A procedure on how to extract intrinsic fluorescence, i.e. fluorescence only due to fluophores, out of biological samples by imaging is given. Measurement of fluorescence of biological tissue will generally be distorted by the presence of absorbers and scatterers, and the measurement of fluorescence for medical diagnosis is limited. A method of recovering intrinsic fluorescence would, however, enable fluorescence measurement as a diagnostical tool. An expression based on a so-called one-dimentional path-length factor X_{1D}, which characterize the effective path length of excitation light penetrated into the biological sample and the escape of fluorescence light from the sample, is proposed and testet in this thesis. The method assumes knowledge of the optical properties the material is assumed, while all other data is acquired by imaging. Software applications for image acquisition and data analysis has been developed, and a user manual of these software applications is provided along with a complete description of procedure on how to recover the intrinsic fluorescence.

The method was tested on homegenous liquid tissue phantoms with different concentrations of fluorophores and absorbers. To complement the measurements, spectrometer was done, but these measurements were not a part of the proposed procedure. The correspondance between experimental results and theory was weak. The obtained values of intrinsic fluorescence were too high for the phantoms with high absorption coefficients compared to the obtained values of intrinsic fluorescence for the phantoms with less absorers. Poor estimation of the optical properties of the materials is suggested as the main reason for unexpected experimental results.

Sammendrag

En prosedyre på hvordan en kan trekke indre fluorescens, dvs. fluorescens bare på grunn fluophorer, ut av biologiske prøver er gitt. Måling av fluorescens i biologiske vev vil bli forstyrret av absorbenter og spredere, og bruken av fluorescens for medisinsk diagnose er da begrenset. En fremgangsmåte for å utvinne indre fluorescens kan imidlertid gjøre måling av fluorescens til et diagnostisk verktøy. Et uttrykk basert på en såkalt én-dimensjonal banelengde-faktor X_{1D}, som karakteriserer den effektive banelengde for eksiteringslys penetrert inn i den biologiske prøven, samt utslipp av fluorescens-lyset fra prøven, er foreslått og testet i denne avhandlingen. Metoden antar kjennskap til materialets optiske egenskaper, mens all annen data vil skaffes ved hjelp av avbilding. Dataprogrammer for å bistå avbilding og gjøre data analyser ble laget, og det ble også en brukerveiledning til disse programmene samt en komplett beskrivelse av prosedyre på hvordan en kan gjenopprette den iboende fluorescensen.

Metoden ble testet på homegene flytende vev fantomer med ulike konsentrasjoner av fluoroforer og absorbenter. For å supplere kamera målingen, ble spektrometer målinger gjort, men disse målingene er ikke en del av den beskrevede metoden.foruten kamera målinger. Sammenhengen mellom eksperimentelle resultater og teorien var svak. Metoden ga for høye verdier for den indre fluorescense på prøvene med mange absorbenter, sammenligner med verdine av indre fluorescens for prøver med mindre absorbenter. Dårlig estimering av de optiske egenskapene til materialene er foreslått som den viktigste årsaken til de uventede eksperimentelle resultatene.

Preface

There excist already a lot of techniques on how to determine intrinsic uorescence by surface measurements. However, while earlier work was limited to point measurements, this work investigates the employemnt of imaging on samples. A functional intrinsic fluorescence measurement system where all the data acquisition is done by a camera, can be a straightforward measurement system of great benefit.

I started working on the thesis in February with limited knowledge within the field, and I became more interested and I really enjoyed the task, as the work was going on. A lot of time was used in developing the calibration theory of the system and the encoding of the software application named 'Fluor'. 'Fluor' is a functional software application that I've written as a part of this thesis. For anyone interested, an email can be sent to me at tobvala@hotmail.com, and I will gladly send over the compiled file and all source files needed in order to run the program.

I would like to thank my supervisor Lise L. Randeberg for her good advice and phantom preparation, and my co-supervisor Matija Milanic, for his good availability, patience and guidance. Another thank goes to Norsk Elektro Optikk, which lent me a computer.

Contents

1 Introduction

The goal of this master thesis is to make a protocol on how to extract intrinsic fluorescence, defined as the the product of the fluorophore absorption coefficient and the fluorescence quantum yield, out of surface measurements of biological samples. Fluorescence quantum yield is here defined as ratio between the photons emitted and the photons absorbed. The observed fluorescence of biological tissue is often significantly distorted by absorbs and scatters. The intrinsic fluorescence is independent of scattered and absorbs, but dependent on the amount of fluorophores. By knowing the intrinsic fluorescence and measuring the fluorophore absorption coefficient, we get the fluorescence quantum yield, which among others, gives information about the biological environment of the fluorophores [20].

There are several tissue diagnosis based on intrinsic fluorescence used researchers. For example, laser-induced fluorescence of colonic tissue can make it possible to distinguish neoplastic from hyperplastic and normal tissue [14], and in [5], fluorescence spectroscopy based on a Monte-Carlo model is used for diagnosis of breast cancer. With spectroscopic measurements, it should be possible to obtain intrinsic fluorescence coefficient and tissue classification for the entire spectrum. Simpler techniques, investigating only two wavelengths, have however been used for cancer diagnosis. In [4], the ratio of fluorescence intensities at 340 and 440 nm, excited at 300 nm, was used to distinguish cancerous breast tissue from normal tissue.

Biological tissues are strongly turbid media, and as a consequence complex light propagation models are required [8]. A change of tissue absorption coefficient, may alter the fluorescence spectra. However, as explained above, intrinsic fluorescence make it possible to isolate change of tissue fluorophores from changes of other optical absorbs and scattered. Several investigators have proposed a connection between fluorescence distortions and diffuse reflectance from biological tissue. More than 50 publications that address the recovery of intrinsic fluorescence are reviewed in [3]. In that paper, the focus is on clinical application and consideration of accuracy and limitations of each technique.

A short presentation of the most important and relevant techniques for this thesis will now be presented.

Background

A photon migration model based on Monte Carlo simulations was introduced by [13]. Here, a model which provides an analytical relationship between the bulk fluorescence and the diffuse reflectance is presented. The distortion is accurately characterized by measuring the diffuse reflectance from the optically thick medium, and extracted from the measured fluorescence to obtain the intrinsic fluorescence. Another technique, presented

in [2], applies Kubelka-Munk theory to find the relationship between the fluorescence of a dilute- and turbid solution. Absorption and scattering coefficients is obtained from diffuse reflectance measurements, and the intrinsic fluorescence is predicted.

A one dimensional diffusion model was derived in [25]. The fluorescence distortion of excitation and emission spectra, together with the diffuse reflectance spectrum, was presented as function of the darkness, which was defined as the ratio of absorption coefficient and reduced scattering coefficient. The darkness parameter was then used to extract intrinsic line shape of fluorescence from measured fluorescence.

Monte Carlo modeling of light transport in tissue yield valuable information about light distributions for any combination of scattering and absorption [23]. Accurate expressions for one-dimensional fluence rate and escape function based upon Monte Carlo simulation was derived in [6]. These expressions were combined in [11] to produce a simple expression for the path-length factor, which characterizes the fluorescence escape and penetration of excitation light in the medium. The intrinsic fluorescence is then related to measured fluorescence by this path-length factor, which in turn is dependent on diffuse reflectance of medium.

Based on photon-migration theory a method for recovering intrinsic fluorescence was developed in [19] and [21]. The method expresses the intrinsic fluorescence in terms of experimentally measured fluorescence and diffuse reflectance, without the use of numerical computations. Analytic expressions for diffuse reflectance derived in [9] was used in the expression for the intrinsic fluorescence. Experimental evidence in [19] verified the correcting method even for absorption and scattering coefficients of same order.

Chosen approach

There are advantages and disadvantages with all of the existing fluorescence recovery methods. However, the chosen approach has to depend on the application. The application of this thesis is *imaging* of biological samples, in contrast to previous groups which only did point measurements.

Kubelka-Munk theory presented in [2] are not suitable when scattering dominates absorption [12]. In addition the Kubelka–Munk theory is not applicable for collimated incident radiation beam, and that might be a problem for the setup of this thesis. Both [2], [25] and [13] assumes uniform optical properties of the medium, which reduce the need for imaging.

Even though the photon-migration model of [19] proved good accuracy, the method is only validated in the 337–700 nm emission range. As we will see later in this report, the emission range of the biological materials of this report, bacon and cheese, have emission range below that range. And the accuracy is better in Monte Carlo simulations, as stated in [3], Monte Carlo simulation "offer the greatest prospective accuracy, constrained only by the quality of tissue model employed". A large drawback of Monte Carlo simulation is that they are relatively time consuming, with high simulation run times, and, the implementation might also be complex. However, Monte Carlo simulations done by other groups can be reused.

The expressions for one-dimensional fluence rate and escape function based upon Monte

Carlo simulation derived in [6], was used in [11] to recover the intrinsic fluorescence. The theory of [11] is what this report is based on. The theory was adapted for camera imaging and a protocol, based on this article made.

2 Theory

2.1 Characterization of fluorescence and Jablonski diagram

Fluorescence is defined as the radiative transition between two electronic states in the same spin multiplicity [20]. The spin multiplicity M is given by the equation $M = (2S + 1)$ [15], where S is the total spin of the system. For a paired orbital, the electrons have spins$+\frac{1}{2}$ and $-\frac{1}{2}$, and then the total spin equals zero. The spin multiplicity M becomes 1 and this is referred to as "singlet state". Unpaired electrons have the same spin orientation, i.e. $+\frac{1}{2}$ and $+\frac{1}{2}$, such that the total spin equals one, spin multiplicity equals three, and consequently this is referred to as "triplet state".

Fluorescence is a form of luminescence, which is the emission of light from any substances stimulated by absorption of incident electromagnetic radiation. There are two different kinds of luminescene; fluorescence and phosphoresence. Fluorescence corresponds to "singlet state", meaning that the excited orbital is paired (by opposite spin) to the second electron in the ground-state orbital, so that electrons are spin allowed to return to the ground state. Then the transitions of electrons to the ground state occurs rapidly, and by emission of a photons. While fluorescence corresponds to "single state", phosphorescence corresponds to "triplet state".

As mentioned, there are two categories of luminescence - fluorescence and phosphorescence - where the difference lies in the nature of the excited state. In fluorescence the excited orbital is paired to the second electron in the ground-state orbital, so that electrons are spin allowed to return to the ground state. Consequently the transition rate of electrons to the ground state is lower for phosphores than for fluorophores.

The phenomena of luminescence, including fluorescence and phosphorescence, can be better understood by introducing the Jablonski diagram. One form of such a Jablonski diagram is illustrated in Figure 2.1, made on a basis of a figure from [17]. The singlet ground, first and second electronic states are depicted by S_0, S_1 and S_2, while the first triplet state is depicted by T_1. It can be seen that the emission from S_1 and T_1 to S_0 is labeled fluorescence and phosphorescence respectively in the figure, which makes sense when remembering that singlet and triple state correspond to fluorescence and phosphorescence respectively.

Transitions between the states are illustrated by vertical arrows in Figure 2.1. The arrows are colored with respect to where their wavelength find place in the electromagnetic spectrum. The excitation light is for instance colored purple and blue, which corresponds to high energetic photons to the left of electromagnetic spectrum. In order for an incoming photon to be absorbed, the energy of the photon, defined as E = hf = hc/lambda, has be larger than or equal to the energy gap between the lowest level of $S_1$and the lowest level of S_2.

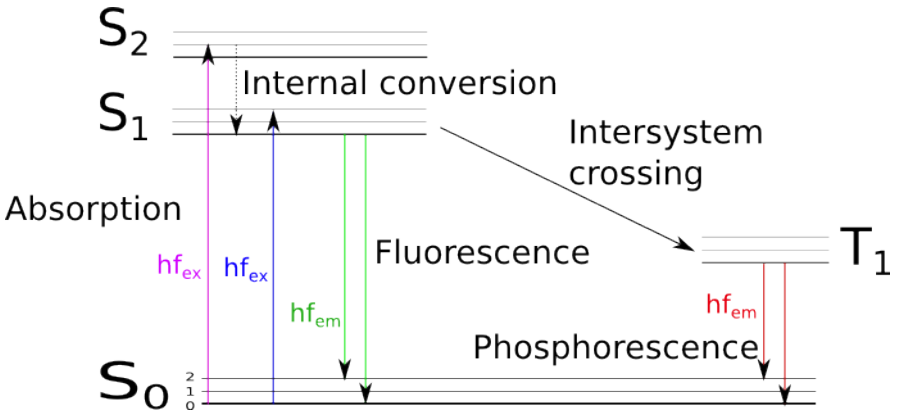


Figure 2.1: One example of a Jablonski diagram based on a Figure 1 from [17].

There can exist several numbers of vibrational energy levels, labeled 0, 1, 2, etc. The fluorophore is usually excited to some of the higher vibrational states of S_1(i.e. not the lowest level of S_1), and then rapidly relaxed to the lowest vibrational energy state (0-level) of S_1. The fluorophore can in some cases drop down to energy level S_0 before it relaxes permanently to the zero level of S_1. However, the average relaxing time to 0-level of S_1 is much shorter than the average time before fluorescence occurs. Then, relaxation completes before the emission. Consequently, the fluorescence is typically the decay from the lowest level of S_1 to S_0, which means that the wavelength of fluorescence is dependent on the energy difference between S_0 and S_1, rather than the energy of the excitation photons.

2.1.1 Quantum yield

Figure 2.2 is a simplification of the Jablonski diagram from Figure 2.1 used to illustrate quantum yield and lifetime. The quantum yield, Q, is defined as:

$$Q = \frac{\#photons emitted}{\#photons absorbed} \tag{2.1}$$

i.e. the ratio of between number of photons emitted the number absorbed [17].

There are two depopulation rate constants of the excited state S_1 in figure Figure 2.2, labeled Γ and κ_{nr}. Γ is the the radiative rate and k_{nr} is the non-radiative decay from S_1 to S_0. The total decay rate, $\Gamma + k_{nr}$, will necessary equals the rate of absorbed photons, such that the quantum yield can be written as: $Q = \frac{\Gamma}{\Gamma + k_{nr}}$.

The average time between when a molecule is excited and when it returns to the ground state, is defined as the lifetime, τ. It equals the reciprocal of the total decay rate and is given by: $\tau = \frac{1}{\Gamma + k_{nr}}$. Typical fluorescence lifetimes are near 10 ns [17].

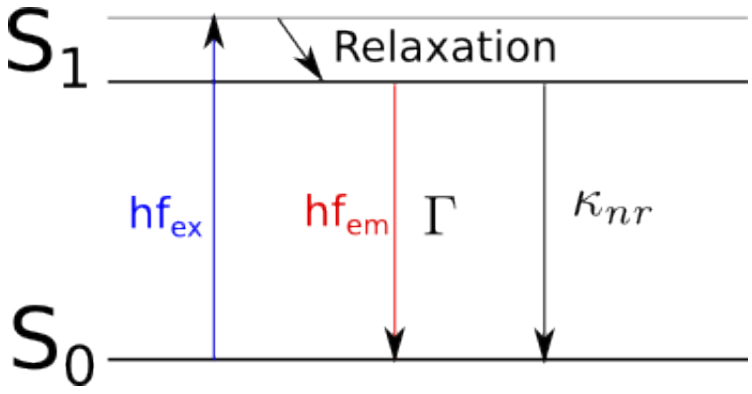


Figure 2.2: Jablonski diagram illustrating decay rates

2.1.2 Photobleaching

The fluorophore can be photochemically destroyed by the excitation light, such that it loses its ability to fluoresce. This phenomena is known as photobleaching. It happens when the excited molecule undergo chemical reactions with molecules in the surrounding environment, and produce irreversible covalent modifications [12]. The amount of fluorophores being photo bleached is dependent on molecular structure and the local environment.

Photobleaching therefore complicate the measurement of fluorophores, as they are eventually photobleached, and the concentration of non-destructed fluorophores is reduced. Theoretical simulations and experimental data of [18] show that photobleaching of fluorescein isn't in general, a single-exponential process. (The single-expontial process is a special case of photobleaching where reactions between molecular oxygen and triplet dye dominates [18].)

2.2 Radiometric quantities

In order to describe the radiative transport, the defintions of the most important radiometric quantities will be given in this section. The definitions are obtained from [16].

2.2.1 Radiant power, P , Φ

The radiant power is defined as the power emitted, transferred or received as radiation. The symbol may be both P and Φ, while the unit is W.

2.2.2 Radiant intensity, I

In a given direction from source, the radiant intensity is defined by the radiant power dP leaving the source in an element of solid angle dΩ containing this direction, divided by that element of solid angle:

$$I = \frac{d\Phi}{d\Omega}$$

The unit of radiant intesity is watt per steradians, i.e. W/sr.

2.2.3 Radiance L

In order to define the radiance, we start out with a point on a surface element dA, illustrated in Figure 2.3. Given this point, and a solid angle $d\Omega$ containing a given direction θ relative to a line perpendicular to dA, the radiance L is defined as the radiant intensity dI in the direction of θ from dA, divided by the area of the orthogonal projection of this element on a plane perpendicular to the given direction:

$$L = \frac{dI}{dA\, cos(\theta)} = \frac{d^2\Phi}{dA\, d\Omega\, cos(\theta)} \tag{2.2}$$

The unit of radiance is $\frac{W}{sr \cdot m^2}$.

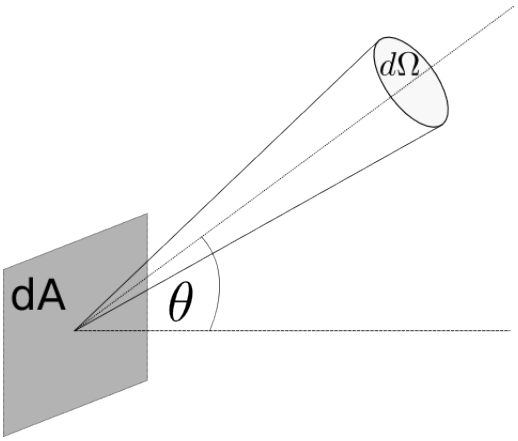


Figure 2.3: Definition of radiance. The figure is based on Figure 1.12 from [**?**]

2.2.4 Irradiance, E

The irradiance is given by the radiant intensity incident on an surface element divided by the area of the surface element. The definition includes all radiant power, and hence the radiation must be integrated over the hemisphere corresponding to the surface element dA:

$$E = \int_{2\pi} L(s) cos(\theta) d\Omega$$

where θ is the angle between the different directions of incident radiant intensity and the normal to the surface at the given point.

2.2.5 Fluence rate, φ

At a given point in space, the fluence rate, φ, is defined as the radiant intensity incident on a small sphere with the point centered, divided by the cross sectional are of the sphere. It can be expressed by integrating the radiance, taken over the 4π solid angle:

$$\varphi(\mathbf{r}) = \int_{4\pi} L(\Omega, \mathbf{r})\, d\Omega$$

The fluence rate is essentially the quotient of the total radiant power passing through the surface of an infinitely small sphere. The unit of fluence rate is $\frac{W}{m^2}$.

2.3 Different properties/techniques/definitions

2.3.1 Escape function

As explained in section 2.1 a fluophore might emit a fluorescence photon, after absorption of a photon of the excitation light. The fluophore might reach the surface of the material and escape. The escape function is defined in [20] as the surface distribution of fluorescence photons as a function of radial position r from a point source of fluorescence at depth z and radial position r=0 within a tissue of thickness D. An infinitely thick tissue sample is assumed, i.e. $D \to \infty$. The escape function is labeled $G_L(r, z)$.

2.3.2 Beers law

In an absorbing-only medium with absorption coefficient μ_a, light attenuates as it propagates according to the following equation [16]:

$$\frac{dI}{I} = -\mu_a\, dx, \tag{2.3}$$

where I denotes the light intensity and x denotes the distance along the light propagation direction. The interpretation of Equation 2.3 is that the percentage of light absorbed in the interval $(x,\ x + dx)$, i.e. $\frac{dI}{I}$, is proportional to the product of μ_a and dx. As x increases, the intensity I decreases, which explains the negative sign of Equation 2.3.

By integration of Equation 2.3 and defining the transmittance $T(x) = \frac{I(x)}{I_0}$, where I_0 is the light intensity at $x = 0$, we get Beers law, defined by::

$$T(x) = \frac{I(x)}{I_0} = \exp(-\mu_a\, x) \tag{2.4}$$

where $T(x)$ represents the probability of the survival of a photon after travelling a distance x. Equation 2.4 gives the following expression for μ_a as a function of the transmittance $T(x)$ and propagation distance x:

$$\mu_a = -\frac{\ln(T(x))}{x} \tag{2.5}$$

The scattering coefficient, μ_s, is defined as the probability of photon scattering in a medium per unit path length [16].

2.3.3 Reflectance

The reflectivety of a surface is the ratio between the incoming light and the reflected light. Here, we distinguish between two kinds of reflection: Diffuse reflection and specular reflection. The information given in this section is taken from [7], if no other reference is given.

Diffuse reflectance

Diffuse reflectance is defined as light scattered back into several different directions relative to incident angle of the incoming light. Perfect diffuse reflectance occurs when there are no preferable angle of reflection, i.e. the light is scattered equally in all directions. A perfect diffuse reflective surface is often called a **Lambertian surface**. Such surfaces can be labeled mathematically by **Lambert's law** , which states that the diffuse reflectance, R_d, is proportional to $\cos(\theta)$. I.e.:

$$R_d \propto \cos(\theta) \tag{2.6}$$

Specular reflectance and refraction

At a boundary between two media, with refractive indices n_1 and n_2, light can be both reflected and transmitted. A perfect specular reflector reflects light in a unique angle relative to the angle of incident light. The light fulfilles the law of reflection, which states that the angle of incidence equals the angle of reflection [22]. The refracted ray obeys Snell's law, which states that the angle of refraction θ_2 is related to the angle of incidence θ_1 by:

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2) \tag{2.7}$$

when the refracted ray lies in the plane of incidence. If $n_1 > n_2$, there is an angle of incidence, the so-called critical angle, which above gives total internal reflection. This angle, labeled θ_c, is given by $\theta_c = \sin^{-1}(n_2/n_1)$.

The reflectivity between media n_1 and n_2 is given by **Fresnel equations**. For unpolarized light with angle of incidence θ_1 and angle of refraction θ_3, the Fresnel reflection coefficient $R = R(\theta_1, \theta_3)$, is given by [22]:

$$R(\theta_1, \theta_3) = \begin{cases} R_0 = \frac{(n_1-n_2)^2}{(n_1+n_2)^2} & \theta_1 = 0 \\ \frac{1}{2}\left[\frac{\sin^2(\theta_1-\theta_3)}{\sin^2(\theta_1+\theta_3)} + \frac{\tan^2(\theta_1-\theta_3)}{\tan^2(\theta_1+\theta_3)}\right] & 0 < \theta_1 < \theta_c \\ 1 & \theta_1 \geq \theta_c = \sin^{-1}(n_2/n_1) \end{cases} \tag{2.8}$$

2.4 Spectral intrinsic fluorescence coefficient

The spectral intrinsic fluorescence coefficient $\beta\ \left[\mathrm{cm}^{-1}\right]$ is defined as in [11], i.e:

$$\beta(\lambda_{ex}, \lambda_{em}) = \mu_{a,fl}(\lambda_{ex})\ q(\lambda_{em}) \tag{2.9}$$

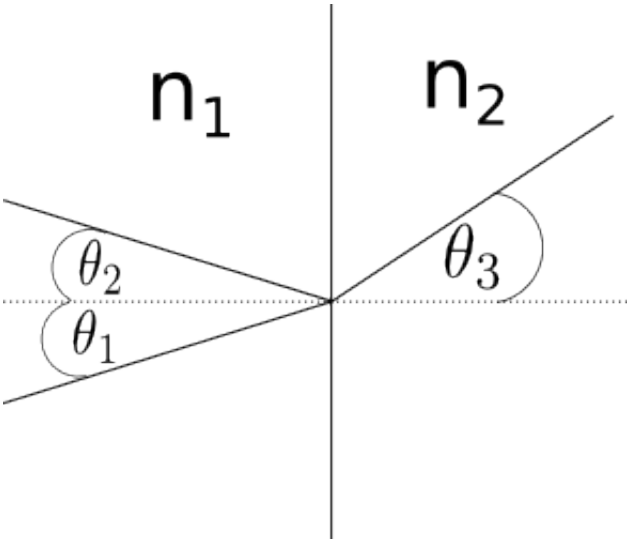


Figure 2.4: Illustration of angle of incidence, angle of reflection and angle of refraction. The angle of incidence equals the angle of reflection, while the relation between angle of incidence and refraction is given by Snell's law

$\mu_{\mathrm{a,fl}}(\lambda_{\mathrm{ex}})$ is the fluorophore absorption coefficient at excitation wavelength λ_{ex}, while $\mathrm{q}(\lambda_{\mathrm{em}})$ is the spectral fluorescence quantum yield, which is a differential form of the fluorescence quantum yield defined in Equation 2.1. The fluorescence quantum yield, Q, expressed in terms of the spectral fluorescence quantum yield is then:

$$Q = \int_{\Delta\lambda_{em}} q(\lambda_{em})\ d\lambda_{em} \tag{2.10}$$

where $\triangle\lambda_{em}$ is the domain of fluorescence emission.

2.4.1 Scattering media

A semi-infinite, homogeneous turbid medium and a normally incident, finite excitation beam is assumed in the following derivations. The product $\beta\,\varphi$, where β is the spectral intrinsic fluorescence coefficient defined in Equation 2.9 and φ is the fluence rate of excitation light, represents the local rate of fluorescence generation within tissue. Multiplying by the escape function G_L defined in section 2.4, we get $G_L\,\beta\,\varphi$, which represents the fluorescence photon distribution at the surface of the sample. One-dimensional quantities, G_P and ϕ_P, that describe the escape function and distribution of excitation light power respectively, as function of depth only, were proposed and verified to have accuracy comparable to Monte Carlo simulations in [6]. The fluorescence escape power is expressed as [11]:

$$F(\lambda_{ex},\lambda_{em}) = \beta(\lambda_{ex},\lambda_{em}) \int_0^\infty dz' \left[\Phi_P\left(z';\lambda_{ex}\right)\ G_L\left(z';\lambda_{em}\right)\right] \tag{2.11}$$

where

$$\Phi_P\left(z';\lambda_{ex}\right) = C_1\ (\lambda_{ex})\ \exp\left(-k_1(\lambda_{ex})z'/\delta(\lambda_{ex})\right) - C_2\ (\lambda_{ex})\ \exp\left(-k_2(\lambda_{ex})z'/\delta(\lambda_{ex})\right) \quad (2.12)$$

$$G_P\left(z';\lambda_{em}\right) C_3\ (\lambda_{em})\ \exp\left(-k_3(\lambda_{em})z'/\delta(\lambda_{em})\right) \quad (2.13)$$

These expressions (Equation 2.12 and Equation 2.13) is developed by Monte Carlo simulations in [6] and so is their coefficients for phantom-air interface, which is given in Table 2.1. δ is the measure of effective penetration depth, which describes the exponential attenuation of light as a functio of depth in one-dimensional geometry.

Parameter	Expression
C_1	$3.04 + 4.90R_d - 2.06\exp(-21.1R_d)$
k_1	$1 - (1 - 1/\sqrt{3})\exp(-18.9R)$
C_2	$2.04 - 1.33R_d - 2.04\exp(-21.1R_d)$
k_2	$1.59\exp(3.36R_d)$
C_3	$0.32 + 0.72R_d - 0.16\exp(-9.11R_d)$
k_3	$1 - 0.30\exp(-6.12R_d)$

Table 2.1: Coefficients of Equation 2.12 and Equation 2.13 as a function of diffuse reflectance R_d

Equation 2.11 written in terms of Equation 2.12 and Equation 2.13 and integrated gives the following expression for fluorescence escape:

$$f\left(\lambda_{ex},\lambda_{em}\right) = \beta\left(\lambda_{ex},\lambda_{em}\right)\ p_{0,tot}\left(\lambda_{ex}\right)\ X\left(\lambda_{ex},\lambda_{em}\right) \quad (2.14)$$

where $p_{tot}\left(\lambda_{ex}\right)$ is the incident excitation beam ans X_{1D} is the so-called one-dimensional path-length factor. It is given by:

$$X_{1D}\left(\lambda_{ex},\lambda_{em}\right) = \frac{C_1\left(\lambda_{ex}\right)C_2\left(\lambda_{em}\right)}{k_1(\lambda_{ex})/\delta(\lambda_{ex}) + k_3(\lambda_{em})/\delta(\lambda_{em})} - \frac{C_2\left(\lambda_{ex}\right)C_3\left(\lambda_{em}\right)}{k_2(\lambda_{ex})/\delta(\lambda_{ex}) + k_3(\lambda_{em})/\delta(\lambda_{em})} \quad (2.15)$$

When scattering is much larger than absorption, the effective penetration depth δ can by expressed by the following equation [11] :

$$\delta = \frac{1}{\left\{3\mu_a\left[\mu_a + \mu_s\left(1-g\right)\right]\right\}^{1/2}} \quad (2.16)$$

In terms of an incident excitation beam with power, we get following expression for fluorescence escape:

$$f\left(\lambda_{ex},\lambda_{em}\right) = \beta\left(\lambda_{ex},\lambda_{em}\right)\ p_{0,tot}\left(\lambda_{ex}\right)\ X_{1D}\left(\lambda_{ex},\lambda_{em}\right) \quad (2.17)$$

such that spectral intrinsic fluorescence becomes:

$$\beta\left(\lambda_{ex},\lambda_{em}\right) = \frac{f\left(\lambda_{ex},\lambda_{em}\right)\ X_{1D}\left(\lambda_{ex},\lambda_{em}\right)}{p_{0,tot}\left(\lambda_{ex}\right)} \quad (2.18)$$

2.4.2 Non-scattering media

An expression for the fluorescence escape from a non scattering media, can be find by considering the following three characteristics of light transport [11]: 1) The exponential attenuation of excitation and emission light; 2) the isotropic nature of emission; and 3) reflections at boundaries where there is a refractive index mismatch. Based on these considerations, a rigorous expression for fluorescence escape expressed in radiant intensity [Wsr^{-1}] as a function of exit angle θ_{t}, was derived in [10]:

$$I\left(\theta_t;\lambda_{ex},\lambda_{em}\right)=\frac{\beta\left(\lambda_{ex},\lambda_{em}\right)\ p_0\left(\lambda_{ex}\right)}{4\pi}\ \frac{1-\exp\left[-\left(\mu_a\left(\lambda_{ex}\right)+{}^{\mu_a(\lambda_{em})}/_{\cos(\theta_i)}\right)L\right]}{\mu_a\left(\lambda_{ex}\right)+{}^{\mu_a(\lambda_{em})}/_{\cos(\theta_i)}} \times\left(1-R_0\right)\left(1-R\right)\frac{n_t^2\,cos\left(\theta_t\right)}{n_i^2\,cos\left(\theta_i\right)} \quad (2.19)$$

$\beta\left(\lambda_{\mathrm{ex}},\lambda_{\mathrm{em}}\right)$ is defined in Equation 2.9, $p_0\left(\lambda_{ex}\right)$ is the incident power of excitation light, $\mu_{\mathrm{a}}\left(\lambda_{\mathrm{ex}}\right)$ and $\mu_{\mathrm{a}}\left(\lambda_{\mathrm{em}}\right)$ are the absorption coefficients of the slab medium at the excitation and emission wavelengths, respectively. The exit angles of fluorescence escape θ_{i} and θ_t, are related by Snell's law, where θ_{i} is the escape within the slab, and θ_{t} is outside the slab. L is the thickness of the slab medium, and $\mathrm{n_i}$ and $\mathrm{n_t}$ are the refractive indices of the slab and the air, respectively. The reflectance coefficients, $\mathrm{R_0}$ and R of Equation 2.19, are the Fresnel reflection coefficients for unpolarized light given in Equation 2.8.

There are several assumptions in the derivation of Equation 2.19, which now will be presented. The excitation light is assumed to be normally incident beam with a finite diameter, while the slab is assumed to be infinitely wide. An index-matched bottom boundary is assumed, which eliminates the consideration of reflected excitation light at the bottom boundary. The refractive index of the slab, $\mathrm{n_i}$, is assumed higher than air, $\mathrm{n_t}=1$. The fluorescence light reaching the top surface may then experience total internal reflectance.

If the slab medium is optically dilute, Equation 2.19 may be simplified. The condition for an optically dilute solution is [11]: $\left(\mu_a\left(\lambda_{ex}\right)+{}^{\mu_a(\lambda_{em})}/_{\cos(\theta_i)}\right)l\ll 1$. The exponential term of Equation 2.19 may then be expanded in a Taylor series with only the first two terms retained. I.e. $\exp\left[-\left(\mu_a\left(\lambda_{ex}\right)+{}^{\mu_a(\lambda_{em})}/_{\cos(\theta_i)}\right)L\right]\Longrightarrow 1-\left(\mu_a\left(\lambda_{ex}\right)+{}^{\mu_a(\lambda_{em})}/_{\cos(\theta_i)}\right)l$, and Equation 2.19 is simplified to:

$$I\left(\theta_t;\lambda_{ex},\lambda_{em}\right) = \frac{\beta\left(\lambda_{ex},\lambda_{em}\right)\ p_0\left(\lambda_{ex}\right)}{4\pi}\,l\left(1-R_0\right)\left(1-R\right)\frac{n_t^2\,cos\left(\theta_t\right)}{n_i^2\,cos\left(\theta_i\right)} \quad (2.20)$$

Equation 2.20 is now rewritten to:

$$I\left(\theta_t;\lambda_{ex},\lambda_{em}\right)=\beta\left(\lambda_{ex},\lambda_{em}\right)\ p_0\left(\lambda_{ex}\right)\ \chi_{dil} \quad (2.21)$$

such that χ_{dil} becomes:

$$\chi_{dil}=\frac{1}{4\pi}\,l\left(1-R_0\right)\left(1-R\right)\frac{n_t^2\,cos\left(\theta_t\right)}{n_i^2\,cos\left(\theta_i\right)} \quad (2.22)$$

2.5 Calibration

2.5.1 Definition of α

The emitted power from a surface area dA may be directed in several different directions, and consequently, only a (small) fraction of total emitted power $\Phi_{tot}(\lambda)$ from a surface element dA will be captured by the detector which covers a solid angle of dΩ. The captured power expressed in W will then be transformed into a new quantity $\Phi_{det}(\lambda)$ which represent the captured power by a camera which covers a small solid angle dΩ expressed in a detector unit, from now on referred to as *digitalnumber*. α is defined such that:

$$\Phi_{tot}(\lambda) = \alpha\,\Phi_{det}(\lambda) \tag{2.23}$$

Since $\Phi_{\mathrm{tot}}(\lambda)$ has the unit W and $\Phi_{\mathrm{det}}(\lambda)$ has the unit *digitalnumber*. The unit of α becomes $^{W}/_{digitalnumber}$.

2.5.2 Derivation of expressions for α (scattering and non-scattering media)

The camera used in the measurements will detect radiant power in a finite solid angle from an area element dA of the sample surface. According to Equation 2.2, the radiant power in a solid angle $d\Omega$ from a surface area dA is given by:

$$d^2\Phi = L\,dA\,\cos(\theta)\,d\Omega \tag{2.24}$$

where θ is the angle between the surface normal of dA and the direction given by $d\Omega$ (see Figure 2.3).

The resolution of the camera corresponds to a resolution of the the sample which is imaged. An area element of the sample which corresponds to one pixel is labeled $\triangle S$, while the solid angle covered by the detector is denoted $\triangle\Omega$. In the case of a spectrometer, and not a camera, $\triangle S$ denotes the area covered the spectrometer probe. Figure 2.5 illustrated the symbols just described. The radiant power from $\triangle S$ within a solid angle $\triangle\Omega$ is denoted p. p is obtained by a) integrating the radiant intensity from an area element dA over the solid angle $\triangle\Omega$, and b) integrating over the surface area $\triangle S$. That is:

$$p = \int_{\triangle S} dA \int_{\triangle\Omega} d\Omega\, L\cos\theta \tag{2.25}$$

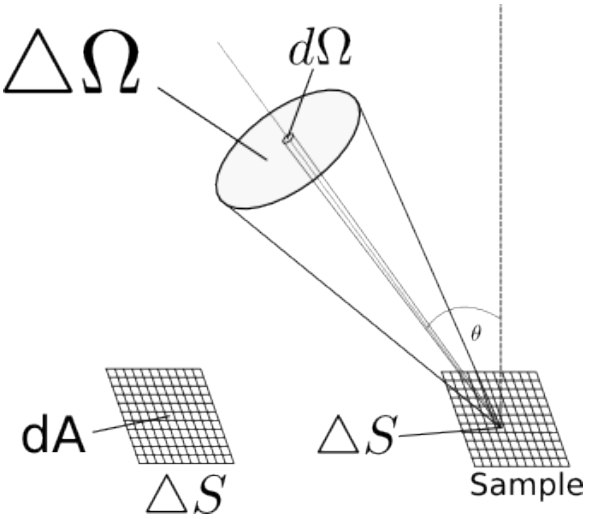


Figure 2.5: $\triangle S$ is the area element of the sample corresponding to a pixel in the image, and is composed of infinitesimal small area elements dA. $\triangle S$ is the solid angle covered by the detector.

The size of solid angle covered by the detector, $\triangle\Omega$, is small compared to the distance between the camera and the surface element dA. Then it's reasonable to assume a constant angle θ over the solid angle $\Delta\Omega$. Further, dA is assumed small compared to the separation distance between camera and sample, so that θ is constant over $\triangle S$. We then get:

$$p = \triangle S\, L\, cos(\theta)\, \triangle\Omega \tag{2.26}$$

Where $\triangle S\, L\, cos(\theta)$ is the radiant intensity, labeled i, emitted from $\triangle S$, with unit $^{W}/_{\mathrm{sr}}$.

The camera consist basically of a detector and an objective, while there isn't any objective in front of the detector of the spectrometer probe. The captured power $p\,(\lambda)$ expressed in W by the detector is transformed into into the readable signal $p_{det}\,(\lambda)$ in *digitalnumbers* by multiplication of a response function $D\,(\lambda)$, i.e. $p_{det}\,(\lambda) = D\,(\lambda)\; p\,(\lambda)$. The unit of $D\,(\lambda)$ is $^{digitalnumber}/_{W}$. There is an objective in front of the detector of the camera with transmission function $T\,(\lambda)$. Consequently, the expression for $p_{det}\,(\lambda)$ in the case of camera becomes $p_{det} = T\,(\lambda)\; D\,(\lambda)\; p\,(\lambda)$. $C\,(\lambda)$ is now defined such that:

$$p_{det} = C\,(\lambda)\; p\,(\lambda) \tag{2.27}$$

Consequently, $C\,(\lambda) = D\,(\lambda)$ for the spectrometer, and $C\,(\lambda) = T\,(\lambda)\; D\,(\lambda)$ for the camera. T is dimensionless, such that the unit of $D\,(\lambda)$ is $^{digitalnumber}/_{W}$.

Equation 2.26 and Equation 2.27 gives the following expression for p_{det}:

$$p_{det}\,(\lambda) = C(\lambda)\, i\,(\lambda)\; \triangle\Omega \tag{2.28}$$

In order du find an expression for α, we need an expression for p_{tot}.

The total radiant power expressed in W emitted from $\triangle S$, i.e. $p_{tot}(\lambda)$, is obtained by integration of the radiant intensity I (from Equation 2.26) over the hemisphere centered at $\triangle S$:

$$p_{tot}(\lambda) = \int_{hemisphere} i \, d\Omega = \int_{hemisphere} L \, \Delta S \, \cos(\Phi) \, d\Omega \tag{2.29}$$

As explained in subsection 2.3.3, the radiance L of a Lambertian surface is independent of exit angle θ. We assume the surface to be Lambertian, and Equation 2.29 becomes:

$$p_{tot}(\lambda) = L \triangle S \int_{hemisphere} \cos(\theta) \, d\Omega = L \triangle S \, \pi \tag{2.30}$$

By rearrangement and multiplication by π in Equation 2.26, the equation $L \triangle S \, \pi = \frac{\pi}{cos(\theta) \triangle \Omega} \, p(\lambda)$ is obtained. Combining this equation with Equation 2.30, we get $p_{tot}(\lambda) = \frac{\pi}{cos(\theta) \triangle \Omega} \, p(\lambda)$. Using Equation 2.27, this expression is obtain for $p_{tot}(\lambda)$:

$$p_{tot}(\lambda) = \frac{p_{det}(\lambda) \, \pi}{C(\lambda) \, \cos(\theta) \, \triangle \Omega} \tag{2.31}$$

When scatteres are introduced into a solution, Lambertian reflectance can be assumed. Consequently the α-factor may be labeled α_{scat} when a Lambertian surface is assumed. Equation 2.23 and Equation 2.31 give:

$$\alpha_{scat} = \frac{\pi}{C(\lambda) \, \cos(\theta) \, \triangle \Omega} \tag{2.32}$$

α_{scat} contains now all system and setup parameters.

A Lambertian surface surface isn't assumed anymore, but it's assumed that the radiant isn't i is known. Equation 2.28 gives the following expression for i:

$$i(\lambda) = \frac{p_{det}(\lambda)}{C(\lambda) \, \triangle \Omega} \tag{2.33}$$

Multiplying Equation 2.33 by $1 = \frac{\pi \cos(\theta)}{\pi \cos(\theta)}$, gives:

$$i(\lambda) = \frac{\pi}{C(\lambda) \, \triangle \Omega \, \cos(\theta)} \frac{\cos(\theta)}{\pi} \, p_{det}(\lambda) = \alpha_{scat} \frac{\cos(\theta)}{\pi} \, p_{det}(\lambda) \tag{2.34}$$

where the expression for α_{scat} is found in Equation 2.32. When no scatteres are introduced into a solution, the solution may be described as a diluted solution, and consequently the factor $\alpha_{scat} \frac{\cos(\theta)}{\pi}$ may be labeled α_{dil}. However, this notation is omitted in the rest of this report.

2.5.3 Determination of spectral intrinsic fluorescence coefficient in scattering media

As will be described in chapter 3, a gray standard with Lambertian surface is used for the calibration of the system. The reflectivety of the gray standard is labeled $R_{gs}(\lambda)$. The

reflected power by the gray standard when illuminated by $p_{0,tot}(\lambda)$ is denoted $p_{0,tot,gs}(\lambda)$, i.e. $p_{0,tot,gs}(\lambda) = R_{gs}(\lambda)\ p_{0,tot}(\lambda)$.

Equation 2.14 is now rewritten to:

$$f_{tot}(\lambda_{ex}, \lambda_{em}) = \beta(\lambda_{ex}, \lambda_{em})\ {}^{p_{0,tot,gs}(\lambda_{ex})}/_{R_{gs}(\lambda_{ex})}\ X_{1D} \tag{2.35}$$

The emission of fluorescence is isotropic, such that a Lambertian surface can be assumed. Then we have $\mathrm{f_{tot}}(\lambda_{\mathrm{em}}) = \alpha_{scat}(\lambda_{em})\ f_{det}(\lambda_{em})$. Gray standards have Lambertian reflectance as wel, i.e. $\mathrm{p_{0,tot,gs}}(\lambda_{\mathrm{ex}}) = \alpha(\lambda_{\mathrm{ex}})\ \mathrm{p_{det,gs}}(\lambda_{ex})$. We get:

$$\alpha(\lambda_{em})\ f_{det}(\lambda_{em}) = \beta(\lambda_{ex}, \lambda_{em})\ {}^{\alpha(\lambda_{ex})\, p_{det,gs}(\lambda_{ex})}/_{R_{gs}(\lambda_{ex})}\ X_{1D} \tag{2.36}$$

and the expression for $\beta(\lambda_{ex}, \lambda_{em})$ becomes:

$$\beta(\lambda_{ex}, \lambda_{em}) = \frac{\alpha(\lambda_{em})\ f_{det}(\lambda_{em})\ R_{gs}(\lambda_{ex})}{\alpha(\lambda_{ex})\ p_{det,gs}(\lambda_{ex})\ X_{1D}} \tag{2.37}$$

2.5.4 Determination of spectral intrinsic fluorescence coefficient in non-scattering media

The radiant intensity of the fluorescence escape from a dilute solution is given by Equation 2.21. As a starting point and these equations:

$$p_{tot}(\lambda_{ex}) = {}^{\alpha(\lambda_{ex})\, p(\lambda_{ex})_{0,det,gs}}/_{R_{gs}(\lambda_{ex})} \tag{2.38}$$

$$I(\theta_t; \lambda_{ex}, \lambda_{em}) = \alpha(\lambda_{em})\ \frac{\cos(\theta)}{\pi}\ f_{det}(\lambda_{em}) \tag{2.39}$$

We get:

$$\alpha(\lambda_{em})\ \frac{\cos(\theta)}{\pi}\ f_{det}(\lambda_{em}) = \beta(\lambda_{ex}, \lambda_{em})\ {}^{\alpha(\lambda_{ex})\, p(\lambda_{ex})_{0,det,gs}}/_{R_{gs}(\lambda_{ex})\ \chi_{dil}} \tag{2.40}$$

The expression for the spectral intrinsic fluorescence coefficient becomes:

$$\beta(\lambda_{ex}, \lambda_{em}) = \frac{\alpha(\lambda_{em})}{\alpha(\lambda_{ex})}\ \frac{\cos(\theta)}{\pi}\ \frac{f_{det}(\lambda_{em})}{{}^{p(\lambda_{ex})_{0,det,gs}}/_{R_{gs}\ \chi_{dil}}} \tag{2.41}$$

2.5.5 Calibration of α_{scat}

Spectrometer

The calibration factor in the case of the spectrometer measurements is assumed to be independent on wavelength, i.e. $\alpha_{scat}(\lambda_{ex}) = \alpha_{scat}(\lambda_{em})$. Then Equation 2.41 is simplified to

$$\beta(\lambda_{ex}, \lambda_{em}) = \frac{4\pi\ f_{det}(\lambda_{em})\ n_i^2\ cos(\theta_i)\ R_{gs}}{p_{det,gs}(\lambda_{ex})\ l\,(1 - R_0)\,(1 - R)\,n_t^2\,\pi} \tag{2.42}$$

for non scattering media, while Equation 2.37 is simplified to

$$\beta\left(\lambda_{ex},\lambda_{em}\right)=\frac{f_{det}\left(\lambda_{em}\right)\ R_{gs}}{p_{0,tot}\left(\lambda_{ex}\right)\ X_{1D}} \tag{2.43}$$

for scattering media.

Camera

The image acquired by the camera consists of many pixels, each corresponding to an area element $\triangle S$ of the pixel. In the calibration of α_{scat} for the camera, the following equation is assumed:

$$\frac{\mathrm{p_{tot}}\left(\lambda\right)}{\sum_{MxN}p_{tot}\left(\lambda\right)}=\frac{p_{det}\left(\lambda\right)}{\sum_{MxN}p_{det}\left(\lambda\right)} \tag{2.44}$$

where $\mathrm{p_{tot}}\left(\lambda\right)$ means total power emitted from a "pixel" $\triangle S$, $\sum_{MxN}p_{tot}\left(\lambda\right)$ means total power emitted from sample (all area elements $\triangle S$), $p_{det}\left(\lambda\right)$ means detected power in detector units from one area element $\triangle S$ and $\sum_{MxN}p_{det}\left(\lambda\right)$ means the sum of all detected power from the sample. What Equation 2.44 tells, is that the measured ratio by camera of power from one pixel to the sum of power for all pixels equals the true ratio.

Spectral values are not obtainable by camera measurements. Instead the received power by camera in *digitalnumbers*, P_{det}, is expressed as $P_{det}=\int p_{det}\left(\lambda\right)\ d\lambda$.

For narrow narrow bands, one can assume $P_{det}=p_{det}\left(\lambda\right)$, such that Equation 2.44 can be written as $\frac{\mathrm{P_{tot}}}{\sum_{MxN}P_{tot}}=\frac{P_{det}}{\sum_{MxN}P_{det}}$, where $P_{tot}=\int p_{tot}\left(\lambda\right)\ d\lambda$. A rearranged gives:

$$\mathrm{P_{tot}[W]}=\frac{\sum_{MxN}P_{tot}[W]}{\sum_{MxN}P_{det}[digitalnumber]}\ P_{det}[digitalnumber] \tag{2.45}$$

By comparing Equation 2.45 and Equation 2.23, we get the following expression for α_{scat}:

$$\alpha=\frac{\sum_{MxN}P_{tot}[W]}{\sum_{MxN}P_{det}[digitalnumber]} \tag{2.46}$$

When gray standard is illuminated by excitation light, and the whole beam is imaged, the total incident power $\mathrm{P_{incident}}$ on all pixels equals $\sum_{MxN}{}^{P_{tot}[W]}/_{R_{gs}}$. Consequently, $\sum_{\mathrm{MxN}}\mathrm{P_{tot}[W]}=\mathrm{P_{incident}}\left[W\right]\ R_{gs}$. We get the following expression for α:

$$\alpha=\frac{P_{incident}\left[W\right]\ R_{gs}}{\sum_{MxN}P_{det}[digitalnumber]} \tag{2.47}$$

As indicated in Equation 2.32, α is dependent on wavelength, and has to be calibrated for each wavelength.

2.5.6 Determination of fluorescence quantum yield by camera measurements

Obtainable data from camera measurements are:

$$F_{det}=\int_{\triangle\lambda_{em}}f_{det}\left(\lambda_{em}\right)\ d\lambda_{em} \tag{2.48}$$

and:

$$P_{det,gs} = \int_{\triangle\lambda_{ex}} p_{det,gs}\left(\lambda_{ex}\right)\, d\lambda_{ex} \tag{2.49}$$

Now, an important approximation is done. The $\triangle\lambda_{ex}$ is assumed to be narrow enough such that $p_{det,gs}\left(\lambda_{ex}\right)$ is approximated to be monochromatic, i.e. it consist of one single wavelength. I.e. $P_{det,gs} \approx p_{det,gs}\left(\lambda_{ex}\right) \Rightarrow P_{det,gs} = P_{det,gs}\left(\lambda_{ex}\right)$.

The expressions for the spectral intrinsic beta coefficient, $\beta\left(\lambda_{ex}, \lambda_{em}\right)$, for the scattering and dilute solution are given in Equation 2.37 and Equation 2.41 respectively. B is now defined as $\beta\left(\lambda_{ex}, \lambda_{em}\right)$ integrated over the emission range, i.e. $B = \int_{\Delta\lambda_{em}} \beta\left(\lambda_{ex}, \lambda_{em}\right) d\lambda_{em}$ and referred to as

An expression for the fluorescence quantum yield is obtained by writing out B:

$$B = \int_{\Delta\lambda_{em}} \beta\left(\lambda_{ex}, \lambda_{em}\right) d\lambda_{em} = \int_{\Delta\lambda_{em}} \mu_{a,fl}\left(\lambda_{ex}\right)\, q\left(\lambda_{em}\right)\, d\lambda_{em} = \mu_{a,fl}\left(\lambda_{ex}\right) \int_{\Delta\lambda_{em}} q\left(\lambda_{em}\right)\, d\lambda_{em} = \mu_{a,fl}\left(\lambda_{ex}\right)\, Q \tag{2.50}$$

That is:

$$Q = \frac{1}{\mu_{a,fl}\left(\lambda_{ex}\right)} \int_{\Delta\lambda_{em}} \beta\left(\lambda_{ex}, \lambda_{em}\right)\, d\lambda_{em} = \frac{B}{\mu_{a,fl}\left(\lambda_{ex}\right)} \tag{2.51}$$

The fluorescence quantum yield is now given by Equation 2.51. We need an expression for the quantum yield Q as a function of the obtainable values $\mathrm{F_{det}}$ and $\mathrm{P_{det,gs}}$ from camera measurements, given in Equation 2.48 and Equation 2.49. The fluorescence quantum yield for scattering media is obtained by inserting the expression for $\beta\left(\lambda_{ex}, \lambda_{em}\right)$ inEquation 2.37 into Equation 2.51, and for non scattering media by inserting Equation 2.41 into Equation 2.51.

Non-scattering

By using the expression for $\beta\left(\lambda_{ex}, \lambda_{em}\right)$ from Equation 2.41 in Equation 2.51, we obtain:

$$Q = \frac{1}{\mu_{a,fl}\left(\lambda_{ex}\right)} \int_{\Delta\lambda_{em}} \frac{\alpha\left(\lambda_{em}\right)}{\alpha\left(\lambda_{ex}\right)} \frac{\cos\left(\theta\right)}{\pi} \frac{f_{det}\left(\lambda_{em}\right)}{p(\lambda_{ex})_{det,gs}/R_{gs}\, \chi_{dil}}\, d\lambda_{em} \tag{2.52}$$

Assumption: $\alpha\left(\lambda_{em}\right)$ and χ_{dil} (i.e. $\mathrm{n_i}$ and $\mathrm{n_t}$) constant through $\triangle\lambda_{em}$,

Which becomes:

$$Q = \frac{1}{\mu_{a,fl}\left(\lambda_{ex}\right)} \frac{\alpha\left(\lambda_{em}\right)}{\alpha\left(\lambda_{ex}\right)} \frac{\cos\left(\theta\right)}{\pi} \frac{1}{p(\lambda_{ex})_{det,gs}/R_{gs}\, \chi_{dil}} \int_{\Delta\lambda_{em}} f_{det}\left(\lambda_{em}\right)\, d\lambda_{em} \tag{2.53}$$

$p\left(\lambda_{ex}\right)_{det,gs} = P_{det,gs}\left(\lambda_{ex}\right)$ and $\int_{\Delta\lambda_{em}} f_{det}\left(\lambda_{em}\right)\, d\lambda_{em} = F_{det}$, where $P_{det,gs}\left(\lambda_{ex}\right)$ and F_{det} are the measured values by camera in digital numbers. Final expression for fluorescence quantum yield:

$$Q = \frac{1}{\mu_{a,fl}\left(\lambda_{ex}\right)} \frac{\alpha\left(\lambda_{em}\right)}{\alpha\left(\lambda_{ex}\right)} \frac{\cos\left(\theta\right)}{\pi} \frac{1}{\chi_{dil}} \frac{F_{det}}{P_{det,gs}(\lambda_{ex})/R_{gs}} \tag{2.54}$$

The expression for B becomes:

$$B = \frac{\alpha(\lambda_{em})}{\alpha(\lambda_{ex})} \frac{\cos(\theta)}{\pi} \frac{1}{\chi_{dil}} \frac{F_{det}}{P_{det,gs}(\lambda_{ex})/R_{gs}} \tag{2.55}$$

Scattering

By similar calculation and approximations as with the non-scattering case, we get the following final expression for fluorescence quantum yield in scattering media:

$$Q = \frac{1}{\mu_{a,fl}(\lambda_{ex})} \frac{\alpha(\lambda_{em})}{\alpha(\lambda_{ex})} \frac{1}{X_{1D}} \frac{F_{det}(\lambda_{em})}{P_{det,gs}(\lambda_{ex})/R_{gs}} \tag{2.56}$$

and B:

$$B = \frac{\alpha(\lambda_{em})}{\alpha(\lambda_{ex})} \frac{1}{X_{1D}} \frac{F_{det}(\lambda_{em})}{P_{det,gs}(\lambda_{ex})/R_{gs}} \tag{2.57}$$

2.5.7 Filter calibration and exposure-time correction fluorescence measurement

To block out reflected excitation light and let only fluorescence light entering the camera, a filter with transimssion function $t(\lambda)$ which is zero for excitation wavelengths λ_{ex} and non-zero for emission wavelengths λ_{em} is needed. In the derivations of the last section (subsection 2.5.6) an ideal emission filter which let all and nothing else than fluorescence power through, i.e. $t(\lambda) = 1$ in the range of emission light and $t(\lambda) = 0$ elsewhere, was assumed. However, the detected power by camera, P_{det}, is expressed as:

$$P_{det} = \int t(\lambda)\, p(\lambda)\, d\lambda \tag{2.58}$$

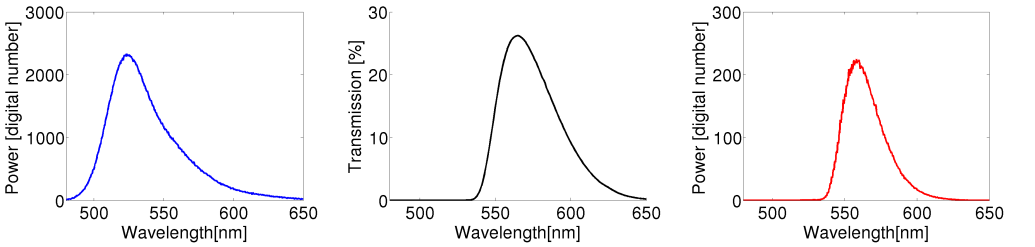
where $t(\lambda)$ is the transmission spectrum of a filter placed in front of the camera, and $p(\lambda)$ is the power entering the filter.

The ratio of power entering the filter to the power transmitted through the filter, labeled η, can be calibrated. An expression for η is:

$$\eta = \frac{\int t(\lambda)\, p(\lambda)\, d\lambda}{\int p(\lambda)\, d\lambda} \tag{2.59}$$

$p(\lambda)$ is here the spectrum of light leaving the sample illuminated by excitation light, and $t(\lambda)$ is the filter transmission spectrum. $p(\lambda)$ can be measured by a spectrometer, and there are also ways to measure filter transmission $t(\lambda)$ (see section 7.2). Figure 2.6a shows an arbitrary fluorescence spectrum, while a transmission spectrum of a filter is shown in 2.6b. Then, $t(\lambda)\, p(\lambda)$ is plotted in Figure 2.6c.

Due to small signal of fluorescence power, the exposure-time of camera may be increased in the measurement of fluorescence. If there is a linear relatiton between detected power and exposure-time, the fluorescence power can be corrected by a factor $\frac{t_c}{t_f}$,



(a) Fluorescence spectrum

(b) Filter spectrum

(c) Fluorescence spectrum in 2.6a multiplied by filter transmission spectrum in 2.6b

Figure 2.6

where t_c is the exposure-time of calibration measurements and t_f is the exposure-time of fluorescence measurements. The total emitted fluorescence power in the unit of watt becomes:

$$F[W] = \frac{F_{det}[digitalnumber]\,\alpha\,\frac{t_c}{t_f}}{\eta} \tag{2.60}$$

where $F_{det}[digitalnumber]$ is the power measured by camera in camera units, and α is the calibration factor described in subsection 2.5.1. That is whenever $\frac{t_c}{\eta\, t_f} \neq 1$, the measured fluorescence $F_{det}[digitalnumber]$ has to be corrected by multiplication of $\frac{t_c}{\eta\, t_f}$, i.e. $F_{det}[digitalnumber]$ becomes $\frac{t_c}{\eta\, t_f} F_{det}[digitalnumber]$.

3 Materials and methods

The camera consisted of an objective (Quartz lens 25 mm, F/2.8, UV25-0378, Near-far-field: '4'), a 20 mm extension tube and a CCD camera (Jai, CM 140 GE-UV). The components was assembled as showed in Figure 3.1, and this composition is from now only referred to as 'camera'.

Figure 3.1: The 'camera' consisting of an objective, 20 mm extension tube and CCD camera.

The camera data acquisition was done be the means of the computer software 'Fluor' customized for the measurements of this thesis. More information about this program is found in section 7.3. 15 text-files was stored per measurement, and then averaged into one text file, from now on referred to as the data. All data except 'fluorescence without filter' (explained in detail below) and corresponding dark current data were acquired with exposure time 6.8 ms. the rest of the data was acquired with exposure time 32 ms.

The area of sample/gray standard to be imaged was (about) $10x7.5\,mm^2$, while the number of pixels of each image was $1392x1040$.

The spectrometer data acquisition was done by a a spectrometer (Ocean Optics, ADC 1000) with a fiber and accompanying probe and a computer software (Oceanoptics, SpectraSuite.Ink).

The excitation light source consisted of a UV diode with peak wavelength 281 nm, driven by a laser diode driver (LCI, model 505). This light source is from now on only referred to as 'excitation light'. The effect of the laser diode driver was 40 mA for the camera measurements and 1.5 mA for the spectrometer measurements.

The emission light source consisted of a white light source (Scott, KL 1500-T)

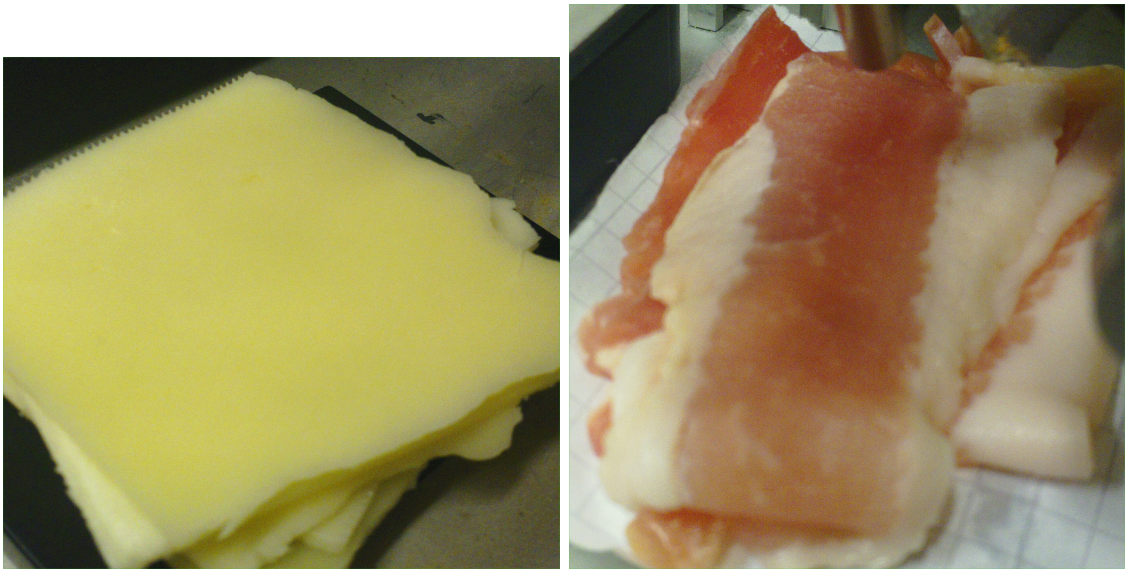
illuminated through two glass filters. The transmission spectra of the glass filters are found in Figure 7.2. This light source, consisting of white light illuminated through glass filters, is from now on referred to as only 'emission light'. The power of the white light source was set to maximum (level 5) for the camera measurements and minimum (level 1) for the spectrometer measurements.

3.1 Phantom preparation

Two samples, a_1 and a_2, were prepared: a_1 consisted only diluted fluorescein (concentration $0.77\,g/l$), while a_2 was a mixture of diluted fluorescein (concentration $0.77\,g/l$) and intralipid ($200\,g/l$.

Sixteen homogeneous liquid tissue phantoms, labeled c_{ij}, for $i \in [1\ 4]$, and $j \in [1\ 4]$, were prepared by mixing water ($9.615\,ml$), intralipid (concentration: $200\,g/l$, volume: $365\,\mu l$), ink (concentration:1 : 100, and volumes $0\,\mu l$ (i=1), $50\,\mu l$ (i=2), $75\,\mu l$ (i=3) and $250\,\mu l$ (i=4)) and fluorescein (concentration:$10\,\mu M$, and volumes $25\,\mu l$ (j=1), $50\,\mu l$ (j=2), $75\,\mu l$ (j=3) and $100\,\mu l$ (j=4)). The recipe of these tissue phantoms is a modified version of another recipe given in [24]. In addition to these 16 phantoms, a reference phantom without either fluorescein or ink, only water and intralipid, was made and labeled c_0.

Two biological materials were prepared, cheese (Synnøve finden, original gulost skivet) and bacon (Folkets, norgesgruppen, skivet), labeled c and d respectively. See Figure 3.2 for illustration.



(a) Cheese (b) Bacon

Figure 3.2: Biological materials

3.2 Camera measurement

Dark Current

Dark current was measured by turning any light source off, so that only background noise was present on the images. Two dark current measurement was done in total, one with exposure time 6.8 ms and one with 32 ms. The dark current data was subtracted from the measurement data, in order to remove dark current from the data.

Calibration

A spectralon diffuse reflectance gray standard (Pro-lite Technology, SRT-50-050) with known spectral reflectance was used in order to calibrate the system. Two calibration measurements were done, i.e. measurement of reflected power by gray standard of excitation and emission light. Calibration of excitation light was done with an excitation filter (Asahi Spectra, XHQA 280) in front of the camera. Dark current data was extracted from the calibration data, and then it was divided by the reflectance of gray standard (0.454 for excitation light, and 0.470 for emission light).

Figure 3.3: Gray standard and container to contain sample

The maximum power of the reflectance was controlled to not exceed saturation for both excitation and emission light. In addition, basically all power of excitation and emission light was controlled to fall within the field of view of the camera.

Sample measurements

Each phantom, was stored in a cylindrical plastic container in a refrigerator of $4°C$ for a couple of days before the measurements was done. An amount of 700 ml of each phantom was measured by a pipette and placed in a small black plastic container with dimentions $2.1 \times 2.1 \times 1.5\ cm^{-3}$ (see Figure 3.4). The amount phantom liquid, from now

on referred to as 'sample', was chosen such that the surface of gray standard and sample was approximately at the same height, i.e. the same distance from camera.

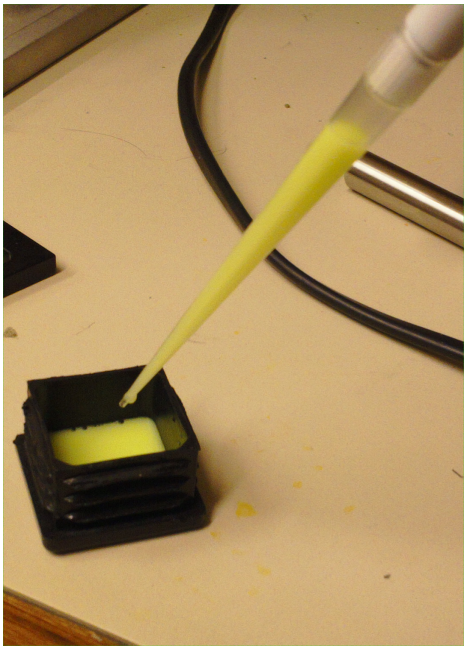


Figure 3.4: An amount of 7 ml of phantom liquid was measured by a pipette and placed in a small, black container.

The sample was illuminated by emission light and data was acquired. Dark current data was subtracted, and then it was divided by calibration data of emission light.

Diffuse reflectance data was acquired by illumination of sample by excitation light, with the excitation filter in front of camera. Then dark current data was subtracted, and further divided by the calibration data of excitation light.

In order to measure the fluorescence, two measurements was done; with and without emission filter in front of camera. The purpose of the emission filter is to block out diffuse reflectance of excitation light. Both measurements was done by illumination of sample by excitation light, but with two different exposure times, i.e. 6.8 ms in the case without filter (as the rest of the measurements), and 32 ms with emission filter.

3.3 Spectroscopy measurements

Basically four measurements was done for each sample: Gray standard illuminated by emission (*) and excitation (**) light, and sample illuminated by emission (***) and excitation (****) light. One dark current measurement was done, whose spectra was subtracted from the other measurements.

The final calibration data was obtained by dividing the gray standard data by the reflectance of the gray standard.

Excitation diffuse reflectance was calculated by dividing (****) by the excitation calibration data, while emission diffuse reflectance was obtained by dividing (***) by the emission calibration data.

3.4 Experimental setup

A sketch of relative positions between light sources, camera, spectrometer and gray standard/sample is provided in Figure 3.5 and Figure 3.6. The exact positions relative to the center of image is given in Table 3.1.

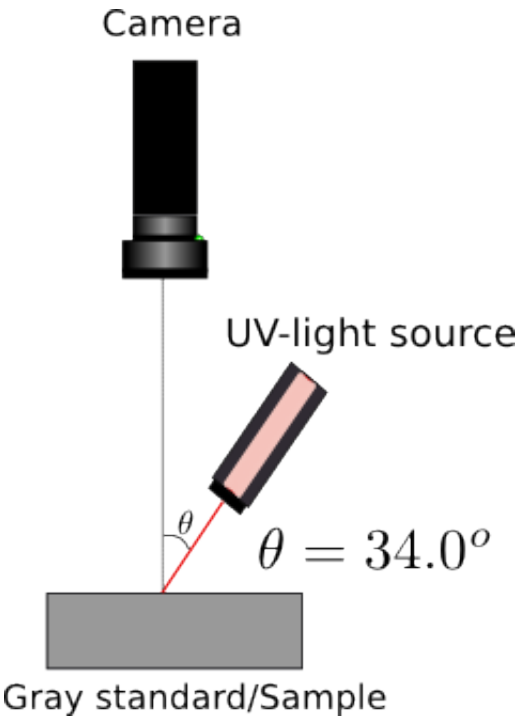


Figure 3.5: Sketch of relative positions of camera, UV light source and gray standard-/sample.

	x [mm]	z [mm]
Camera	0	52
Spectrometer	6.3	9.5
Excitation light	7.5	11.1
White light	34	57

Table 3.1: Positions relative to center of image. x and z means horizontal and vertical positions respectively.

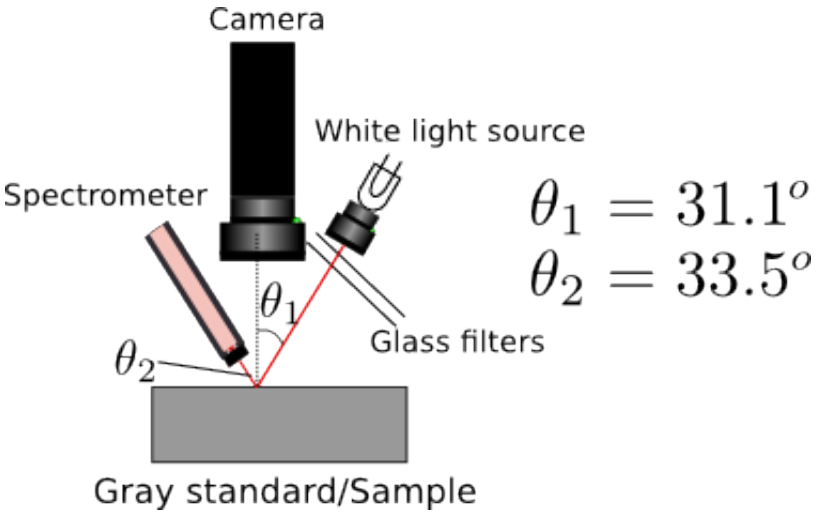


Figure 3.6: Sketch of relative positions between camera, emission light source, spectrometer and gray standard/sample.

3.5 Additional measurements

If not otherwise stated, the procedure and setup previously described, is valid in this section as well.

Properties of excitation filter

As a part of troubleshooting, the properties of the excitation filter had to be characterized. Gray standard was illuminated by emission light.

Spectrometer

The x and z coordinates of the spectrometer probe was modified to 19.2 and 32 mm respectively (which gives the angle θ_2 from Figure 3.6 an an angle value of 29.5 degrees). The power of emission light was modified to level 2. A calibration measurement with gray standard and emission light was done. Then the excitation filter was placed in front of the spectrometer probe as showed in Figure 3.9, and a new spectrometer measurement was done.

Camera

In the camera measurement, data of dark current, and reflected power of emission light by gray standard with and without excitation light in front of camera was acquired.

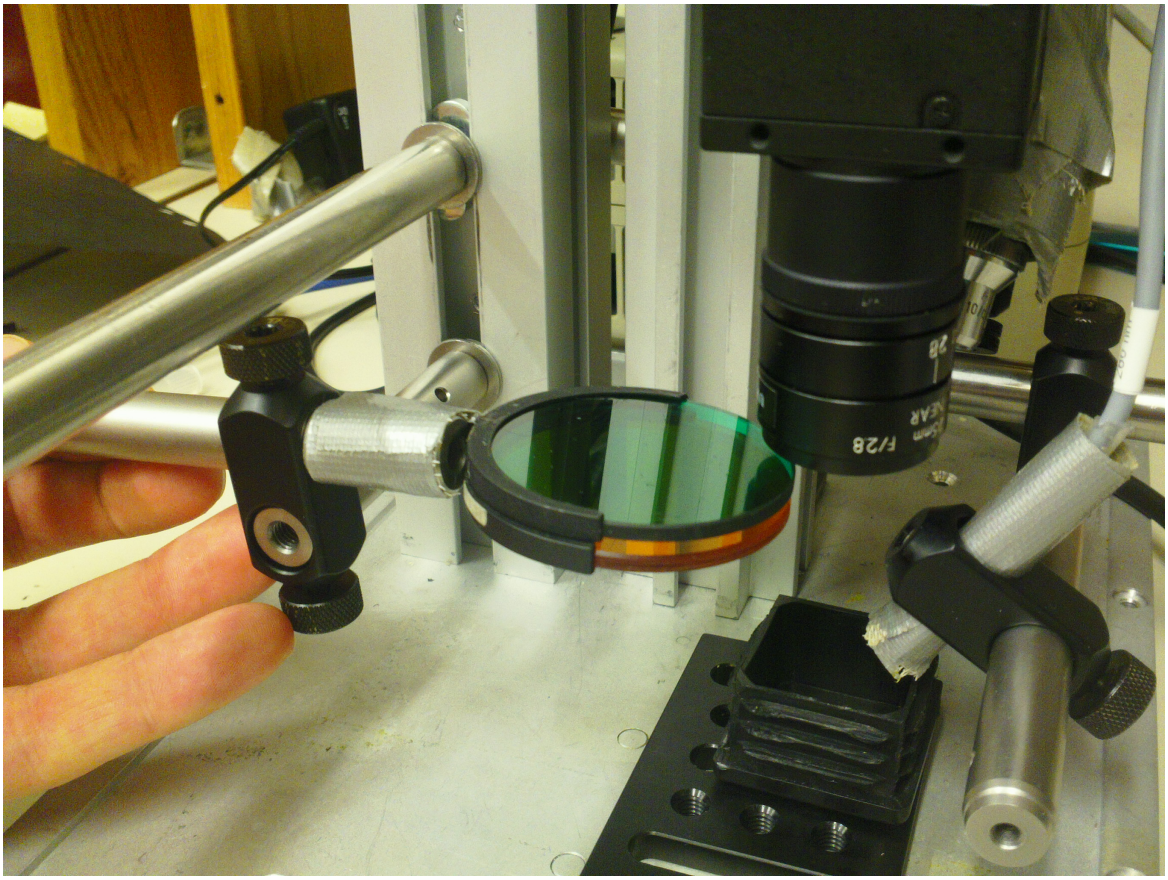


Figure 3.7: Position of excitation light and black container to contain sample, and placement of emission filter in front of camera.

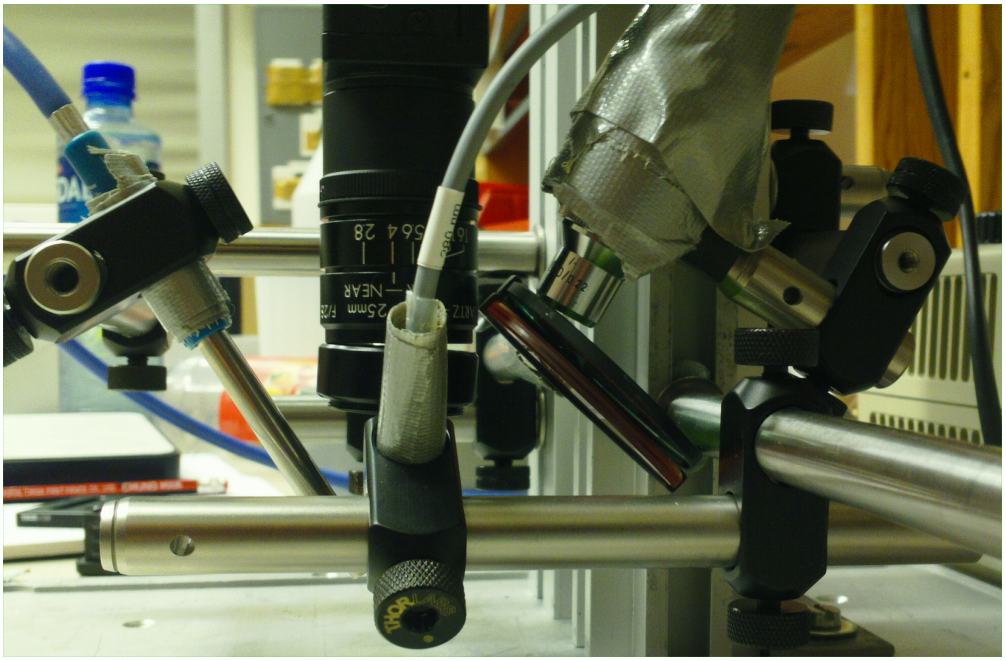


Figure 3.8: Spectrometer probe at the left side, white light through glass filters to the right, excitation filter in beneath the camera and excitation light at this side of the camera.

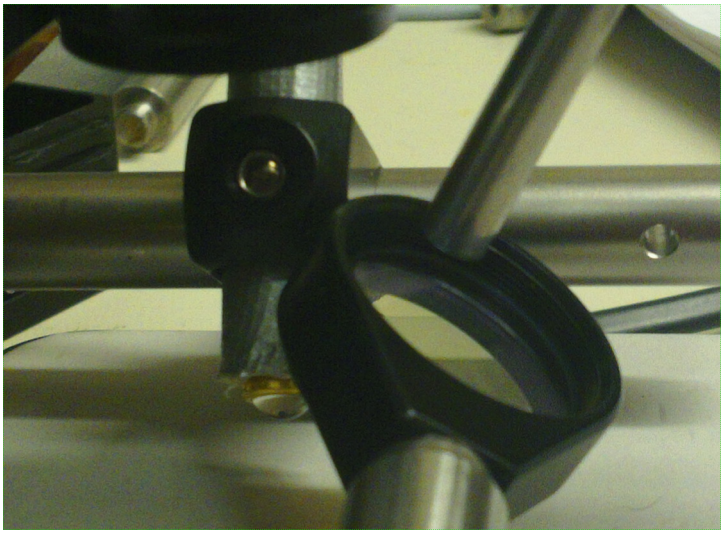


Figure 3.9: Excitation placed in front of spectrometer probe

Determination of photobleaching

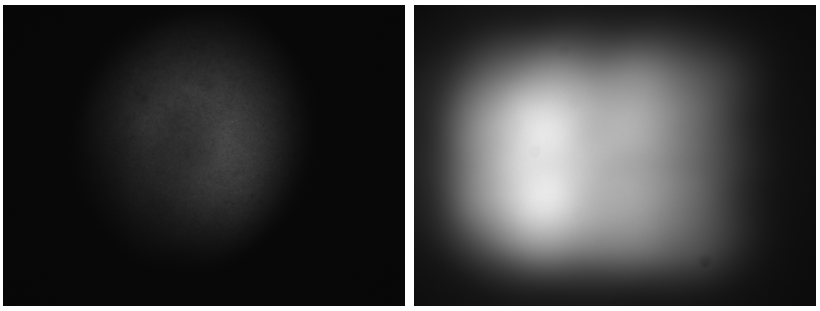
7 ml of phantom a_1 was prepared as illustrated in Figure 3.4. 60 text-files in total were acquired in intervals of 10 seconds. I.e. the first was acquired at $t = 0$, the second at $t = 10$ and so on until the last image which was acquired at $t = 590$.

Determination of η and α

In the determination of η from Equation 2.59 the fluorescence spectrum of the phantom consisting of 75 μl ink, and 75 μl fluorescein was used. The spectrum was then multiplied by the filter spectrum showed in Figure 7.2 in the range 480 to 650 nm.

In order to calibrate the α from Equation 2.47, the total power of the excitation and emission light source, i.e. $P_{incident}\,[W]$, was measured by means of a power-meter (Thorlabs PM100D console SN p0005536, Thorlabs S130VC SN photo-diode 12112303 range 200-1100 nm). The active area of the power-meter was illuminated by the whole beam of excitation and emission light.

$\sum_{MxN} P_{det}[digitalnumber]$ from Equation 2.47 was calibrated by addition of all pixel values obtained by measurement of gray standard illuminated by emission light. The emission and excitation light beam was directed such that the whole beam was captured by camera. See Figure 3.10 for illustration of emission and excitation light reflected by gray standard.



(a) Emission light reflected by gray standard

(b) Excitation light reflected by gray standard

Figure 3.10

Estimation of effective penetration depth

In order to calculate the one-dimensional path-length factor X_{1D}, expressed in Equation 2.15, the effective penetration depth δ, has to be estimated. δ is here calculated by Equation 2.16. Similar phantoms as c_{ij} were made in [24], but with ink volumes of $0\,\mu l$, $50\,\mu l$, $75\,\mu l$ and $25\,\mu l$, and $10\mu M$ diluted fluorescein with volumes of $25\,\mu l$, $50\,\mu l$, $75\,\mu l$ and $100\,\mu l$, added the the same volume fraction as in c_{ij}. The values of μ_a and μ_s were measured accurately in [24] with a time-of-flight (TOF) spectroscopy. The results of the μ_a measurements

expressed in cm^{-1} were 0.012, 0.26, 0.37 and 0.14 for the ink concentrations respectively, while for the μ_s measurements, it was, 10.10 cm^{-1} and 8.30 cm^{-1} at the wavelengths 532 and 600 nm respectively.

The phantoms c_{ij} of this report is equivalent to [24], except that c_{i4} is multiplied by 10 compared to the phantoms of [24]. Concequently μ_a for the ink concentrations $0\,\mu l$, $50\,\mu l$, $75\,\mu l$ and $250\,\mu l$ is estimated to have the values 0.012, 0.26, 0.37 and 0.14 cm^{-1} respectively. The estimated value of μ_s is 10.10 cm^{-1} and 8.30 cm^{-1} for the excitation and emission light of this report.

3.5.1 Calculation of X_{dil}

The parameters used to calculate X_{dil} from Equation 2.22, is expressed in Table 3.2.

Parameter	Value
l	$1.4cm$
n_t	1
n_i	1.33
θ_t	$34.00°$
θ_t	$24.86°$

Table 3.2: Parameters used in the calculation of X_{dil}

4 Results

4.1 Raw data

4.1.1 Additional measurements

Bleaching

One average value per image was calculated from rows 260:780 and columns 348:1044, and plotted in Figure 4.1 as a function of when the image was acquired.

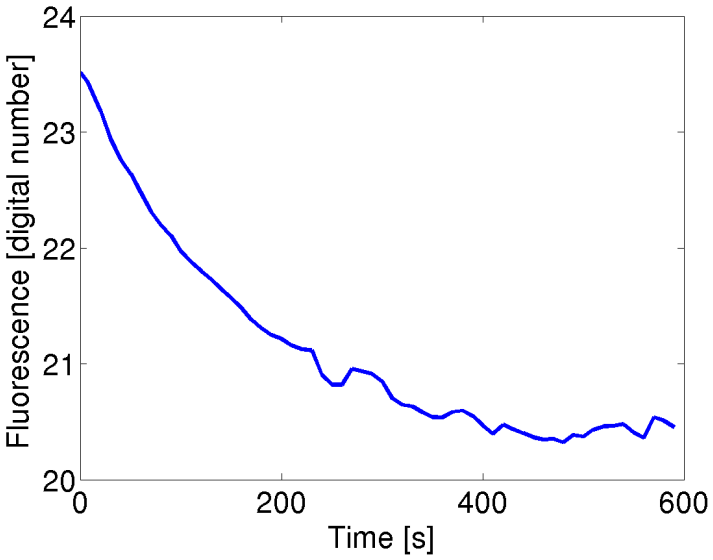


Figure 4.1: Bleaching. Fluorescence intensity as a function time, with constant power of excitation light.

Verification of excitation filter

The pixels to be averaged was column 390:650 and row 522:870. The value of reflected power without filter was measured to be 3.10, while it was 1.25 with filter (unit of digital number).

The result of the spectrometer measurement is showed in Figure 4.2

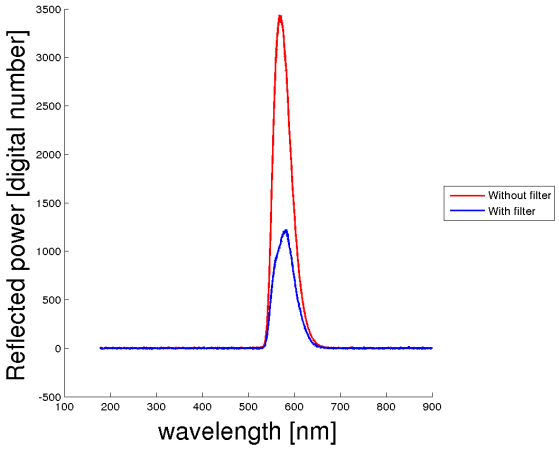


Figure 4.2: Gray standard illuminated by emission light, with (blue graph) and whitout (red graph) excitation filter in front of spectrometer probe.

Determination of η and α

In the determination of η from Equation 2.59 the fluorescence spectrum of phantom $c_{33}$was used. The fluorescence spectrum of the phantom consisting of 75 μl ink, and 75 μl fluorescein was integrated in the range 480 to 650 nm, to find the denomirator of Equation 2.59 and then the fluorescence spectrum were multiplied by the filter spectrum showed in Figure 7.2 and integrated in the same range to find the numerator of Equation 2.59. The obtained value for η was calculated to be 0.0527.

$P_0\,[W]$ for emission light was measured to be 43.1 μW, while the tabulated value of gray standard reflectance for wavelength of 570 nm was 0.47. The value of $\sum_{MxN} P_{det}[digitalnumber]$ for emission light was calculated to be 1.44×10^7. Equation 2.47 then gives $\alpha = 1.41 \times 10^{-12}$.

The measured value of excitation light source was 400 μW, and with tabulated gray standard value 0.454 and a calculated value of $\sum_{MxN} P_{det}[digitalnumber]$ equal to 1.12×10^8, gave $\alpha = 1.41 \times 10^{-12}$ by the use of Equation 2.47.

Estimation of X_{dil} and X_{1D}

The value of X_{dil} for phantom a_1 and X_{1D} for phantom a_2 is found in Table 4.1.

Quantity	Value $[cm^{-1}]$
X_{dil}	0.052
X_{1D}	0.0012

Table 4.1

The X_{1D} values for c_{ij}

2.956	0.593	0.482	0.225
2.960	0.608	0.495	0.222
3.026	0.596	0.492	0.227
3.024	0.620	0.508	0.232

Table 4.2: The values of X_{1D} for all phantoms c_{ij}, where i and j denote the row and column number respectively

4.1.2 Non-spectral measurements

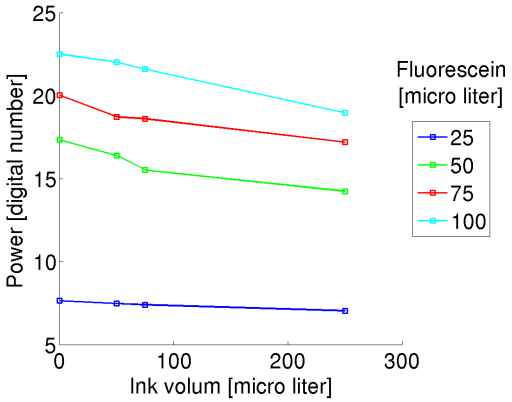
For the camera measurements, an average value of a part of image is calculated and plotted for each sample. The selection of the image where the average is calculated from, is row 348:1044 and column 260:780. The fluorescence power of all c_{ij} phantoms is illustrated in Figure 4.3. Figure 4.3a and Figure 4.3b is the fluoresence power in the unit of digital number measured by the camera and spectrometer respectively. α was calibrated by an exposure-time of $t_\alpha = 6.8ms$ while fluorescence image was acquired with exposure-time $t_f = 30ms$. Figure 4.3c illustrates the fluorescence in the unit of watts, calculated by Equation 2.60

	Cheese	Bacon	a_1	a_2
Diffuse emission reflectance	0.310	0.103	0.004	0.222
Diffuse excitation reflectance	0.044	0.037	0.049	0.105
Fluorescence without filter	29.5	21.1	22.9	47.4
Flourescence with filter	-	-	9.6	25.5

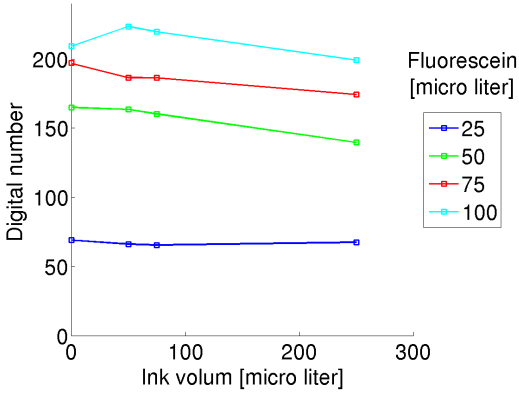
Table 4.3

Excitation light	Emission light
157.70	32.01
157.84	30.41
157.15	30.64
155.01	31.24
158.42	29.26

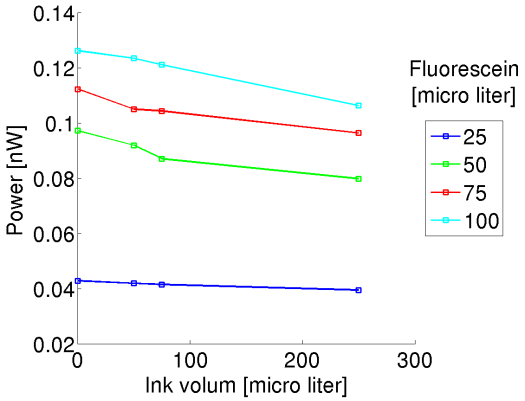
Table 4.4: This table shows the values from 5 different measurement of gray standard illuminated a light source. The power of the excitation and emission light source was constant during the 5 measurements.



(a) Camera.

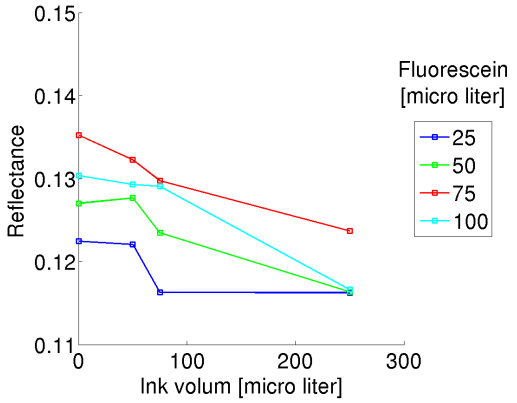


(b) Spectrometer. Average values for wavelengths between 523 and 528 nm.

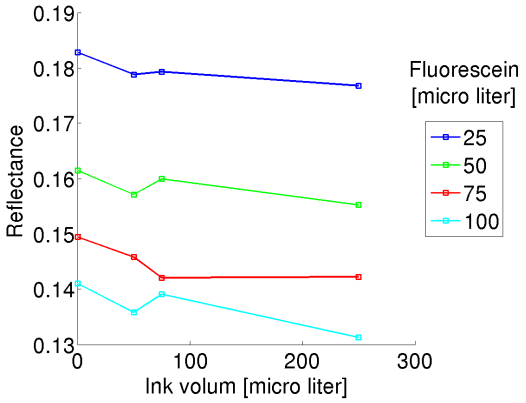


(c) By the use of

Figure 4.3: Fluorescence measurements

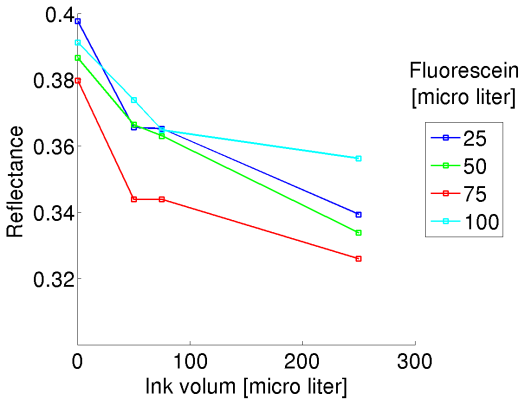


(a) Camera.

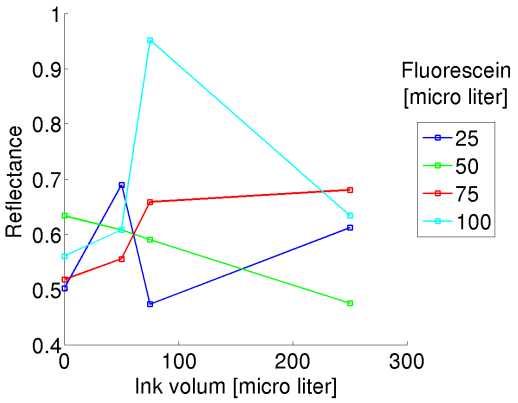


(b) Spectrometer. Average values for wavelengths between 275 and 289 nm.

Figure 4.4: Diffuse excitation reflection



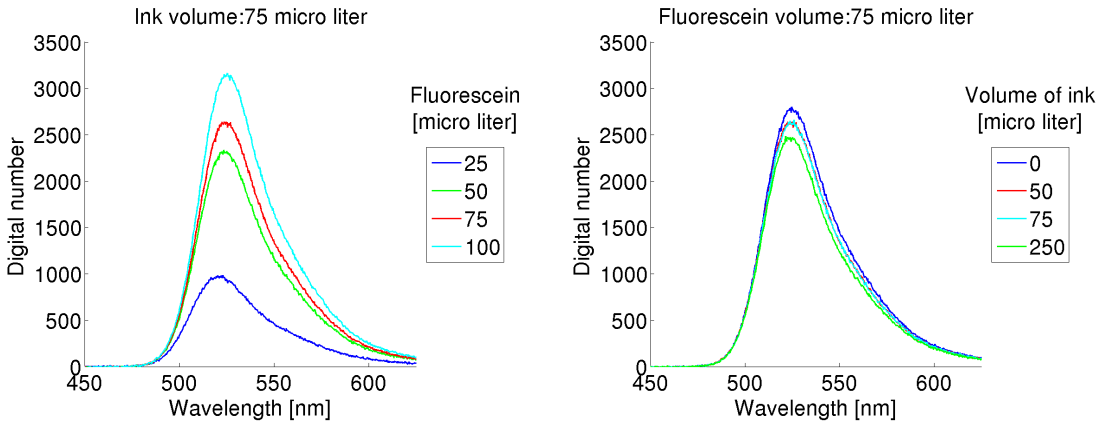
(a) Camera



(b) Spectrometer. Average values for wavelengths between 590 and 600 nm.

Figure 4.5: Diffuse emission reflection

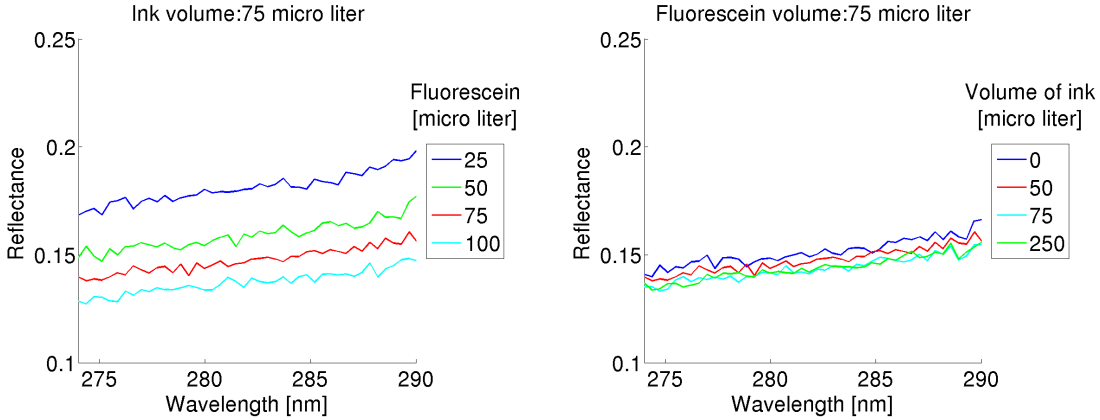
4.1.3 Spectrometer measurements



(a) Different fluorescein concentrations. Volume of ink was 0.75 μl

(b) Different ink concentrations. Volume of fluorescein was 0.75 μl.

Figure 4.6: Sample illuminated by excitation light



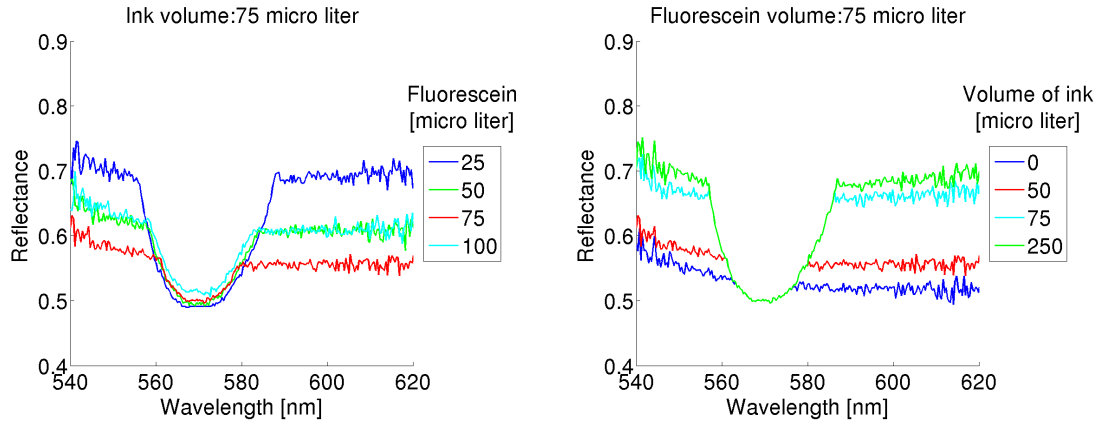
(a) Different concentrations of fluorescein. Volume of ink was 0.75 μl

(b) Different concentrations of ink. Volume of fluorescein was 0.75 μl

Figure 4.7: Diffuse reflectance of excitation light.

4.2 Recovering of spectral intrinsic fluorescence

The spectral intrinsic fluorescence was recovered by Equation 2.37 and the correction factors ${}^{t_c}/_{t_f}$ and η introduced in subsection 2.5.7. The spectral intrinsic fluorescence for the camera measurements of c_{ij} is plotted in Figure 4.11, while table carries the intrinsic



(a) Different concentrations of fluorescein. Volume of ink was 75 μl

(b) Different concentrations of ink. Volume of fluorescein was 75 μl

Figure 4.8: Diffuse reflectance of emission light.

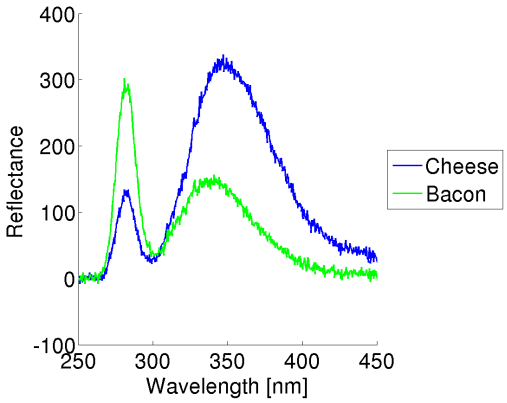
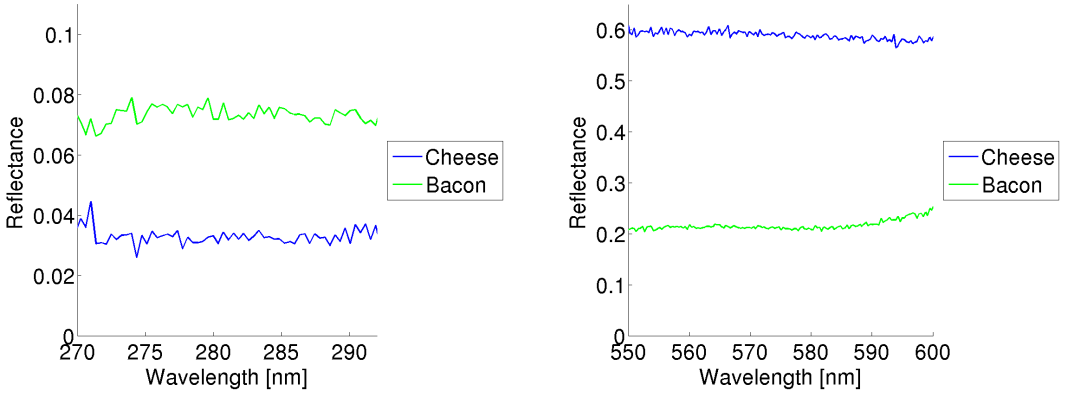


Figure 4.9: Biological materials illuminated by excitation light



(a) Excitation light (b) Emission light

Figure 4.10: Diffuse reflectance of biological materials

fluorescence of a_1. Equation 2.41 was used to calculate the intrinsic fluorescence of phantom a_2. The result is found in Table 4.5.

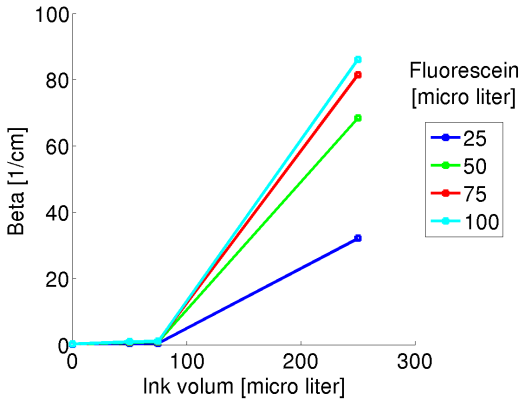


Figure 4.11: Intrinsic fluorescence coefficient

Phantom	β
a_1	12.01
a_2	1705

Table 4.5: Intrinsic fluorescence coefficient of phantom a_1 and a_2

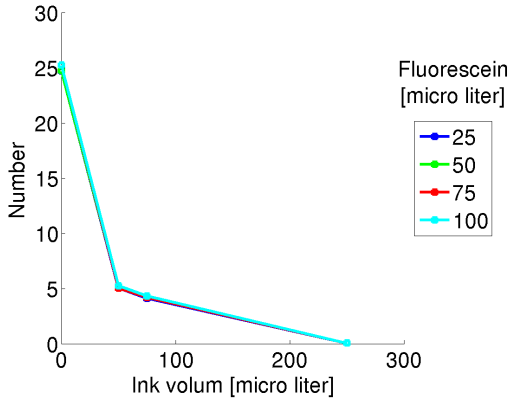


Figure 4.12: X1D

5 Discussion

According to [19] the diffuse reflectance is expected to decrease as a function of increasing fluophore concentration. However, this didn't happened with the diffuse reflectance for camera measurements represented in 4.4a. The reason for unexpected results of excitation diffuse reflectance, is verified was an excitation filter not working properly. The excitation filter was supposed to block out fluorescence light, letting only excitation light through.

As stated in section 4.1.1, the detected power in digital power was 3.10 without excitation filter, while it was 1.25 with filter. Consequently $^{1.25}/_{3.10} = 0.40$ of the emission light was let through the excitation filter. The limited property of the excitation filter is also proved by spectrometer measurement shown in Figure 4.2. Ideally, the blue graph of Figure 4.2 should be flat so that no light of fluorescence wavelength was able to come through the filter. However, this is not what happened, and the camera measurement of diffuse excitation reflectance is messed up.

Despite limited property of excitation filter, 4.4a shows some relation between increased amount of absorbers and decreased diffuse reflectance. However, Figure 4.3a shows that with increasing ink consentration, the fluorescence power decreases. That is, if excitation diffuse reflectance was constant, the camera measurements would still give decreasing diffuse reflectance, due to the fact that both reflected excitation light and fluorescence light propagates through the filter. Taking this into acount, excitation diffuse reflection is vaguely dependent on ink concentration.

The limited property of the excitation filter, explains the "bad" results of camera measurements of diffuse reflectance. Anyway, the spectrometer measurement was done without any filter, and should give "good" results. As indicated in Figure 4.7a and 4.4b, increased fluorescein concentration gave lower diffuse excitation reflectance, as expected according to [19]. A vaguely dependence between ink concentration and excitation reflectance is illustrated in 4.7b and 4.4b. In other words, both camera measurements and spectrometer measurements gave vaguely dependence between reflectance of excitation light and ink concentration.

Obviously, there is something wrong with the measured diffuse reflectance of emission light. In Figure 4.5b there is a correspondance between increased amount of absorbers and lower reflectance. Diffuse reflectance of emission light measured by spectrometer, illustrated in 4.5b seems to be totally random. The lest reasonable results is when fluorescein volume is 75 μl, whose spectra for different ink concentrations are illustrated in 4.8b. Here, the dependence is totally opposite: Higher reflecatnce du to higher concentrations of absorbers. This "opposite relationship" is however, not conformed for other fluorescein concentrations.

An unstable emission light sources may be one of the reasons why the emission diffuse

reflectance measurements showed strange results. In table Table 4.4, we see that the calibrated value of emission light source varies even though the the power emitted was supposed to be constant. The highest value of these calibrated values was 32.01, which is 9.39% higher than the lowest value 29.26.

A comparison between scattering and non-scattering intrinsic fluorescence coefficients in Equation 2.57 and Equation 2.55, gives the following ratio between B_{scat} and $B_{non-scat}$: $B_{scat}/B_{non-scat} = \frac{F_{det,scat}}{F_{det,non-scat}} \frac{X_{dil}}{X_{1D}} \frac{\pi}{cos(\theta)}$. The value of $\frac{X_{dil}}{X_{1D}}$ from table Table 4.1 is 43.33, while the value of $\frac{\pi}{cos(\theta)}$ becomes 3.79 for $\theta = 34°$, such that $\frac{X_{dil}}{X_{1D}} \frac{\pi}{cos(\theta)} = 164.2$. In order to obtain the same intrinsic fluorescence B for scattering and non-scattering samples, we must have $\frac{F_{det,non-scat}}{F_{det,scat}} = \frac{X_{dil}}{X_{1D}} \frac{\pi}{cos(\theta)} = 164.2$. This means that detected fluorescence from the diluted solution has to be a way larger than the scattering solution. The obtained value of $\frac{F_{det,non-scat}}{F_{det,scat}}$ from Table 4.5 was $\frac{25.5}{9.6} \approx 2.66$, which deviates a lot from 164.2. The same value of X_{dil} as in [11] was however obtained, when the parameters θ_t and l where chenged to $\theta_t = 10°$ and $l = 1cm$, just like the variables of the calculation of X_{dil} in [11] . In other words, the value of X_{dil} is according to [11], and the estimation of X_{1D} might be wrong. The absoption coefficient of phantom a_1 is plotted in Figure 7.1. The peak absorption is at 489nm, where $\mu_a = 348\text{cm}^{-1}$. The absorption curve drops rapidly, and at 520 nm, it is $\mu_a = 40.6cm^{-1}$, while at 570 nm, $\mu_a = 4\text{cm}^{-1}$. The emission range of filtered fluorescence, illustrated in Figure 2.6c, is about [540 600], and an absorption coefficient of 4cm^{-1} sounds reasonable. As well do the absorption coefficient of the excitation light of 92cm^{-1}. The estimations of the scattering coefficients $\mu_s(\lambda_{ex})$ and $\mu_s(\lambda_{em})$ is vaguely determined. Figure 5.1 shows how X_{1D} is dependent on $\mu_s(\lambda_{ex})$ and$\mu_s(\lambda_{em})$. $\mu_s(\lambda_{ex})$ ($\mu_s(\lambda_{em})$) was varied while $\mu_s(\lambda_{em})$ ($\mu_s(\lambda_{em})$) was equal to 10cm^{-1} and every other parameter were those given by the measurement of a_2 (given in Table 4.3).

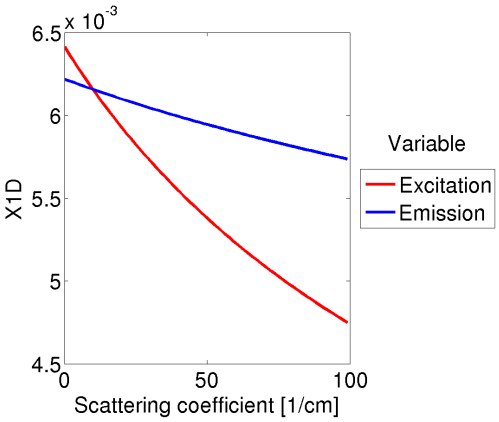


Figure 5.1: Variation of X1D as a function of emission scattering coefficient (blue graph) and excitation scattering coefficient (red graph).

Figure 5.1 tells us that wrong estimation of scattering coefficents $\mu_s(\lambda_{ex})$ and$\mu_s(\lambda_{em})$ will affect X_{1D}, but only slightly.

The results of the spectrometer measurements of the biological samples illustrated in

Figure 4.9 clearly shows that the biological materials of this report fluoresce. Cheese fluorescence more than bacon in both spectrometer measurements (Figure 4.9) and camera measurements (Table 4.3). The results from both camera measurements (Table 4.3) and spectrometer measurements (Figure 4.10) showed that cheese reflect more light at emission light, while bacon reflects more at excitation light. An emission filter for the wavelengths of fluorescence of the biological samples where note available, and consequently the fluorescence values of the camera measurements includes both excitation diffuse reflectance and fluorescence, and are not appropriate to calculate intrinsic fluorescence from.

The obtained values of the calibration factors in section 4.1.1 gave ${}^{\alpha(\lambda_{ex})}/_{\alpha(\lambda_{em})} = 1.15$. That is, if equal power of excitation and emission light is sent into the camera, the emission light would be amplified by a factor 1.15 compared to the emission light. The spectral response illustrated in section 7.4 implies however, that this factor should be about 2.

6 Conclusion

The correspondance between the theory and the experimental results of this report was weak. There could be many reasons for this, among others is small accuracy of the estimation of absorption- and scattering coefficients of the material. The correspondance between the obtained data and the concentration of absorbers was weak, while the dependence between obtained data and fluorophore concentration was strong. The obtained values of intrinsic fluorescence were too high for the phantoms with high absorption coefficients compared to the obtained values of intrinsic fluorescence for the phantoms with less absorers. However, a proposed measurement procedure with a corresponding theory is proposed in this thesis. Too improve and investigate the feature of the method further, the optical properties of the phantoms have to be accurately determined.

7 Appendix

7.1 Measurement of fluorophore absorption coefficient ($\mu_{a,fl}$)

The transmission as a function of wavelength of different concentrations of diluted fluorescein sodijm+pbs was measured by a spectrometer (Hewlett Packard 8453). 10 measurements was done with concentration $0.7700 g/l$, while one measurement was done with one fourth of the concentration, i.e. $\frac{0.7700}{4} g/l = 0.1925 g/l$. The concentration of $0.7700 g/l$ is what we want to calculate the absorption coefficient from. Hence the absorption coefficient from the $0.1925 g/l$ concentration has to be corrected by a multiply of 4. The propagation distance x in the measurements was $0.1cm$. MATLAB R2012a was used to calculate and plot the absorption coefficient as a function of wavelength based on the transmission data and Equation 2.5 for each wavelength λ. The MATLAB-code was as follows.

```
%Calculation absorption coefficient based on Transmission data.
%The transmission files FULL1.csv-FULL10.csv and D3W1F.CSV contains
%transmittance data as function of wavelength.
%The files FULL1.csv - FULL10.csv contains the concentration from which we
%want to calculate the absorption coefficient.


close all;
clear all;

x=0.1; %Propagation distance [cm].
n = 10; %Number of FULL.csv files

%Load transmission data from FULL1.CSV into the vector AVG
AVG = dlmread('FULL1.CSV');

%Load transmission data from D3W1F.CSV into the vector DSW
D3W1F = dlmread('D3W1F.CSV');

%Calculation of avarage of FULL1.csv-FULL10.csv
for i=2:n;
    filnavn=strcat('FULL',num2str(i),'.CSV');
    u = dlmread(filnavn);
    AVG(:,2) = AVG(:,2) + u(:,2);
```

```
end
AVG(:,2) = AVG(:,2)/n;

%Transformation of transmission data into percent
AVG(:,2)=0.01*AVG(:,2);
D3W1F(:,2)=0.01*D3W1F(:,2);


%Calculation of absorption coefficients [cm^-1] of the transmission datas
u_AVG = -log(AVG(:,2))/x;
u_D3W1F = -log(D3W1F(:,2))/x;

u_D3W1F = u_D3W1F*4; %Corrected absorption coefficient

hold on;
plot(AVG(:,1),u_AVG); hold on;
plot(D3W1F(:,1),u_D3W1F,'r');
ylabel('Absorption coefficient [1/cm]');
xlabel('Wavelength[nm]');
```

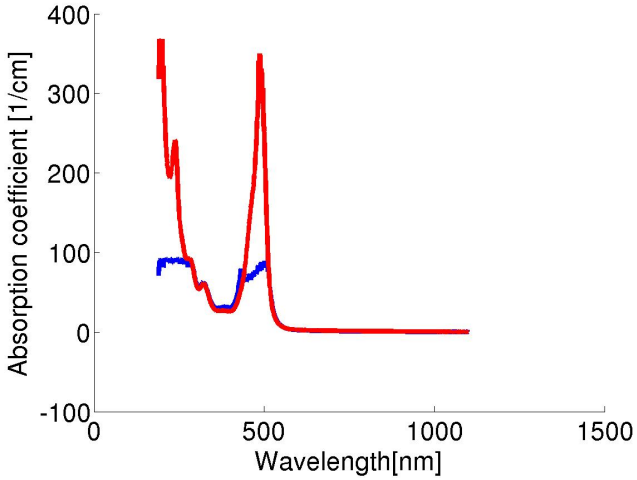


Figure 7.1: The absorption coefficient $\mu_{a,fl}$ of fluorescein sodijm+pbs as a function of wavelength. Blue graph corresponds to a concentration of 0.7700 g/l, while red graph corresponds to a corrected spectrum of the concentration 0.1925g/l.

7.2 Measurement of filter transmission

The transmission of the glass-filters (introduced in chapter 3) was measured by a spectrometer (Hewlett Packard 8453). MATLAB was used to plot the transmission of each

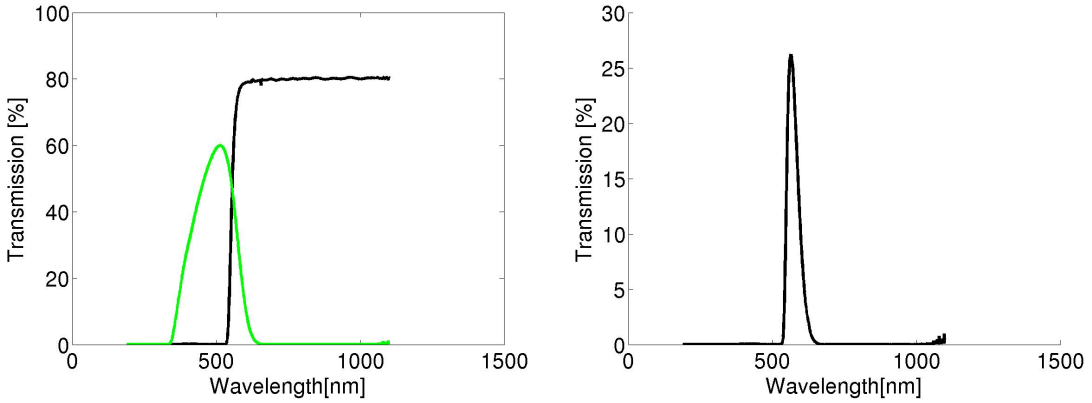
glass-filter, and to calculate total transmittance through both filters. The plots are found in Figure 7.2, while the MATLAB-code was as follows:

```
%This script plots the transmission spectra of the orange and green
%glass-filters. ORANGE.csv and GREEN.csv corresponds to the obtained
%transmission data for the orange and green transmission filter
%respectively. The transmission csv-file consist of three columns:
%wavelength, filter transmission and reference transmission

close all;
clear all;

ORANGE=dlmread('ORANGE.CSV');
GREEN=dlmread('GREEN.CSV');

%Plotting the data
plot(ORANGE(:,1),ORANGE(:,2),'k');
ylabel('Transmission [%]'); xlabel('Wavelength[nm]');hold on;
plot(GREEN(:,1),GREEN(:,2),'g');
figure; ylabel('Transmission [%]'); xlabel('Wavelength[nm]');hold on;
plot(GREEN(:,1),0.01*ORANGE(:,2).*GREEN(:,2),'k');
```



(b) Calculated total transmission of green and orange glass-filter

Figure 7.2: Transmission of glass-filters

7.3 Software

The computer software made to comply the fluorescence measurement in this thesis consists of two parts. The first part is a computer program named 'Fluor' whose task is

basically to acquire image data from camera measurements. The second part consists of MATLAB scripts (MATLAB R2012a) whose task is basically to do data processing on the data acquired by 'Fluor'. The software was made by means of sample code provided by the Camera producer (JAI) and the complete handbook [1] which contains description of all camera functions.

The program was made as an MFC application project file in Visual Studio 2010.

7.3.1 User description

Fluor

This section contains screen-shots of the computer software 'Fluor'. Illustrating colored rectangles and circles are made for convenience, but are not found in the real program. The program is run by clicking the exe-file, 'Fluor.exe' in the folder 'Debug' of the Visual studio project folder. See Figure 7.3 for illustration. The whole Project file has to be provided in order to run the program.

First step is therefore, as just explained and illustrated in Figure 7.3, to double click on 'Fluor.exe'. Be sure that the camera is turned on and connected to the computer. If not, the error message in Figure 7.4 will show up. Press then OK, exit the program, and enter it one more time with the camera turned on and connected. If the camera was found by the software, the camera ID is provided inside the gray rectangle in Figure 7.6.

Click on the icon surrounded by a red circle in figure Figure 7.6, and choose the preferable folder to contain the measurement data, and then click 'OK'. The data of the different measurements are then saved in corresponding folders inside the chosen target folder. In each of these measurement folders, the text and tiff files (averaged later by MATLAB script) are saved. An example is provided in figure 7.5a, where 'My Measurements/1/' is chosen as the target folder.

The number of text and tiff files stored per measurements can decided by the user, by typing the wanted number in the edit boxes surrounded by the green (number of tiff files) and blue (number of text files) circle.

By clicking on an arbitrary type of measurement, an explanation of the corresponding measurement procedure can be read in the instruction field, highlighted by a red rectangle in figure Figure 7.7. All instructions are summarized in Table 7.1.

Before doing any measurement, click 'Start Acquistion' surrounded by a blue rectangle in Figure 7.7. The maximum value of the image (of unit digital number) can then be read inside the yellow rectangle of Figure 7.7. This value is used as a guideline to adjust exposure-time, gain and power of light sources, so that the signal isn't saturated. Illuminated gray standard by excitation light should be the target during these adjustments. The signal should be as strong as possible without being saturated. When the instructions from the instruction field and the setup is fulfilled, there is time to begin the measurements. Click 'Save image', surrounded by a black rectangle in figure Figure 7.7, and wait during the image acquisition. The user can read the progress in the the instruction field, illustrated in Figure 7.8. When the acquisition is finished, the user is informed by a text box showed in Figure 7.9. The user then has to click 'OK' in the

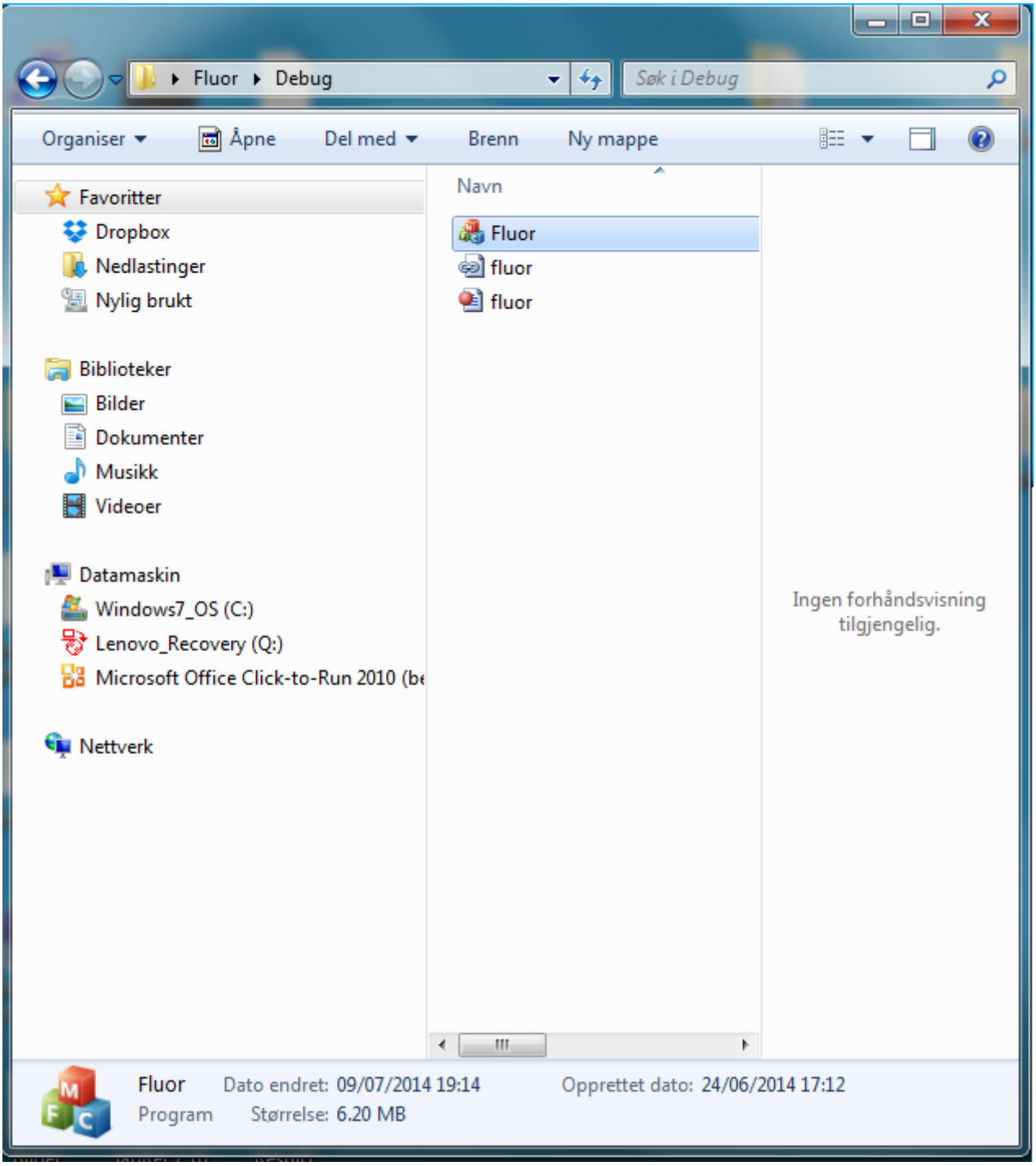


Figure 7.3: The project file for this program is labeled 'Fluor'. The exe-file is marked in this figure, and is found in the folder 'Debug'.

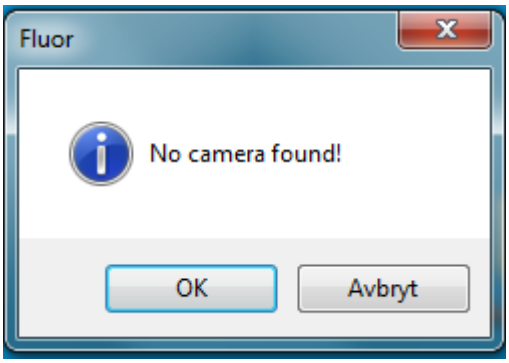
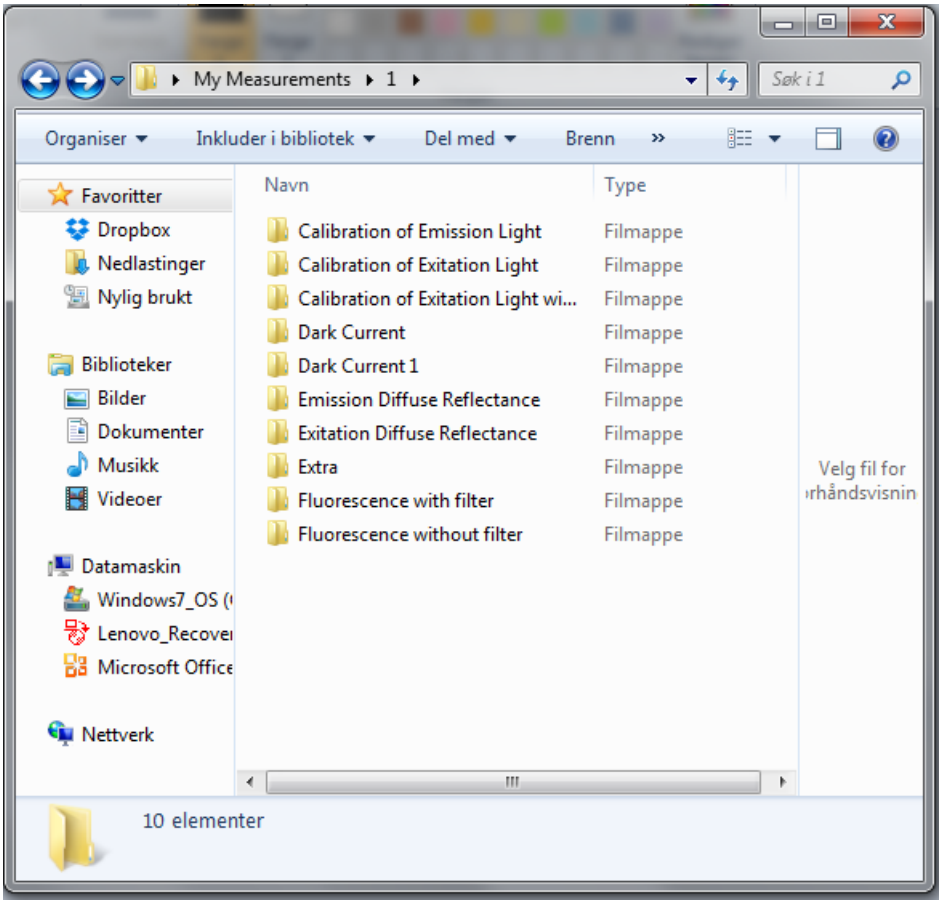


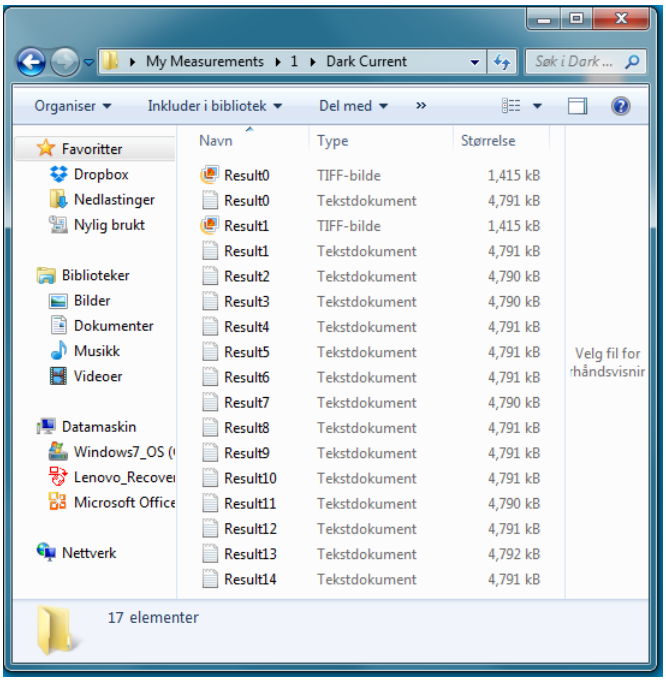
Figure 7.4: Error Message received if Camera wasn't found.

Type of measurement	Instruction
Dark Current	\nDark Current Measurement:\n\n1. Turn off all light sources so that only background noise is present.\n2. Click 'Save Image'.
Calibration of Emission Light	\nCalibration of Emission Light:\n\n1. Illuminate the gray standard by emission light\n2. Adjust exposure time, gain and power of the light source so that the maximum value of image is (just) below saturation.\n3. Click 'Save Image'\n4. Turn off light source when done.
Calibration of Excitation Light	\nCalibration of Excitation Ligh:\n\n1. Illuminate the gray standard by the excitation light\n2. Keep the exposure time and gain as from 'Calibration of emission light', but adjust the power of the light source so that the maximum value of image is (just) below saturation.\n3. Click 'Save Image'\n4. Turn off light source when done.
Calibration of Excitation Light with filter	\nCalibration of Excitation Light with filter:\n\nAs Calibration of Excitation Light, buth with excitation filter in front of camera.
Excitation Diffuse Reflectance	\nExitation Diffuse Reflectance:\n\n1. Keep the excitation filter in front of camera. \n2. Illuminate the sample by the excitation light.\n4. Click 'Save Image'\n5. Turn off light source when done\n\n(Remember to keep the same setup as in the calibration measurements including positions, exposure, gain and power of the light source)
Emission Diffuse Reflectance	\nEmission Diffuse Reflectance:\n\n1. Illuminate the sample by the white light source through glass-filters.\n2. Click 'Save Image'\n3.Turn of light source when done.\n\n(Remember to keep the same setup as in the calibration measurements including positions, exposure, gain and power of the light source)
Fluorescence without filter	\nFluorescence without filter:\n\n1. Illuminate the sample by the excitation light. \n2. Click 'Save Image'. \n3. Turn off excitation light when done.
Fluorescence with filter	\nFluorescence Measurements with filter:\n\n1. Increase gain/exposure-time, and place the emission filter in front of camera\n2. Illuminate the sample by the excitation light. \n3. Click 'Save Image'. \n4. Turn off excitation light when done.
Dark Current 1	\nDark Current 1:\n\n1. Keep the exposure time and gain as in the 'Fluorescence with filter' measurements. \nTurn off all light sources so that only background noise is present.\n3. Click save image.
Extra	\nExtra:\n\nEventually extra measurements

Table 7.1: Instructions showed inside the instruction field surrounded by a red rectangle in Figure 7.7 when the corresponding type of measurement is selected.



(a) The data is saved in corresponding measurement folders inside each sample folder specified by the user.



(b) Inside each measurement folder, the text- and tiff files are saved. In this example, 2 tiff files, and 15 text files are saved in the dark current measurement.

Figure 7.5: Hierarchy of saved data

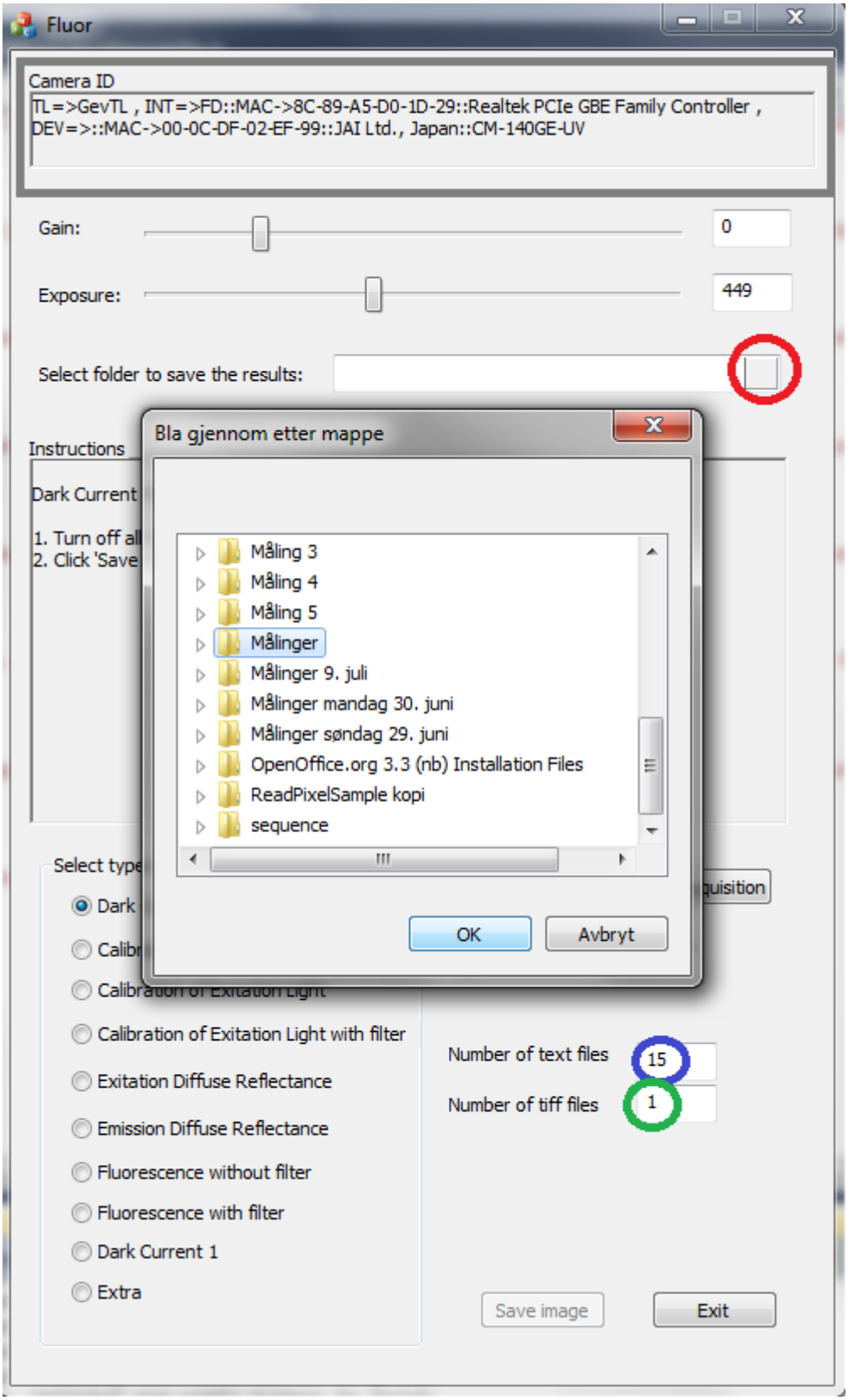


Figure 7.6: The user decides which folder to save the measurement data by clicking inside the red circle. Number of text and tiff files can be specified in the blue and green circle respectively. Information about the camera ID is provided inside the gray rectangle.

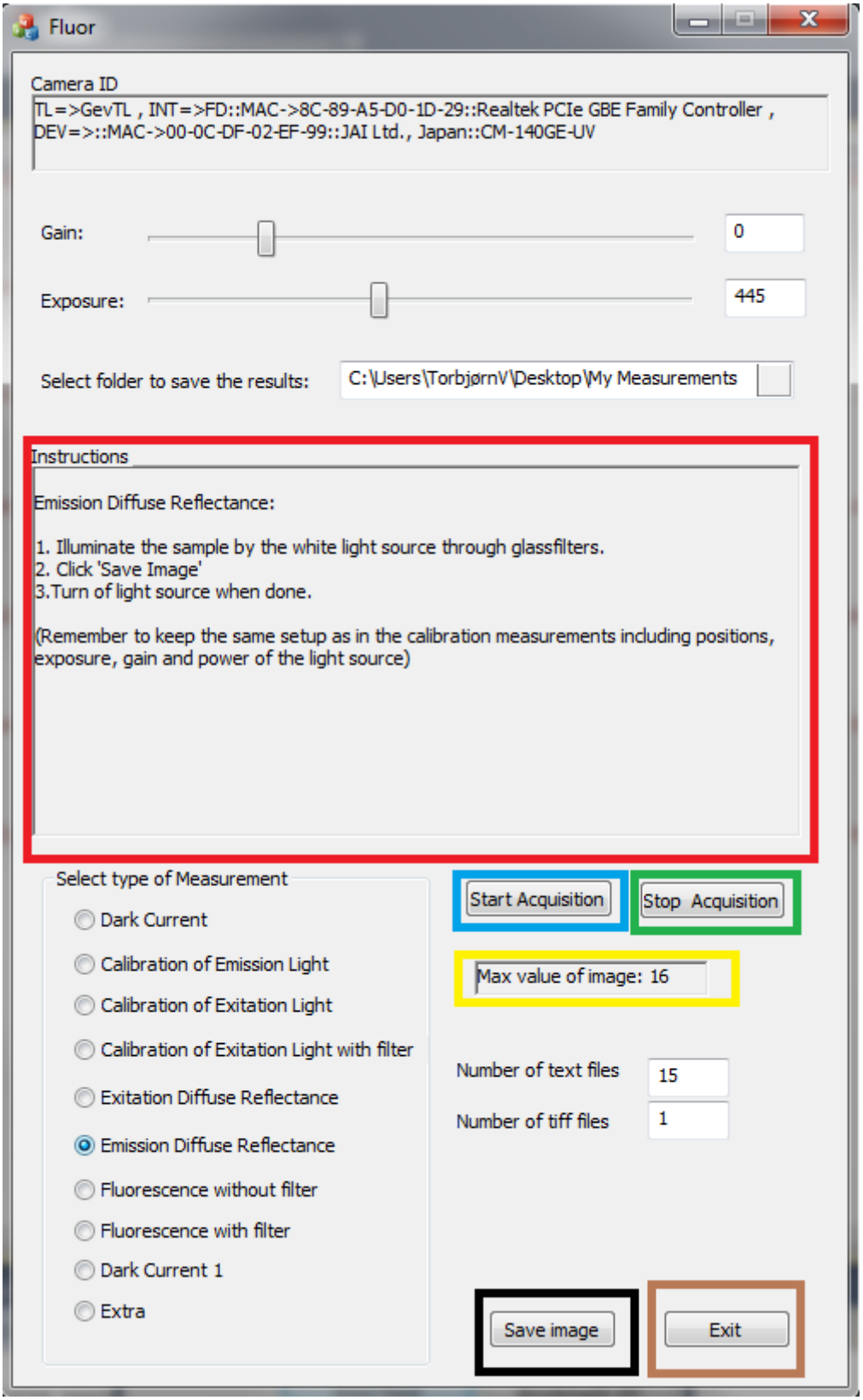


Figure 7.7: Example of instructions of a measurement procedure.

text-box, and there is time to prepare new measurements. When the instructions for the new measurements is fulfilled, a new measurement can be done in the same way as just described.

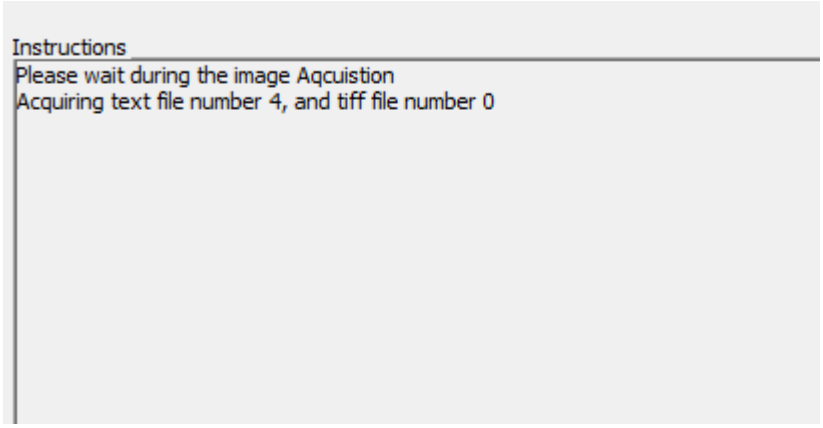


Figure 7.8: The user is told to wait until the image acqusition is finished. Number of acquired files can constantly be read in the instruction field.

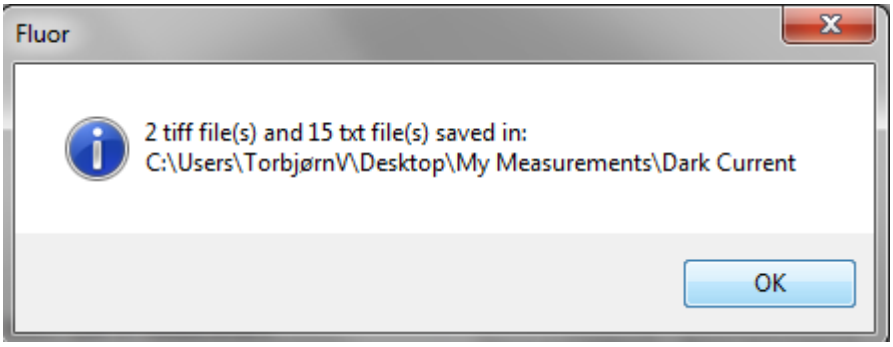


Figure 7.9: The user is informed when all files are acquired. In this example, 2 tiff files and 15 text-files is saved in C:\Users\Torbjørn\Desktop\My Measurements\Dark Current

When all measurements are done, acquisition is stopped by clicking 'Stop Acquisition' (surrounded by a green rectangle in Figure 7.7). Measurements data are saved in corresponding folders in the folder specified by The program is closed by clicking 'Exit' (surrounded by a brown rectangle in Figure 7.7).

There are several error messages in the program, trying to prevent the user to do something he/she doesn't want to do. Examples of error-messages are found in Figure 7.10.

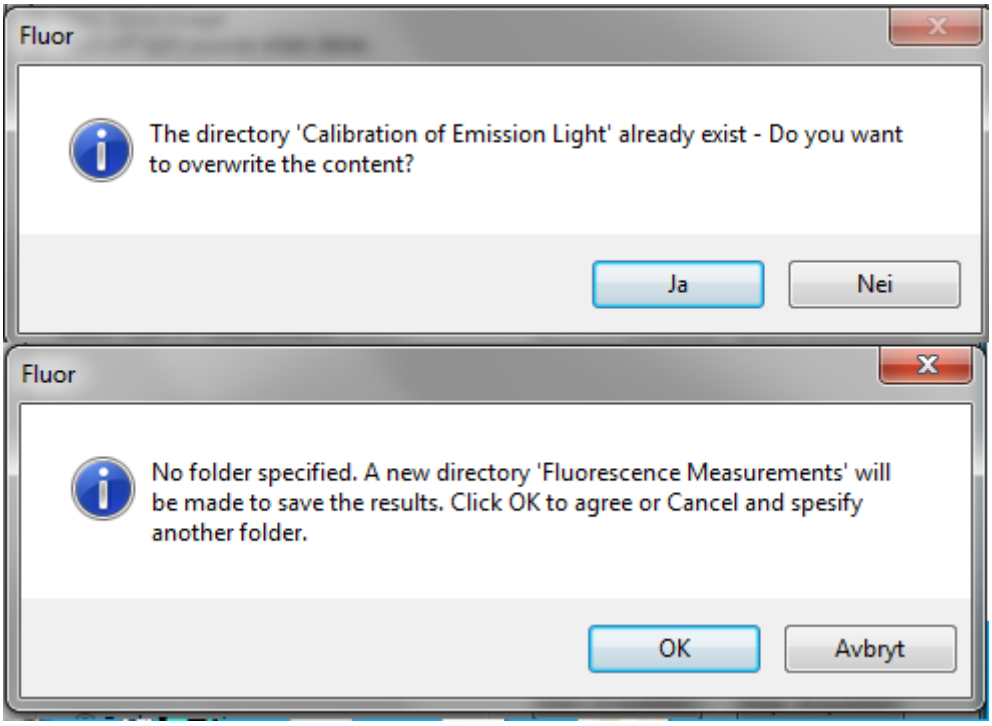


Figure 7.10: Two examples of error messages.

MATLAB scripts

When the image data is acquired by 'Fluor', all measurement data are saved in corresponding folders as showed in 7.5a. The MATLAB scripts have to be saved in the folder 'My Measurements' to work properly. 20 samples are investigated in the explanation of this section. The data of the 20 samples are saved in 'My Measurements/[number of measurement]/'. Measurements of sample 12 is for example saved in the target folder 'My Measurements/12/'. See Figure 7.11 for illustration. There are at most 10 measurements for each sample, where the folders of all these measurements are illustrated in 7.5a. However, some measurements, i.e. Dark Current and calibration measurements are not repeated for each sample. The user has to specify inside the script which sample folder that contain dark current and calibration measurements, and for which samples these measurements apply. In order to let the scripts work properly, they have to be stored in the folder 'My Measurement', and, 'My Measurement' has to be selected as the 'Current Folder' in MATLAB.

The user has to fill in information in the script 'userInput.m', before the scripts are run. This script is attached below with values of this report filled in, but these values are changed by the user for her/his excercise.

```
%userInput.m
%The user have to fill in relevant information in this script, in order to
```

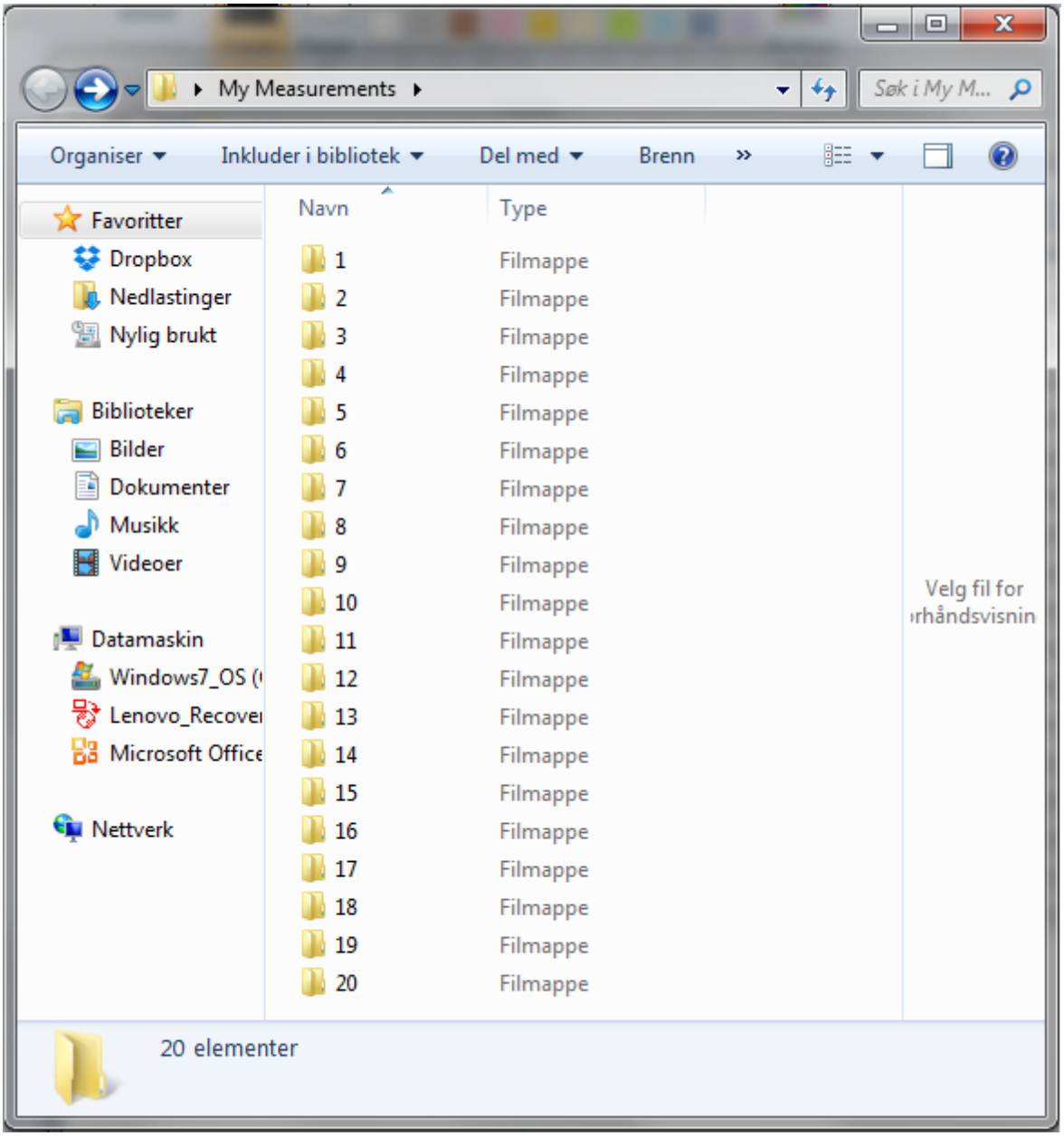


Figure 7.11: 20 different sample

```
%get the intrinsic fluorescence to be calculated

sampleFolders=1:20; %Number of samples illuminated
numberofImages=15; %number of text files stored per measurement
R_gs_ex=0.454; %Gray standard reflectance at exitation wavelength
R_gs_em=0.470; %Gray standard reflectance at emission wavelength
calibrationFolders=[1 2 3 4 5]; %Sample folders which contain calibration
measurements
DCFolders=[1 11]; %Sample folders which contain Dark Current measurements
calibrationFoldersMatrix=[1 6 11 16;2 7 12 17;3 8 13 18;4 9 14 19;5 10 15 20];
%Element i of calibrationFolders is the calibration of samples of row number i of
this matrix. Hence number of rows in this matrix has to be equal to number of elements in calib
DCFoldersMatrix=[1:10;11:20]; %Element i of DCFolders is the dark current
measurement for the elements in row number of row i in this matrix. Hence number of rows in thi
averageRows=260:780; %Which rows to be averaged
averageColumns=348:1044; %Which columns to be averaged
n_factor=0.0527; %Calculated by another script

t_c=30; %exposure-time in ms used in aqcuisition of calibration images
t_f=6.8; %exposure-time in ms used in aqcuisition of fluorescence images
P_ex=400; %Power of excitation beam measured by powermeter expressed in microW
P_em=43.1; %Power of emission beam measured by powermeter expressed in microW

u_a=[0.012 0.26 0.37 1.4; 0.012 0.26 0.37 1.4;
0.012 0.26 0.37 1.4; 0.012 0.26 0.37 1.4]; %Matrix of absorption coefficients.
Each coefficient corresponds to the sample in the same matrix position as in the matrix calibra
u_s_em=8.30;%Scattering coefficient at the wavelength of emission light [1/cm]
u_s_ex=10;%Scattering coefficient at the wavelength of excitation light [1/cm]

%Nonscattering paramteres (not used for calculations of scattering samples)
n_i = 1.33; %Refractive index of slab (water)
n_t = 1; %Refractive index of air
theta_t=34; %Angle of incident UV light relativ to surface sample normal in
degrees.
theta_i = asin((n_i/n_t)*sin(theta_t)); %Angle of UV light within the slab,
related to theta_t by Snell's law
L = 1.4; % Depth [cm] (thickness) of slab.

save('userInput.mat');
```

The script 'LoadingAndSaveofData.m' is attached below, and consists of three scripts: 'getAverage.m', 'extractDC.m' and 'saveVariables.m'. The code of these scripts is attached in the next script. 'getAverage.m' calculates the average of the text-files saved per measurement, 'extractDC.m' extracts the dark current noise from every text-file and

'saveVariables.m' saves the data into variables.

After 'LoadingAndSaveofData.m' is run, the script 'calibrationFactors.m' has to be run. These script calculates the α-values for excitation and emission wavelength.

The user can either calculate the intrinsic fluorescence of scattering or non-scattering media, by running either 'intrinsic_scat.m' or 'intrinsic_nonscat.m'.

The MATLAB scripts 'userInput.m', 'LoadingAndSaveofData.m', 'intrinsic_scat.m' and 'intrinsic_nonscat.m' together with the software application 'Fluor' is a part of a software package made for this report which is valid for any setup specified by the user in 'userInput.m'. Two other scripts are attached in the next section as well, i.e. 'SpectroPlot.m' and 'CameraPlot.m', which are scripts used to plot some of the figures in chapter 4.

7.3.2 Code

7.3.2.1 Matlab

```
%LoadingAndSaveofData.m

close all;
clear all;

getAverage;
extractDC;
saveVariables;

%getAverage.m


clear all;
close all;

%-----------------Information filled in by user-------------------------

load('userInput.mat');
%-----------------------------------------------------------------------

for j=1:20

    path1=strcat('./',num2str(j),'/Exitation Diffuse Reflectance/Result0.txt');
    path2=strcat('./',num2str(j),'/Emission Diffuse Reflectance/Result0.txt');
    path3=strcat('./',num2str(j),'/Fluorescence without filter/Result0.txt');
    path4=strcat('./',num2str(j),'/Fluorescence with filter/Result0.txt');
    path5=strcat('./',num2str(j),'/Extra/Result0.txt');

    avg1=dlmread(path1);
```

```
    avg2=dlmread(path2);
    avg3=dlmread(path3);
    avg4=dlmread(path4);
    avg5=dlmread(path5);


    for i=1:numberofImages-1

        path1=strcat('./',num2str(j),'/Exitation Diffuse Reflectance/Result',
num2str(i),'.txt');
        path2=strcat('./',num2str(j),'/Emission Diffuse Reflectance/Result',
num2str(i),'.txt');
        path3=strcat('./',num2str(j),'/Fluorescence without filter/Result',
num2str(i),'.txt');
        path4=strcat('./',num2str(j),'/Fluorescence with filter/Result',
num2str(i),'.txt');
        path5=strcat('./',num2str(j),'/Extra/Result',num2str(i),'.txt');


        avg1=avg1+dlmread(path1);
        avg2=avg2+dlmread(path2);
        avg3=avg3+dlmread(path3);
        avg4=avg4+dlmread(path4);
        avg5=avg5+dlmread(path5);

    end

    avg1=avg1/numberofImages;
    avg2=avg2/numberofImages;
    avg3=avg3/numberofImages;
    avg4=avg4/numberofImages;
    avg5=avg5/numberofImages;


    path1=strcat('./',num2str(j),'/Exitation Diffuse Reflectance/Average.txt');
    path2=strcat('./',num2str(j),'/Emission Diffuse Reflectance/Average.txt');
    path3=strcat('./',num2str(j),'/Fluorescence without filter/Average.txt');
    path4=strcat('./',num2str(j),'/Fluorescence with filter/Average.txt');
    path5=strcat('./',num2str(j),'/Extra/Average.txt');

    dlmwrite(path1,avg1);
    dlmwrite(path2,avg2);
    dlmwrite(path3,avg3);
    dlmwrite(path4,avg4);
```

```
    dlmwrite(path5,avg5);

    j

end

for j=calibrationFolders

    path1=strcat('./',num2str(j),'/Calibration of Emission Light/Result0.txt');
    path2=strcat('./',num2str(j),'/Calibration of Exitation Light/Result0.txt');
    path3=strcat('./',num2str(j),'/Calibration of Exitation Light with filter/Result0

    avg1=dlmread(path1);
    avg2=dlmread(path2);
    avg3=dlmread(path3);

    for i=1:numberofImages-1

        path1=strcat('./',num2str(j),'/Calibration of Emission Light/Result',num2str(
        path2=strcat('./',num2str(j),'/Calibration of Exitation Light/Result',num2str(
        path3=strcat('./',num2str(j),'/Calibration of Exitation Light with filter/Res


        avg1=avg1+dlmread(path1);
        avg2=avg2+dlmread(path2);
        avg3=avg3+dlmread(path3);

    end

    avg1=avg1/numberofImages;
    avg2=avg2/numberofImages;
    avg3=avg3/numberofImages;


    path1=strcat('./',num2str(j),'/Calibration of Emission Lighte/Average.txt');
    path2=strcat('./',num2str(j),'/Calibration of Exitation Light/Average.txt');
    path3=strcat('./',num2str(j),'/Calibration of Exitation Light with filter/Average

    dlmwrite(path1,avg1);
    dlmwrite(path2,avg2);
    dlmwrite(path3,avg3);

end
```

```
for j=DCFolders

    path1=strcat('./',num2str(j),'/Dark Current/Result0.txt');
    path2=strcat('./',num2str(j),'/Dark Current 1/Result0.txt');

    avg1=dlmread(path1);
    avg2=dlmread(path2);

    for i=1:numberofImages-1

        path1=strcat('./',num2str(j),'/Dark Current/Result',num2str(i),'.txt');
        path2=strcat('./',num2str(j),'/Dark Current 1/Result',num2str(i),'.txt');

        avg1=avg1+dlmread(path1);
        avg2=avg2+dlmread(path2);
    end

    avg1=avg1/numberofImages;
    avg2=avg2/numberofImages;


    path1=strcat('./',num2str(j),'/Dark Current/Average.txt');
    path2=strcat('./',num2str(j),'/Dark Current 1/Average.txt');

    dlmwrite(path1,avg1);
    dlmwrite(path2,avg2);

end

%saveVariables.m

%The measurement data is saved in a mxn matrix. m and n are specified by
%the user in userInput.m. m is the number of calibration measurements,
%while n is the number of sample measurements done for each calibration

load('userInput.mat');

for i=1:length(calibrationFolders)

    cal_path1=strcat('./',num2str(calibrationFolders(i)),'/Calibration of Exitation Light/Avera
    cal_path2=strcat('./',num2str(calibrationFolders(i)),'/Calibration of Exitation Light with
    cal_path3=strcat('./',num2str(calibrationFolders(i)),'/Calibration of Emission Light/Averag

    cal_var1=dlmread(cal_path1);
```

```
    cal_var2=dlmread(cal_path2);
    cal_var3=dlmread(cal_path3);

    cal_ex=mean(mean(cal_var1(averageRows,averageColumns)))/R_gs_ex;
    cal_ex_fil=mean(mean(cal_var2(averageRows,averageColumns)))/R_gs_ex;
    cal_em=mean(mean(cal_var3(averageRows,averageColumns)))/R_gs_em;

    for j=calibrationFoldersMatrix(i,:)
        path1=strcat('./',num2str(j),'/Emission Diffuse Reflectance/AverageminusDC.tx
        path2=strcat('./',num2str(j),'/Exitation Diffuse Reflectance/AverageminusDC.t
        path3=strcat('./',num2str(j),'/Fluorescence without filter/AverageminusDC.txt
        path4=strcat('./',num2str(j),'/Fluorescence with filter/AverageminusDC.txt');
        path5=strcat('./',num2str(j),'/Extra/AverageminusDC.txt');

        var1=dlmread(path1);
        var2=dlmread(path2);
        var3=dlmread(path3);
        var4=dlmread(path4);
        var5=dlmread(path5);


        d_em(i,j)=mean(mean(var1(averageRows,averageColumns)))/cal_em;
        d_ex(i,j)=mean(mean(var2(averageRows,averageColumns)))/cal_ex_fil;
        f(i,j)=mean(mean(var3(averageRows,averageColumns)));
        f_fil(i,j)=mean(mean(var4(averageRows,averageColumns)));
        extra(i,j)=mean(mean(var5(averageRows,averageColumns)));
        cal_ex(i,j)=cal_ex;
    end
end

save('variables.mat');

%calibrationFactors.m

close all;
clear all;

load('userInput.m');


j=0;
for i=CalibrationFolders
    j=j+1
    file_ex=strcat('./',num2str(i),'/Calibration of Exitation Light/AverageminusDC.tx
```

```
    file_em=strcat('./',num2str(i),'/Calibration of Emission Light/AverageminusDC.txt');

    var_ex=dlmread(file_ex);
    var_em=dlmread(file_em);

    sum_ex=sum(var_ex(:));
    sum_em=sum(var_em(:));
    a_ex(j)=(R_gs_ex*P_ex*10^-6)/sum_ex;
    a_em(j)=(R_gs_em*P_em*10^-6)/sum_em;

end

save('calibrationFactors.mat');

%intrinsic_scat.m

Calcultation of intrinsic fluorescence for scattering samples

close all;
clear all;


font=22;
line=3;

userInput;
load('calibrationFactors.mat');
load('variables.mat');

h1=figure;
h2=figure;
h3=figure;


for i=1:length(calibrationFoldersMatrix(:,1))

    switch(i)
        case 1
            color='bs-';
        case 2
            color='gs-';
        case 3
            color='rs-';
        case 4
```

```
            color='cs-';
    end
    for i=1:length(calibrationFoldersMatrix(1,:))

        sigma_ex(i,j)=1./((3*u_a(i,j)*(u_a(i,j)+u_s_ex*(1-g)))^0.5);
        sigma_em(i,j)=1./((3*u_a(i,j)*(u_a(i,j)+u_s_em*(1-g)))^0.5);


        C1_em(i,j)=3.04+4.9*d_em(i,j)-2.06*exp(-21.1*d_em(i,j));
        k1_em(i,j)=1-(1-1/sqrt(3))*exp(-18.9*d_em(i,j));
        C2_em(i,j)=2.04-1.33*d_em(i,j)-2.04*exp(-21.1*d_em(i,j));
        k2_em(i,j)=1.59*exp(3.36*d_em(i,j));
        C3_em(i,j)=0.32+0.72*d_em(i,j)-0.16*exp(-9.11*d_em(i,j));
        k3_em(i,j)=1-0.30*exp(-6.12*d_em(i,j));

        C1_ex(i,j)=3.04+4.9*d_ex(i,j)-2.06*exp(-21.1*d_ex(i,j));
        k1_ex(i,j)=1-(1-1/sqrt(3))*exp(-18.9*d_ex(i,j));
        C2_ex(i,j)=2.04-1.33*d_ex(i,j)-2.04*exp(-21.1*d_ex(i,j));
        k2_ex(i,j)=1.59*exp(3.36*d_ex(i,j));
        C3_ex(i,j)=0.32+0.72*d_ex(i,j)-0.16*exp(-9.11*d_ex(i,j));
        k3_ex(i,j)=1-0.30*exp(-6.12*d_ex(i,j));

        X1D(i,j)= (C1_ex(i,j).*C3_em(i,j))./(k1_ex(i,j)./sigma_ex(i,j) + k3_em(i,j)./

        b(i,j)=(a_em(i)*f(i,j)*t_c*R_gs_ex)/(n_factor*t_f*a_ex(i)*cal_ex(i)*X1D(i,j))




    end
    figure(h1); hold on;
    plot(inkVolum,b(i,1:4),color,'LineWidth',line);
    figure(h2); hold on;
    plot(inkVolum,X1D(i,1:4),color,'LineWidth',line);
    i
end




figure(h1);
l1=legend('25','50','75','100','Location','EastOutside'); xlabel('Ink volum [micro li
set(gca,'FontSize',font);
```

```
v = get(l1,'title');set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);


figure(h2);
l2=legend('25','50','75','100','Location','EastOutside');xlabel('Ink volum [micro liter]','Font
set(gca,'FontSize',font);
v = get(l2,'title');set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);


saveas(h1,'Intrinsic fluorescence','png');
saveas(h2,'X1D','png');

%intrinsic_nonscat.m

%Calcultation of intrinsic fluorescence for non-scattering samples

close all;
clear all;

userInput;
load('calibrationFactors.mat');
load('variables.mat');


%Fresnel reflection coefficient for unpolarized light. R_0 for normal
%incdent, R_theta for theta_i between 0 and theta_critical=arcsin(n_t/n_i)
R_0 = ((n_i-n_t)/(n_i+n_t))^2;
R_theta = 0.5 * ((sin(theta_i-theta_t)/sin(theta_i+theta_t))^2 + (tan(theta_i-theta_t)/tan(thet


X_dil=(L*(1-R_0)*(1-R_theta)*n_t^2*cos(theta_t))/(4*pi*n_i^2*cos(theta_i));

for i=1:length(calibrationFoldersMatrix(:,1))
    for j=1:length(calibrationFoldersMatrix(1,:))
        B(i,j)=(a_em(i)*cos(theta_i)*f(i,j)*t_c*R_gs_ex)/(n_factor*t_f*a_ex(i)*pi*cal_ex(i)*X_d

    end
end

save('Bnonscat.mat');

%CameraPlots.m

%This is an example of a script used to plot one of the figure in the
```

```
%result section of the thesis

clear all;
close all;
load('data.mat');

h1=figure;
h2=figure;
h3=figure;

font=22;
line=1.5;


j=[1 3 4 2];
inkVolum=[0 50 75 250];

R_gs_ex=0.449;
R_gs_em=0.485;


for i=2:5
    switch(i)
        case 2
            color='bs-';
        case 3
            color='gs-';
        case 4
            color='rs-';
        case 5
            color='cs-';
    end
    figure(h1); hold on;
    plot(inkVolum,f(i,j),color,'LineWidth',line);
    figure(h2); hold on;
    plot(inkVolum,R_gs_em*d_em(i,j),color,'LineWidth',line);
    figure(h3); hold on;
    plot(inkVolum,R_gs_ex*d_ex(i,j),color,'LineWidth',line);

end

figure(h1);
l1=legend('25','50','75','100','Location','EastOutside');
xlabel('Ink volum [micro liter]','FontSize',font);
```

```
ylabel('Digital number','FontSize',font);
set(gca,'FontSize',font);
v = get(l1,'title');
set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);
saveas(h1,'Fluorescence without filter','png');

figure(h2);
l2=legend('25','50','75','100','Location','EastOutside');
xlabel('Ink volum [micro liter]','FontSize',font);
ylabel('Reflectance','FontSize',font);
set(gca,'FontSize',font);
v = get(l2,'title');
set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);
saveas(h2,'Emission Diffuse Reflectance','png');

figure(h3);
l3=legend('25','50','75','100','Location','EastOutside');
xlabel('Ink volum [micro liter]','FontSize',font);
ylabel('Reflectance','FontSize',font);
set(gca,'FontSize',font);
v = get(l3,'title');
set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);
saveas(h3,'Exitation Diffuse Reflectance','png');

%SpectroPlots

%This was used to plot and save spectrometerdata for all samples
%c_ij of the report

close all;
clear all;

load('data.mat');
LoadGrayStandardReflectance;
font=22;
line=1.5;

inkVolum=[0 50 75 250];

for j=1:4


    abs1=figure;
    abs2=figure;
```

```
abs3=figure;
abs4=figure;

for i=2:5
    switch(i)
        case 2
            color='b';
        case 3
            color='g';
        case 4
            color='r';
        case 5
            color='c';
    end

    figure(abs1); hold on;
    plot(d_ex(i,j).wavelength,(transpose(refspectra).*(d_ex(i,j).raw-d_ex(i,j).da
    figure(abs2); hold on;
    plot(d_ex(i,j).wavelength,(transpose(refspectra).*(d_em(i,j).raw-d_em(i,j).da
    figure(abs3); hold on;
    plot(d_ex(i,j).wavelength,(d_ex(i,j).raw-d_ex(i,j).dark),color,'LineWidth',li
    figure(abs4); hold on;
    plot(d_em(i,j).wavelength,(d_em(i,j).raw-d_ex(i,j).dark),color,'LineWidth',li


end

name=strcat('Ink volume:  ',num2str(inkVolum(j)),' micro liter');

figure(abs1); title(name,'FontSize',font);
l1=legend('25','50','75','100','Location','EastOutside'); xlabel('Wavelength [nm]
set(gca,'FontSize',font);
axis([274 290 0.10 0.25]);
v = get(l1,'title');set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',fo
saveas(abs1,strcat('Diffuse reflectance exitation light -',name),'png');

figure(abs2); title(name,'FontSize',font);
l2=legend('25','50','75','100','Location','EastOutside'); xlabel('Wavelength [nm]
set(gca,'FontSize',font);
axis([540 620 0.4 0.9]);
v = get(l2,'title');set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',fo
saveas(abs2,strcat('Diffuse reflectance emission light -',name),'png');
```

```
    figure(abs3); title(name,'FontSize',font);
    l3=legend('25','50','75','100','Location','EastOutside'); xlabel('Wavelength [nm]','FontSiz
    set(gca,'FontSize',font);
    axis([450 625 0 3500]);
    v = get(l3,'title');set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);
    saveas(abs3,strcat('Sample illuminated by exitation light -',name),'png');


    figure(abs4); title(name,'FontSize',font);
    l4=legend('25','50','75','100','Location','EastOutside'); xlabel('Wavelength [nm]','FontSiz
    set(gca,'FontSize',font);
    v = get(l4,'title');set(v,'string',{'Fluorescein', '[micro liter]'},'FontSize',font);
    saveas(abs4,strcat('Sample illuminated by emission light -',name),'png');



end


abs5=figure;
for k=1:5
    figure(abs5); hold on;
    plot(cal_em(k).wavelength,(cal_em(k).raw-cal_em(k).dark),'b');
end

figure(abs5); title(strcat('Gs illuminated by emission light, absolute values. ',num2str(5),' c
saveas(abs5,strcat('Gs illuminated by emission light, absolute values'),'png');

abs6=figure;
for k=1:5
    figure(abs6); hold on;
    plot(cal_ex(k).wavelength,(cal_ex(k).raw-cal_ex(k).dark),'b');
end
```

7.3.2.2 Fluor

```
// fluorDlg.h : header file
//

#pragma once
#include <Jai_Factory.h>
#include "afxeditbrowsectrl.h"
```

```
#include "afxwin.h"
#include "afxcmn.h"
#include <direct.h>


// CfluorDlg dialog
class CfluorDlg : public CDialogEx
{
// Construction
public:
        CfluorDlg(CWnd* pParent = NULL);          // standard
            constructor

// Dialog Data
        enum { IDD = IDD_FLUOR_DIALOG };

        protected:
        virtual void DoDataExchange(CDataExchange* pDX);
            // DDX/DDV support


// Implementation
protected:
        HICON m_hIcon;

        // Generated message map functions
        virtual BOOL OnInitDialog();
        afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
        afx_msg void OnPaint();
        afx_msg HCURSOR OnQueryDragIcon();
        DECLARE_MESSAGE_MAP()

public:
        FACTORY_HANDLE m_hFactory; // Factory Handle
        CAM_HANDLE m_hCamera; // Camera Handle
        VIEW_HANDLE g_hView;
        THRD_HANDLE m_hThread;
    NODE_HANDLE m_hGainNode;
    NODE_HANDLE m_hExposureNode;
        int8_t m_CameraId[J_CAMERA_ID_SIZE];
        RECT m_RectToMeasure;
        FILE *OutputFile;
        int64_t m_pixelformat;
        int32_t maks;
```

```
        J_tIMAGE_INFO m_CnvImageInfo;
        int64_t width;
        int64_t height;
        bool m_bSave;
        bool m_busy;
        CString buff;
        PIXELVALUE m_PixelValue;
        uint32_t          m_PixelType;
        CRITICAL_SECTION     m_CriticalSection;

        BOOL OpenFactoryAndCamera();
        void CloseFactoryAndCamera();
    void ShowErrorMsg(CString message, J_STATUS_TYPE error);
        void StreamCBFunc(J_tIMAGE_INFO * pAqImageInfo);
        void CfluorDlg::InitializeControls();
        void CfluorDlg::updateInformation(void);
        CString instruction;
        int m_nLEDType;
        int numberOfTextImages;
        int numberOfTiffImages;
        int64_t txtNum;
        int tiffNum;
        CString dir;
        CString path;

        CMFCEditBrowseCtrl yo;
        CEdit m_EditCtrlText;
        CEdit m_EditCtrlTiff;
        CStatic m_cameraidStatic;
        CStatic m_instructionStatic;
        CStatic m_maxvalueStatic;
        afx_msg void OnBnClickedButton1();
        afx_msg void OnBnClickedButton2();
        afx_msg void OnBnClickedButton3();
    afx_msg void OnHScroll(UINT nSBCode, UINT nPos, CScrollBar*
       pScrollBar);
        CSliderCtrl CSCtrlGain;
        CSliderCtrl CSCtrlExposure;

        afx_msg void OnTimer(UINT_PTR nIDEvent);
        afx_msg void OnBnClickedExit();
        afx_msg void OnBnClickedStart();
};
```

```
// fluorDlg.cpp : implementation file
//

#include "stdafx.h"
#include "fluor.h"
#include "fluorDlg.h"
#include "afxdialogex.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// CAboutDlg dialog used for App About

class CAboutDlg : public CDialogEx
{
public:
        CAboutDlg();

// Dialog Data
        enum { IDD = IDD_ABOUTBOX };

        protected:
        virtual void DoDataExchange(CDataExchange* pDX);    //
            DDX/DDV support

// Implementation
protected:
        DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialogEx::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
```

```
END_MESSAGE_MAP()


// CfluorDlg dialog



CfluorDlg::CfluorDlg(CWnd* pParent /*=NULL*/)
        : CDialogEx(CfluorDlg::IDD, pParent)
{
        m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

        m_hFactory = NULL;
    m_hCamera = NULL;
    g_hView = NULL;
    m_hThread = NULL;
        m_CnvImageInfo.pImageBuffer = NULL;
        m_hExposureNode = NULL;
        m_bSave=false;
        m_busy=false;
        maks=0;
        numberOfTextImages=15;
        numberOfTiffImages=1;
        txtNum=numberOfTextImages;
        tiffNum=numberOfTiffImages;
        OutputFile=NULL;
        m_nLEDType=0;
        // We use a critical section to protect the update of
           the pixel value display
    InitializeCriticalSection(&m_CriticalSection);

}

void CfluorDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialogEx::DoDataExchange(pDX);
        DDX_Control(pDX, IDC_MFCEDITBROWSE1, yo);
        DDX_Control(pDX, IDC_NUM_TEXT_EDIT, m_EditCtrlText);
        DDX_Control(pDX, IDC_NUM_TIFF_EDIT, m_EditCtrlTiff);
        DDX_Control(pDX, IDC_STATIC6, m_cameraidStatic);
        DDX_Control(pDX, IDC_STATIC7, m_instructionStatic);
        DDX_Control(pDX, IDC_STATIC8, m_maxvalueStatic);
        DDX_Control(pDX, IDC_SLIDER_GAIN, CSCtrlGain);
```

```
        DDX_Control(pDX, IDC_SLIDER_EXPOSURE, CSCtrlExposure);
        DDX_Radio(pDX, IDC_RADIO1, m_nLEDType);
}

BEGIN_MESSAGE_MAP(CfluorDlg, CDialogEx)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_HSCROLL()
        ON_WM_TIMER()
        ON_BN_CLICKED(IDC_BUTTON1, &CfluorDlg::
            OnBnClickedButton1)
        ON_BN_CLICKED(IDC_BUTTON3, &CfluorDlg::
            OnBnClickedButton3)
        ON_BN_CLICKED(IDC_EXIT, &CfluorDlg::OnBnClickedExit)
        ON_BN_CLICKED(IDC_START, &CfluorDlg::OnBnClickedStart)
END_MESSAGE_MAP()


// CfluorDlg message handlers

BOOL CfluorDlg::OnInitDialog()
{
        CDialogEx::OnInitDialog();

        // Add "About..." menu item to system menu.

        // IDM_ABOUTBOX must be in the system command range.
        ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);

        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
                BOOL bNameValid;
                CString strAboutMenu;
                bNameValid = strAboutMenu.LoadString(
                    IDS_ABOUTBOX);
                ASSERT(bNameValid);
                if (!strAboutMenu.IsEmpty())
                {
                        pSysMenu->AppendMenu(MF_SEPARATOR);
                        pSysMenu->AppendMenu(MF_STRING,
                            IDM_ABOUTBOX, strAboutMenu);
                }
```

```
        }

        // Set the icon for this dialog.  The framework does
            this automatically
        //  when the application's main window is not a dialog
        SetIcon(m_hIcon, TRUE);                     // Set big icon
        SetIcon(m_hIcon, FALSE);                    // Set small
            icon

        yo.EnableFolderBrowseButton();

        if(OpenFactoryAndCamera())
        {
                m_cameraidStatic.SetWindowText(CString(
                    m_CameraId));
        }
        else return FALSE;

//      updateInformation();
        yo.EnableFolderBrowseButton();

        // TODO: Add extra initialization here

        SetTimer(1, 100, NULL);

        CfluorDlg::InitializeControls();

        return TRUE;  // return TRUE  unless you set the focus
            to a control
}

void CfluorDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialogEx::OnSysCommand(nID, lParam);
        }
}
```

```
// If you add a minimize button to your dialog, you will need
    the code below
//  to draw the icon.  For MFC applications using the document/
    view model,
//  this is automatically done for you by the framework.

void CfluorDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for
                    painting

                SendMessage(WM_ICONERASEBKGND, reinterpret_cast<
                    WPARAM>(dc.GetSafeHdc()), 0);

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialogEx::OnPaint();
        }
}

// The system calls this function to obtain the cursor to
    display while the user drags
//  the minimized window.
HCURSOR CfluorDlg::OnQueryDragIcon()
{
        return static_cast<HCURSOR>(m_hIcon);
}

BOOL CfluorDlg::OpenFactoryAndCamera() {
        J_STATUS_TYPE    status = J_ST_SUCCESS;
        bool8_t                 bHasChanged = false;
```

```
uint32_t                iNumOfCameras = 0;
uint32_t                iCameraIdSize = J_CAMERA_ID_SIZE
    ;

//Open factory
status = J_Factory_Open("", &m_hFactory);
if(status != J_ST_SUCCESS)
{
        ShowErrorMsg(CString("Unable to open factory!"),
            status);
        return FALSE;
}

//Look for cameras
status = J_Factory_UpdateCameraList(m_hFactory, &
    bHasChanged);
if(status != J_ST_SUCCESS)
{
        ShowErrorMsg(CString("Unable to update camera
            list!"), status);
        return FALSE;
}

//Get number of cameras
status = J_Factory_GetNumOfCameras(m_hFactory, &
    iNumOfCameras);
if(status != J_ST_SUCCESS)
{
        ShowErrorMsg(CString("Unable to get number of
            cameras found!"), status);
        return FALSE;
}

//Get the first camera information
if(iNumOfCameras == 0)
{
        AfxMessageBox(_T("No camera found!"),
            MB_OKCANCEL | MB_ICONINFORMATION);
        return FALSE;
}
else if(iNumOfCameras > 0)
{
        status = J_Factory_GetCameraIDByIndex(m_hFactory
            , 0, m_CameraId, &iCameraIdSize);
```

```
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to get
                            camera ID!"), status);
                        return FALSE;
                }

                status = J_Camera_Open(m_hFactory, m_CameraId, &
                    m_hCamera);
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to open
                            camera!"), status);
                        return FALSE;
                }
        }

        return TRUE;


        }

void CfluorDlg::CloseFactoryAndCamera()
{
        J_STATUS_TYPE    status = J_ST_SUCCESS;

        if(m_hCamera)
        {
                status = J_Camera_Close(m_hCamera);
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to close
                            camera!"), status);
                        return;
                }
                m_hCamera = NULL;
        }

        if(m_hFactory)
        {
                status = J_Factory_Close(m_hFactory);
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to close
```

```
                    factory!"), status);
                return;
            }
            m_hFactory = NULL;
        }

    // Stop UI update timer
    KillTimer(1);
}

void CfluorDlg::ShowErrorMsg(CString message, J_STATUS_TYPE
    error)
{
    CString errorMsg;

    errorMsg.Format(_T("%s: Error = %d: "), message, error);

    switch(error)
    {
    case J_ST_INVALID_BUFFER_SIZE:      errorMsg += "Invalid
        buffer size ";                    break;
    case J_ST_INVALID_HANDLE:           errorMsg += "Invalid
        handle ";                         break;
    case J_ST_INVALID_ID:                     errorMsg += "
        Invalid ID ";                                 break
        ;
    case J_ST_ACCESS_DENIED:            errorMsg += "Access
        denied ";                        break;
    case J_ST_NO_DATA:                        errorMsg += "No
        data ";                                   break;
    case J_ST_ERROR:                          errorMsg += "
        Generic error ";                    break;
    case J_ST_INVALID_PARAMETER:        errorMsg += "Invalid
        parameter ";                      break;
    case J_ST_TIMEOUT:                        errorMsg += "
        Timeout ";                                  break
        ;
    case J_ST_INVALID_FILENAME:         errorMsg += "Invalid
        file name ";                      break;
    case J_ST_INVALID_ADDRESS:          errorMsg += "Invalid
        address ";                        break;
    case J_ST_FILE_IO:                        errorMsg += "
        File IO error ";                    break;
    case J_ST_GC_ERROR:                       errorMsg += "
```

```
        GenICam error ";                                    break;
    case J_ST_VALIDATION_ERROR:          errorMsg += "Settings
        File Validation Error ";          break;
    case J_ST_VALIDATION_WARNING:        errorMsg += "Settings
        File Validation Warning ";    break;
    }

    AfxMessageBox(errorMsg, MB_OKCANCEL | MB_ICONINFORMATION);

}

void CfluorDlg::OnBnClickedButton1()
{
        m_bSave=true;
        txtNum=0;
        tiffNum=0;
        CString temp;
        m_EditCtrlText.GetWindowText(temp);
        numberOfTextImages=_ttoi(temp);
        m_EditCtrlTiff.GetWindowText(temp);
        numberOfTiffImages=_ttoi(temp);
}

void CfluorDlg::OnBnClickedStart()
{
        J_STATUS_TYPE   retval;
        THRD_HANDLE hThread;
        NODE_HANDLE hNode;
        SIZE ViewSize = {100, 100};
        POINT TopLeft = {0, 0};
        int64_t pixelFormat;
        int64_t nodeValue;

        // Get Width from the camera
                if (J_Camera_GetValueInt64(m_hCamera, "Width", &
                    width) != J_ST_SUCCESS) {
                    AfxMessageBox(_T("Cannot get width"),
                    MB_OKCANCEL | MB_ICONINFORMATION); return;}
        ViewSize.cx = (LONG)width; // Set window size cx

        // Get Height from the camera
        if (J_Camera_GetValueInt64(m_hCamera, "Height", &height)
            != J_ST_SUCCESS)
        AfxMessageBox(_T("Cannot get Heigth"), MB_OKCANCEL |
```

```
        MB_ICONINFORMATION);
    ViewSize.cy = (LONG)height;//(LONG)height; // Set window
        size cy

    // Open view window
    if (J_Image_OpenViewWindow(_T("Live view!"),
    &TopLeft,
    &ViewSize,
    &g_hView) != J_ST_SUCCESS)
    AfxMessageBox(_T("Cannot open view window"), MB_OKCANCEL
        | MB_ICONINFORMATION);

// Get pixelformat from the camera
retval = J_Camera_GetValueInt64(m_hCamera, "PixelFormat", &
    pixelFormat);
if (retval != J_ST_SUCCESS)
{
    ShowErrorMsg(CString("Unable to get PixelFormat value!")
        , retval);
    return;
}

// Calculate number of bits (not bytes) per pixel using
    macro
int bpp = J_BitsPerPixel(pixelFormat);



// Open stream
retval = J_Image_OpenStream(m_hCamera, 0, reinterpret_cast<
    J_IMG_CALLBACK_OBJECT>(this), reinterpret_cast<
    J_IMG_CALLBACK_FUNCTION>(&CfluorDlg::StreamCBFunc), &
    m_hThread, (ViewSize.cx*ViewSize.cy*bpp)/8);
if (retval != J_ST_SUCCESS) {
    ShowErrorMsg(CString("Could not open stream!"), retval);
    return;
}

    retval = J_Camera_ExecuteCommand(m_hCamera, "
        AcquisitionStart");
    if(retval != J_ST_SUCCESS)
    {
            ShowErrorMsg(CString("Unable to Start
                Acquisition!"), retval);
```

```
                return;
        }

        GetDlgItem(IDC_BUTTON1)->EnableWindow(1);
}

void CfluorDlg::StreamCBFunc(J_tIMAGE_INFO * pAqImageInfo)
{
        J_STATUS_TYPE status = J_ST_SUCCESS;
        int exposurevalue;

        // allocate memory for converted image only once per
            test
        if (m_CnvImageInfo.pImageBuffer == NULL)
        {
                status = J_Image_Malloc(pAqImageInfo, &
                    m_CnvImageInfo);
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to Malloc
                            image buffer!"),status);
                        return;
                }
        }

        //convert raw to tiff
        status = J_Image_FromRawToImage(pAqImageInfo, &
            m_CnvImageInfo);
        if(status != J_ST_SUCCESS)
        {
                ShowErrorMsg(CString("Unable to convert image
                    from Raw to Image format!"),status);
                return;
        }

        if(g_hView && (pAqImageInfo->iAwaitDelivery < 3))
        {
                status = J_Image_ShowImage(g_hView, &
                    m_CnvImageInfo);
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to show
                            image!"),status);
                        return;
```

```
        }
}


//Calculating maximum value of image
POINT pt;
int tempmaks=0;
PIXELVALUE pixelValue;
CString valueString;

for(uint32_t i = 0; i < 1040; i++)
        {
                for(uint32_t j = 0; j < 1392; j++)
                {
                pt.x = i;
                pt.y = j;
    if (J_Image_GetPixel(&m_CnvImageInfo, &pt, &
       pixelValue) == J_ST_SUCCESS)
    {
        EnterCriticalSection(&m_CriticalSection);
        m_PixelType = m_CnvImageInfo.iPixelType;
                        if(pixelValue.PixelValueUnion.
                           Mono8Type.Value>tempmaks)
                           tempmaks=pixelValue.
                           PixelValueUnion.Mono8Type.
                           Value;
                        LeaveCriticalSection(&
                           m_CriticalSection);
                }
        }
}
maks=tempmaks;

//Check if 'Save image' is pushed
if(m_bSave || txtNum<numberOfTextImages || tiffNum<
   numberOfTiffImages ) {

        if (m_bSave) {
                yo.GetWindowText(buff);
                m_busy=true;

                if (buff.IsEmpty()) {
                        if(AfxMessageBox(_T("No␣folder␣
                           specified.␣A␣new␣directory␣'
```

```
                Fluorescence Measurements'
                will be made to save the
                results. Click OK to agree
                or Cancel and spesify
                another folder."),
                MB_OKCANCEL |
                MB_ICONINFORMATION)==IDOK) {
                    TCHAR NPath[MAX_PATH];
                    GetCurrentDirectory(
                        MAX_PATH, NPath);
                    buff.Format(_T("%s%s"),
                        NPath,_T("\\
                        Fluorescence
                        Measurements"));
                    CreateDirectory(buff,
                        NULL);
                }
                else {
                    m_bSave=false;
                    txtNum=
                        numberOfTextImages;
                    tiffNum=
                        numberOfTiffImages;
                    return;
                }
            }

            switch(m_nLEDType){
                case 0:
                    path.Format(_T("%s%s"),
                        buff,_T("\\Dark
                        Current"));
                    CreateDirectory(path,
                        NULL);
                    if(GetLastError()==
                        ERROR_ALREADY_EXISTS
                        ) if(AfxMessageBox(
                        _T("The directory '
                        Dark Current
                        Directory' already
                        exist - Do you want
                        to overwrite the
                        content?"), MB_YESNO
                        |
```

```
                MB_ICONINFORMATION)
                !=IDYES) return;
            break;
        case 1:
            path.Format(_T("%s%s"),
                buff,_T("\\
                Calibration of
                Emission Light"));
            CreateDirectory(path,
                NULL);
            if(GetLastError()==
                ERROR_ALREADY_EXISTS
                ) if(AfxMessageBox(
                _T("The directory '
                Calibration of
                Emission Light '
                already exist - Do
                you want to
                overwrite the
                content?"), MB_YESNO
                  |
                MB_ICONINFORMATION)
                !=IDYES)      return;
            break;
        case 2:
            path.Format(_T("%s%s"),
                buff,_T("\\
                Calibration of
                Exitation Light"));
            CreateDirectory(path,
                NULL);
            if(GetLastError()==
                ERROR_ALREADY_EXISTS
                ) if(AfxMessageBox(
                _T("The directory '
                Calibration of
                Exitation Light '
                already exist - Do
                you want to
                overwrite the
                content?"), MB_YESNO
                  |
                MB_ICONINFORMATION)
                !=IDYES) return;
```

```
                break;
            case 3:
                path.Format(_T("%s%s"),
                    buff,_T("\\
                    Calibration␣of␣
                    Exitation␣Light␣with
                    ␣filter"));
                CreateDirectory(path,
                    NULL);
                if(GetLastError()==
                    ERROR_ALREADY_EXISTS
                    ) if(AfxMessageBox(
                    _T("The␣directory␣'
                    Calibration␣of␣
                    Exitation␣Light␣with
                    ␣filter'␣already␣
                    exist␣-␣Do␣you␣want␣
                    to␣overwrite␣the␣
                    content?"), MB_YESNO
                    |
                    MB_ICONINFORMATION)
                    !=IDYES)
                    return;
                break;
            case 4:
                path.Format(_T("%s%s"),
                    buff,_T("\\Exitation
                    ␣Diffuse␣Reflectance
                    "));
                CreateDirectory(path,
                    NULL);
                if(GetLastError()==
                    ERROR_ALREADY_EXISTS
                    ) if(AfxMessageBox(
                    _T("The␣directory␣'
                    Exitation␣Diffuse␣
                    Reflectance'␣already
                    ␣exist␣-␣Do␣you␣want
                    ␣to␣overwrite␣the␣
                    content?"), MB_YESNO
                    |
                    MB_ICONINFORMATION)
                    !=IDYES)      return;
                break;
```

```
case 5:
        path.Format(_T("%s%s"),
            buff,_T("\\Emission␣
            Diffuse␣Reflectance"
            ));
        CreateDirectory(path,
            NULL);
        if(GetLastError()==
            ERROR_ALREADY_EXISTS
            ) if(AfxMessageBox(
            _T("The␣directory␣'
            Emission␣Diffuse␣
            Reflectance'␣already
            ␣exist␣-␣Do␣you␣want
            ␣to␣overwrite␣the␣
            content?"), MB_YESNO
              |
            MB_ICONINFORMATION)
            !=IDYES)        return
            ;
        break;
case 6:
        path.Format(_T("%s%s"),
            buff,_T("\\
            Fluorescence␣without
            "));
        CreateDirectory(path,
            NULL);
        if(GetLastError()==
            ERROR_ALREADY_EXISTS
            ) if(AfxMessageBox(
            _T("The␣directory␣'
            Fluorescence'␣
            already␣exist␣-␣Do␣
            you␣want␣to␣
            overwrite␣the␣
            content?"), MB_YESNO
              |
            MB_ICONINFORMATION)
            !=IDYES)        return
            ;
        break;
case 7:
        path.Format(_T("%s%s"),
```

```
                buff,_T("\\
                Fluorescence_with_
                filter"));
            CreateDirectory(path,
                NULL);
            if(GetLastError()==
                ERROR_ALREADY_EXISTS
                ) if(AfxMessageBox(
                _T("The_directory_'
                Fluorescence_with_
                filter'_already_
                exist_-_Do_you_want_
                to_overwrite_the_
                content?"), MB_YESNO
                  |
                MB_ICONINFORMATION)
                !=IDYES)  return;
            break;
        case 8:
            path.Format(_T("%s%s"),
                buff,_T("\\Dark_
                Current_1"));
            CreateDirectory(path,
                NULL);
            if(GetLastError()==
                ERROR_ALREADY_EXISTS
                ) if(AfxMessageBox(
                _T("The_directory_'
                Dark_Current_1'_
                already_exist_-_Do_
                you_want_to_
                overwrite_the_
                content?"), MB_YESNO
                  |
                MB_ICONINFORMATION)
                !=IDYES)  return;
            break;
        case 9:
            path.Format(_T("%s%s"),
                buff,_T("\\Extra"));
            CreateDirectory(path,
                NULL);
            if(GetLastError()==
                ERROR_ALREADY_EXISTS
```

```
                                    ) if(AfxMessageBox(
                                    _T("The directory '
                                    Extra' already exist
                                    - Do you want to
                                    overwrite the
                                    content?"), MB_YESNO
                                     |
                                    MB_ICONINFORMATION)
                                    !=IDYES)       return;
                                break;
        }
}
m_bSave=false;
CString MsgStr;

//save as tiff
if(tiffNum<numberOfTiffImages) {
        dir.Format(_T("%s%s%d"), path,_T("\\
            Result"),txtNum);
        if(J_Image_SaveFileEx(&m_CnvImageInfo,
            dir+".tiff", J_FF_TIFF) !=
            J_ST_SUCCESS) AfxMessageBox(_T("
            Cannot safe tiff file!"),
            MB_OKCANCEL | MB_ICONINFORMATION);
        tiffNum++;
}


if(txtNum<numberOfTextImages) {

        dir.Format(_T("%s%s%d"), path,_T("\\
            Result"),txtNum);

        errno_t err = _tfopen_s(&OutputFile, dir
            +".txt", _T("w"));

        if(err!=0) {
                AfxMessageBox(_T("Cannot open
                    file!"), MB_OKCANCEL |
                    MB_ICONINFORMATION);
                return;
        }

        //Write all pixel values to file
```

```
                        uint8_t *pPixel = m_CnvImageInfo.
                            pImageBuffer;

                        for(uint32_t i = 0; i < m_CnvImageInfo.
                            iImageSize; i++)
                        {
                                fprintf(OutputFile, "%d ", *
                                    pPixel);
                                if((i+1)%width==0)
                                        fprintf(OutputFile,
                                    "\n");
                                pPixel++;
                        }
                        if(OutputFile) fclose(OutputFile);
                        txtNum++;
                }

                MsgStr.Format(_T("Please wait during the image
                    Aqcuistion\nAcquiring text file number %d,
                    and tiff file number %d"),txtNum, tiffNum);
                m_instructionStatic.SetWindowText(MsgStr);



                if(max(numberOfTextImages,numberOfTiffImages)==
                    max(txtNum,tiffNum)){
                        MsgStr.Format(_T("%d tiff file(s) and %d
                            txt file(s) saved in:\n%s"),
                            numberOfTiffImages,
                            numberOfTextImages,path);
                        AfxMessageBox(MsgStr, MB_OK |
                            MB_ICONINFORMATION);
                        m_busy=false;
                }
                }
        }


void CfluorDlg::OnBnClickedButton3()
{
        J_STATUS_TYPE status = J_ST_SUCCESS;

        // Stop Acquision
        if (m_hCamera) {
```

```
                status = J_Camera_ExecuteCommand(m_hCamera, "
                    AcquisitionStop");
        }

        if(m_hThread)
        {
                // Close stream
                status = J_Image_CloseStream(m_hThread);
                m_hThread = NULL;
        }

        // Close view window
        if(g_hView)
        {
                status = J_Image_CloseViewWindow(g_hView);
                if(status != J_ST_SUCCESS)
                {
                        ShowErrorMsg(CString("Unable to close
                            View Window!"),status);
                        return;
                }

                g_hView = NULL;
        }


    // Free the conversion buffer
    if (m_CnvImageInfo.pImageBuffer != NULL)
    {
        J_Image_Free(&m_CnvImageInfo);
        m_CnvImageInfo.pImageBuffer = NULL;
    }

        if(OutputFile)
        {

                fclose(OutputFile);
        }

        maks=0;

        GetDlgItem(IDC_BUTTON1)->EnableWindow(0);
}
```

```
void CfluorDlg::InitializeControls() {
        J_STATUS_TYPE retval;
    int64_t int64Val;

    // Get Gain Node
    retval = J_Camera_GetNodeByName(m_hCamera, "GainRaw", &
       m_hGainNode);
    if (retval == J_ST_SUCCESS)
    {
        int minGain = 0;
        int maxGain = 0;
        int currentGain = 0;

        // Get/Set Min
        retval = J_Node_GetMinInt64(m_hGainNode, &int64Val);
        CSCtrlGain.SetRangeMin((int)int64Val, TRUE);

        minGain = (int)int64Val;

        // Get/Set Max
        retval = J_Node_GetMaxInt64(m_hGainNode, &int64Val);
        CSCtrlGain.SetRangeMax((int)int64Val, TRUE);

        maxGain = (int)int64Val;

        // Get/Set Value
        retval = J_Node_GetValueInt64(m_hGainNode, TRUE, &
           int64Val);
        CSCtrlGain.SetPos((int)int64Val);
        SetDlgItemInt(IDC_IMAGE_GAIN_EDIT, int64Val);

        currentGain = (int)int64Val;
    }

        retval = J_Camera_GetNodeByName(m_hCamera, "
           ExposureTimeRaw", &m_hExposureNode);
    if (retval == J_ST_SUCCESS)
    {
        int minExposure = 0;
        int maxExposure = 0;
        int currentExposure = 0;

        // Get/Set Min
        retval = J_Node_GetMinInt64(m_hExposureNode, &int64Val);
```

```
        CSCtrlExposure.SetRangeMin((int)int64Val, TRUE);

                // Get/Set Max
        retval = J_Node_GetMaxInt64(m_hExposureNode, &int64Val);
        CSCtrlExposure.SetRangeMax((int)int64Val, TRUE);
        maxExposure = (int)int64Val;

        // Get/Set Value
        retval = J_Node_GetValueInt64(m_hExposureNode, TRUE, &
            int64Val);
        CSCtrlExposure.SetPos((int)int64Val);
                SetDlgItemInt(IDC_IMAGE_EXPOSURE_EDIT, int64Val)
                    ;

        currentExposure = (int)int64Val;
                CSCtrlExposure.SetTicFreq((maxExposure-
                    minExposure)/20);

    }
        SetDlgItemInt(IDC_NUM_TEXT_EDIT, numberOfTextImages);
    SetDlgItemInt(IDC_NUM_TIFF_EDIT, numberOfTiffImages);
}

void CfluorDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar*
    pScrollBar) {
        CSliderCtrl* pSCtrl;
    int iPos;
    J_STATUS_TYPE   retval;
    NODE_HANDLE hNode;
    //- Gain ---------------------------------------------------

    // Get SliderCtrl for Gain
    pSCtrl = (CSliderCtrl*)GetDlgItem(IDC_SLIDER_GAIN);
    // Is this scroll event from the Gain Slider?
    if (pSCtrl == (CSliderCtrl*)pScrollBar)
    {
        // Get current slider position
        iPos = CSCtrlGain.GetPos();

        // Set Value
        retval = J_Node_SetValueInt64(m_hGainNode, TRUE, (
            int64_t)iPos);

        // Update the Text Control with the new value
```

```
            SetDlgItemInt(IDC_IMAGE_GAIN_EDIT, iPos);
        }

        //- Exposure ------------------------------------------------------

        // Get SliderCtrl for Exposure
        pSCtrl = (CSliderCtrl*)GetDlgItem(IDC_SLIDER_EXPOSURE);
        // Is this scroll event from the Exposure Slider?
        if (pSCtrl == (CSliderCtrl*)pScrollBar)
        {
            // Get current slider position
            iPos = CSCtrlExposure.GetPos();

            // Set Value
            retval = J_Node_SetValueInt64(m_hExposureNode, TRUE, (
                int64_t)iPos);

            // Update the Text Control with the new value
            SetDlgItemInt(IDC_IMAGE_EXPOSURE_EDIT, iPos);
        }

        CDialogEx::OnHScroll(nSBCode, nPos, pScrollBar);
}


void CfluorDlg::OnBnClickedExit()
{
            CDialog::OnCancel();
}

void CfluorDlg::OnTimer(UINT_PTR nIDEvent)
{
        CString valueString;
            EnterCriticalSection(&m_CriticalSection);

        if (nIDEvent == 1)
        {
                        valueString.Format(_T("Max value of image: %d"),
                            maks);
                        m_maxvalueStatic.SetWindowText(valueString);
            }
            LeaveCriticalSection(&m_CriticalSection);
            updateInformation();
```

```
        UpdateData (TRUE);

    CDialog::OnTimer(nIDEvent);
}

void CfluorDlg::updateInformation(void)
{
        CString informationString;
        if(!m_busy) {
                EnterCriticalSection(&m_CriticalSection);
                switch(m_nLEDType){
                case 0:
                        informationString.Format(_T("\nDark
                            Current Measurement:\n\n1. Turn off
                            all light sources so that only
                            background noise is present.\n2.
                            Click 'Save Image'."));
                        m_instructionStatic.SetWindowText(
                            informationString);
                        break;
                case 1:
                        informationString.Format(_T("\
                            nCalibration of Emission Light:\n\n1
                            . Illuminate the gray standard by
                            emission light\n2. Adjust exposure
                            time, gain and power of the light
                            source so that the maximum value of
                            image is (just) below saturation.\n3
                            . Click 'Save Image'\n4. Turn off
                            light source when done."));
                        m_instructionStatic.SetWindowText(
                            informationString);
                        break;
                case 2:
                        informationString.Format(_T("\
                            nCalibration of Exitation Ligh:\n\n1
                            . Illuminate the gray standard by
                            the exitation light\n2. Keep the
                            exposure time and gain as from '
                            Calibration of emission light', but
                            adjust the power of the light source
                            so that the maximum value of image
                            is (just) below saturation.\n3.
                            Click 'Save Image'\n4. Turn off
```

```
                light␣source␣when␣done."));
            m_instructionStatic.SetWindowText(
                informationString);
            break;
        case 3:
            informationString.Format(_T("\
                nCalibration␣of␣Exitation␣Light␣with
                ␣filter:\n\nAs␣Calibration␣of␣
                Exitation␣Light,␣buth␣with␣exitation
                ␣filter␣in␣front␣of␣camera."));
            m_instructionStatic.SetWindowText(
                informationString);
            break;
        case 4:
            informationString.Format(_T("\nExitation
                ␣Diffuse␣Reflectance:\n\n1.␣Keep␣the
                ␣exitation␣filter␣in␣front␣of␣camera
                .␣\n2.␣Illuminate␣the␣sample␣by␣the␣
                exitation␣light.\n4.␣Click␣'Save␣
                Image'\n5.␣Turn␣off␣light␣source␣
                when␣done\n\n(Remember␣to␣keep␣the␣
                same␣setup␣as␣in␣the␣calibration␣
                measurements␣including␣positions,␣
                exposure,␣gain␣and␣power␣of␣the␣
                light␣source)"));
            m_instructionStatic.SetWindowText(
                informationString);
            break;
        case 5:
            informationString.Format(_T("\nEmission␣
                Diffuse␣Reflectance:\n\n1.␣
                Illuminate␣the␣sample␣by␣the␣white␣
                light␣source␣through␣glassfilters.\
                n2.␣Click␣'Save␣Image'\n3.Turn␣of␣
                light␣source␣when␣done.\n\n(Remember
                ␣to␣keep␣the␣same␣setup␣as␣in␣the␣
                calibration␣measurements␣including␣
                positions,␣exposure,␣gain␣and␣power␣
                of␣the␣light␣source)"));
            m_instructionStatic.SetWindowText(
                informationString);
            break;
        case 6:
            informationString.Format(_T("\
```

```
                    nFluorescence without filter:\n\n1.
                        Illuminate the sample by the
                        exitation light. \n2. Click 'Save
                        Image'. \n3. Turn off exitation
                        light when done."));
                    m_instructionStatic.SetWindowText(
                        informationString);
                    break;
                case 7:
                    informationString.Format(_T("\
                        nFluorescence Measurements with
                        filter:\n\n1. Increase gain/
                        exposuretime, and place the emission
                        filter in front of camera\n2.
                        Illuminate the sample by the
                        exitation light. \n3. Click 'Save
                        Image'. \n4. Turn off exitation
                        light when done."));
                    m_instructionStatic.SetWindowText(
                        informationString);
                    break;
                case 8:
                    informationString.Format(_T("\nDark
                        Current 1:\n\n1. Keep the exposure
                        time and gain as in the '
                        Fluorescence with filter'
                        measurements. \nTurn off all light
                        sources so that only blackground
                        noise is present.\n3. Click save
                        image."));
                    m_instructionStatic.SetWindowText(
                        informationString);
                    break;
                case 9:
                    informationString.Format(_T("\nExtra:\n\
                        nEventually extra measurements"));
                    m_instructionStatic.SetWindowText(
                        informationString);
                    break;
                }
                LeaveCriticalSection(&m_CriticalSection);
        }
}
```

7.4 Datasheet camera

CM-140 GE-UV
Progressive Scan

C3 Camera Suite
Unlimited
Digital
Switchability

- *Member of C3 Compact series*
- *1392 (h) x 1040 (v) 4.65 µm square pixels*
- *1/2" progressive scan - Monochrome*
- *16 frames/second with full resolution in continuous operation*
- *Increased frame rate with vertical binning and partial scan*
- *Exposure time from 31 µs to 2 sec. using Pulse Width trigger mode*
- *Programmable exposure from 61.2 µs to 32.2 ms*
- *Sequencer trigger mode for on-the-fly change of gain, exposure and ROI*
- *LVAL-synchronous/-asynchronous operation (auto-detect)*
- *Auto iris lens video output allows a wider range of light*
- *GigE Vision interface with 10 or 8-bit output*
- *Programmable GPIO with opto-isolated inputs and outputs*
- *Comprehensive software tools and SDK for Windows XP/Vista*

GiGE VISION™

Bibliography

[1] *JAI SDK - Generated by Doxygen 1.5.9*, September 2012.

[2] N. Ramanujam A. J. Durkin, S. Jaikumar and R. Richards-Kortum. Realtion between fluorescence spectra of dilute and turbid samples. 33, 1994.

[3] Robert S Bradley and Maureen S Thorniley. A review of attenuation correction techniques for tissue fluorescence. *J. R. Soc. Interface*, 3:1–13, 2006.

[4] W. L. S. Glassman G. C. Tang K. M. Yoo H. R. Zhu D. L. Akins S. S. Lubicz J. Cleary R. Prudente E. Celmer A. Caron C. H. Liu, B. B. Das and R. R. Alfano. Raman, fluorescence, and time-resolved light scattering as optical diagnostic-techniques to seperate diseased and normal biomedical media. *J. Photochem*, 16:1893–1901, 1992.

[5] Tara M. Breslin Josephine Harter Changfang Zhu, Gregory M. Palmer and Nirmala Ramanujam. Diagnosis of breast cancer using fluorescence and diffuse reflectance spectroscopy: a monte-carlo-model-based approach. 13, 2008.

[6] Steven L. Jacques PhD2 andA.J. Welch PhD1 Craig M. Gardner PhD1. Light transport in tissue: Accurate expressions for one-dimensional fluence rate and escape function based upon monte carlo simulation. *Lasers in Surgery and Medicine*, 18:129–138, 1996.

[7] Dave Shreiner Edward Angel. *INTERACTIVE COMPUTER GRAPHICS A TOP-DOWN APPROACH WITH SHADER-BASED OPENGL*. 6 edition.

[8] José Luis ARCE-DIEGO Félix FANJUL-VÉLEZ. Light propagation in turbid media: Application to biological tissues. *IEEE*, 2011.

[9] V. Backman R. Manoharan M. Fitzmaurice J. Van Dam G. Zonios, L. T. Perelman and M. S. Feld. Diffuse reflectance spectroscopy of human adenomatous polyps. *Appl. Opt.*, 38:6628–6637, 1999.

[10] C. M. Gardner. *Modeling fluorescence escape from tissue phantoms*. PhD thesis, University of Texas at Austin, 1995.

[11] Steven L. Jacques Graig M. Gardner and Ashley J. Welch. Fluorescence spectroscopy of tissue: recovery of intrinsic fluorescence from measured fluorescence. 35, 1996.

[12] D. W. Luebbers J. Hoffmann and H. M. Heise. Applicability of the kubelka munk theory for the evaluation of reflectancespectra demonstrated for haemoglobin free perfused heart tissue. *Phys. Med. Biol.*, 43:3571–3588, 1998.

[13] Michael S. Feld Jun Wu and Richard P. Rava. Analytic model for extracting intrinsic fluorescence in turbid media. 32, 1993.

[14] C. C. Compton T. J. Flotte J. M. Ritcher N. S. Nishioka K. T. Schomacker, J. K. Frisoli and T. F. Deutsch. Ultraviolet laser-induced fluorescence of colonic tissue: basic biology and diagnostic potential. *Laser Surg. Med.*, 12:6372, 1992.

[15] K. Namboori K.I. Ramachandran, G. Deepa. *Computational Chemestry and Molecular Modeling*. Springer, 2008.

[16] Hsin-i Wu L. V. Wang. *Biomedical Optics - Principles and imaging*. Wiley, 2007.

[17] Joseph R. Lakowicz. *Principles of Fluorescence Spectroscopy*. Springer, Center for Fluorescence Spectroscopy University of Maryland School of Medicine Baltimore, MD 21201 USA, 3 edition, 2006.

[18] *§ 1. Ted Young * Loling Song, *f E. J. Hennink and Hans J. Tanke*. Photobleaching kinetics of fluorescein in quantitative fluorescence microscopy. 68, 1995.

[19] Qingguo Zhang Jun Wu Markus G. Müller, Irene Georgakoudi and Michael S. Feld. Intrinsic fluorescence spectroscopy in turbid media: disentangling effects of scattering and absorption. 2001.

[20] Brian W. Pogue Mary-Ann Mycek. *Handbook of Biomedical Fluorescence*. Marcel Dekker, 2003.

[21] Jun Wu Qingguo Zhang, Markus G. Müller and Michael S. Feld. Turbidity-free f luorescence spectroscopy of biological tissue. 25, 2000.

[22] Teich Saleh. *Fundamentals of photonics*. WILEY INTERSCIENCE, 2007.

[23] BRIAN C. WILSON STEPHEN T. FLOCK, MICHAEL S. PATTERSON and DOUGLAS R. WYMAN. Monte carlo modeling of light propagation in highly scattering tissues-i: Model predictions and comparison with diffusion theory. *IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING*, 36:1162–1167, 1989.

[24] H. Xie. Drug quantification in turbid media by fluorescence imaging combined with light-absorption cerrection using white mone carlo simmulations. 16(6):066002, June 2011.

[25] N. N. Zhadin and R. R. Alfano. Correction of the internal absorption effect in fluorescence emission and excitation spectra from absorbing and highly scattering media: theory and experiment. 3, 1998.