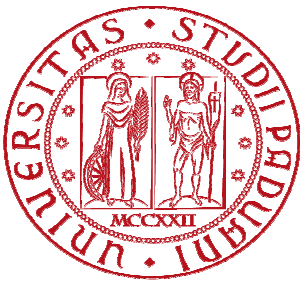

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI FISICA ED ASTRONOMIA "G. GALILEI"

TESI DI LAUREA MAGISTRALE IN FISICA

Development of a model based on differential equations of reaction-diffusion for the estimation of intracellular Ca^{2+} concentration

Laureando:

Alessandro FAGGIANI

Relatore:

Dr. Mario BORTOLOZZI

ANNO ACCADEMICO 2015/2016

Abstract

Ca^{2+} ions are vital for several functions in the human body such as muscle contraction, neuronal communication, fertilization and gene transcription. The estimation of Ca^{2+} concentration in living cell is thus a key point in biophysical research. This thesis work is conceived to derive and validate a novel mathematical formula that, based on experimental data obtained by fluorescent Ca^{2+} dyes, overcomes the general approximation that Ca^{2+} is in equilibrium with its reactants, found to fail in the presence of a rapid Ca^{2+} influx, e.g. during neuronal depolarization. Our formula, which permits to derive a more realistic estimate of the intracellular Ca^{2+} concentration from the analysis of temporal and spatial derivatives of the dye fluorescence signal, was validated by a numerical simulation in the Matlab framework based on reaction-diffusion differential equations. Experimental validation of the formula was addressed by miming the cellular environment under controlled conditions of an artificial couvette. In this system, we performed Ca^{2+} imaging experiments by the UV-flash photolysis technique in order to reproduce a nearly instantaneous and localized Ca^{2+} increase that rapidly declines due to the effect of buffering and diffusion. Our future perspective is to use the new formula to obtain a more accurate description of Ca^{2+} nanodomains that control neuronal activity.

Gli ioni Ca^{2+} sono vitali per numerose funzioni nel corpo umano come la contrazione muscolare, la comunicazione neuronale, la fecondazione e la trascrizione genica. La stima della concentrazione di Ca^{2+} nelle cellule viventi è un punto chiave per la ricerca biofisica. Questo lavoro di tesi è pensato per derivare e validare una nuova formula matematica che, basandosi su dati sperimentali ottenuti da dye fluorescenti selettivi per il Ca^{2+}, superi la comune approssimazione che assume il Ca^{2+} in equilibrio con i suoi reagenti (buffer), anche in presenza di rapidi influssi dello stesso, per esempio durante processi di depolarizzazione neuronale. La nostra formula, analizzando le derivate spaziali e temporali dei segnali di fluorescenza del dye, permette la derivazione di una più realistica concentrazione di Ca^{2+} intracellulare ed è stata validata tramite simulazioni numeriche basate su un sistema di equazioni di reazione-diffusione in Matlab. La verifica della formula dal punto di vista sperimentale è stata condotta riproducendo l'ambiente cellulare nelle condizioni controllate di una couvette. In questo sistema abbiamo effettuato esperimenti di imaging combinati a fotoliberazione UV di Ca^{2+} per riprodurre un aumento pressoché istantaneo e localizzato di Ca^{2+} che successivamente viene ridotto da effetti di buffering e diffusione. La nostra prospettiva futura è quella di utilizzare la nuova formula per ottenere una più accurata descrizione dei nano domini di Ca^{2+} che controllano l'attività neurale.

Index

Introduction

Calcium ions (Ca^{2+}) regulate multiple processes in cells, such as intracellular signal transduction mechanisms ranging from excitation-contraction coupling to synaptic transmission and genetic transcription ([1-4]). Optical measurement of the intracellular concentration of Ca^{2+} ($[Ca^{2+}]_i$) is paramount to understanding cell physiology and function. Several molecular probes, namely fluorescent dyes, capable of sensing the local ion concentration with high selectivity have been developed over the last twenty years. These are based on BAPTA(1,2-bis(o-aminophenoxy)ethane-*N*,*N*,*N*',*N*'-tetraacetic acid), a pH-insensitive evolution of the widely used Ca^{2}+-selective chelator EGTA (ethylene glycol tetraacetic acid) (Figure 0.1).

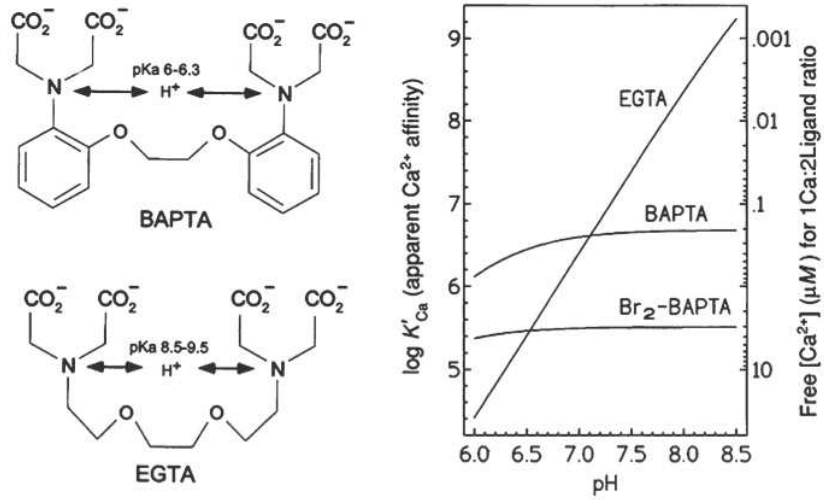

Figure 0.1: Chemical structure and pH dependence of Ca^{2+} affinity of BAPTA and EGTA

Chelation is the binding or complexation of a bi- or multidentate ligand with a single metal ion. The mechanism of Ca^{2+} chelation by BAPTA is shown in (Figure 0.2).

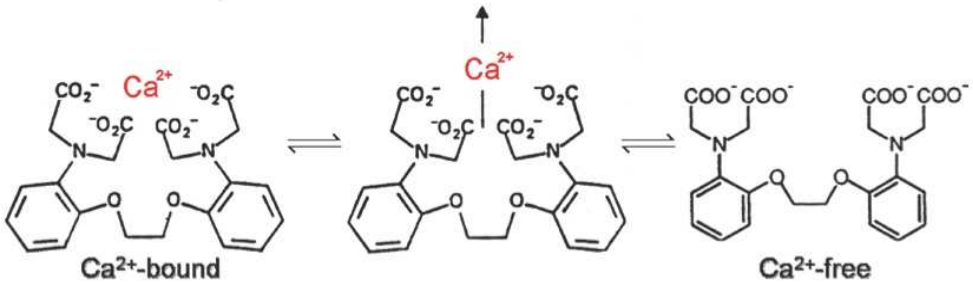

Figure 0.2: Mechanism of Ca2+ chelation by BAPTA. The presence of four carboxylic acid (usually written as –CoOH) functional groups makes possible the binding of Ca2+ ions

Chelation is the binding or complexation of a bi- or multidentate ligand. Chelating agents form multiple bonds with a single metal ion. Chelation of Ca^{2+} by a buffer B, to form a complex CaB, is described by the reaction

$$[Ca^{2+}]+[B]\underset{k_{off}^{B}}{\overset{k_{on}^{B}}{\rightleftharpoons}}[CaB] \qquad (0.1)$$

and the corresponding kinetic equation is

$$\frac{d[CaB]}{dt} = k_{on}^{B}[Ca^{2+}][B] - k_{off}^{B}[CaB] \tag{0.2}$$

where square brackets are used to indicate concentration, k_{on}^{B} is the rate constant for Ca^{2+} binding to B and k_{off}^{B} is the rate constant for Ca^{2+} dissociation. At chemical equilibrium

$$\frac{d[CaB]}{dt} = 0 \tag{0.3}$$

therefore

$$\frac{[Ca^{2+}][B]}{[CaB]} = \frac{k_{off}^{B}}{k_{on}^{B}} \equiv k_{d}^{B} \tag{0.4}$$

In the above equation, which represents an instance of the law of mass action under equilibrium conditions, k_{d}^{B} is the equilibrium or dissociation constant (for BAPTA: k_{d}^{B} = 0.192 μM [1]; k_{on}^{B} = 500 $\mu M^{-1}\ s^{-1}$ [2]; $k_{off}^{B} = k_{on}^{B} \times k_{d}^{B}$ = 96 s^{-1} [3]). Ca^{2+}-selective fluorescent probes share a modular design consisting of a metal-binding site (or sensor) A, covalently coupled to a fluorophore B, therefore

$$[A] = [B] \tag{0.5}$$

In order for such a fluorescent probe to provide useful information about its environment, it is necessary that its spectral properties be altered in a suitable manner by the parameter to be measured (a change in fluorescence yield, a shift in the excitation or emission spectrum).

The cell needs to localize the calcium signals in time and space to achieve a high bandwidth of signal transmission at specific sites: this task is entrusted to mobile calcium buffers [5]. In cells like sensory cells [6], Ca^{2+} concentration increase abruptly following opening of voltage-dependent channels [7-10]. These channels show a cooperative behaviour and cluster, creating “hotspots”, microdomains of elevated $[Ca^{2+}]_i$ [6, 11, 12] (Figure 0.3) .

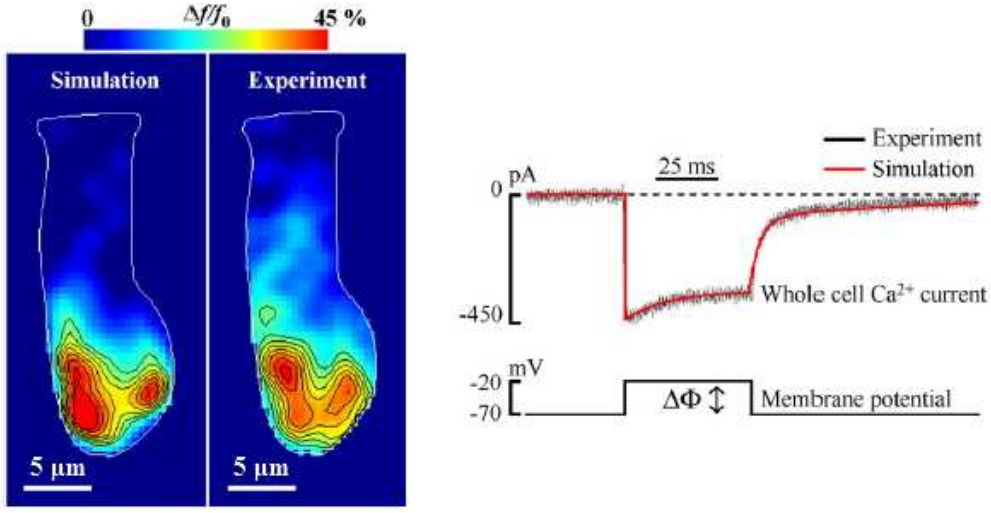

Figure 0.3: Simulated and experimental Calcium hotspots (left) in a hair cell evoked by 50 ms depolarization -20 mV (right) performed by patch-clamp.

To study the dynamics of Ca^{2+} imaging techniques are used to observe temporal and spatial dynamics of Ca^{2+} ions. Typically, these techniques combine Ca^{2+}-sensitive fluorescent dyes, patch clamp and optical microscopy together with various stimulation protocols [13-16]. These methods are limited by optical resolution in time and space because of the speed and the "punctual" localization of the Ca^{2+} influx, so they are insufficient to study Ca^{2+} dynamics [12]. Simulation and mathematical methods are necessary, both to calculate the expected time course of calcium concetration increase and their spatial extent in the presence of multiple buffers [5, 17-19] and to extrapolate the cytosolic calcium concentration from fluorescent measurements. Numerical simulations involving Ca^{2+} and its reactants, together with a realistic geometry, are CPU time consuming, whereas available formulas to directly derive Ca^{2+} from dye fluorescence suppose the equilibrium of reactants, which is not a good approximation during a fast Ca^{2+} rise.

The starting point of this thesis work was the mathematical derivation of a formula that, starting from the mass action law of Ca^{2+} with its buffers (the dye and the endogenous buffers), is able to take into account the k_{on} and k_{off} binding and unbinding constants as well as the diffusion coefficient of the dye, which we found to be as critical parameters. Typically, Ca^{2+} concentration is reconstructed supposing the equilibrium of the reactants in cells. Because of the speed of Ca^{2+} influxes, non-equilibrium conditions are fundamental. As it can be seen in Figure 0.4 [6], the results of the simulations does not fit with the experimental and simulated fluorescence, so the common reconstruction formulas does not reproduce the real Ca^{2+} concentration, in particular when it increases abruptly. As far as it is known, this is the first time that a similar formula is used to reconstruct the Ca^{2+} concentration. A set of unidimensional simulations was performed in the Matlab framework. These simulations are based on a system of reaction-diffusion differential equations that models the different reagents concentrations within a cell with cylindrical symmetry. A virtual fluorescence was also generated in order to apply our formula to the simulated hotspot region, placed in the central voxel of the cell. We found that, by the equilibrium formula, the dye fluorescence acts as a low-pass filter version of the real Ca^{2+}, while our formula is able to correctly predict the real Ca^{2+} concentration.

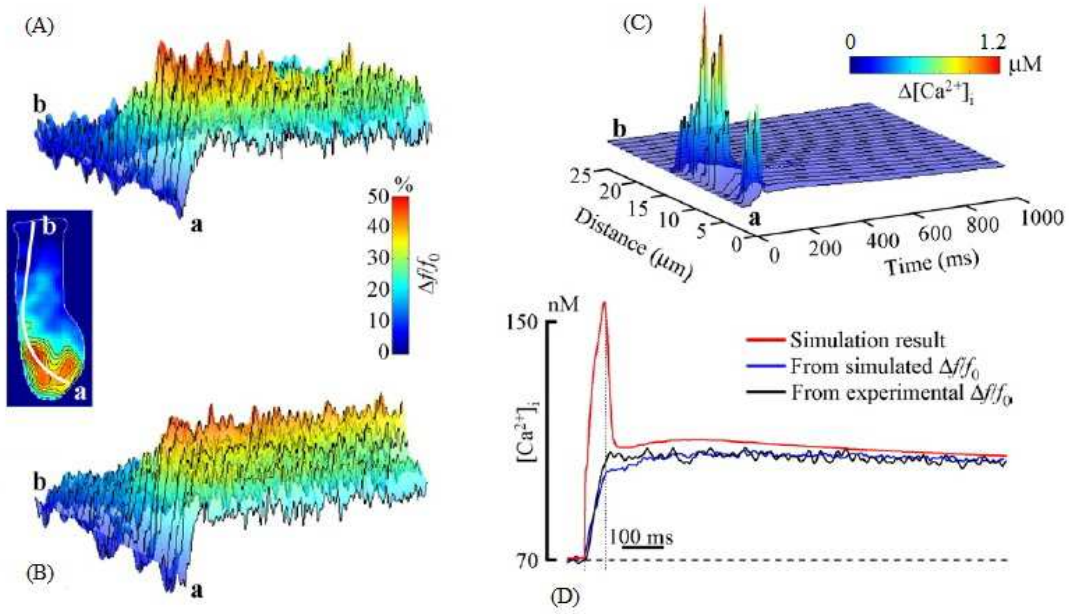

Figure 0.4: Fluorescence signals vs. cytosolic free Ca^{2+} concentration. (A) Pseudo-line-scan representation of $\Delta f / f_0$ signals obtained from the experiment; abscissae represent time and ordinates are distance along the white line (a to b) passing through a number of hotspots (inset). (B) Simulated $\Delta f / f_0$ signals for a model cell containing 1.7 mM BAPTA 29 kDa. (C) $\Delta[Ca^{2+}]_i$ changes corresponding to the simulation in (B), obtained by integrating voxel Ca^{2+}-concentrations along the optical (*z*) axis. (D) Time course of the simulated $\Delta[Ca^{2+}]_i$, integrated over the entire cell (red trace) and $\Delta[Ca^{2+}]_i$ values derived either from the simulated (blue trace) or from the experimental (black trace) $\Delta f / f_0$ whole cell signals, based on the law of mass action at equilibrium.

The second step of this work was to upgrade the simulation in three dimensions. To do this, a user friendly software was developed to build a realistic geometry based on a combination of regular geometric objects (ellipsoids, cylinders, boxes, cones). The reaction-diffusion equations were solved by dividing the diffusible space into small cubes and by using a finite difference algorithm to simulate diffusion.

In order to validate the new formula, Ca^{2+} imaging experiments have been performed. In particular, we combined patch-clamp with optical fluorescence microscopy and flash photolysis techniques. Couvette samples were used as prototypes of cells by loading them with known concentrations of fluorescent dye and the caged-Ca^{2+} compound (NP-EGTA), that is activated by a UV-laser focused on a small region (about 2 μm) of the sample, thus generating an artificial Ca^{2+} hotspot. A fast and sensitive CMOS camera (PCO-EDGE) monitors fluorescence dynamics up to 100 frames per second and the data were used to reconstruct the real Ca^{2+} concentration by our formula as well as were fitted by numerical simulations. We expect that the Ca^{2+} transient reconstructed by our formula correctly fits the numerical simulation results, at least at a better extent than the "traditional" formula that supposes the equilibrium of reactants. Improving the quantitative description of the intracellular Ca^{2+} dynamics within the cell is a fundamental requirement to describe how Ca^{2+} delivers precise information in space and time related to several physiological functions, e.g. muscle contraction, neuronal excitation or genetic expression.

1 Materials and methods

This chapter summarizes the theoretical and experimental approaches utilized in this thesis work. The formulas cited from the literature will be explained and discussed [20], then, we will go deeper in the software code and algorithms wrote to perform the simulations and, lastly, we will report and discuss all the instruments, reactants and samples that are used for the experiments.

1.1 Calcium Imaging

Several molecular probes, namely fluorescent dyes, capable of sensing the local ion concentration with high selectivity have been developed over the last 20 years. In short, when these molecules are excited by a specific wavelength, they emit light at a wavelength that depends on their bound (to Ca^{2+} ions) or unbound state (see section 1.6.3).
Suppose now that we have a system of fluorophores A at a total concentration c_T, which we excite with light of a given intensity and wavelength λ, we can summarize the excitation process as

$$A + photon \xrightarrow{k_A} A^* \tag{1.1}$$

Where k_A is the excitation rate constant (in units of s^{-1}) and A^* represents fluorophores in the excited state. The system relaxes either non-radiatively (nr), with a rate k_{nr}, or radiatively (r), emitting a photon of longer wavelength (i.e. reduced energy $h\nu'$) with a rate constant k_r (Figure 1.1).

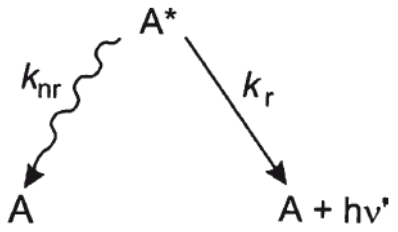

Figure 1.1: Radiative and nonradiative decay from the excited state

The overall relaxation rate constant k_M is given by

$$k_M = k_r + k_{nr} = \frac{1}{\tau_{ex}} \tag{1.2}$$

(in units of s^{-1}) where τ_{ex} is the excited state lifetime (typically a few ns). Under constant illumination, a steady state is rapidly reached such that

$$k_A \alpha \left(c_T - [A^*] \right) = k_M [A^*] \tag{1.3}$$

where the dimensionless parameter α represents the fraction of absorbed photons. Therefore, the equilibrium (steady state) concentration of excited state fluorophores $[A^*]_{eq}$ is given by

$$[A^*]_{eq} = \frac{\alpha c_T}{\alpha + k_M / k_A} \tag{1.4}$$

where, in general, $k_M \ll k_A$. The fluorescence emission intensity $f(t)$ is proportional to the number of photons emitted in the process of relaxation from the excited state

$$[A^*] \xrightarrow{k_r} A + photon \tag{1.5}$$

therefore, its steady state value is

$$f = k_r [A^*]_{eq} = \frac{\alpha k_r c_T}{\alpha + k_M / k_A} \tag{1.6}$$

where the result is expressed in mols of photons emitted per unit time and unit volume of dye solution.

For a given c_T, α is proportional to the product $\varepsilon(\lambda) \cdot l$, where $\varepsilon(\lambda)$ is the molar absorption coefficient (in units of $mol^{-1}\ m^{-1}$) and l is the length of the path traversed by the illuminating beam through the absorbing medium. The fluorescence quantum yield (sometimes termed quantum efficiency) is a gauge for measuring the efficiency of fluorescence emission relative to all of the possible pathways for relaxation and is generally expressed as

$$\eta_F = \frac{k_r}{k_M} \tag{1.7}$$

Therefore, we conclude that f depends on factors such as illumination intensity, molar concentration of fluorescent probes, fluorescence quantum yield, molar absorption coefficient, and path length. Let us then assume that the concentration of Ca^{2+}-selective fluorescent probes is kept low enough that the relationship between fluorescence emission intensity and concentration is indeed linear, as predicted by (1.4). In general, the concentration $[F]$ and $[CaF]$ of the Ca^{2+}-free (b) and Ca^{2+}-bound (b) forms differ with respect to quantum yield and absorption. Therefore, we write f as a linear combination

$$f = S_f [F] + S_b [CaF] \tag{1.8}$$

where the proportionality constants S_b and S_f lump all (system-dependent) factors such as illumination intensity, η_F, $\varepsilon(\lambda)$ and l. We are interested in measuring $[Ca^{2+}]$ in a closed system (e.g. the cell cytoplasm).

Hence, we must also include the mass balance equation

$$c_T = [F] + [CaF] \tag{1.9}$$

We then define

$$f_{\max} = S_b c_T \tag{1.10}$$

as the fluorescence emission under Ca^{2+} saturation conditions and

$$f_{\min} = S_f c_T \tag{1.11}$$

as the corresponding emission under Ca^{2+}-free conditions. Combining equations (1.11), (1.10), (1.9) and (1.8), we can write

$$\frac{f - f_{\min}}{f_{\max} - f} = \frac{[CaF]}{[F]} \tag{1.12}$$

At chemical equilibrium $\frac{[Ca^{2+}][F]}{[CaF]} = \frac{k_{off,B}}{k_{on,B}} = k_{d,B}$ (see section 1.6.3), so

$$[Ca^{2+}] = k_{d,B} \frac{[CaF]}{[F]} \tag{1.13}$$

Therefore, we conclude that

$$[Ca^{2+}] = k_{d,B} \frac{f - f_{\min}}{f_{\max} - f} \tag{1.14}$$

Equation (1.14) expresses a quantitative relationship between the physiologically relevant equilibrium $[Ca^{2+}]$, the dissociation constant $k_{d,B}$ and optically measurable quantities $f_{\min}$, $f_{\max}$ and f for single wavelength Ca^{2+}-selective probes. However, there are a number of caveats and problems with the practical use of (1.14). First, we note that the denominator vanishes as $f \to f_{\max}$. Consequently, even small fluctuations in the estimate of f (e.g. due to instrumental noise) may cause unacceptably large fluctuations in the estimation of $[Ca^{2+}]$. Furthermore, (1.14) is difficult to apply to imaging experiments where $f_{\max}$, $f_{\min}$ and f change rapidly over time due to photo-bleaching.

Provided that adverse effects are kept under control, (1.14) can be used to estimate the change in concentration $\Delta[Ca^{2+}] = [Ca^{2+}] - [Ca^{2+}]_{rest}$ where $[Ca^{2+}]$ is given by (1.14). For stimuli that keep $[Ca^{2+}]$ within the approximately linear region of (1.14), $\Delta[Ca^{2+}]$ can be expressed as

$$\Delta[Ca^{2+}] \cong \left.\frac{d[Ca^{2+}]}{df}\right|_{f_0} (f - f_0) \cong k\frac{\Delta f}{f_0} \tag{1.15}$$

Where

$$k = k_{d,B}\frac{f_0(f_{\max} - f_{\min})}{(f_{\max} - f_0)^2} \tag{1.16}$$

As discussed in the Introduction, correct estimation of $[Ca^{2+}]_i$ during a fast $[Ca^{2+}]_i$ variation requires to abandon the hypothesis that Ca^{2+} is in equilibrium with its reactants (i.e. the dye and the endogenous buffers). Non-equilibrium conditions can be correctly described by considering dye binding and unbinding kinetics, but also by considering dye diffusion away from the Ca^{2+} source (see Results).

1.2 Calcium concentration reconstruction formulas

Experimentally, Ca^{2+} concentration variations can be estimated from fluorescence measures. In particular, if we consider the fluorescence signals emitted by the single wavelength dye [15] [16] Oregon Green BAPTA-1 (OGB-1), the fluorescence signal can be expressed by (1.8) and we can defined the parameter $\alpha = \frac{S_{CaF}}{S_F} = \frac{f_{\max}}{f_{\min}}$ [21]. The concentration of the fluorescent dye is governed by the formula

$$\frac{d[F]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] \tag{1.17}$$

where $k_{on,F}$ and $k_{off,F}$ are the rate constants of the bound and unbound fluorescent dye. From equation (1.8),

$$[F] = \frac{f/S_F - \alpha[F]_{tot}}{1-\alpha} \tag{1.18}$$

where $[F]_{tot}$ is defined as $[F]_{tot} = [F] + [CaF]$. $[Ca^{2+}]$ can be expressed from (1.17),

$$[Ca^{2+}] = \frac{k_{off,F}([F]_{tot} - [F]) - \frac{d[F]}{dt}}{k_{on,F}[F]} \tag{1.19}$$

Inserting (1.17) in (1.19),

$$\frac{d[F]}{dt} = \frac{f'}{S_F(1-\alpha)} \tag{1.20}$$

Combining (1.18), (1.19) and (1.20),

$$[Ca^{2+}] = k_{d,F}\frac{S_F[F]_{tot} - f}{f - \alpha S_F[F]_{tot}} + \frac{f'}{k_{on,F}(\alpha S_F[F]_{tot} - f)} \tag{1.21}$$

The first term represent the equilibrium contribution to the concentration of Ca^{2+}, the second one the dynamic non-equilibrium contribution. $[Ca^{2+}]$ in the initial condition of equilibrium can be calculated by the formula (1.21), without the dynamic term, using f_0 instead of f

$$[Ca^{2+}]_0 = k_{d,F}\frac{S_F[F]_{tot} - f_0}{f_0 - \alpha S_F[F]_{tot}} \tag{1.22}$$

$[F]_{tot}$ can be calculated from (1.22),

$$[F]_{tot} = \frac{f_0[Ca^{2+}]_0 + k_{d,F} f_0}{S_F(\alpha[Ca^{2+}]_0 + k_{d,F})} \tag{1.23}$$

The first term, inserting (1.23) in (1.21) and dividing by f_0 the numerator and denominator, can be expressed as

$$k_{d,F}\frac{S_F[F]_{tot} - f}{f - \alpha S_F[F]_{tot}} = k_{d,F}\frac{\dfrac{[Ca^{2+}]_0 - \alpha[Ca^{2+}]_0 f/f_0 - k_d^F \Delta f/f_0}{\alpha[Ca^{2+}]_0 + k_{d,F}}}{\dfrac{\alpha[Ca^{2+}]_0 \Delta f/f_0 - \alpha k_{d,F} + k_{d,F} f/f_0}{\alpha[Ca^{2+}]_0 + k_{d,F}}} \tag{1.24}$$

Adding and subtracting $\dfrac{\alpha[Ca^{2+}]_0}{\alpha[Ca^{2+}]_0 + k_{d,F}}$ in the numerator and $\dfrac{k_{d,F}}{\alpha[Ca^{2+}]_0 + k_{d,F}}$ in the denominator of (1.24),

$$k_{d,F}\frac{S_F[F]_{tot} - f}{f - \alpha S_F[F]_{tot}} = k_{d,F}\frac{[Ca^{2+}]_0\left(\dfrac{\alpha - 1}{\alpha[Ca^{2+}]_0 + k_{d,F}}\right) + \Delta f/f_0}{k_{d,F}\left(\dfrac{\alpha - 1}{\alpha[Ca^{2+}]_0 + k_{d,F}}\right) - \Delta f/f_0} \tag{1.25}$$

In the second term of (1.24), $f' = \Delta f'$; inserting (1.23) in (1.21) and dividing by f_0 the numerator and the denominator,

$$\frac{f'}{k_{on,F}\left(\alpha S_F[F]_{tot}-f\right)}=\frac{\Delta f'/f_0}{k_{on,F}\left(\frac{\alpha k_{d,F}-\alpha_0[Ca^{2+}]_0\Delta f/f-k_{d,F}f/f_0}{\alpha[Ca^{2+}]_0+k_{d,F}}\right)} \tag{1.26}$$

Adding and subtracting $k_{d,F}$ inside the parenthesis of the denominator,

$$\frac{f'}{k_{on,F}\left(\alpha S_F[F]_{tot}-f\right)}=\frac{\Delta f'/f_0}{k_{on,F}\left[k_{d,F}\left(\frac{\alpha-1}{\alpha[Ca^{2+}]_0+k_{d,F}}\right)-\Delta f'/f_0\right]} \tag{1.27}$$

Considering the terms (1.25) and (1.27), the equation (1.21) can be expressed as

$$[Ca^{2+}]=k_{d,F}\left(\frac{[Ca^{2+}]_0\gamma+\Delta f/f_0}{k_{d,F}\gamma-\Delta f/f_0}\right)+\frac{\Delta f'/f_0}{k_{on,F}\left(k_{d,F}\gamma-\Delta f/f_0\right)} \tag{1.28}$$

where a new parameter γ is defined as

$$\gamma=\frac{\alpha-1}{k_{d,F}+\alpha[Ca^{2+}]_0} \tag{1.29}$$

The formula (1.28) has two different terms, the first one is equivalent to the equilibrium formula whereas the second one is the contribution by the derivation of the fluorescence signal. The formula (1.28) is incomplete as the contribution given by diffusion of Ca^{2+} and its reactants is not considered. In the Results section we will derive the complete formula that will take into account the diffusion term (see section 2.1.1.2).
The equilibrium term can be expressed also without the γ parameter as in [6]:

$$[Ca^{2+}]=k_{d,F}\frac{[Ca^{2+}]_0\left(\alpha-1+\alpha\cdot\Delta f/f_0\right)+k_{d,F}\cdot\Delta f/f_0}{k_{d,F}\left(\alpha-1-\Delta f/f_0\right)-\alpha\cdot[Ca^{2+}]_0\cdot\Delta f/f_0} \tag{1.30}$$

Both formulas (1.28) and (1.30) will be used in the following to reconstruct the free Ca^{2+} concentration.

1.3 One-dimensional simulation for the validation of calcium reconstruction formula

In order to validate the complete formula combining the equilibrium, derivative and diffusive terms (see section 1.2 and 2.1.1.2), one-dimensional simulations were performed using the user-friendly simulation software SimulBort developed by Prof. Mario Bortolozzi. SimulBort allows the user to solve arbitrary systems of differential equations (typically reaction-diffusion equations) in a 1-dimensional geometry, thus equivalent to a cell with cylindrical simmetry.

Furthermore, it is possible to simulate the fluorescence signals of the dye, provided it is inserted as a variable in the equations. In our simulations, the cell was simulated by a line of 25.25 μm divided in 101 voxels of 0.25 μm. The spatial resolution was chosen similar to the spatial resolution of an optical fluorescence microscope. The system is simulated for the total duration of 1 s, dividing the interval in 10000 steps of 0.0001 s.

The Ca^{2+} influx was limited to the central voxel (number 50) and was activated after 0.5 s from the beginning of the simulation. It is simulated in a similar way to the article [6]. A term named $J_{Ca^{2+}}$ expressed in μM/s is added in the first equation of (1.34). The authors of [6] use a whole cell current $I_{Ca^{2+}}$ that reproduces experimental data, defined as

$$\begin{cases} I_{Ca^{2+}} = -361\text{ pA} + (361\text{ pA} - 461\text{ pA})\, e^{\frac{0.5\text{ s} - t}{0.023\text{ s}}} & \text{for } 0.5\text{ s} < t < 0.548\text{ s} \\ I_{Ca^{2+}} = -(286\text{ pA})\, e^{\frac{0.548\text{ s} - t}{0.004\text{ s}}} + (286\text{ pA} - 373\text{ pA})\, e^{\frac{0.548s - t}{0.088s}} & \text{for } t > 0.548\text{ s} \end{cases} \tag{1.31}$$

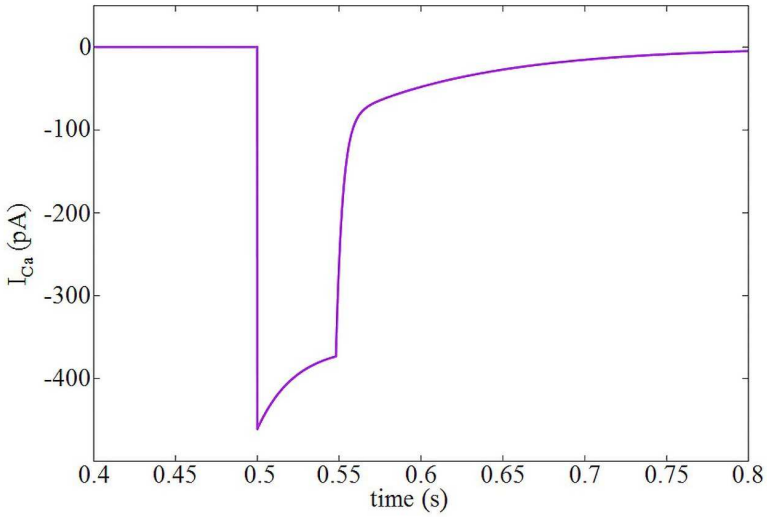

Figure 1.2: : Current $I_{Ca^{2+}}$ in the whole cell

Than, $J_{Ca^{2+}}$ (Figure 1.3) can be calculated as

$$J_{Ca^{2+}} = \beta \frac{I_{Ca^{2+}}}{2F_C} \tag{1.32}$$

where $F_C = 9.6485 \cdot 10^4$ C/mol is the Faraday's constant and β is a proportionality constant. The simulation cell model described in [6] is different from the one we used (the first is 3-dimensional, this is 1-dimensional), so β has to be changed to obtain a plausible $J_{Ca^{2+}}$, which enables Ca^{2+} concentration in the central voxel to be bigger than 1 μM and the dye not to saturate. $I_{Ca^{2+}}$ is referred to the entire cell, but Ca^{2+} comes from 10 hotspot that are considered as points in [6], so the influx for every voxel can be approximated as $\frac{I_{Ca^{2+}}}{10}$. In our simulation the influx of Ca^{2+} is simulated in the central voxel as if it was a single hotspot of [6]. To express $J_{Ca^{2+}}$ in μM/s, (1.32) have to be multiplied by a factor 10^8. β is chosen as $\beta = 0.5$.

$$J_{Ca^{2+}} = \frac{10^8 \cdot 0.5 \cdot I_{Ca^{2+}}}{2F_C} \tag{1.33}$$

where $I_{Ca^{2+}}$ is expressed in pA.

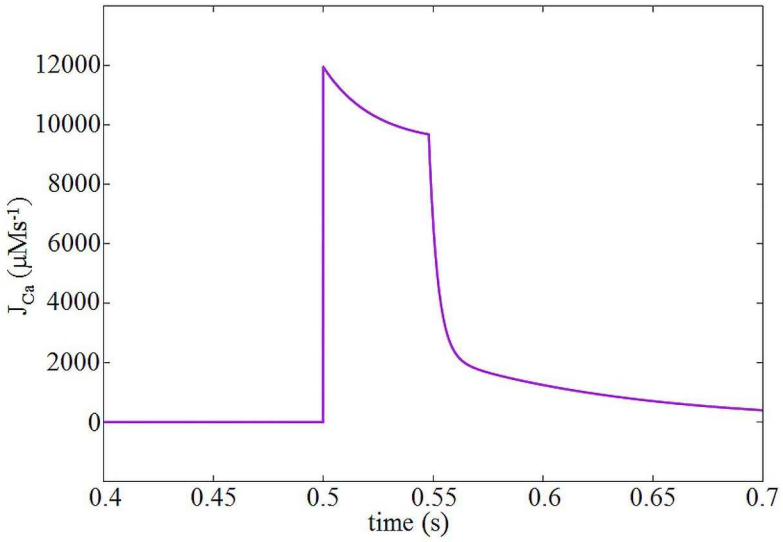

Figure 1.3: Ca^{2+} influx $J_{Ca^{2+}}$ in the central voxel

All the results of the 1-dimensional simulations will be reported in section 2.1.

1.3.1 Reaction-diffusion simulation with calbindin buffer

The system simulated by SimulBort was composed by Ca^{2+}, the Ca^{2+} dye (OGB1) and a Ca^{2+} buffer (calbindin) typical of neuronal cells [6]:

$$\begin{cases} \dfrac{d[Ca^{2+}]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] - k_{on,B}[Ca^{2+}][B] + k_{off,B}[CaB] + D_{Ca}\nabla^2[Ca^{2+}] \\ \dfrac{d[F]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] + D_F\nabla^2[F] \\ \dfrac{d[CaF]}{dt} = k_{on,F}[Ca^{2+}][F] - k_{off,F}[CaF] + D_{CaF}\nabla^2[CaF] \\ \dfrac{d[B]}{dt} = -k_{on,B}[Ca^{2+}][B] + k_{off,B}[CaB] + D_B\nabla^2[B] \\ \dfrac{d[CaB]}{dt} = k_{on,B}[Ca^{2+}][B] - k_{off,B}[CaB] + D_{CaB}\nabla^2[CaB] \end{cases} \tag{1.34}$$

where $[F]$ and $[CaF]$ are free dye and dye bound to Ca^{2+}, whereas $[B]$ and $[CaB]$ are free calbindin and calbindin bound to Ca^{2+}. Competition between F and B will be important for the evolution of the free Ca^{2+} concentration during the following Ca^{2+} influx. In the system (1.34) there are also diffusive terms described by the Laplacian of the species concentration.

In Table 1.1 we show all the values for the parameters used in the simulation and for the reconstruction of $[Ca^{2+}]$.

Table 1.1: Parameters used in the simulation

Parameter	Value
D_{Ca} [6]	440 $\mu m^2 s^{-1}$
$D_F \simeq D_{CaF}$ [6]	220 $\mu m^2 s^{-1}$
$D_B \simeq D_{CaB}$ [6]	70 $\mu m^2 s^{-1}$
$k_{on,F}$ [22]	930 $\mu M^{-1} s^{-1}$
$k_{off,F}$	192 s^{-1}
$k_{d,F} = \frac{k_{off,F}}{k_{on,F}}$ [21]	0.206 μM
$k_{on,B}$ [23]	55 $\mu M^{-1} s^{-1}$
$k_{off,B}$	11.3 s^{-1}
$k_{d,B} = \frac{k_{off,B}}{k_{on,B}}$ [23]	0.206 μM
α [21]	5

$[F]$ and $[CaF]$ can be calculated from $[Ca^{2+}]$, $k_{d,F}$ and $[F]_{tot}$ with the system of equation (1.35). $[F]_{tot}$ is the total dye (Oregon Green 488 BAPTA-1) concentration inside the cell, $[F]_{tot} = [F] + [CaF]$. (1.35) is derived from equation (1.17) in equilibrium conditions.

$$\begin{cases} [F] = \dfrac{k_{d,F}[F]_{tot}}{[Ca^{2+}] + k_{d,F}} \\ [CaF] = \dfrac{[F]_{tot}[Ca^{2+}]}{[Ca^{2+}] + k_{d,F}} \end{cases} \tag{1.35}$$

The same formula can also be used for the buffers, substituting $[F]$ with $[B]$ and $[CaF]$ with $[CaB]$.

In Table 1.2 we show the initial values of the simulation. Total concentration are chosen similar to article [6].

Table 1.2: Initial values of the simulation

Concentration	Value
$[Ca^{2+}]_0$ [24, 25]	0.100 μM
$[F]_{tot}$	50.000 μM
$[F]_0$	33.569 μM
$[CaF]_0$	16.431 μM
$[B]_{tot}$ [26]	2000 μM
$[B]_0$	1346.405 μM
$[CaB]_0$	653.595 μM

1.3.2 Reaction-diffusion simulation with dye diffusion coefficient equal to 0

In the first test we imposed $D_F = 0$ while maintaining the other parameters and initial conditions equal to those of the simulation shown in Table 1.2: in this way, the $[Ca^{2+}]$ reconstructed with the formulas (1.28) and (1.30) has to overlap with the simulated Ca^{2+} without any kind of diffusive correction.

1.3.3 Reaction-diffusion simulation with BAPTA 29kDa buffer as endogenous buffer

Another test was done simulating a system with the same endogenous buffer of [6]: it is an artificial Calcium buffer called BAPTA 29 kDa, that shows the same rate of BAPTA while being heavier, so his diffusive coefficient is smaller. In Table 1.3 we reported the new parameters of BAPTA 29 kDa used in the simulation.

Table 1.3: Parameters of BAPTA 29 kDa

Concentration	Value
$[B]_{tot}$	1700 μM
$[B]_0$	1117.808 μM
$[CaB]_0$	582.192 μM
$k_{on,B}$	500 $\mu M^{-1}s^{-1}$
$k_{off,B}$	96 s^{-1}
$k_{d,B} = \frac{k_{off,B}}{k_{on,B}}$	0.192 μM
D_B	220 $\mu m^2 s^{-1}$

1.3.4 Reaction-diffusion simulation with calretinin as endogenous buffer

Ca^{2+} buffering in cellular systems is usually simulated in the simplest way considering the Ca^{2+} buffers to have a unique binding site. Here we would like to test if our formula predicts correct results in the presence of a more realistic buffer, such as the model of Calretinin with five binding sites with different affinities for Ca^{2+} [27]. In this model, Ca^{2+} ions bound to Calretinin in five different sites: the molecule possesses two pairs of cooperative binding sites, $T_I T_{II}$ and $T_{III} T_{IV}$ that can be considered indistinguishable, so that $T_I T_{II} \simeq T_{III} T_{IV}$, and can bind to two Ca^{2+} ions, and a independent site T_V that binds to a single ion. In order to model cooperativity, two states ("relaxed" R and "tensed" T) are set for the TT binding site, each having its own set of rate constants. A binding site is in the "tensed" state (T), with a low affinity for Ca^{2+}, when no Ca^{2+} is bound to the other site in the pair, whereas a binding site is "relaxed" (R), with a high affinity for Ca^{2+}, when the other site already has a Ca^{2+} ion bound. These three binding sites can be considered as separated molecules with the same diffusion coefficient D_{Cr}. The simplified structure of Calretinin is reported in Figure 1.4.

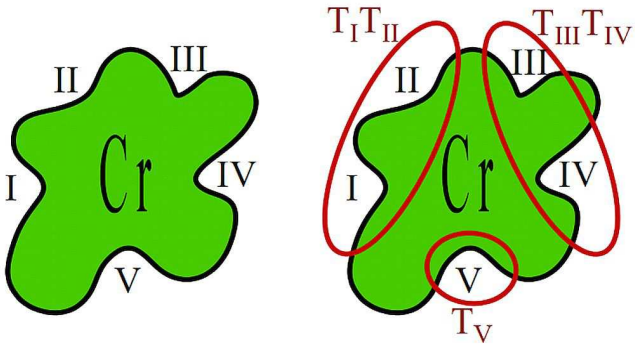

Figure 1.4: Simplified structure of a Calretinin molecule

The model assumes that binding of Ca^{2+} to one site always leads to a rapid transition $T \to R$ to the other site and that unbinding of Ca^{2+} from one site always leads to a rapid transition $R \to T$ to the other site. A standard equilibrium equation can be used to describe the independent site T_V. Calretinin binding to Calcium can be modeled using the system of reactions (1.36) [28], reported in Figure 1.5.

$$\begin{cases} [TT] + [Ca^{2+}] \underset{k_{off,T}}{\overset{2k_{on,T}}{\rightleftharpoons}} [CaTT] \\ [CaTT] + [Ca^{2+}] \underset{2k_{off,R}}{\overset{k_{on,R}}{\rightleftharpoons}} [Ca_2TT] \\ [T_V] + [Ca^{2+}] \underset{k_{off,V}}{\overset{k_{on,V}}{\rightleftharpoons}} [CaT_V] \end{cases} \tag{1.36}$$

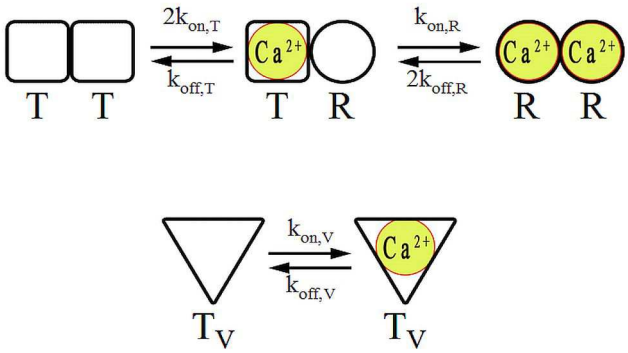

Figure 1.5: Scheme of the reactions considered in the cooperative binding model for Calretinin

Where $[TT]$ is the concentration of the double binding sites, $[T_V]$ is the concentration of the indipendent one, $[CaTT]$ and $[Ca_2TT]$ are the concentrations of the double sites, bound to one or two Ca^{2+} ions, $[CaT]$ is the concentration of the single site bound to a Ca^{2+} ion. Morover, the condition

$$[TT]_{tot} = 2[T]_{tot} = 2[Cr]_{tot} \tag{1.37}$$

Where

$$\begin{cases} [T_V]_{tot} = [CaT_V] + [T_V] \\ [TT]_{tot} = [Ca_2TT] + [CaTT] + [TT] \end{cases} \tag{1.38}$$

must be satisfied. In order to introduce properly this endogenous buffer in the simulation, new differential equations have to be added in the system (1.34), and the first one has to be modified. Now, the simulation program has to solve the system of equations (1.39).

$$
\begin{cases}
\dfrac{d[Ca^{2+}]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] - k_{on,B}[Ca^{2+}][B] + k_{off,B}[CaB] - k_{on,V}[Ca^{2+}][T_V] + \\
\qquad + k_{off,V}[CaT_V] - k_{on,R}[Ca^{2+}][CaTT] + 2k_{off,R}[Ca_2TT] + \\
\qquad - 2k_{on,T}[Ca^{2+}][TT] + k_{off,T}[CaTT] + D_{Ca}\nabla^2[Ca^{2+}] \\
\dfrac{d[F]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] + D_F\nabla^2[F] \\
\dfrac{d[CaF]}{dt} = k_{on,F}[Ca^{2+}][F] - k_{off,F}[CaF] + D_{CaF}\nabla^2[CaF] \\
\dfrac{d[B]}{dt} = -k_{on,B}[Ca^{2+}][B] + k_{off,B}[CaB] + D_B\nabla^2[B] \\
\dfrac{d[CaB]}{dt} = k_{on,B}[Ca^{2+}][B] - k_{off,B}[CaB] + D_{CaB}\nabla^2[CaB] \\
\dfrac{d[T_V]}{dt} = -k_{on,V}[Ca^{2+}][T_V] + k_{off,V}[CaT_V] + D_{Cr}\nabla^2[T_V] \\
\dfrac{d[TT]}{dt} = k_{off,T}[CaTT] - 2k_{on,T}[Ca^{2+}][TT] + D_{Cr}\nabla^2[TT] \\
\dfrac{d[CaT_V]}{dt} = k_{on,V}[Ca^{2+}][T_V] - k_{off,V}[CaT_V] + D_{CrCa}\nabla^2[CaT_V] \\
\dfrac{d[CaTT]}{dt} = 2k_{on,T}[Ca^{2+}][TT] - k_{off,T}[CaTT] - k_{on,R}[Ca^{2+}][CaTT] + +2k_{off,R}[Ca_2TT] + D_{CrCa}\nabla^2[CaTT] \\
\dfrac{d[Ca_2TT]}{dt} = k_{on,R}[CaTT][Ca^{2+}] - 2k_{off,R}[Ca_2TT] + D_{CrCa_2}\nabla^2[Ca_2TT]
\end{cases}
\tag{1.39}
$$

Within this system, there are many new parameters and new initial conditions that have to be take into consideration. In Table 1.4 we reported the new parameters of the simulation.

Table 1.4: New parameters of the simulation

Parameter	Value
$D_{Cr} \simeq D_{CrCa} \simeq D_{CrCa_2}$	70 $\mu m^2 s^{-1}$
$k_{on,R}$	310 $\mu M^{-1} s^{-1}$
$k_{off,R}$	20 s^{-1}
$k_{d,R} = \dfrac{k_{off,R}}{k_{on,R}}$	0.068 μM
$k_{on,T}$	1.8 $\mu M^{-1} s^{-1}$
$k_{off,T}$	53 s^{-1}
$k_{d,T} = \dfrac{k_{off,T}}{k_{on,T}}$	28 μM
$k_{on,V}$	7.3 $\mu M^{-1} s^{-1}$
$k_{off,V}$	252 s^{-1}
$k_{d,V} = \dfrac{k_{off,V}}{k_{on,V}}$	36 μM

The diffusive coefficient is chosen as article [26]. The cytosolic Calretinin concentration $[Cr]_{tot}$ is assumed to be 1.2 mM [26]. To find the initial concentrations of the new chemical species introduced to simulated the Calretinin, the system (1.40) has to be solved.

$$\begin{cases} k_{d,T}[CaTT] = 2[Ca^{2+}][TT] \\ k_{d,V}[CaT_V] = [Ca^{2+}][T_V] \\ k_{d,R}[Ca_2TT] = [CaTT][Ca^{2+}] \\ 2[Cr]_{tot} = [CaTT] + [TT] + [Ca_2TT] \\ [Cr]_{tot} = [T_V] + [CaT_V] \end{cases} \tag{1.40}$$

where the firsts three equation derives from the seventh, the eighth and the tenth equation of (1.39) for an equilibrium state, the latst two derives from (1.38). The solutions of this system are

$$\begin{cases} [T_V]_0 = \dfrac{k_{d,V} \cdot [T_V]_{tot}}{[Ca^{2+}]_0 + k_{d,V}} \\ [TT]_0 = \dfrac{2 \cdot k_{d,R} \cdot k_{d,T} \cdot [T_V]_{tot}}{2\left([Ca^{2+}]_0\right)^2 + 2k_{d,R}[Ca^{2+}]_0 + k_{d,R}k_{d,T}} \\ [Ca_2TT]_0 = \dfrac{4\left([Ca^{2+}]_0\right)^2 [T_V]_{tot}}{2\left([Ca^{2+}]_0\right)^2 + 2k_{d,R}[Ca^{2+}]_0 + k_{d,R}k_{d,T}} \\ [CaTT]_0 = \dfrac{4k_{d,R}[Ca^{2+}]_0[T_V]_{tot}}{2\left([Ca^{2+}]_0\right)^2 + 2k_{d,R}[Ca^{2+}]_0 + k_{d,R}k_{d,T}} \\ [CaT_V]_0 = \dfrac{[Ca^{2+}]_0[T_V]_{tot}}{[Ca^{2+}]_0 + k_{d,V}} \end{cases} \tag{1.41}$$

A similar system is calculated in the supplementary materials of [26]. In Table 1.5 we reported the new initial concentrations of the simulation.

Table 1.5: Initial concentrations of the simulation with Calretinin

Concentration	Value
$[Ca^{2+}]_0$ [24, 25]	0.100 μM
$[Cr]_{tot}$	1200 μM
$[T_V]_{tot}$	1200 μM
$[TT]_{tot}$	600 μM
$[T_V]_0$	1196.676 μM
$[TT]_0$	2358.382 μM
$[Ca_2TT]_0$	24.773 μM
$[CaTT]_0$	16.846 μM
$[CaT_V]_0$	3.324 μM

1.3.5 Reaction-diffusion simulation with different fluorescent dyes

As several different dyes are used in biological experiments to study the Ca^{2+} concentration, we performed simulations with diverse parameters for the Ca^{2+} dye. Table 1.6 and Table 1.7 reports the parameters and the initial conditions used in four simulations using different fluorescent dyes: Oregon Green 488 BAPTA-1, Oregon Green 488 BAPTA-5N, Fluo-3 and Fluo-4. D_F , calculated using the formula in supplementary material of [6] for all the dyes, is considered the same for the four molecules even if they have slightly different molecular weights. Diffusive coefficients differ at most for 3% from $D_F = 220\,\mu m^2 s^{-1}$. Initial concentrations are calculated with (1.35).

Table 1.6: Rate constant of the four fluorescent dyes

Parameter	Value	Parameter	Value
$k_{on,\text{OGB-1}}$ [22]	930 $\mu M^{-1}s^{-1}$	$k_{on,\text{Fluo-3}}$ [29]	13.1 $\mu M^{-1}s^{-1}$
$k_{off,\text{OGB-1}}$	192 s^{-1}	$k_{off,\text{Fluo-3}}$ [29]	33.67 s^{-1}
$k_{d,\text{OGB-1}} = \frac{k_{off,\text{OGB-1}}}{k_{on,\text{OGB-1}}}$ [21]	0.206 μM	$k_{d,\text{Fluo-3}} = \frac{k_{off,\text{Fluo-3}}}{k_{on,\text{Fluo-3}}}$ [30]	2.57 μM
$\alpha_{\text{OGB-1}}$ [21]	5	$\alpha_{\text{Fluo-3}}$ [29]	200
$k_{on,\text{OGB-5N}}$ [23]	124 $\mu M^{-1}s^{-1}$	$k_{on,\text{Fluo-4}}$	1044 $\mu M^{-1}s^{-1}$
$k_{off,\text{OGB-5N}}$	5600 s^{-1}	$k_{off,\text{Fluo-4}}$ [31]	350 s^{-1}
$k_{d,\text{OGB-5N}} = \frac{k_{off,\text{OGB-5N}}}{k_{on,\text{OGB-5N}}}$ [23]	45.16 μM	$k_{d,\text{Fluo-4}} = \frac{k_{off,\text{Fluo-4}}}{k_{on,\text{Fluo-4}}}$ [30]	0.335 μM
$\alpha_{\text{OGB-5N}}$ [23]	30.8	$\alpha_{\text{Fluo-4}} \approx \alpha_{\text{Fluo-3}}$	200

Table 1.7: Initial concentration of the four fluorescent dyes

Concentration	Value
$[Ca^{2+}]_0$ [24, 25]	0.100 μM
$[F]_{tot}$	50 μM
$[\text{OGB-1}]_0$	33.660 μM
$[Ca\text{OBG-1}]_0$	16.340 μM
$[\text{OGB-5N}]_0$	49.890 μM
$[Ca\text{OGB-5N}]_0$	0.110 μM
$[\text{Fluo-3}]_0$	48.127 μM
$[Ca\text{Fluo-3}]_0$	1.873 μM
$[\text{Fluo-4}]_0$	38.506 μM
$[Ca\text{Fluo-4}]_0$	11.494 μM

1.3.6 Reaction-diffusion simulation with OGB-1 or OGB-1 dextran in combination with NP-EGTA as endogenous buffer

In order to validate our formula in respect to the diffusion term, we performed simulations changing the diffusion coefficient of the dye, with values correspondent to OGB-1 and OGB-1 dextran. The endogenous buffer was equivalent to 2 mM unphotolyzed NP-EGTA. The OGB-1 dextran posseses the same characteristics of the OGB-1, except from the weight. It weights about 11000 MW in spite of 1000 MW, so its diffusive coefficient is smaller [32]. For this simulation Calcium influx was changed and lasts 5 ms in the central voxel. This simulation was supposed to reproduce in a very simplified way a photorelease of Calcium. NP-EGTA [33] can be considered always unphotolyzed, due to the small area of the Calcium release, with respect to the entire cell, so the photoproducts of NP-EGTA can be neglected. We report in Table 1.8 and Table 1.9 the parameters and the initial concentrations.

Table 1.8: Parameters of the simulation with OGB-1 and NP-EGTA

Parameter	Value
$k_{on,F}$	930 $\mu M^{-1}s^{-1}$
$k_{off,F}$	192 s^{-1}
$k_{d,F}$	0.206 μM
$k_{on,NP}$	17 $\mu M^{-1}s^{-1}$
$k_{off,NP}$	1.36 s^{-1}
$k_{d,NP}$	0.08 μM
D_F	220 $\mu m^2 s^{-1}$
$D_{F,dex}$ [32]	16 $\mu m^2 s^{-1}$
D_{NP} [32]	100 $\mu m^2 s^{-1}$

Table 1.9: Parameters of the simulation with OGB-1 and NP-EGTA

Parameter	Value
$[Ca^{2+}]_0$	0.05 μM
$[F]_0$	200 μM
$[CaF]_0$	50 μM
$[NP]_0$	1155 μM
$[CaNP]_0$	845 μM

1.4 Three-dimensional simulation software

In order to validate our formula in a 3-D space, we upgraded SimulBort from 1- to 3-dimensional geometry. In this chapter we will describe the simulation software upgrade from “SimulBort” to the new “SimulCell”. Firstly, we developed a software capable of creating volumes with regular shapes. Through this, users can create a simple volume that reproduce the shape of a 3-dimensional cell. Then, we simulated and validated a diffusion process within several kind of simple volumes (in particular, inside a sphere). Finally, we also upgraded the other modules of SimulBort, in order to make them capable of working with a 3D geometry. The software was written in Matlab, using a GUI interface. The fundamental idea was that everyone, not only the programmer himself, could easily use it.

1.4.1 SimulCell

The main interface of “SimulBort” was upgraded as it is shown in Figure 1.6.

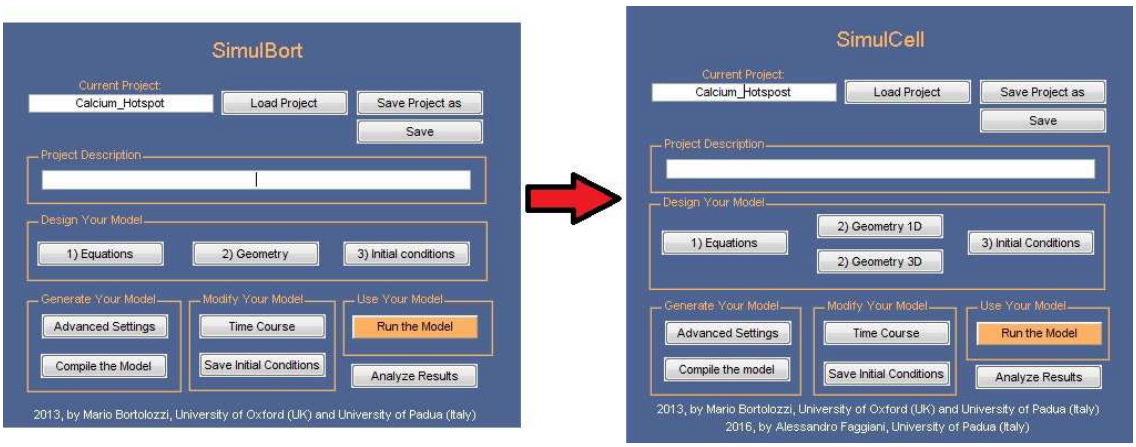

Figure 1.6: Main interface of SimulBort (left) and SimulCell (right)

The button “1) equation” opens the window shown in Figure 1.7 that remains the same in SimulCell. Inide this window, is possible to define a system of differential equations with their variables, parameters and inputs. Inputs are function or sentences that change the value of one or more parameters as a function of the time of the simulation. When the user pushes the compile button, the equations are recognized by a parser and then re-written in a solver-compatible way.

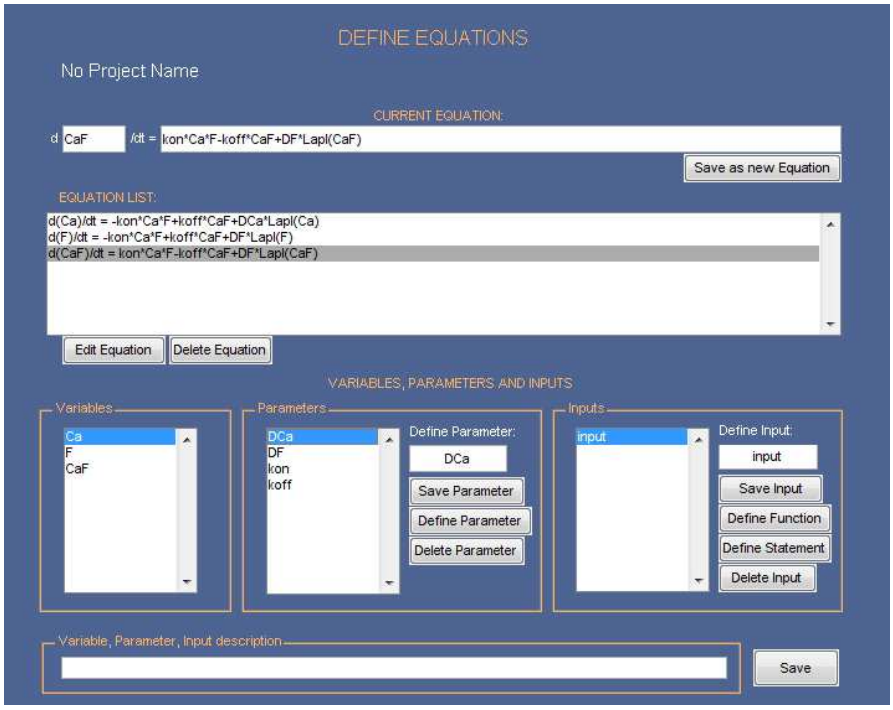

Figure 1.7: Equation section of SimulBort and SimulCell

1.4.2 Geometry

As we already said in the previous chapters, SimulBort is a software for unidimensional simulations. The geometry section permits to create a line divided in voxels at a known distance or other specific patterns, where the variables and the parameters of their equation can be user-defined. (Figure 1.8)

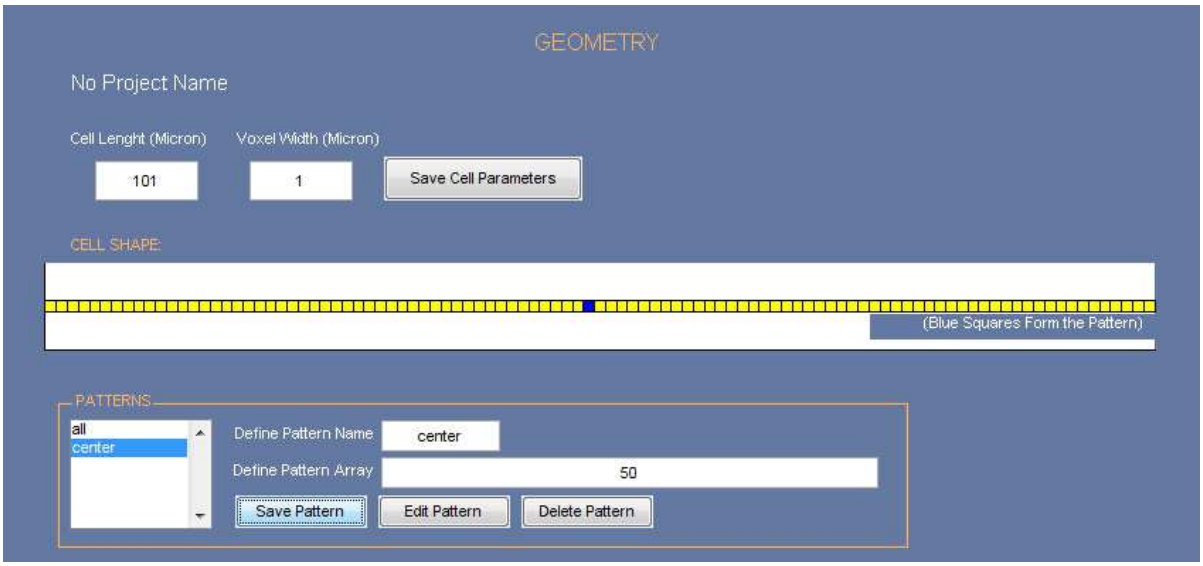

Figure 1.8: Geometry section of SimulBort

We made a big effort in order to upgrade this section to a 3-dimensional enabled module. The upgraded geometry section is shown in Figure 1.9.

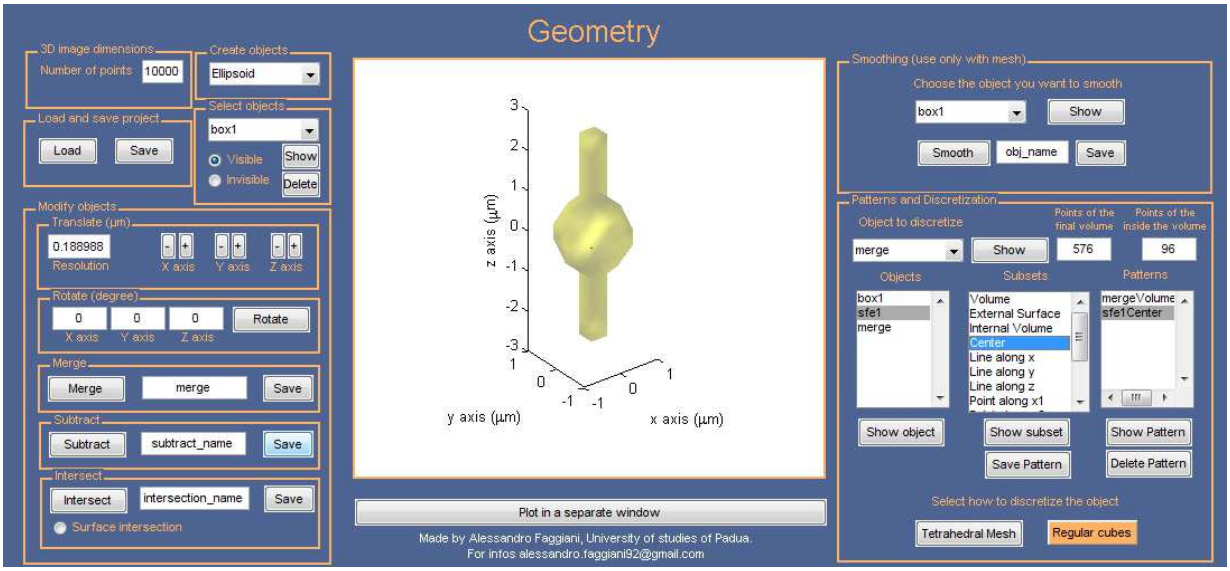

Figure 1.9: Geometry section of SimulCell

When this window is open, a variable named 'dataF' is created. This is a structure that contains every parameter of the geometry. The 3D-objects are 3-dimensional arrays of ones and zeros commonly known as volumetric images. The ones values are associated with the solid part of the object, the zeros are the external empty parts. The images are created in the white central section using the "patch" and "isosurface" function of Matlab. In Appendix 5.1.1 we reported how an object is created from the volumetric image named `dataF.Objects.box1.V`. Togheter with the previous variable, a structure named 'temp', that contains temporal files used for some of the functions, is also created.
We will breafly describe each feature of the software in the following section.

Create Objects Panel

Opening the popup list, the user can choose between 4 simple geometric shapes. After one shape is selected, a new window appears, in which he can choose the size of the object, display it or save it with a name. First, a zeros 3D-matrix with sides 3/2 times bigger that the sizes previously chosen is created. If the user modifies the input size of the object, he modifies the 3D-matrix too. There is also a section named "number of points"; it enables the user to choose the resolution of his object. This number is associated with the number of points `dataF.N_tot` in which the 3D-matrix will be divided. This operation is computed as reported in Appendix 5.1.2. Every shape is created by a for-loop that assigns the value 1 to the points that respect a specific condition. Every new object is centered in (0, 0, 0). The shapes, the parameters and the conditions are shown in Table 1.10.
If there are already other existing objects, they are reshaped and recreated in a bigger 3D-matrix of zeros, maintaining the best resolution (number of points) possible; therefore, every object resides inside the same 3D-matrix. This is necessary for the correct functioning of the software.

Table 1.10: Shapes, parameters and conditions used in the “Create Objects” section

Shape	Parameter	Condition
Box	x dimension (box.x) y dimension (box.y) z dimension (box.z)	dataF.X <= x/2 & dataF.Y <= y/2 & dataF.Z <= z/2
Ellipsoid	radius x (ellipsoid.r1) radius y (ellipsoid..r2) radius z (ellipsoid..r3)	(dataF.X/ellipsoid.r1)^2 + (dataF.Y/ellipsoid.r2)^2 + (dataF.Z/ellipsoid.r3)^2 <= 1
Cylinder	radius x (cylinder.r1) radius y (cylinder.r2) height (cylinder.h)	(dataF.X/cylinder.r1)^2 + (dataF.Y/cylinder.r2)^2 <= 1 & dataF.Z >= -cylinder.h/2 & dataF.Z <= cylinder.h/2
Cone	radius x (cone.r1) radius y (cone.r2) height (cone.h)	(dataF.X/cone.r1)^2 + (dataF.Y/cone.r2)^2 - ((dataF.Z -cone.h/2)/cone.h)^2 <= 0 & dataF.Z >= -cone.h/2 & dataF.Z <= cone.h/2

Number of points Panel

This command allows to modify the number of points in which all the objects are discretized. This changes the number of dataF.N_tot. The same procedure shown in section “Create objects” is repeted to create a new grid of values dataF.X, dataF.Y, dataF.Z. If there are objects already saved, this command go through all of them doing a resize. This operation is possible because every complex object can be decomposed into its primary components created in “Create objects” and then reassambled in a smaller or bigger 3D-matrix.

Select objects Panel

In this section the user can choose which objects he wants to visualize. The object selected in the popup menu is shown in blue, the other visible objects are yellow. Visible and invisible objects are also copied inside two structure named vis and invis.

Translate Panel

This feature allows the objects to be moved. The “resolution” section shows the smallest possible translation for a given object, which can be changed in order to make objects move faster. Only the visible objects (the blue and the yellow ones) can be translated. In Appendix 5.1.3 we reported the two functions used to translate an object in the positive (translation_xp) (Figure 1.10) or negative direction (translation_xm) of the x-axis. The other functions for y-axis and z-axis are similar.

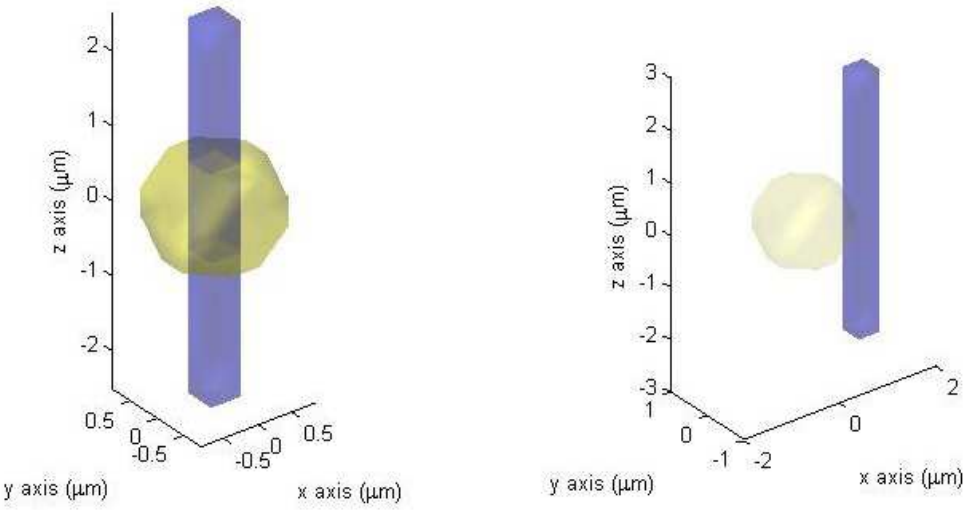

Figure 1.10: Translation of the blue parallelepiped on positive direction on x axis

Everytime an object is translated, a variable named dataF.Objects.example.translation is saved. This variable is a vector of 3 values that remembers to the user where the object is translated from the center (0, 0, 0).

Rotate Panel

This feature allows to rotate the visible objects around the x, y and z-axis. This function can only work with 3D-cube matrix with an even number of discretizations in every side, so that every object is recreated in a 3D-cube of zeros with the smallest surrounding size possible. The objects can be rotated applying a combination of five affine transformation: a translation of the object to the center of the box of zeros, a combination of the three rotation around the three axes and the inverse of the first translation (Figure 1.11). See Appendix 5.1.4 for the detailed Matlab functions.

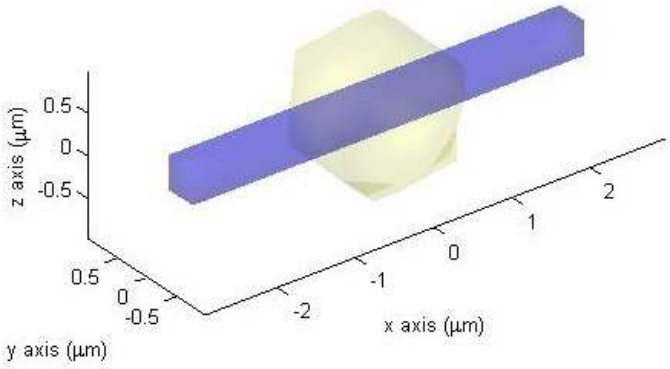

Figure 1.11: Rotation of the blue parallelepiped shown in Figure 1.10

Merge Panel

This feature allows to merge objects and to save the results as a new object (Figure 1.12). By pressing the button, all the visible objects become one. This operation is done inside a for-cycle for all the visible objects, summing all their volumetric images point by point . All the points of the sum-object that are bigger than 1 are set to 1.

Subtract Panel

This feature allows to subtract the visible objects to a selected one (Figure 1.12). This operation is done inside a for-cycle for all the visible objects, summing all their volumetric images point by point except for the selected one. All the points of the sum-object that are bigger than 1 are set to 1. Then, this volumetric image is subtracted point by point from the selected object. The values smaller than 0 in the final object are set to 0.

Intersect Panel

This feature allows to find the volume intersection between the visible objects and a selected one (Figure 1.12). This operation is done inside a for-cycle for all the visible objects, summing all their volumetric images point by point except for the selected one. All the points of the sum-object that are bigger than 1 are set to 1. Then, this volumetric image is added point by point to the selected object. The 0 and 1 values of this final object are set to 0, the values bigger than 1 are set to 1. There is also the possibility to save the surface intersection. This is computed using the existing `imedge3d` function.

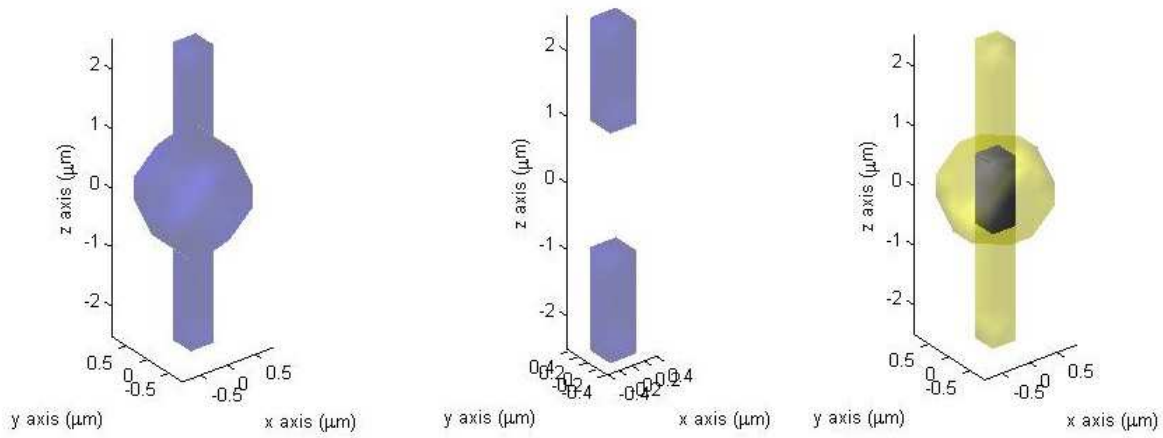

Figure 1.12: Merge, subtraction and intersection operation on the object of Figure 1.10

Smoothing Panel

This section allows the user to smooth the surface of his objects. It uses the Matlab's function `smooth3`. It can not be used if the user wants to discretize the final object pushing the button "regular cubes". It works only with tetrahedral mesh.

Patterns and discretization Panel

This is the last step of the creation of the user geometry. Here, the user can select the final object and define a specific pattern inside it. This pattern will have the same function of the SimulBort's patterns. They can be created starting from any finalized object and selecting one subsets inside (e.g. the entire volume, the external surface, its center, a line that pass through the center…). After doing this, the user can choose how to discretize his object: with a tetrahedral mesh (feature under development) or with regular cubes. In the latter case, the 3D-matrix defining the object is resized in order to have the minimum number of points outside the object (to make the simulation faster). Furthermore, pushing the button "regular cubes" creates a new structure called `dataM`. The final volumetric image is called `dataM.geometry.V`. This is the connection between the geometry section and the SimulCell code. Here there are the variables that was used by SimulBort for the simulation. In this structure there are the coordinates of the points of the simulation `dataM.voxels.centers`, the matrix of the neighbours of every voxel `dataM.voxels.neighbours` (every line of the matrix contain the six indexes of the neighbours of the voxel that have the number of the line as index. If the neighbour is outside the object, the value is 0), the volumes of every voxel `dataM.voxels.volumes`, the name and the indexes of every pattern `dataM.patterns.name` and `dataM.patterns.array`, and here also are computed the neighbouring and the intersecting patterns for every one of them: `dataM.geometry.pattern.neighbouring_names` and `dataM.geometry.pattern.intersection_names`. One pattern is considered intersecting with another if the sum of their volumetric images is bigger than one; it is considered neighbouring if his volumetric image summed with the other is not bigger than one, but with a edge bigger than one discretization per direction has intersections. The main part of the code is reported in Appendix 5.1.5.

The object volume can be discretized with a tetrahedral mesh or with a regular grid. The main advantage of the former method is that tetrahedra can adapt to every shape and can be thickened in particular zones while preserving the four neighbours structure (except for the tetrahedra on the borders). Nevertheless, this discretization is difficult to treat due to the simple elements algorithm necessary to optimally simulate a diffusion process and its not so easy integration with the algorithms already existing inside SimulBort. A regular grid can't be local thickened and can't adapt, but is easier to use for simulating with a finite difference algorithm and to integrate with the pre-existing code.
The tetrahedral mesh can be done using Jigsaw [34] and Iso2Mesh [35], two free Matlab's plugins. Figure 1.13 shows an example of a mesh done inside a sphere.

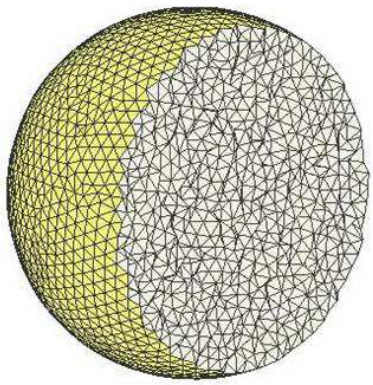

Figure 1.13: Volume of a sphere discretized with a tetrahedral mesh

Regular grid is simply made using Matlab: the volume discretized in this way is divided in boxes of the same dimensions (Figure 1.14).

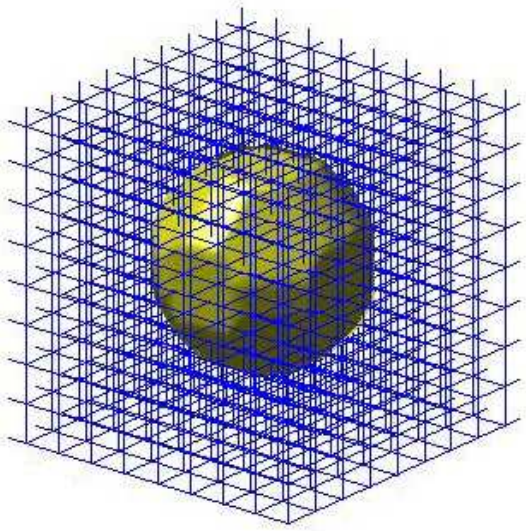

Figure 1.14: Example of the discretization using regular cubes

Diffusion simulation has been made both in tetrahedral mesh and regular grid using different algorithms. The results are always compared with theoretical models. For tetrahedral mesh, we tested different algorithms to avoid the finite elements methods but no one worked correctly due to the irregularity of the tetrahedral. Regular grid algorithms are simpler and more effective than simulation made with tetrahedral mesh, thus were chosen as the go-to approch.

1.4.3 Initial Conditions

In this section, both in SimulBort and SimulCell, users can define the variables, parameters and inputs to apply to the geometric pattern that they have previously created. In Figure 1.15 we show how this section appears for 1-dimensional simulations.

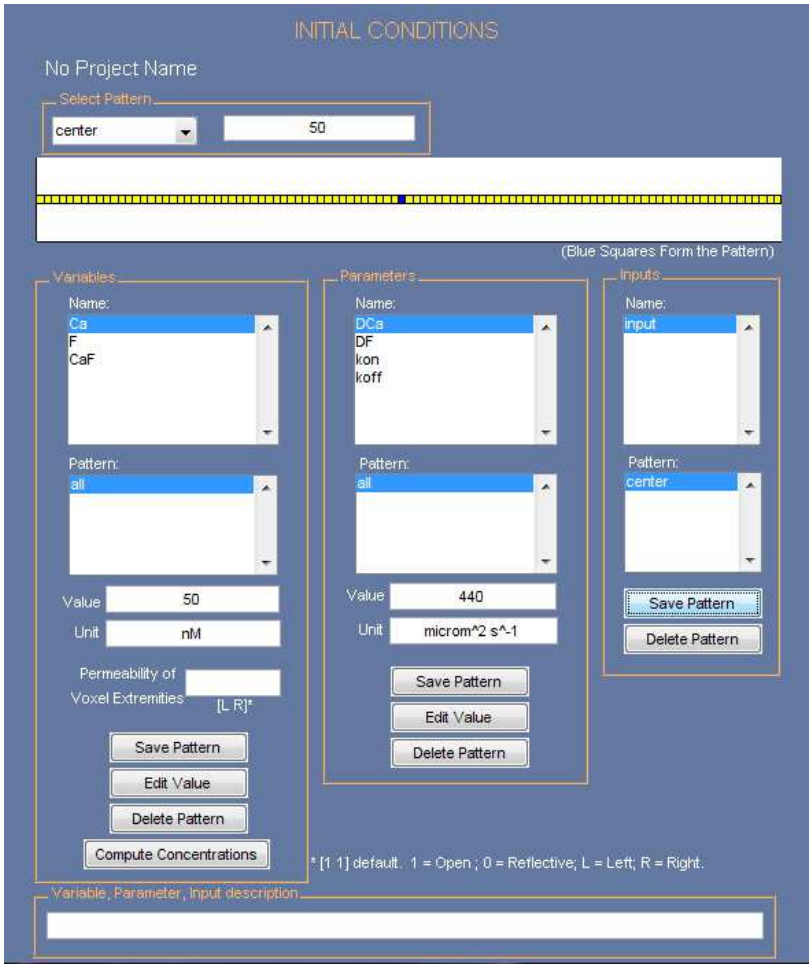

Figure 1.15: Initial conditions section for 1-dimensional simulations

This section had to be upgraded in order to work with 3-dimensional geometry. In Figure 1.16 we reported the graphical interface of the 3-dimensional initial conditions section. Except from the permeabilities panel, it resembles closely the previous version. In the new panel, once the user select a pattern from the list, the list of the adjoining patterns (already calculated in the Geometry section) appears. When the user defines the value of a variable, SimulCell creates a structure named `dataM.variabls.initial_conditions{1}`. The number `{1}` is associated to the variable: it is the position of that variable in the variable list. This structure contains the following fields: `pattern_name`, `pattern_array`, `pattern_value`, `pattern_unit`, `pattern_permeability` and `pattern_indexes`. The first four fields are present also inside `dataM.parameters.initial_conditions{1}` and the first two inside `dataM.inputs.initial_conditions{1}`. `pattern_name` is a string containing the name of the pattern for which the value is defined, `pattern_array` is a cell array with the indexes of the chosen pattern, `pattern_value` is the value of the variable in that pattern and `pattern_unit` are the units of that variable. The new field `pattern_permeability` is a N×M matrix containing the permeability values (from 0 to 1, 0 if the patterns is impermeable, 1 if it is permeable) where N is the number of voxels of that pattern and M the number of neighbours (in 3-D, M = 6); e.g. in the corresponding column of row number 7 we can find the permeability of voxel 7 towards neighbours voxels in the three spatial dimensions. `pattern_indexes` is an array N×M containing the indexes of the neighbours of every voxel of the pattern. By default, every pattern is completely permeable with itself and with the other adjoining patterns and impermeable with the external. These conditions can be modified in the

designed space in the permeabilities panel. The code used to modify `pattern_permeability` and `pattern_indexes` is reported in Appendix 5.1.6.

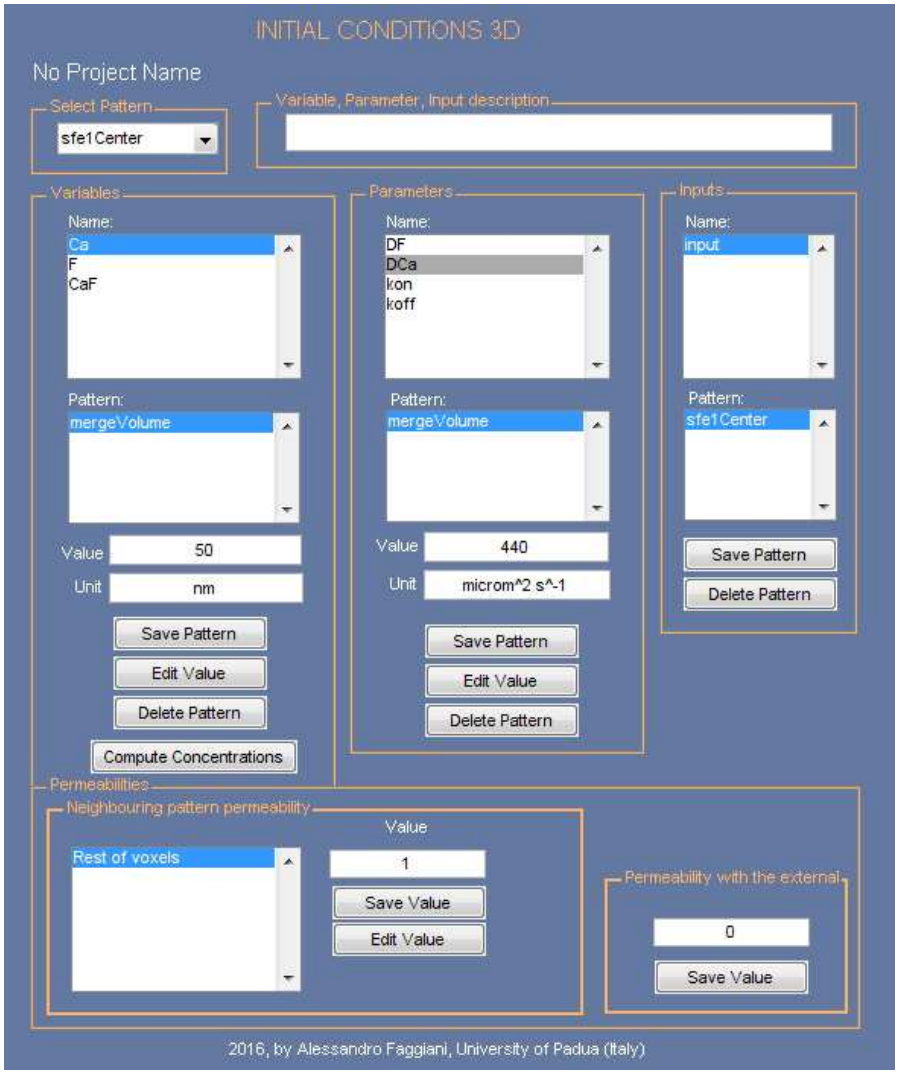

Figure 1.16: Initial conditions section for 3-dimensional simulations

1.4.4 Generate, use and analyze the model

In these section, the model is compiled and solved by differential equation solvers. In the section "Compile" the equations are re-written in order to be inserted inside the solver. Some small changes had to be made to the code in order to upgrade it to 3-dimensions but the core code remains the same as in SimulBort. The most important change is in the function `formatOdeMEX.m` where the laplacian operator is discretized (Appendix 5.1.7).

Another important difference from the 1-dimensional software is the "Analyze Results" section (Figure 1.17). Here, it is possible to visualize the results of the simulation plotting, every variable as a function of time and, also, if some variables are associated to fluorescent dyes concentrations, the fluorescence ratio f / f_0. This ratio is calculated using the formula (1.8) in the following way:

$$\frac{f}{f_0} = \frac{S_f[F] + S_b[CaF]}{S_f[F]_0 + S_b[CaF]_0} = \frac{[F] + \alpha[CaF]}{[F]_0 + \alpha[CaF]_0} \tag{1.42}$$

In the panel "Select a Pattern" is possible to choose a pattern and a subset of it in order to visualize the results just inside that specific area (i.e. Figure 1.17 shows a selected volume of the pattern "sfe1Volume").

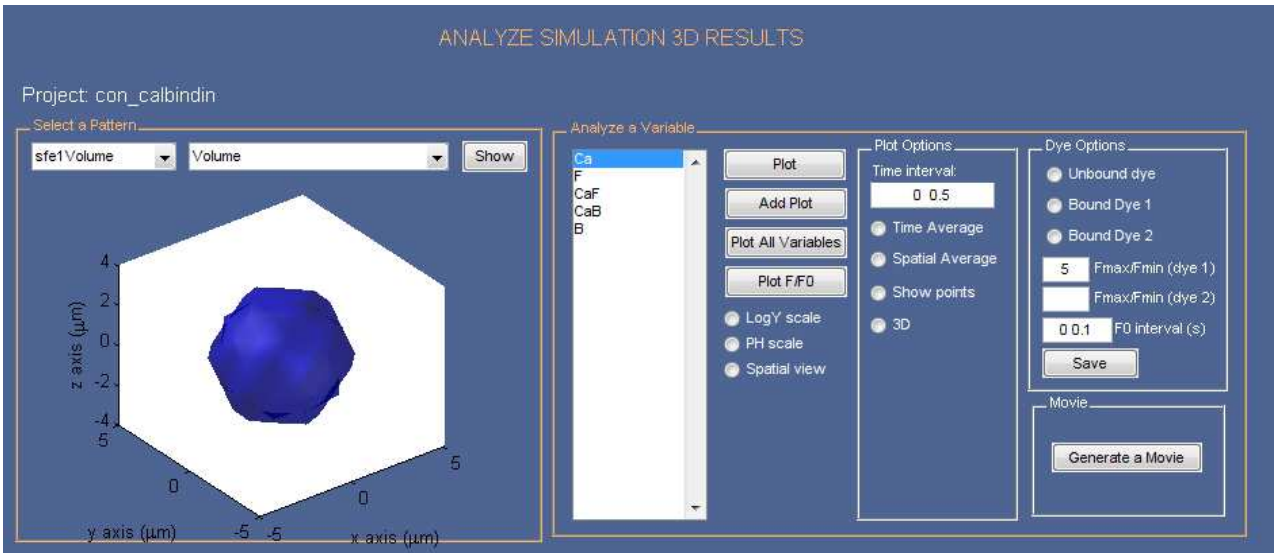

Figure 1.17: Analyze results section for 3-dimensional simulations

1.4.5 Generate a movie

The last section introduced in SimulCell is "Generate a Movie". This button opens another graphical interface (Figure 1.18).

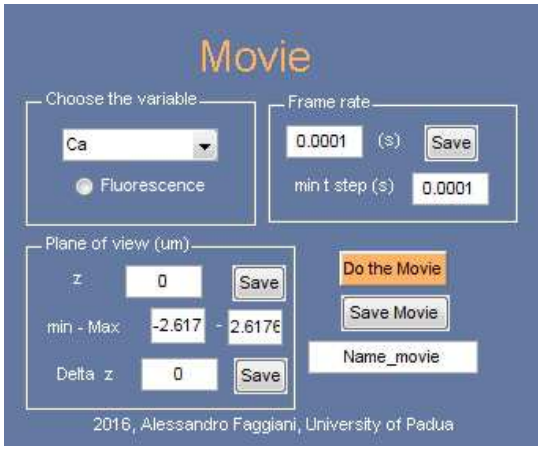

Figure 1.18: Movie section of SimulCell

This section allows the user to generate a movie of the fluorescence signal or the concentration of the reactants that simulate the video that could be recorded with a microscope. Users can choose the frame rate of the video; the duration of a frame must not be smaller than the time step of the simulation (min t step). If it is bigger, the concentration's values saved in `dataM.results.s` are mediated in every frame. The user can also choose the position of the focus plane of

the microscope modifying the value z. Delta z is the thickness of the focus plane of the microscope, so that if it is a confocal plane this value is smaller than a widefield one. If this value is 0, the thickness considered is the voxel length. When the user push the button "Do the Movie", if one variable is selected, the movie reproduces the variations of concentrations of that variable in the plane selected, if "Fluorescence" is selected, the movie shows the variation of the fluorescence signal. "Fluorescence" simulates the acquisition of a camera: the fluorescence's values are multiplied by weight values that reproduce the changes of the intensity in function of the amount of defocus from the focal plane (Figure 1.19). Then, the weighted fluorescent values are mediated along the z axis [36]

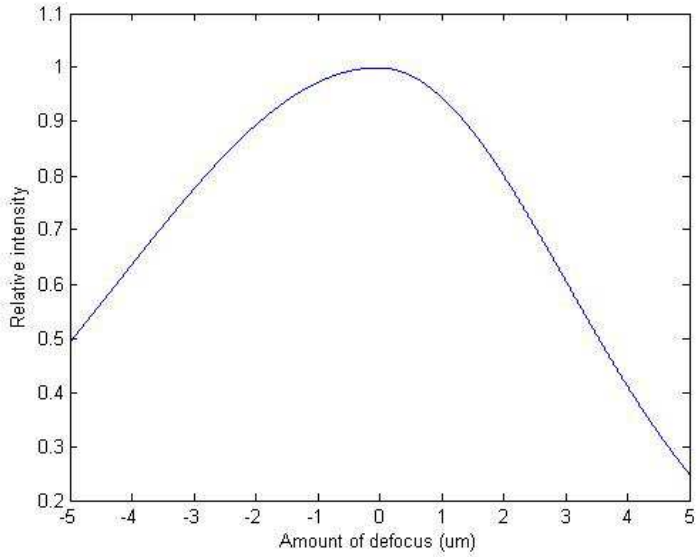

Figure 1.19: Relative intensity of the detected light of an optical fluorescence microscope in function of the amount of defocus

1.5 Three-dimensional simulations

1.5.1 Diffusion process simulation

The diffusion was first simulated outside of the SimulCell software in order to be compared with the theoretical prediction, then it was compared with SimulCell's results.

The diffusion process was simulated in a homogenous and isotrophic volume with spherical shape in which the species *c* diffuses slowly enough that its concentration at the border of the diffusion space remains negligible for the whole duration of the simulation. The simulation is thus equivalent to the diffusion in a free space.

Fick's second law

$$\frac{dc}{dt} = D \cdot \nabla^2 c \tag{1.43}$$

was discretized to simulate diffusion in squared voxels connected by probabilities related to x,y,z displacements of the *c* species. We first considered the random-walk step probabilities for the three spatial directions:

$$\begin{cases} p_x = \dfrac{6 \cdot D \cdot dt}{h_x^{\,2}} \\ p_y = \dfrac{6 \cdot D \cdot dt}{h_y^{\,2}} \\ p_z = \dfrac{6 \cdot D \cdot dt}{h_z^{\,2}} \end{cases} \tag{1.44}$$

Secondly, the probabilities of step in a neighboring box p_{neigh} or the probabilities of staying still p_{stay} are calculated with equations (1.45)

$$\begin{cases} p_{neigh}(i) = \dfrac{p_d(i)}{n(i)} \\ p_{stay}(i) = 1 - \sum\limits_{neigh(i)} p_{neigh(i)}(i) \end{cases} \tag{1.45}$$

where $n(i)$ is the number of neighbors $neigh(i)$ of the i-th box and $p_d(i) = p_x(i)$, $p_y(i)$ or $p_z(i)$. The probabilities are zeros for boxes outside of the spherical volume. Concentration in every box is calculated using this probabilities as weights, using the formula

$$c_i(t+dt) = c_i(t) \cdot p_{stay}(i) + \sum_{neigh(i)} c_i(t) \cdot p_{neigh(i)}(i) \tag{1.46}$$

where dt is the time step of the simulation.

The theoretical concentration as a function of time and space is given by

$$c(x,y,z,t) = \frac{c(x_0,y_0,z_0,t_0) \cdot (h_x \cdot h_y \cdot h_z)}{(4\pi D t)^{3/2}} \cdot e^{-\frac{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}{4Dt}} \tag{1.47}$$

where $c(x_0,y_0,z_0,t_0)$ is the initial concentration that is supposed to be completely localized inside a single box.

It is important to take into account the condition (1.48) for $x_{\min}$ that is the minimum space step of the simulation (the minimum between h_x, h_y and h_z).

$$x_{\min} > \sqrt{2D \cdot dt} \tag{1.48}$$

If the condition (1.48) is not satisfied, the results of the simulation will be wrong and the algorithm automatically changes its time step to the bigest possible.

Another important consideration to do is about the boundary conditions: if the volume is finite, the outside concentration must be constant (in this case 0 μM). In the simulations with a volume that has a boundary, reflecting boundary conditions are used. This condition is taken into account by the equation (1.45).

The simulation is performed using the initial conditions of Table 1.11. This simulations reproduce the diffusion on a infinite homogeneous volume, because the boundary is not reached. This condition is obtained using a small diffusion coefficient D.

Table 1.11: Initial conditions used for the diffusion process simulation

Parameter	Value
D	0.2 $\mu m^2/s$
dt	0.0002 s
$c(x_0, y_0, z_0, t_0)$	10 µM
Sphere radius r	3 µm
$h_x = h_y = h_z$	0.27 µm

Secondly, the diffusion process is simulated using a bigger diffusive coefficient, $D = 2\,\mu m^2/s$. In this case, the concentration is not equal to zero in the boundary regions, so that the boundary conditions are important.
Lastly, the same procedure is followed solving the Fick's law with SimulCell. The initial conditions are the same of Table 1.11.

1.5.2 Reaction-diffusion simulation with OGB-1 or OGB-1 dextran in combination with NP-EGTA as endogenous buffer

The same simulations of section 1.3.6 are performed in 3-dimensions. The results are than compared. The intial concentration and the parameters are the same of the 1-dimensional simulations. The chosen geometry is a sphere with a radius of 3 µm.

1.5.3 Flash photolysis simulation

The last simulation wants to reproduce the experiments of flash photolysis. The system of equations solved by SimulCell is (1.49). The initial concentrations and parameters are the same of sections 1.3.6 and 1.5.2. There must be introducted two more chemical species that reproduced the flashed NP-EGTA that changes its $k_{off,NP}$ from 1.36 s^{-1} to $8.0 \cdot 10^5$ s^{-1} [33] when it is enlighted by a UV laser. These two have initial concentrations equal to 0.

$$\begin{cases} \dfrac{d[Ca^{2+}]}{dt} = -k_{on,C}[Cg][Ca^{2+}] + k_{off,C}[CgCa] - k_{on,P}[Cg_p][Ca^{2+}] + k_{off,P}[Cg_pCa] \\ -k_{on,F}[F][Ca^{2+}] + k_{off,F}[CaF] + D_{Ca}\nabla^2[Ca^{2+}] \\ \dfrac{d[Cg]}{dt} = -k_{on,C}[Cg][Ca^{2+}] + k_{off,C}[CgCa] - \delta(t - t_{flash})\, f[Cg] + D_{Cg}\nabla^2[Cg] \\ \dfrac{d[CgCa]}{dt} = k_{on,C}[Cg][Ca^{2+}] - k_{off,C}[CgCa] - \delta(t - t_{flash})\, f[CgCa] + D_{Cg}\nabla^2[CgCa] \\ \dfrac{d[Cg_p]}{dt} = -k_{on,P}[Cg_p][Ca^{2+}] + k_{off,P}[Cg_pCa^{2+}] + \delta(t - t_{flash})\, f[Cg] + D_{Cg}\nabla^2[Cg_p] \\ \dfrac{d[Cg_pCa]}{dt} = k_{on,P}[Cg_p][Ca^{2+}] - k_{off,P}[Cg_pCa] + \delta(t - t_{flash})\, f[CgCa] + D_{Cg}\nabla^2[Cg_pCa] \\ \dfrac{d[F]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] + D_F\nabla^2[F] \\ \dfrac{d[CaF]}{dt} = k_{on,F}[Ca^{2+}][F] - k_{off,F}[CaF] + D_F\nabla^2[CaF] \end{cases} \tag{1.49}$$

$[Cg]$ is the $[NP-EGTA]$ and $[CgCa]$ the concentration of their Ca^{2+} complexes. $[Cg_p]$ and $[Cg_pCa]$ represent the concentration of photoproduct without and with Ca^{2+}. $\delta(t - t_{flash})$ is a Dirac delta "function" that is used to model the instantaneous conversion of a fraction of Cg and CgCa to Cg_p and Cg_pCa. This conversion is modelled by the parameter f. In Table 1.12 we reported this values as they are presented in the article [33] except for f, since they are different from experiment to experiment, as a function of the used laser. In our case, f and $\delta(t - t_{flash})$ are chosen in order to fit our experimental data. $\delta(t - t_{flash})$ is one for 5 ms, a time similar to the duration of the laser in our experiments. The parameters of the fluorescent dye are the same of the previous sections.

Table 1.12: Parameters of NP-EGTA for the Calcium uncaging process

Parameter	Value
$k_{on,C}$	$1.7 \cdot 10^7$ $M^{-1}s^{-1}$
$k_{off,C}$	1.36 s^{-1}
$k_{on,P}$	$1.7 \cdot 10^7$ $M^{-1}s^{-1}$
$k_{off,P}$	$8.0 \cdot 10^5$ s^{-1}
$k_{1D} = \dfrac{k_{1R}}{k_{1F}}$	$8.0 \cdot 10^{-8}$ M
$k_{4D} = \dfrac{k_{4R}}{k_{4F}}$	$1.0 \cdot 10^{-3}$ M
D_{Cg}	100 $\mu m^{-2}s^{-1}$

1.6 Experimental Apparatus

1.6.1 Optical fluorescence microscopy

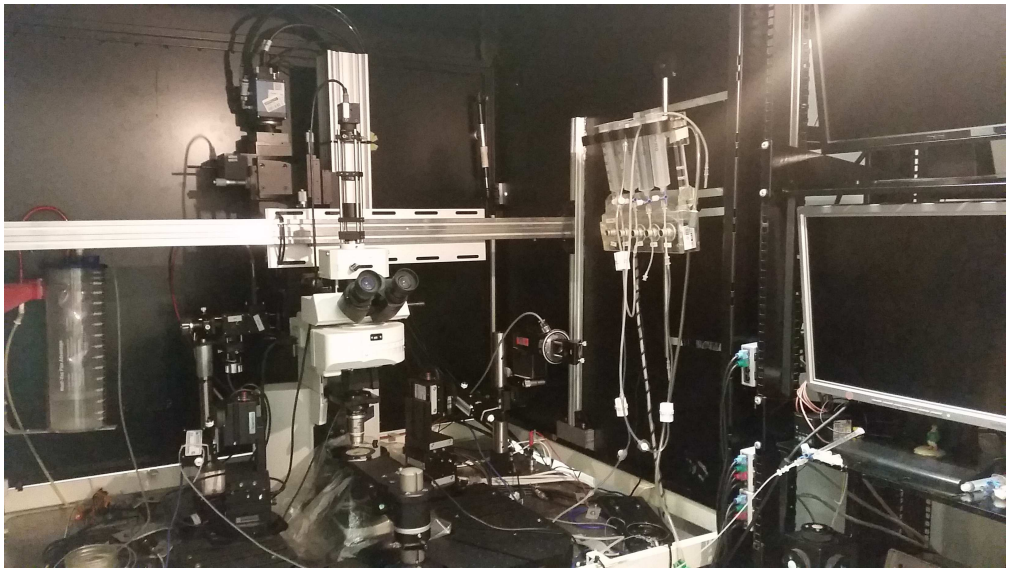

Figure 1.20: Optical fluorescence microscopy setup

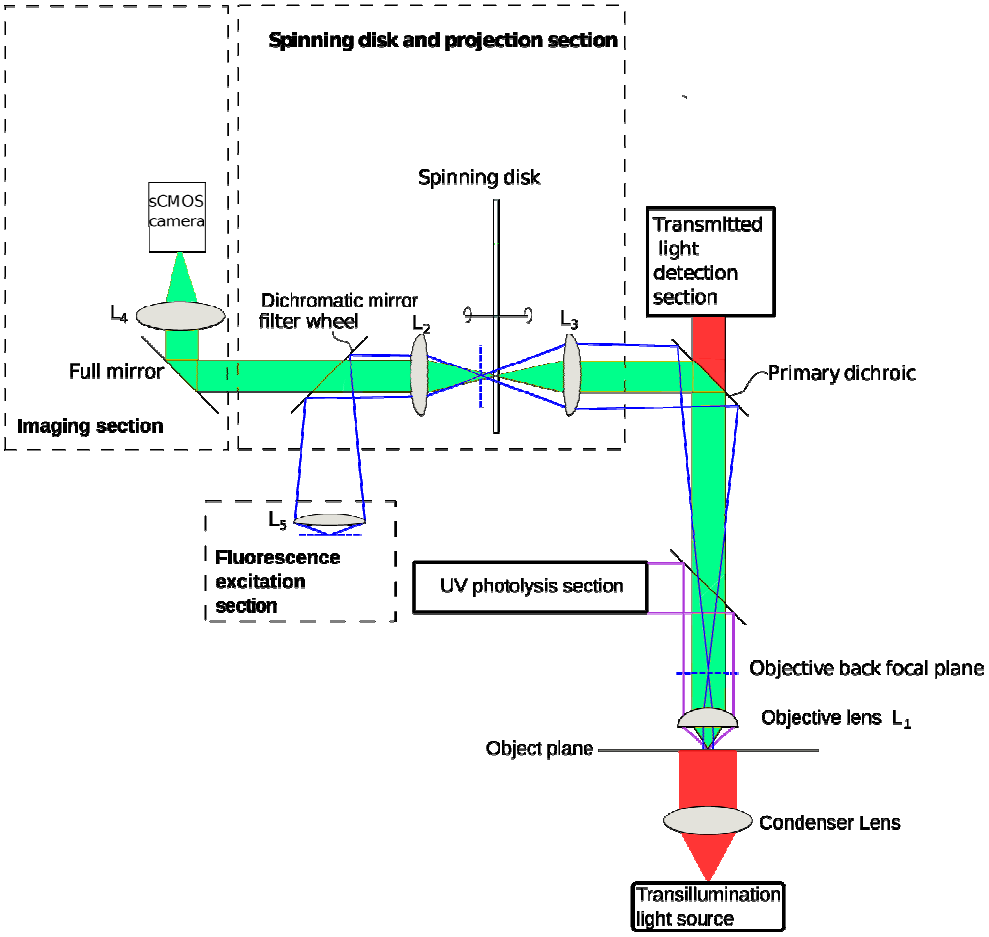

Figure 1.21: Schematic representation of the optical microscopy setup

In Figure 1.20 and Figure 1.21 we show a photo and a schematic representation of the optical fluorescent microscopy setup [37].

The microscope is a fixed stage Olympus BX51WI mounted on an anti-vibration table. The transillumination light sources are a green 520 nm or red 660 nm LED (epi-fluorescence lamphouse) used under Köhler illumination in the objective plane. This light source is used to record a transmission image of the sample directly by the oculars placed in the transmitted light detection section. These images are used to select the area of the sample to be analyzed. In order to obtain a fluorescence image of the sample, a blue 470 nm LED or a monochromator set to 471 nm coupled with a 480/40 emission filter are used as excitation source. In the UV photolysis section, there is a UV 379 nm laser that is used to photo-release Calcium bound with Caged-Calcium molecules (this is focused by the same objective used to focus image in the objective plane). The UV laser beam is reflected to the sample by a dichroic mirror 400 DCLP placed between the objective and the primary dichroic.

The objective is a water immersion Nikon Fluor 40x/0.80w, DIC M, WD 0.20. The emission light produced by the sample is collected by the objective lenses and recorded by a PCO-edge camera (PCO.EDGE 5.5). This camera allows to record very fast frames, till 10 ms/frame. To reach this recording speed the area of the sensor must be restricted. The area used for the final experiments at 10 ms/frame is 960x560 pixels, that are about 61x105 μm. The calibration is made recording an image of a ronchi test pattern with 40 lines per mm.

Depending on the wavelength used in the experiments, different dichroic mirrors and emission filters are used to filter light that reaches the sensor of the cameras (they are not shown in Figure 1.21). The dichroic mirror filter wheel has been always used with a 495 dichroic mirror. In Table 1.13 are reported the chosen configurations of dichroic mirrors and emission filters in function of the excitation light. All the parts of this experimental apparatus (cameras, leds, laser, shutters…) are controlled by Roboscope, a custom software built in VIMM laboratories. The protocols used in this software are reported in section 1.6.5.

Table 1.13: Configuration of dichroic mirrors and emission filters used for the experiments

Light source	Emission filter	Primary dichroic mirror
LED 520 nm	ET 514/44	515 + 520/40
LED 660 nm	BA610IF	None
LED 470 nm	EO 514/44	585 + 590 LP
Monochromator 471 nm	EO 514/44	585 + 590 LP

This optical microscope can be used in two ways: as a simple optical fluorescent microscope or as a confocal one. In the latter case, a Nipkow disk [37] can be inserted in the optical pathway as can be seen in Figure 1.21. In order to obtain the confocal effect, and an improvement of the resolution, the out-of-focus light is blocked by pinholes in the optical path. The spinning disk provides the confocal effect by parallelizing the process with a special design pinholes pattern. For this thesis work, we used a spinning disk with a stripes pattern (Figure 1.22). An alternation of transparent and dark lines was obtained by laser ablation of a chromium deposition over a round glass window; the pattern consists of two orthogonal sets of transparent stripes (38 μm width, 400 μm pitch) which enables a homogeneous scan of the field of view by a 360° disk rotation.

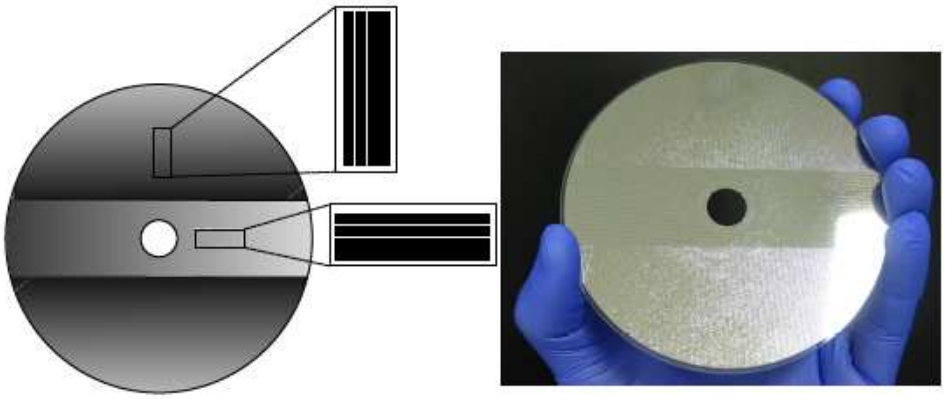

Figure 1.22: Spinning disk with stripes pattern

The disk rotation is driven by a DC motor (e.g. A-max 121394, Maxon Motor, Fall River, MA) mounted on a x-y-z micromanipulator (e.g. PT3, Thorlabs) that is used to connect the rotary disk mount to the transversal support bar and to precisely position the disk in the light path.
The samples are placed inside a micro-chamber: this is important because it allows to surround the samples with a bath solution. For living cell samples the bath solution is a mix that reproduce the extracellular medium, for non-living samples is just distilled water.

1.6.2 Laser

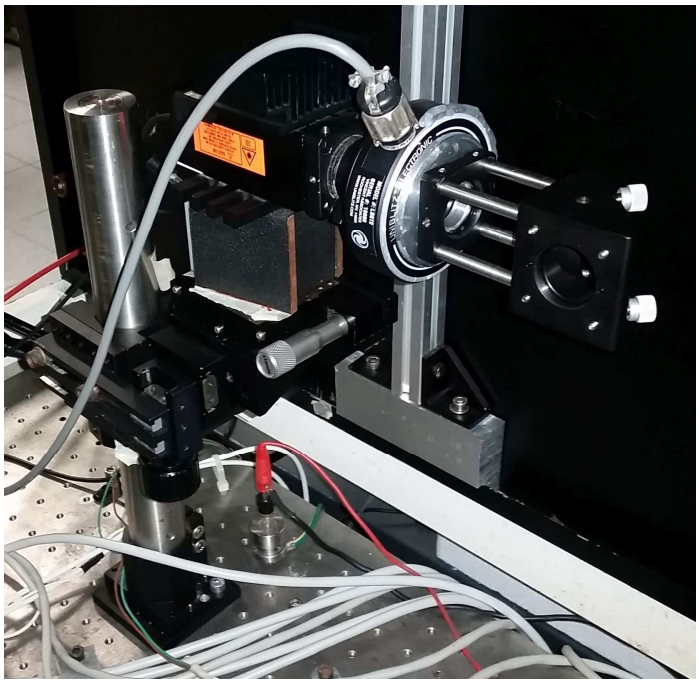

Figure 1.23: Laser used for the experiments

The laser used for the flash photolysis is a semiconductor UV laser (RGBLaseLLC) with a central wavelength of 379 nm and an optical output power of 20 mW. The Abbe equations (1.48) predicts that the radius of the laser beam shrinked by the objective is

$$r = \frac{0.61\lambda}{NA} \approx 300\ \mu m \tag{1.50}$$

We performed a qualitative measurement that is described in section 1.6.4.1 to have an idea of the dimension of the laser spot (Figure 1.25).

The basis of the laser can be moved by two linear translators on z axis and on the axis parallel to the exit beam. There is also a mirror that can be tilted to move the laser spot that reach the dichroic mirror placed before the objective of the microscope. In Figure 1.24 is shown a scheme of the laser optical path with the power of the laser spot measured by a Thorlabs PM100D powermeter with a Thorlabs S175C probe in different place of the pathway.

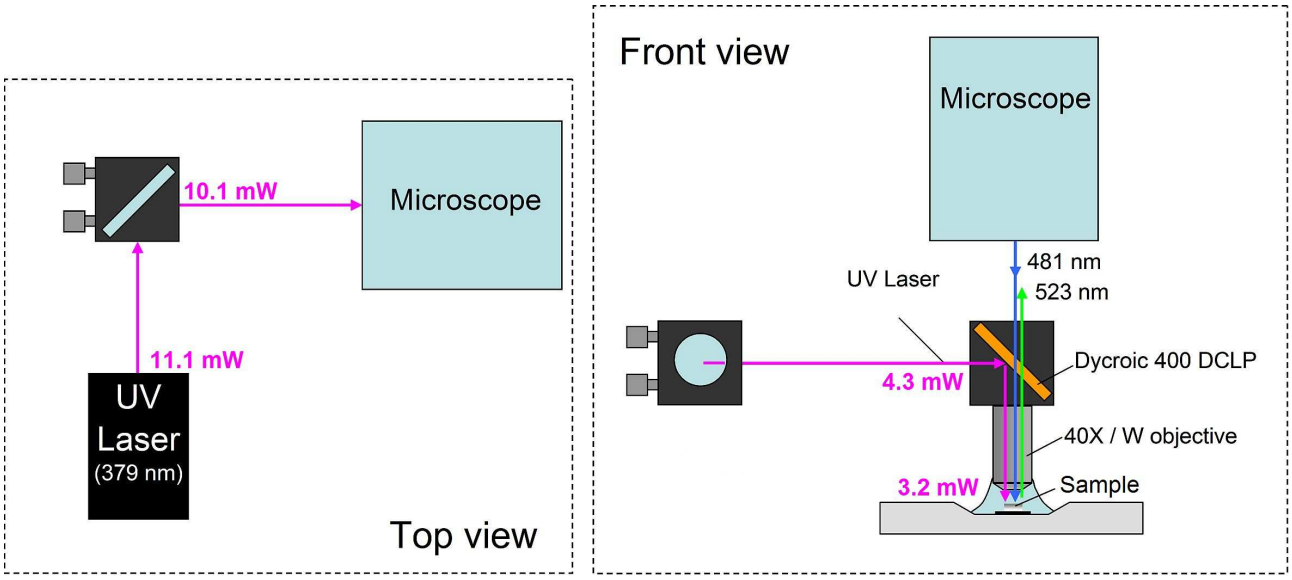

Figure 1.24: Scheme of the laser optical pathway

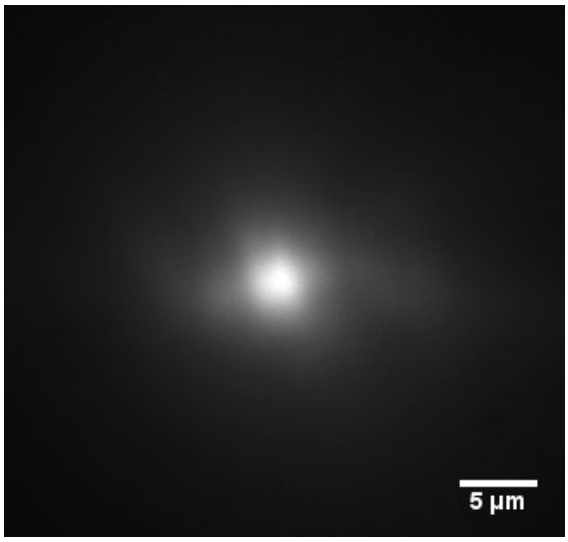

Figure 1.25: Laser spot in a solution of fluorescein.

1.6.3 Reactants

In this section we will describe the principal reactants used for the experiments: the fluorescent dyes, the caged-Calcium compound and the intracellular solution.

1.6.3.1 Intracellular solution

This particular solution is used instead of water in the samples to reproduct the conditions that are typical of the inside of a cell, particularly the pH value. In the Table 1.14 we show the reactants used to produce this solution.

Table 1.14: Reactants used for the intracellular solution

Reactant	Molecular weight (g/mol)	Molarity (mM)
K-asp	180.2	115
KCl	74.56	10
NaCl	58.44	10
HEPES	238.30	10
KOH	56.11	(1000)

HEPES (*4-(2-hydroxyethyl)-1-piperazineethanesulfonic acid*) is a zwitterionic organic chemical buffering agent widely used in cell culture, largely because it is better at maintaining physiological pH values. The solution of KOH is 1 M and it is added to the solution until the pH reaches a physiological value of about 7.2. The final intracellular solution has an osmotic pressure of about 280 mOsm and a pH of 7.2.

1.6.3.2 Extracellular solution

This particular solution is used instead of water in the bath solution of the samples to reproduct the conditions that are typical outside a cell, particularly the pH and osmotic pressure values.
In the Table 1.15 are shown the reactants used to produce this solution.

Table 1.15: List of ionic concentration in extracellular solution

Ionic species	Molarity (mM)
K^+	5
Cl^-	160
Na^+	150
Ca^{2+}	2

1.6.3.3 Fluorescent dyes

A fluorophore is a fluorecsent chemical compound that can re-emit light upon light excitation. Fluorophores are sometimes used alone, as tracers in fluids, as a dye for staining of certain structures, as a substrate of enzymes or as a probe or indicator (when its fluorescence is affected by environmental aspects such as polarity or ions). More generally they are covalently bonded to a macromolecule, serving as a marker (or dye, or tag, or reporter) for affine or bioactive reagents (antibodies, peptides, nucleic acids) [20].

The fluorophore absorbs light energy of a specific wavelength and re-emits light at a longer wavelength. The absorbed wavelengths, energy transfer efficiency, and time before emission depend on both the fluorophore structure and its chemical environment, as the molecule in its excited state interacts with surrounding molecules. Wavelengths of maximum absorption (≈ excitation) and emission (for example, Absorption/Emission = 485 nm/517 nm) are the typical terms used to refer to a given fluorophore, but the whole spectrum may be important to consider. The excitation wavelength spectrum may be a very narrow or broader band, or it may be all beyond a cutoff level. The emission spectrum is usually sharper than the excitation spectrum, and it is of a longer wavelength and correspondingly lower energy. For Calcium imaging Calcium probes named Calcium fluorescent dyes are fundamental because they are capable of sensing local on concentration with high selectivity.

In order for such a fluorescent probe to provide useful information about its environment (e.g. Ca^{2+} concentration), it is necessary that its spectral properties be altered in a suitable manner by the parameter to be measured. For most biological applications, any one of the following three property changes is appropriate (Figure 1.26):

a. A change in fluorescence yield
b. A shift in the excitation or emission spectrum
c. A combination of two

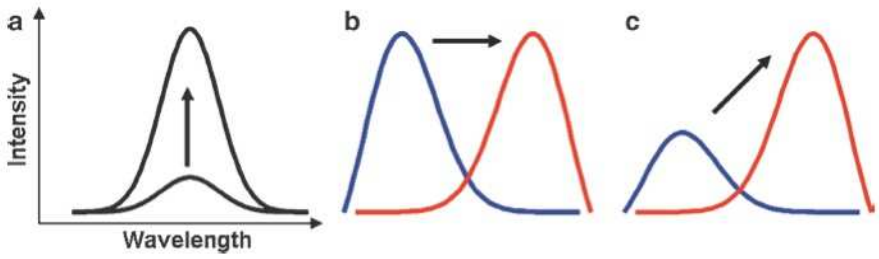

Figure 1.26: Spectral proprieties that can be utilized to measure Ca^{2+} concentration

As discussed in the introduction, there are three types of fluorescent dyes: case (a) comprises the Fluo family of the so-called *single wave- length* uorescent Ca^{2+} indicators which are the ones that are used for this thesis work, particularly Fluo-4 and Oregon Green BAPTA-1. This two types of probes can be cell permeant (AM (acetoxymethyl ester) form) or cell impermeant (salt form). The first ones must be loaded in the extracellular medium and they are absorbed and de-esterificated by cells: after this processes they works but the concentration that is absorbed by every celli is different and unknown. The second ones are loaded in the cells by microinjection: this fact permits to control the concentration that are inserted inside the analyzed cell. For this thesis work are used fluorescent dyes in salt form. The specification of

the two fluorescence dyes used are shown below.

Fluo-4, Pentapotassium Salt, cell impermeant

- Molecular formula: $C_{36}H_{25}F_2K_5N_2O_{13}$
- Molecular weigh: 927.0874 g/mol
- Excitation/emission wavelength: 494/516 nm
- Fluorescence intensity increase upon binding Ca^{2+}: > 100 fold
- k_d: 335 nM

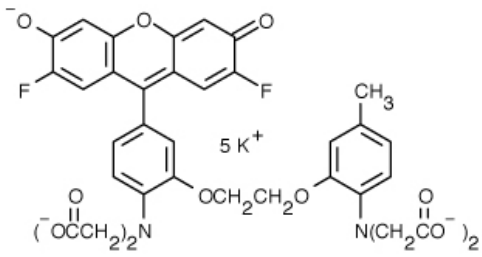

Figure 1.27: Scheme of Fluo-4 molecule

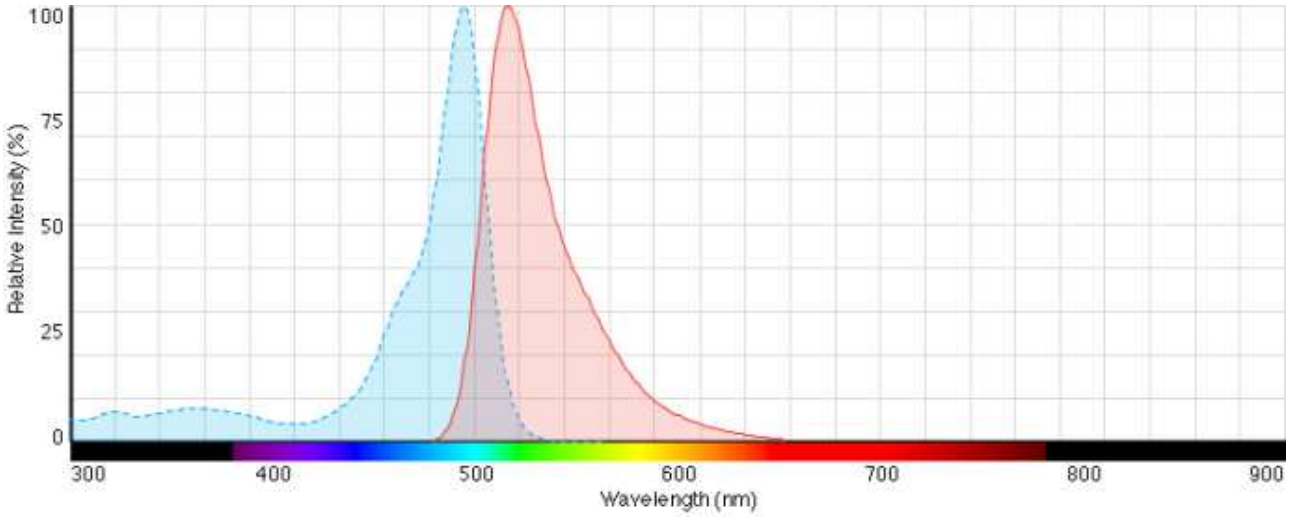

Figure 1.28: Fluo-4 spectra: excitation spectrum in cyan, emission spectrum in red

Oregon Green 488 BAPTA-1, Hexapotassium salt, cell impermeant

- Molecular formula: $C_{43}H_{27}F_2K_6N_3O_{16}$
- Molecular weigh: 1114.2834 g/mol
- Excitation/emission wavelength: 494/523 nm
- Fluorescence intensity increase upon binding Ca^{2+}: > 14 fold
- k_d: 206 nM

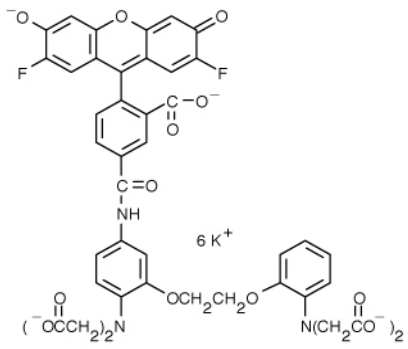

Figure 1.29: Scheme of OGB-1 molecule

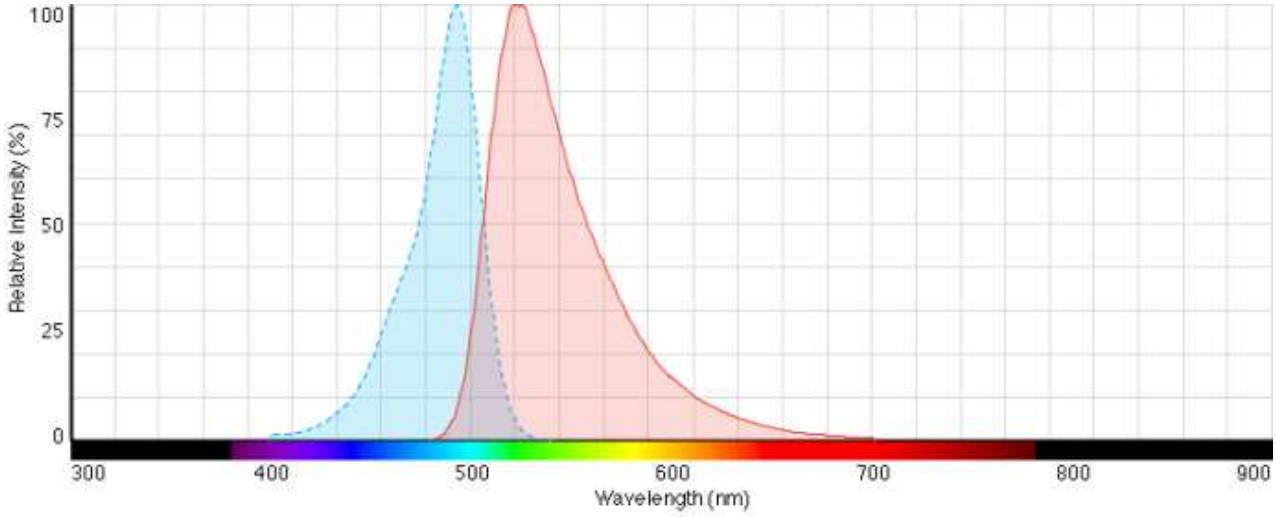

Figure 1.30: Oregon Green BAPTA-1 spectra: excitation spectrum in cyan, emission spectrum in red

Oregon Green 488 BAPTA-1 dextran, Potassium salt, 10000 MW, cell impermeant

This fluorescent dye has the same specification of the OGB-1 except for the molecular weight. The dextran size is about 10000 MW.

1.6.3.4 Caged Calcium molecules

Calcium cage molecules are photolabile Ca^{2+} chelators that can be used to increase the intracellular concentration of Ca^{2+}. They change their binding rates and their proprieties after been flashed by a UV light. The caged Calcium chosen for this thesis work is NP-EGTA (O-Nitrophenyl EGTA, Tetrapotassium Salt, cell impermeant).

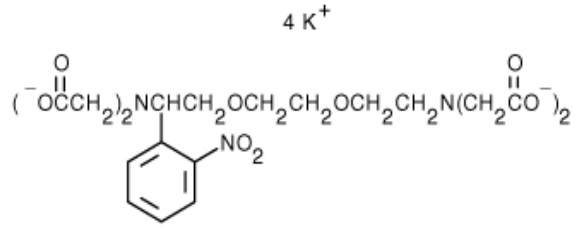

Figure 1.31: Scheme of NP-EGTA molecule

NP-caged EGTA, potassium salt is a cell-impermeant photolabile chelator that exhibits a high selectivity for Ca^{2+} upon UV illumination, its k_d increases from 80 nM to >1 mM. The NP-EGTA salt can be complexed with Ca^{2+} to generate a caged calcium complex that will rapidly deliver Ca^{2+} upon photolysis. In is shown the process of Ca^{2+} uncaging by photolysis of NP-EGTA. Ca^{2+} is efficiently released by photochemical lysis of the chelator backbone, converting a high affinity, tetracarboxylic acid chelator into two low affinity dicarboxylic molecules [33]. They are shown just the first and the last step of the photolysis [38].

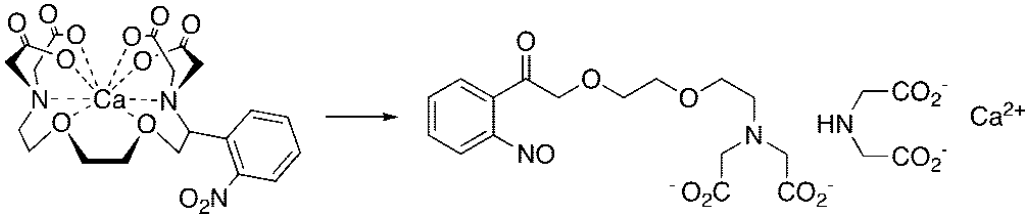

Figure 1.32: Calcium uncaging by photolysis of NP-EGTA

The uncaging process can be modelled by a simplified system of equation reported in section 1.5.3 [33], with the scheme reported in Figure 1.33

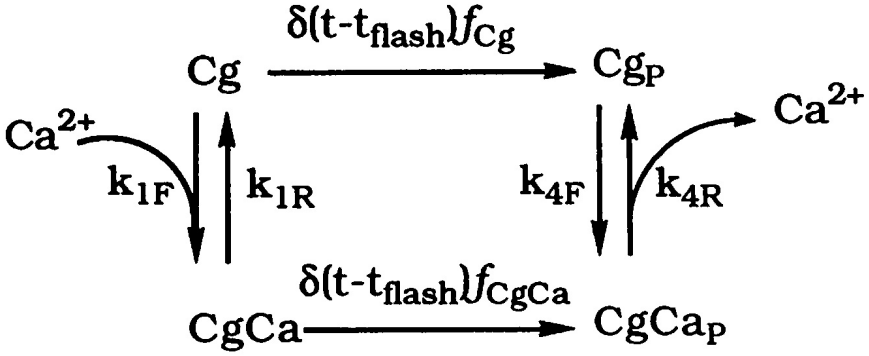

Figure 1.33: Kinetic scheme for modeling the rate of Ca^{2+} release upon photolysis of caged Ca^{2+}

As the fluorescent dyes, NP-EGTA can be cell permeant, too. This one has been used for a test with HeLa cells.

1.6.4 Samples

The samples that we used for the experiments are “sandwiches”, couvettes and HeLa cells.

1.6.4.1 “Sandwich” couvette

“Sandwiches” are prototypes of cells made by two glass coverslips. Between this two coverslips there is a spacer of about 70 μm. The space between the coverslips is filled with little drops (about 0.50 μl) of different solutions.
The first samples are filled with drops of a 2.5 mM solution of fluorescein: they are used to focalize the laser and to characterize it, recording his changes in size and intensity. In this way, it is possible to find the smaller laser spot possible. His diameter is $3.0 \pm 0.3\,\mu m$ and will be the size of the uncaging spot. The other samples are filled with drops of solutions with known concentration of reactants. In Figure 1.34 is shown a schematic representation of the “sandwiches” samples. This type of samples are used for the final measurements (the final “receipt” is reported in section 1.6.5).

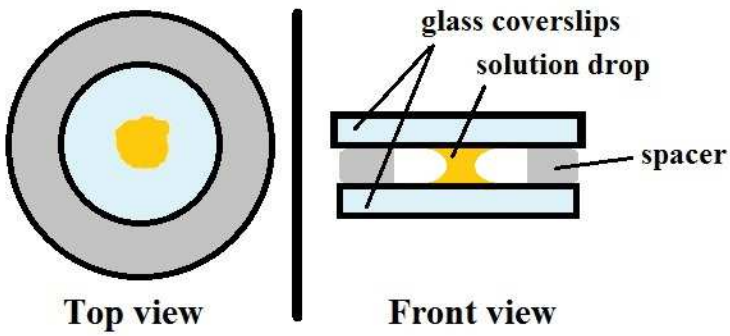

Figure 1.34: Scheme of the sandwiches samples

1.6.4.2 “Pipette” couvette

These samples are made by a Patch glass capillary (1.5 mm OD, 1.16 mm ID, HARVARD APPARATUS LTD) pulled by a micropuller. It is possible to pull the glass in one stage using a microelectrode puller and a standard heating coil (Figure 1.35). The temperature of the heating coil is about 57°C. The capillary breaks in two pieces, taking the form of a little pipette with a tip that has a diameter of about 2-4 μm.

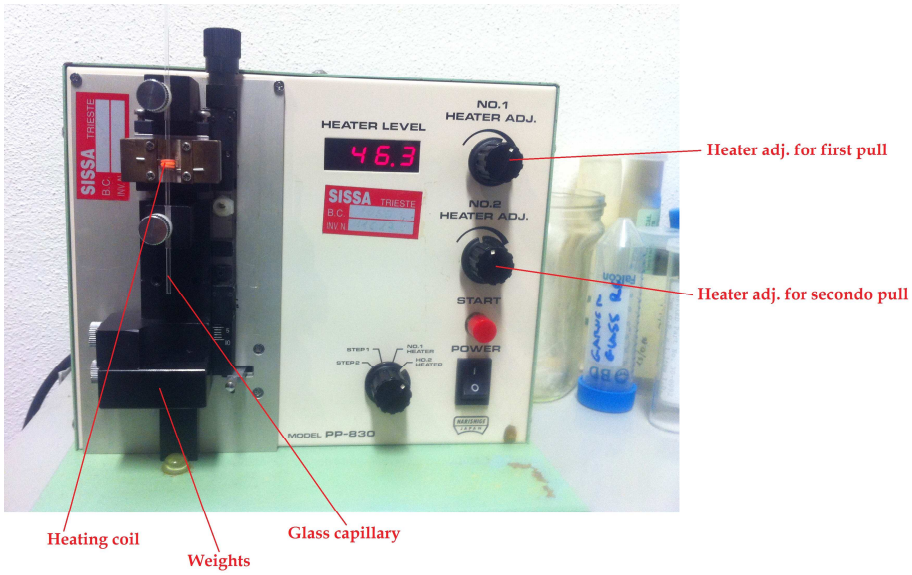

Figure 1.35: Vertical micro-puller

This capillary can be used for patch clamp to fill the cells with known solution but can be also used as micro-containers of solutions if their head is closed. The latter case is what we call “couvette samples”.
The tip of the pipette is closed by a micro-forge (Figure 1.36 and Figure 1.37).

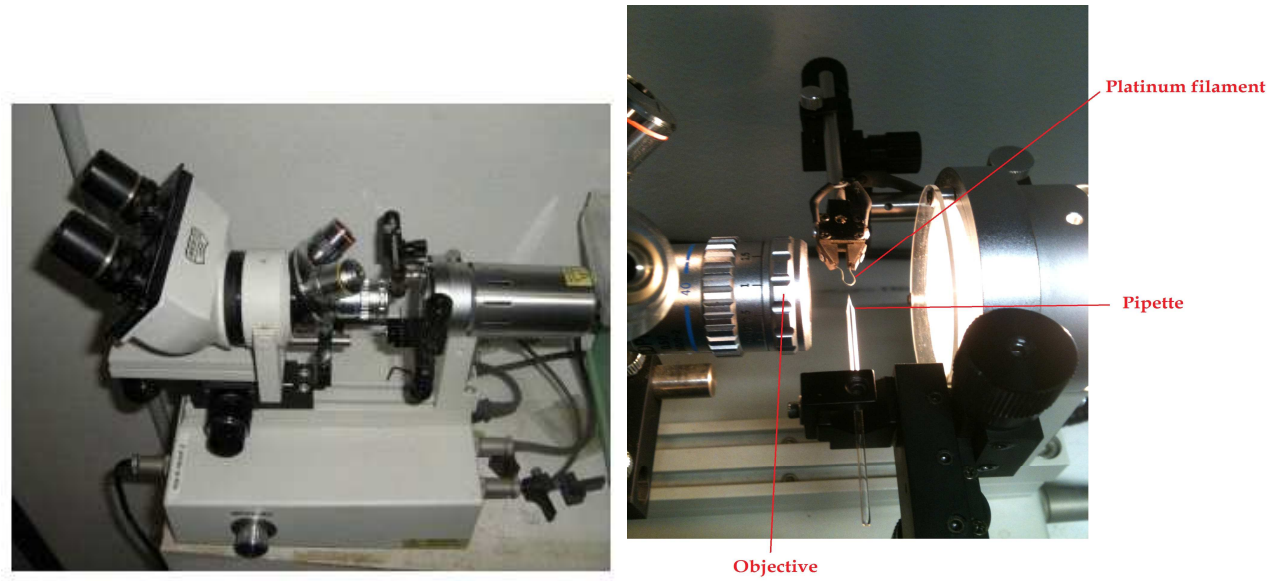

Figure 1.36: Microforge: the heat is supplied by the platinum filament. The tip of the pipette is brought to within 10 µm until it is closed.

This closed pipettes are filled with the chosen solution and are corked and pasted on a glass coverslip with hot glue. These samples are used for test measurments.

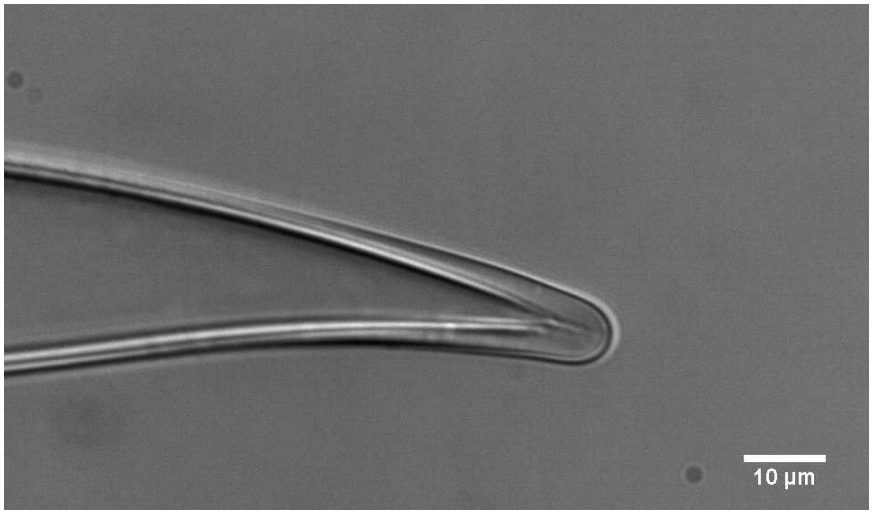

Figure 1.37: Couvette's tip recorded with transillumination of 520 nm

This samples are used for testing the reactants used for the experiments. Many tests have been made, since we had different aliquots of NP-EGTA but not all of them worked well. These tests are also useful to find a good "receipt" for the UV flash photolysis samples. The protocols used in Roboscope are the same of the sandwiches samples.

1.6.4.3 HeLa cells

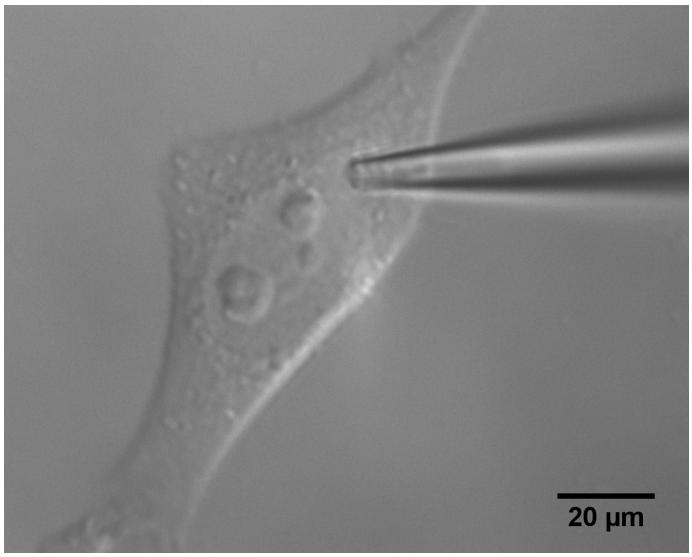

Figure 1.38: HeLa cell used for a test

This kind of cells are used as cell model systems in many biology laboratories around the world.

HeLa cells are the first continuous cancer cell line scientist managed to grow in vitro; they are isolated from the aggressive glandular cervical cancer of a young woman, Henrietta Lacks, in 1951 in Baltimore. The cell line was called HeLa from the first two letters of the woman's name.

They are particularly used in bio labs because they are "immortal" in that they can divide an unlimited number of times in a cell culture plate as long as fundamental cell survival conditions are met [39]. Cells are grown in Dulbecco's

modified Eagle's medium (DMEM) supplemented with penicillin streptomycin (pen strep) and fetal bovine serum (FBS) at 37°C in a humidified incubator with 5% CO_2.

Cells are routinely subcultured by trypsinazation with a change of medium three times a week. They are plated in 35 m plastic Petri dishes; each dish hosted up to five cover glasses with a diameter of 12 mm on which cells are disseminated and allowed to grow. These cells are used for this thesis work in the preliminary test experiments.

These samples are used in order to have an idea of how a localized flash photolysis works in cells and to test if the apparatus can perform it. The "receipt" and the protocol used in Roboscobe are reported in section 1.6.5. In Figure 1.39 is shown the process of photorelease revealed by a fluorescentce measurement. The laser spot is set at the right edge of the cell.

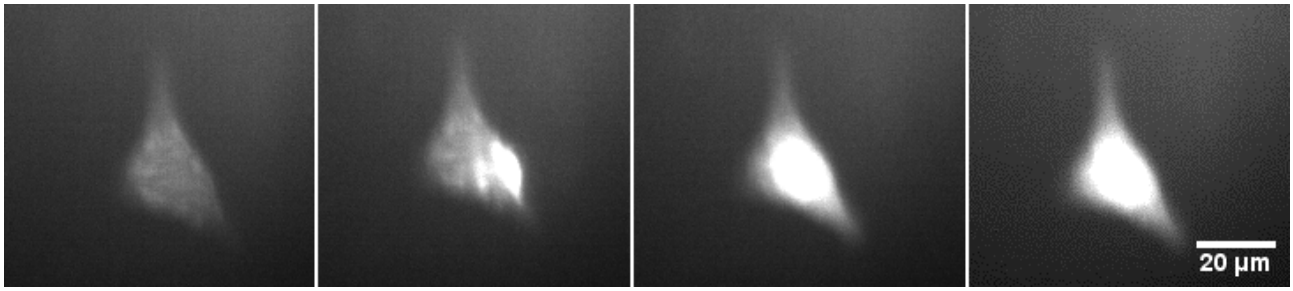

Figure 1.39: UV flash photolysis on HeLa cell

In Figure 1.40 are also reported the $\Delta f / f_0$ traces on these samples. They show a typical Ca^{2+}-release behaviour. We are not sure if all that fluorescent is due to Ca^{2+} diffusion or if it is also caused by other processes, such as Ca^{2+}-induces Ca^{2+}-release (because cells were not loaded with specific blockers). The decrease of the fluorescence as a function of time is due to the Ca^{2+} channels of the cellular membrane.

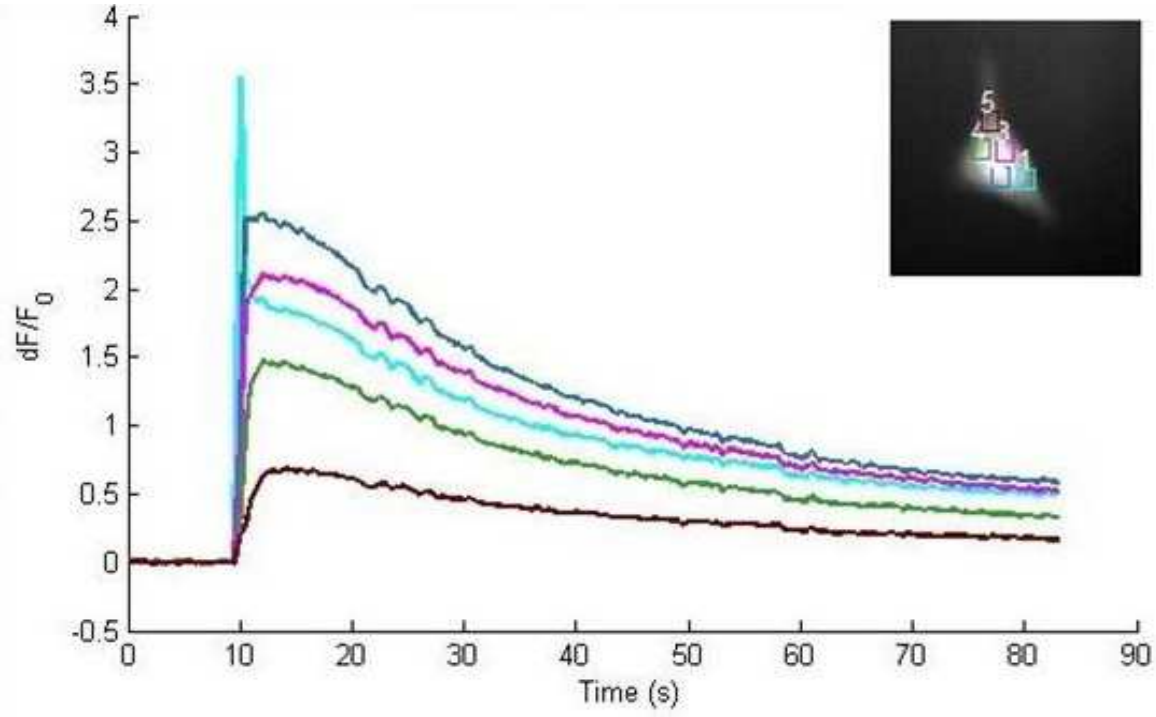

Figure 1.40: $\Delta f / f_0$ traces of the flash photolysis on HeLa cells

1.6.5 Experimental protocols

In this section we reported the "receipts" of the solutions used for the experiments and the protocols followed by Roboscope for the recording of the images.

1.6.5.1 Solutions

For the first couvette and sandwiches samples we used different concentrations of the reactants, trying to find the best concentration for our experiments.
For the HeLa cells samples, the fluorescent dye and the caged Calcium are loaded inside the grown medium of the cell (Table 1.16). Then, the cells are placed on the microscope, surrounded by extracellular medium.

Table 1.16: Solution for HeLa cells sample

Reactant	Concentration
Fluo-4 AM	50 μM
NP-EGTA AM	50 μM

The sandwiches samples that are the samples used for the final experiments, are filled with the solution reported in Table 1.17, using intracellular solution instead of water to dilute the solutions. We called these sample "NP". For the final analysis, some sandwiches are also filled with a solution without NP-EGTA, named OGB, and others with the fluorescent dye completely bound and unbound with Ca^{2+} ions, used for the calibration of F_{max} and F_{min} , named OGBmax and OGBmin. Some samples are also filled with only intracellular solution in order to evaluate the background. We built three samples for every solution to mediate the results during the analysis.

Table 1.17: Receipt used for couvettes ("sandwiches")

Sample	Reactant	Concentration
NP	$CaCl_2$	900 μM
	NP-EGTA salt	2 mM
	OGB-1 or OGB-1 (dextran)	250 μM
OGB	$CaCl_2$	50 μM
	OGB-1 or OGB-1 (dextran)	250 μM
OGBmax	$CaCl_2$	2 mM
	OGB-1 or OGB-1 (dextran)	250 μM
OGBmin	BAPTA	5 mM
	OGB-1 or OGB-1 (dextran)	250 μM

1.6.5.2 Protocols

In this section we reported the protocols used for the test experiments with HeLa cells (Table 1.18) and for the final "sandwhich" samples (Table 1.19).

Table 1.18: Protocol for HeLa cells sample

Parameter	Value
binning	2
Frame duration	500 ms
Exposure time	50 ms
Laser duration	~300 ms
Duration of exposure with LED 470 nm	50 ms/frame

Table 1.19: Protocol for final "sandwich" samples

Parameter	Value
binning	1
Frame duration	11 ms
Exposure time	10 ms
Laser duration	~2 ms or ~100 ms
Duration of exposure with the monochromator 471 nm	11 ms

2 Results

In this chapter we will be expose all the achieved results: the development of the novel formula for the reconstruction of Calcium concentration and the results of the simulations and the experiments.

2.1 Calcium influx simulated in one-dimensional geometry

2.1.1 Reaction-diffusion simulation with calbindin buffer

Here are reported the results from the first simulation described in section 1.3.1.

2.1.1.1 Reconstruction of the real Ca^{2+} concentration by equilibrium and derivative formulas

In Figure 2.1 is shown the simulated $[Ca^{2+}]$ in voxels 50, 45, 40, 30, 20 and 10. The initial conditions and the parameters chosen for this simulation are reported in section 1.3.1.

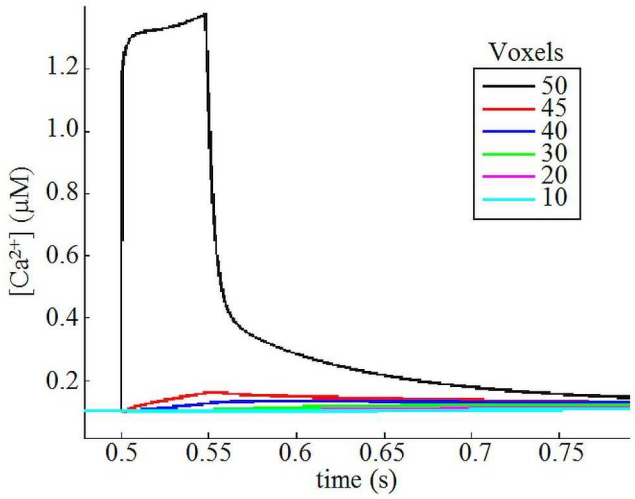

Figure 2.1: Simulated $[Ca^{2+}]$ in different voxels

In Figure 2.2 are reported the concentrations of the dye bound and unbound with Ca^{2+} to show that the dye does not saturate in every voxel (it is necessary for a correct reconstruction).

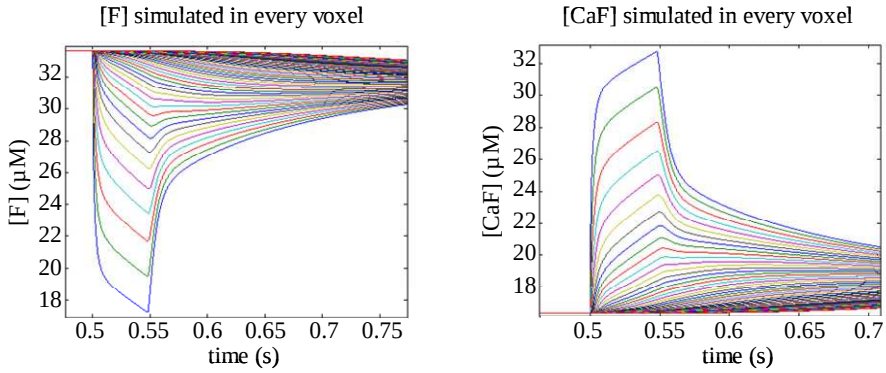

Figure 2.2: simulated dye concentration unbound (left) and bound (right) with Ca^{2+} in every voxel

Starting from the simulated $\Delta f / f_0$ (Figure 2.3), $[Ca^{2+}]$ can be initially reconstructed by the equilibrium formula (1.30) (Figure 2.4).

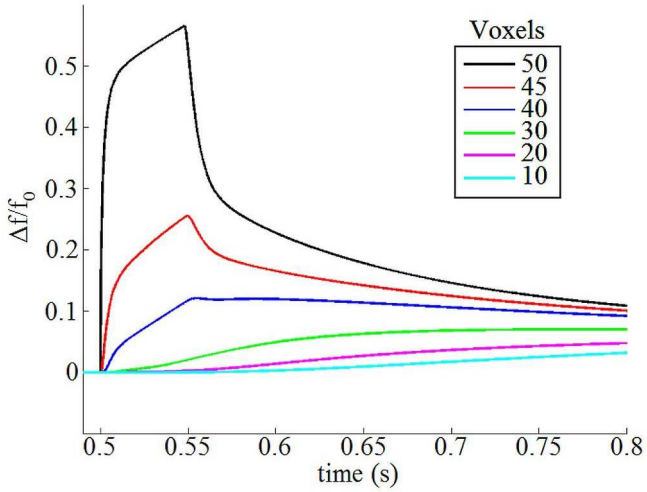

Figure 2.3: Simulated $\Delta f / f_0$ in different voxels

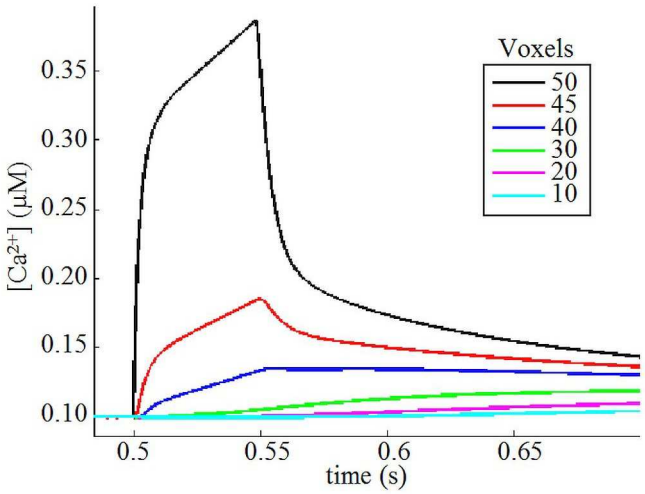

Figure 2.4: Reconstructed $[Ca^{2+}]$ with the equilibrium formula (1.30) in different voxels

The concentration $[Ca^{2+}]$ can be reconstructed from $\Delta f / f_0$ with the formula (1.28) as can be seen in Figure 2.5.

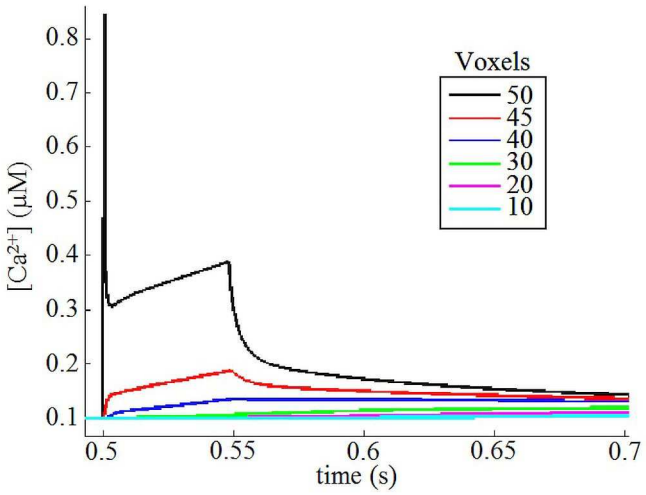

Figure 2.5: Reconstructed $[Ca^{2+}]$ with (1.28) in different voxels

The spikes of $[Ca^{2+}]$ in the voxel 50 are due to the $\Delta f\,'/\,f_0$ term. The variation of fluorescence in the voxel 50, where the $[Ca^{2+}]$ influx appear for 0.01 s, is very fast both in the beginning and in the end, so the derivative term presents some spikes (Figure 2.6) that are smoothed by diffusion for other voxels. The derivative is evaluated point by point by a simple two-points algorithm

$$\Delta f\,'/\,f_0 = \frac{(\Delta f\,/\,f_0)_{i+1} - (\Delta f\,/\,f_0)_i}{\Delta t} \tag{2.1}$$

Where Δt is the temporal gap between two successive points, that is the resolution time of the simulation; in this case $\Delta t = 0.0001$ s.

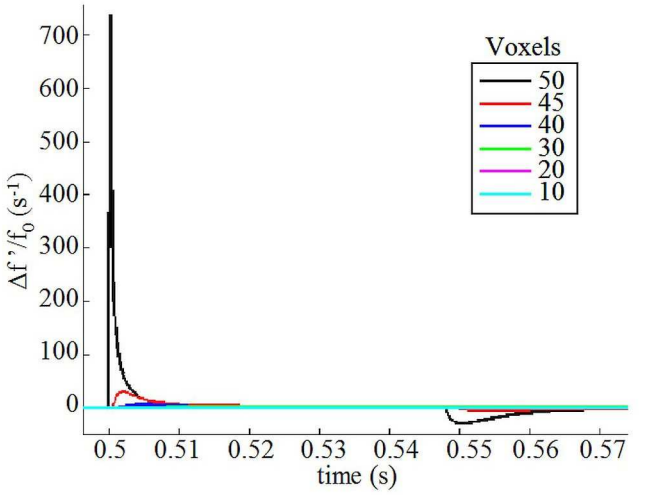

Figure 2.6: $\Delta f\,'/\,f_0$ in different voxels

In Figure 2.7 the simulated and the two reconstructed concentration are compared in voxels 50, while in in voxel 45 and 20 in Figure 2.8.

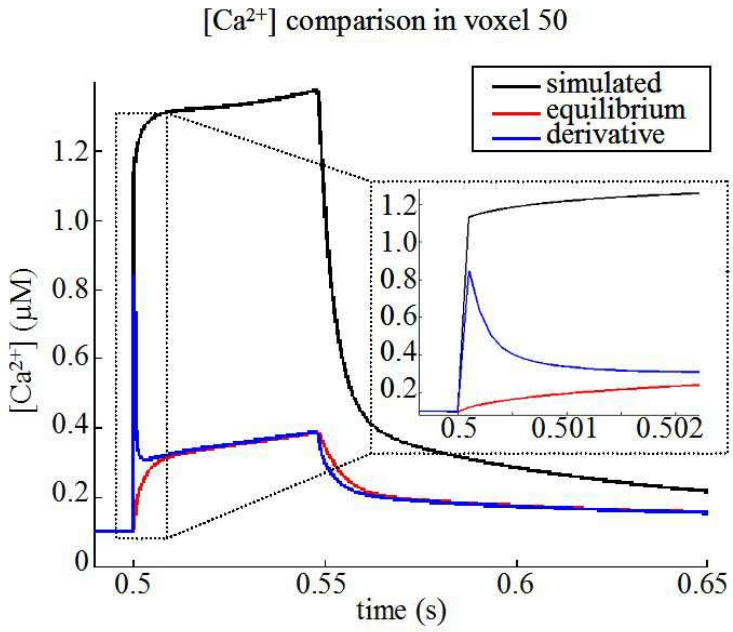

Figure 2.7: Comparison in voxel 50

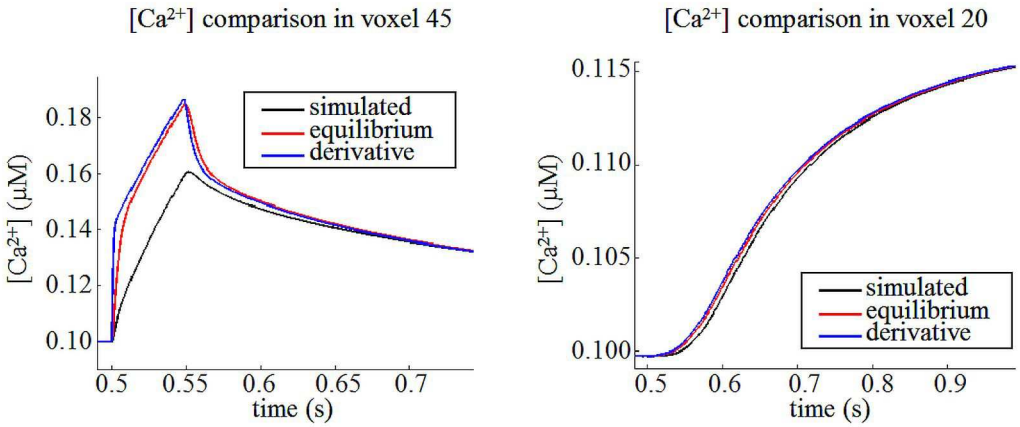

Figure 2.8: Comparison in voxel 45 (left) and 20 (right)

The reconstruction is unsatisfying. It is different from the simulation, particularly for the central voxels where is the Calcium influx. It is interesting that the formula (1.28) reconstructed better $[Ca^{2+}]$ than the formula (1.30), particularly in the beginning of the influx. This happens because it takes into account the dynamic of the system with the first derivative.

2.1.1.2 Reconstruction of the real Calcium concentration by the new formula

In the reconstruction equations (1.28)(1.30) a diffusion contribution is ignored. This term is introduced in the typical reaction equation.

$$\frac{d[F]}{dt} = -k_{on,F}[Ca^{2+}][F] + k_{off,F}[CaF] + D_F \nabla^2 [F] \tag{2.2}$$

We know that [20]

$$[F] = \frac{f / S_F - \alpha [F]_{tot}}{1-\alpha} \tag{2.3}$$

And inserting it in (2.2)

$$[Ca^{2+}] = k_{d,F} \frac{[F]_{tot}(1-\alpha) S_F - f - \alpha S_F [F]_{tot} + D_F \frac{S_f (1-\alpha) \nabla^2 f}{k_{off,F}} - \frac{1}{k_{off,F}} f'}{f - \alpha S_F [F]_{tot}} \tag{2.4}$$

The Laplace operator can be discretized in the following way

$$\nabla^2 f_i = \frac{f_{i+1} - 2 f_i + f_{i-1}}{\Delta x^2} \tag{2.5}$$

where f_i is the fluorescence intensity of the dye at the position i, f_{i+1} and f_{i-1} are the intensity at positions $i + \Delta x$ and $i - \Delta x$, where Δx is the spatial resolution of the simulation, in this case $\Delta x = 0.25$ μm.

Calling (1.30) as $[Ca^{2+}]_R$, considering the discretization (2.6), (2.4) can be expressed as

$$[Ca^{2+}] = [Ca^{2+}]_R + k_{d,F} D_F \frac{(\Delta f / f_0)_{i+1} - 2(\Delta f / f_0)_i + (\Delta f / f_0)_{i-1}}{k_{off,F} (\Delta x)^2 \left(\frac{f - \alpha S_F [F]_{tot}}{f_0} \right)} \tag{2.6}$$

Summing and subtracting f_0 inside the parenthesis of the denominator of (2.6),

$$[Ca^{2+}] = [Ca^{2+}]_R + k_{d,F} D_F \frac{(\Delta f / f_0)_{i+1} - 2(\Delta f / f_0)_i + (\Delta f / f_0)_{i-1}}{k_{off,F} (\Delta x)^2 \left(\Delta f / f_0 - \gamma k_d^F \right)} \tag{2.7}$$

Where

$$\gamma k_{d,F} = \frac{\alpha S_F [F]_{tot}}{f_0} - 1 \tag{2.8}$$

As in the previous section, from Figure 2.9 the reconstructed $[Ca^{2+}]$ with the diffusive correction can be seen.

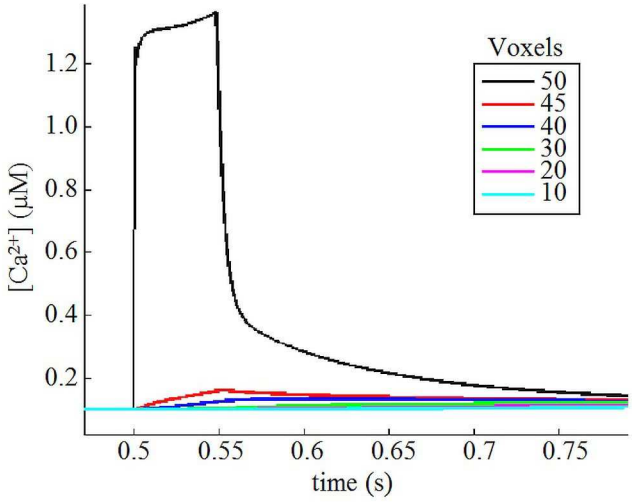

Figure 2.9: $[Ca^{2+}]$ reconstructed with diffusive correction

As can be seen in Figure 2.10, in voxel 50 the reconstruction with the diffusive term overlaps with the simulated $[Ca^{2+}]$.

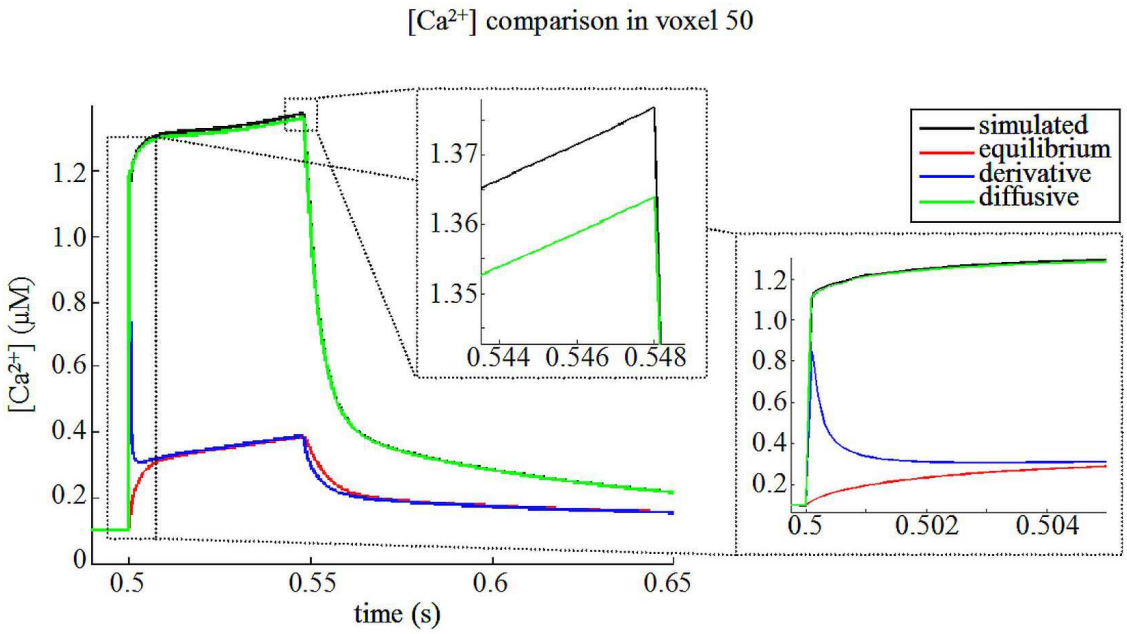

Figure 2.10: $[Ca^{2+}]$ comparison in voxel 50

These four estimates are compared in Figure 2.11 for the voxels 45 and 20 as before.

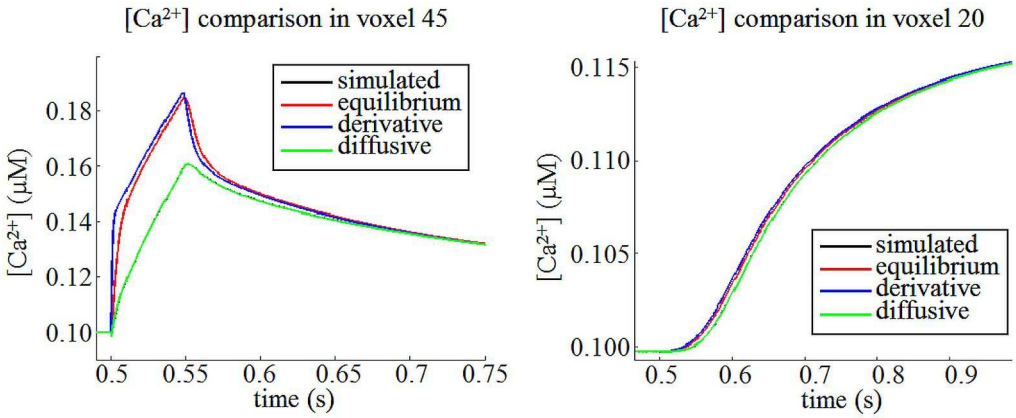

Figure 2.11: $[Ca^{2+}]$ comparison in voxels 45 (left) and 20 (right)

For all the voxels, the equation (2.7) describes very well the behaviour of $[Ca^{2+}]$; simulated and corrected plots overlaps. The diffusive correction is fundamental to reconstructed the $[Ca^{2+}]$ concentration.

2.1.2 Comparison of equilibrium and new formula by changing the dye diffusion coefficient.

As described in section 1.3.2, this is the first validity test of the novel formula. With $D_F = 0$, all the reconstruction formulas have to overlap with the simulated Calcium concentration. This fact is reported in Figure 2.12. It can be seen that reconstructions from derivative and equilibrium formula are very similar: the first is better than the second especially in the beginning of the influx, because in that moment the dynamic of the system is very important. The percentage error between the reconstructions and the simulation on the top of the influx is about 6 %.

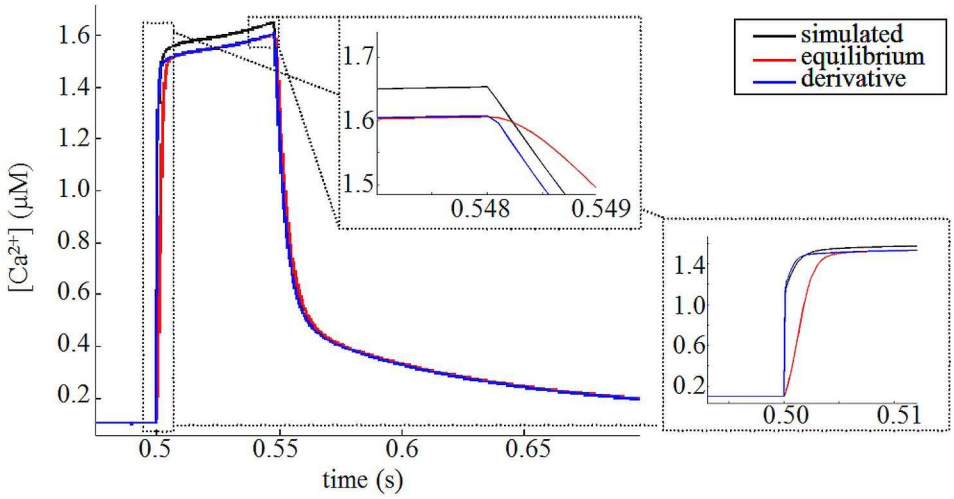

Figure 2.12: $[Ca^{2+}]$ reconstructions without dye diffusion in voxel 50

2.1.3 BAPTA 29kDa buffer as the endogenous buffer

As above, in Figure 2.13 reconstructions and simulation using BAPTA 29 kDa as endogenous buffer are compared. The initial conditions and parameters are reported in section 1.3.3.

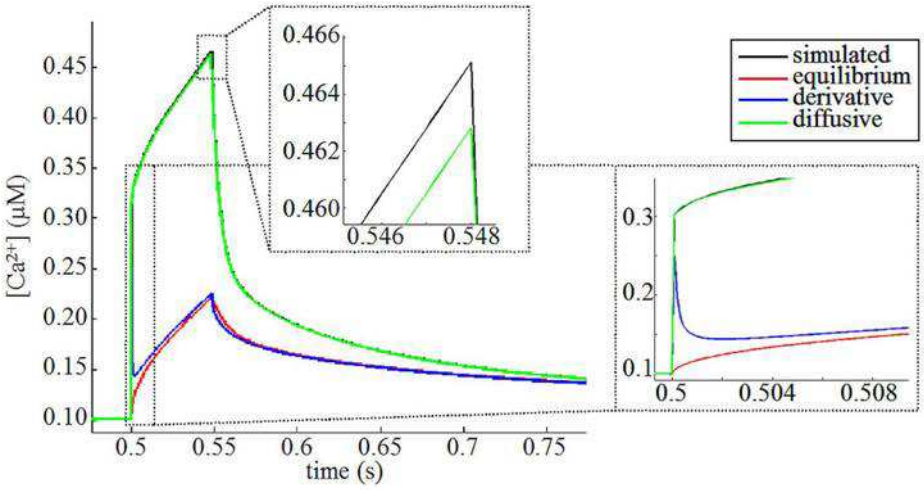

Figure 2.13: Comparison using BAPTA 29 kDa as endogenous buffer in voxel 50

Simulated $[Ca^{2+}]$ and reconstructed with formula (2.7) and simulated overlap well; the percentage error is about 0.4 %.

2.1.4 Calretinin as the endogenous buffer

The simulated $[Ca^{2+}]$ obtained with the initial conditions and parameters reported in section 1.3.4 is shown in Figure 2.14. In Figure 2.15 is shown the comparison between the reconstructions and the simulation.

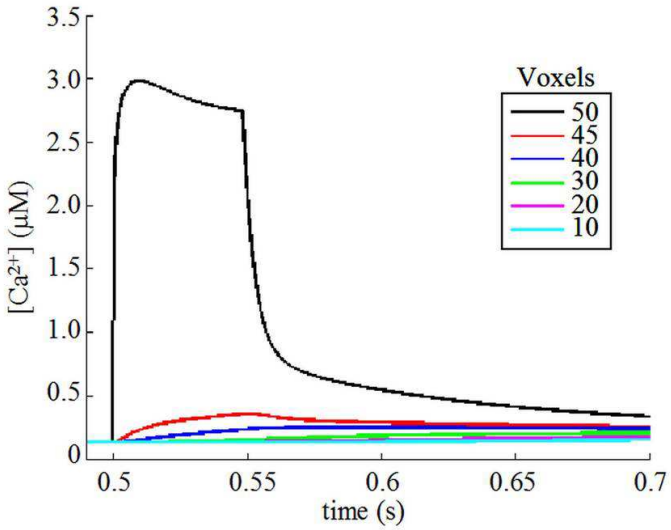

Figure 2.14: $[Ca^{2+}]$ simulated with Calretinin as endogenous buffer

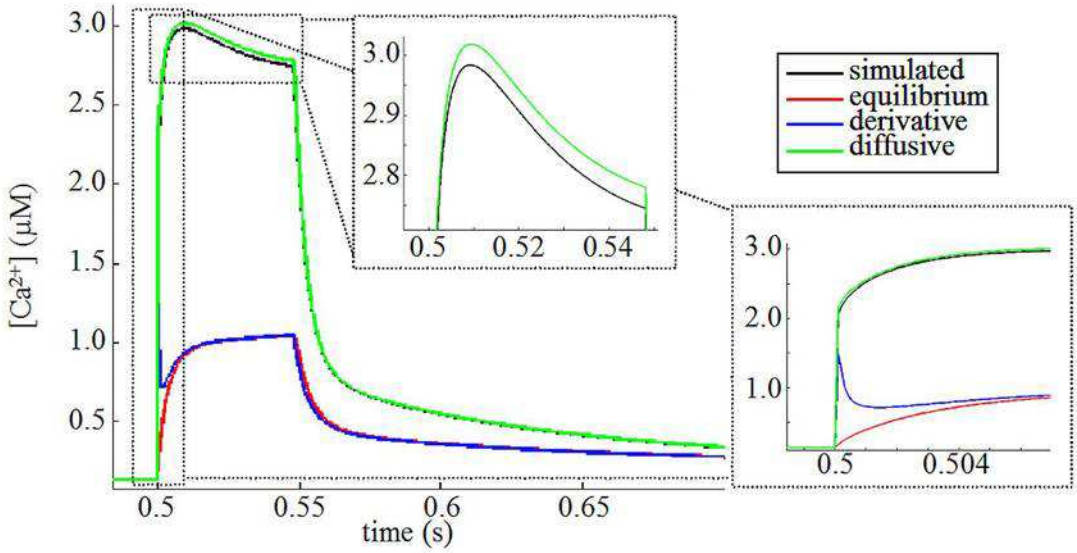

Figure 2.15: $[Ca^{2+}]$ comparison using Calretinin as endogenous buffer in voxel 50

The percentage error of the reconstruction comparing the top of $[Ca^{2+}]$ reconstructed and simulated is about 2 %

.

2.1.4.1 Reconstruction of the real calcium concentration by the new formula with different dyes

In Figure 2.16 are reported $[Ca^{2+}]$ reconstructed and simulated with these four different dyes. The initial conditions and the parameters are reported in section 1.3.5.

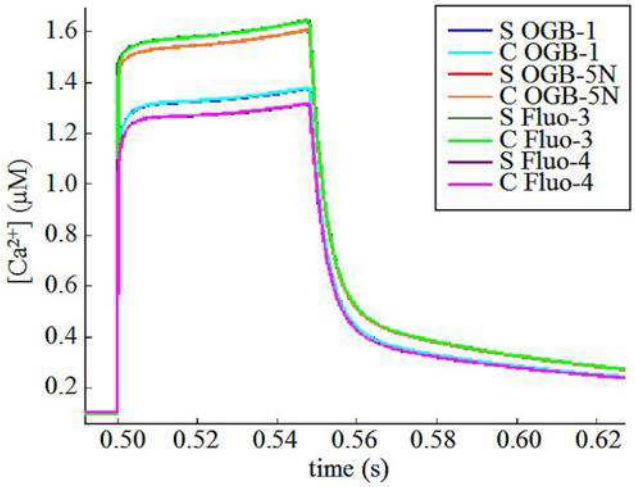

Figure 2.16: $[Ca^{2+}]$ simulated and reconstructed with four different fluorescent dyes

In Figure 2.17 are shown an enlargements on the tops of $[Ca^{2+}]$ concentrations of Figure 2.16.

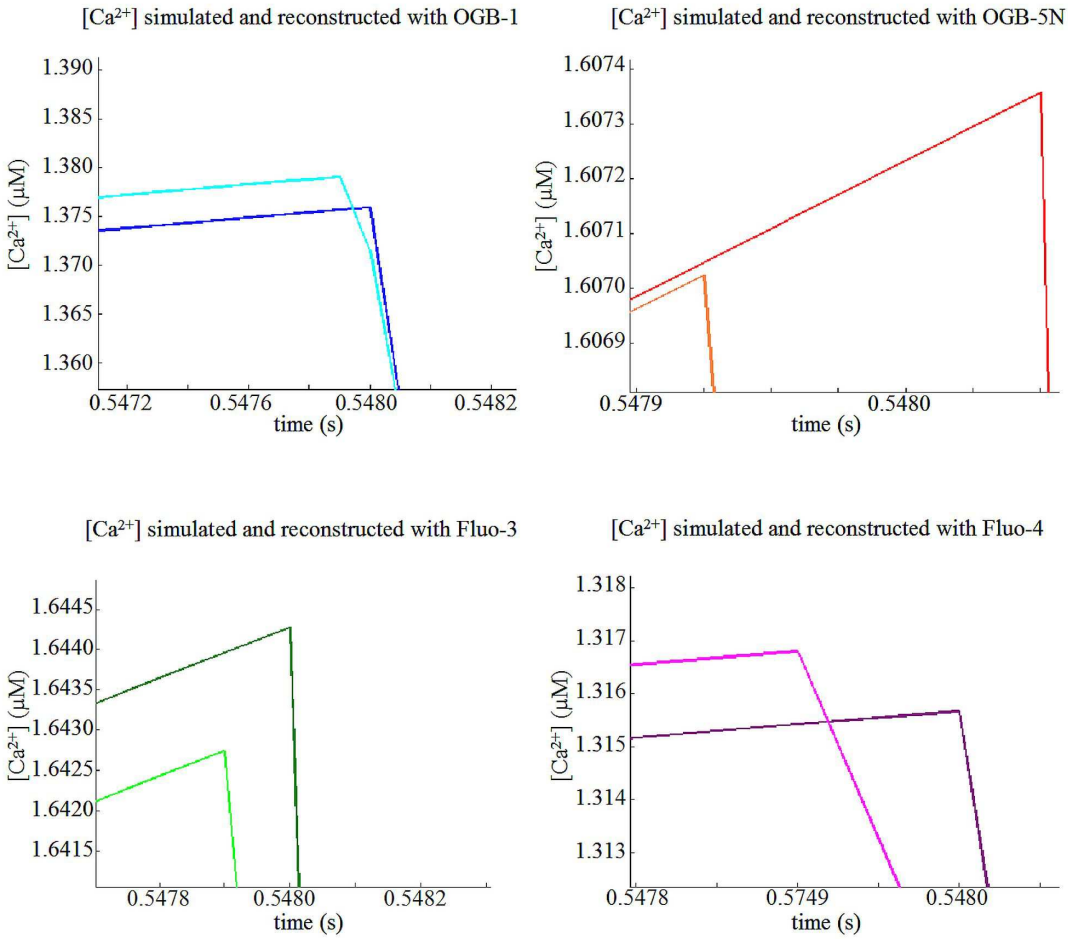

Figure 2.17: Enlargements on the tops of $[Ca^{2+}]$ curves of Figure 2.16

The reconstructions fit very well the simulations with all the four dyes, with a maximum percentage error of about 0.7 % for Oregon Green BAPTA-1.

2.1.4.2 Reconstruction of the real Calcium concentration by the new formula using OGB1 or OGB1-dextran in combination with NP-EGTA

The results of the simulation described in section 1.3.6 are shown in the figure below (Figure 2.18). In Figure 2.19 are reported the Calcium concentration reconstruction by the three formulas compared with simulated Calcium concentration.

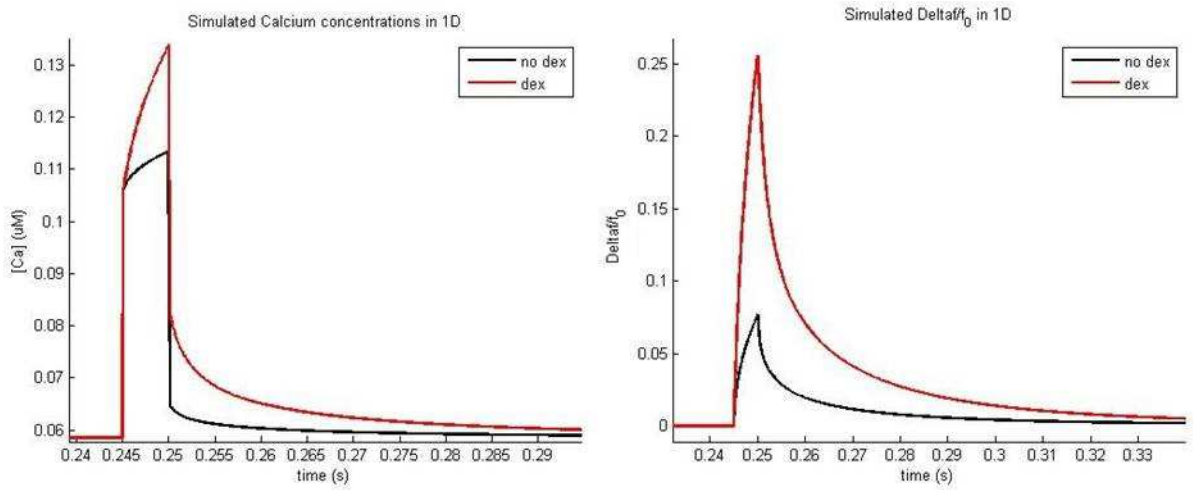

Figure 2.18: Simulated Calcium concentration (left) and $\Delta f / f_0$ (right) for 1-dimensional simulations with OGB-1 and OGB-1 dextran

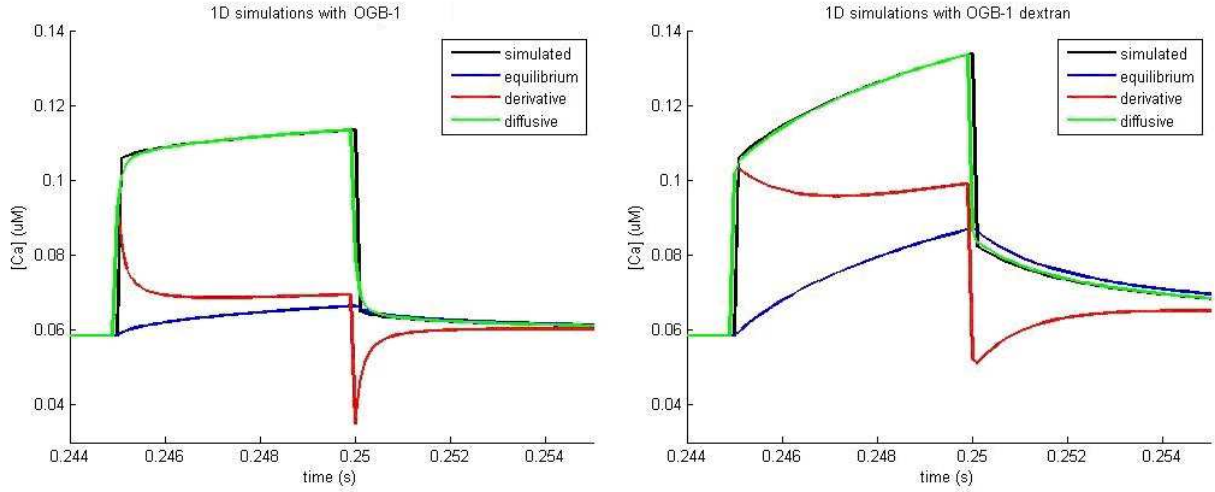

Figure 2.19: Comparison between reconstructed Calcium concentrations with the three formulas. 1-D simulations with OGB-1 (left), with OGB-1 dextran (right)

2.2 Three-dimensional simulations

2.2.1 Validation of diffusion process

A diffusion process is simulated with SimulCell software and outside by a self-made code and compared with the theoretical predictions using the initial conditions reported in Table 1.11 of section 1.5.1. In

Figure 2.20 is shown the diffusion process for five temporal steps. The points of the self-made code and SimulCell overlaps perfectly.

Secondly, the diffusion process is simulated using a bigger diffusive coefficient, $D = 2\,\mu m^2/s$. In this case, the concentration is not equal to zero in the boundary regions, so the boundary conditions are important. The code works well in this case, too. The total concentration remains the same for all the times but, obviously, the points do not overlap

with the theoretical prediction for an infinite homogeneous medium when the particles reach the boundary. This process is shown in

Figure 2.21.

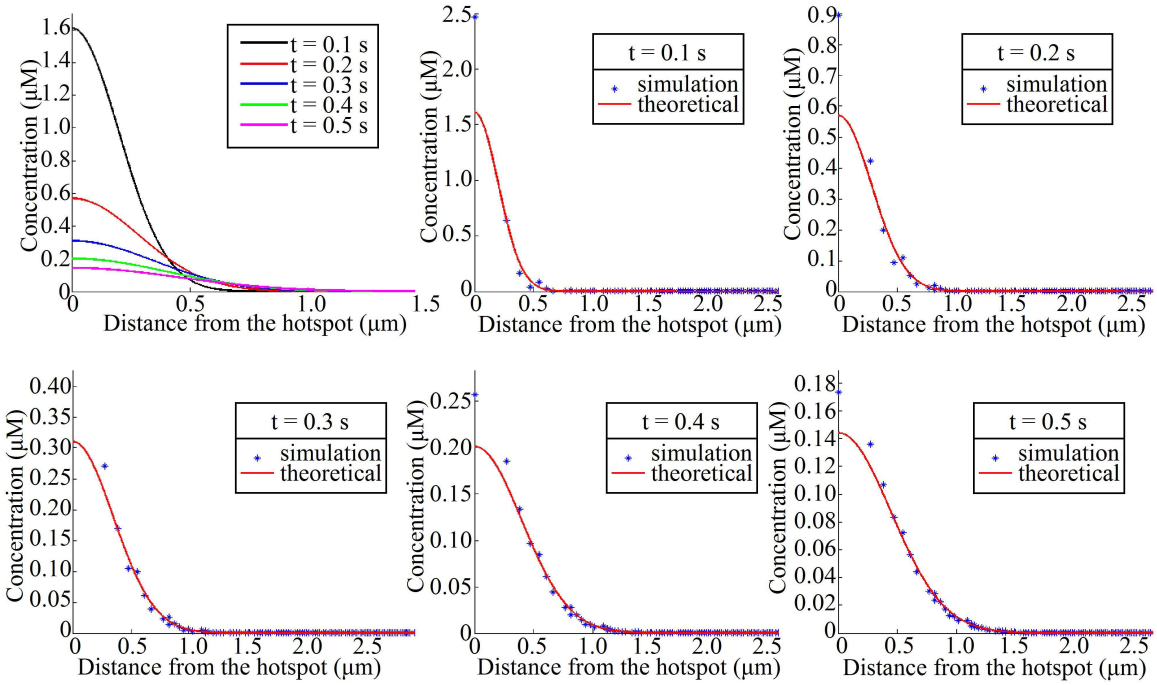

Figure 2.20: Simulation of a diffusion process in a spherical volume using parameters of Table 1.11. In the left top image is shown theoretical diffusion process expect at different time from the beginning of the diffusion. In the other images are shown the simulated points (blue) and the theoretical predictions (red line) at different time from the beginning of the diffusion process.

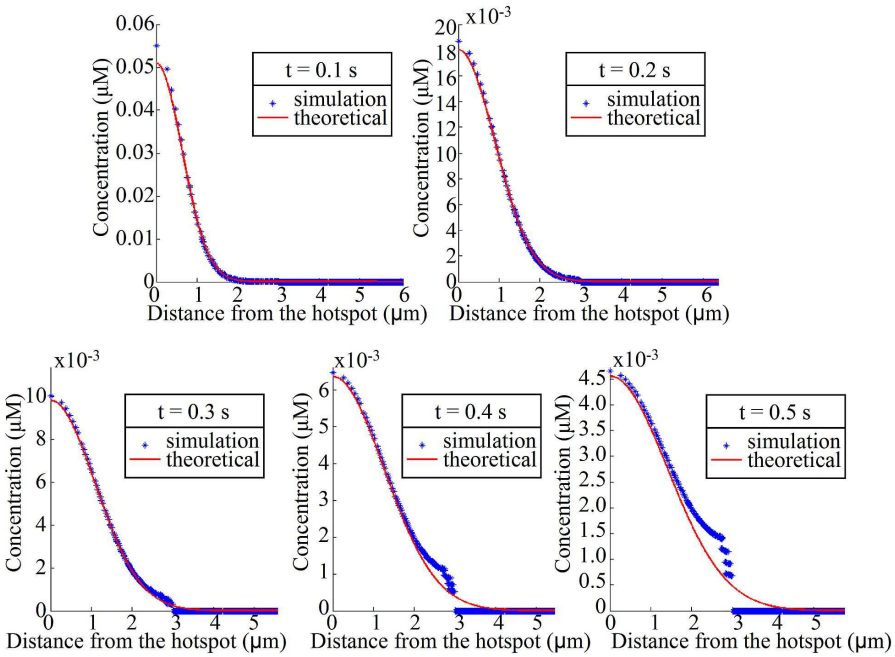

Figure 2.21: Simulation of a diffusion process in a spherical volume with $D = 2\ \mu m^2/s$. In the other images are shown the simulated points (blue) and the theoretical predictions (red line) at different time from the beginning of the diffusion process.

2.2.1.1 3D reconstruction of the real Calcium concentration by the new formula using OGB1 or OGB1-dextran in combination with NP-EGTA

These simulations are described in section 1.5.2. In Figure 2.22 are reported the simulated $\Delta f / f_0$ for the two simulations and in Figure 2.23 the reconstructed Calcium concentration reconstruction obtained by the three formulas compared with simulated Calcium concentration. In Figure 2.24 and Figure 2.25 are also reported the comparison between the simulated Calcium and the $\Delta f / f_0$ in 1D and 3D.

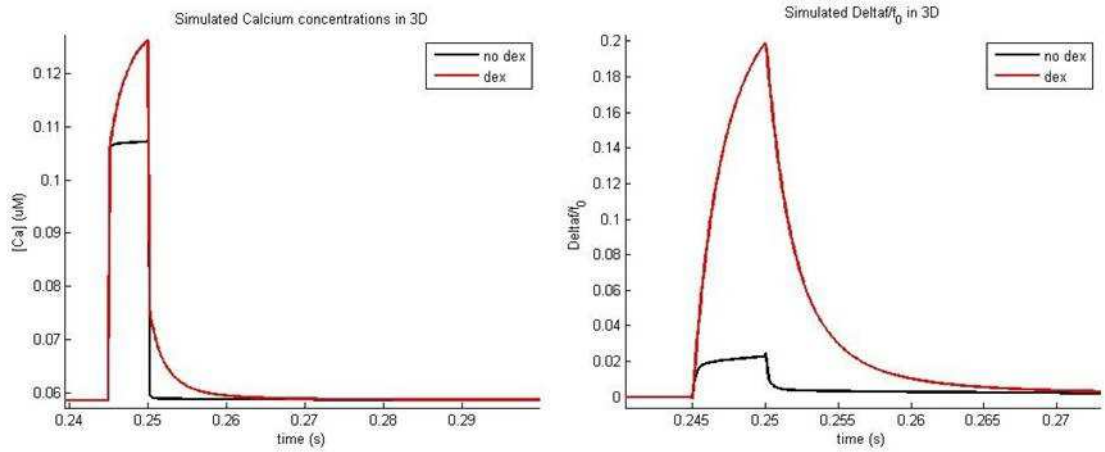

Figure 2.22: Simulated Calcium concentration (left) and $\Delta f / f_0$ (right) for 3-dimensional simulations with OGB-1 and OGB-1 dextran

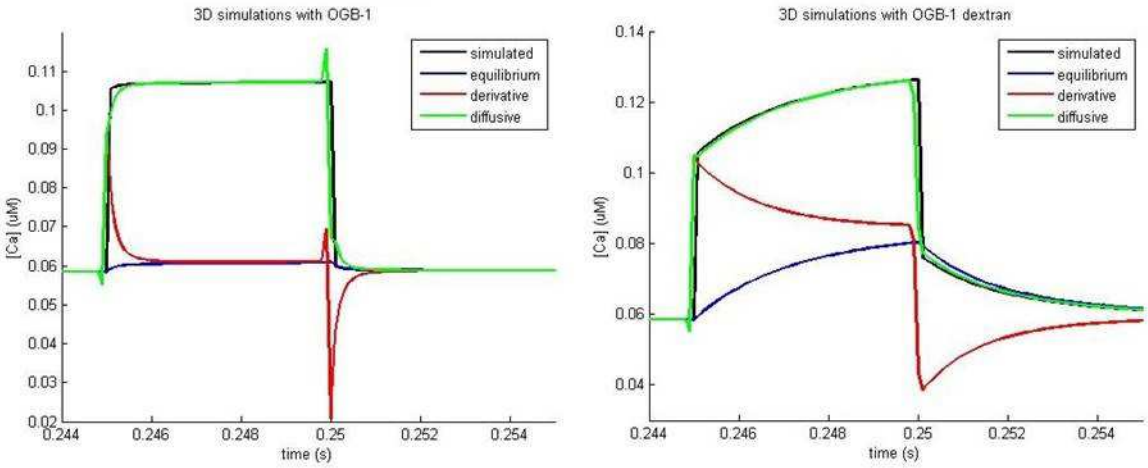

Figure 2.23: Comparison between reconstructed Calcium concentrations with the three formulas. 3-D simulations with simple OGB-1 (left), with OGB-1 dextran (right)

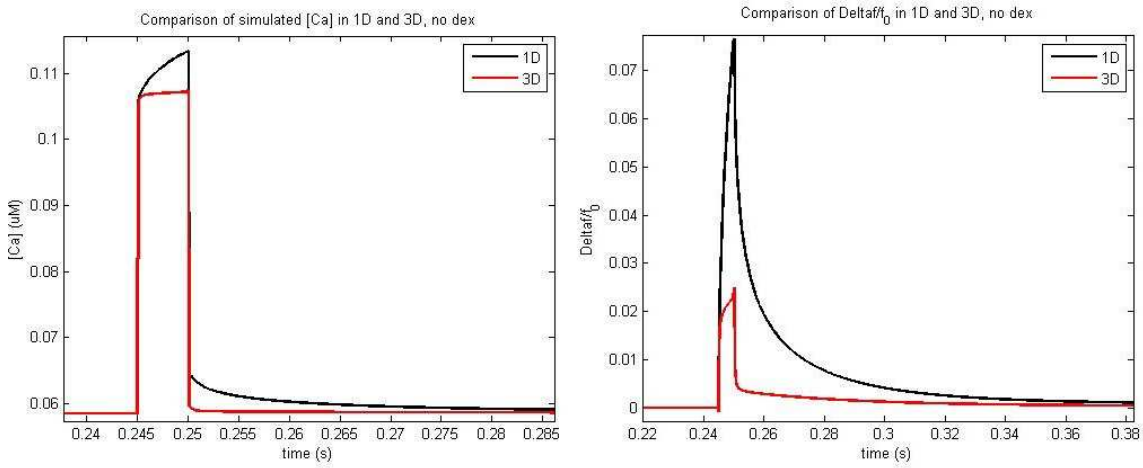

Figure 2.24: Comparison of simulated Calcium and $\Delta f / f_0$ for "no dex" simulations in 1D and 3D

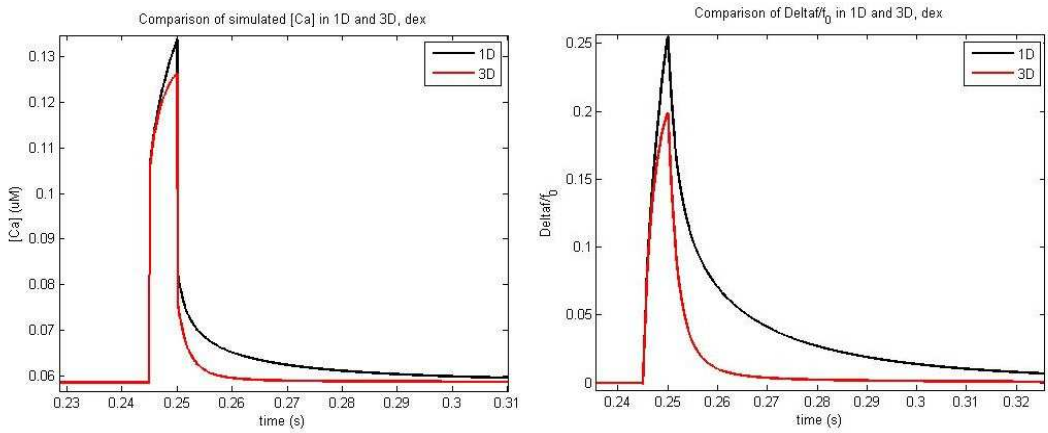

Figure 2.25: Comparison of simulated Calcium and $\Delta f / f_0$ for "dex" simulations in 1D and 3D

2.3 Experiments

2.3.1 First analysis

In this section we reported the results from the experiments performed with "sandwiches" samples.

The same procedure is followed for both the sample with the fluorescent dye dextran-conjugated and the dextran-free dye . We reported only the analysis performed on the sample with OGB-1 dextran step by step. Every image sequence recorded with the microscope was modified centering the laser spot on the center of the image. In this way, it is possible to process the images pixel by pixel. The traces for every sample are the mean values calculated inside a central region of interest (ROI) that surround the laser spot (ROI 1 of Figure 2.26). The ROI is the same for every sample.

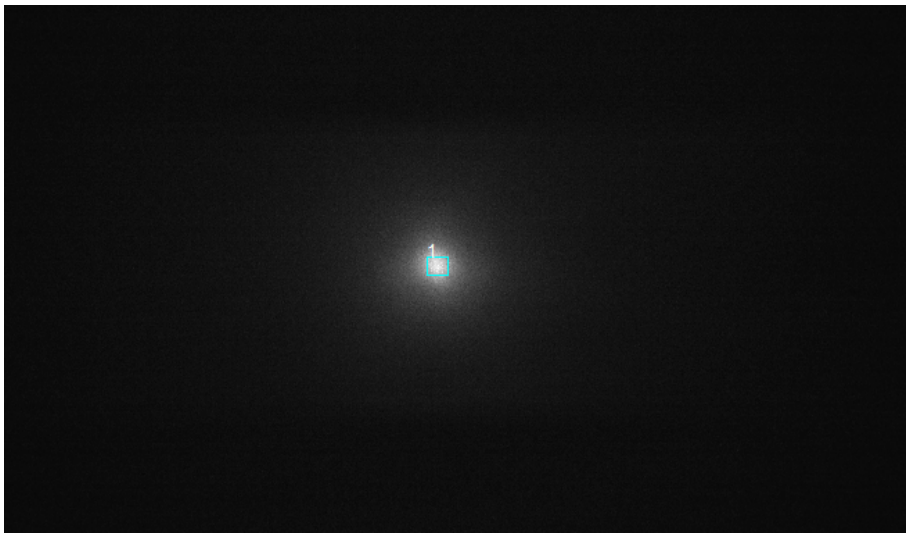

Figure 2.26: A ROI correspondent to the laser spot was used to analyze the fluorescence signal.

Firstly, the samples filled with intracellular solution were mediated.

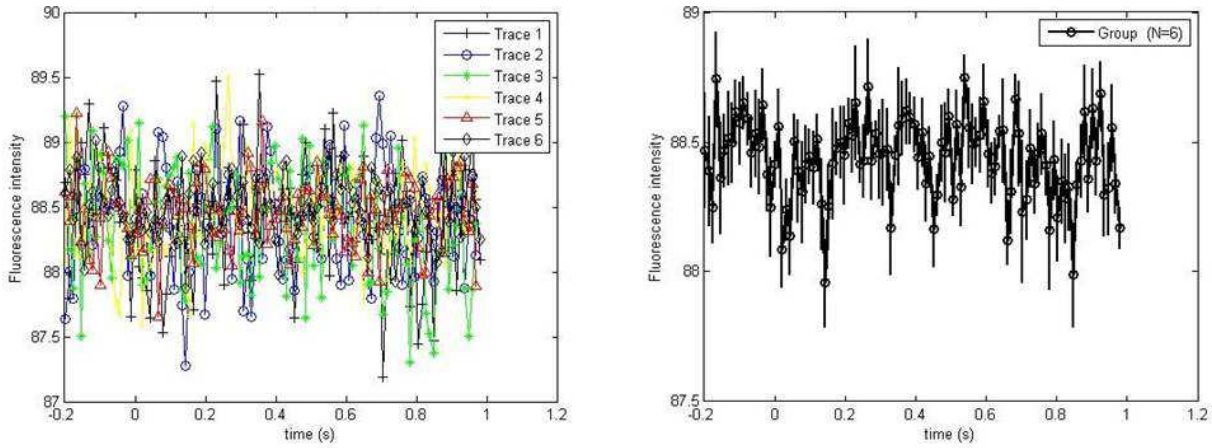

Figure 2.27: Fluorescence traces of intracellular (left) and mean (right)

These samples were used as background refererence and subtracted from all the traces of the other samples.
The OGBmax and OGBmin samples (after the background subtraction) were also mediated. These are used to estimate the $f_{\min}$ and $f_{\max}$ values of the fluorescent dyes. This estimation can be obtained mediating every frame of those two samples and then mediating the fluorescence intensity inside the ROI 1. The values are reported in table Table 2.1.

Table 2.1: $f_{\min}$, $f_{\max}$ and α of Oregon Green BAPTA-1 and Oregon Green BAPTA-1 dextran

	$f_{\min}$	$f_{\max}$	α
NP + OGB-1	29.62	270.34	9.12
NP + OGB-1 dextran	29.15	104.75	3.59

In Figure 2.28 we show the NP samples. From every sample the background was subtracted.

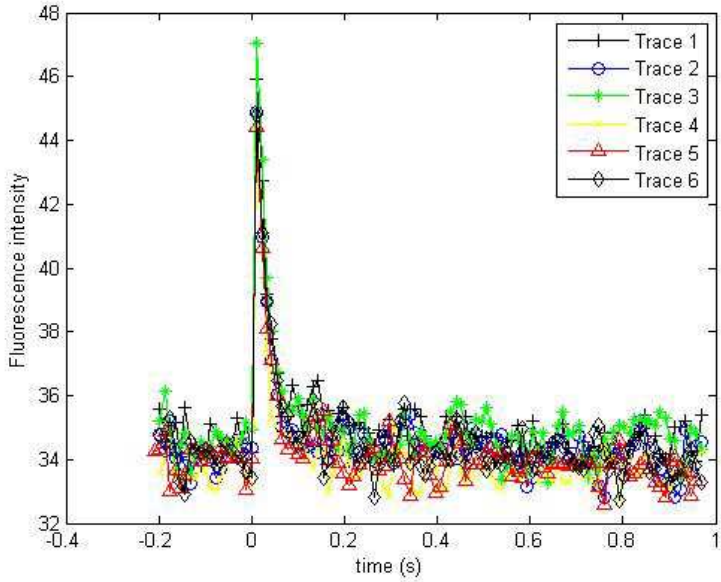

Figure 2.28: Fluorescence traces of NP samples

Then the $\Delta f / f_0$ was evaluated for every sample. f_0 is the mean of the fluorescence intensity before the Ca^{2+} influx. Finally, $\Delta f / f_0$ was mediated (Figure 2.29).

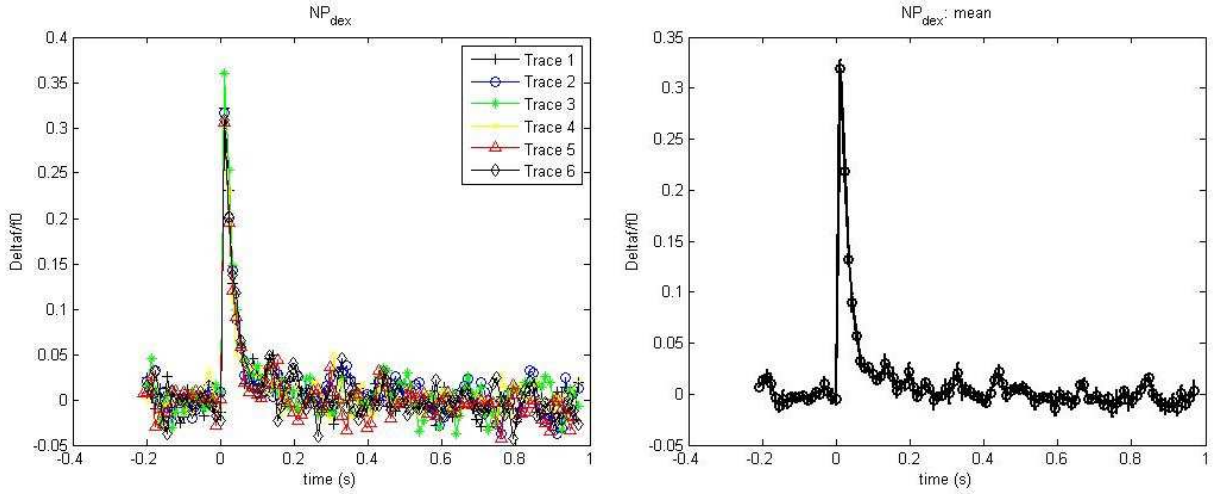

Figure 2.29: $\Delta f / f_0$ for samples with OGB-1 dextran (left), mean of $\Delta f / f_0$ (right)

The same procedure was followed for the sample with the dextran-free OGB-1 (Figure 2.30) and then the results were compared (Figure 2.31). In this case, we used only 3 of the 6 measurments, because one of the two sandwiches did show no comparable traces.

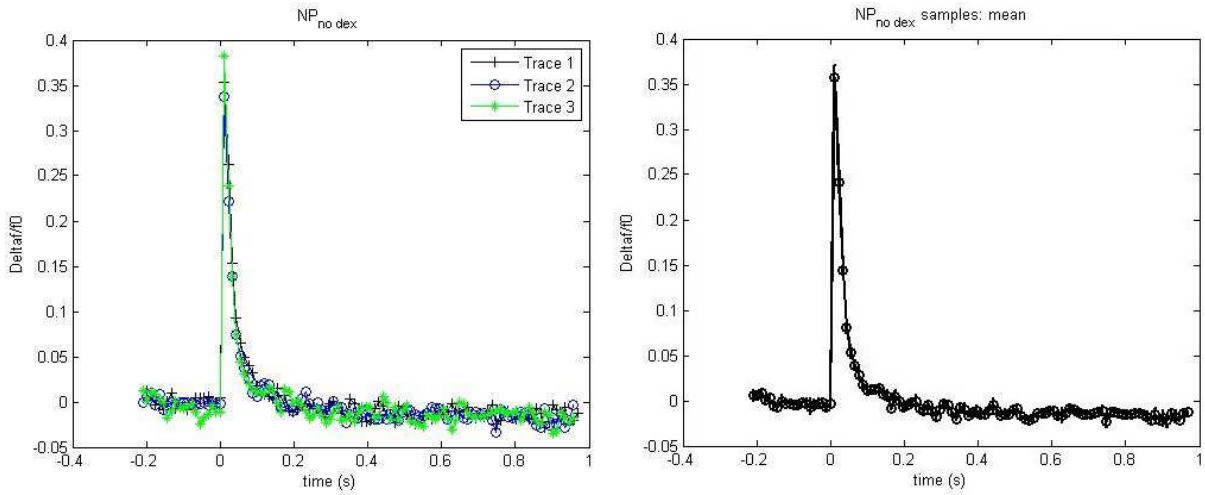

Figure 2.30: $\Delta f / f_0$ for samples with simple OGB-1 (left), mean of $\Delta f / f_0$ (right)

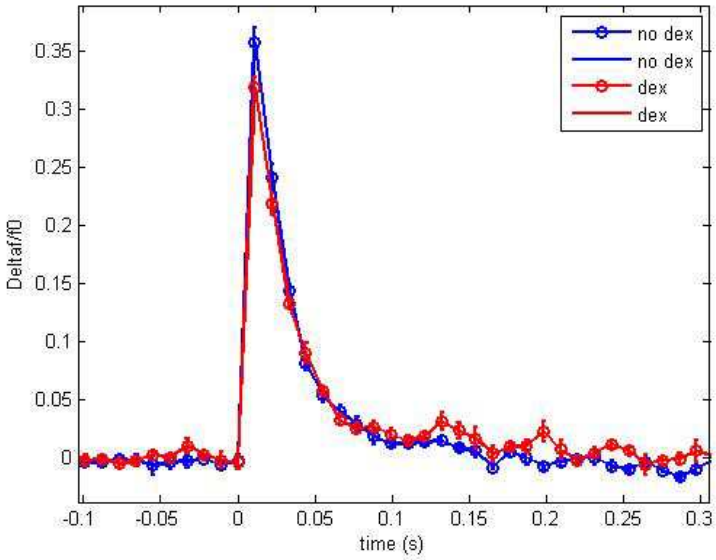

Figure 2.31: Comparison of the $\Delta f / f_0$ of the NP samples with OGB-1 and OGB-1 dextran

From these measurements was possible to evaluate the basal concentration of Ca^{2+} using formula (1.14) (Table 2.2).

Table 2.2: $[Ca^{2+}]_0$ and γ of the NP_{dex} and $NP_{no\ dex}$ samples

	$[Ca^{2+}]_0$	γ
$NP_{no\ dex}$	58 nM	11.09
NP_{dex}	15 nM	9.97

Knowing these two parameters, α, D_F, k_{on} and k_d of the fluorescent dye, was possible to reconstruct the Calcium concentration with the equilibrium formula (1.30). In Figure 2.30 is also reported the comparion between these two Calcium reconstructions.

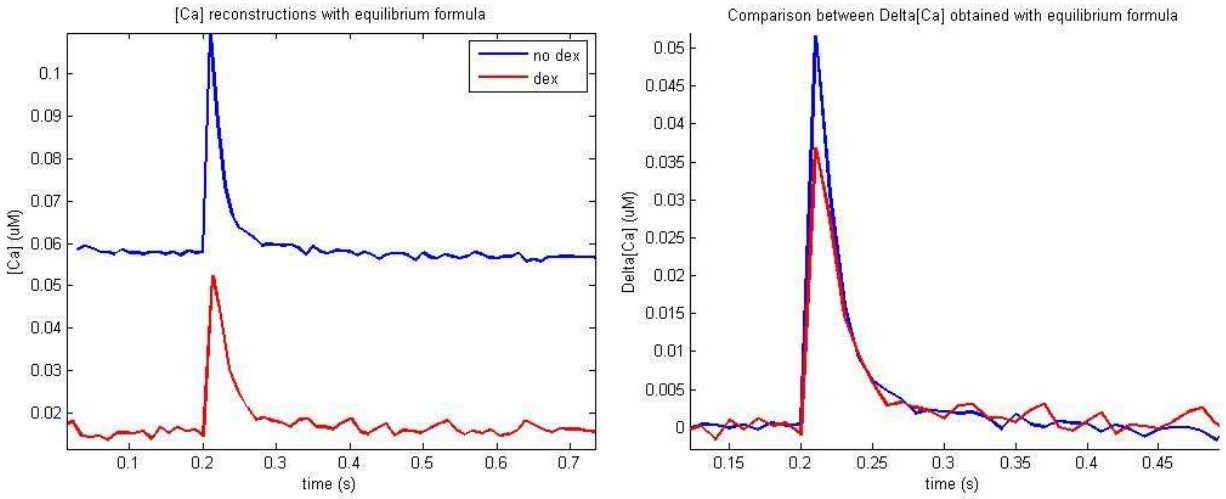

Figure 2.32: Reconstructed Calcium concentration with equilibrium formula (left) and comparison between $\Delta[Ca^{2+}]$

In order to use the novel formula (2.7) is necessary to know the laplacian $\nabla^2(\Delta f / f_0)$. This variable could be obtained by the analysis of the image sequences. The same procedure followed for the traces could be performed pixel by pixel also for the images, obtaining comparable results. We used a confocal microscope, so the enlighted and recorded area of the sample was supposed to be a plane. Therefore, the laplacian is 2D and could be obtained by the following discretization. The vertical terms could be neglected because we supposed they have the same value of the central one.

$$\nabla^2\left(\frac{\Delta f}{f_0}\right) = \frac{\frac{\Delta f(x+\Delta x, y)}{f_0} + \frac{\Delta f(x-\Delta x, y)}{f_0} + \frac{\Delta f(x, y-\Delta y)}{f_0} + \frac{\Delta f(x, y+\Delta y)}{f_0} - 4\frac{\Delta f(x, y)}{f_0}}{\Delta x^2} \tag{2.9}$$

In this way, the laplacian could be determined pixel by pixel. The mean of the laplacian inside the ROI number 1 is reported in Figure 2.33.

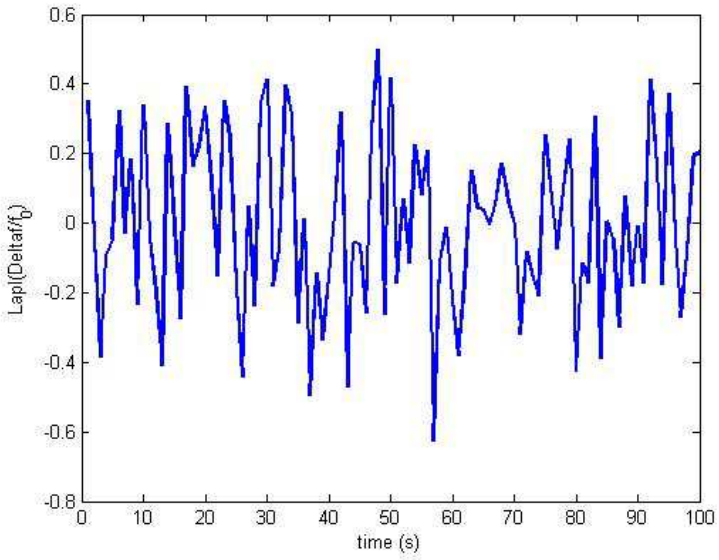

Figure 2.33: $\nabla^2(\Delta f / f_0)$ obtained pixel by pixel for NP_{dex} samples

This laplacian term is very noisy and it could not be used for the reconstruction. We tried to obtain a better laplacian term applying to the $\Delta f / f_0$ sequences some filters (average and Gaussian) and increasing the binning of the images but the results were still too noisy and unusable. Therefore, we built a "ROI laplacian" in the following way: we considered the central ROI as a big pixel of value $\Delta f / f_0$ (mean) . His neighbours were the other adjoined ROIs of the same size. Thus, using (2.9) we obtained the laplacian reporte in Figure 2.34.

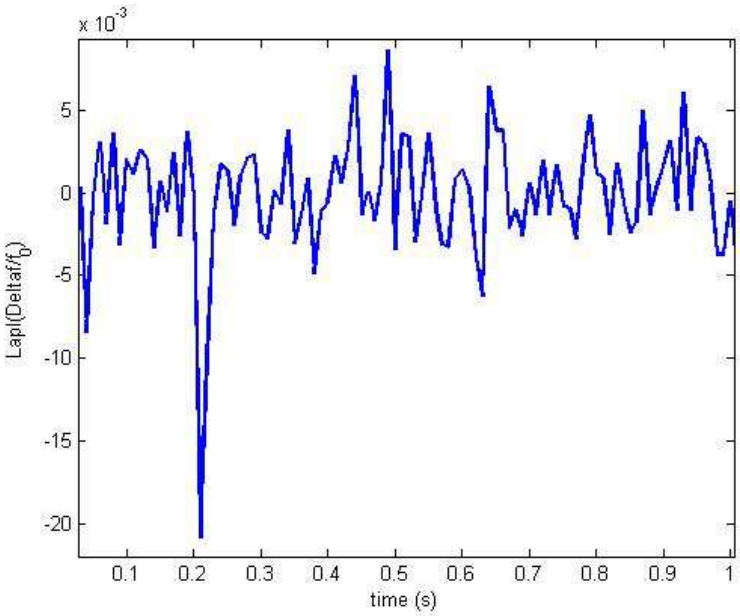

Figure 2.34: "roi laplacian" for NP_{dex} samples

The laplacian now could be inserted into the formula (1.28), obtaining

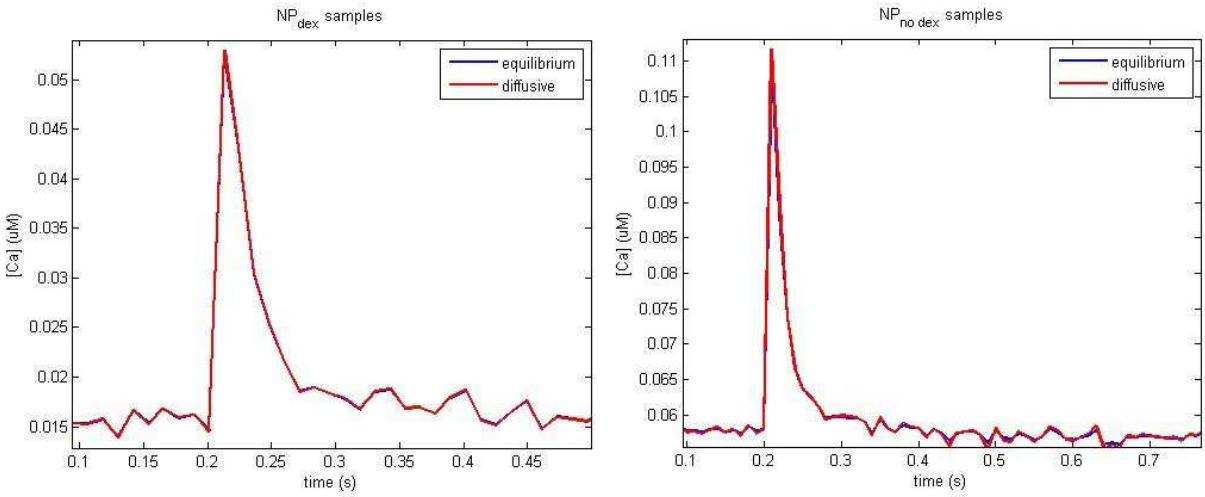

Figure 2.35: Comparison between $[Ca^{2+}]$ reconstructions with equilibrium and diffusive formulas for NP_{dex} samples (left) and $NP_{no\ dex}$ samples (right)

It seemed that the diffusive contribution did not improve the reconstruction, so we calculated the mean fluorescence intensity for concentric ROIs. In this way, we could understand if the diffusion was significant for our image sequences.

In Figure 2.36 we reported the mean fluorescence intensity as a function of the internal radius (in pixel) of the concentric ROI for $NP_{no\ dex}$ samples. The numbers in the legend correspond to the number of frames after the laser impulse.

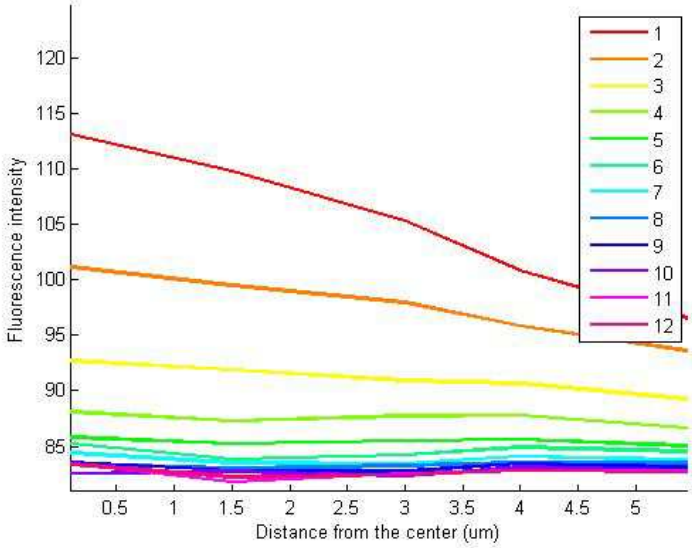

Figure 2.36: Mean fluorescence intensity recorded in the frames after the laser impulse in function of the radius of concentric rois

2.3.2 Comparison between experiments and simulation

Once the simulation of section 1.5.3 was finished we have tried to compare its results with the experiments for the $NP_{no\ dex}$ sample, because it was the one that had the basal Calcium concentration similar to what we expected. For the simulation, we chose the parameter f with a value of 250. Unfortunately, this value did not fit with the experimental data, but the simulation could be used anyway to better understand the data's meaning. In Figure 2.37 are shown the simulated Calcium concentration and the Calcium reconstructed through the equilibrium and diffusive formula. The simulation was meadiated in order to have the same temporal step of the experiments and nearly the same region of interest.

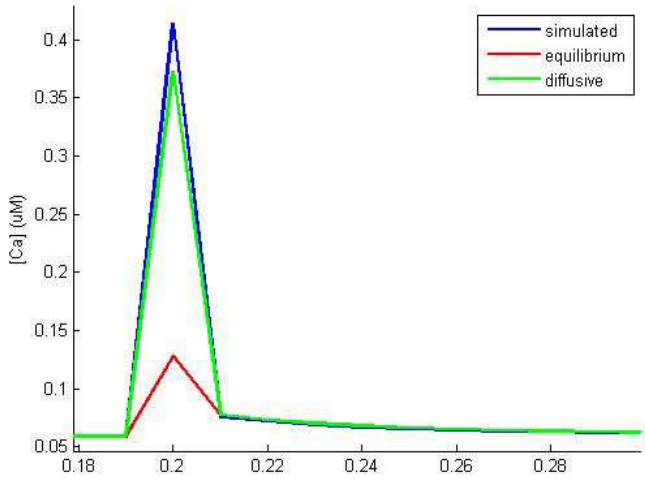

Figure 2.37: Comparison between simulated $[Ca^{2+}]$ and reconstructed $[Ca^{2+}]$ by equilibrium and diffusive formula for the simulation of section 1.5.3

For the experiments, the frame in which the laser was activated was deleted, thus the same was done for this simulations, obtaining the Figure 2.38. The equilibrium traces overlaped perfectly with the diffusive traces.

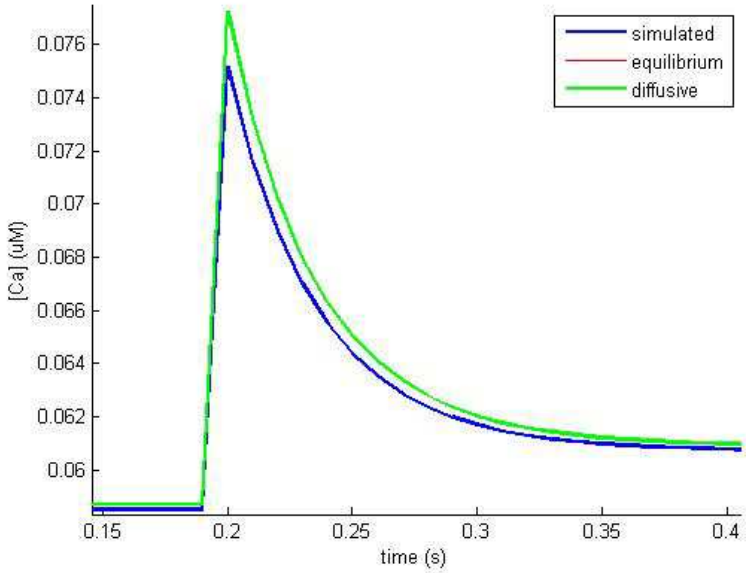

Figure 2.38: Comparison between simulated $[Ca^{2+}]$ and reconstructed $[Ca^{2+}]$ by equilibrium and diffusive formula for the simulation of section 1.5.3 without the laser frame

2.3.3 Second analysis

It seemed that the equilibrium conditions were reached since the first frame after the laser flash. For this reason, we tried to analyze the image sequence without deleting the laser frame. We had to consider the OGB samples (Table

1.17): these samples were used to modulate the fluorescent signal of the NP_{dex} and $NP_{no\ dex}$ samples. In fact, OGB-1 and OGB1-dextran were undesirably excited by the UV-laser, so the fluorescence signals recorded from the NP samples were a convolution of the OGB emission due to the laser and the emission due to the calcium binding and unbinding. As before, the image sequences used were a mean between the same type of samples without the background. A modulation process was developed in order to remove the former fluorescence contribute. Firstly, the fluorescent signals recorded for the OGB samples were normalized, dividing them for their basal fluorescence (2.10).

$$OGB_{norm} = \frac{OGB}{OGB_0} \tag{2.10}$$

where OGB is the image matrix with the fluorescence intensities as entries and OGB_0 is the one with the basal fluorescence intensities. Then, a fictional f_0 for NP samples, called F_0, was evaluated as follows

$$F_0 = NP_0 \cdot OGB_{norm} \tag{2.11}$$

where NP_0 is the image matrix with the basal fluorescent intensities of the NP samples as entries. $\frac{\Delta f}{f_0}$ is estimated as

$$\frac{\Delta f}{f_0} = \frac{NP - F_0}{F_0} \tag{2.12}$$

In Figure 2.39 we reported the NP, OGB and F_0 traces obtained mediating the fluorescent values inside the ROI 1 (see section 2.3.1) for samples filled with simple OGB-1 and OGB-1 dextran.

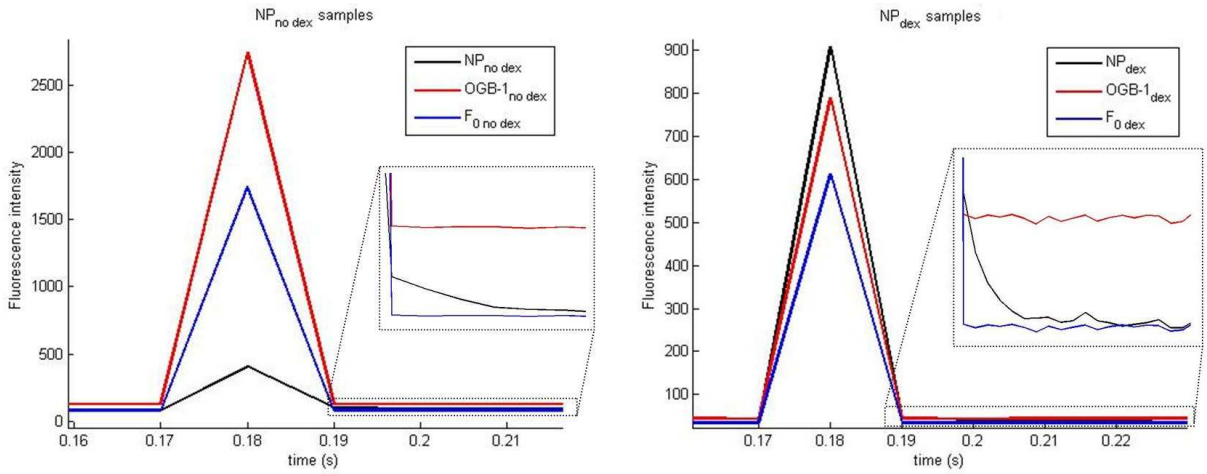

Figure 2.39: NP, OGB and F_0 traces of samples with OGB-1 (left) and OGB-1 dextran (right)

In Figure 2.38 we show the $\Delta f / f_0$ obtained with (2.12).

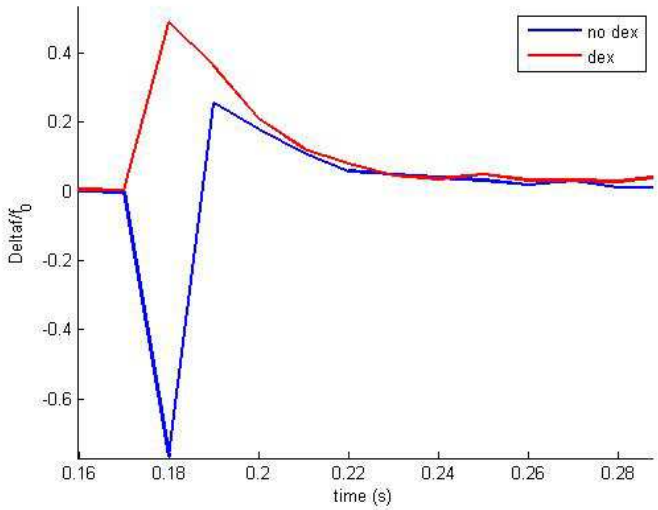

Figure 2.40: $\Delta f / f_0$ obtained with (2.12)

In Figure 2.41 are reported the Ca^{2+} concentration reconstructions obtained by the same formulas used in section 2.3.1.

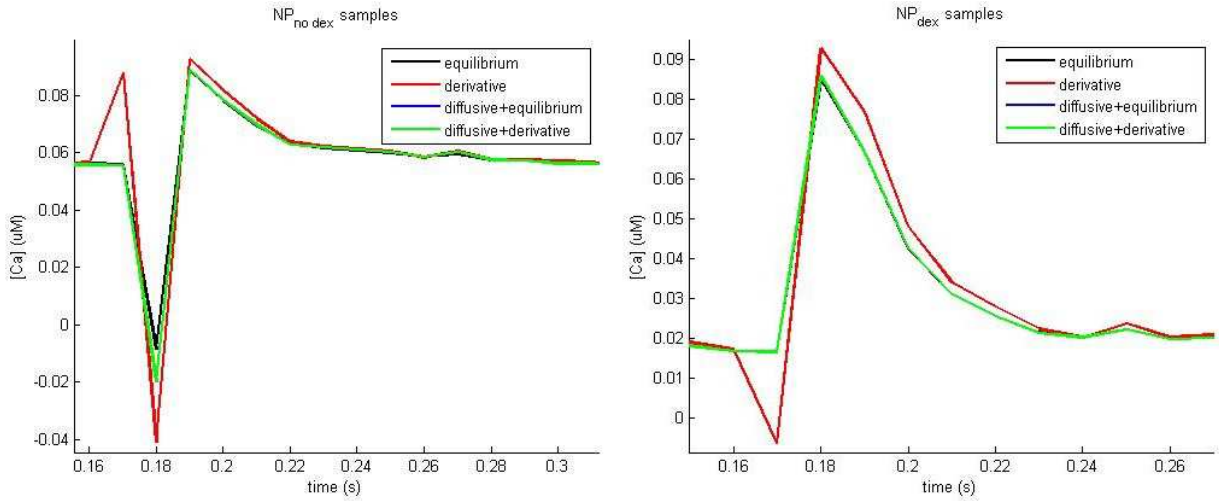

Figure 2.41: Calcium concentration reconstructions for $NP_{no\ dex}$ samples (left) and NP_{dex} samples (right)

3 Discussion

In this thesis work, we have developed a novel formula for the reconstruction of the intracellular Ca^{2+} concentration, based on knowledge of the dye fluorescence signal and the related experimental parameters (k_{on}^{F}, k_{off}^{F}, k_{d}^{F}, α and D_F), as well as on the resting Ca^{2+} concentration estimated before the Ca^{2+} influx. Considering a formula that assumes the equilibrium of the reactants would lead to uncorrect reconstruction of the real Ca^{2+} concentration when the calcium influx is high and localized, e.g. in the case of a Ca^{2+} hotspot [6]. To overcome this limitation, our novel formula takes into account both the temporal and spatial dynamics of the dye by calculating the temporal derivative of the fluorescence signal ("derivative" formula (1.28)) as well as its laplacian ("diffusive" formula (2.7)). This formula was initially tested and validated by numerical simulations in a 1-dimensional geometry. We tested the validity of our formula changing several critical parameters, such as type of dye, buffer or the Ca^{2+} influx dynamics (section 2.1). In each tested condition, the equilibrium formula failed to reconstruct the real Ca^{2+} concentration during the Ca^{2+} influx, whereas the "derivative" formula (1.28) improved the estimation, in particular at the onset of the Ca^{2+} influx. Small artifacts (e.g., spikes) were observed due to discretization in the first derivative of the fluorescent signal (Figure 2.19, Figure 2.23). Using the "diffusive" formula significantly improved the reconstruction of the real Ca^{2+} concentration (section 2.1.1.2) in respect to the "derivative" formula alone, permitting to obtain nearly exact estimations in every condition tested (Figure 2.10, Figure 2.13, Figure 2.15, Figure 2.16, Figure 2.19). The maximum error in the Ca^{2+} peak estimation was around 1-2%, which is a significative improvement in respect to the equilibrium formula giving errors around 70-80%.

The second part of our theoretical work focused on testing the new formula in three dimensional environment by upgrading the 1-dimensional user-friendly simulation software developed in Prof. Bortolozzi's lab that we used for the first validation. A new simulation software, named SimulCell (section 1.4.1), included the tool to design objects in 3D and to set the initial conditions. SimulCell was designed to allow the user to create objects with arbitrary shape, starting from elementary geometrical solids. In the geometry (1.4.2) and initial conditions (1.4.3) sections the user can define "patterns", i.e. regions of the simulation space, with different properties that can also be used to analyze the results of the simulation.

Once developed the SimulCell software, we performed 3-D simulations (section 2.2). After validating the diffusion algorithm (Fick's law equation, section 2.2.1, Figure 2.20), we performed same simulations performed in 1-D (section 2.2.1.1), obtaining the same succesfull results with the novel "diffusive" formula (Figure 2.23). As we expected, we found that, after the offset of the Ca^{2+} influx, the $\Delta f / f_0$ approaches the equilibrium condition faster than in the 1-D case, due to the higher number of degree of freedom of the dye (Figure 2.24 and Figure 2.25).

Simulating the same dye with different diffusive coefficient (OGB-1 and OGB-1 dextran, section 2.2.1.1) suggested us that the same Ca^{2+} peak (0.11 μM), is predicted as different if we use the equilibrium formula obtaining 0.066 μM for OGB-1 and 0.081 μM for OGB-1 dextran (Figure 2.19 and Figure 2.23), instead of the "diffusive" formula that provided 0.11 μM as the real Ca^{2+}.

Based on these considerations, we performed Ca^{2+} imaging experiments combined with UV-flash photolysis technique in an artificial couvette that mimics the intracellular environment under controlled conditions. Images were acquired by using a confocal spinning disk developed in Prof. Bortolozzi's lab. We set up the experimental conditions in order to have a significative photorelease of Ca^{2+} by NP-EGTA, while acquiring images at the high frame rate (100

Hz). We tested also reliability of the Ca^{2+} stimulus by repeating the experiments in different couvettes as well as different regions of the couvette itself (section 2.3.1). Unexpectedly, we found a different α values for normal OGB-1 and OGB-1 dextran (Table 2.1), maybe due to quancing effects in the dextran dye due to the higher molecular wheight. The recorded $\Delta f / f_0$ values were similar for both OGB-1 and OGB-1 dextran experiments when analyzed after the Ca^{2+} photorelease (Figure 2.31), suggesting that the amount of Ca^{2+} released by the NP-EGTA is similar but the system is already in equilibrium conditions (Figure 2.32), as confirmed by application of the "diffusive" formula that does not provide significative differences.

We found technical problems in the estimation of the Laplacian computed pixel by pixel due to the relatively low signal-to-noise ratio (Figure 2.33), despite application of spatial and temporal filters to the experimental movie. Analysis of the laplacian was then performed by a binning approximation that we called "ROI laplacian" (Figure 2.34), avoiding the frames of Ca^{2+} photoliberation due the artefact of dye excitation by the laser. The "diffusive" formula provided results similar to the equilibrium formula when considering the frames after the Ca^{2+} photoliberation. This result was interpreted by the simulation (Figure 2.37) that indicates that Ca^{2+} is not at equilibrium with its reactants only during its liberation, i.e. the laser duration in the experiment (Figure 2.38). As we suspected, the "diffusive" formula is effective only during the Ca^{2+} influx, thus we tried to consider also the laser frame by performing a "modulation process" explained in section 2.3.3. This process did not help us to obtain good $\Delta f / f_0$ traces and also creates unwanted artifacts (for example negative pick in $\Delta f / f_0$ traces, Figure 2.40), with unsatisfactory results, expecially for the OGB-1 experiments.

4 Conclusions

In this thesis work, we developed and tested a novel formula ("diffusive" formula) useful to estimate the real Ca^{2+} concentration in non-equilibrium conditions of Ca^{2+} and its reactants, e.g. during a Ca^{2+} influx at the neuronal synapse. The fluorescent signal of the Ca^{2+} dye and a few experimental parameters are required to derive the real Ca^{2+} concentration by the formula. The formula was validated by both simulations in a 1-D and 3-D environment by a user-friendly software upgraded during the thesis. Knowledge of the resting Ca^{2+} concentration is required a priori, thus representing a limitation in the precision of the Ca^{2+} estimate as this parameter can be experimentally estimated only at the end of the experiment. A future upgrade of the formula will be the derivation of a parameter equivalent to the $\Delta f / f_0$, thus predicting Ca^{2+} variations independent from the knowledge of the resting Ca^{2+} .

Experimental validation of the formula will be also performed in a future work, overcoming artifacts found in our experiments, by (i) combining imaging with dual patch-clamp technique in neurons to apply voltage transients that control the opening and closure of voltage-activated Ca^{2+} channels and (ii) improving the off-line analysis of the Laplacian by advanced image filters.

5 Appendix

5.1 SimulCell

5.1.1 Geometry

```
%-- Generate the faces and vertices of the object surface.
dataF.Objects.box1.FV = isosurface(dataF.X,dataF.Y,dataF.Z,dataF.Objects.box1.V,0.95);
%-- DataF.X, DataF.Y, DataF.Z are the coordinates value attributed to every point of the volumetric
image. 0.95 is the isovalue: the isosurface connects points that have the specified value much the
way contour lines connect points of equal elevation

%-- Draw the object
dataF.Objects.box1.p = patch(dataF.Objects.box1.FV);

%-- Reshape the object representation
isonormals(dataF.X,dataF.Y,dataF.Z, dataF.Objects.box1.V, dataF.Objects.box1.p);

%-- Choose the colour and the appearance of the object
    set(dataF.Objects.box1.p, 'FaceColor', 'blue', 'FaceAlpha', 0.3, 'EdgeColor', 'none');
    daspect([1 1 1])
    view(3)
    camlight(-45,45);
    lighting phong
```

5.1.2 Create Objects

```
%-- dataF.Lx and the others are the dimension of the box of zeros. For example, dataF.Lx = box.x
where box.x is the x dimension that is wrote in the box window
dataF.V_tot = dataF.Lx*dataF.Ly*dataF.Lz;

%-- This is the smaller distance between one point with his neighbour
dataF.resolution = nthroot(dataF.V_tot/dataF.N_tot, 3);

%-- These are the number of division for every axes of the 3D-matrix
dataF.Nx = round(dataF.Lx/dataF.resolution);
dataF.Ny = round(dataF.Ly/dataF.resolution);
dataF.Nz = round(dataF.Lz/dataF.resolution);

%-- These are the value attribute to every point of the 3D-matrix
dataF.valuesx = -dataF.Lx:(2*dataF.Lx./(dataF.Nx-1)):dataF.Lx;
dataF.valuesy = -dataF.Ly:(2*dataF.Ly./(dataF.Ny-1)):dataF.Ly;
dataF.valuesz = -dataF.Lz:(2*dataF.Lz./(dataF.Nz-1)):dataF.Lz;
[dataF.X,dataF.Y,dataF.Z] = meshgrid(dataF.valuesy,dataF.valuesx,dataF.valuesz);
```

5.1.3 Translate

```
function V_translated = translation_xp(dataF.Objects.example.V, dataF.X, dataF.Y, dataF.Z,
dataF.resolution)

gap = abs(dataF.X(1,1,1) - dataF.X(1,2,1));
space = round(dataF.resolution/gap);
N = size(V);

for x=(space+1):N(1)
        for y=1:N(2)
            for z=1:N(3)
                V_xtp(x, y, z) = V(x-space,y,z);
            end
        end
end

V_xtp(1:space, :,:) = 0;
V_translated = V_xtp;

end


function V_translated = translation_xm(dataF.Objects.example.V, dataF.X, dataF.Y, dataF.Z,
dataF.resolution)

gap = abs(dataF.X(1,1,1) - dataF.X(1,2,1));
space = round(dataF.resolution/gap);
N = size(V);

for x=(N(1)-space):-1:1
    for y=N(2):-1:1
        for z=N(3):-1:1
            V_xtm(x, y, z) = V(x+space,y,z);
        end
    end
end

V_xtm((N(1)-space+1):N(1),:,:) = 0;
V_translated = V_xtm;

end
```

5.1.4 Rotate

```
function V_rotated = rotate(dataF.Objects.Example.V, dataF.anglex, dataF.angley, dataF.anglez)
V = dataF.Objects.Example.V
N = size(V);
```

```
%%% This is the code used to rehsape the the volumetric image in a cube
%%% with even number of discretizations
if mod(N(1),2)==0
else
    V = cat(1, zeros(1, N(2), N(3)), V);
    N = size(V);
end
if mod(N(2),2)==0
else
    V = cat(2, zeros(N(1), 1, N(3)), V);
    N = size(V);
end
if mod(N(3),2)==0
else
    V = cat(3, zeros(N(1), N(2), 3), V);
    N = size(V);
end
maxo = max(N);
x_add = maxo - N(1);
y_add = maxo - N(2);
z_add = maxo - N(3);
if x_add == 0 & y_add ~= 0 & z_add ~= 0
   V = cat(2, zeros(N(1), round(y_add/2), N(3)), V);
   V = cat(2, V, zeros(N(1), round(y_add/2), N(3)));
   N = size(V);
   V = cat(3, zeros(N(1), N(2), round(z_add/2)), V);
   V = cat(3, zeros(N(1), N(2), round(z_add/2)), V);
elseif y_add == 0 & x_add ~= 0 & z_add ~= 0
   V = cat(1, zeros(round(x_add/2), N(2), N(3)), V);
   V = cat(1, V, zeros(round(x_add/2), N(2), N(3)));
   N = size(V);
   V = cat(3, zeros(N(1), N(2), round(z_add/2)), V);
   V = cat(3, zeros(N(1), N(2), round(z_add/2)), V);
elseif z_add == 0 & y_add ~= 0 & x_add ~= 0
   V = cat(2, zeros(N(1), round(y_add/2), N(3)), V);
   V = cat(2, V, zeros(N(1), round(y_add/2), N(3)));
   N=size(V);
   V = cat(1, zeros(round(x_add/2), N(2), N(3)), V);
   V = cat(1, V, zeros(round(x_add/2), N(2), N(3)));
elseif x_add == 0 & y_add == 0 & z_add ~= 0
   V = cat(3, zeros(N(1), N(2), round(z_add/2)), V);
   V = cat(3, zeros(N(1), N(2), round(z_add/2)), V);
elseif x_add == 0 & y_add ~= 0 & z_add == 0
   V = cat(2, zeros(N(1), 1, N(3)), V);
   V = cat(2, V, zeros(N(1), 1, N(3)));
elseif x_add ~= 0 & y_add == 0 & z_add ~= 0
   V = cat(3, zeros(N(1), N(2), 1), V);
   V = cat(3, zeros(N(1), N(2), 1), V);
else
end
%%% affine transformation applied to the volumetric image
V_center = (size(V)+1)/2;
```

```
T1 = [1 0 0 0
    0 1 0 0
    0 0 1 0
    -V_center 1];
T_x = [1 0 0 0
    0 cos(anglex) -sin(anglex) 0
    0 sin(anglex) cos(anglex) 0
    0 0 0 1];
T_y = [cos(angley) 0 -sin(angley) 0
    0 1 0 0
    sin(angley) 0 cos(angley) 0
    0 0 0 1];
T_z = [cos(anglez) -sin(anglez) 0 0
    sin(anglez) cos(anglez) 0 0
    0 0 1 0
    0 0 0 1];
T2 = T_z*T_y*T_x;
T3 = [1 0 0 0
    0 1 0 0
    0 0 1 0
    V_center 1];
T = T1*T2*T3;
tform = maketform('affine', T);
R = makeresampler('linear', 'fill');
TDIMS_A = [1 2 3];
TDIMS_B = [1 2 3];
TSIZE_B = size(V);
TMAP_B = [];
F = 0;
V_rotated = tformarray(V, tform, R, TDIMS_A, TDIMS_B, TSIZE_B, TMAP_B, F);
end
```

5.1.5 Patterns and discretization

```
%%% definition of the centers of the voxels.

x = 1:1:size(dataM.geometry.V,1);
x = repmat(x, 1, size(dataM.geometry.V,2));
x = repmat(x, 1, size(dataM.geometry.V,3));
x = x';

y = ones(size(dataM.geometry.V,1), 1);
for i=2:size(dataM.geometry.V,2)
    y = cat(1, y, i*ones(size(dataM.geometry.V,1),1));
end
y = repmat(y, size(dataM.geometry.V,3), 1);

z = ones(size(dataM.geometry.V,1)*size(dataM.geometry.V,2),1);
for i=2:size(dataM.geometry.V,3)
```

```
    z = cat(1, z, i*ones(size(dataM.geometry.V,1)*size(dataM.geometry.V,2),1));
end
dataM.voxels.centers = [x y z];

%%% definition of the neighbours indexes of every voxel.

V_vert = dataM.geometry.V(:);

ind_out = find(V_vert==0);

dataM.voxels.neighbours = zeros(length(V_vert), 6);
neighbours_out = ismember(dataM.voxels.neighbours, ind_out);

for i=1:length(V_vert)
    dataM.voxels.neighbours(i,1) = (i-1);
    dataM.voxels.neighbours(i,2) = (i+1);
    dataM.voxels.neighbours(i,3) = (i-size(dataM.geometry.V,1));
    dataM.voxels.neighbours(i,4) = (i+size(dataM.geometry.V,1));
    dataM.voxels.neighbours(i,5) = (i-(size(dataM.geometry.V,1)*size(dataM.geometry.V,2)));
    dataM.voxels.neighbours(i,6) = (i+(size(dataM.geometry.V,1)*size(dataM.geometry.V,2)));
end

dataM.voxels.neighbours(find(dataM.voxels.neighbours>size(dataM.voxels.neighbours,1) |
dataM.voxels.neighbours<0))=0;
index = find(V_vert==0);
dataM.voxels.neighbours(index,:) = 0;

dataM.voxels.neighbours_out = ismember(dataM.voxels.neighbours, ind_out);

dataM.voxels.neighbours = dataM.voxels.neighbours.*(~dataM.voxels.neighbours_out);


%%% definition of the volume of every voxel.
hx = abs(dataF.valuesx(1) - dataF.valuesx(2));
hy = abs(dataF.valuesy(1) - dataF.valuesy(2));
hz = abs(dataF.valuesz(1) - dataF.valuesz(2));
dataM.voxels.volumes = ones(length(V_vert),1);
dataM.voxels.volumes = (hx*hy*hz)*dataM.voxels.volumes;

%%% definition of the names of the patterns.
dataM.patterns.name = final_pattern_list';

%%% Reshaping of the patterns. 'pattern' is a structure containing all the patterns, pattern_2 is a
structure that will contain the reshaped patterns

for k=1:length(final_pattern_list)
    for i=1:length(popup_list)
        str_comp = strfind(final_pattern_list{k}, popup_list{i});
        str_comp(end+1) = 2;
        if str_comp(1)==1
            ob.V = dataF.Objects.(popup_list{i}).V;
            ob.V(1:(dataM.restricted.minSum1-2), :, :) = [];
```

```
ob.V(:, 1:(dataM.restricted.minSum2-2), :) = [];
ob.V(:, :, 1:(dataM.restricted.minSum3-2)) = [];
ob.V((dataM.restricted.maxSum1-(dataM.restricted.minSum1-2)+2):end, :, :) = [];
ob.V(:, (dataM.restricted.maxSum2-(dataM.restricted.minSum2-2)+2):end, :) = [];
ob.V(:, :, (dataM.restricted.maxSum3-(dataM.restricted.minSum3-2)+2):end) = [];
length_str = length(popup_list{i});
sub.name = final_pattern_list{k};
sub.name(1:length(popup_list{i})) = [];
sub.V = ob.V;

if strcmp(sub.name, 'Volume')
    sub.indexes = find(sub.V==1);
elseif strcmp(sub.name, 'External Surface')
    sub.indexes = find(imedge3D(sub.V)==1);
    sub.V = 0*sub.V;
    sub.V(sub.indexes) = 1;
elseif strcmp(sub.name, 'Internal Volume')
    edge_index = find(imedge3d(sub.V)==1);
    sub.V(edge_index) = 0;
    sub.indexes = find(sub.V==1);
elseif strcmp(sub.name, 'Center')
    sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
    sub.V(central_voxel(2), central_voxel(1), central_voxel(3)) = 1;
    sub.indexes = find(sub.V==1);
elseif strcmp(sub.name, 'Line along x')
    sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
    sub.V(central_voxel(1),:,central_voxel(3)) = 1;
    sub.V = sub.V.*ob.V;
    sub.indexes = find(sub.V==1);
elseif strcmp(sub.name, 'Line along y')
    sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
    sub.V(:,central_voxel(2),central_voxel(3)) = 1;
    sub.V = sub.V.*ob.V;
    sub.indexes = find(sub.V==1);
elseif strcmp(sub.name, 'Line along z')
    sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
    sub.V(central_voxel(1),central_voxel(2),:) = 1;
    sub.V = sub.V.*ob.V;
    sub.indexes = find(sub.V==1);
elseif strcmp(sub.name, 'Point along x1')
    sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
    sub.V(central_voxel(1),:,central_voxel(3)) = 1;
    sub.V = sub.V.*ob.V;
    sub.indexes = find(sub.V==1);
    sub.indexes = min(sub.indexes);
    sub.V = zeros(size(ob.V,1), size(ob.V,2), size(ob.V,3));
    sub.V(sub.indexes) = 1;
elseif strcmp(sub.name, 'Point along x2')
    sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
    sub.V(central_voxel(1),:,central_voxel(3)) = 1;
    sub.V = sub.V.*ob.V;
    sub.indexes = find(sub.V==1);
```

```
                sub.indexes = max(sub.indexes);
                sub.V = zeros(size(ob.V,1), size(ob.V,2), size(ob.V,3));
                sub.V(sub.indexes) = 1;
            elseif strcmp(sub.name, 'Point along y1')
                sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
                sub.V(:,central_voxel(2),central_voxel(3)) = 1;
                sub.V = sub.V.*ob.V;
                sub.indexes = find(sub.V==1);
                sub.indexes = min(sub.indexes);
                sub.V = zeros(size(ob.V,1), size(ob.V,2), size(ob.V,3));
                sub.V(sub.indexes) = 1;
            elseif strcmp(sub.name, 'Point along y2')
                sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
                sub.V(:,central_voxel(2),central_voxel(3)) = 1;
                sub.V = sub.V.*ob.V;
                sub.indexes = find(sub.V==1);
                sub.indexes = max(sub.indexes);
                sub.V = zeros(size(ob.V,1), size(ob.V,2), size(ob.V,3));
                sub.V(sub.indexes) = 1;
            elseif strcmp(sub.name, 'Point along z1')
                sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
                sub.V(central_voxel(1),central_voxel(2),:) = 1;
                sub.V = sub.V.*ob.V;
                sub.indexes = find(sub.V==1);
                sub.indexes = min(sub.indexes);
                sub.V = zeros(size(ob.V,1), size(ob.V,2), size(ob.V,3));
                sub.V(sub.indexes) = 1;
            elseif strcmp(sub.name, 'Point along z2')
                sub.V = zeros(size(sub.V,1), size(sub.V,2), size(sub.V,3));
                sub.V(central_voxel(1),central_voxel(2),:) = 1;
                sub.V = sub.V.*ob.V;
                sub.indexes = find(sub.V==1);
                sub.indexes = max(sub.indexes);
                sub.V = zeros(size(ob.V,1), size(ob.V,2), size(ob.V,3));
                sub.V(sub.indexes) = 1;
            end
        end
    end

    pattern_2(k).V = sub.V;
    pattern_2(k).indexes = sub.indexes;
end




for i=1:size(pattern_2,2)
    pattern_2(i).indexes = find(pattern_2(i).V==1);
    dataM.patterns.array{i} = num2str(pattern_2(i).indexes');
    pattern_2(i).names = final_pattern_list{i};
end
```

```
dataM.geometry.pattern = pattern_2;

%%% for every pattern are searched the pattern that adjoin or intersect with others

for i=1:length(dataM.geometry.pattern)
    dataM.geometry.pattern(i).neighbouring_names = dataM.patterns.name;
    dataM.geometry.pattern(i). neighbouring_names = {};
    dataM.geometry.pattern(i).intersection_names = dataM.patterns.name;
    dataM.geometry.pattern(i).intersection_names = {};
end

for i=1:length(dataM.geometry.pattern)
    for j=1:length(dataM.geometry.pattern)
        V_xp = translation_xp(dataM.geometry.pattern(i).V, dataM.geometry.X, dataM.geometry.Y,
dataM.geometry.Z, abs(dataM.geometry.X(1,1,1) - dataM.geometry.X(1,2,1)));
        V_xm = translation_xm(dataM.geometry.pattern(i).V, dataM.geometry.X, dataM.geometry.Y,
dataM.geometry.Z, abs(dataM.geometry.X(1,1,1) - dataM.geometry.X(1,2,1)));
        V_yp = translation_yp(dataM.geometry.pattern(i).V, dataM.geometry.X, dataM.geometry.Y,
dataM.geometry.Z, abs(dataM.geometry.X(1,1,1) - dataM.geometry.X(1,2,1)));
        V_ym = translation_ym(dataM.geometry.pattern(i).V, dataM.geometry.X, dataM.geometry.Y,
dataM.geometry.Z, abs(dataM.geometry.X(1,1,1) - dataM.geometry.X(1,2,1)));
        V_zp = translation_zp(dataM.geometry.pattern(i).V, dataM.geometry.X, dataM.geometry.Y,
dataM.geometry.Z, abs(dataM.geometry.X(1,1,1) - dataM.geometry.X(1,2,1)));
        V_zm = translation_zm(dataM.geometry.pattern(i).V, dataM.geometry.X, dataM.geometry.Y,
dataM.geometry.Z, abs(dataM.geometry.X(1,1,1) - dataM.geometry.X(1,2,1)));
        V_big = dataM.geometry.pattern(i).V + V_xp + V_xm + V_yp + V_ym + V_zp + V_zm;
        V_big(find(V_big>0)) = 1;
        intersection = dataM.geometry.pattern(i).V + dataM.geometry.pattern(j).V;
        intersection_list = find(intersection == 2);
        neighbourV = V_big + dataM.geometry.pattern(j).V;
        neighbour_list = find(neighbourV == 2);

        if isempty(intersection_list) & isempty(neighbour_list)==0 & strcmp(dataM.patterns.name{i},
dataM.patterns.name{j})==0
            dataM.geometry.pattern(i). neighbour_names{j} = dataM.patterns.name{j};
        elseif isempty(intersection_list)==0 & strcmp(dataM.patterns.name{i},
dataM.patterns.name{j})==0
            dataM.geometry.pattern(i).intersection_names{j} = dataM.patterns.name{j};
        end

        emptyCellsC = cellfun(@isempty,dataM.geometry.pattern(i). neighbour_names);
        emptyCellsI = cellfun(@isempty,dataM.geometry.pattern(i).intersection_names);
        dataM.geometry.pattern(i). neighbour_names(emptyCellsC) = [];
        dataM.geometry.pattern(i).intersection_names(emptyCellsI) = [];
    end
end
```

5.1.6 Initial conditions

```
global dataM
%Extract initial conditions
a=get(handles.VariableList,'Value');
pat_ind=get(handles.PatternList,'Value');
pattern_list=get(handles.PatternList,'string');
VarPatt_list = get(handles.VarPatternList, 'string');
VarPattInd = get(handles.VarPatternList, 'value');
perm_list = get(handles.PermList, 'string');
dataM.permeabilities.external = str2num(get(handles.ExternalPermeability, 'string'));
c=get(handles.PermList, 'Value');
selected_pattern = VarPatt_list{VarPattInd};
selected_perm_pattern = perm_list{c};

perm_value = str2num(get(handles.PermValue, 'String'));

%Save initial conditions and the new pattern
try
    init = dataM.variables.initial_condition{a};
catch
    init =[];
end

b = get(handles.VarPatternList, 'value');

if isfield(init(b), 'permeability')==0
    init(b).permeability.names = perm_list;
    init(b).permeability.values = ones(1, length(perm_list));
end

if strcmp(selected_perm_pattern, 'Rest of voxels')
    V=dataM.geometry.V;

    if length(perm_list)>1
        ind_rest = find(strcmp(perm_list, 'Rest of voxels'));
        perm_list(ind_rest) = [];
        for i=1:length(perm_list)
            index = str2num(dataM.patterns.array{find(strcmp(dataM.patterns.name,
perm_list{i})==1)})';
            V(index) = 0;
        end
        index = str2num(dataM.patterns.array{find(strcmp(dataM.patterns.name,
selected_pattern)==1)})';        V(index) = 0;
    else
        index = str2num(dataM.patterns.array{find(strcmp(dataM.patterns.name,
selected_pattern)==1)})';        V(index) = 0;
    end

    index_selected_pattern = find(V==1);
    init(b).pattern_indexes = dataM.voxels.neighbours(str2num(init(b).pattern_array)',:);   %%%
these are the neighbours of every point in the pattern
    init(b).pattern_permeability = dataM.voxels.neighbours(str2num(init(b).pattern_array)',:);
    %%% by default, every pattern is permeable with itself and with the others and impermeabile with
the external
    init(b).pattern_permeability(find(init(b).pattern_permeability~=0))=1;   %è permeabile con se
stesso

init(b).pattern_permeability(find(init(b).pattern_permeability==0))=dataM.permeabilities.external;
%%% Permeability with the external
    matrix_perm = ismember(init(b).pattern_indexes, index_selected_pattern);
%%% The ones are the indexes that belongs to the two pattern, so they are the boarder indexes

    index_perm_neigh = find(matrix_perm==1);
    init(b).pattern_permeability(index_perm_neigh) = perm_value;
else
    index_selected_pattern = str2num(dataM.patterns.array{find(strcmp(dataM.patterns.name,
perm_list{c})==1)})';
    init(b).pattern_indexes = dataM.voxels.neighbours(str2num(init(b).pattern_array)'
init(b).pattern_permeability = dataM.voxels.neighbours(str2num(init(b).pattern_array)',:);
    init(b).pattern_permeability(find(init(b).pattern_permeability~=0))=1

init(b).pattern_permeability(find(init(b).pattern_permeability==0))=dataM.permeabilities.external
matrix_perm = ismember(init(b).pattern_indexes, index_selected_pattern    index_perm_neigh =
find(matrix_perm==1);
    init(b).pattern_permeability(index_perm_neigh) = perm_value;
```

```
end

d = find(strcmp(perm_list, selected_perm_pattern)==1);
init(b).permeability.values(d) = perm_value;
init(b).permeability.index_perm_neigh = index_perm_neigh;
dataM.variables.initial_condition{a} = init;
```

5.1.7 Laplacian parser

```
    %Generate the equation system for other odes
    for i=1:L_eq,
        for j=1:L_vox,
            eq=Equations{i};
            for k=1:L_var,
                var=dataM.variables.name{k};
                perm_str = dataM.variables.laplacian_str{k};
                perm_str_j = perm_str{j};
                perm1 = perm_str_j{1};
                perm2 = perm_str_j{2};
                perm3 = perm_str_j{3};
                perm4 = perm_str_j{4};
                perm5 = perm_str_j{5};
                perm6 = perm_str_j{6};
                perm12 = perm_str_j{7};

                %Parser for the Laplacian operator
                Lapl=['Lapl(' var ')'];
                [int_str]=strfind(eq,Lapl);
                %The s index is determined automatically by the order of
                %variable names and by the current voxel.



                s_add12=['s(' NumToStr{L_vox*(k-1)+j} ')']; % central

                if j==1
                     s_add1= '0.0'; % x left
                else
                     s_add1=['s(' NumToStr{L_vox*(k-1)+j-1} ')']; % x left
                end

                if j==L_vox
                    s_add2 = '0.0'; %x right
                else
                    s_add2 = ['s(' NumToStr{L_vox*(k-1)+j+1} ')']; % x right
                end

                if j>=1 & j<(size(dataM.geometry.V,1)+1)
                    s_add3 = '0.0'; % y left
                else
                    s_add2 = ['s(' NumToStr{L_vox*(k-1)+j-size(dataM.geometry.V,1) } ')']; % y left
                end

                if j>(L_vox-size(dataM.geometry.V,1))
                    s_add4 = '0.0'; %y right
                else
                    s_add4=['s(' NumToStr{L_vox*(k-1)+j+size(dataM.geometry.V,1)} ')']; % y right
                end

                if j>=1 & j<(size(dataM.geometry.V,1)*size(dataM.geometry.V,2)+1)
                    s_add5 = '0.0'; %z left
                else
                    s_add5=['s(' NumToStr{L_vox*(k-1)+j-
size(dataM.geometry.V,1)*size(dataM.geometry.V,2)} ')']; % z left
                end

                if j>(L_vox - size(dataM.geometry.V,1)*size(dataM.geometry.V,2))
                    s_add6 = '0.0'; % z right
                else
                    s_add6=['s(' NumToStr{L_vox*(k-
1)+j+size(dataM.geometry.V,1)*size(dataM.geometry.V,2)} ')']; % z right
                end
```

```
                s_add=[perm1 '*' s_add1 '-' perm12 '*' s_add12 '+' perm2 '*' s_add2 '+' perm3 '*'
s_add3 '+' perm4 '*' s_add4 '+' perm5 '*' s_add5 '+' perm6 '*' s_add6];




                for kk = flipdim(int_str,2),
                    eq(kk:(kk+length(Lapl)-1))=[];
                    eq=[eq(1:(kk-1)) '(' s_add ')/' num2str(dx2) eq(kk:end)];
                end
                %Parser for the normal algebra of variables
                [int_str]=strfind(eq,var);
                %The s index is determined automatically by the order of
                %variable names and by the current voxel.
                s_add=['s(' NumToStr{L_vox*(k-1)+j} ')'];
                for kk = flipdim(int_str,2),
                    if kk==1,
                        if length(var)==length(eq),
                            eq=s_add;
                        else
                            if ~isempty(intersect(eq(kk+length(var)),'><=^+-/*,(); ')),
                                eq(kk:(kk+length(var)-1))=[];
                                eq=[eq(1:(kk-1)) s_add eq(kk:end)];
                            end
                        end
                    elseif (kk+length(var))>length(eq),
                        if ~isempty(intersect(eq(kk-1),'d^+-/*,(); ')),
                            eq(kk:(kk+length(var)-1))=[];
                            eq=[eq(1:(kk-1)) s_add eq(kk:end)];
                        end
                    else
                        if ~isempty(intersect(eq(kk-1),'d^+-/*,(); ')) &&
~isempty(intersect(eq(kk+length(var)),'><=^+-/*,(); ')),
                            eq(kk:(kk+length(var)-1))=[];
                            eq=[eq(1:(kk-1)) s_add eq(kk:end)];
                        end
                    end
                end
            end
            for k=1:L_par,
                par=dataM.parameters.name{k};
                [int_str]=strfind(eq,par);
                %The p index is determined automatically by the order of
                %parameter names and by the current voxel.
                p_add=['p(' NumToStr{L_vox*(k-1)+j} ')'];
                for kk = flipdim(int_str,2),
                    if kk==1,
                        if length(par)==length(eq),
                            eq=p_add;
                        else
                            if ~isempty(intersect(eq(kk+length(par)),'><=^+-/*,(); ')),
                                eq(kk:(kk+length(par)-1))=[];
                                eq=[eq(1:(kk-1)) p_add eq(kk:end)];
                            end
                        end
                    elseif (kk+length(par))>length(eq),
                        if ~isempty(intersect(eq(kk-1),'><=^+-/*,(); ')),
                            eq(kk:(kk+length(par)-1))=[];
                            eq=[eq(1:(kk-1)) p_add eq(kk:end)];
                        end
                    else
                        if ~isempty(intersect(eq(kk-1),'><=^+-/*,(); ')) &&
~isempty(intersect(eq(kk+length(par)),'><=^+-/*,(); ')),
                            eq(kk:(kk+length(par)-1))=[];
                            eq=[eq(1:(kk-1)) p_add eq(kk:end)];
                        end
                    end
                end
            end
            prec_inp = 0;
            for k=1:L_inp,
                %Identify the voxels where the input is present
                inp_ind=dataM.inputs.voxel{k};
                inp=dataM.inputs.name{k};
                %Check if the input variable is present in the current equation
```

```
                [int_str]=strfind(eq,inp);
                for kk = flipdim(int_str,2),
                    if kk==1,
                        if kk+length(inp)>length(eq),
                            eq(kk:(kk+length(inp)-1))=[];
                            %If the voxel is correct, insert the input variable
                            %in the equation with the correct index
                            inp_j = find(inp_ind==j);
                            if inp_j,
                                inp_add=['u(' NumToStr{inp_j+prec_inp} ')'];
                                eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                            else
                                inp_add=['0.0'];
                                eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                            end
                        else
                            if ~isempty(intersect(eq(kk+length(inp)),'><=^+-/*,(); ')),
                                eq(kk:(kk+length(inp)-1))=[];
                                %If the voxel is correct, insert the input variable
                                %in the equation with the correct index
                                inp_j = find(inp_ind==j);
                                if inp_j,
                                    inp_add=['u(' NumToStr{inp_j+prec_inp} ')'];
                                    eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                                else
                                    inp_add=['0.0'];
                                    eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                                end
                            end
                        end
                    elseif (kk+length(inp))>length(eq),
                        if ~isempty(intersect(eq(kk-1),'><=^+-/*,(); ')),
                            eq(kk:(kk+length(inp)-1))=[];
                            %If the voxel is correct, insert the input variable
                            %in the equation with the correct index
                            inp_j = find(inp_ind==j);
                            if inp_j,
                                inp_add=['u(' NumToStr{inp_j+prec_inp} ')'];
                                eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                            else
                                inp_add=['0.0'];
                                eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                            end
                        end
                    else
                        if ~isempty(intersect(eq(kk-1),'><=^+-/*,(); ')) &&
~isempty(intersect(eq(kk+length(inp)),'><=^+-/*,(); ')),
                            eq(kk:(kk+length(inp)-1))=[];
                            %If the voxel is correct, insert the input variable
                            %in the equation with the correct index
                            inp_j = find(inp_ind==j);
                            if inp_j,
                                inp_add=['u(' NumToStr{inp_j+prec_inp} ')'];
                                eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                            else
                                inp_add=['0.0'];
                                eq=[eq(1:(kk-1)) inp_add eq(kk:end)];
                            end
                        end
                    end
                end
                if dataM.inputs.type{k}==1,
                prec_inp = prec_inp + length(inp_ind);
                end
            end
            eq=['ds(' NumToStr{L_vox*(i-1)+j} ')=' eq ';'];
            eq_fid=[eq_fid ['\n' eq]];
        end
    end
    eq_fid=[eq_fid '\n'];
end
```

Bibliography

1. Augustine, G.J.C., M. P.; Smith, S. J., *Calcium entry and transmitter release at voltage-clamped nerve terminals of squid.* The Journal of Physiology, 1985. **367**: p. 163-181.
2. Cheng, H.L., W. J.; Cannell, M. B., *Calcium Sparks: Elementary Events Underlying Excitation-Contraction Coupling in Heart Muscle.* Science, 1993. **262**: p. 740-744.
3. Bruns, D.J., R., *Real-time measurement of transmitter release from single synaptic vesicles.* Nature, 1995. **377**: p. 62-65.
4. Clapham, D.E., *Calcium Signaling.* Cell, 1995. **80**: p. 259-268.
5. Roberts, W.M., *Localization of Calcium Signals by a Mobile Calcium Buffer in Frog Saccular Hair Cells.* The Journal of Neuroscience, 1994. **14(5)**: p. 3246-3262.
6. Bortolozzi, M.L., Andrea; Mammano, Fabio, *Calcium microdomains at presynaptic active zones of vertebrate hair cells unmasked by stochastic deconvolution.* Cell Calcium, 2008. **44**: p. 158-168.
7. Lewis, R.S.H., A. J., *Voltage- and ion-dependent conductances in solitary vertebrate hair cells.* Nature, 1983. **304**: p. 538-541.
8. Art, J.J.F., R., *Variation of membrane properties in hair cells isolated from the turtle cochlea.* The Journal of Physiology, 1987. **385**: p. 207-242.
9. Roberts, W.M.J., R. A.; Hudspeth, A. J., *Colocalization of Ion Channels Involved in Frequency Selectivity and Synaptic Transmission at Presynaptic Active Zones of Hair Cells.* The Journal of Neuroscience, 1990. **10(11)**: p. 3662-3684.
10. Zenisek, D.D., V. ; Wan, L. ; Almers, W., *Imaging Calcium Entry Sites and Ribbon Structures in Two Presynaptic Cells.* The Journal of Neuroscience, 2003. **23(7)**: p. 2538-2548.
11. Berridge, M.J., *Calcium microdomains: organization and function.* Cell Calcium, 2006. **40**(5-6): p. 405-12.
12. Oheim, M., F. Kirchhoff, and W. Stuhmer, *Calcium microdomains in regulated exocytosis.* Cell Calcium, 2006. **40**(5-6): p. 423-39.
13. Tucker, T.F., R., *Confocal Imaging of Calcium Microdomains and Calcium Extrusion in Turtle Hair Cells.* Neuron, 1995. **15**: p. 1323-1335.
14. Issa, N.P.H., A. J., *Clustering of Ca2+ channels and Ca2+-activated K+ channels at fluorescently labeled presynaptic active zones of hair cells.* Neurobiology, 1994. **91**: p. 7578-7882.
15. Rispoli, G., et al., *Dynamics of intracellular calcium in hair cells isolated from the semicircular canal of the frog.* Cell Calcium, 2001. **30**(2): p. 131-40.
16. Lelli, A.P., Paola; Martini, Marta; Ciubotaru, Catalin D.; Prigioni, Ivo; Valli, Paolo; Rossi, Maria L.; Mammano, Fabio, *Presynaptic Calcium Stores Modulate Afferent Release in Vestibular Hair Cells.* The Journal of Neuroscience, 2003. **23(17)**: p. 6894-6903.
17. Simon, S.M.L., R. R., *Compartmentalization of the submembrane calcium activity during calcium influx and its significance in transmitter release.* Biophysical Journal, 1985. **48**: p. 485-498.
18. Nowycky, M.C.P., M. J., *Time courses of calcium and calcium-bound buffers following calcium influx in a model cell.* Biophysical Journal, 1993. **64**: p. 77-91.
19. Klingauf, J.N., E., *Modeling Buffered Ca2+ Diffusion Near the Membrane.* Biophysical Journal, 1997. **72(2 Pt 1)**: p. 674-690.

20. Mammano, F.B., Mario, *Ca2+ Imaging: Principles of Analysis and Enhancement*, in *Calcium Measurement Methods*, O.H.P. Alexei Verkhratsky, Editor. 2010, Humana Press, a part of Springer Science+Business Media. p. 57-80.
21. Maravall, M.M., Z. F.; Sabatini, B. L.; Svoboda K., *Estimating Intracellular Calcium Concentrations and Buffering without Wavelength Ratioing.* Biophysical Journal, 2000. **78(5)**: p. 2655-2667.
22. Eberhard, M. and P. Erne, *Calcium binding to fluorescent calcium indicators: Calcium green, calcium orange and calcium crimson.* Biochemical and Biophysical Research Communications, 1991. **180**(1): p. 209-215.
23. Nägerl, U.V.N., D.; Mody, I.; Vergara, J. L., *Binding Kinetics of Calbindin-D28k Determined by Flash Photolysis of Caged Ca2+*. Biophysical Journal, 2000. **79**: p. 3009-3018.
24. Ohtani, M., et al., *Cholinergic agonists increase intracellular calcium concentration in frog vestibular hair cells.* Hearing Research, 1994. **80**(2): p. 167-173.
25. Lumpkin, E.A.H., A. J., *Detection of Ca2+ entry through mechanosensitive channels localizes the site of mechanoelectrical transduction in hair cells.* Biophysics, 1995. **92**: p. 10297-10301.
26. Edmonds, B.R., R. ; Schwaller, B.; Roberts, W.M., *Calretinin modifies presynaptic calcium signaling in frog saccular hair cells.* Nature neuroscience, 2000. **3**: p. 786-790.
27. Saftenku, E.E., *Effects of calretinin on Ca2+ signals in cerebellar granule cells: implications of cooperative Ca2+ binding.* Cerebellum, 2012. **11**(1): p. 102-20.
28. Faas, G.C.S., B.; Vergara, J. L.; Mody, I., *Resolving the Fast Kinetics of Cooperative Binding: Ca2þ Buffering by Calretinin.* PLoS Biology, 2007. **5**(11).
29. Harkins, A.B.K., N; Baylor, S. M., *Resting Myoplasmic Free Calcium in Frog Skeletal Muscle Fibers Estimated with Fluo-3.* Biophysical Journal, 1993. **65**: p. 865-881.
30. Paredes, R.M., et al., *Chemical calcium indicators.* Methods, 2008. **46**(3): p. 143-51.
31. Goldberg, J.H.T., G.; Aronov, D.; Yuste, R., *Calcium Microdomains in Aspiny Dendrites.* Neuron, 2003. **40**: p. 807-821.
32. Zhang, C.L., et al., *Action potentials induce uniform calcium influx in mammalian myelinated optic nerves.* J Neurophysiol, 2006. **96**(2): p. 695-709.
33. Ellis-Davies, G.C.R.K., J. H.; Barsoti, R. J., *Laser Photolysis of Caged Calcium: Rates of Calcium Release by Nitrophenyl-EGTA and DM-Nitrophen.* Biophysical Journal, 1996. **70**: p. 1006-1016.
34. Engwirda, D., *Locally-optimal Delaunay-refinement and optimisation-based mesh generation*, in *School of Mathematics and Statistics*. 2015, The University of Sydney.
35. Fang, Q.B., D. A., *Tetrahedral mesh generation from volumetric binary and gray-scale images.* IEEE International Symposium on Biomedial Imaging: Nano to Macro, 2009: p. 1142-1145.
36. Hiraoka, Y.S., J. W.; Agard, D. A., *Determination of three-dimensional imaging proprieties of a light microscope system. Partial confocal behaviour in epifluorescence microscopy.* Biophysical Journal, 1990. **57**(2): p. 325-333.
37. Ceriani, F., et al., *Design and Construction of a Cost-Effective Spinning Disk System for Live Imaging of Inner Ear Tissue*, in *Auditory and Vestibular Research: Methods and Protocols*, B. Sokolowski, Editor. 2016, Springer New York: New York, NY. p. 223-241.
38. Ellis-Davies, G.C.R., *Neurobiology with Caged Calcium.* Chemical Reviews, 2008. **108**(5): p. 1603-1613.
39. Masters, J.R., *HeLa cells 50 years on: the good, the bad and the ugly.* Nature Reviews Cancer, 2002. **2**(4): p. 315-319.