UNIVERSITY OF NAPLES "FEDERICO II"

Filippo Menolascina

Synthetic Gene Networks Identification and Control by means of Microfluidic Devices

A dissertation submitted in partial fulfillment of the requirements for the degree
Doctor of Philosophy
in
Computational Biology and Bioinformatics

Advisors:
Dr. Diego DI BERNARDO
Co-Advisors:
Prof. Sandro BANFI
Prof. Mario DI BERNARDO

Year 2011

To my parents.

To Prof. Vito Ranieri.
Your lessons will harbor
in me forever.

ACKNOWLEDGEMENTS

The last three years have probably been the most intense of all my life; I have spent much of this time learning new things, carrying out experiments, traveling thousands miles and meeting people that allowed me to improve my work. For all this I want to sincerely thank Dr. Diego di Bernardo; it is difficult to overstate my gratitude to him. With his enthusiasm, his inspiration, and his great efforts to explain things clearly and simply, he helped in achieving the results that are at the basis of this dissertation. Throughout my PhD, he provided encouragement, sound advice, good teaching, good company, and lots of good ideas.

A special thank goes to Prof. Mario di Bernardo for having honored me with his supervision: every word he said has been a lesson for me. I will do my best to keep applying what he taught me so as to perpetrate this cycle. It has been a privilege for me to work with Dr. Stefania Santini; her suggestions have been central to obtain the results reported in this manuscript. I would like to acknowledge the contribution of Dr. Sandro Banfi, Dr. Roman Polishchuk and Dr. Luca De Stefano who served as thesis committee members and contributed at different levels to the work reported in these pages. Prof. Jeff Hasty, at University of California at San Diego, hosted me in his laboratory and his collabora-

tors Dr. Mike Ferry, Ivan Razinkov and Octavio Mondragon-Palomino, introduced me to the *inner secrets* of microfluidics. To these people, as well as to Emanuele Orabona, who helped me with microfluidic device production here in Naples, go my most sincere thanks.

I would like to acknowledge the work of Prof. Carlo Sansone and Raffaele La Brocca in the optimisation of the image segmentation code used for yeast cells tracking. Velia Siciliano as well as Chiara Fracassi and Nicoletta Moretti were instrumental in obtaining the PFL and NOPFL cell lines that allowed the study reported in Chapter 6 of this thesis; thank you girls for your work.

I am indebted to my many colleagues for providing a stimulating and fun environment in which to learn and grow. I wish to thank Dr. Francesco Iorio, for the many fruitful discussions and amazing company. I cannot avoid mentioning the past and present members of Diego di Bernardo's laboratory: Vincenzo Belcastro, Annamaria Carissimo, Gopu Dharmalingam, Stefania Criscuolo, Giulia Cuccato, Luisa Cutillo, Gennaro Gambardella, Immacolata Garzilli, Mario Lauria, Lucia Marucci, Margherita Mutarelli, Roberto Pagliarini, Maria Aurelia Ricci, Rossella Rispoli and Simona Ventre all contributed to make happier my stay in Naples. Among my colleagues, I do wish to sincerely thank Gianfranco Fiore: part of the work presented in this manuscript would have been much more complex without his talent and propensity for hard work. I wish him all the best and, most importantly, to be successful in realizing his dreams. I am sincerely grateful to my best friends Elisabetta Mancini and Emma De Marinis for having struggled to make me feeling home closer along these years; I am sure there is no distance that can take us apart. Last, and most importantly, I would like to thank Alda, my brother Alberto and my parents Alberta and Antonio for having loved and supported me while sharing with me the highs and lows of these years. I will do my best to make you feel proud of me.

CONTENTS

CHAPTER 1

INTRODUCTION

What I cannot create,
I do not understand.

Richard Feynman

Contents

Summary

A brief introduction on role of systems biology in the modeling of biological systems is provided in this chapter. Related works in the field are introduced, as well as, a primer on a key technology used to extend the applications of control systems theory to life sciences: microfluidics.

1.1 Background and motivation

Being an intrinsically qualitative science, biology has long suffered from a scarce attitude towards the quantification of complex regulatory mechanisms underlying life. Nevertheless the whole field underwent major changes due to the recent introduction of novel and powerful technologies: Next-Generation Sequencing platforms, for example, allowed to achieve an unprecedented level of precision in the identification and abundance estimation and of nucleic acids within a cell. Although several issues still need to be addressed for a truly reliable quantification of nucleic acids and protein abundance, the availability of such technologies boosted the interest of the scientific community towards a more quantitative understanding of the inner cell dynamics governing life.

The roots of the *quest for mathematical models* in biology should be looked for in the field of biochemistry: enzyme kinetics has been the field where most of the methods, we now recognise as milestones in systems biology, were first developed. After the 1960s and 1970s, the achievements in molecular biology in the 1980s, and the advent of functional genomics in the 1990s, catalysed the development of gene network modeling that flourished in the first decade on the new millennium.

Pretty soon, it became evident that the study of the properties of the gene regulatory networks via mathematical modeling could provide new insights into fundamental biological mechanisms. Control Engineering comes into play at this point, by providing a well-established and

comprehensive set of analytical tools allowing full exploration of the dynamical properties of biochemical networks.

This is why key biological phenomena, like cell cycle, got renamed as "limit cycles", a very well known term in the control theory jargon. Interestingly, this was not only a reinterpretation procedure since, among the many well established formal methods in control and systems theory, there were tools that allowed to formally predict basic biological mechanisms such as the G0-G1 transition within the framework of bifurcation theory [1, 2].

It can be argued that the ability to move from the mere description of the behavior of a biological system to the its prediction, has been the real added value brought by control engineering in the field of Systems Biology. After decades of gene network modeling, it is time to use the information provided by models not only to predict the behavior of the cell, but to control it.

Proving the controllability of gene networks in vivo is a quite complex task both from the theoretical and practical point of view; genes within the cell are connected in very complex networks whose topology is often unknown. Moreover, due to biological variability, despite cells having exactly the same genome, i.e. genetic information is identical, their behavior may differ.

In order to address the issues mentioned above, in the following Chapters, I will demonstrate how control engineering can be successfully applied to precisely control gene expression in a population of cells. In particular, I constructed an integrated technological platform based on a *microfludic* device to observe and manipulate engineered cells: controlling endogenous circuits, as a matter of fact, is hardly feasible due to uncertainty on the needed mathematical model. Therefore I decided to use a strain of S. cerevisiae, modified to integrate IRMA [3], a complex synthetic gene network whose dynamics can be elicited by Galactose/Glucose

administration and quantified by fluorescence microscopy. The control algorithm has been implemented on a computer that acquired the images and closed the feedback loop by automatically modifying the input compound sensed by the cells in real time. I demonstrated that using this platform, it is indeed possible to control the behavior of a population of yeast cells.

Few other examples of such an attempt can be found in literature and most of them are computational [4–7]. Recently in [8] and [9] authors described the development of in-vivo strategies based on microfluidics and optogenetics to control endogenous and synthetic circuits respectively. Nevertheless, it should be noted that control horizon of these works is in the range of hundreds of minutes (at most) and, some of them, require human intervention. Here, I will describe a fully automated platform that bypasses these issues by coupling microfluidic devices to time-lapse microscopy.

In Chapter 6, I will also show that system identification techniques, coupled to control engineering principles, can be successfully applied to predict the behavior of a transcriptional positive feedback.

1.2 Control theory in systems and synthetic biology

Control engineering grew out of the need to analyse dynamical systems, that is systems whose internal state evolves with time, and to modify their behavior at will. Two main approaches have been used to achieve this objective: the open and closed loop control (represented in Fig. 1.1); in closed loop control, moreover, we distinguish the feedforward and feedback loop control configurations. The feedback loop control (panel C in Fig. 1.1) is by far, the mostly employed scheme. Its working principle is pretty simple: in order to obtain a perfect control ($y \cong r$),

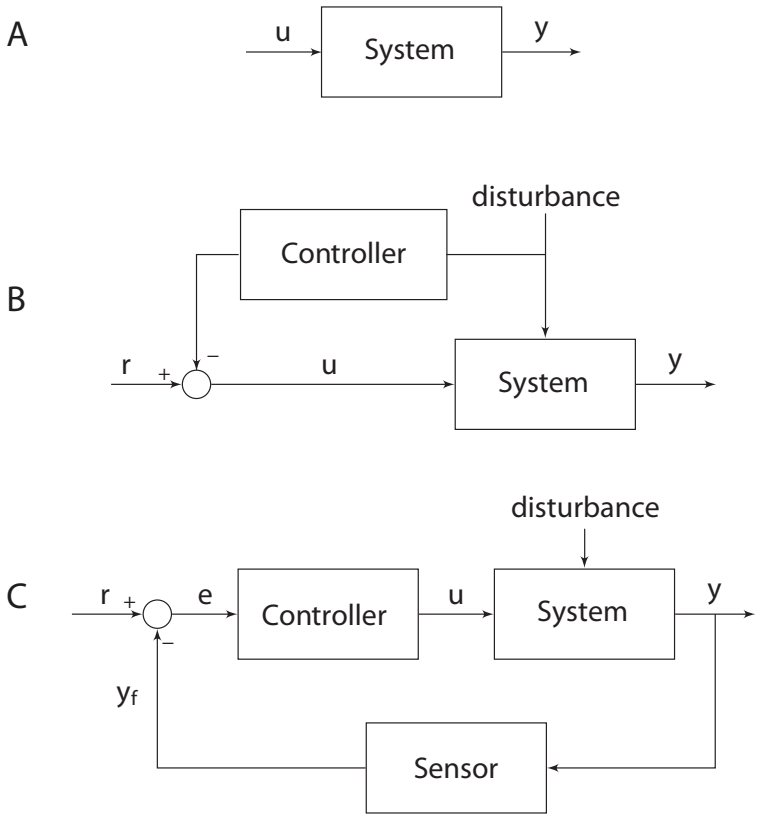

Figure 1.1: **Classes of control strategies.** In open loop control (panel A) the system is simply forced with a desired input u, no readout of the actual system response output y is used. In feedforward configuration (panel B) a sensor (or "measuring device") is used to directly measure the disturbance as it enters the process. The controller is fed with the information transmitted from this sensor. The feedforward controller computes necessary change in u, so that, when the effect of the disturbance is combined with the effect of the change in the manipulated variable, there will be no change in the controlled variable at all. The controlled variable is always kept at its setpoint and hence disturbances have no effect on the process. This perspective is abandoned in feedback control configuration (panel C); in this case the desired output, r is directly compared with y to compute the error e. The greater the error e the more distant the process is from the desired state. The controller uses e to compute the optimal value of u that will minimise the error e.

one has to stimulate (through the input u) a process more strongly if the difference between the desired and actual output ($e = r - y$) is big and less if the system is behaving approximately as wanted ($e \cong 0$).

Since its invention, in 1934 by Harold Black [10], this configuration has evolved in much more complex strategies that include e.g. prediction (Model Predictive Control schemes) and Adaptation (Adaptive Control techniques) [11]. Nevertheless, the basic idea remains unchanged and so do the main types of control experiments: as a matter of fact, on the basis of the nature of the r signal two different kinds of control tasks can be envisioned.

The simplest kind of control objective to be achieved is one in which the system is required to reach an output value that does not change over time (panel A Fig. 1.2).

A more complex control experiment is meant to force the system to follow a time varying reference signal r (panel B of Fig. 1.2). Even though these tasks seem artificial in nature, most of the biological processes, from chemotaxis to cell homeostasis, can be easily interpreted from a control theoretical perspective [12]. As a matter of fact, cells do naturally need to *(a)* maintain feasible equilibria for the reactions taking place within their membranes and *(b)* to use signals from the external environment to adapt to new conditions [13]. During the last decades control theory has been acknowledged as a privileged discipline in the study of biological mechanisms [1] and this catalysed the rising of Systems Biology. Aiming at providing a quantitative understanding of biological mechanisms underlying life, systems biology, has its foundations in control and systems theory; tools like sensitivity analysis [14] and optimal input design strategies [15] [16] have been extensively applied to biological systems. Recently, this effort has been complemented by the growing interest for the modeling of biological networks built from scratch in the framework of Synthetic Biology. Applying principles from system identification has dramatically improved our ability to find suitable mathematical models for synthetic circuits, as well as, identifying kinetic parameters to be tuned in order to optimise gene networks behavior [17]. These achievements significantly

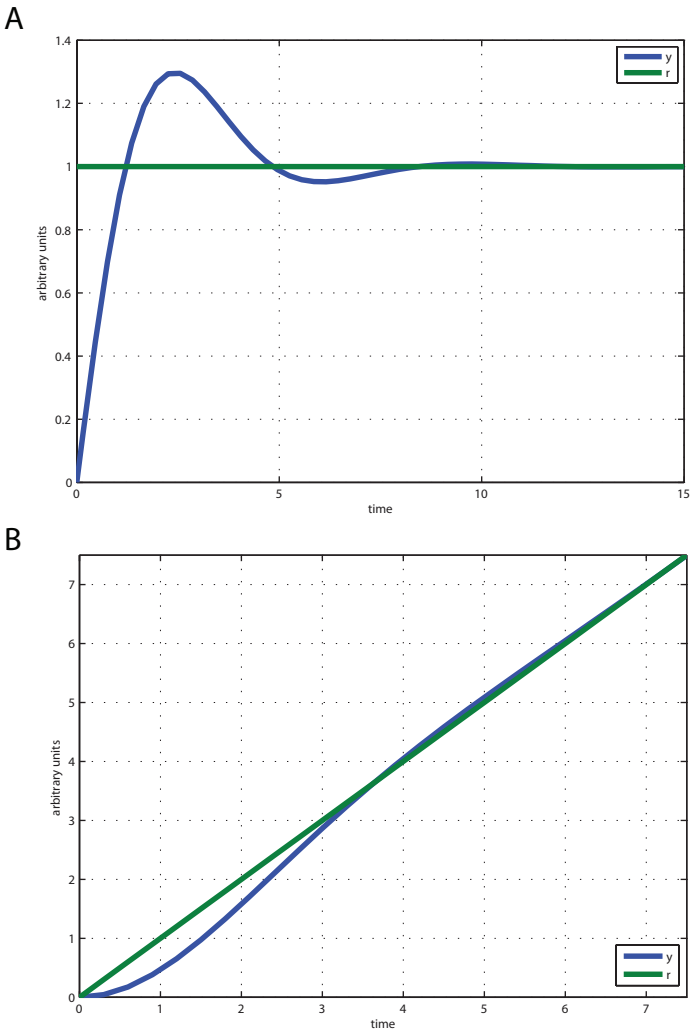

Figure 1.2: **Set-point and signal tracking control experiments.** Panel A shows a typical set-point experiment: the green signal, the desired output, stays fixed over time, while the blue signal, the actual system's output, approaches it as time progresses. In panel B a signal tracking task is presented. In this case the desired output is a time varying signal, a ramp ($r(t) = t$).

improved our understanding of the innermost cellular mechanisms: a significant breakthrough in this context has been provided by the recent availability of single cell quantification techniques [18]. Moving from assays mainly meant to observe population level behaviors (Western Blot,

qPCR, FACS etc) to microfluidics based experiments [19], we were able to track trajectories of single cells and, thus, gain deeper understanding of phenomena such as biological noise.

1.3 Microfluidics: a paradigm shift

The growing attention of the scientific community towards the quantitative aspects behind gene networks led to the introduction of new technologies meant to ease the analytical study of their dynamics. Fluorescence Activated Cell Sorting (FACS) technology is one of these: by providing information (e.g. fluorescence, cell-cycle phase etc) on a large cell population ($10^4 - 10^6$ cells), FACS can return "snapshots" of how a population, as a whole, behaves in time; unfortunately, FACS is not able to tell us much about single cell's dynamics. The impact of such a limitation is represented in Figure 1.3.

Microfluidics allows to overcome FACS limitations, but at the same time providing the possibility of following single cell dynamics in real-time with minimal invasiveness as compared to FACS; all of this at low operating costs.

Microfluidics gained momentum in the recent years as a result of the convergence of three well consolidated trends in biomedicine: *(a)* limitation of the operating costs associated to experiments, *(b)* the need to obtain real-time in-vivo quantification of cellular behaviors and *(c)* reduction of the experimental equipment dimensions.

Although these goals appear in conflict, microfluidics proved to be able to address all of them at once by proposing the use of polymer based devices in which microscopic channels (only a couple of hairs wide) and chambers were *engraved* [20][19]. The polymer, processed so as to display the desired structures, can be attached to a glass slide to allow time-lapse imaging of trapped cells.

Thanks to such a strategy researchers in this field have been able to

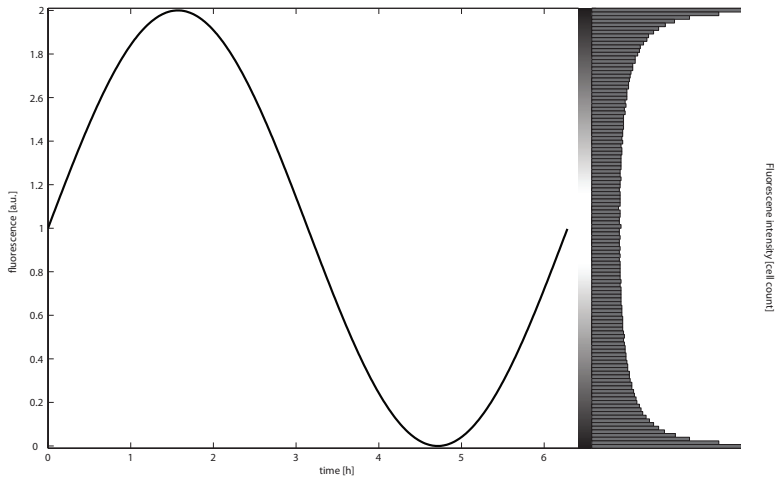

Figure 1.3: **Hypothetical behavior of an oscillatory gene network whose oscillations can be followed by means of fluorescence.** (Left Panel): the expected signal quantified in a single cell. (Right side): hypothetical histogram produced by FACS shows how a population of un-synchronised cells would look like: at each time point a bimodal distribution would be observed reflecting how cells are distributed between the maximum and minimum fluorescence values. Since several other circuits may give rise to the same FACS result, it is evident that this technique is not suitable in this context.

successfully accomplish cell loading, trapping and medium feeding within these devices in the recent years [21].

Since then, several applications of microfluidic devices have been proposed ranging from single cell genomics [22] to microbial community behavior studies [23] (refer to [24] for a comprehensive review).

Quickly, microfluidics has become one of the hot topics in the life sciences; both molecular and cell biology took great advantage from the use of principles governing fluid flows at the micrometer scale. As research proceeded, long term experiments (24-72 hrs) became feasible: at this point the dimension shrinkage resulted in limited reagent consumption (and reduced costs) as well as in a much finer control over compound concentrations in the micro-environments [25].

Single cell trajectories, as well as, macro-scale behaviors can be anal-

ysed simply by coupling time-lapse microscopy with microfluidics and image processing algorithms. Automatic image analysis can be used to segment wide field images to quickly obtain meaningful information about the property of interest (fluorescence of reporter gene, cell motility etc).

Given the specific characteristic of microfluidics-based experimental platforms, I decided to implement this strategy for our control experiment: yeast cell long-term viability within these devices had already been demonstrated [25] and an effective technology for compound concentration fine tuning had been demonstrated in [26].

In order to take full advantage of microfluidics, three steps are required: *(a)* a device has to be design that matches specifications set by the experiment to be carried out, *(b)* the device enters a round of iterative optimisation during which a Finite Element Methods (FEM) are used to check that hydrodynamical conditions satisfy biocompatibility requirements, *(c)* the device is prototyped and fabricated. All of these steps will be briefly reviewed in the Chapter 4 with a specific emphasis on the choice that were made in our setup to improve the yield of the whole process.

CHAPTER 2

IRMA: A SYNTHETIC TESTBED FOR IN-VIVO CONTROL

Remember that all models are wrong;
the practical question is how wrong
do they have to be to not be useful.

George E. P. Box

Contents

Summary

A description of the synthetic gene network, IRMA, which I used as testbed for the in-vivo control experiments is presented in this Chapter. A brief overview of its design is introduced, as well as, a mathematical model capturing its dynamics. Details concerning the intrinsic nature of IRMA are exposed to set the ground for further discussion about the hurdles encountered in controlling biological systems.

2.1 IRMA: topology and peculiarities

IRMA is a synthetic gene network built in yeast S. Cerevisiae. It consists of five genes, CBF1-GFP, ASH1, SWI5, GAL4 and GAL80 and its topology comprises both transcriptional and protein-protein regulation mechanisms. The IRMA network features several *motifs* [27], such as positive and negative feedback loops, that contribute to increase its complexity moving it closer to endogenous gene networks shape.

A detailed discussion of its biological implementation can be found in [3]. Fig. 2.1 shows a schematic diagram of the activation/repression links among the five genes. Note that the only external input is the presence or absence of Galactose (GAL) in the medium; this sugar prevents GAL80 from binding GAL4 thereby allowing the latter to activate the GAL10 promoter driving the *master regulator*, SWI5, transcription. This gives rise to the activation of the whole network as reported in Fig. 2.2.

As it is evident from Fig. 2.1, when Glucose (GLU) is added to the medium, GAL80 is free to inhibit GAL4 protein thus destroying the long feedback loop involving CBF1-GFP/GAL4 and SWI5 and leading to the network "switch off" (panel C in Fig. 2.2).

Therefore, IRMA is a synthetic gene network that can be either switched ON or OFF by providing yeast with Galactose, or Glucose, respectively. Due to metabolic considerations concerning sugar-related

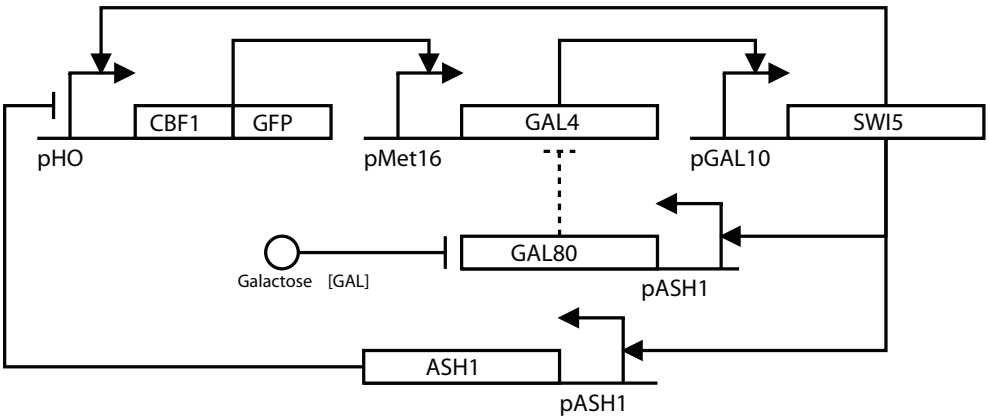

Figure 2.1: **IRMA gene circuit.** IRMA is composed of 5 genes encoding for transcription factors modulating the expression of each other. Both the transcription factors in the network and the promoters driving their expression are reported in this figure (adapted from [3]). Solid lines model transcriptional interactions, while dashed lines are meant to represent protein-protein interactions.

energy-gathering activities, yeasts will tend to consume Glucose before switching to Galactose (mathematical modeling of this phenomenon has been presented in [26]). This leads yeast to use Glucose first, even when it is found at concentration as low as 2 % in the growth medium [26]. This physiological property needs to be carefully considered when designing a microfluidic device: given this Glucose-sensitivity, if the mixing strategy has uncertainties as low as few percent points, this would invalidate the whole experiment.

From these observations, it follows that the actual input to the network is Galactose, whose presence or absence can be modeled by a simple binary variable.

IRMA's state (ON/OFF levels) can be effectively evaluated by quantifying the amount of fluorescence in the cell's nucleus. The CBF1 coding sequence had been fused to the GFP gene leading to the translation of a fusion protein that stays in the nucleus permanently. Moreover, the promoter driving the expression of CBF1-GFP gene, the HO promoter, has been found to introduce a 100 minute time-delay due to the sequential

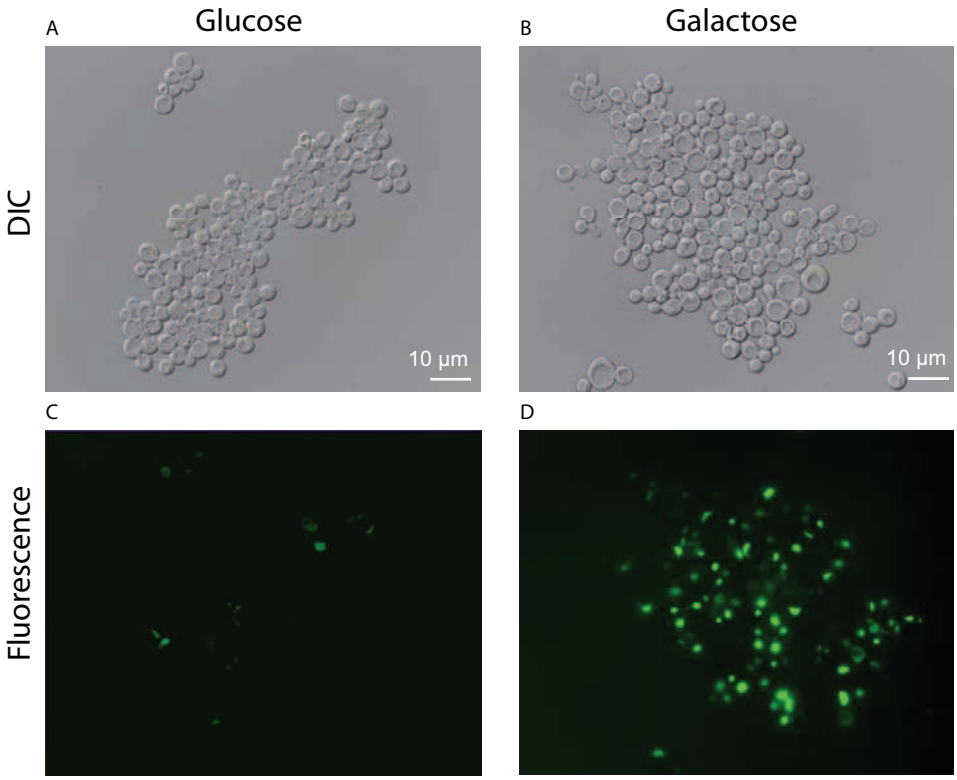

Figure 2.2: **Live imaging of IRMA cells grown in Glucose and Galactose containing medium.** Scale bar, 10 μm; 100X magnification. When grown in Glucose supplemented media yeasts only display a basal fluorescence (panel C), while, when Galactose is added to their growth conditions, the level of fluorescence increases (panel D).

recruitment of chromatin-modifying complexes (which follows binding of SWI5 and other transcription factors) [28].

All of these aspects (network architecture complexity, transcriptional time delay, sensitivity to external signals, some widely shared properties in the class of gene networks [29]) render IRMA a suitable testbed for an in-vivo control algorithm.

2.2 Mathematical model of the IRMA network

In order to capture the dynamics of the network the following set of nonlinear of delayed differential equations (DDEs) was derived in [3]:

$$\frac{dx_1}{dt} = \alpha_1 + v_1 \left(\frac{x_3^{h_1}(t-\tau)}{(k_1^{h_1} + x_3^{h_1}(t-\tau)) \cdot \left(1 + \frac{{x_5}^{h_2}}{k_2^{h_2}}\right)} \right) - d_1 x_1 \tag{2.1a}$$

$$\frac{dx_2}{dt} = \alpha_2 + v_2 \left(\frac{x_1^{h_3}}{k_3^{h_3} + x_1^{h_3}} \right) - (d_2 - \Delta(\beta_1))x_2 \tag{2.1b}$$

$$\frac{dx_3}{dt} = \alpha_3 + \widehat{v_3} \left(\frac{x_2^{h_4}}{\widehat{k_4}^{h_4} + x_2^{h_4}(1 + \frac{x_4^4}{\widehat{\gamma}^4})} \right) - d_3 x_3 \tag{2.1c}$$

$$\frac{dx_4}{dt} = \alpha_4 + v_4 \left(\frac{x_3^{h_5}}{k_5^{h_5} + x_3^{h_5}} \right) - (d_4 - \Delta(\beta_2))x_4 \tag{2.1d}$$

$$\frac{dx_5}{dt} = \alpha_5 + v_5 \left(\frac{x_3^{h_6}}{k_6^{h_6} + x_3^{h_6}} \right) - d_5 x_5 \tag{2.1e}$$

where $x_1 = [CBF1], x_2 = [GAL4], x_3 = [SWI5], x_4 = [GAL80], x_5 = [ASH1]$ are the states of the dynamical system; transcriptional regulations are modelled by Hill functions and the multiple regulation on CBF1 is modelled by a product (AND regulation). The input u is meant to be a binary variable encoding the presence ($u = 1$) or absence ($u = 0$) of Galactose.

Note that u enters non linearly in the model affecting the values of the the parameters $\hat{v}_3, \hat{k}_4$ and $\hat{\gamma}$ (also referred to as $\hat{z}_i$ parameters), which are therefore dependent on whether Galactose or Glucose are alternatively administered.

A time delay τ is present in the equation for x_1 to reflect the time delay in CBF1-GFP transcription, as outlined previously. A list of all the model parameters can be found in the table in Appendix A. The terms

$\Delta(\beta_1)$ and $\Delta(\beta_2)$ are included in the equations to capture the initial transient due to washing step required to move cells from Galactose to Glucose in the experimental setup used in [3].

Specifically, in order to carry out the medium shift (from Glucose to Galactose and vice-versa) and activate/deactivate the synthetic network, yeasts were forced to undergo a *washing step*. During this procedure Glucose medium is completely washed out and the fresh new Galactose supplemented medium is added to the cells. A transient increase in mRNA levels of GAL4 and GAL80 has been persistently observed during the experiments, thus providing evidence that this effect was NOT dependent on Galactose per se, but due to the transient stess caused by the washing step [3]. In [3] the authors claim that this effect is due to the transient deprivation of primary carbon source during the washing steps which attenuates the degradation levels of GAL4 and GAL80 mRNAs; this hypothesis seems to be supported by the fact that MET16 gene shares the same GAL4 promoter and does not seem to be affected by the same increase. We will not need to model this behavior in alternative experimental contexts. The protocol used in microfluidic devices, for example, is different from the one employed in reported experiments [3]; cells will be continually exposed to carbon sources (either Glucose or Galactose) and thus no stress neither from carbon source deprivation nor from cell handling (as suggested by Prof. Botstein - private communication) will be induced.

Moreovoer, following considerations on the behavior of $\hat{v}_3, \hat{k}_4$ and $\hat{\gamma}$ parameters, I decided to re-derive a hybrid model implementing all the observations presented so far.

2.3 Derivation of a hybrid model

One of the most interesting peculiarities of IRMA lies in the nature of the $\hat{z}_i$ input-dependent parameters introduced above. These parameters affect the transcription of SWI5 mRNA (refer to Eq. 2.2c) and are mainly intended to capture a molecular process thoroughly described in [30] (for other references on the same topic see [31–34]).

Basically the behavior of GAL10 promoter reflects the switch-like activation function exhibited by the GAL circuit, thus leading to a sudden change of kinetic parameters when yeasts are deprived of their primary carbon source (Glucose) and administered with Galactose.

Capturing this effect implies revisiting the model in Eq. 2.1; several strategies can be used to introduce a switching between the sets of values these parameters assume. Here I chose to model IRMA as a hybrid system.

Hybrid systems very often arise in the modeling of gene networks [35–37]. This is mainly due to the fact that it is quite common to observe threshold dependent and switch-like activation or inhibition functions governing the dynamics of protein-protein or protein-gene interactions.

The new model is the composition of a vector field F_1 representing IRMA's dynamics when Glucose is administered and a different vector field, F_2, capturing IRMA's behavior when Galactose is provided. The results of these modeling steps are represented in Fig. 2.3 and the new model is reported in Eqs. 2.2

As it can be seen from Fig. 2.3 transitions from F_1 to F_2 and vice versa are activated by triggering of the guard condition (based on Galactose concentration).

$$\frac{dx_1}{dt} = \alpha_1 + v_1 \left(\frac{x_3^{h_1}(t-\tau)}{(k_1^{h_1} + x_3^{h_1}(t-\tau)) \cdot \left(1 + \frac{{x_5}^{h_2}}{k_2^{h_2}}\right)} \right) - d_1 x_1 \qquad (2.2a)$$

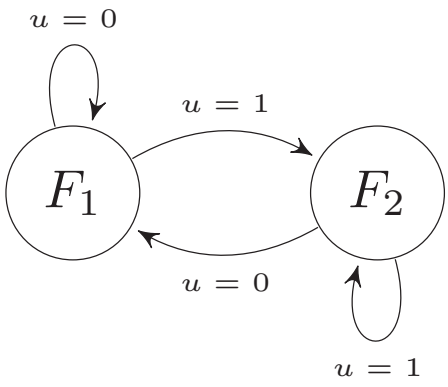

Figure 2.3: **IRMA hybrid model.**

$$\frac{dx_2}{dt} = \alpha_2 + v_2 \left(\frac{x_1^{h_3}}{k_3^{h_3} + x_1^{h_3}} \right) - d_2 x_2, \tag{2.2b}$$

$$\frac{dx_3}{dt} = \alpha_3 + \widehat{v_3} \left(\frac{x_2^{h_4}}{\widehat{k_4}^{h_4} + x_2^{h_4}(1 + \frac{x_4^4}{\widehat{\gamma}^4})} \right) - d_3 x_3 \tag{2.2c}$$

$$\frac{dx_4}{dt} = \alpha_4 + v_4 \left(\frac{x_3^{h_5}}{k_5^{h_5} + x_3^{h_5}} \right) - d_4 x_4 \tag{2.2d}$$

$$\frac{dx_5}{dt} = \alpha_5 + v_5 \left(\frac{x_3^{h_6}}{k_6^{h_6} + x_3^{h_6}} \right) - d_5 x_5 \tag{2.2e}$$

$\Delta(\beta_i)$ terms in the degradation of GAL4 (2.2b) and GAL80 (2.2d) have been deleted in agreement with previous considerations.

2.4 Qualitative analysis of the hybrid model

From a control perspective, the IRMA model is, therefore, a highly nonlinear, hybrid, time-delayed dynamical Single-Input (Galactose) Single-Output (CBF1-GFP) system of the form:

$$\dot{x} = \begin{cases} F_1(x, \mu), & \text{if } u = 0, \\ F_2(x, \hat{\mu}), & \text{if } u = 1 \end{cases}$$

where $x = [x_1\ x_2\ x_3\ x_4\ x_5]^T$, μ the vector of parameter values in Glucose ($u = 0$) and $\hat{\mu}$ the vector of parameter values in Galactose ($u = 1$).

The model dynamics during switch on and switch off phases are depicted in Fig. 2.4.

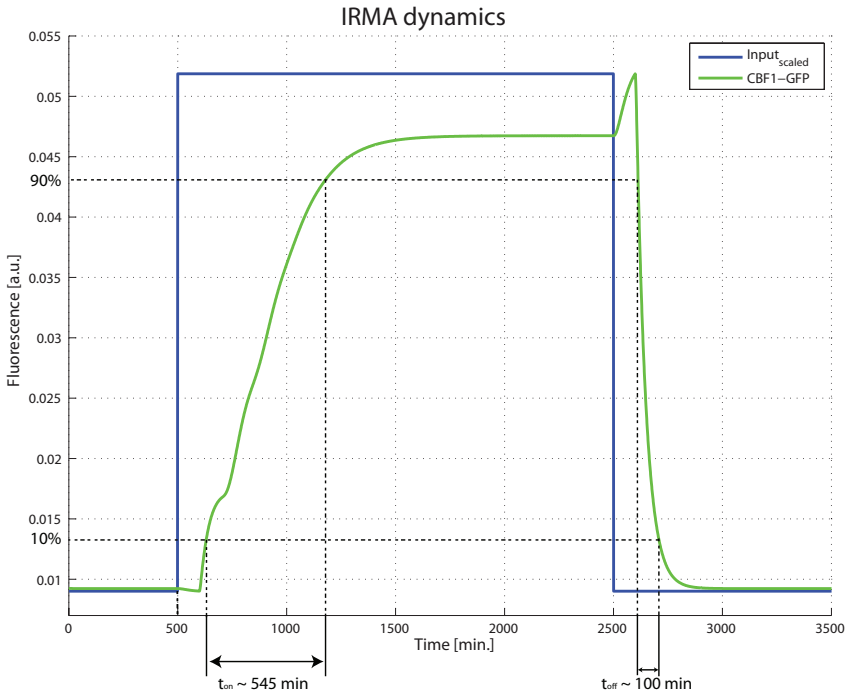

Figure 2.4: **IRMA dynamics exposed.** In this plot the switch on and switch off dynamics of IRMA have been studied by applying 2000 minute long pulse of Galactose (in blue, higher level meaning $u = 1$ and lower $u = 0$). Rising (t_{on}) and falling (t_{off}) times for fluorescence (in green) have been calculated as the time required to go from 10% to 90% ((t_{on}) and from 90% to 10% (t_{off}), of the dynamic range of the CBF1-GFP signal.

The IRMA model displays one stable steady state for each of the two vector fields; the higher steady state is almost 4-fold larger than the Glucose steady-state.

As evident from Fig. 2.4, the switch-on time is much longer than the switch-off time. This aspect should be carefully taken into consideration when designing experiments, in particular when designing signal tracking

control tests. As a matter of fact, the presence of increasing reference signal has dramatic impact on the minimum duration of the experiment, a variable that is positively correlated with the chance of technical issues to occur.

CHAPTER 3

ANALYSIS AND SYNTHESIS OF CONTROL STRATEGIES FOR IRMA

Understand it to predict it
and predict it to rule it.

Auguste Comte

Contents

Summary

The main objective of this Chapter is to address both set-point regulation and signal tracking of the concentration of CBF1 protein in a population of yeast cells carrying the IRMA synthetic network. (For a definition of set-point regulation and tracking refer to paragraph 1.2).

In the following section, I will review some of the main hurdles in controlling gene networks, and the IRMA synthetic network specifically, and I will propose suitable control strategies addressing all of these issues at once.

I will then introduce two control strategies considered to be the most general approaches to the control of hybrid systems: namely *time-dependent switching* and *state-dependent switching* [38].

In addition, I will also discuss two alternative control strategies, namely relay control and control action coding through duration/frequency/phase modulation.
Part of this work has been published in [39].

3.1 Analysis of control strategies for gene networks

Piece wise affine systems very often arise in the modeling of gene networks [35, 36]. As already outlined, this is mainly due to the threshold dependent and switch-like activation or inhibition functions governing the dynamics of protein-protein or protein-gene interactions. In the IRMA network, we observe that a switch-like transcription activation function affects the dynamics related to GAL10 promoter controlling SWI5 mRNA transcription.

Thus IRMA can be well described by a hybrid system consisting of two subsystems namely S_1 (described by F_1 vector field) and S_2 (described by F_2 vector field in Fig. 2.3). The hybrid nature of IRMA requires

developing a hybrid control strategy aimed at determining a suitable *switching sequence* between the two subsystems.

Interestingly, the problem at hand shows extended affinities with the classical problem of water warming by means of a boiler. As a matter of fact, in IRMA we can distinguish two different modes of action: S_1 represents the *OFF* mode of action and S_2 the *ON* one. In the classical water warming example, we can distinguish an *ON* state (in which water is warmed by an electrical resistance in the boiler) and an *OFF* state (in which the resistance is not powered anymore and water starts to relax at room temperature). Therefore we can take advantage both of the inner characteristics of the system at hand, and of the existence of the solution of the classical water boiler problem, to propose a suitable control strategy addressing the hybrid nature of IRMA.

Two control approaches are considered to be the most general approaches to the control of hybrid systems: namely *time-based switching* and *state-based switching* [38].

In general, time-dependent schemes are based on switching considering time alone; in state-based switching, information from the state of the system is used to decide what kind of control action to supply the system with.

Going back to the water boiler example, two different strategies can be envisioned on the basis of this classification. Time-based strategy would provide for an input that switches on the fire for a programmed amount of time and off for the cooling phase (see panel A of Fig. 3.1). Of course, if no information is used on the state of the system, the control task is likely to suffer from the limitations of a *blind* approach. In state-based switching, on the contrary, typically the states of the system are used to compute a feedback switching strategy (see panel B of Fig. 3.1). On the basis of what has been presented so far, it can be argued that time-dependent switching strategy can be interpreted as open-loop

control schemes, while state-based switching methods are classified as closed control approaches. In addition to state-based switching strategies, output-based switching methods can be used, when the state is not directly measurable and states are not observable.

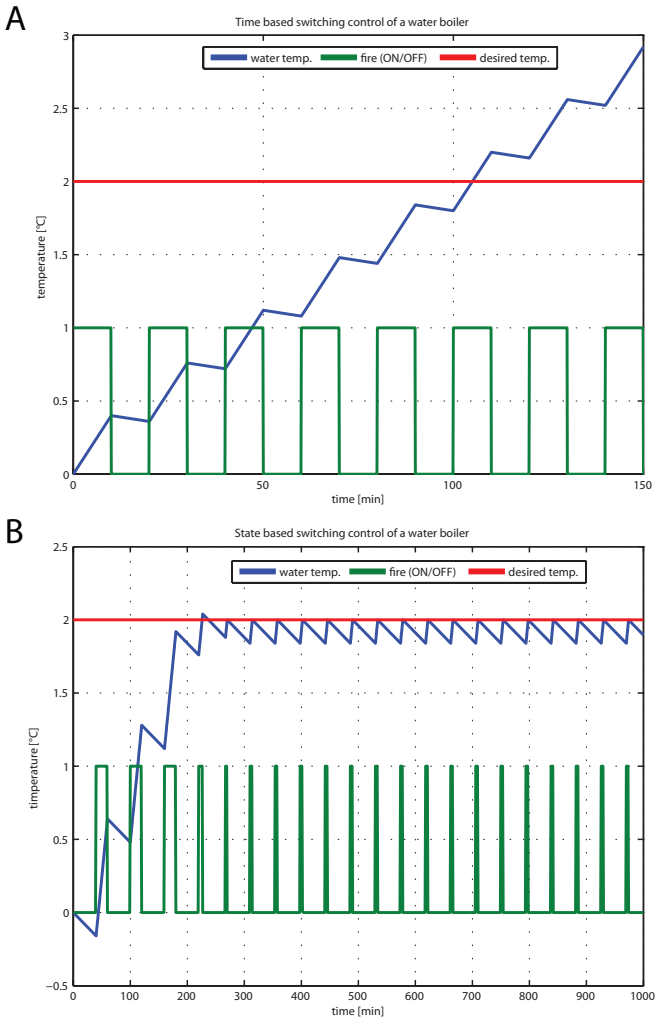

Figure 3.1: **Time-based (panel A) versus state-based (panel B) control of a water boiler.** The water boiler is a hybrid system being a composition of two vector field F_1 and F_2 meant to represent the *fire ON* and *fire OFF* modes. In both the cases the water boiler started from 0°C and was required to achieve 2°C (indicated by the red line in the plots). The state-based control scheme is able to force the actual boiler temperature (in blue in the plots) to oscillate around the set-point by switching the input signal (in green in these plots) ON (1 level) and OFF (0 level).

Therefore, as evident from the plots reported in Fig. 3.1, state-based switching control strategies usually outperform time-based techniques that, in practice, can only be used in cases of perfectly modeled processes. For what concerns IRMA, most likely the model in our hands is affected by unmodelled dynamics and strong non-linearities, therefore a state-based switching control has been chosen to accomplish both set-point and signal tracking control.

Another peculiarity of IRMA, which is shared by naturally occurring gene networks, is the presence of a time-delay affecting the transcription of CBF1, as extensively described in [3].

In the following sections, taking into account the above considerations, I will present three different control strategies which I have implemented to control expression of CBF1-GFP in a population of yeast cells carrying the IRMA network.

3.2 A control strategy based on the Smith-Predictor approach

Addressing the control problems for systems with time-delays is a major issue in control theory and several alternative strategies have been proposed to tackle it; one of the most effective and simple approaches has been proposed in 1957 by Otto Smith [40]. Even though this control scheme underwent some modifications and rearrangements in the recent years (see [41] for an extensive review) the main idea remained unchanged.

Given a system with a dead-time, the Smith Predictor scheme uses an anticipated version of the system output signal (coming out of a delay-free linearisation of the actual plant) to cancel the delay and compute the new input signal. This idea is described in Fig. 3.2. It should be noted that Smith predictor has been specifically developed with Linear

Time Invariant (LTI) systems in mind [40] and, only recently, results have been published about the applicability of such a principle to non-linear systems [42].

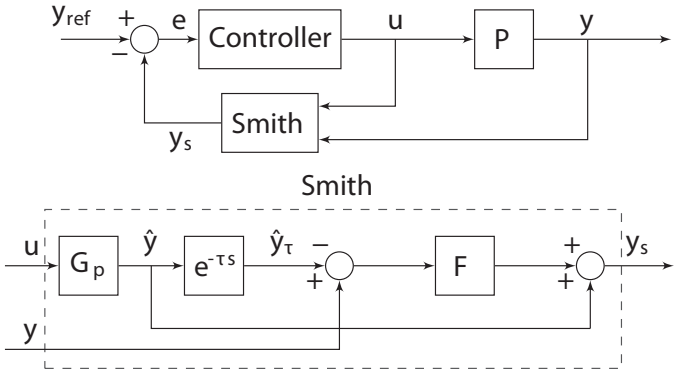

Figure 3.2: **Smith Predictor block diagram.** The upper block diagram represents the whole control scheme documented in this work while the lower block magnifies the Smith predictor control block referred to in the previous schematic. The y_{ref} signal sets the desired output y for the controlled system P. The Smith predictor block uses the input u and output y related to the actual plant P to compute an anticipated version of the y obtained by simulating the response $\hat{y}$ of a linearised time-delay deprived mathematical model of P. This signal is immediately used to assess the effectiveness of the control action by feeding it back to the first comparator that computes the error e made by the system. Moreover, the actual output y of the plant, is compared and filtered (through the F block) with a delayed (as effect of the $e^{-\tau}$ block contribution) version of the $\hat{y}$ signal to account for discrepancies between the predicted and real plant behavior.

We developed two distinct strategies based on the scheme in Fig. 3.2; both of them are output-based switching approaches, but they are characterised by different complexity levels and capabilities.

3.2.1 Relay based control

The first control scheme I envisaged mainly relies on the theory of relay control; given a pre-determined output-dependent condition (therefore this is, *de-facto*, a state-dependent control), say the scalar function $H(y)$, the controller determines at each time instant if it has to "switch"

the system on the basis of a simple boolean evaluation, obtained by comparing the value of $H(y)$ against a predetermined threshold.

This strategy fully exploits the characteristics of hybrid systems previously referred to. In particular, given the fact that F_1 represents the OFF state and F_2 the ON state we can set up a simple control algorithm having the following form (the closed loop system state space form is presented below):

$$\dot{x} = \begin{cases} F_1(x,\tau), & \text{if } H(y) > 0 \\ F_2(x,\tau), & \text{if } H(y) < 0 \end{cases} \tag{3.1}$$

With

$$H(y) = y(x) - y_{ref} \tag{3.2}$$

The rationale for this control algorithm follows from the considerations outlined above: when the $H(y)$ function, equal to the error e (for example, by taking $H(y) = y - y_{ref}$) in Fig. 3.2, is above zero the amount of CBF1-GFP produced by the system is greater than the desired one and thus the controller is instructed to switch off the system by interrupting Galactose administration. On the other hand, when $H(y)$ is smaller than zero, the amount of CBF1-GFP produced by yeast is smaller than the desired one and the controller switches on IRMA, by delivering Galactose to yeasts. Thus we introduce a switching hyperplane in the phase space

$$H(y) = 0 \tag{3.3}$$

This situation is depicted in Fig. 3.3.

The switching surface can then be defined as

$$S := \{x \in \mathbb{R}^5 : H(y) = 0\} \tag{3.4}$$

The control scheme resulting from these considerations is presented in Fig. 3.4

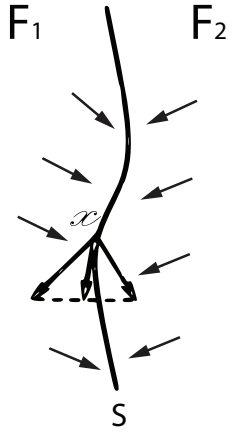

Figure 3.3: **Switching hyperplane introduced by the relay control action.** Arrows represent hypothetical trajectories of the actual plant in response to the control action; F_1 and F_2 vector fields would be alternatively activated as a result of the change of sign of the computed error. The switching surface S corresponds to $H(y) = 0$

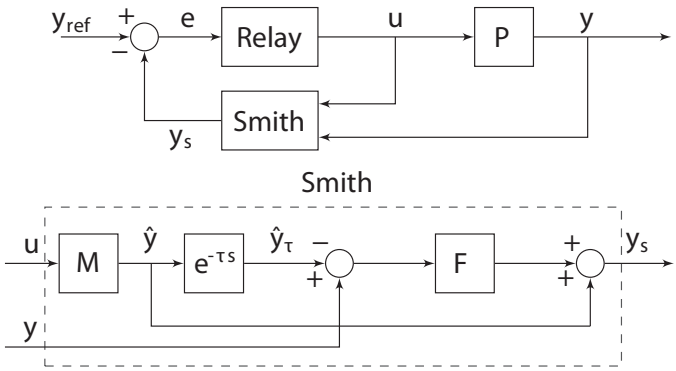

Figure 3.4: **Smith Predictor control scheme coupled to a relay controller**. In this scheme the control action u is computed by the Relay block that takes as input the error signal e and, on the basis of its sign fires a binary signal: 1 (Galactose) is the error is positive and 0 (Glucose) when $e \leq 0$. The block M contains the non-linear model (deprived of the delay) of IRMA used for the delay cancellation.

Simulation results of the proposed control strategy for set-point regulation or tracking are presented in Fig. 3.5 and in Table 3.1. It can be observed that the controller guarantees excellent tracking and regulation performance but, as a trade-off, fast switching is required which might

exceed what is physically admissible in the real plant.

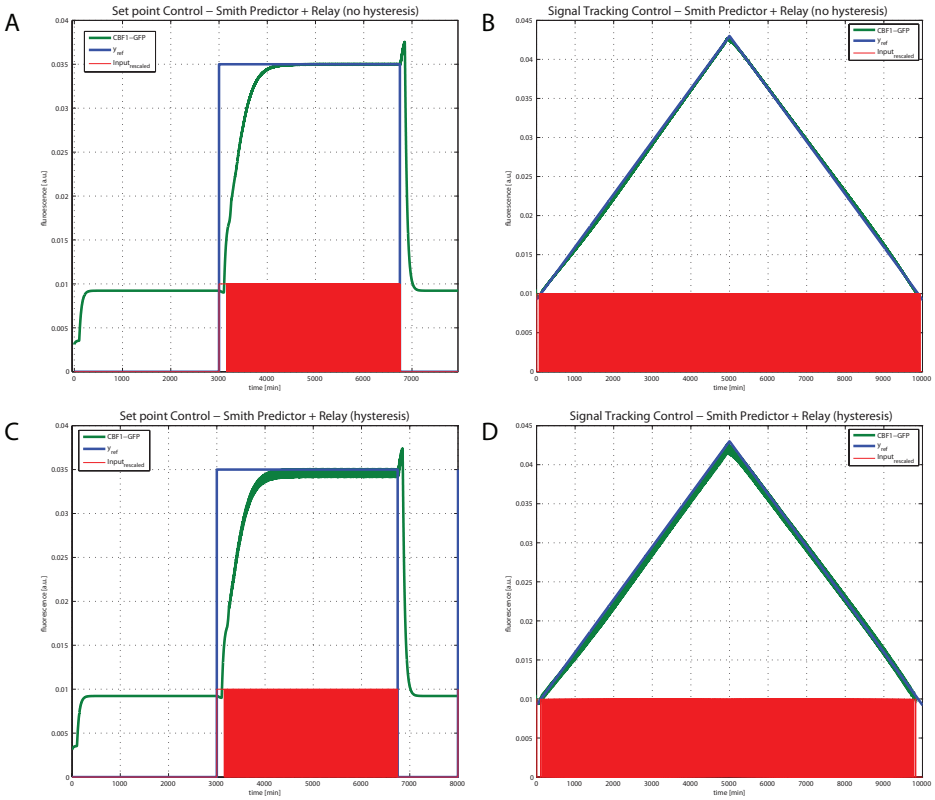

Figure 3.5: **Comparison of set point and signal tracking control with relay and relay/hysteresis control algorithms** Hysteresis leads to slightly poorer quality in the control experiment but attenuates significantly the number of switchings (and thus control power) required to accomplish the assigned task.

The control strategy described so far is characterised by a simple design, which directly translates into a light weight computational effort required for the its implementation. Both regulation and tracking goals can be achieved with this approach.

Unfortunately the main drawback of this approach is the fast switching dynamics between F_1 and F_2 generated by the controller, when the system's states hit the switching surface. This problem is often referred to as *chattering* and leads to high power loss due to frequent swichings

[43]. This is a very common problem arising in the field of relay feedback and it is known that it can be limited by introducing a hysteresis based controller. This design will be discussed in the next paragraph.

3.2.2 Relay controller with hysteresis

By introducing a hysteresis element to the relay controller, we add a new non-linearity to our system but, at the same time, this can be successfully used to overcome the issues arising from chattering.

Hysteresis allows to decouple switch on and switch off thresholds as represented in Fig. 3.6.

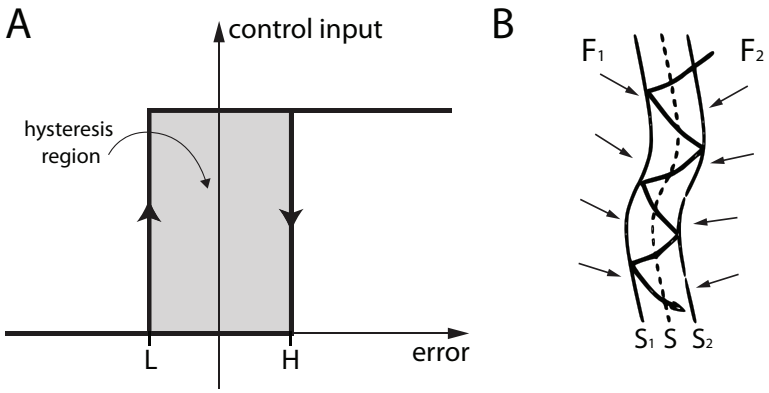

Figure 3.6: **Hysteretic behavior.** In panel A the function that relates the input (error signal e) and the output (control signal u) is reported. The upper and lower thresholds (L and H) are decoupled and thus, if the system is the in lower state (0, switched off) it switches to the ON state only when it reaches the upper threshold (H). On the other hand, if the system is in the ON state it will only switch to the OFF state when the lower threshold will be reached (L). The *hysteresis region* (in light grey in this figure) is associated to the *memory effect* of hysteresis-based devices and its width directly influences its effect on chattering prevention. In panel B the effect on the switching hyperplane is presented: the switching frequency of the system's state decreases as a function of the difference between L and H.

In this configuration, since the system has two thresholds, whose location is a design variable of the control scheme, there is room for mitigating the effects of chattering, theoretically at will: in practice eliminating any form of chattering would result in no switching at all, which renders the

control scheme itself useless.

As a rule of thumb, the greater is the difference between the upper threshold and the lower one, the smaller the effects of chattering will be (and the lower the quality of the control).

The simulation results due to the addition of an hysteresis region ($L = 5E - 5$ and $H = 5E - 4$) are shown in Fig. 3.5 (panels C and D) and in Table 3.1.

The control strategy outlined above dramatically improves on the relay based control strategy alone, but keeps suffering from the over-actuating control action. Moreover, this approach shows a quite limited number of degrees of freedom since the only design variable we can change is the amplitude of the hysteresis region. A finer control can be achieved by substituting the relay based control strategy (which is a typical comparison-dependent approach) with a *control action encoding* strategy. This approach will be the topic of the next section.

3.2.3 Pulse modulation control

Signal modulation based control has been extensively exploited in power electronics systems [44]. Theoretical principles behind these control strategies have been borrowed from telecommunication engineering and are mainly based on the real time modulation of the properties of a signal (sine wave, square wave etc).

Interestingly, a recent contribution to the field [45] showed that prokaryotes encode signals using stochastic pulse frequency modulation through a compact regulatory architecture.

Here, we recall that the synthetic network at hand behaves like a switch, thus we need a control signal taking only these two values. We argue that the best choice (even in terms of computational efficiency) is to select a square-wave output for the control scheme. The carrier signal is typically a square wave characterised by an amplitude, period

or frequency, and duty cycle (percentage of the period the signal is ON). For what concerns the signal modulation, there exist three types of analog modulations:

1. Pulse Amplitude Modulation (PAM): given the carrier signal, this modulation acts on the maximum level of the output signal by increasing or lowering it as the signal to be encoded varies over time;

2. Pulse Frequency Modulation (PFM): given the carrier signal, with a prescribed amplitude and duty cycle, this modulation acts on the frequency of the output signal by increasing or lowering it as the signal to be encoded increases or decreases over time;

3. Pulse Width Modulation (PWM): given the carrier signal, with a prescribed period (frequency), this modulation acts on the duty cycle of the output signal by increasing or lowering it as the signal to be encoded varies over time;

The modulation I chose for the control scheme is PWM, since it appeared the most adequate for the problem at hand as it has been traditionally used to control uncertain and noisy systems in power electronics.

In Pulse Width Modulation we distinguish three signals:

1. an input signal: the signal to be encoded;

2. a reference (or carrier) signal: the signal used for comparison;

3. an output signal: the signal being returned as the output of the PWM block.

The processing is quite simple and involves few steps represented in Fig. 3.7: the input signal (internal signal in Fig. 3.7) is compared with

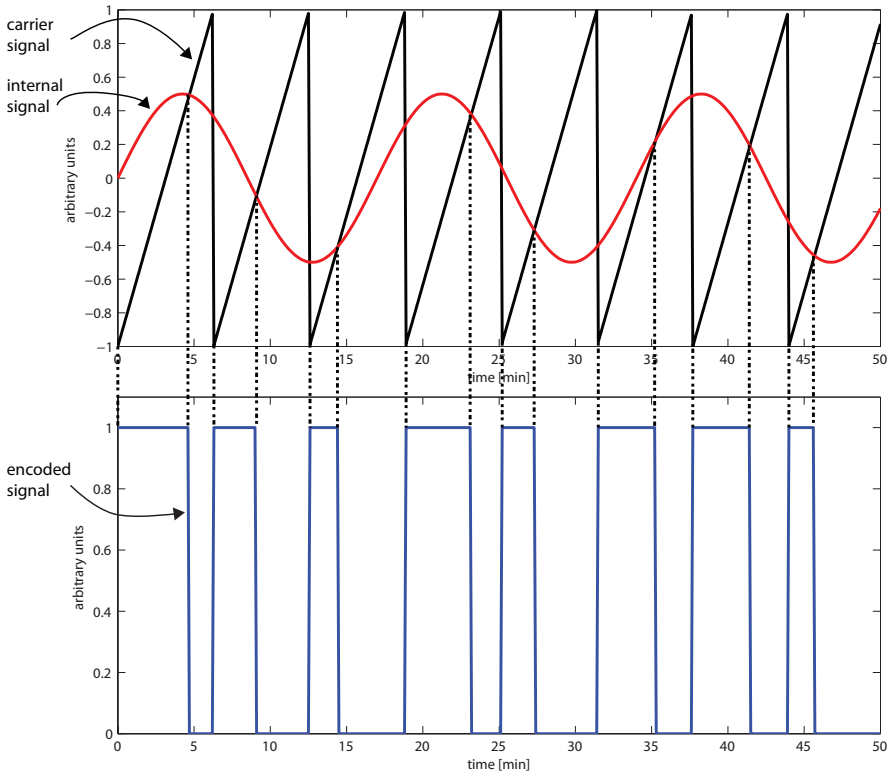

Figure 3.7: **Pulse Width Modulation exposed.** The internal and carrier signals are compared against each other and, whenever the first is smaller than the second one, the output of the PWM block is set to 1, otherwise it is 0.

the carrier signal in each period and whenever it is bigger than the carrier, the output signal is set to the upper state, otherwise it is equal to zero.

This control scheme entails three design variables: *(a)* the carrier signal to be used (triangular wave, sine wave etc), *(b)* its amplitude and *(c)* frequency. We chose as reference signal, a sawtooth wave whose amplitude has been set to 10% of the CBF1 level at steady state in Glucose (it should be noted that the amplitude of the carrier signal directly influences the tolerated error: the higher the amplitude, the higher the tolerated error, and viceversa) and whose frequency has been chosen so that:

$$f_{carrier} = \frac{1}{T_{carrier}} = \frac{1}{2\ t_{off}} \tag{3.5}$$

where t_{off} is IRMA's switch off time as defined in 2.4. Moreover, in order to allow only one switch on and switch off in the same period ($5\ min$) of the carried signal we implemented a strategy based on a flip-flop controlled by a set signal (given at the beginning of each period) and reset by the following switch off. The control scheme described so far is depicted in Fig. 3.8.

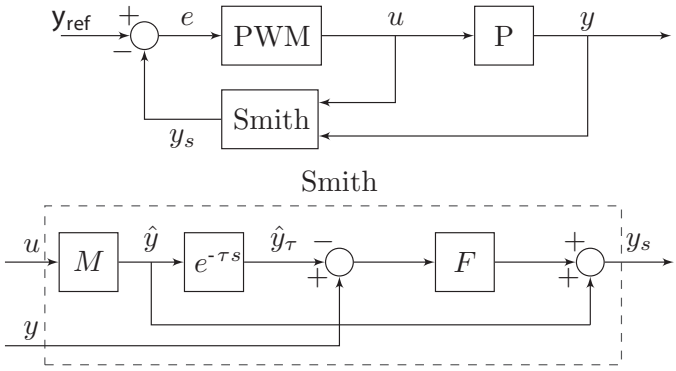

Figure 3.8: **Smith Predictor control scheme coupled to a PWM encoder**. In this scheme the control action u is computed by the PWM block that compares the error signal e with the internal sawtooth and puts in output a signal encoded as depicted in Fig. 3.7.

The results of this control scheme are presented in Fig. 3.11(panels A and C) and in Table 3.1. Both regulation and tracking goals can be achieved by employing the strategy outlined above.

The strategy based on PWM is powerful and allows the encoding of theoretically any kind of real-valued signals. This scheme inherits most of the potentialities of the strategies outlined above (namely the one based on relay) while improving on the previous ones when it comes to the evaluation of levels of control action. By combining a fine tuning of the $f_{carrier}$ and of the amplitude of the carrier signal it is possible to minimise the side effects of high-frequency switchings, at the same time, obtaining reasonable performances in terms of signal tracking.

The most notable strength of this approach lies in the virtually infinite encoding capability of the PWM scheme allowing the mapping of any number in the set of real numbers in a control action taking integer values in the set $\{0, 1\}$.

A weakness of this scheme is that it is slightly more complex than the relay-based one, since it involves more than a simple comparison. The section devoted to the flip-flop mechanism adds some degree of complexity, minor in fact, to the whole control algorithm.

It should be noted that, in the configuration presented above, the flexibility of the PWM encoding is only partially exploited since the error is directly encoded in the control action. This scheme resembles a proportional-coded controller, which is a classic control approach, and thus puts in evidence some direct extensions of this approach that will be discussed in the next paragraph.

3.2.4 Proportional/Integral-Pulse Width Modulation control strategy

Given the good results achieved with the previous configuration, we tested the hypothesis that, by using a linear combination of proportional and integral measures on the error, we can improve on the previous approach.

Analysing the issues observed in the previous experiments, I noticed the presence of a residual steady-state error preventing the control system to perform as expected.

To this end, I slightly modified the control scheme presented in Fig. 3.8 in order to include a PI controller upstream of the PWM modulator.

Proportional-Integral-Derivative (PID) control [46] is a widely used strategy employed in output feedback loop controllers. The underlying idea is very simple: the bigger the error, the stronger the control action the system needs to minimise it. The approach used by a simple PID

controller is, therefore, to provide the process to be controlled with a control input u that is a linear combination of three quantities: the instantaneous error $e(t)$, its integral over time $\int_0^t e(t)\ dt$ and the time derivative of $e(t)$, $\frac{de(t)}{dt}$ (see Fig. 3.9).

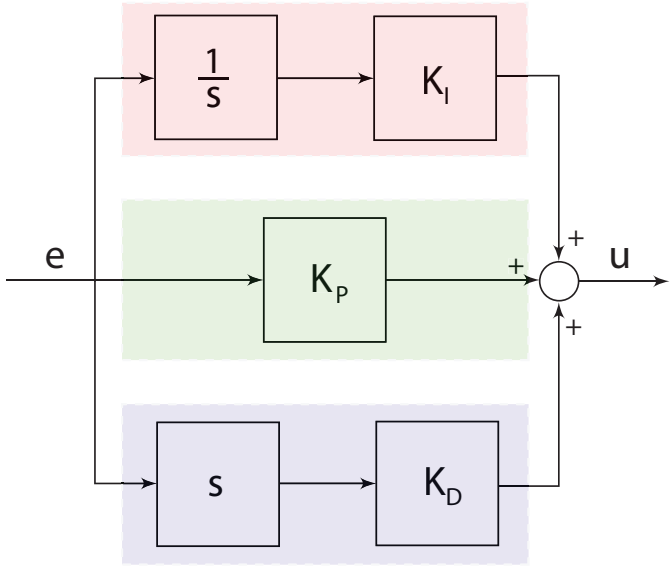

Figure 3.9: **Proportional-Integral-Derivative controller.** In this scheme the error signal e is used by *integral* (red box), the *proportional* (green box) and *derivative* branches to compute the corresponding components of the linear combination. The three values are summed up in the rightmost sum block to obtain the final control input. While in the *proportional* section the error is just multiplied by the K_I gain, in the the upper and lower arms of the scheme it is integrated (through the $\frac{1}{s}$ block) and derived (s block) respectively.

Therefore u will turn out to be equal to:

$$u(t) = K_P \cdot e(t) + K_I \cdot \int_0^t e(t)\ dt + K_D \cdot \frac{de(t)}{dt} \tag{3.6}$$

where K_P, K_I and K_D are referred to as proportionality, integral and derivative gains respectively. These are the actual *weights* of each of the terms involved in the final summation: alternative methods have been proposed to find optimal values for these gains ranging from empir-

ical manual tuning, to Ziegler-Nichols, Cohen-Coon and Astrom methods [46].

In principle, proportional control is the first and simplest to be implemented; unfortunately, it has been shown to be affected by problems like steady state errors. This means that it cannot force some types of systems to achieve exactly the desired level; this results in a finite difference between the reference and actual output signal at steady state. In order to address this problem, integral action is introduced to enforce the control when an error persists for a long time (as in the case of steady state errors). Unfortunately, this control action suffers from other issues like the so called *windup phenomenon* (discussed in Chapter 5) in which control input value increases dramatically just because of a limitation of the actuator or some other saturation effects. Whereas the integral action constitutes the "memory" of the controller, the derivative is usually meant to provide the control action with an information about the "future trend"of the error. As stated above we focused on a PI controller.

Derivative control action could also have been used, in order to introduce, in the computation of the control action, a *predictive* information: unfortunately the presence of high noise levels (as often happens in the field of biology) prevents its use.

The PI-PWM controller is described in Fig. 3.10: the error signal e coming out of the first comparator undergoes a preliminary processing in which both integral and proportional action are computed.

Fig. 3.11 and Table 3.1 provide an overview of the performances of the Smith-Predictor based configuration performances.

Given the results displayed by this approach, I investigated whether they were retained even in the case of quickly changing reference signals. As a matter of fact, from theory we know that Smith Predictor is best suited for set-point experiments and it should not be expected to perform similarly in complex signal tracking contexts. In order to test the

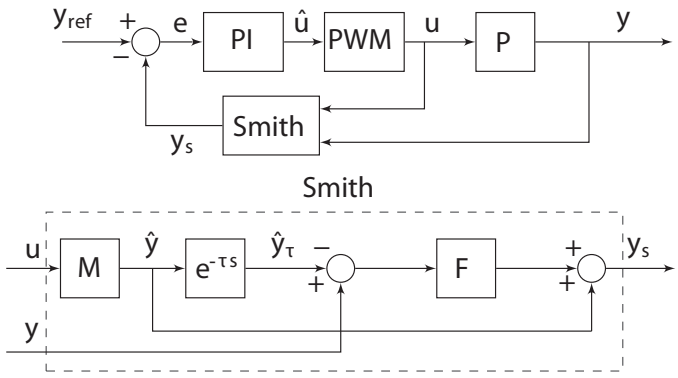

Figure 3.10: **Smith Predictor control scheme coupled to a PI controller and a PWM encoder**. In this scheme we referred to the actual living plant, namely yeast carrying the IRMA network, as P. Whereas M is the non-linear model of the network decoupled from the delay contribution $e^{-\tau s}$. The block PI is the controller while PWM represents the signal modulating component. The F block is the low-pass filter used to reduce high frequency noise (in our setup we empirically chose $F = \frac{1}{1+20s}$).

feasibility of meaningful signal regulation experiments in-vivo, we first set the maximum experiment duration to $t = 2000\ min$ (this choice has been made to reflect intrinsic limitations of the platform we introduce in Chapter 4). At this point we designed two experiments: in the first one (Fig. 3.12 panel A) a triangular wave has been used as reference signal while, in the second (Fig. 3.12 panel B) a trapezoidal waveform has been employed for the same purpose.

In-silico experiments presented in Fig. 3.12 allow us to conclude that, while Smith-Predictor based configurations are designed to accomplish set-point regulation, conditions can be found in which these same strategies attain competitive results in signal regulation control problems.

Moreover the PI-PWM control strategy fully exploits the potentialities of the PWM-based approaches. By introducing a contribution dependent on the history of the error signal (namely the integral one), the control algorithm minimises the tracking error even in the case that it is limited but persistent (also referred to as steady-state errors). Remark-

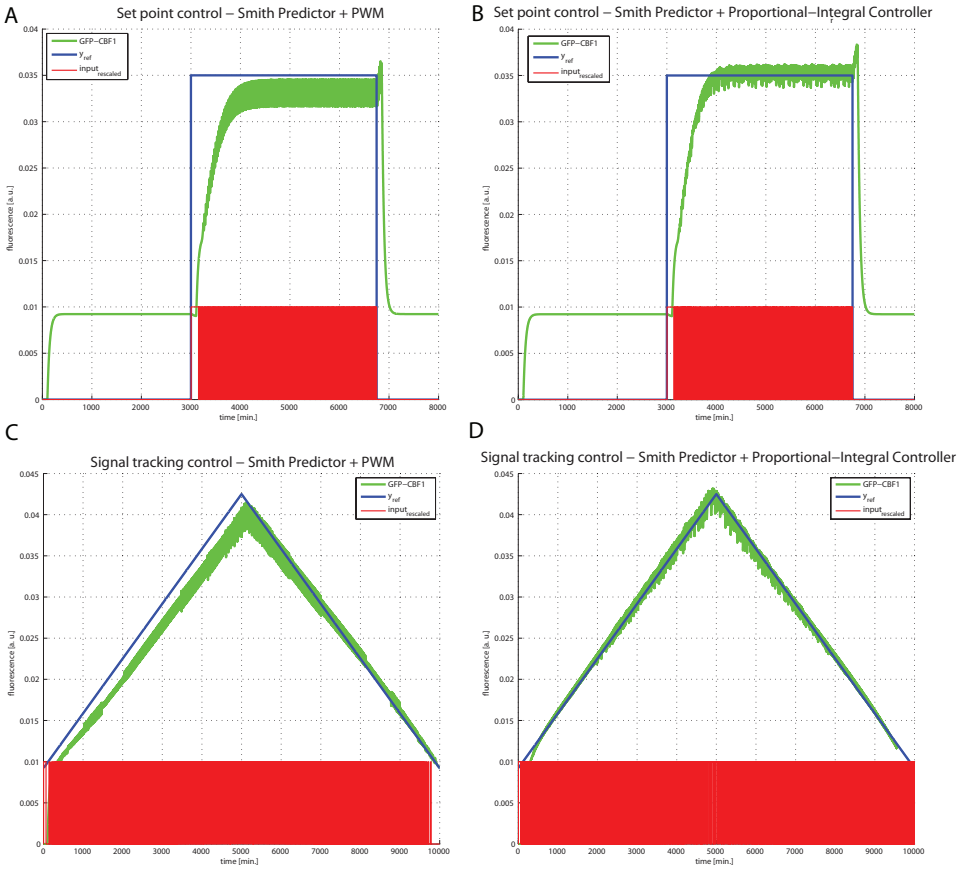

Figure 3.11: **Comparison of set point and signal tracking control with Smith Predictor + PWM and Smith Predictor +PI/PWM strategies** In panel A and C the performances of the Smith Predictor + PWM and Smith Predictor + PI/PWM strategies can be observed. The first control scheme forces the system pseudo-oscillate around a mean value which is different from the desired one. This issue can be effectively addressed by using a PI controller (P=172.5 I=2.1, these parameters were obtained by comparing Ziegler-Nichols, CHR, ITAE and Cohen-Coon methods in simulation; the latter resulted to be the best among the tested ones) as shown in panels B and D. The same consideration applies to the case of signal tracking control experiments: the integral contribution allows improved the quality of the control by introducing an memory term for the error that, while accumulating, enforces the $\hat{u}$ signal to correct this phenomenon.

ably, even though this is probably the most promising control approach we have analysed, we need to bear in mind that some issues still need to be considered.

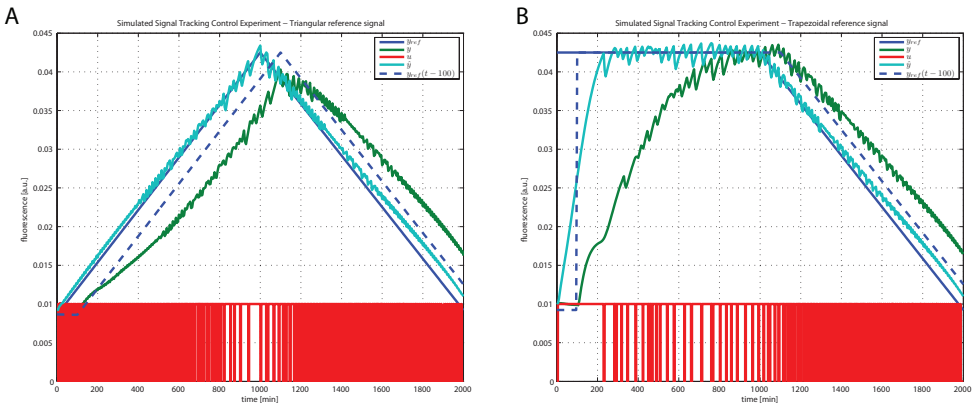

Figure 3.12: **Signal tracking control for triangular (panel A) and trapezoidal (panel B) reference signals.** The result of signal tracking control are reported for two reference signals: the solid blue line represent the signal to be followed while the green signal is the actual system output. Although the performances can be improved, interesting information can be obtained by observing the agreement between the reference signals and the predicted system's output $\hat{y}$ (light blue signal in the plot). This tells us the system is tightly controlling the un-delayed model. A dual perspective can be gained by observing the matching between the delayed reference signal ($y_{ref}(t-100)$) and the actual system's output (y).

In particular, we are assuming that all of the parameters in our model of IRMA are constant (which is a quite unrealistic assumption, since biological processes are known to undergo major modifications in response to varying boundary conditions just like temperature or PH. Model Predictive or Adaptive Control strategies can be foreseen to minimise these phenomena). We are also assuming that the parameters' values are exact, thus no room is left for parameters' uncertainty.

Given the promising results obtained by this approach, we decided to choose the PI-PWM approach for the in-vivo control strategy. Nevertheless, it has not escaped to us that other solutions, likely to be more optimal, can be found in control theory, when it comes to signal tracking. Nevertheless our control design was heavily informed by the many practical constraints for its implementation. Therefore we decided to choose this control approach as an acceptable trade-off between performance and simplicity of implementation.

Control strategy	ITAE S.P.	ITAE S.T.	Switch S.P.	Switch S.T.
Relay (no hysteresis)	311525,36	310686,77	19	69
Relay (hysteresis)	314698,69	311162,03	18	59
PWM	162588,52	304345,84	18	142
PI+PWM	153602,16	500921,22	66	206
SP+Relay (no hysteresis)	141165,07	856,49	2809	463
SP+Relay (hysteresis)	18228,665	149185,823	1553	622
SP+PWM	67095,21	1046,08	1142	290
SP+PI+PWM	16929,75	528,95	533	190

Table 3.1: **Summary of the performances of the proposed control strategies.** The Integral Time Absolute Error (ITAE) (defined as ITAE $= \int_0^T t|e(t)|dt$ where T is the current time instant [47]) has been compared for the set point (S.P.) and signal tracking (S.T.) experiments previously presented (Fig. 3.11). Switch S.P. and Switch S.T. represent the number of switches of each strategy in the two experimental contexts I considered. Smith Predictor (SP) based control schemes have been grouped together. From this result it is evident that, Smith Predictor based solutions, especially SP+PWM and SP+PI+PWM configurations, are able to conjugate lower ITAE levels with fewer actuation switchings, thus leading to a good compromise of the two measures.

Moreover, bearing in mind the potential impact of model mismatches and unmodelled dynamics on the overall quality of the control scheme, we also designed two simpler alternative strategies meant to be model insensitive.

We, thus, modified the above schemes by excluding the contribution of the model to the control input (i.e. by removing the Smith Predictor); specifically, in one strategy *(a)*, we used the simple relay control approach; in the second of these control algorithms *(b)* we used a PI/PWM controller. We will describe both in the following sections.

3.3 Relay-based control

With the objective of obtaining the simplest possible control scheme, we decided to implement a relay based algorithm, shown in Fig. 3.13.

The occurrence of oscillations and other undesirable phenomena shown

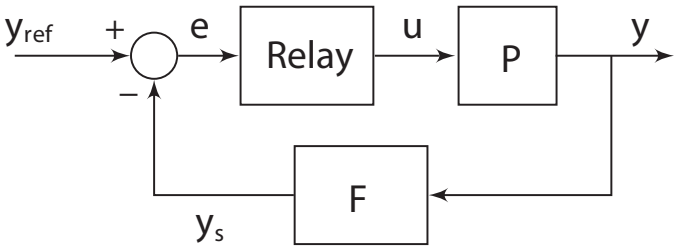

Figure 3.13: **Relay based control scheme**. In this scheme a simple relay block is used to compute the input u that directly controls the plant P. This block works as previously outlined and represents the core of one of the simplest control algorithm for a dynamical system. The output of the system is filtered through the F filter in order to get rid of high-frequency components, likely to be due to experimental noise.

in Fig. 3.14 are expected as the controller is now unable to compensate the presence of delays in the plant. Nevertheless, the simplification of the control approach due to the lack of a Smith predictor makes it appealing for the in-vivo implementation that will be carried out in the next Chapter.

3.4 Proportional-Integral/Pulse Width Modulation control

In order to avoid any dependence of our control strategy on the model, we designed the controller in Fig. 3.15.

In this case the PI controller is meant to avoid steady state error, while the PWM is used to encode the control action. No specific solution is present to address the delay and thus a computational evaluation of the actual performances of the algorithm may be hard to be extrapolated from in-silico experiments (see Fig. 3.16). Again we see the presence of a highly oscillatory closed-loop behaviour which severely affects the performance of the control scheme.

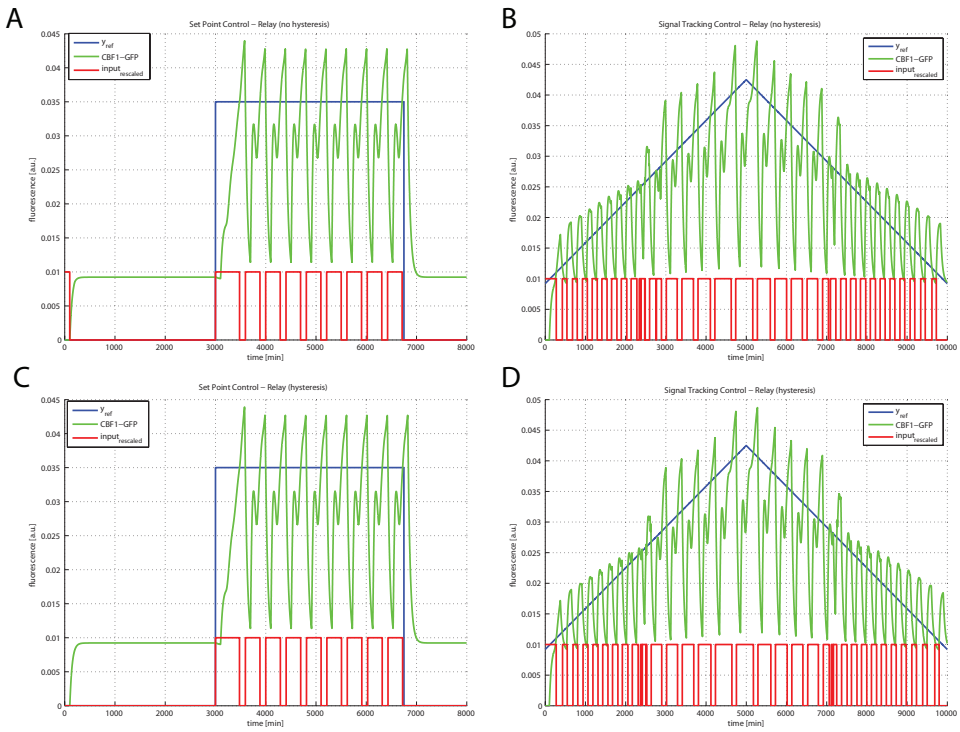

Figure 3.14: **Comparison of set point and signal tracking control with Relay w/o hysteresis.** In panels A and C the performances in the set point experiments are presented while, in panels B and D signal tracking results are reported. Upper and lower threshold for hysteresis region have been set to $5E-4$ and $5E-5$ respectively. While the number of switchings is very low, these strategies are not expected to perform adequately in-silico, in presence of a time-delay (whose impact is clearly visible in the oscillations observed in these plots).

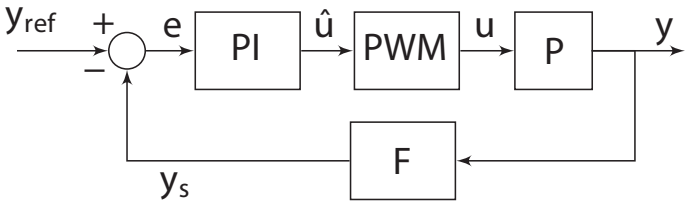

Figure 3.15: **PI/PWM control configuration**. In this scheme a PI controller is used to modulate the error signal e; the output of the PI block is used by the PWM encoder to directly control the plant P. The output of the system is filtered through the F filter in order to get rid of high-frequency components, likely to be due to experimental noise.

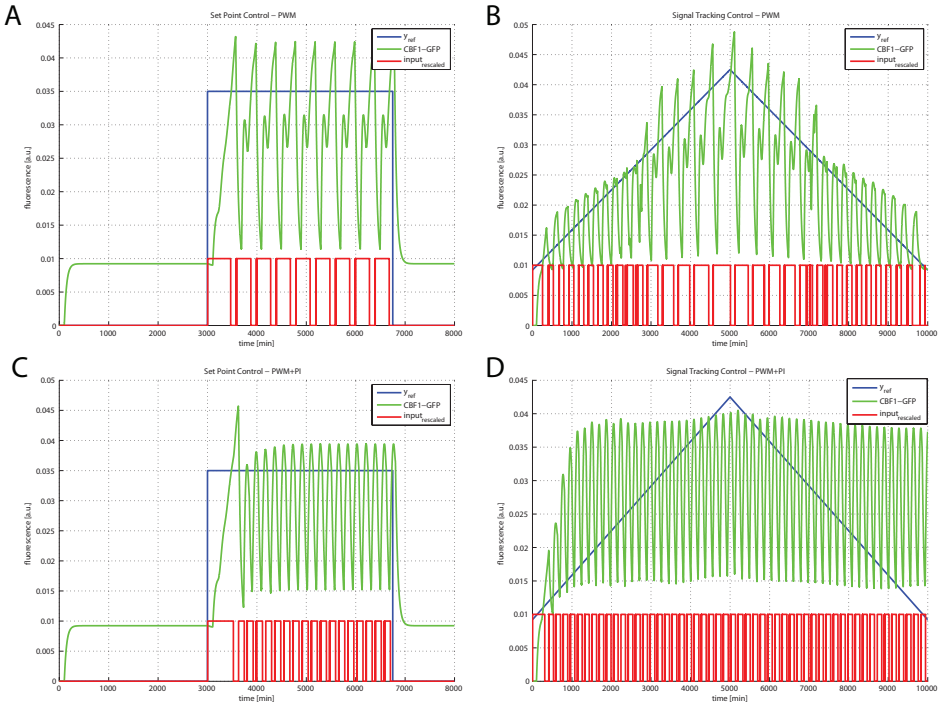

Figure 3.16: **Comparison of set point and signal tracking control with PWM and PI+PWM control schemes.** In panels A and C the performances in the set point experiments are presented while, in panels B and D signal tracking results are reported. PI and PWM parameters remain unchanged compared to the analogous cases of Smith-Predictor based control schemes. The impact of time-delay dramatically affects the quality of the control experiment that can be obtained by applying these strategies.

CHAPTER 4

IMPLEMENTATION OF CONTROL STRATEGIES

Progress often occurs at
the boundaries of disciplines.

Nigel Shadbolt

Contents

Summary

In this chapter I will present a detailed description of the implementation of the control strategies described in Chapter 3. A suitable design of a technological platform is presented to address theoretical and practical constraints arising in this context.

Being at the heart of the whole platform, microfluidic devices design, testing and fabrication is introduced. Strategies to compound concentration modulation in yeasts' extracellular environment, as well as microscopy images', processing will be provided to complement the argumentation.

4.1 Closed loop control platform design and development

The path leading to the development of in-vivo control experiments passes through the design and implementation of a suitable technological platform satisfying the theoretical and practical requirements. These constraints have been reported here for the sake of clarity:

Theoretical constraints:

1. *Adaptation avoidance*: adaptation often occurs in biological systems and allows living organisms to respond adequately to changing environment [13, 48]. In our case, a frequent switching between the two inducer compounds (Galactose and Glucose) would likely result in cells adapting to this situation and choosing to adopt energetically convenient strategies (e.g. not

using Galactose when Glucose is not present as long as the latter is provided within a short time frame). This phenomenon has not been modeled in our control strategy and would thus introduce dangerous deviations from the expected behaviors (invalidating the model-based control scheme);

2. *Optimal sampling time selection*: as evident, we are interested in not introducing any distortion in the *wild type* behavior of IRMA cells. Therefore preventing any stressogenic stimuli is of primary importance. The choice of the sampling interval, which involves exposure to strong light sources, directly impacts the stress level induced on cells [49]. This is a typical engineering trade-off: frequent sampling would result in high time-resolution and high quality control, but would induce phototoxicity/photo-bleaching phenomena [50]. These should be avoided in order to obtain the best possible experimental conditions;

3. *Binary input*: the available model allows us to elicit IRMA's dynamics with a single input being intrinsically binary in nature (Galactose presence). This means that the cellular behaviors for intermediate concentrations of Glucose and Galactose cannot be directly predicted from the model and would dramatically affect the quality of control. Moreover, since Glucose is yeasts' primary carbon source, they would tend to consume it before metabolising Galactose as mentioned in previous Chapters. For this reason we need a means to perfectly control the type and concentration of sugars in the cell growing medium; this method should be very precise and reliable in order not to introduce any uncertainty in the input (i.e. Glucose or Galactose) provided to the cells.

Technological constraints:

1. *Cell trapping and medium delivery*: an effective and efficient way to trap cells and get them exposed to a controlled time-varying concentration of an inducer molecule is required to carry out in-vivo control experiments. As previously noted, microfluidics is the technology of choice to address these issues efficiently;

2. *Long-term experiments ($t > 24$ h)*: a strategy needs to be devised to allow experiments to last for time periods that exceed the 24 hours. This constraint is directly imposed by the transcriptional dynamics we observed for IRMA: as a matter of fact $\tau_{on} > 500$ min does constitute a *limiting factor* towards the accomplishment of reasonably short experiments. Imaging technologies, and time-lapse microscopy in particular, are able address these issues;

3. *Automatic quantification of fluoresce in single-cells*: the control algorithm needs a reliable estimate of the cell fluorescence within the cell population, since this is the variable to be controlled. This can be achieved via the development of image analysis algorithms. In yeast cells, image analysis is simplified by the following observations: *(a)* yeast cells have usually a medium/low-eccentricity ellipsoidal shape; *(b)* the CBF1-GFP protein is retained within the nucleus, thus being spatially well concentrated and hence quite bright;

4. *Computation of the control action in real-time*: the control algorithms presented in Chapter 3 need to be translated in a real-time (RT) control strategy. Several alternative approaches can be envisioned for this task, ranging from automatic code generation from the MATLAB Simulink proto-

types to Finite State Automata (FSA) implementations. This task displays an intrinsic trade-off between the efforts required to achieve the objective (i.e. to easily obtain a RT implementation of the proposed algorithm) and the ability to efficiently modify and manipulate the implementation to adapt it to new hypotheses;

5. *Translation of the computed input signal in an actuation*: an interface needs to be designed that translates the input signal generated by the control algorithm into a concentration of Glucose or Galactose in the cell growing medium. In terms of the microfluidic device, the actuation will consist in a change of the medium flowing in the microfludics device (i.e. Glucose-medium or Galactose-medium). Therefore, we need to propose a strategy that takes advantage of a pressure generation system to finely regulate the flow within the microfludics device. It should be noted that any actuation strategy should consider the potentially harmful impact of shear-stress and similar phenomena on cells [51]. A range of solutions have been proposed in literature to this challenging issue. They include perfusion systems [52], picoliter syringe pumps [53] and hydrostatic pressure generation equipments [26]. All these solutions are characterised by different levels of expensiveness/easiness of implementation.

In order to address these issues, I designed and developed the closed loop control platform that depicted in Fig. 4.1.

The information flow in Fig. 4.1 outlines the working principles of this platform: the control scheme, in the form of a Finite State Automaton, is implemented in a computer, which actuates a fluorescence microscope capturing at 5 min intervals images both in the bright field (BF) and in fluorescence of the trapped yeast cells. An image segmentation pipeline

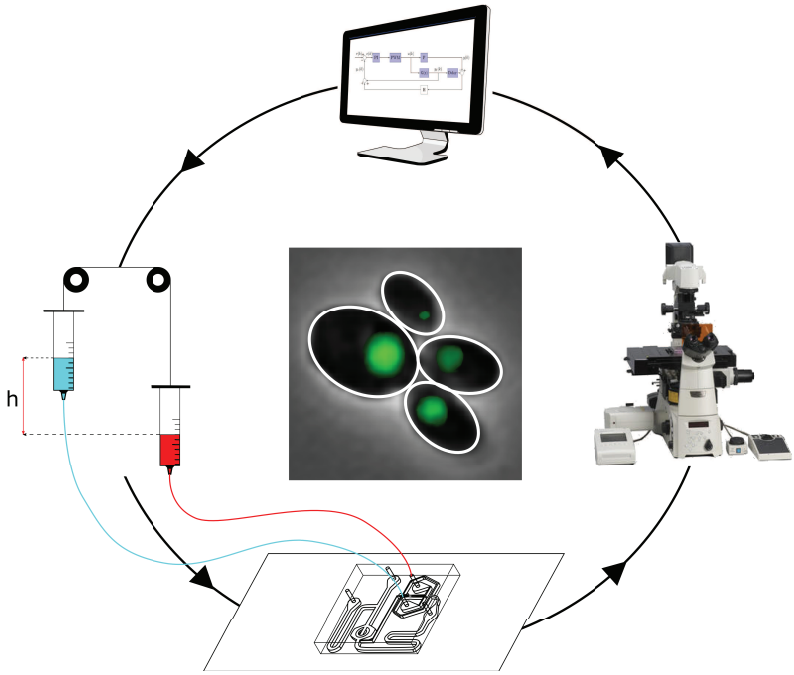

Figure 4.1: **Overview of the platform designed for the implementation of control experiments.** Both the main components of this platform, as well as the information flow, are highlighted in this scheme.

is able to locate cells using BF images and to return an average measure of GFP fluorescence within the cells at population level at each time point. The control algorithm, then, compares the measured GFP fluorescence with the desired level and computes the new control input to be supplied to the cells. The control input is then translated into an actuation strategy that takes advantage of a re-engineered Dial-A-Wave hydrostatic pressure generator. Computer-controlled stepper motors can be programmed to translate the binary control input in heights of syringes filled with Galactose- and Glucose-medium. Relative differences between the free surfaces of the two fluids will generate a hydrostatic pressure that will drive the flow of either medium through capillaries directly to the microfluidic device (see Fig. 4.15). Once the input has been applied to the cells, a new iteration starts in which, again, images of the yeast cells are taken and transferred to the computer to close the control

loop.

A description of the main components composing the designed technological platform is presented in the following paragraphs.

4.2 Microfluidics for in-vivo control: from device design to experiments

The next paragraphs will highlight the main aspects underlying the employment of microfluidic technology in the modeling of biological networks.

4.2.1 Designing a microfluidic device

Fluid physics at the micrometer scale has its foundation in well understood principles [54]; thus making the design of a microfluidic device a relatively simple task. The first and most relevant aspect to consider when designing a network of channels and chambers is that, due to the characteristic length scale of the devices, the only possible flow regime is the laminar one, where turbulent mixing cannot occur. In terms of fluid physics, this means that, in microfluidic devices, the Reynolds number will be much smaller than one: $\mathrm{Re} \ll 1$.

We recall that

$$\mathrm{R}e = \frac{\rho\, U\, l}{\mu} \tag{4.1}$$

where ρ is the density of the fluid, U is the speed of the fluid, l is the hydraulic diameter of the channel and μ is the dynamic viscosity of the fluid. From physics we know that for $\mathrm{Re} \ll 2000$ the flow is laminar, this means that the fluid can be decomposed into different layers, flowing parallel.

This rules out the possibility of having turbulent flows that, although unpredictable (and thus less desirable for a controlled environment), would allow a fast mixing to occur between two different media, hence

quickly generating a desired concentration of a molecule of interest. This leads to the second major aspect of microfluidics, that is the intermixing between two different species can only occur by diffusion. This behavior is described by the Peclet number:

$$\mathrm{Pe} = v\, C_p\, \frac{L}{k} \tag{4.2}$$

where v average fluid velocity, C_p fluid thermal capacitance, k thermal conductivity coefficient and L equivalent channel diameter.

The Peclet number measures the ratio between advection and diffusion, and it can be interpreted as how far "downstream" a molecule is carried, versus how far it diffuses across the channel in a given unit of time. This quantity provides us with a relevant information: it can be easily demonstrated that for two channels with equivalent Peclet numbers, the narrower one will require a shorter length for complete mixing.

The relevance of this result lies in the fact that it expands the number of degrees of freedom during design, by increasing the number of optimal and suboptimal solutions to a specific problem (e.g. volumetric minimisation of the channels for reagents' costs containment).

The third aspect to be considered when dealing with microfluidic system design is that, even though arguments have been raised against their applicability, Navier-Stokes equations indeed can be used [55] to model fluid flow at the micrometer scale.

This result, together with the already mentioned validity of the laminar flow regime, allows to simply relate the difference in pressure ΔP at the inlet and outlet of an ideal channel with a fluidic resistance R and subjected to a flow Q by writing:

$$\Delta P = Q \cdot R \tag{4.3}$$

The similarity of Eq. 4.3 with Ohm's second law for electrical circuits is clear. Therefore, several principles used in designing electrical circuits

can be conveniently applied to fluidic networks engineering.

Each microfluidic device design, meant to serve the purposes discussed in this work, should satisfy three major requirements:

1. cells need to be "trapped" in a specific region of the device to be used as imaging field;

2. an effective mixing strategy has to be designed in order to expose cells to the right concentration of the desired compounds;

3. the generation of stressogenic stimuli to the cells must be minimised.

Taking into account all these considerations, I first designed the device presented in Fig.4.2.

The channels in this device are 10 μm in height and thus only allow growth of cells in a monolayer, in order to ease image segmentation. The mixing section takes advantage of a Y shaped configuration and a resistance-like serpentine is used to increase the mixing channel length while keeping the linear dimensions of the device small.

As trapping chamber, I used a "Tesla diode" chamber [25]: this chamber is shaped so that a secondary flow is derived from the one exiting the mixing section.

In this cell trap, the cells are subjected to limited flow velocities and can then tether and grow without stress-inducing constraints.

This device satisfies all the constraints mentioned above. Moreover an electrical equivalent can be easily obtained as reported in Fig. 4.3.

The details of this device will be discussed in the following paragraph, together with a description of the main challenges it poses.

In addition to this device, I was able to use the MFD0005_a thoroughly described in [56] thanks to a collaboration with Prof. Jeff Hasty at University of California at San Diego. The design of this device is presented in Fig. 4.4.

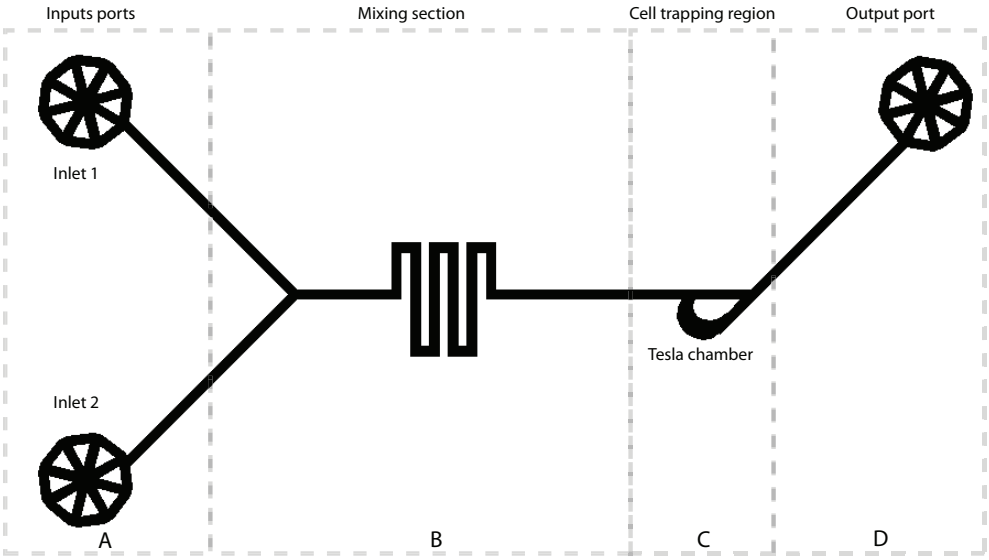

Figure 4.2: **Overview of the *Yeast Controller* device.** The four sections of this device are highlighted: the input ports (A) are used to supply the two media supplemented either with Glucose or Galactose. The mixing section (B) allows complete mixing of the two streams coming from the two inlets. The cell trapping region features the Tesla chamber (C) [26] that allows cell trapping by deriving a secondary flow from the main stream. Finally, media are extracted from the device through the output port (D) that serves as input port while cell loading is operated.

While this device displays a much more intricate design, it satisfies all of the design constraints. Moreover it simplifies the cell loading step, required at the start of each experiment to place yeast cells within the chamber in the right number. At the same time this device minimises the number of cells growing outside of the chamber thus delaying the occurrence of clogging. Remarkably, this design limits the growth of the cell monolayer to the ring-shaped trap and it is characterised by different channel heights across the device. Moreover the input mixing section is meant to avoid flow reversal, and thus is characterised by a higher level of robustness. The specifications of this device will be discussed in the next section.

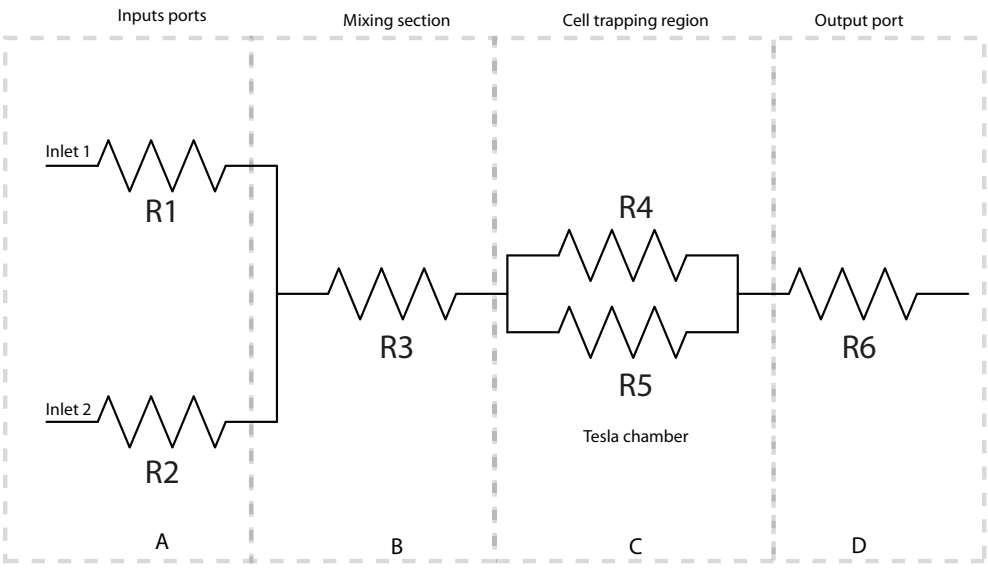

Figure 4.3: **Electrical equivalent circuit of the *Yeast Controller*.** As it can be seen each channel of the device is substituted by its fluidic resistance R_i.

4.2.2 Simulating flows: Finite Element Analysis

Finite Elements Analysis (FEA) has been extensively used in engineering to predict the behavior of physical systems governed by partial differential equations (PDEs) (e.g. magnetic fields, acoustics etc). The solution approach aims at eliminating the differential equation completely (steady state problems), or approximating PDEs with system of ordinary differential equations (ODEs), which are then numerically integrated using standard numerical techniques such as Euler's method, Runge-Kutta method etc.

As we discussed previously, for what concerns microfluidic devices, we are interested in two main aspects: fluid speed, in oder to prevent the generation of shear stress, and mass transport phenomena, in order to guarantee perfect mixing between the desired compounds.

The satisfaction of biocompatibility and operational constraints is of utmost importance for any microfludics device to be used for our purposes and should thus be checked before fabrication.

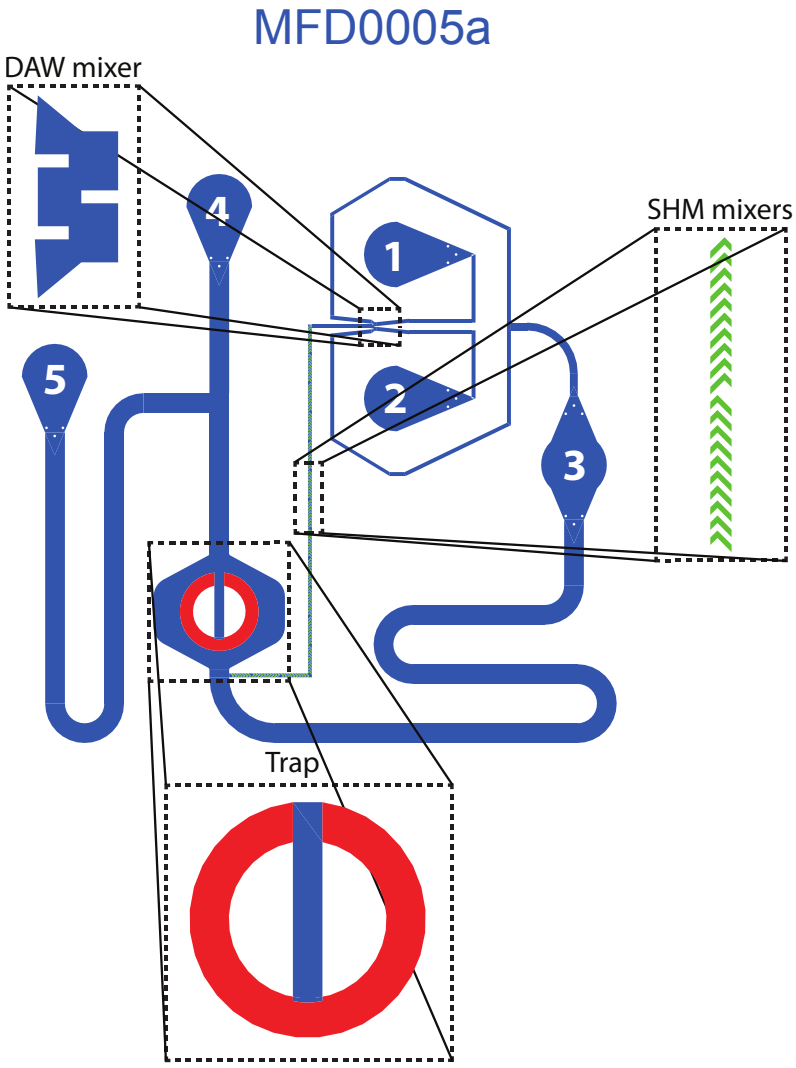

Figure 4.4: **Design of the** MFD0005$_a$ **device.** Ports 1 and 2 are meant to host Galactose/Raffinose and Glucose supplemented syringe connections. Yeast cells are trapped (*Trap* inset in this figure) as depicted in panel A of Fig. 4.5 . After having been loaded within the device cells are exposed to arbitrary stimuli obtained by tuning the mixing ratio at the *Dial-A-Wave* section (*DAW mixer* inset) between the media connected to ports 1 and 2; in order to increase the mixing efficiency Staggering Herringbone Mixers (SHM) has been implemented as reported in the *SHM Mixers* inset. Ports 3, 4 and 5 are generally considered outlet ports; but while the former two always serve as flow outlets, port 5, during cell loading, is meant to connect the yeasts filled syringe to the device.

Unfortunately, although linear static solutions can be found for the first problem, it is much harder to find similarly simple results for the

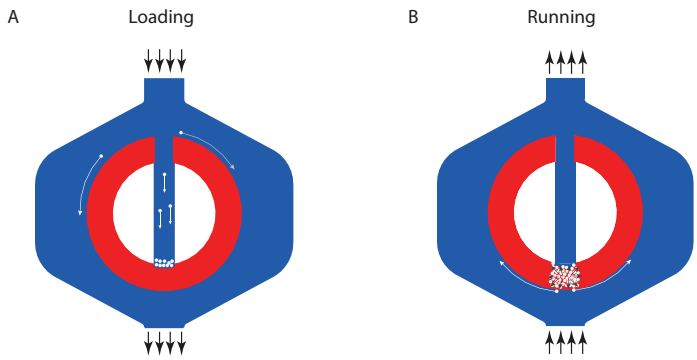

Figure 4.5: **Main device operations for** MFD0005$_a$**.** *Cell loading* is accomplished by connecting a syringe filled with suspended yeast cells to port 5 of the device (see Fig. 4.4). The flow (incoming arrows in the upper part of panel A) generated by this syringe, pushes the yeast cells (white circles) towards the trap. The height of the red ring is 10 μm and, therefore, prevents cells from stacking (yeast cell diameter varies from 5 μm to 10 μm). After cells have been trapped the flow is reverted so that media coming from ports 1 and 2 reaches the yeasts that can grow until they fill the chamber and, eventually, the whole device (we refer to this situation as *device clogging* which ends the useful duration of any experiment).

mass transport problem. This is where FEA comes into play, by providing an agile approach for microfluidic device performance evaluation and optimisation.

FEA requires a Finite Element Method (FEM), which can be easily derived for microfluidic device. Boundary conditions can be used to study how the flow will distribute within the device (Navier-Stoke's equation) and, by coupling mass transport phenomena (Fick's laws of diffusion), it is possible to evaluate whether all of the specifications are satisfied.

To this aim, we carried out this in-silico analysis of the microfludic devices before moving to prototyping.

Figure 4.6 presents these results for the *Yeast Controller* device.

All of the relevant operations, namely fluid mixing and cell loading/-trapping, can be successfully accomplished in this device while meeting the previously discussed specifications. Unfortunately while using this device we noticed that after 15-20 hours, a set of issues involving the mixing strategy arose, as recapitulated in Fig. 4.7.

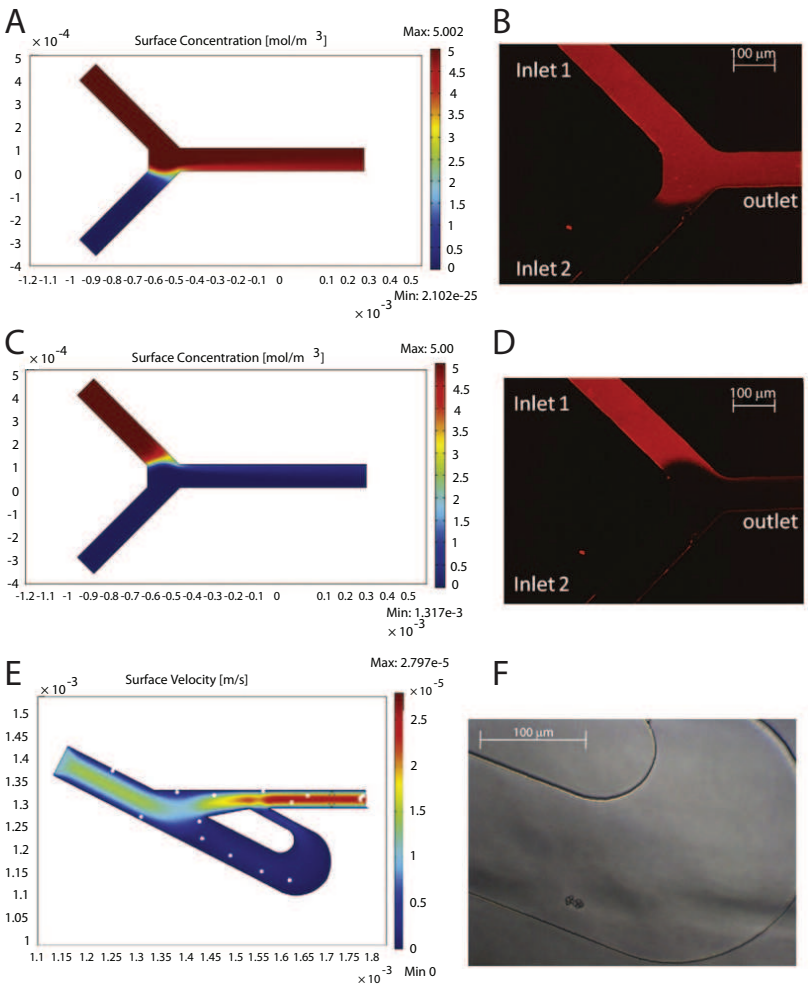

Figure 4.6: **Computational fluid dynamic simulations.** Finite elements model predictions and experimental validation are presented side-by-side for the cases of 0 % (panel A and B, simulation and experimental results)and 100 % (panel C and D, simulation and experimental results) Galactose (supplemented in this experiment with a fluorescent dye, Sulforhodamine B) to Glucose ratio. Additionally cell (white circled in panel E) trapping strategy has been tested computationally: darker color in the surface velocity analysis of the *Tesla Chamber* confirms the dramatic reduction of fluid speed in this region of the device (sustaining our expectations and in full satisfaction of bio-compatibility requirements). As observable in panel F, the region where cells will get stuck is perfectly captured by in silico analysis (panel E).

Therefore application of this microfluidic device are limited to short experiments or, alternatively, should be coupled to the use of expensive

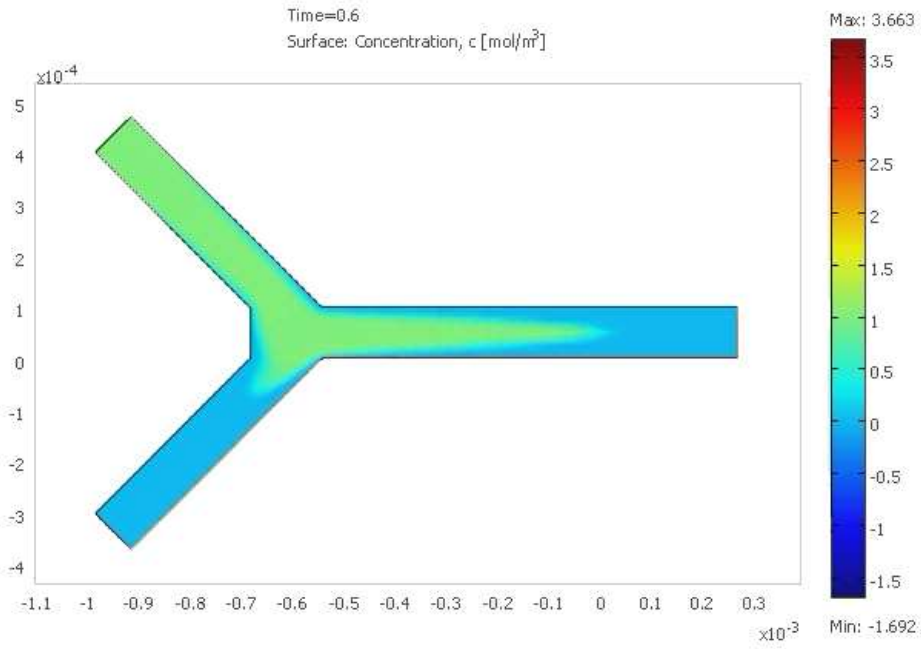

Figure 4.7: **Finite element simulation of flow reversal.** Due to intrinsic limitations, this mixing strategy leads to an unwanted intermixing between the two input compounds. This is mainly due to the fact that both the 0 % and 100 % Galactose/Glucose mixing ratios are very sensitive to a fine regulation of the pressure applied to the two inlets. As a result of this if, for some reason, precision is lost in pressure regulation during the experiment, one of the fluids (the green one in this figure) can cross the Y section and fill the inlet destined to the other input compound. Evidently this situation is highly unfavourable since it introduces and untolerable uncertaintly on the type and amount of stimulus provided to the cells.

pico-pumps.

In order to avoid the flow reversal issue, the MFD0005$_a$ uses the so called Dial-A-Wave (DAW) mixer: finite element simulations of this configuration are presented in Fig. 4.8.

Interestingly, the DAW mixer demonstrates a lower degree of dependence on the accuracy of the pressure-at-inlets level: as a matter of fact if the pressure at one of the two inlets is higher (or lower) than the expected, the flow will not proceed towards the lower pressure inlet but

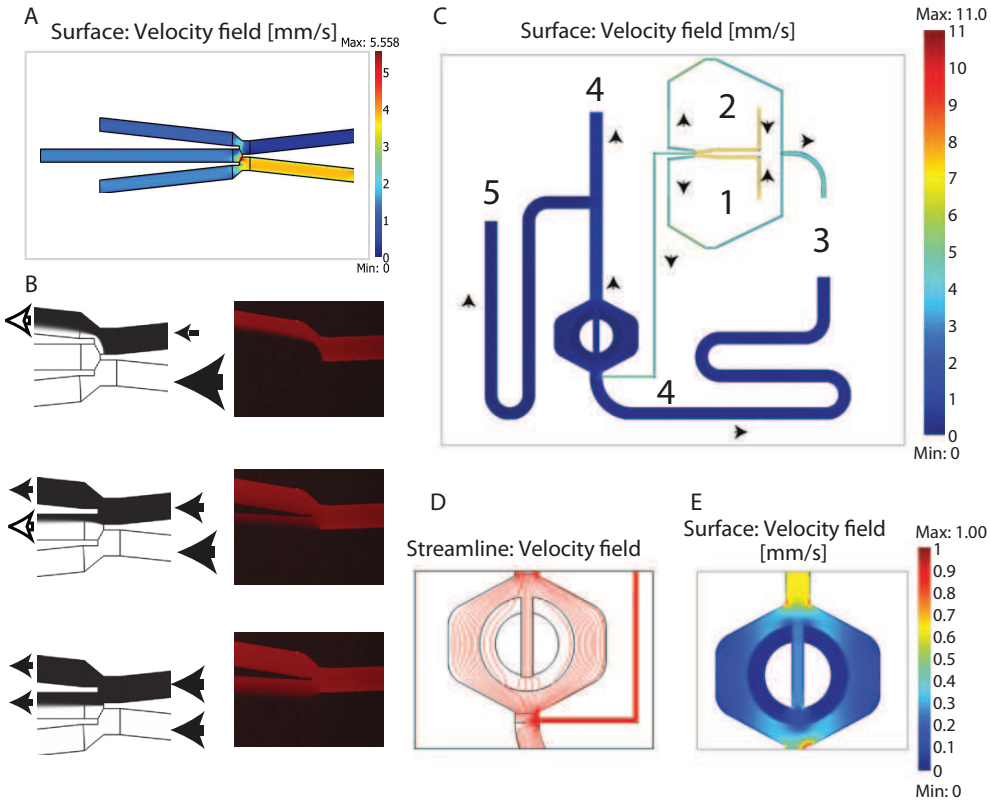

Figure 4.8: **Computational and experimental analysis of** MFD0005$_a$**.** Panel A shows simulation of the velocity field that generate the 0 % mixing ratio between the compounds at inlet 1 and 2 (first plot in panel B). In this case the compound in black in panel B does not reach the the cells which are trapped close to the end of the central of the three exit channels). By varying the pressures applied at the two inlets it is possible to obtain different mixing ratios as showed in panel B; computational predictions have been tested in vitro by adding Sulforhodamine B (Sigma-Aldrich) to the medium connected to port 2. Surface velocity fields computed for the whole device (panel C) confirm the low fluid speed at the cell trap; this result is further analysed in panels D and E where flow stream lines and velocity fields confirm the absence of any potentially harmful phenomenon.

will be directed to one of the two side channels around the central one in DAW that leads the flux to port 3, an outlet (see panels B and C in Fig. 4.8). This prevents any undesirable intermixing event to occur thus limiting the chances of technical errors during long experiments.

On the basis of this result we modified the design of *Yeast Controller* (v2.0) so as to reflect the one depicted in Fig. 4.9.

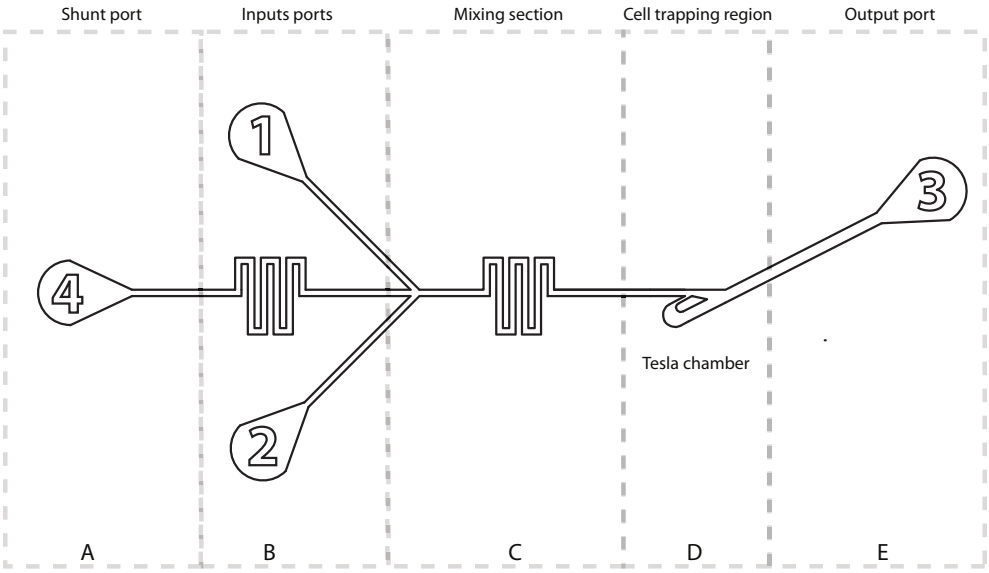

Figure 4.9: **Revised design of the yeast controller.** A shunt channel (connected to the outlet port 4) has been added to the mixing section so as to allow medium overflow to be redirected to an outlet port instead of giving rise to intermixing.

Thanks to this design the issue related to flow reversal has been totally overcome thus rendering *Yeast Controller* device viable again for long term experiments.

Once device designs have been tested/optimised in silico and their correct operations have been checked, it is possible to move to the actual device prototyping and fabrication.

4.2.3 Prototyping and fabricating microfluidic devices

The production of microfluidic devices is usually accomplished in clean rooms in order to prevent dust particles from infiltrating the polymer. For our purposes we used the class 100 clean room of the "Istituto di Microelettronica e Microsistemi" of the CNR, in Naples. On the basis of

the design obtained after the in-silico verification/optimisation process, a mask is generated that features all the structures meant to generate the fluidic circuit. The technique used to obtain the master mold (used for the *replica molding*) is common to many electronics-related processes: it is soft-photolithography. Usually photosensitive resins, called photoresists, are spun over silicon wafer; the previously obtained mask is put between the wafer and a UV light source as reported in Fig. 4.10 (see [21] for a review).

UV light exposed photoresist will polymerise (depending on the type of photoresist alternative behaviors can be observed) and become resistant to a developer solution. This reagent will eliminate all uncrosslinked photoresist molecules leaving on the wafer a precise profile of the desired structures. Fig. 4.11 show the result of this process for the optimised *Yeast Controller* device.

Once the master is available the replica-molding protocol can be used to obtain a large number of identical devices. The fabrication of a microfluidic device starts with the pouring of the Polydymethilsiloxane (PDMS) liquid pre-polymer/curing agent mix on the master mold as shown in Fig. 4.12.

The petri dish containing the polymer is then baked at 80 °C for 2 hours and then the PDMS is cut and peeled off from the master as shown in Fig. 4.13.

The new devices are then attached to glass slides by reactive oxygen treatment of both the polymer pieces and slides. At the end of this step the device are ready for inspection for anomalies and defects as shown in Fig. 4.14.

4.2.4 Experimental setup

Once microfluidic devices have been fabricated they are inspected for defects and the ones passing Quality Control are autoclaved to ensure

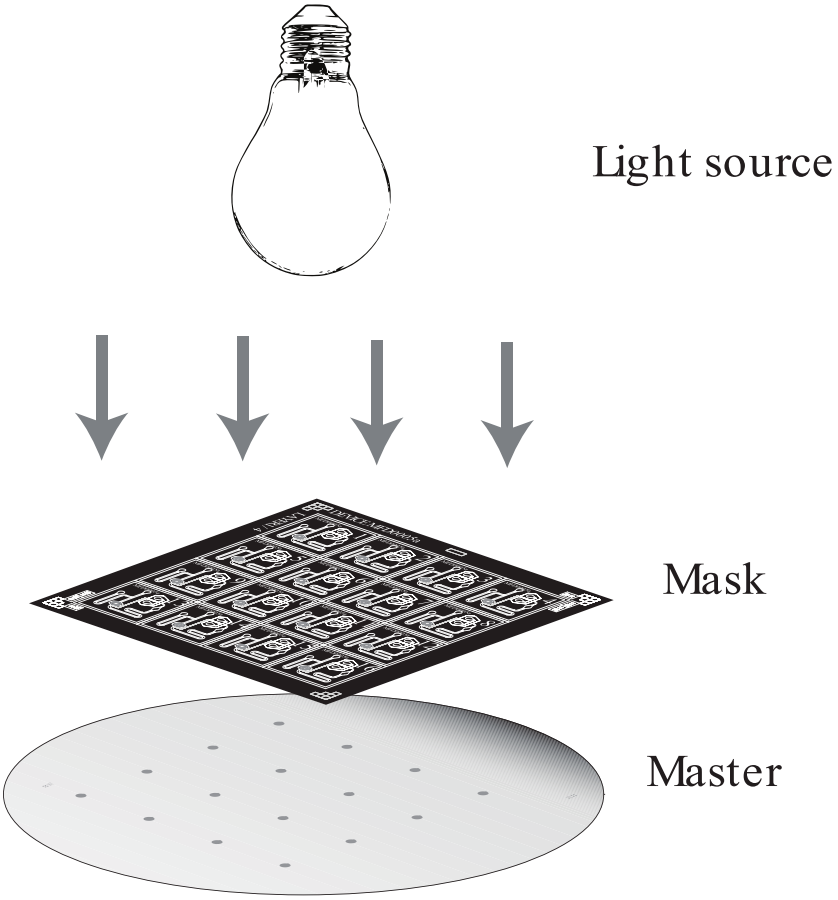

Figure 4.10: **Fabrication of master mold for microfluidic device production.** A mask is used to filter light that polymerises resins spun on the silicon wafer. Once unpolymerised photoresist has been removed, the wafer is ready for entering the production process.

sterility and stored in sealed petri dishes at room temperature.

On the day of the experiment (day 1) a microfluidic device is first secured to an inverted microscope sample holder and re-inspected at 4X and 10X magnification. Unless new defects are found, the device is connected through capillaries to syringes filled with ddH2O that are meant to wet the channels and eliminate any residual air bubble.

Once the *wetting* is accomplished these syringes are disconnected from

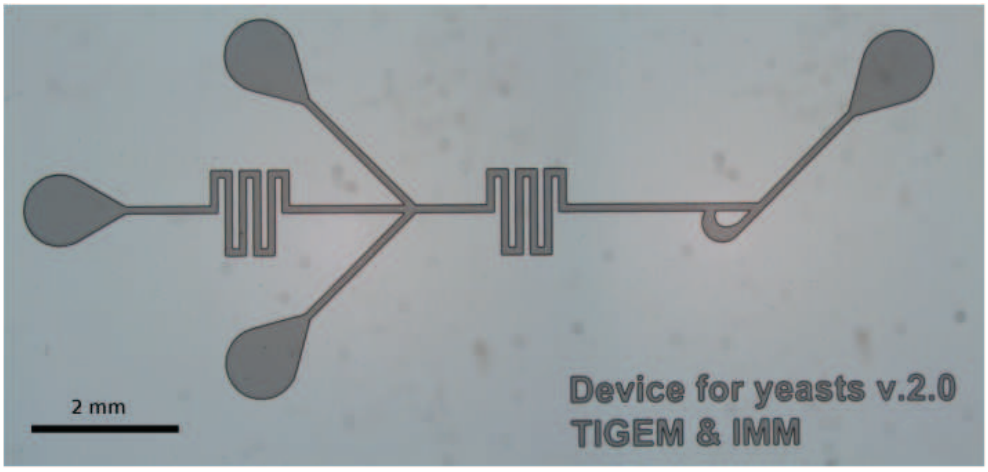

Figure 4.11: **Silicon wafer of the master mold of the improved *Yeast Controller* (v2.0).** Microfluidic structures are clearly visible at 5X magnification.

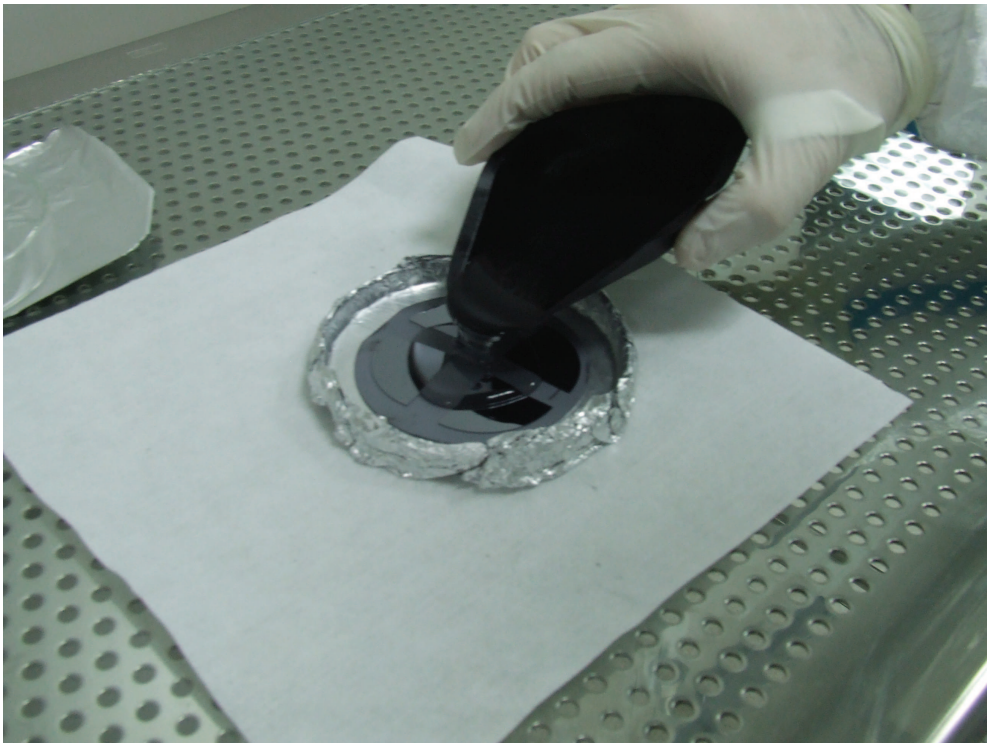

Figure 4.12: **PDMS deposition over the four master molds.**

the device. Outlet ports are connected through capillaries to new ddH2O water filled syringes, while inlet ports are connected to the two syringes

Figure 4.13: **PDMS is cut with a razor and gently peeled off from the wafer.** At the end of this process a piece of polymer will be obtained with microfluidic structures engraved and visible on one of the wider surfaces of the polymer. Holes ensuring connections through ports can be easily obtained by using 20G needles.

(filled with Glucose-medium and Galactose-medium): Synthetic Complete (SC) Medium with either Glucose or Galactose is used, since it has low autofluorescence.

Moreover, as documented in [3] it is worth supplementing Galactose-medium with Raffinose in order to decrease the energetic stress induced by the availability of Galactose as unique carbon source in the medium.

Sulforhodamine B (Sigma-Aldrich) is added to SC+Galactose+Raffinose medium in one syringe; this fluorophore, visible at wavelengths of 630 nm, is used to track the presence of inducer compounds in the device.

A syringe containing Glucose supplemented SC medium is then attached to the device. Syringes are attached to the Dial-A-Wave system

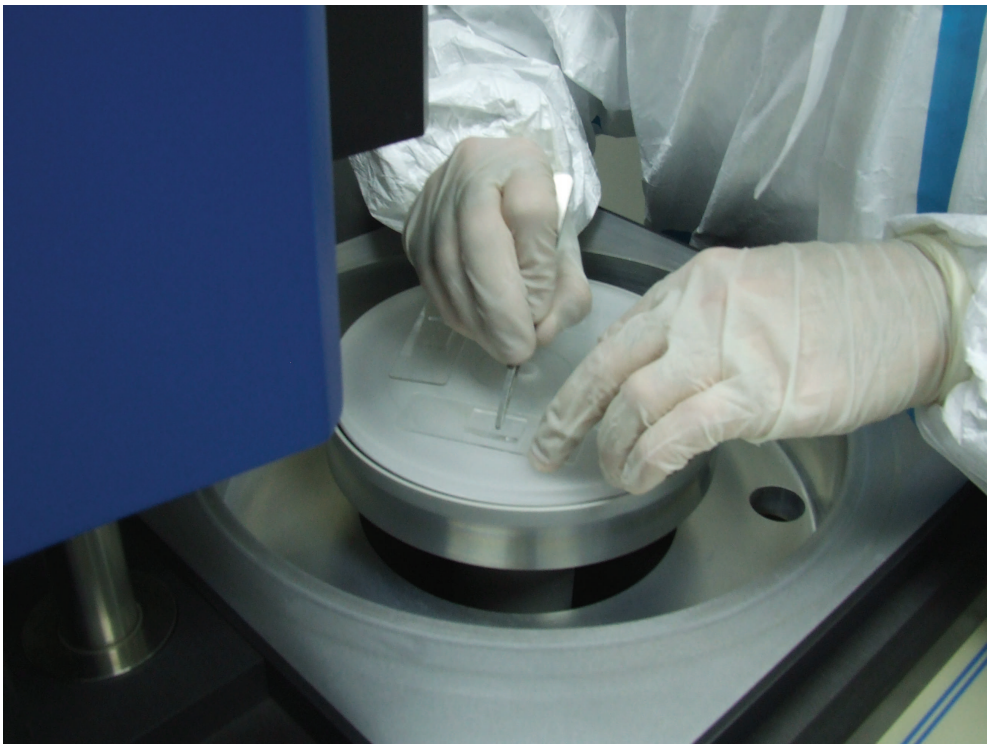

Figure 4.14: **Microfluidic devices ready for the inspection.**

described below. At this point the device is tested against improper fluxes by switching the heights of syringes and checking the correct signal in the cell trap. Cells are then loaded in the device and the actual control algorithm is launched.

A detailed overview of the experimental protocol is reported in Appendix C.

4.3 Actuation strategy: re-engineering of the Dial-A-Wave system

In order to modulate the concentration of compounds within a cell trap, in a microfluidic device, a pressure source is needed. Several alternative methods can be used: electro-mechanical solutions guarantee unequaled precision in the regulation, but tend to me more expensive.

On the other hand hydrostatic pressure generators are able conjugate adequate control performances with very limited costs. For this reason, inspired by the work of [26], I considered the latter class of systems and designed a new solution for the DAW junction (proposed in chapter 19 of [56] by Ferry and colleagues) that further decreases the costs associated to such a platform. This will be the subject of the next paragraph.

4.3.1 Hydrostatic pressure actuation: linear rails design and development

Hydrostatic pressure is the ratio of the force over surface exerted by a fluid at equilibrium due to the force of gravity. Therefore, once the device wetting has been accomplished, fluids can be forced to flow within its channels, by connecting syringes filled with desired compounds.

Observe that the syringes used in this design, have no plunger, since these are used simply as tanks to store the media. Namely, each of these syringes will exert a hydrostatic force over the fluids in the channel that is simply proportional to its height.

Hence, when the two syringes are set at different heights, a differential pressure will be generated at the level of the mixing section (see Fig. 4.8 for details on the dynamics of fluids withing the device). This phenomenon is illustrated in Fig. 4.15 for the DAW junction.

Therefore, it is possible to effectively control the medium reaching the cells by rising or lowering the height of the two syringes attached to the inlet ports.

In order to automate the platform, I translated this requirement into a classical positioning control problem. Motorised linear rails can be designed and controlled by a computer program to allow fast and precise operations. In order not to introduce any unnecessary level of complexity, stepper motors can be used in place of the usual DC motor: cheap controller boards can be employed to transfer control signals to such motors.

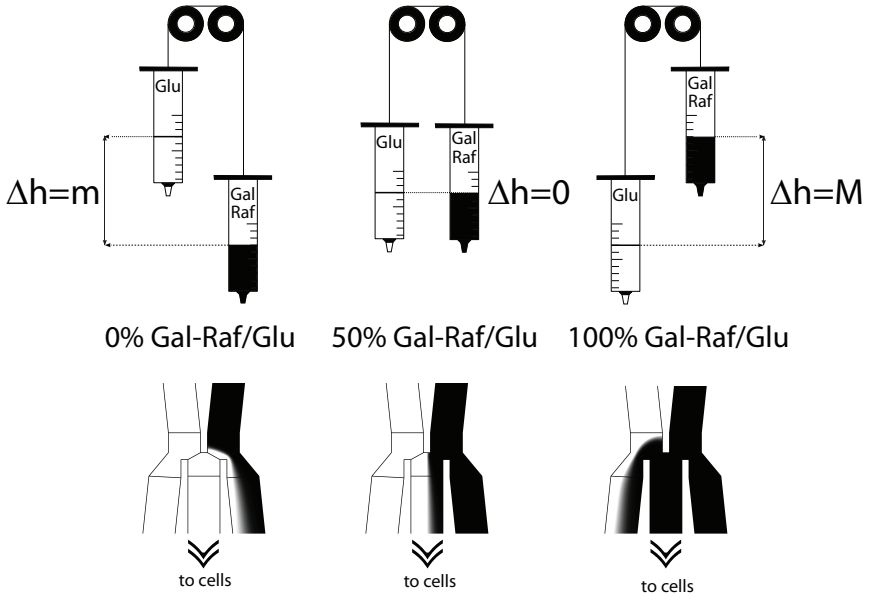

Figure 4.15: **Hyrostatic pressure effect in the DAW junction in** MFD0005_a**.** Syringes can be constrained to slide on pulleys so that the difference in their heights Δh is always constant. For $\Delta h = m$ we obtain a GAL-RAF/GLU mixing ratio equal to 0%, meaning that only Glucose reaches cells. The dual situation ($\Delta H = M$ and $M = -m$) gives raise to 100% mixing ratio. When the two syringes are at the same level the mixing is equal to 50%. This configuration would only require one motorised linear rail for two syringes and could be used to further minimise implementation costs.

thus providing a suitable interface for our in-vivo control algorithm.

For these reasons, I designed a stepper-motor based linear rail that I fabricated in two copies, one per syringe (this design was developed with the contribution of Gianfranco Fiore at the University of Naples).

The design is presented in Fig. 4.16. Aluminium has been used for the frame in order to keep the total weights of the rails limited and to ease the fabrication operations (drilling etc). The result of the rails fabrication are presented in Fig. 4.17.

The sizing of the stepper motors is the central topic of the next paragraph.

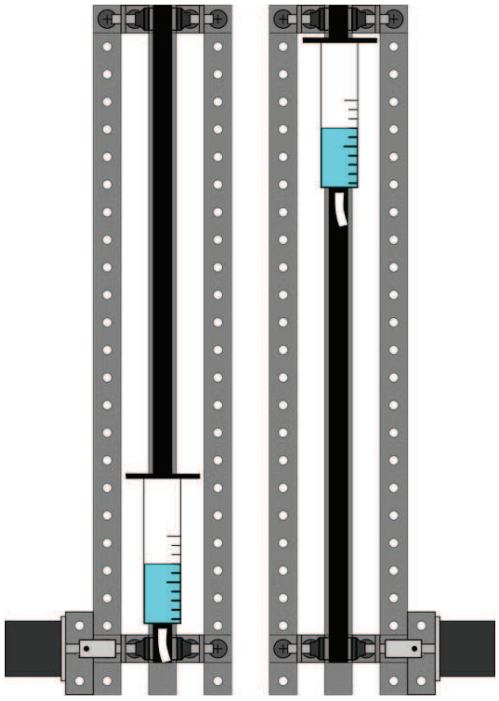

Figure 4.16: **Motorised linear rails.** The two designed linear rails designed are presented in this 2D model. The external frame sustains the holder for the stepper motor and the two pulleys on which the timing belt is meant to slide. The sliding element is mechanically secured to the belt in order to allow coherent movements.

4.3.2 Stepper motors-based actuation: from sizing to control

The choice of the stepper motors has been guided by the static and dynamic constraints they need to satisfy. Thus, it is important to define the load, fixed to the belt and its acceleration profile. Approximately the max load is about 0.2 Kg (filled syringe and glass beaker) and the rising time is 10 seconds (negligible if compared to the time interval needed to acquire images). Moreover, considering the task to be accomplished, motors with a low power demand are preferred. A motor featuring has been chosen whose specifications are reported in Tab. 4.1.

All of the parameters in Tab. 4.1 above can be easily interpreted; a separate discussion must be made for the holding torque and the detent

Figure 4.17: **Linear rails.** The finilised linear rails are presented in this figure.

torque. The holding torque is the maximum motor torque when it is electrically powered, while the detent one is the motor torque when power is switched off. Considering actuator static and dynamical constraints, it is necessary to verify whether it is possible to use this motor for the selected task; first of all, from catalog specifications of the timing belt,

Rated voltage	12 Vdc
Phase current	0.6 A
Holding torque	50 Ncm
Detent torque	3.5 Ncm
Rotor inertia	120 gcm^2
Shaft diameter	6.35 mm
Shaft length	19 mm
Step angle	1.8°
Step Accuracy	0.09°

Table 4.1: **Specifications of the selected stepper motor selected.**

pulleys weight, inertia and load inertia are calculated obtaining:

Pulley mass:
$M_P = \left[\pi \left(\frac{14.7 \cdot 10^{-3}}{2} \cdot 17.5 \cdot 10^{-3}\right)\right] m^3 \cdot 2700 \frac{Kg}{m^3} = 8.02 \cdot 10^{-3} Kg$

Pulleys total inertia:
$J_P = 2 \cdot \left[\frac{1}{2} M_P \cdot r_P^2\right] = 8.02 \cdot 10^{-3} Kg \cdot (7.35 \cdot 10^{-3})^2 = 4.33 \cdot 10^{-7} Kg \cdot m^2$

Load inertia:
$J_L = M_L \cdot r_P^2 = 0.2 Kg \cdot (5.75 \cdot 10^{-3} m)^2 = 6.62 \cdot 10^{-6} Kg \cdot m^{-2}$

Thus, given load mass M_L, the load weight force is expressed by:

$$P_L = \left(0.2 Kg \cdot 9.81 \frac{m}{s^2}\right) \cong 2N \tag{4.4}$$

Once P_L has been calculated, it is possible to check whether the constraint on the motor detent torque is satisfied by calculating load holding torque T_L:

$$T_L = P_L \cdot r_P = 0.0115 Nm = 1.15 Ncm \tag{4.5}$$

The value obtained for T_L is smaller than the motor detent torque, thus this constraint is satisfied and when the motor is not powered the load holds its position on the linear guide.

To verify the constraint on the motor holding torque, it is necessary to analyze the linear actuator dynamical behavior. In particular the sum of the load holding torque and the torque that is necessary to move the load (with desired velocity and acceleration profiles) must be calculated.

Considering an activation time of 10 s, a trapezoidal velocity profile is defined, with acceleration/deceleratation interval equal to 3 s; therefore, maximum velocity v_M and maximum acceleration a_M are calculated as follows:

$$v_M = \frac{0.6}{\frac{3+3}{2} + 4} = 0.086\frac{m}{s} \tag{4.6}$$

$$a_M = \frac{v_M}{3} = 0.029\frac{m}{s^2} \tag{4.7}$$

The total system (rotor + pulleys + load) inertia J_T is equal to:

$$J_{=}J_R + J_P + J_L = 1.906 \cdot 10^{-5} Kgm^2 \tag{4.8}$$

The maximum linear acceleration must be converted into angular acceleration (pulley acceleration must be the same of the entire system) through the following formula:

$$\frac{d\omega_P}{dt} = \frac{1}{\gamma_P} \cdot a_M = 5\frac{rad}{s^2} \tag{4.9}$$

Thereby the total torque needed to move the entire system at the maximum value of acceleration is given by:

$$T_T = J_T \cdot \frac{d\omega_P}{dt} + T_L = 0.116Nm \tag{4.10}$$

This value is smaller than the motor holding torque, this other constraint is satisfied, and thus the chosen motor is suitable for the specific task required.

In order to drive the motors with the appropriate sequence of pulses, a control PCB has been used. This controller board, with an appropriate dynamic link library (DLL), allows to control both the motors through

the USB PC port; to perform this task routines have been included in the control algorithm written in Matlab environment (see Appendix B for details on the stepper control code).

4.4 Microscopy and image analysis

The closed-loop control platform described above, employs an inverted epifluorescence Nikon-TI Eclipse microscope (Fig. 4.18).

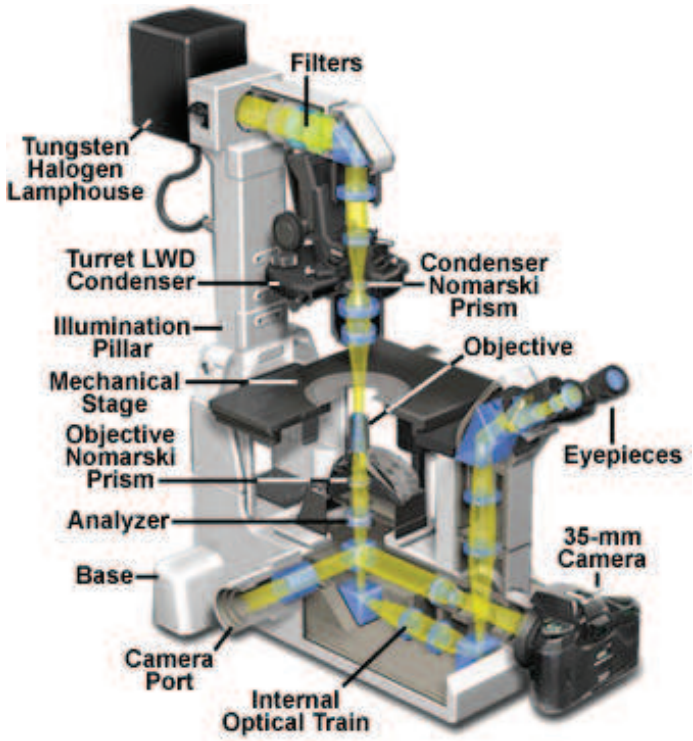

Figure 4.18: **Inverted epifluorescence microscope.** In this configuration the objectives are assembled on a motorised nosepiece. The optical train conveys light to either eyepieces of to a CCD camera that digitises the image and sends it to the computer controlling the acquisition.

In this configuration, images are taken by objectives that are mounted on the base of the microscope instead of on the tower (as it happens with upright microscopes). This configuration is, to some extent, necessary in microfluidics based experiments since high magnification objectives (40X-

60X) usually require very short *working distances* (i.e. distances between the lens and the specimen). This would be incompatible with upright configurations because of the presence of the usually thick ($3 - 4$ mm) PDMS layers required to secure the capillaries to the device by means of metallic needles.

Once the device has been attached to the holder on the moving stage, the experiment can begin.

The microscope is programmed to acquire two types of images: *(a)* a bright field image and *(b)* two fluorescence images (one for the green spectrum for GFP and one for the red spectrum for Sulforhodamine B). An example of both the types of images is presented in Fig. 4.19

Once cells have been imaged, image analysis algorithms can be applied to estimate their fluorescence [18]. To this end, I developed a custom image processing algorithm to fully exploit the peculiarities of our platform. My aim was to carry out both cell segmentation and cell-tracking, maximising sensitivity as primary objective, and then refining the results by improving specificity.

Cell tracking and lineage reconstruction in microscopy image sequences can be performed in two steps:

1. the first step consists in the segmentation of each frame in order to extract the position of the cells;

2. the second step consists in detecting single cell movements through the identification of same cells present in two consecutive frames.

In designing the image processing algorithm (see [57] for details), we aimed at making the solution robust to image sequence variability in terms of intensity contrast between the pixels belonging to the cells and the pixels belonging to the background. This is a key factor in the field of biology: as mentioned above, experimental conditions may vary dramatically on the basis of the employed technology (light microscopy vs confo-

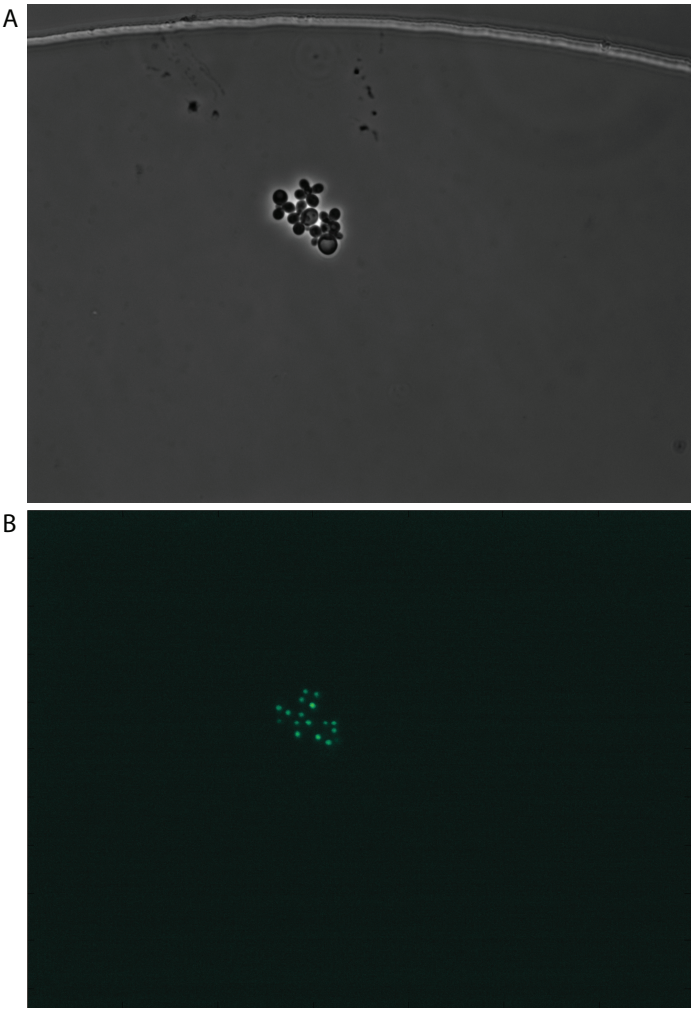

Figure 4.19: **Bright field and fluorescence images.** Here the same field has been imaged in the spectra of transmitted (panel A) and reflected light (panel B, GFP filter). The nuclear fluorescence of IRMA cells trapped in MFD0005$_a$ can be appreciated in panel B.

cal etc), of the optical configurations (magnification used, lens numerical aperture etc) and of supporting materials (e.g. coverslip chamber-slide vs microfluidic devices).

In order to make our solution as insensitive as possible to these factors, we based our segmentation method on global edge linking system.

Yeast cells in bright field phase contrast images occur in clustered, low intensity, convex and often quasi-circular shapes surrounded by a

white halo (Fig. 4.20 panel A).

The contrast between the pixels belonging to the cells and the pixels belonging to the halos is usually so high that edge points can be detected by the evaluation of the magnitude of the gradient calculated in each point of the image (Fig. 4.20 panel B).

Due to the yeast cell shape, edge points can be connected with the Circular Hough Transform (CHT) [58]. CHT can detect almost all of the cells whithin the image, even when cells edges overlap (see panel C in Fig 4.20); unfortunately the CHT algorithm is computationally expensive (both in memory and time) and shows poor specificity capabilities.

Therefore, in order to limit the computational cost associated to the algorithm, we designed a preprocessing of the image that first selects the regions in the image where cells are located (Fig. 4.21 panel A). Those regions are selected with a mask (Fig. 4.21 panels B and C), obtained as a result of a thresholding and of morphological operations (aperture, closure and filling as detailed in Appendix B). False detections are reduced as a result of the segmentation of these regions. Moreover, since the area of the regions in the image containing cells is smaller than the area of the background, the computational time of the CHT is considerably reduced.

Once cells have been located in the bright field, a binary filter is built to detect only pixels of the GFP fluorescence field within cells. Let us define the fluorescence field image I as:

$$I : (p) \in \Omega \subset \mathbb{N}^2 \tag{4.11}$$

then

$$I(x, y) \in \left[0, 2^{-L} - 1\right] \subset \mathbb{N} \tag{4.12}$$

with x and y generic coordinates and L the number of bits used for image encoding.

The mask image M can be similarly defined as:

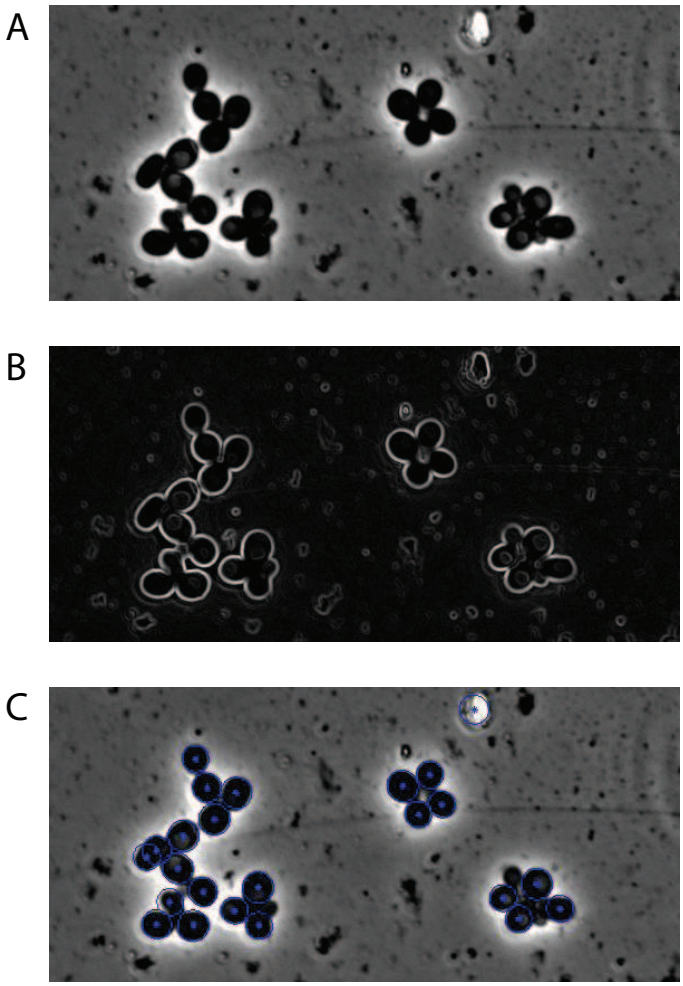

Figure 4.20: **Yeast cells in a bright field phase contrast microscopy image.** In panel A the original image is shown, while the gradient magnitude image is present in panel B. Circles detected by CHT represent the identified cells in panel C.

$$M : (p) \in 0, 1 \tag{4.13}$$

where $M(x, y) = 1$ denotes a cell belonging pixel while $M(x, y) = 0$ indicates background pixels.

The latter class of pixels is useful to estimate the amplitude of the background signal, which can be subtracted to the raw signal to obtain a normalised fluorescence intensity. In order to compute the normalised

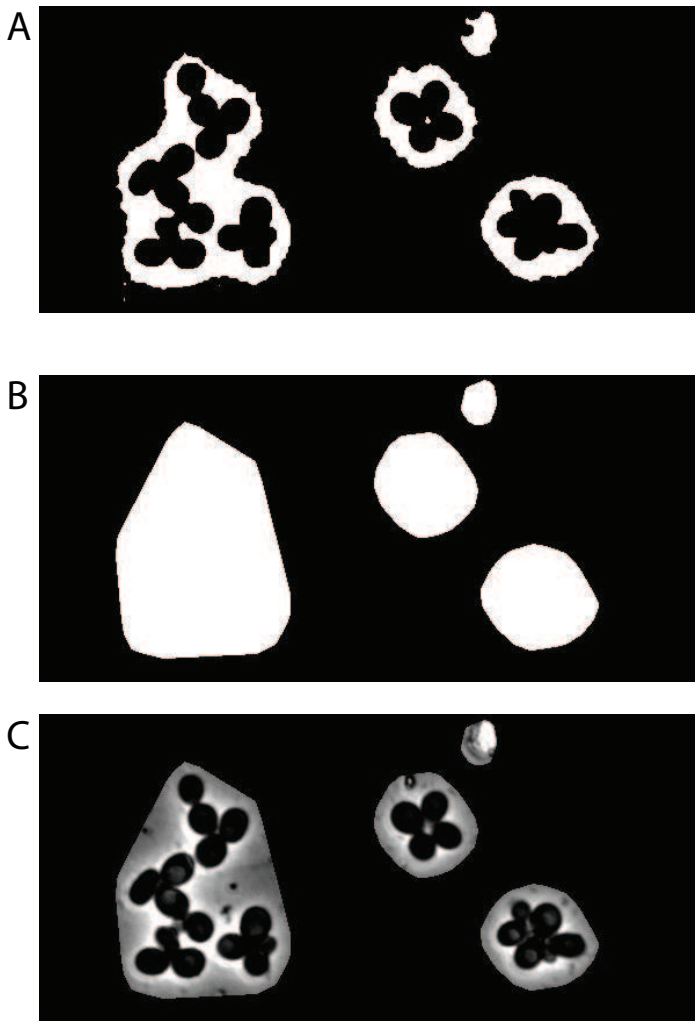

Figure 4.21: **Convex hull derivation for bright field cell images.** Panel A shows the segmented image obtained by applying described morphological operators. In panel B the convex hulls computed for the figure in panel A have been reported; panel C presents the segment of the original figure that is further considered for computations.

signal, we use the following equation:

$$\mathrm{GFP}_{\mathrm{avg}} = \underbrace{\sum_i \sum_j I(x,y) \cdot M(x,y)}_{\text{raw GFP signal}} - \underbrace{\sum_i \sum_j I(x,y) \cdot \neg M(x,y)}_{\text{background signal}} \quad (4.14)$$

with i and j spanning the rows and columns, respectively, of the

arrays. $\neg M(x, y)$ is a transformation of M that is simply meant to complement the binary values of the original matrix (so as to select image areas not belonging to cells).

The quantity GFP_{avg} becomes the y signal value the control algorithm will use for its computations.

4.5 Software implementation of the control algorithm

When an experiment starts, and for its duration, it is necessary to manage all the devices included in the technological platform in order to achieve the control aim. This task must be accomplished by the control algorithm.

As previously stated, we decided to implement our control algorithm as a Finite State Automaton. The algorithm has been coded in the MATLAB framework; the code was meant to ensure that the control scheme is effective and reliable.

The control scheme is depicted in Fig. 3.8; in this case, however, time-discrete signals should be considered. The control action is calculated when an image is acquired by the microscope and analysed to compute the normalised fluorescence signal. This is then used to compute the feedback signal y_s, which is compared against the reference signal, to obtain the error e.

Given the error, a simulation (over a 300 seconds time horizon) of the cascade PI - PWM is carried out to obtain the control input u.

As represented in Fig. 4.22, at the k^{th} step, $e(k)$ is known, thus it is possible to calculate the control input u applied to both the plant and the IRMA model M. The control input is used to compute the time during which the Galactose syringe has to stay up via the PWM output duty-cycle (over 5 minutes), whereas M is simulated with u as input (see

Fig. 4.23). At the $(k+1)^{th}$ instant a new image is acquired and the feedback computation takes place.

The error $e(k+1)$ is available for a new control iteration and each step is repeated again.

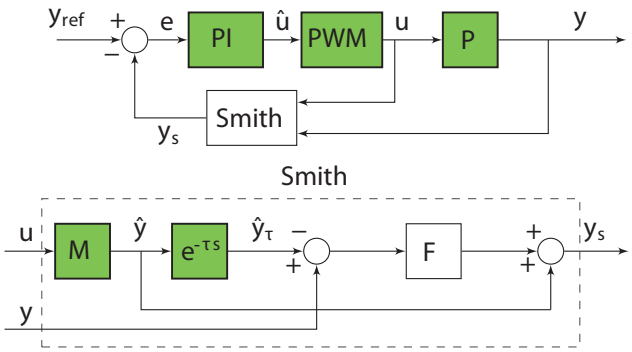

Figure 4.22: **Control scheme at k^{th} instant** Active blocks in green.

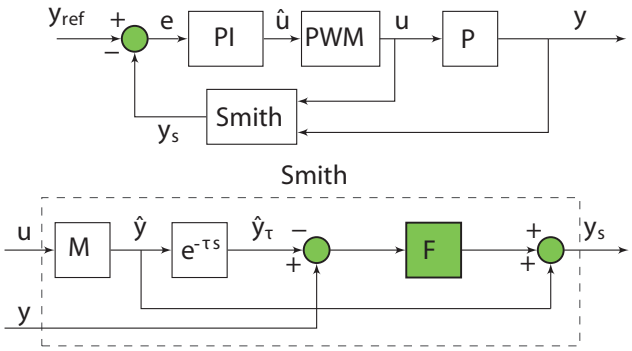

Figure 4.23: **Control scheme on $(k+1)^{th}$ instant**. Active blocks in green.

A FSA can be easily derived from the steps outlined above: Fig. 4.24 represents a suitable implementation.

In the initial state (state 0 in Fig. 4.24), the signal calibration (initialisation) is carried out. If this phase ends without issues (e.g. errors due to image acquisition or biological issues), it is then possible to move on to state 1; here, given the error and the cascade PI-PWM simulation,

the control input u is calculated; in state 2, M is simulated using the calculated input signal; in state 3 the control input is translated into a signal sent to the stepper motors to provide the cells with the input computed by the control algorithm; in state 4 the delayed version of $\hat{y}$ is calculated (if necessary depending on the control scheme used as described in Chapter 3); during state 5 the presence of the k^{th} image is verified against technical issues, in this case the image processing algorithm is launched in order to obtain system state measure. Once this is available, it is possible to calculate y_s and the error e for the next control iteration. If an error occurs during image acquisition, then the algorithm leaves the internal chain of the automaton, otherwise it returns to state 1 for a new control iteration.

More details concerning the implementation of the in-vivo control algorithm can be found in Appendix B.

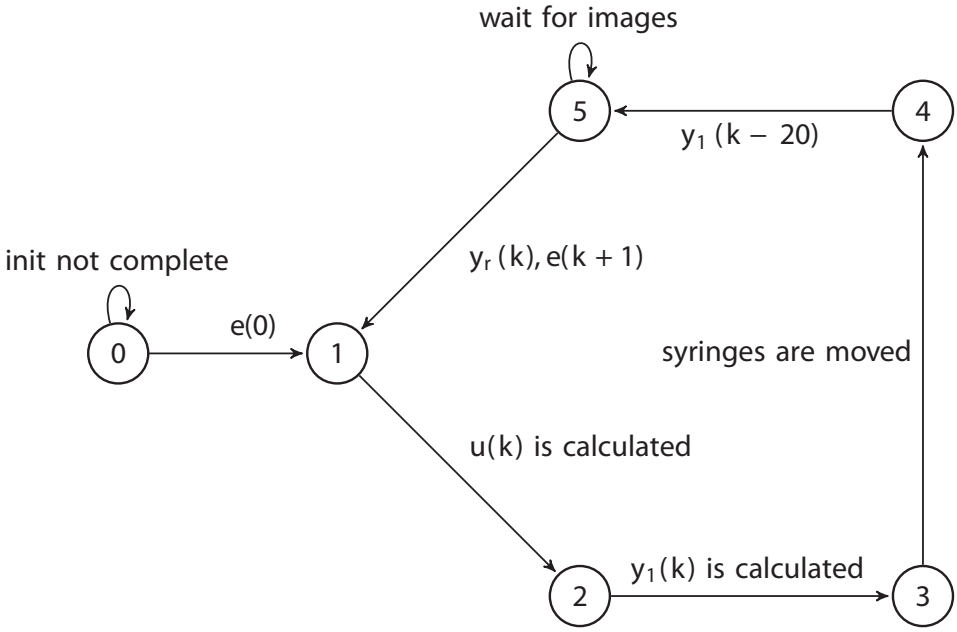

Figure 4.24: **Finite state automaton representing control algorithm.** The error transitions from states 0 and 5 are omitted for the sake of readability. With these transitions the algorithm achieves a new state (also omitted in the picture) where the data handled up to that point are saved, this state is also reached at the end of control.

CHAPTER 5

EXPERIMENTAL RESULTS

Science is facts; just as houses are made up of stones
so is science made of facts; but a pile of stones is
not a house and a collection of facts is not a science.

HENRI POINCARÈ

Contents

5.1 Experimental design

A key issue must be addressed in order to connect the *in silico* control algorithm with the *in vivo* yeast cells: the translation of fluorescence units, as quantified by the image analysis, to the unit of measure of the CBF1-GFP protein used by the in silico control algorithm (Chapter 3).

Several alternative strategies can be envisioned to obtain this result. However, when it comes to biological experiments, limiting, as much as possible, the level of complexity of procedures is of utmost importance. Therefore, I designed the following experimental protocol: during the first

10 hours following the cell loading into the microfludic device, a **calibration experiment** is carried out to measure the fluorescence levels of the yeast cells in Galactose (ON) and in Glucose (OFF). Specifically, cells are grown overnight in a Galactose-supplemented medium using standard protocols; cells are then loaded into the microfluidic device, and once in place, they are exposed for a period of 3 hrs to the Galactose/Raffinose medium, and then switched to Glucose-medium for 7 hrs, as shown in Fig. 5.1.

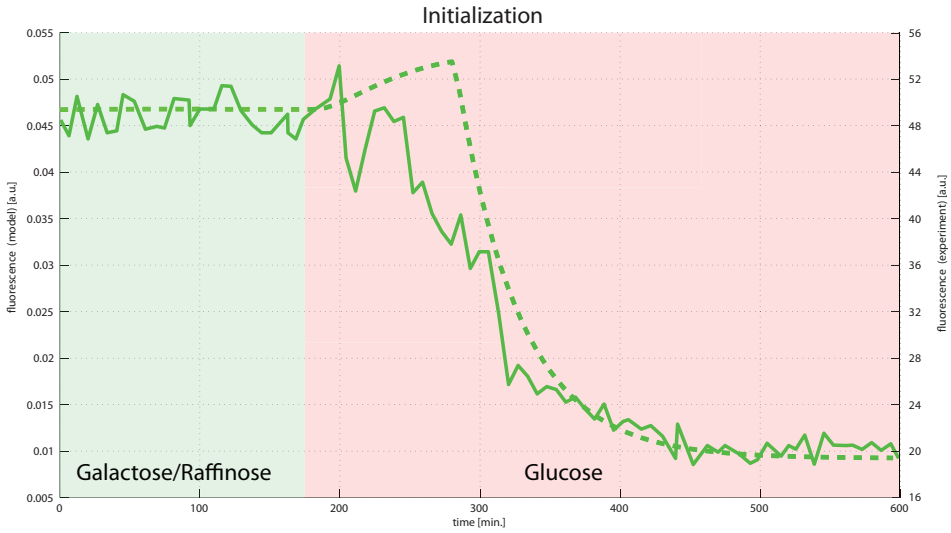

Figure 5.1: **Initialization procedure.** Cells are kept three hours in Galactose/Raffinose supplemented medium and then Glucose is provided for 7 hours. Dashed green line: the amount of fluorescence predicted by the model. Solid green line: response measured by the image segmentation algorithm. The main objective of this step it to find a method to relate these two quantities: the amount of fluorescence predicted by the model and the quantified one.

The calibration phase enables conversion of the experimentally quantified fluorescence levels to the units used by the control algorithm to model the CBF1-GFP concentration.

To this end, the average value of fluorescence across the first 10 samples of the time-series is set as the Galactose steady-state fluorescence level (ON); similarly, the last 10 samples are averaged to obtain the Glu-

cose steady-state (OFF). It is now possible to compare these values to the model derived steady-state values in Galactose and Glucose, and linearly rescale the fluorescence levels into the model unit of measure, as shown in Fig. 5.1. Details of this procedure are reported in Appendix B.

5.2 In-vivo closed-loop control experiments

In what follows, I will describe the results of the in-vivo control experiments carried out with the Smith-Predictor+PI+PWM configuration (described in Chapter 3, section 3.2.4, indeed preliminary experiments showed this to be the most promising approach) and by using the microfluidic device MFD0005$_a$ described in Chapter 4, section 4.2.1.

The control objective was to force the average fluorescence level of the cell population to track a reference signal consisting of a saw-tooth wave, as shown in Fig. 5.2 and simulated in Fig. 3.12.

Specifically, starting from the Glucose steady state, cells were required to reach the 85% of the maximum amount of CBF1-GFP protein they could produce in Galactose, following a linear increase lasting for 1000 min. Once this value is reached, cells are required to linearly decrease the expression of CBF1-GFP towards the Glucose steady-state, in 1000 minutes.

This control task is very demanding, since it is not a simple set-point regulation task, requiring cells to reach a constant expression level of CBF1-GFP, but a rather more difficult task, where cells must track a time-varying reference signal.

Moreover the duration of the experiment is 2000 min ($\approx$ 33 hrs) encompassing more than 15 generations of yeasts cells.

Simulated numerical results for this control task are reported in Fig. 3.12, demonstrating that at least theoretically such a control task can indeed be achieved, with this control strategy.

Experimental results are presented in Fig. 5.2 and in Fig. 5.3.

Some considerations can be derived from the analysis of these experiments. Interestingly the control algorithm does limit the magnitude of the error (in black in Fig. 5.2 and Fig. 5.3) to be very small; this directly translates in a very encouraging result. While the reference signal is accurately tracked during the switch off phase (1000 min to 2000 min), the performance during the first 1000 min of the experiment (switch on phase) is not as precise.

The reason for this difference in the performances can be found by analysing the different signals involved in the control feedback loop, reported in Fig. 5.2 and Fig. 5.3.

The close matching between y_s (the feedback output signal) with the reference signal y_{ref} shows that the controller is working correctly during the whole experiment, keeping the error $e = y_{ref} - y_s$ small.

Therefore, this implies that the pitfalls of this control strategy can be attributed to mismatches between the mathematical model and the actual plant. Namely, the error e depends on the Smith-Predictor signal (y_s), which is a low-pass filtered combination of the actual IRMA output y and of the model predicted $\hat{y}$.

Hence, even if the error $e = 0$, uncertainties in the model (M) will be reflected in imprecision in the quality of the tracking, that is, the measured fluorescence y may be different from y_{ref}. In addition, the Smith predictor is not designed for control tracking problems, as shown in Chapter 3 and in simulation in Fig. 3.12. Nevertheless, the results obtained are quite striking and even better than simulations, at least regarding the offset between the reference signal and the actual fluorescence (compare Fig. 3.12 with Fig. 5.2) due to the delay present in the system. Specifically, we reasoned that this improvement in the actual experiments can be explained only by a much smaller delay than what is estimated in the model (100 min). The biological explanation for the smaller delay can

be due to the epigenetic control of the HO promoter; the network keeps activating this promoter, thus the chromatin stays "open" thus reducing the delay.

Two main potential sources of errors can be found in the proposed control scheme: *(a)* the saturation of the control actuation due to the action of integral component (namely windup effect) and *(b)* dependence on mathematical model precision.

To avoid the effects due to the possible presence of wind-up, a possible solution is to modify the PI scheme as shown in Fig 5.4 (see also [46]). The experimental implementation of this strategy is currently under investigation and will be reported elsewhere. We expect there will be a trade-off between the advantages due to the scheme being independent from the model being used and the absence of any action compensating the presence of delays.

It is worth mentioning here that, in any case, the results reported in this Chapter show that long-term automatic control of gene expression in cell populations is indeed possible and practical.

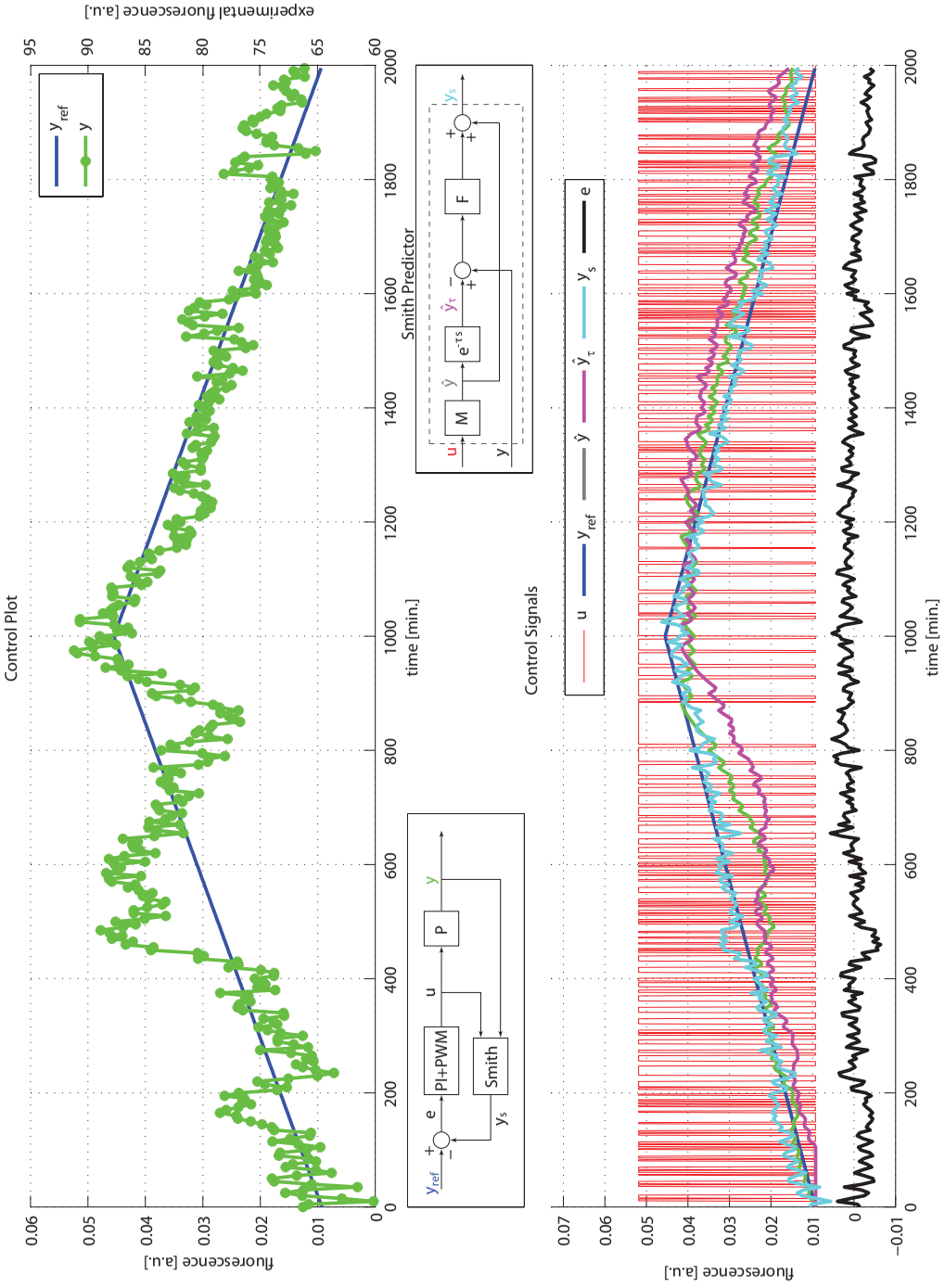

Figure 5.2: **In-vivo signal tracking control experiment #1.** The first plot shows the reference and output signals while the second graph illustrates the dynamics of signals in the control scheme.

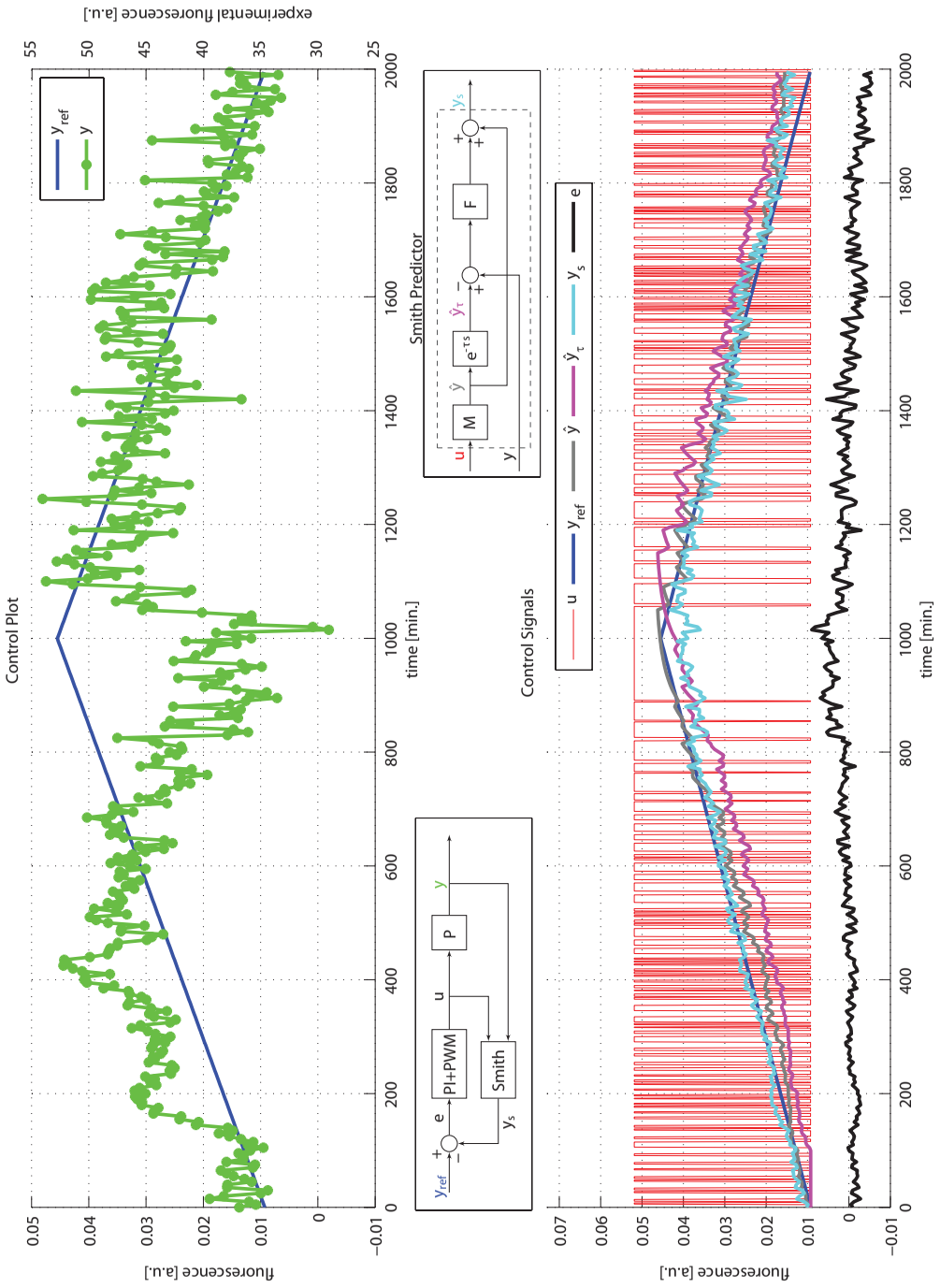

Figure 5.3: **In-vivo signal tracking control experiment #2.**The first plot shows the reference and output signals while the second graph illustrates the dynamics of signals in the control scheme.

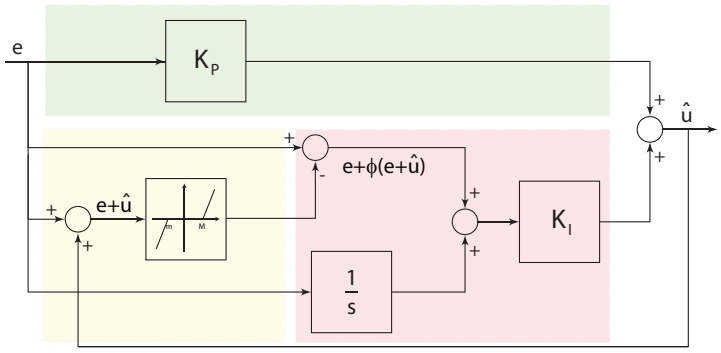

Figure 5.4: **Block diagram of a PI controller with antiwindup based on back-calculation.** For $e + \phi(e + \hat{u}) = 0$, i.e. when m $\leq e + \hat{u} \leq$ M, the controller behaves as explained above since $\hat{u} = K_P \cdot e(t) + K_I \cdot \int_0^t e(t)\, dt$. As a matter of fact the integrator block $\frac{1}{s}$ is meant to integrate all the components of e, but the last one. Whereas, in case $e+\hat{u} > M$ or $e+\hat{u} < m$ (when saturation is likely to have occurred), a correction equal to $-\hat{u}$ is introduced since $e + \phi(e + \hat{u}) = e - (e + \hat{u}) = -\hat{u}$. The colored boxes are meant to highlight parts similarly indicated in Fig. 3.9; the yellow box is the core of the anti-windup scheme.

CHAPTER 6

MODELLING OF A POSITIVE FEEDBACK LOOP CIRCUIT IN A MAMMALIAN CELL LINE

The ideal engineer is a composite. He is not a scientist, he is not a mathematician, he is not a sociologist or a writer; but he may use the knowledge and techniques of any or all of these disciplines in solving engineering problems.

Norbert Dougherty

Contents

Summary

In this chapter we illustrate the process of modelling and analysing of a Positive Feedback Loop synthetic gene circuit developed in CHO cell line. Beyond characterising this regulatory motif, often found in transcriptional and signalling pathways, we show that this motif exhibits a dynamic behaviour very different from that obtained when the autoregulation is removed. This difference is intrinsic to the specific wiring diagram chosen by the cell to control its behaviour (feedback versus non-feedback configurations), and can be instrumental in understanding the complex network of regulation occurring in a cell. Part of this work has been published in [17].

6.1 Engineering a synthetic Positive Feedback Loop

So far I employed IRMA, a synthetic network, as a tool to demonstrate our working hypothesis. Synthetic biology, nevertheless, can be used to uncover the design principles of natural biological systems through the rational construction of simplified regulatory networks[59]. So far, it has been shown that feedback and feed-forward loops are essential regulatory motifs involved in transcriptional regulation[60]. For example, natural circadian clocks consist of feedback loops in which a set of transcriptional activators and repressors are connected by mutual feedback; these clocks are able to maintain a 24-h periodicity in protein expression[61]. How exactly this is achieved is still under debate, but it is an intrinsic property of the network topology.[62]. Therefore, understanding the relationship between topology and dynamics of transcriptional regulatory networks

in mammalian cells is essential to elucidate the biology of complex regulatory and signaling pathways. We aimed at characterising an inducible transcriptional positive feedback loop (PFL) in mammalian cells consisting of the *CMV-TET* promoter, responsive to the Tetracycline-controlled transactivator tTA, driving expression of the tTA protein itself and of a fluorescent reporter protein (Fig. 6.1 panel A).

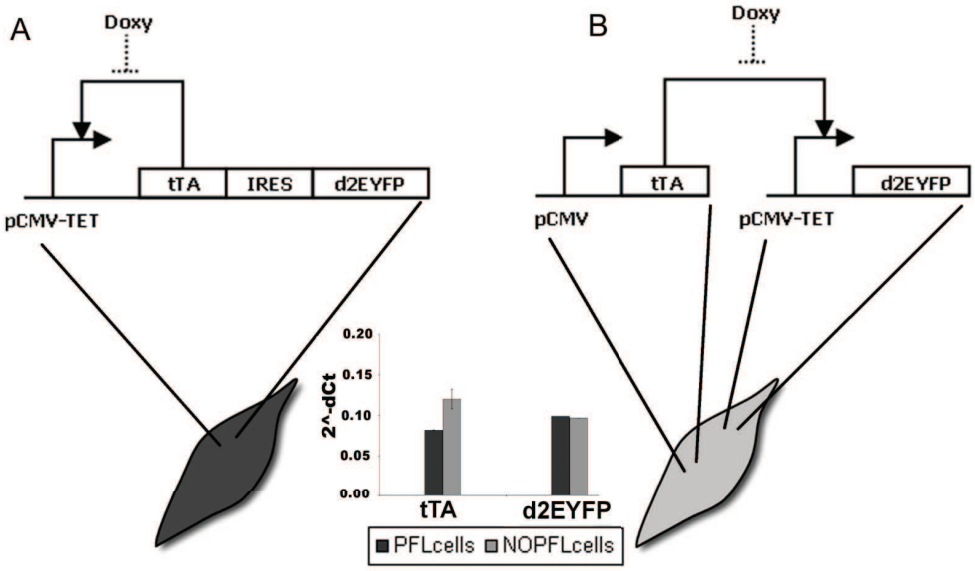

Figure 6.1: **Design of the expression system.** Panel A, PFL : the promoter CMV-TET consists of seven direct repeats of a 42-bp sequence containing the tet operator sequences (tetO), located just upstream of the minimal CMV promoter (PminCMV). The Tetracycline-controlled transactivator tTA derives from the addition of the VP16 activation domain to the transcriptional repressor TetR. The d2EYFP is the destabilised yellow-green variant of enhanced green fluorescent protein. Panel B, NOPFL: the CMV promoter drives the expression of the tTA, which in turns drives the transcription of the d2EYFP from the CMV-TET promoter. (Inset) RealTime PCR performed on DNA extracted from PFL and NOPLF cells shows that the DNA levels of tTA and d2EYFP are comparable among the two clonal cell populations.

Most of the studies carried out so far in mammalian cells are based on plasmid transfection, which prevents precise quantitative measurements due to the unpredictable amount of plasmids that enters each cell, and to the transient nature of transfection. To overcome these limitations,

we generated a clonal population of Chinese Hamster Ovary cells (CHO) carrying a stably integrated version of the PFL, which were then used to evaluate the dynamical properties of this transcriptional motif. We also performed control experiments by generating a clonal population of cells lacking the positive feedback loop (NOPFL), as shown in panel B of Fig. 6.1. In vivo quantitative measurements of fluorescence intensity in time, following addition of the inducer molecule (Doxycyline) able to *switch off* the network, were used to fit an Ordinary Differential Equations (ODE) model of the PFL and NOPFL networks. The models were able to reproduce the experimental data, and, interestingly, highlighted differences in the dynamic properties of the PFL versus the NOPFL networks, which are due to the intrinsic differences in the two network topologies. It has been suggested theoretically that the PFL motif can slow down response times of transcriptional regulatory networks [63]. We experimentally demonstrated that this is the case, and we reported for the first time a quantitative characterisation of a transcriptional positive feedback loop in mammalian cells, which can be instrumental in better understanding the properties of natural occurring transcriptional and signaling networks.

6.2 Construction of the inducible positive feedback loop (PFL) and of the corresponding control network (NOPFL)

Our approach is based on the use of well known and characterised regulators of gene expression, in order to achieve a complete control of the network behaviour. The positive feedback loop (PFL) is shown in panel A Fig. 6.1. In particular, we took advantage of the inducible Tet regulatory system; the expression of Tetracycline-controlled transactivator tTA is self-controlled by a *CMV-TET* promoter, responsive to the tTA itself

unless the Tetracycline, or its analogous Doxycycline, is added to the medium in which cells are grown [64]. To follow the dynamics of the PFL, a destabilised yellow-green variant of the enhanced green fluorescent protein (d2EYFP) (Clontech), with a reported half-life of approximately two hours, had to be placed under the control of the same promoter. To this end, we constructed a unique cassette with an Intra Ribosomal Entry Sequence (IRES) in between of the transactivator tTA and the d2EYFP, which enables a single mRNA to encode for two different proteins (panel A in Fig. 6.1).

In order to stably express in CHO cells the PFL network, and to generate a clonal population, we inserted the cassette in panel A of Fig. 6.1 in a lentiviral vector [65][66]. Infected cells were first sorted by Fluorescence Activated Cell Sorter (FACS) and then a clonal population of CHO cells carrying the PFL was generated by single cell expansion (PFL cells).

To capture the dynamic properties intrinsic to the PFL, we needed to generate a control network lacking the positive feedback loop (NOPFL), but using the same biological *parts* as in the PFL network. As shown in panel B of Fig. 6.1, we constructed a cassette containing the same *CMV-TET* promoter upstream of the d2EYFP. The tTA protein, this time, was placed under the control of a constitutive promoter, thus breaking the feedback loop. Using the same strategy described above, we generated a clonal population of CHO cells carrying the NOPFL network (NOPFL cells). We experimentally verified that both PFL and NOPFL cells have the same number of tTA/d2EYFP DNA integrations (Fig. 6.1 inset).

6.3 Experimental investigation of the dynamic behaviour of the PFL and NOPFL networks

To observe the dynamics of the PFL and NOPFL networks, we performed time-series experiments in which stably-integrated CHO-PFL cells and CHO-NOPFL cells were imaged using time-lapse fluorescence microscopy. The experimental design consisted in treating both PFL and NOPFL cells with different amounts of Doxycycline in order to *switch off* the circuit, by preventing the tTA protein from binding the *CMV-TET* promoter. We tested the following Doxycyline concentrations: 1ng/mL, 10ng/mL, 100ng/mL and 1μg/mL and followed the dynamic behaviour of both the PFL and NOPFL cells for 43h, collecting images every 15 min, and quantifying the average fluorescence intensity of the cell population (Image acquisition and processing). In this way, we averaged out cell-to-cell variability in the response, since at the beginning of each experiment the tracked microscopy field contained at least 15 cells.

Experiments were carried out at a temperature of 32 °C in order to limit cell motility and reduce the risk associated to data loss occurring when cells exit the tracked field [67]. The average fluorescence intensity of the reporter gene across the cell population for both the PFL and NOPFL networks is shown in Fig. 6.2 for the different concentrations of Doxycycline indicated.

The most striking feature is the slow down in the switch off time of the PFL cells as compared to the NOPFL cells; moreover the switch off time of the PFL is affected by Doxycycline concentrations, whereas NOPFL cells always switch off with approximately the same dynamics. We derived a model of the PFL and NOPFL networks using Ordinary Differential Equations (ODEs). ODEs are commonly used to describe the average behaviour of a population of cells [68] and have been shown

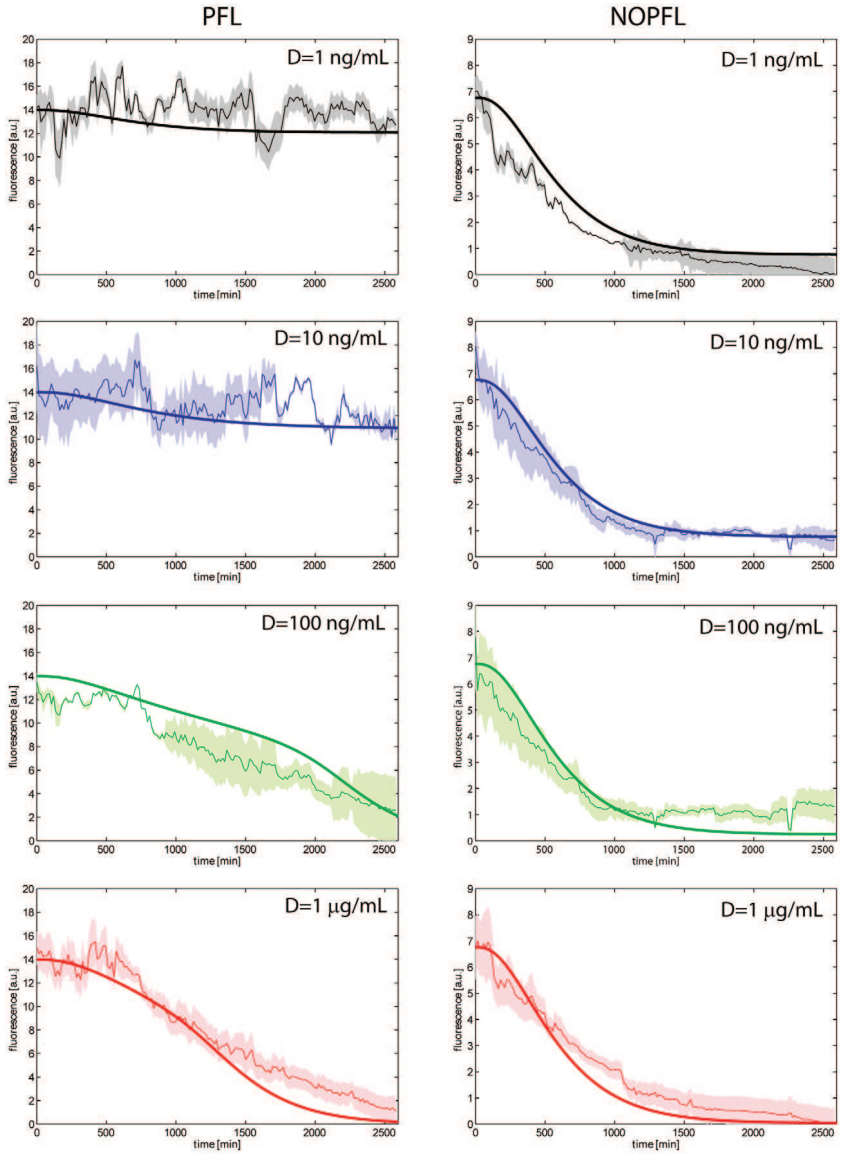

Figure 6.2: **Experimental and simulated switch off time-course across the PFL and NPFL cell population.** Experimental data (thin lines) and model simulations (thick lines) were reported for the PFL (left) and NOPFL (right) cells. Shaded areas represent standard deviations from replicate experiments.

to be appropriate for the analysis of synthetic networks in a number of cases ([69], [70], [71], [72], [73], [3]). In our settings, the use of such a modelling approach is valid, since we are measuring the average behaviour

of a clonal population of cells.

For each species, i.e. each mRNA and correspondent protein concentration, we wrote one equation, which expresses the change in concentration of the species in a given time interval, as the result of a production term and a degradation term. We assumed:

1. Hill functions to model the rate of gene transcription, including basal activity to describe the leakiness of the *CMV-TET* promoter;
2. linear degradation for all genes and proteins;
3. linear dynamics for the translation;
4. Hill functions to model the effect of the inducer (Doxycycline);
5. distinct dynamics for the unfolded (inactive) and folded (active) forms of the reporter protein (d2EYFP).

The last assumption was introduced in order to take into account d2EYFP maturation time needed for correct protein folding[72]. Thus, we introduced two differential equations as in [72]: one for the translation of mRNA to the unfolded d2EYFP protein, and one for the folded protein d2EYFP.

Letting x_1 be the *tTA* IRES *d2EYFP* mRNA concentration, x_2 the tTA protein concentration, x_3 the unfolded d2EYFP protein concentration and x_4 the folded d2EYFP protein concentration, the PFL network can be described as follows:

$$\frac{dx_1}{dt} = v_1 \left(\alpha_1 + (1-\alpha_1) \frac{\left(\frac{\theta^{h_2}}{\theta^{h_2}+D^{h_2}} x_2 \right)^{h_1}}{K_1^{h_1} + \left(\frac{\theta^{h_2}}{\theta^{h_2}+D^{h_2}} x_2 \right)^{h_1}} \right) - d_1 x_1, \quad (6.1)$$

$$\frac{dx_2}{dt} = v_2 x_1 - d_2 x_2, \quad (6.2)$$

$$\frac{dx_3}{dt} = v_2 x_1 - (d_3 + K_f) x_3, \quad (6.3)$$

$$\frac{dx_4}{dt} = K_f x_3 - d_3 x_4. \quad (6.4)$$

Note that, due to the presence of the IRES sequence, the concentrations of tTA protein and d2EYPF protein depend on the same variable (x_1), that is the concentration of the single mRNA transcript encoding for both proteins.

For the NOPFL network, we let x_1 represent only the *d2EYFP* mRNA concentration, and we assumed a constant level ($\bar{x}_2$) of the tTA protein, due to the constitutive promoter driving *tTA* expression in the NOPFL cells. The equations thus become:

$$\frac{dx_1}{dt} = v_1 \left(\alpha_1 + (1-\alpha_1) \frac{\left(\frac{\theta^{h_2}}{\theta^{h_2}+D^{h_2}} \bar{x}_2 \right)^{h_1}}{K_1^{h_1} + \left(\frac{\theta^{h_2}}{\theta^{h_2}+D^{h_2}} \bar{x}_2 \right)^{h_1}} \right) - d_1 x_1, \quad (6.5)$$

$$\frac{dx_3}{dt} = v_2 x_1 - (d_3 + K_f) x_3, \quad (6.6)$$

$$\frac{dx_4}{dt} = K_f x_3 - d_3 x_4. \quad (6.7)$$

6.3.1 Experimental determination of the reporter protein degradation

We experimentally evaluated the degradation rate of the reporter protein (d2EYFP) since this is a key parameter which affects the observed fluorescence dynamics. To evaluate d2EYFP degradation rate, stably

integrated NOPFL cells were treated with Cyclohexamide to a final concentration of 10μg/mL, 50μg/mL, 100μg/mL or 500μg/mL, to inhibit protein synthesis[74]. The fluorescence intensity of NOFPL cells was followed for 12 hrs and images were acquired at 15 min intervals. The resulting d2EYFP dynamics are shown on Fig. 6.3 and appear very similar, independently of the Cyclohexamide concentrations. The experimental data were fitted to an exponential curve $ke^{-d_3 t}$, and the degradation coefficient d_3 was used to obtain the half-life ($\tau_{\frac{1}{2}}$) of the d2EYFP protein: $\tau_{\frac{1}{2}}$=log(2)/ d_3 (Fig. 6.3 and Table 6.1).

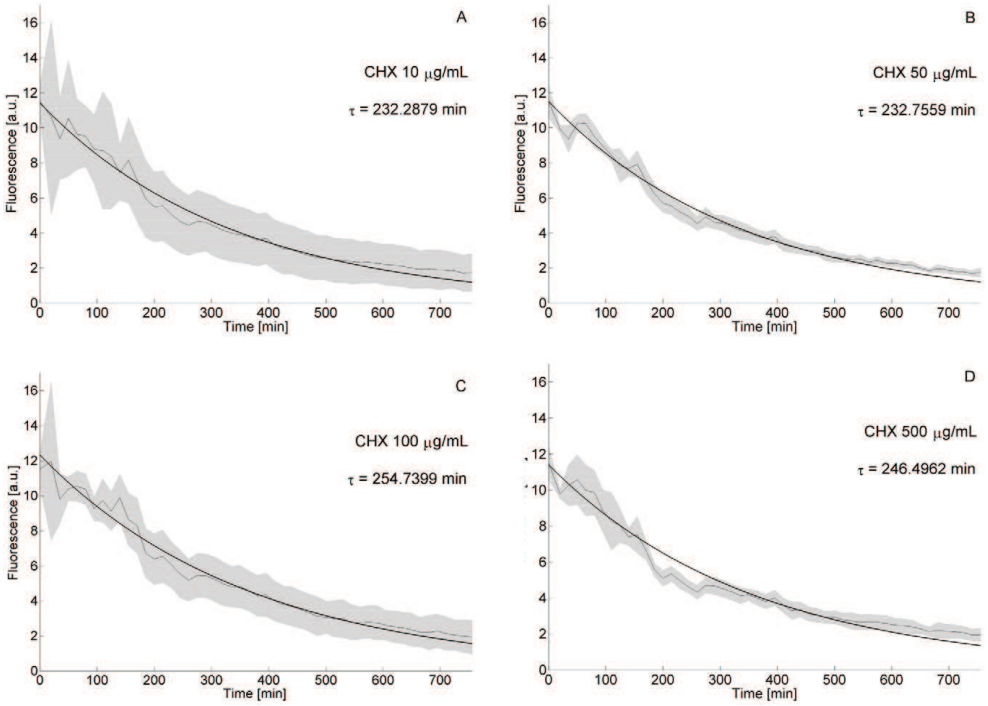

Figure 6.3: **Degradation kinetics of d2EYFP.** *CMV-TET*-d2EYFP stably integrated CHO AA8 TET-OFF cells were treated at t=0 with different concentrations of Cyclohexamide (CHX): panel A, 10μg/mL; panel B, 50μg/mL; panel C, 100μg/mL; panel D, 500μg/mL. Fluorescence intensity was followed up to 750 minutes. Sampling time is equal to 15 min. The thin line represents the mean over biological triplicates; the shaded area represents the standard error. Experimental data were used to fit the exponential decay of d2EYFP protein levels, and thus to derive its half-life (τ). Fluorescence intensity in untreated cells was not subjected to any significant decay (data not shown).

We estimated $\tau_{\frac{1}{2}}$ to be in the range 3.6h-4.4h. The estimated value is about two-fold the reported d2EYFP half-life of 2h [75]; we believe that this discrepancy is likely due to the fact that cells were grown at a temperature 32 °C, rather than the usual 37 °C.

Table 6.1: Parameters identified after the fitting procedures: parameters values as well as standard deviation are reported for each parameter.

Parameter	Description	Fitted value	STD
K_1 [nM]	Activation coefficient	4.81	1.06
α_1	Basal activity CMV-TET promoter	1.13E-05	3.62E-05
v_1 [nM min^{-1}]	Maximal transcription rate CMV-TET promoter	7.54E-02	1.97E-02
v_2 [min^{-1}]	General translation rate	2.71E-02	1.22E-02
d_1 [min^{-1}]	Degradation rate tTA mRNA	1.01E-02	1.22E-03
d_2 [min^{-1}]	Degradation rate tTA protein	1.00E-02	3.42E-03
d_3 [min^{-1}]	Degradation rate d2EYFP protein	3.24E-03	2.66E-04
h_1	Hill coefficient of the CMV-TET promoter	3.16	1.40E-01
θ [nM]	Affinity Doxycycline - CMV-TET promoter interaction	1.00	8.85E-03
K_f [min^{-1}]	Folding rate d2EYFP	1.24E-03	1.41E-02
$\bar{x}_2$ [nM]	Steady state tTA in NOPFL	13.69	7.63E-01
h_2	Hill coefficient for Doxycyline	6.03E-02	7.19E-03

6.3.2 Image acquisition and processing

Images were acquired using an inverted epifluorescence microscope (Nikon Eclipse TI-E, Nikon Instruments) equipped with an incubation chamber (H201-OP R2 ,Okolab), a digital camera (Andor Clara, Andor), a 20X objective (Obj. CFI PF DLL 20X Ph1, Nikon) and a 512-nm/529-nm (B/G/R) d2EYFP-specific excitation/emission filter set. Temperature was maintained at a constant level as the experimental setup required, while CO2 concentration was set to be 5% of the total air volume injected in the incubation chamber. Both phase-contrast images and fluorescent fields were acquired at intervals of 15 minutes. Exposure times for the phase-contrast field was set to 2 ms (transmitted light lamp volt-

age was set to 4.5 V) while 300 ms (Intensilight lamp set at 10% of the maximum power) was chosen as exposure time for the fluorescent images: this choice was meant to prevent photobleaching while optimising the ratio between the quality of the images and reflected-light-induced stress on the cells. Experiments were carried out using NIS-Elements AR v.3.10 644 (Nikon Instruments) software package and the Perfect Focus System (Nikon Instruments) to maintain the same focal plane during the whole duration of the experiment. At the end of the acquisition process, images were extracted as raw data for the fluorescence quantification procedures. The experiments were set up so that at the beginning of each experiment the first image contained at least 15 cells and no more than 30 cells, to avoid cells exiting the image during the time-lapse experiment due to cell replication and "over-crowding". Image segmentation was carried out in Mathworks Matlab R2010b (Mathworks Inc.); the algorithm we implemented to quantify fluorescence was meant to distinguish the foreground (living cells) from the background in each image of the bright field. We used morphological operators such as erosion and dilation (*imerode* and *imdilate* functions from the MATLAB image processing toolbox). Thus two binary masks were built in order to compute separately the mean d2EYFP fluorescence of the foreground and the background using an element by element matrix multiplication between the binary images and the fluorescent one. The average fluorescence intensity across the cell population was then computed as the difference between the foreground and the background for each image at each time point (i.e. no single cell fluorescence quantification is performed).

6.4 Derivation of the model parameters

The parameter estimation problem can be defined as an optimisation problem, where the goal is to minimise a performance measure defined as the error between model predictions and observations, which in our case

are the time-series during the "switch off" experiments in Fig. 6.2. In our setting, there are 12 parameters to be fitted, 11 of which are common to both the PFL and NOPLF models (Table 6.1).

Several alternative strategies can be pursued in order to obtain best estimates of the model parameters, ranging from Newton's method to Genetic Algorithms (GAs). In this work we employed the Trust Region method (TRM) implemented in PottersWheel [76]; thanks to the multi−model and multi−experimental capabilities of this tool, we were able to identify the 12 parameters by simultaneously fitting Eqs. (1) to (7) to all of the experimental time-courses at once. These time-courses include all of the different Doxycycline concentration for both the PFL and NOPFL cells for a total of 24 time-courses, when taking experimental replicates into account.

In order to estimate confidence intervals for each parameter, we run multiple times the TRM identification procedures on the same experimental time-series data, using parameter perturbation routines to allow extensive sampling in the parameters' space.

In order not to introduce any specific bias in our search, we only set admissible ranges for the parameter values to be identified, which reflected physical and biological feasibility, either obtained from literature [71] or directly estimated (degradation rate of $d2EYFP = d_3$). The result of the parameter fitting procedure are reported in Table 6.1 along with the estimated standard deviation, which are in most of the cases one order of magnitude less than the corresponding parameter value, or at most of the same order of magnitude.

We observed that the paramter h_2 in Table 6.1, which affects the strength of Doxycycline repression on the tTA protein activity, is much smaller than 1. Usually Hill coefficients are greater than 1, therefore we wondered what could be a possible biological explanation for this behaviour. We observed that for the range of Doxycycline concetration used

in the experiment (1ng/mL to 1μg/mL), using the parameters' values in Table 1, the function: $\frac{\theta^{h_2}}{\theta^{h_2}+D^{h_2}}$ can be approximated by the function: $\alpha + \beta\frac{\theta}{\theta+D}$ ($\alpha \approx 0.4$ and $\beta \approx 0.6$). This means that a Michealis-Menten function can also describe the effect of Doxycyline on tTA acitivity, but a certain level of leakiness (α) must be taken into account; that is even for large concentrations of Doxycycline, the activity of the tTA protein cannot be completely shut down.

The *switch off* time-series experiments were simulated with both the PFL and NOPFL models using the fitted parameters as shown in Fig. 6.2. The inferred models are able to recapitulate the observed dynamics in response to different inducer concentrations and experimental settings. Observe that the parameters for both the PFL and NOPFL models are identical, except for $\bar{x}_2$ in the NOPFL equations, which is not present in the PFL model. Hence, the observed differences in the dynamical behaviour of the PFL and NOPFL networks are due to the intrinsic differences in their topology, and are robust to changes in parameters values, as demonstrated in the next section.

6.4.1 Model simulations and parameter identification

Numerical simulations were run using Matlab 2010b (Mathworks Inc.). We used *ode23s* solver (a detailed discussion of the numerical methods used by *ode23* can be found in [77]). For the parameter identification, we used the PottersWheel toolbox [76] implemented in MATLAB. Two sets of parameters were identified: the dynamical parameters governing the model and a scaling factor meant to approximate the transduction contribution of the microscopy equipment. Since Doxycycline has been only added at time $t = 0$ min in our experiment we forced the fitting procedures to start from the model predicted ON steady state.

We defined the following objective function:

$$\chi^2 = \sum_{i=1}^{N} \frac{(y_{Exp}(i) - y_{Model}(i))^2}{\sigma_{Exp}(i)^2} \tag{6.8}$$

where N is the number of experimental data points, y_{Model} are the predicted values of the mathematical model (using the inferred parameters), y_{exp} are the experimental data points and $\sigma_{Exp}(i)^2$ is the sample variance computed over the experimental replicates.
As optimisation algorithm we used Trust Region Method (TRM) in a logarithmic parameter space: at the k^{th} iteration of the optimisation procedure the TRM approximates the shape of the function f to be minimised with the model m_k thus trying to solve the following problem:

$$\min_{p} m_k(x_k + p_k) \tag{6.9}$$

being $x_k + p_k$ the new parameter vector considered as solution at the k^{th} iteration. If the model m_k has quadratic form the vector p_k can be obtained by observing that:

$$m_k(x_k + p_k) = f_k + p_k^T + \nabla f_k + \frac{1}{2} p_k^T B_k p_k \tag{6.10}$$

and therefore

$$p_k = -\nabla_k \frac{\nabla f_k}{\parallel \nabla f_k \parallel} \tag{6.11}$$

In order to allow an extensive exploration of the parameters' space, and to avoid local minima, we used a quasi-random number generator routine in PottersWheel [76] to select an initial guess of the parameters' values, and then launched the TRM procedure M times (M=100 in our settings), requiring the cost function in eq. 6.8 to be $\chi^2/N \leq 0.5$ [76].

The values in Table 1 represent parameters for which the cost function (eq. 6.8) is the smallest across the M runs; whereas the standard devation of each parameter in Table 1 is evaluated by considering all of the M runs.

Moreover, in order to compare switch off times among the different experiments, we computed the τ_{off} defined as the time the circuit needed

to achieve the 50% of the mean initial fluorescence MIF calculated for each experiment as follows:

$$MIF = \frac{\sum_{i=1}^{3} f_i}{3} \tag{6.12}$$

with f_i fluorescence of the i^{th} frame in the sequence smoothed by moving average filtering.

6.5 Dynamic properties of the PFL and NOPFL networks

In order to further investigate the relationship between topology and dynamical properties, we first observed that the NOPFL model described by Eq.6.5-6.7 is a system of linear time-invariant ODEs, for which the theory of liner dynamical systems applies. From the theory, we know that changes in Doxycycline concentration in Eq.6.5 will not affect the dynamic behaviour of the model, which is governed by the smallest among three degradation terms $d_1, d_3, (d_3 + K_f)$. The concentration of Doxycycline affects only the steady-state values, i.e. how much the network will switch off, but not its dynamics, i.e. how fast it will switch off. Therefore, independently of the values of the parameters, the model of the NOPFL network predicts that for any concentration of Doxycycline, the network will switch off with the same dynamics, albeit possibly reaching different steady-state levels.

Fig. 6.4 reports the *switch off* time, τ_{off}, for both the PFL (dashed) and the NOPFL (solid) networks as a function of Doxycycline concentration, computed via numerical simulations of the two models with the parameters estimated in Table 6.1.

τ_{off} is defined as the time taken by the fluorescence intensity to reach 50% of its final steady-state value (OFF), following treatment with Doxycycline at a given concentration. As expected, the τ_{off} for the NOPFL

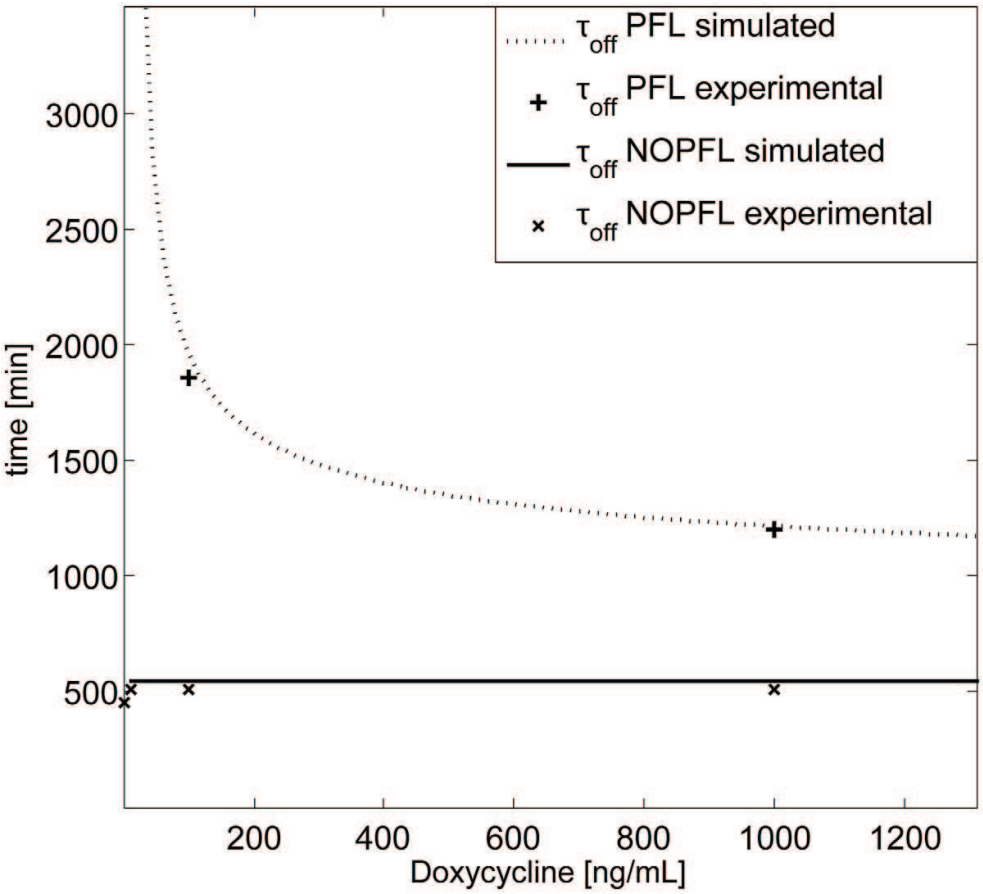

Figure 6.4: **Switch off time τ_{off} for varying Doxycycline concentrations from experimental data and model predictions.**The model predictions for the switch off times τ_{off} are shown for PFL (dashed thick line) and NOPFL (solid line). Experimental quantification of the τ_{off} for PFL and NOPFL models have been reported for comparison with + and × respectively. Observe that the experimental τ_{off} for the PFL at 1ng/mL and 10ng/mL could not be estimated since the PFL is not switching off in the experimental observation time (43h).

network is constant and does not change with Doxycyline. This is in agreement with the experimental observations; in Fig. 6.4, the switch off time for the NOPFL network for the different concentration of Doxycycline was estimated from the experimental time-series data (× in Fig. 6.4).

On the other hand, the PFL network has a very different behaviour, as can be seen in Fig. 6.4. Specifically, for a range of Doxycycline con-

centrations, the PFL τ_{off} is considerably longer (+ in Fig. 6.4) that the NOPFL counterpart, which again is in agreement with the experimentally observed behaviour. In order to investigate the origin of the observed dynamical behavior of the PFL circuit, we analysed the phase portrait associated to the *d2EYFP−tTA mRNA* and the tTA protein, which allows to directly observe trajectories of two state variables at once. Moreover, by imposing the steady-state conditions (i.e. $\dot{x}_i = 0$), we can derive nullclines, as well as, the their intersection points, which correspond to the equilibrium points of the network. In Fig. 6.5 the nullclines for different Doxycycline concentrations are shown.

When no Doxycycline is present, two stable points (OFF and ON) and one unstable equilibrium points coexist in the same phase portrait, thus providing evidence for the bistability of the PFL network, a shared property among positive feedback loops[78]. However, as Doxycycline concentration increases, the bistability is lost (Fig. 6.5), and the only possible equilibrium point is the *OFF* state.
Moreover, by studying how the nullclines change (as the concentration of Doxycycline varies), it is possible to gain useful hints on the bistability that can be easily verified with, e.g. straightforward FACS experiments. As can be seen in Fig. 6.6, for the lowest Doxycycline concentration ($1\ ng/mL$) the OFF state basin is much smaller than the one leading to the ON state (panel A in Fig. 6.6, red dots represent initial conditions ending up in the OFF state, blue dots initial conditions having ON state as final point). This situation dramatically changes as the Doxycycline concentration increases: the ratio between the amplitudes of the two basins of attraction becomes close to 1 in the case of $10\ ng/mL$ Doxycycline concentration. For higher inducer concentrations the number initial states ending up in the ON state suddenly decreases thus giving rise to monostability of the circuit.

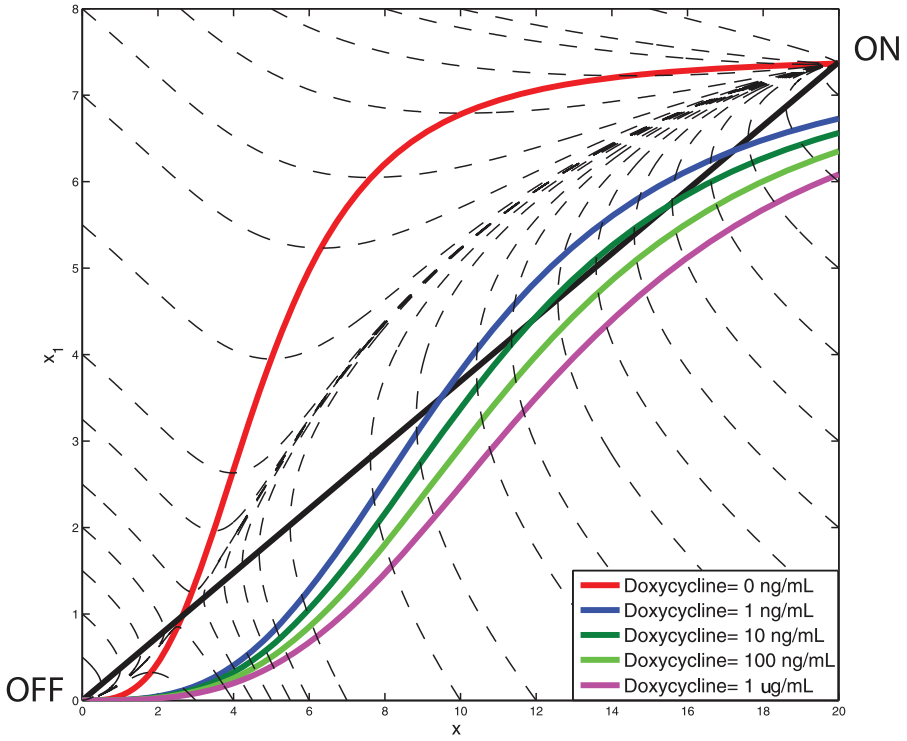

Figure 6.5: **Phase portrait of the PFL model.**The tTA-d2EYFP mRNA concentration (y axis) has been plotted against tTA protein concentration (x axis). Varying Doxycycline concentrations (1 ng/mL through 1 μg/mL) were used to investigate the dependence of the two stable equilibria (*ON* and *OFF* in the graph) on the amplitude of the input. The shape and dimensions of the two basins of attraction (the set of initial conditions ending up in one of the two stable steady states) can be studied with the same technique: in this figure the grey shaded area represents the basin of attraction of the *OFF* equilibrium for Doxycycline= 0 nM.

6.6 Final considerations

I have demonstrated, in a mammalian experimental system, that a transcriptional positive feedback loop can slow down the *switch off* times, as compared to an equivalent network without auto−regulation. The reason for a cell to *choose* a PFL control strategy for transcriptional regulation, rather than the NOPFL strategy, could be due to the intrinsic

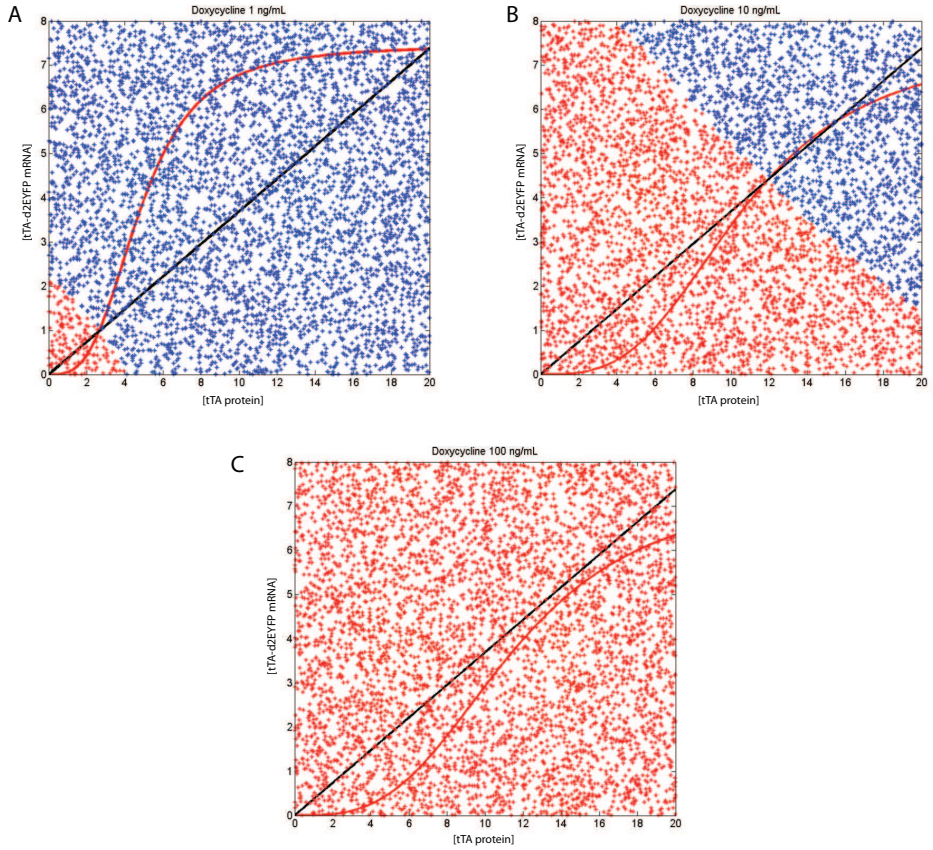

Figure 6.6: **Basins of attraction of the PFL circuit at varying Doxycycline concentrations**. tTA -d2EYFP mRNA concentration (y axis) is plotted against tTA protein concentration (x axis). Multiple simulations have been run for each of the three selected inducer concentrations (1 ng/mL, 10 ng/mL and 100 ng/mL. Each point represents an initial condition and it has been colored in red if, subjected to the specific Doxycycline concentration, the system ended in the OFF steady state. In blue, points ending up in the ON steady state are indicated.

robustness of this approach to transient activation of the network. For example, in a signalling pathway, a ligand (equivalent to Doxycycline in our PFL) could cause a transcription factor to stop transcribing itself, as well as, a set of target genes, to initiate a specific response. However, in order for the pathway not to respond to a transient concentration of the ligand, the PFL strategy has to be chosen, otherwise the response would start immediately (NOPFL case). Moreover, the response time of

the PFL network can be modulated by the ligand concentration, if this is really high, the system will switch off as quickly as possible (Fig. 6.5), alternatively the ligand can be present at low, or medium, concentration, but it should persist for a long time, in order for the pathway to respond. This kind of behaviour has been recently described as *persistence detection* in cellular signal processing to indicate the ability of the genetic circuit to distinguish between transient and persistent signals [79].

Interestingly, it has been shown in E. coli[80] that the transcriptional negative feedback loop (NFL) has an opposite effect, that is, it can significantly speed up the rise-times of transcription, but has very little effect on the switch-off times. From numerical simulations, we verified that for the PFL the slow down effect is only in the switch off times, whereas rise-times are barely affected (data not shown).

The duality between positive and negative feedback has been predicted in a biological setting[63], and it is a well established concept in *control engineering*, a branch of engineering which deals with the design of automated mechanisms to control a variable of interest (the altitude of an airplane, or more simply, the temperature of a room via thermostat)[81]. Specifically, the negative feedback loop is a classic control engineering approach to speed up the response times of a sytem, thus quickly achieving a desired value of a variable of interest. Positive feedback loops, instead, can slow down the response of the system to external input, and are used by control engineers to build *memory* elements, also known as switches, which are able to be in one of two steady-states (OFF or ON), and which are robust against unwanted transient perturbations that may inadvertently switch off (or on) the system.

We indeed verified that the PFL can exhibit bistability for zero or low concentrations of Doxycyline (Fig. 6.5). A bistable genetic network will cause a population of cells to divide in two sub-populations, each in one of the two possible states (OFF or ON). In yeast, this has been ex-

perimentally verified using a simple PFL based on the rtTA system[78]. In our mammalian PFL, this behaviour was not detected experimentally (FACS analysis data now shown). This can be easily explained by observing that the PFL model is bistable but the basin of attraction of the OFF equilibrium point (Fig. 6.5) is much smaller as compared to that of the ON state, when no Doxycycline is present. Therefore, just few cells will be in the OFF state and these will not be enough to be significantly detected experimentally. We predict however that for intermediate concentration of Doxycycline (100ng/mL in Fig. 6.5) the basin of attraction will be comparable and bistability should be detected experimentally. We believe our work can be instrumental to characterise the behaviour of naturally occurring regulatory pathways in mammalian cells and to prove that, despite the complexity of these pathways, they may be understood relying on the knowledge of the behaviour of simplified regulatory network as the one here reported.

CHAPTER 7

CONCLUSIONS AND FINAL REMARKS

Permitte divis cetera.

Quinto Orazio Flacco

Contents

In this Thesis, I demonstrated how modelling and control of synthetic gene networks can be achieved by using principles from control and systems theory. The first step consisted in analysing the expected challenges and in looking for suitable solutions to them. The first choice made in this context concerned the selection of a suitable testbed to prove real-time in-vivo controllability of gene networks in a cell population. I needed a benchmark that had the following properties: *(a)* a reasonable level of complexity *(b)* an available differential equation model *(c)* decoupled from the rest of endogenous gene circuitry. As described in Chapter 2, IRMA satisfied these properties. IRMA is synthetic gene network built in S. cerevisiae, thus a eukaryotic model system, made up of five yeast genes completely rearranged in their regulatory network topology to avoid any cross talk with wild type genome. A non-linear, hybrid, time-delayed mathematical model for IRMA has been re-derived to test the effectiveness of the designed control strategies in-silico. Several alternative strategies have been designed, tested and presented in Chapter 3 to address the problem at hand. A major goal I struggled to achieve during control system design was to minimise the level of complexity of the proposed strategies: the risk in ignoring this lays in obtaining solutions showing astonishing results in-silico but failing to display the robustness required in real-world experiments. Therefore, the proposed schemes spanned from the simple relay based control to the Smith Predictor based configuration that includes a proportional-integral controller and a Pulse-Width-Modulation strategy. Designs, peculiarities and performances of these control schemes have been exposed in Chapter 3.

In order to translate these designs in in-vivo control strategies, I proposed and implemented a new technological platform presented in Chapter 4. As previously outlined, a paradigm shift was needed in order to demonstrate the working hypothesis. As a matter of fact, flask-based experiments could not be used to accomplish real-time in-vivo control

experiment because of the several limitations (theoretical and practical) listed in Chapter 4. Therefore I decided to take advantage of the latest achievements in the field of microfluidics. Device design, optimisation and fabrication has been presented together with the results of a collaboration with Prof. Jeff Hasty at University of California at San Diego. During my visit at Prof. Hasty's laboratory in late 2009, I have been able to acquire technical expertise that allowed me to re-engineer their platform in Dr. di Bernardo Laboratory at TIGEM as described in Chapter 4. Once the platform was set up, I carried out experimental procedures to obtain in-vivo real-time control whose results have been reported in Chapter 5.

At the same time I could apply principles from control theory to the identification of a well-known gene network motif, namely a Positive Feedback Loop, in a mammalian cell line. The results of this study have been presented in Chapter 6 with considerations on dynamical properties of this simple circuit and some hints at potential advantages it can confer to cells.

All these results, taken together confirm that control theory principles can be successfully applied to both the identification and control of gene regulatory networks and open new roads for research in the fields of systems and synthetic biology.

7.1 Future works

The development of a control strategy for synthetic gene networks allows unprecedented opportunities in the fields of systems and synthetic biology that span from model refinement through optimal experimental design to the development of adaptive algorithms that take into account change in parameters due to modifications in environmental conditions. IRMA model refinement is, for sure, the first and more natural extension of this work. As demonstrated in Chapter 5, obtaining better bi-

ological models would significantly improve the performances of model based control strategies. I believe that the technological platform I developed and described in this work will play a key role in this field: not only microfluidics/time-lapse microscopy based strategies will allow us to perform highly informative experiments, but it will make us able to investigate single cell behavior more directly as well. Our ability to effectively and quickly change, with outstanding precision, the extracellular environment is central towards the accomplishment of both these results. Novel hypotheses on the stochastic nature of gene networks, as well as on the role of biological noise in cell decision making, will be easily tested in-vivo or in-vitro.

Our improved ability to finely modulate inducer compounds, thanks to microfluidics, will generate a side effect not to be underestimated: the identification of gene network will be likely to become much easier as a result of the application of advanced system identification theory to biology. As a matter of fact, as outlined in [16], Optimal Input Design strategies for complex gene networks have been already designed and tested in silico and only need in-vivo/vitro implementation. As a result of this, experimental costs associated to gene network identification will decrease and precise reverse engineering will become feasible for larger and larger biological networks.

As mentioned above, the availability of improved mathematical models will allow us to apply more advanced control techniques like Model Predictive Control and Adpative or Robust Control that will likely iterate the process of opening new horisons for application of control theoretic principles to biology. In this context I consider the development of a formal theory of controllability of systems being characterised by the dynamical properties we referred to in Chapter 2 (strong non-linearities, time delays etc) as a milestone in the field (see [82] for recent developments in the field). By achieving this goal and proving the same principle in

higher model systems (like mammalian cell lines) we will be able to move to the final and most challenging step: turning this knowledge in therapeutic strategies meant to revert pathological states by treating them as improper working points of the most complicate dynamical systems we know about, the human body.

APPENDIX A

Par.	Description	Estimated value
k_1	M. M. const. (*SWI5*→*HO* pr)	1 $[a.u.]$
k_2	M. M. const. (*ASH1*→*HO* pr)	0.0356 $[a.u.]$
k_3	M. M. const. (*CBF1*→*MET16* pr)	0.0372 $[a.u.]$
k_4	M. M. const. (*GAL4*→*GAL10* pr)	(Glu) 0.0938 $[a.u.]$; (Gal) 0.01 $[a.u.]$
k_5	M. M. const.(*SWI5*→*ASH1* pr (*GAL80*))	1.814 $[a.u.]$
k_6	M. M. const. (*SWI5*→*ASH1* pr (*ASH1*))	1.814 $[a.u.]$
α_1	Basal act. of the *HO* pr	0 $[a.u.\, min^{-1}]$
α_2	Basal act. of the *MET16* pr	1.49 $\cdot 10^{-4} [a.u.\, min^{-1}]$
α_3	Basal act. of the *GAL10* pr	$3 \cdot 10^{-3} [a.u.\, min^{-1}]$
α_4	Basal act. of the *ASH1* pr (*GAL80*)	7.4 $\cdot 10^{-4} [a.u.\, min^{-1}]$
α_5	Basal act. of the *ASH1* pr (*ASH1*)	6.1 $\cdot 10^{-4} [a.u.\, min^{-1}]$
v_1	Max. *HO* pr trans.	0.04 $[a.u.\, min^{-1}]$
v_2	Max. *MET16* pr trans.	8.82 $\cdot 10^{-4}$ $[a.u.\, min^{-1}]$
v_3	Max. *GAL10* pr trans.	(Glu) 0.0022 $[a.u.\, min^{-1}]$; (Gal) 0.022 $[a.u.\, min^{-1}]$
v_4	Max. *ASH1* pr (*GAL80*) trans.	0.0147 $[a.u.\, min^{-1}]$
v_5	Max. *ASH1* pr (*ASH1*) trans.	0.0182 $[a.u.\, min^{-1}]$
d_1	Deg. rate of *CBF1*	0.0222 $[min^{-1}]$
d_2	Deg. rate of *GAL4*	0.0478 $[min^{-1}]$
d_3	Deg. rate of *SWI5*	0.4217 $[min^{-1}]$
d_4	Deg. rate of *GAL80*	0.0980 $[min^{-1}]$
d_5	Deg. rate of *ASH1*	0.05 $[min^{-1}]$
γ	Affinity in the law of *SWI5*	(Glu) 0.2 [a.u.]; (Gal) 0.6 [a.u.]
h_1	Hill coeff. (*SWI5*→*HO* pr)	1
h_2	Hill coeff. (*ASH1*→*HO* pr)	1
h_3	Hill coeff. (*CBF1*→*MET16* pr)	1
h_4	Hill coeff. (*GAL4*→*GAL10* pr)	4
h_5	Hill coeff. (*SWI5*→*ASH1* pr (*GAL80*))	1
h_6	Hill coeff. (*SWI5*→*ASH1* pr (*ASH1*))	1
τ	Time delay	100 [min]

Table 7.1: **Estimated parameters for the IRMA dynamical model as presented in [3]**. *M.M. const.* stands for Michaelis Menten constant; *Basal act. of a promoter parameters* are meant to model the leakyness of the specific promoter thus providing for the uncontrollable, costitutive transcription rate of the downstream gene. The maximum transcription rate of the promoter is reported in the v_i parameters while protein degradation rates are expressed in d_i parameters.

APPENDIX B

Server.m

```
1                                       %Server.m
2  clear all;
3  close all;
4  clc;
5
6  %INIZIALIZZAZIONE DELLE VARIABILI
7
8  %definisco la directory di origine delle immagini di init
9  source = 'Y:\data\Filippo\clc\ndxxx\';%deve essere una ...
       cartella locale
10
11 %file di log in cui sono scritte le istruzioni di ...
       movimentazione delle
12 %siringhe
13 txtname = strcat(source,'log.txt');%deve essere un ...
       file in condivisione
14
15 %files in cui scrivo i valori dello stato e dell'input ...
       misurati
16 gmeanspath = strcat(source,'gmeans.txt');
17 rmeanspath = strcat(source,'rmeans.txt');
18 %file in cui salvo lo scaling factor, il numero di ...
       immagini dell'init in
19 %galattosio ed il numero di immagini dell'init in glucosio
20 varfile = strcat(source,'variables.mat');
```

```
21 %immagine da cui ricavare la crop area per tutto ...
       l'esperimento
22 image = 't000000c1.tif';
23 act=0;%durata funzione di azionamento attuatore.
24 ton=0;%tempo in cui la siringa Ã¨ in alto.
25 vact=[];%array con tutti i tempi della funzione di ...
       attuazione.
26 vton=[];%array con tutti gli intervalli di tempo in ...
       cui la siringa ï¿½ su
27 inp=[];%array degli input misurati
28
29 yp=[];%array delle uscite del plant
30 yg=[];%array delle uscite del modello lineare
31 ygd=[];%array delle uscite del modello lineare ritardate
32 yf=[];%array delle uscite del filtro H
33 yr=[];%array valori feedback
34 e=[]; %array degli errori
35 intError=[];%array degli integrali dell'errore
36
37 y0=[];%condizioni iniziali per la simulazione del ...
       modello lineare
38 y1d=0;%uscita del blocco delay
39
40 %variabili per la scalatura dello stato
41 statebounds=[];
42
43 state=1;% stato dell'impianto (state=1 siringa ...
       su).l'esperimento inizia con
44 %la siringa in alto
45 %il filtro ha funzione di trasferimento ...
       H(s)=1/(20s+1), posto in ingresso
46 %ad esso un segnale a(t)=A, la risposta sara : ...
       b(t)=A*(1-exp(-.05*t)).
47 %considerando T=5 avra' b(T)=a(T)(1-exp(-.25)),quindi ...
       b(T)=a(T)*.2212
48
49 %definizione della durata dell'init
50 galtime=input('Inserisci le ore di init in cui le ...
       cellule sono in galattosio: ');
51 glutime=input('Inserisci le ore di init in cui le ...
       cellule sono in glucosio: ');
52 save('galtime.mat','galtime');
53
54
55 %CALCOLO DELLA CROP AREA DELLE IMMAGINI
56 disp(['CREARE UNA IMMAGINE CON IL MICROSCOPIO, ...
       NOMINARLA ',image,' E PREMERE UN TASTO PER ...
       CONTINUARE...']);
57 pause;
```

```
58 clc
59 CropRect=Crop1stImage(source,image);
60 save('CropRect.mat','CropRect');
61
62 %INIZIALIZZAZIONE DEL FILE DI LOG
63 Write2Txt(txtname,0)
64 disp('AVVIARE FILE SCRIPT CLIENT.M SU PC CLIENT E ...
       PREMERE UN TASTO PER CONTINUARE...')
65 pause;
66 %INIZIALIZZAZIONE DEI FILE DI TESTO CHE MEMORIZZANO I ...
       VALORI DI yp E input
67 WriteGmeans(gmeanspath,yp);
68 WriteRmeans(rmeanspath,inp);
69
70 %SCELTA STRATEGIA DI CONTROLLO:
71 ctr=0;
72 while ctr==0||ctr>3
73     disp('Digita 1 per il controllo con predittore di ...
           Smith');
74     disp('Digita 2 per il controllo con retroazione ...
           dello stato e PID+PWM');
75     disp('Digita 3 per il controllo con retroazione ...
           dello stato e Relè');
76     ctr=input('Digita la tua scelta: ');
77 end
78 %INIT DEL CONTROLLO : CALCOLO USCITE MODELLO
79 init=1;%flag di esito dell'init.
80 inittime=galtime+glutime;%durata totale dell'init
81 tspan=[0 (inittime*60)];%intervallo di tempo in minuti ...
       dell'init
82 %cbf1 e' l'andamento della proteina
83 %y0 e' il vettore che contiene le condizioni iniziali ...
       per la prossima
84 %simulazione
85 [cbf1,y0] = state0_ModSim(tspan);
86 numbinitpics = length(cbf1);
87 numbinitpicsgal = galtime*12;
88 numbinitpicsglu = glutime*12;
89 %simulo il modello lineare durante l'init
90 [yg,y0g] = state0_ModSimNoDelay(tspan);
91
92 %INIT DEL CONTROLLO : CALCOLO DEL SEGNALE DI ...
       RIFERIMENTO PER IL CONTROLLO
93 refok=0;
94 while refok==0
95 clc
96 disp('CALCOLO DEL SEGNALE DI RIFERIMENTO PER IL ...
       CONTROLLO');
97 [r,exp_time]=state0_choose_reference(max(cbf1),min(cbf1));
```

```
98  ref=[linspace(0,exp_time,exp_time)',r'];
99  if ctr==1
100    sim('schemeST',[0.0 exp_time]);
101 elseif ctr==2
102     sim('schemeSTSimpleFB',[0.0 exp_time]);
103 elseif ctr==3
104     sim('schemeSTSimpleFBRelay',[0.0 exp_time]);
105 end
106 figure
107 plot(linspace(0,exp_time,exp_time),r,'k');hold on;
108 plot(t,output,'g');
109 plot(t,ctrinp./100,'r');
110 grid on;
111 legend('reference signal','simulated ...
       output','simulated input_{scaled}');
112 refok = input('digita 1 se il riferimento Ã¨ corretto, ...
       altrimenti digita 0 per ripeterne il calcolo: ');
113 end
114
115
116 clc
117 %campionamento del vettore di riferimento
118 reference = r(1:5:end);
119 clear r ref refok output ctrinp;
120
121 disp('AVVIARE ACQUISIZIONE IMMAGINI DAL MICROSCOPIO');
122 disp('la prima immagine dovrÃ  essere t000001c1.tif');
123 disp('premere un tasto per continuare...');
124 pause;
125
126 %INIT DEL CONTROLLO: CALCOLO ANDAMENTO FLUORESCENZA
127 clear i j;
128         %richiamo la funzione di init del plant
129         [f,yp,inp,state,i] = state0_PlantInit...
130         (numbinitpicsgal,numbinitpics,yp,inp,state,...
131         source, txtname);
132         if f==1%se l'init del plant Ã¨ andato a buon fine.
133             %realizzo la scalatura del vettore di ...
                    stato ...
134             %misurato nello
135             %spazio del vettore di stato simulato
136             ypnotscaled=yp;
137
138             upperBound=mean(ypnotscaled(1:10));
139             lowerBound=mean(ypnotscaled(end-10:end));
140             if lowerBound > upperBound
141                 disp('Lower bound > di upper Bound. ...
                        Init ...
142                 terminato.')
```

```
143                 init=0;
144                 s=6;
145             end
146             scaledstate=(yp-lowerBound)./(upperBound-...
147             lowerBound)*(cbf1(1)-min(cbf1))+min(cbf1);
148             yp=scaledstate;
149             clear scaledstate;
150             %memorizzo i valori minimo e massimo dello ...
                    stato del modello
151             %ritardato simulato.
152             statebounds=[min(cbf1) max(cbf1)];
153             %effettuo la scalatura dello stato del plant
154             %yp=state0_PlantStateScaling(yp,statebounds);
155             %scrivo su file di testo il vettore yp ed ...
                    il vettore inp
156             WriteGmeans(gmeanspath,yp);
157             WriteRmeans(rmeanspath,inp);
158             %plot dell'inizializzazione
159             figure
160             plot(5*linspace(0,tspan(end)/5,tspan(end)...
161             /5),cbf1,'o');grid;hold on;
162             plot(5*linspace(0,tspan(end)/5,tspan(end)...
163             /5),yp,'g*');
164             title('System Init');
165             xlabel('Time [min.]');
166             ylabel('Fluorescence [a.u.]');
167             legend('Sim. model state','Plant state ...
                    (cells fluorescence)');
168             %calcolo l'errore per la prima iterazione ...
                    di controllo:
169             %.2212*2 tiene conto di H(s)
170             %err_1 e' l'errore all'istante -1, e mi ...
                    serve per realizzare il
171             %mantenitore del primo ordine sull'errore ...
                    all'inizio del controllo
172             err_1=reference(1)-((yp(end-1)-yg(end-21))...
173             *.2212*2+yg(end-1));
174             ygd=[yg(end-21)];
175             yf=[(yp(end-1)-yg(end-21))*.2212*2];
176             yr=[((yp(end-1)-yg(end-21))*.2212*2+...
177             yg(end-1))];
178             e=[e err_1];
179             err0=reference(1)-((yp(end)-yg(end-20))...
180             *.2212*2+yg(end));
181             ygd=[ygd,yg(end-20)];
182             yf=[yf,(yp(end)-yg(end-20))*.2212*2];
183             yr=[yr,((yp(end)-yg(end-20))*.2212*2+yg(end))];
184             e=[e err0];
185             clear err_1 err0;
```

```
186             disp('Inizializzazione del sistema ...
                    terminata, ...
187             calcolato errore per la prima iterazione');
188             s=1;
189             save(varfile,'ypnotscaled','numbinitpicsgal',...
190             'numbinitpicsglu');
191         else
192             disp('Init terminato con errore')
193             init=0;
194             s=6;%sono condizioni che mi servono per ...
                    non entrare ...
195             %nel successivo ciclo while
196         end
197
198 %AUTOMA A STATI FINITI CHE IMPLEMENTA IL CONTROLLO A ...
    SECONDA DELLA
199 %STRATEGIA SELEZIONATA
200
201 j=1;%indice che scorre le componenti del segnale di ...
    riferimento
202
203
204 if ctr==1%PREDITTORE DI SMITH
205  %calcolo l'errore per la prima iterazione di ...
        controllo nel caso
206  %stia adoperando il predittore di smith.
207  %.2212*2 tiene conto di H(s)
208  %err_1 e' l'errore all'istante -1, e mi serve per ...
        realizzare il
209  %mantenitore del primo ordine sull'errore all'inizio ...
        del controllo
210  err_1=reference(1)-((yp(end-1)-yg(end-21))*.2212*2+...
211  yg(end-1));
212  ygd=[yg(end-21)];
213  yf=[(yp(end-1)-yg(end-21))*.2212*2];
214  yr=[((yp(end-1)-yg(end-21))*.2212*2+yg(end-1))];
215  e=[e err_1];
216  err0=reference(1)-((yp(end)-yg(end-20))*.2212*2+yg(end));
217  ygd=[ygd,yg(end-20)];
218  yf=[yf,(yp(end)-yg(end-20))*.2212*2];
219  yr=[yr,((yp(end)-yg(end-20))*.2212*2+yg(end))];
220  e=[e err0];
221  clear err_1 err0;
222    while j<=length(reference)&&init==1&&(s>=1&&s<=5)
223     if j==1
224         clc;
225         disp('init ok, inizio del controllo');
226     end
227 clc;
```

```
228 disp(['Iterazione di controllo numero: ',num2str(j)]);
229
230 %PRIMO STATO predittore di smith
231
232     while s==1
233     disp('STATO 1: CALCOLO DEL CONTROLLO u(k).');
234     %Calcolo offset e pendenza errore.
235             a=e(end-1);b=e(end);
236             m=(b-a)/5;%pendenza errore
237             a=e(end);%offset errore
238         disp(['offset errore: ',num2str(a),', pendenza ...
                errore: ',num2str(m)]);
239         %calcolo l'onda quadra in uscita al blocco PWM ...
                simulando
240         %controller.mdl
241         sim('controller');
242
243         %calcolo il ton del controllo e il valore del ...
                controllo stesso.
244         %U Ã¨ la matrice che va in ingresso alla ...
                simulazione del modello
245         %lineare (senza ritardo), la prima colonna Ã¨ ...
                un vettore dei
246         %tempi, la seconda Ã¨ un vettore che contiene ...
                l'azione dicontrollo.
247         [ton,U] = ...
                state1_calculatingInput(uc,controllertime);
248         vton = [vton,ton];%memorizzo l'attuale ton
249         clear uc controllertime;%le cancello perchÃ¨ ...
                le ho memorizzate nella matrice u.
250
251         disp('FINE STATO 1');
252         s=2;
253     end
254
255 %SECONDO STATO predittore di smith
256
257     while s==2
258     disp('STATO 2: SIMULAZIONE DEL MODELLO LINEARE PER ...
           IL CONTROLLO ...
259     u(k) CALCOLATO');
260         % simulazione del modello lineare con u come ...
                input e y0g come
261         % vettore delle condizioni iniziali.
262         t1=U(:,1)';%vettore dei tempi dell'input
263         u=U(:,2)';%input del modello
264         clear U;
265         tspan=[0 5];%il modello lineare va simulato ...
                per 5 minuti
```

```
266         [t Y] = ode45(@(t,y) ...
                LinearModel(t,y,t1,u),tspan,y0g);
267         clear t;
268         %le condizioni iniziali per la prossima ...
                simulazione coincidono con
269         %l'ultima riga dell'uscita del sistema simulato;
270         y0g=Y(end,:)';
271         clear Y;
272         %concateno l'ultima uscita ricavata al vettore ...
                yg(1Â° elemento della
273         %colonna)
274         yg=[yg,y0g(1)];
275         %fine dello stato 2
276         disp('FINE STATO 2');
277
278     s=3;
279     end
280
281 %TERZO STATO predittore di smith
282
283     while s==3
284     disp('STATO 3: CONTROLLO DEL PLANT, MOVIMENTAZIONE ...
            SIRINGHE');
285         %comando la siringa per tenerla in alto per il ...
                tempo ton, ottengo
286         %come parametro di uscita lo stato e la durata ...
                della funzione di
287         %azionamento (vedere l'help della funzione ...
                richiamata per dettagli
288         %sul funzionamento).
289         [act,state]=state3_MoveSyringes(ton,state,txtname);
290         %memorizzo l'attuale act
291         vact = [vact,act];
292         disp(['Durata attuazione: ',num2str(act),' ...
                sec.']);
293         disp('FINE STATO 3');
294
295     s=4;
296     end
297
298 %QUARTO STATO predittore di smith
299
300     while s==4
301     disp('STATO 4: CALCOLO DELLA USCITA RITARDATA DEL ...
            MODELLO LINEARE');
302         %y1 Ã¨ l'ultima componente del vettore yg
303         y1=yg(end);
304         %l'uscita del blocco delay y1d devo prendere ...
                la 20esima componente
```

```
305         %all'indietro di yg.
306         y1d=yg(end-20);
307         ygd=[ygd,y1d];
308         disp('FINE STATO 4');
309
310     s=5;
311     end
312
313 %QUINTO STATO predittore di smith
314
315     while s==5
316     disp('STATO 5: CALCOLO DELLO STATO DEL PLANT E ...
            ERRORE');
317         %la funzione richiamata di seguito calcola lo ...
                stato del plant reale
318         %andando ad elaborare l'i-esima immagine ...
                acquisita dal microscopio.
319         %viene valutato il tampo act dell'attuazione ...
                al fine di attendere i
320         %5 minuti che intercorrono fra l'acquisizione ...
                di due immagini
321         %successive, se ï¿½ possibile elaborare ...
                l'immagine il valore
322         %restituito dalla funzione sarï¿½ maggiore di ...
                0 ed allora si prosegue
323         %con il controllo, altrimenti si esce ...
                dall'algoritmo.
324         [mean,rmean,f]= state5_PlantState(act,source,i);
325         if f==1%se l'acquisizione ï¿½ andata a buon fine
326         i=i+1;%incremento contatore immagini.
327         %scalo in aribitrary units
328         scaledmean=mean;
329         scaledmean=(mean-lowerBound)./(upperBound...
330         -lowerBound)*(max(cbf1)-min(cbf1))+min(cbf1);
331         mean = scaledmean;
332         %memorizzazione dello stato del plant nel ...
                vettore yp
333         yp=[yp mean];
334         %memorizzazione dell'input misurato nel ...
                vettore inp
335         inp=[inp rmean];
336         %scrivo sui file di testo i vettori yp ed inp
337         WriteGmeans(gmeanspath,yp);
338         WriteRmeans(gmeanspath,inp);
339         %calcolo dell'errore, confronto del ...
                riferimento con il risultato
340         %della retroazione
341         %.2212*2 tiene conto di H(s)
342         yf=[yf,(yp(end)-ygd(end))*.2212*2];
```

```
343         yr=[yr,(y1+(yp(end)-y1d)*.2212*2)];
344         errk=reference(j)-(y1+(yp(end)-y1d)*.2212*2);
345         e=[e errk];
346         s=1;
347 
348         else
349             s=6;
350 
351         end
352     end
353 
354 j=j+1;
355 end
356 
357 elseif ctr==2%retroazione dello stato con pid+pwm
358 %calcolo dell'errore alla prima iterazione di controllo
359 [b,a]=butter(2,0.3);
360 yf=filter(b,a,yp);
361 
362 e = [(reference(j)-yf(end-1)),(reference(j)-yf(end))];
363 
364   while j<=length(reference)&&init==1&&(s>=1&&s<=5)
365     if j==1
366         clc;
367         disp('init ok, inizio del controllo con ...
               retroazione ...
368         dello stato e PID+PWM');
369     end
370 clc;
371 disp(['Iterazione di controllo numero: ',num2str(j)]);
372 
373 %PRIMO STATO feedback con filtro - PID+PWM
374 
375     while s==1
376     disp('STATO 1: CALCOLO DEL CONTROLLO u(k).');
377     %Calcolo offset e pendenza errore.
378             a=e(end-1);b=e(end);
379             m=(b-a)/5;%pendenza errore
380             a=e(end);%offset errore
381         disp(['offset errore: ',num2str(a),', pendenza ...
               errore: ',num2str(m)]);
382         %calcolo l'onda quadra in uscita al blocco PWM ...
               simulando
383         %controller.mdl
384         sim('controller');
385 
386         %AGGIUNTO DA FILIPPO per il debug dell'anti windup
387         intError=[intError, integralError(end)]
388 
```

```
389
390         %calcolo il ton del controllo e il valore del ...
                controllo stesso.
391         %U e' la matrice che va in ingresso alla ...
                simulazione del modello
392         %lineare (senza ritardo), la prima colonna Ã¨ ...
                un vettore dei
393         %tempi, la seconda Ã¨ un vettore che contiene ...
                l'azione dicontrollo.
394         [ton,U] = ...
                state1_calculatingInput(uc,controllertime);
395         vton = [vton,ton];%memorizzo l'attuale ton
396         clear uc controllertime;%le cancello perchÃ¨ ...
                le ho ...
397         %memorizzate nella matrice u.
398         disp('FINE STATO 1');
399         s=2;
400     end
401
402 %SECONDO STATO feedback con filtro – PID+PWM
403
404     while s==2
405     disp('STATO 2: –');
406     s=3;
407     end
408
409 %TERZO STATO feedback con filtro – PID+PWM
410
411     while s==3
412     disp('STATO 3: CONTROLLO DEL PLANT, MOVIMENTAZIONE ...
            SIRINGHE');
413         %comando la siringa per tenerla in alto per il ...
                tempo ton, ottengo
414         %come parametro di uscita lo stato e la durata ...
                della funzione di
415         %azionamento (vedere l'help della funzione ...
                richiamata per dettagli
416         %sul funzionamento).
417         [act,state]=state3_MoveSyringes(ton,state,txtname);
418         %memorizzo l'attuale act
419         vact = [vact,act];
420         disp(['Durata attuazione: ',num2str(act),' ...
                sec.']);
421         disp('FINE STATO 3');
422
423     s=4;
424     end
425
426 %QUARTO STATO feedback con filtro – PID+PWM
```

```
427
428     while s==4
429     disp('STATO 4: -');
430     s=5;
431     end
432
433 %QUINTO STATO feedback con filtro - PID+PWM
434
435     while s==5
436     disp('STATO 5: CALCOLO DELLO STATO DEL PLANT E ...
            ERRORE');
437         %la funzione richiamata di seguito calcola lo ...
                stato del plant reale
438         %andando ad elaborare l'i-esima immagine ...
                acquisita dal microscopio.
439         %viene valutato il tampo act dell'attuazione ...
                al fine di attendere i
440         %5 minuti che intercorrono fra l'acquisizione ...
                di due immagini
441         %successive, se Ã¨ possibile elaborare ...
                l'immagine il valore
442         %restituito dalla funzione sarÃ  maggiore di 0 ...
                ed allora si prosegue
443         %con il controllo, altrimenti si esce ...
                dall'algoritmo.
444         [mean,rmean,f]= state5_PlantState(act,source,i);
445         if f==1%se l'acquisizione ï¿½ andata a buon fine
446         i=i+1;%incremento contatore immagini.
447         %scalo in aribitrary units
448         scaledmean=mean;
449         scaledmean=(mean-lowerBound)./(upperBound...
450         -lowerBound)*(max(cbf1)-min(cbf1))+min(cbf1);
451         mean = scaledmean;
452         %memorizzazione dello stato del plant nel ...
                vettore yp
453         yp=[yp mean];
454         %memorizzazione dell'input misurato nel ...
                vettore inp
455         inp=[inp rmean];
456         %scrivo sui file di testo i vettori yp ed inp
457         WriteGmeans(gmeanspath,yp);
458         WriteRmeans(gmeanspath,inp);
459         %calcolo il valore dell'uscita del filtro ...
                sulla catena di
460         %retroazione
461         [b,a]=butter(2,0.3);
462         yf=filter(b,a,yp);
463         %calcolo il valore dell'errore
464         e=[e,(reference(j)-yf(end))];
```

```
465         s=1;
466
467         else
468             s=6;
469
470         end
471     end
472
473 j=j+1;
474   end
475
476 elseif ctr==3%retroazione dello stato con relÃ¨
477 %la variabile relay indica lo stato in cui dovrÃ  ...
       essere posto il relÃ¨, relay = 0
478 %implica relÃ¨ spento, relay = 1 implica relÃ¨ acceso.
479 %calcolo dell'errore alla prima iterazione di controllo
480 [b,a]=butter(2,0.3);
481 yf=filter(b,a,yp);
482 e = reference(j)-yf(end);
483
484   while j<=length(reference)&&init==1&&(s>=1&&s<=5)
485     if j==1
486         clc;
487         disp('init ok, inizio del controllo con ...
               retroazione ...
488         dello stato e PID+PWM');
489     end
490 clc;
491 disp(['Iterazione di controllo numero: ',num2str(j)]);
492
493 %PRIMO STATO relay
494
495     while s==1
496     disp('STATO 1: CALCOLO DELLO STATO DEL RELAY');
497     if reference(j) > yf(end)
498         relay=1;
499     elseif reference(j) <= yf(end)
500         relay=0;
501     end
502         disp('FINE STATO 1');
503         s=2;
504     end
505
506 %SECONDO STATO relay
507
508     while s==2
509     disp('STATO 2: -');
510     s=3;
511     end
```

```
512
513 %TERZO STATO relay
514
515     while s==3
516     disp('STATO 3: CONTROLLO DEL PLANT, MOVIMENTAZIONE ...
            SIRINGHE');
517     if state==1
518         if relay == 0
519             ton=0;
520             Write2Txt(txtname,-1);
521             state = 0;
522         elseif relay ==1
523             ton=300;
524             state=1;
525         end
526     elseif state==0
527         if relay == 0
528             ton=0;
529             state = 0;
530         elseif relay ==1
531             ton=300;
532             Write2Txt(txtname,1);
533             state = 1;
534         end
535     end
536    %memorizzo l'attuale act
537     vton=[vton,ton];
538     vact = [vact,act];
539     disp(['Durata attuazione: ',num2str(ton),' sec.'])
540     disp('FINE STATO 3');
541
542     s=4;
543     end
544
545 %QUARTO STATO relay
546
547     while s==4
548     disp('STATO 4: -');
549     s=5;
550     end
551
552 %QUINTO STATO relay
553
554     while s==5
555     disp('STATO 5: CALCOLO DELLO STATO DEL PLANT E ...
            ERRORE');
556         %la funzione richiamata di seguito calcola lo ...
               stato del plant reale
```

```
557         %andando ad elaborare l'i-esima immagine ...
                acquisita dal microscopio.
558         %viene valutato il tampo act dell'attuazione ...
                al fine di attendere i
559         %5 minuti che intercorrono fra l'acquisizione ...
                di due immagini
560         %successive, se ï¿½ possibile elaborare ...
                l'immagine il valore
561         %restituito dalla funzione sarï¿½ maggiore di ...
                0 ed allora si prosegue
562         %con il controllo, altrimenti si esce ...
                dall'algoritmo.
563         [mean,rmean,f]= state5_PlantState(act,source,i);
564         if f==1%se l'acquisizione ï¿½ andata a buon fine
565         i=i+1;%incremento contatore immagini.
566         %scalo in aribitrary units
567         scaledmean=mean;
568         scaledmean=(mean-lowerBound)./(upperBound-...
569         lowerBound)*(max(cbf1)-min(cbf1))+min(cbf1);
570         mean = scaledmean;
571         %memorizzazione dello stato del plant nel ...
                vettore yp
572         yp=[yp mean];
573         %memorizzazione dell'input misurato nel ...
                vettore inp
574         inp=[inp rmean];
575         %scrivo sui file di testo i vettori yp ed inp
576         WriteGmeans(gmeanspath,yp);
577         WriteRmeans(gmeanspath,inp);
578         %calcolo il valore dell'uscita del filtro ...
                sulla catena di
579         %retroazione
580         [b,a]=butter(2,0.3);
581         yf=filter(b,a,yp);
582         %calcolo il valore dell'errore
583         e=[e,(reference(j)-yf(end))];
584         s=1;
585
586         else
587             s=6;
588
589         end
590     end
591
592 j=j+1;
593   end
594 end
595
596
```

```
597 if init==1
598
599 %interrompo l'esecuzione del client
600 Write2Txt(txtname,2);
601
602 disp(['Controllo terminato, numero di immagini ...
       processate: ',...
603 num2str(i-1)]);
604 disp(['Numero di immagini processate durante il ...
       controllo: ',...
605 num2str(i-numbinitpics-1)]);
606 disp(['Numero di iterazioni di controllo: ...
       ',num2str(j-1)]);
607 close all;
608
609 dimp=length(yp);
610 dimg=length(yg);
611
612 ingresso = ton2input(vton);
613
614 else
615     Write2Txt(txtname,2);%stoppo l'esecuzione del client
616     disp('controllo interrotto durante la fase di init');
617 end
```

CircularHough_Grd.m

```
function [accum, varargout] = CircularHough_Grd(img, radrange, varargin)
% Validation of arguments
if ndims(img) ≠ 2 || ¬isnumeric(img),
    error('CircularHough_Grd: ''img'' has to be 2 dimensional');
end
if ¬all(size(img) ≥ 32),
    error('CircularHough_Grd: ''img'' has to be larger than 32-by-32');
end

if numel(radrange) ≠ 2 || ¬isnumeric(radrange),
    error(['CircularHough_Grd: ''radrange'' has to be ', ...
        'a two-element vector']);
end
prm_r_range = sort(max( [0,0;radrange(1),radrange(2)] ));

% Parameters (default values)
prm_grdthres = 10;
prm_fltrLM_R = 8;
prm_multirad = 0.5;
func_compu_cen = true;
func_compu_radii = true;

% Validation of arguments
vap_grdthres = 1;
if nargin > (1 + vap_grdthres),
    if isnumeric(varargin{vap_grdthres}) && ...
            varargin{vap_grdthres}(1) ≥ 0,
        prm_grdthres = varargin{vap_grdthres}(1);
    else
        error(['CircularHough_Grd: ''grdthres'' has to be ', ...
            'a non-negative number']);
    end
end

vap_fltr4LM = 2;    % filter for the search of local maxima
if nargin > (1 + vap_fltr4LM),
    if isnumeric(varargin{vap_fltr4LM}) && ...
        varargin{vap_fltr4LM}(1) ≥ 3,
```

```
38          prm_fltrLM_R = varargin{vap_fltr4LM}(1);
39      else
40          error(['CircularHough_Grd: ''fltr4LM_R'' has ...
                to be ', ...
41              'larger than or equal to 3']);
42      end
43  end
44
45  vap_multirad = 3;
46  if nargin > (1 + vap_multirad),
47      if isnumeric(varargin{vap_multirad}) && ...
48          varargin{vap_multirad}(1) ≥ 0.1 && ...
49          varargin{vap_multirad}(1) ≤ 1,
50      prm_multirad = varargin{vap_multirad}(1);
51      else
52          error(['CircularHough_Grd: ''multirad'' has to ...
                be ', ...
53              'within the range [0.1, 1]']);
54      end
55  end
56
57  vap_fltr4accum = 4; % filter for smoothing the ...
        accumulation array
58  if nargin > (1 + vap_fltr4accum),
59      if isnumeric(varargin{vap_fltr4accum}) && ...
60              ndims(varargin{vap_fltr4accum}) == 2 && ...
61              all(size(varargin{vap_fltr4accum}) ≥ 3),
62          fltr4accum = varargin{vap_fltr4accum};
63      else
64          error(['CircularHough_Grd: ''fltr4accum'' has ...
                to be ', ...
65              'a 2-D matrix with a minimum size of ...
                    3-by-3']);
66      end
67  else
68      % Default filter (5-by-5)
69      fltr4accum = ones(5,5);
70      fltr4accum(2:4,2:4) = 2;
71      fltr4accum(3,3) = 6;
72  end
73
74  func_compu_cen = ( nargout > 1 );
75  func_compu_radii = ( nargout > 2 );
76
77  % Reserved parameters
78  dbg_on = false;       % debug information
79  dbg_bfigno = 4;
80  if nargout > 3,  dbg_on = true;  end
81
```

```
82
83 %%%%%%%% Building accumulation array ...
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
84
85 % Convert the image to single if it is not of
86 % class float (single or double)
87 img_is_double = isa(img, 'double');
88 if ¬(img_is_double || isa(img, 'single')),
89     imgf = single(img);
90 end
91
92 % Compute the gradient and the magnitude of gradient
93 if img_is_double,
94     [grdx, grdy] = gradient(img);
95 else
96     [grdx, grdy] = gradient(imgf);
97 end
98 grdmag = sqrt(grdx.^2 + grdy.^2);
99
100 % Get the linear indices, as well as the subscripts, ...
       of the pixels
101 % whose gradient magnitudes are larger than the given ...
       threshold
102 grdmasklin = find(grdmag > prm_grdthres);
103 [grdmask_IdxI, grdmask_IdxJ] = ind2sub(size(grdmag), ...
       grdmasklin);
104
105 % Compute the linear indices (as well as the ...
       subscripts) of
106 % all the votings to the accumulation array.
107 % The Matlab function 'accumarray' accepts only double ...
       variable,
108 % so all indices are forced into double at this point.
109 % A row in matrix 'lin2accum_aJ' contains the J ...
       indices (into the
110 % accumulation array) of all the votings that are ...
       introduced by a
111 % same pixel in the image. Similarly with matrix ...
       'lin2accum_aI'.
112 rr_4linaccum = double( prm_r_range );
113 linaccum_dr = [ (-rr_4linaccum(2) + 0.5) : ...
       -rr_4linaccum(1) , ...
114     (rr_4linaccum(1) + 0.5) : rr_4linaccum(2) ];
115
116 lin2accum_aJ = floor( ...
117     double(grdx(grdmasklin)./grdmag(grdmasklin)) * ...
           linaccum_dr + ...
118     repmat( double(grdmask_IdxJ)+0.5 , ...
           [1,length(linaccum_dr)] ) ...
```

```
119 );
120 lin2accum_aI = floor( ...
121     double(grdy(grdmasklin)./grdmag(grdmasklin)) * ...
        linaccum_dr + ...
122     repmat( double(grdmask_IdxI)+0.5 , ...
        [1,length(linaccum_dr)] ) ...
123 );
124
125 % Clip the votings that are out of the accumulation array
126 mask_valid_aJaI = ...
127     lin2accum_aJ > 0 & lin2accum_aJ < (size(grdmag,2) ...
        + 1) & ...
128     lin2accum_aI > 0 & lin2accum_aI < (size(grdmag,1) ...
        + 1);
129
130 mask_valid_aJaI_reverse = ¬ mask_valid_aJaI;
131 lin2accum_aJ = lin2accum_aJ .* mask_valid_aJaI + ...
    mask_valid_aJaI_reverse;
132 lin2accum_aI = lin2accum_aI .* mask_valid_aJaI + ...
    mask_valid_aJaI_reverse;
133 clear mask_valid_aJaI_reverse;
134
135 % Linear indices (of the votings) into the ...
    accumulation array
136 lin2accum = sub2ind( size(grdmag), lin2accum_aI, ...
    lin2accum_aJ );
137
138 lin2accum_size = size( lin2accum );
139 lin2accum = reshape( lin2accum, [numel(lin2accum),1] );
140 clear lin2accum_aI lin2accum_aJ;
141
142 % Weights of the votings, currently using the gradient ...
    maginitudes
143 % but in fact any scheme can be used (application ...
    dependent)
144 weight4accum = ...
145     repmat( double(grdmag(grdmasklin)) , ...
        [lin2accum_size(2),1] ) .* ...
146     mask_valid_aJaI(:);
147 clear mask_valid_aJaI;
148
149 % Build the accumulation array using Matlab function ...
    'accumarray'
150 accum = accumarray( lin2accum , weight4accum );
151 accum = [ accum ; zeros( numel(grdmag) - numel(accum) ...
    , 1 ) ];
152 accum = reshape( accum, size(grdmag) );
153
154
```

```
155 %%%%%%%% Locating local maxima in the accumulation ...
        array %%%%%%%%%%%%
156
157 % Stop if no need to locate the center positions of ...
        circles
158 if ¬func_compu_cen,
159     return;
160 end
161 clear lin2accum weight4accum;
162
163 % Parameters to locate the local maxima in the ...
        accumulation array
164 % -- Segmentation of 'accum' before locating LM
165 prm_useaoi = true;
166 prm_aoithres_s = 2;
167 prm_aoiminsize = floor(min([ min(size(accum)) * 0.25, ...
168     prm_r_range(2) * 1.5 ]));
169
170 % -- Filter for searching for local maxima
171 prm_fltrLM_s = 1.35;
172 prm_fltrLM_r = ceil( prm_fltrLM_R * 0.6 );
173 prm_fltrLM_npix = max([ 6, ceil((prm_fltrLM_R/2)^1.8) ]);
174
175 % -- Lower bound of the intensity of local maxima
176 prm_LM_LoBndRa = 0.2;  % minimum ratio of LM to the ...
        max of 'accum'
177
178 % Smooth the accumulation array
179 fltr4accum = fltr4accum / sum(fltr4accum(:));
180 accum = filter2( fltr4accum, accum );
181
182 % Select a number of Areas-Of-Interest from the ...
        accumulation array
183 if prm_useaoi,
184     % Threshold value for 'accum'
185     prm_llm_thres1 = prm_grdthres * prm_aoithres_s;
186
187     % Thresholding over the accumulation array
188     accummask = ( accum > prm_llm_thres1 );
189
190     % Segmentation over the mask
191     [accumlabel, accum_nRgn] = bwlabel( accummask, 8 );
192
193     % Select AOIs from segmented regions
194     accumAOI = ones(0,4);
195     for k = 1 : accum_nRgn,
196         accumrgn_lin = find( accumlabel == k );
197         [accumrgn_IdxI, accumrgn_IdxJ] = ...
198             ind2sub( size(accumlabel), accumrgn_lin );
```

```
199         rgn_top = min( accumrgn_IdxI );
200         rgn_bottom = max( accumrgn_IdxI );
201         rgn_left = min( accumrgn_IdxJ );
202         rgn_right = max( accumrgn_IdxJ );
203         % The AOIs selected must satisfy a minimum size
204         if ( (rgn_right - rgn_left + 1) >= ...
                prm_aoiminsize && ...
205                 (rgn_bottom - rgn_top + 1) >= ...
                        prm_aoiminsize ),
206             accumAOI = [ accumAOI; ...
207                 rgn_top, rgn_bottom, rgn_left, ...
                        rgn_right ];
208         end
209     end
210 else
211     % Whole accumulation array as the one AOI
212     accumAOI = [1, size(accum,1), 1, size(accum,2)];
213 end
214
215 % Thresholding of 'accum' by a lower bound
216 prm_LM_LoBnd = max(accum(:)) * prm_LM_LoBndRa;
217
218 % Build the filter for searching for local maxima
219 fltr4LM = zeros(2 * prm_fltrLM_R + 1);
220
221 [mesh4fLM_x, mesh4fLM_y] = meshgrid(-prm_fltrLM_R : ...
        prm_fltrLM_R);
222 mesh4fLM_r = sqrt( mesh4fLM_x.^2 + mesh4fLM_y.^2 );
223 fltr4LM_mask = ...
224     ( mesh4fLM_r > prm_fltrLM_r & mesh4fLM_r <= ...
            prm_fltrLM_R );
225 fltr4LM = fltr4LM - ...
226     fltr4LM_mask * (prm_fltrLM_s / sum(fltr4LM_mask(:)));
227
228 if prm_fltrLM_R >= 4,
229     fltr4LM_mask = ( mesh4fLM_r < (prm_fltrLM_r - 1) );
230 else
231     fltr4LM_mask = ( mesh4fLM_r < prm_fltrLM_r );
232 end
233 fltr4LM = fltr4LM + fltr4LM_mask / sum(fltr4LM_mask(:));
234
235 % **** Debug code (begin)
236 if dbg_on,
237     dbg_LMmask = zeros(size(accum));
238 end
239 % **** Debug code (end)
240
241 % For each of the AOIs selected, locate the local maxima
242 circen = zeros(0,2);
```

```
243 for k = 1 : size(accumAOI, 1),
244     aoi = accumAOI(k,:);    % just for referencing ...
        convenience
245
246     % Thresholding of 'accum' by a lower bound
247     accumaoi_LBMask = ...
248         ( accum(aoi(1):aoi(2), aoi(3):aoi(4)) > ...
            prm_LM_LoBnd );
249
250     % Apply the local maxima filter
251     candLM = conv2( accum(aoi(1):aoi(2), ...
        aoi(3):aoi(4)) , ...
252         fltr4LM , 'same' );
253     candLM_mask = ( candLM > 0 );
254
255     % Clear the margins of 'candLM_mask'
256     candLM_mask([1:prm_fltrLM_R, ...
        (end-prm_fltrLM_R+1):end], :) = 0;
257     candLM_mask(:, [1:prm_fltrLM_R, ...
        (end-prm_fltrLM_R+1):end]) = 0;
258
259     % **** Debug code (begin)
260     if dbg_on,
261         dbg_LMmask(aoi(1):aoi(2), aoi(3):aoi(4)) = ...
262             dbg_LMmask(aoi(1):aoi(2), aoi(3):aoi(4)) + ...
263             accumaoi_LBMask + 2 * candLM_mask;
264     end
265     % **** Debug code (end)
266
267     % Group the local maxima candidates by adjacency, ...
        compute the
268     % centroid position for each group and take that ...
        as the center
269     % of one circle detected
270     [candLM_label, candLM_nRgn] = bwlabel( ...
        candLM_mask, 8 );
271
272     for ilabel = 1 : candLM_nRgn,
273         % Indices (to current AOI) of the pixels in ...
            the group
274         candgrp_masklin = find( candLM_label == ilabel );
275         [candgrp_IdxI, candgrp_IdxJ] = ...
276             ind2sub( size(candLM_label) , ...
                candgrp_masklin );
277
278         % Indices (to 'accum') of the pixels in the group
279         candgrp_IdxI = candgrp_IdxI + ( aoi(1) - 1 );
280         candgrp_IdxJ = candgrp_IdxJ + ( aoi(3) - 1 );
281         candgrp_idx2acm = ...
```

```
282             sub2ind( size(accum) , candgrp_IdxI , ...
                    candgrp_IdxJ );
283
284         % Minimum number of qulified pixels in the group
285         if sum(accumaoi_LBMask(candgrp_masklin)) < ...
                prm_fltrLM_npix,
286             continue;
287         end
288
289         % Compute the centroid position
290         candgrp_acmsum = sum( accum(candgrp_idx2acm) );
291         cc_x = sum( candgrp_IdxJ .* ...
                accum(candgrp_idx2acm) ) / ...
292              candgrp_acmsum;
293         cc_y = sum( candgrp_IdxI .* ...
                accum(candgrp_idx2acm) ) / ...
294              candgrp_acmsum;
295         circen = [circen; cc_x, cc_y];
296     end
297 end
298
299 % **** Debug code (begin)
300 if dbg_on,
301     figure(dbg_bfigno); imagesc(dbg_LMmask); axis image;
302     title('Generated map of local maxima');
303     if size(accumAOI, 1) == 1,
304         figure(dbg_bfigno+1);
305         surf(candLM, 'EdgeColor', 'none'); axis ij;
306         title('Accumulation array after local maximum ...
                filtering');
307     end
308 end
309 % **** Debug code (end)
310
311
312 %%%%%%%% Estimation of the Radii of Circles %%%%%%%%%%%%
313
314 % Stop if no need to estimate the radii of circles
315 if ¬func_compu_radii,
316     varargout{1} = circen;
317     return;
318 end
319
320 % Parameters for the estimation of the radii of circles
321 fltr4SgnCv = [2 1 1];
322 fltr4SgnCv = fltr4SgnCv / sum(fltr4SgnCv);
323
324 % Find circle's radius using its signature curve
325 cirrad = zeros( size(circen,1), 1 );
```

```
326
327 for k = 1 : size(circen,1),
328     % Neighborhood region of the circle for building ...
        the sgn. curve
329     circen_round = round( circen(k,:) );
330     SCvR_I0 = circen_round(2) - prm_r_range(2) - 1;
331     if SCvR_I0 < 1,
332         SCvR_I0 = 1;
333     end
334     SCvR_I1 = circen_round(2) + prm_r_range(2) + 1;
335     if SCvR_I1 > size(grdx,1),
336         SCvR_I1 = size(grdx,1);
337     end
338     SCvR_J0 = circen_round(1) - prm_r_range(2) - 1;
339     if SCvR_J0 < 1,
340         SCvR_J0 = 1;
341     end
342     SCvR_J1 = circen_round(1) + prm_r_range(2) + 1;
343     if SCvR_J1 > size(grdx,2),
344         SCvR_J1 = size(grdx,2);
345     end
346
347     % Build the sgn. curve
348     SgnCvMat_dx = repmat( (SCvR_J0:SCvR_J1) - ...
        circen(k,1) , ...
349         [SCvR_I1 - SCvR_I0 + 1 , 1] );
350     SgnCvMat_dy = repmat( (SCvR_I0:SCvR_I1)' - ...
        circen(k,2) , ...
351         [1 , SCvR_J1 - SCvR_J0 + 1] );
352     SgnCvMat_r = sqrt( SgnCvMat_dx .^2 + SgnCvMat_dy ...
        .^2 );
353     SgnCvMat_rp1 = round(SgnCvMat_r) + 1;
354
355     f4SgnCv = abs( ...
356         double(grdx(SCvR_I0:SCvR_I1, SCvR_J0:SCvR_J1)) ...
            .* SgnCvMat_dx + ...
357         double(grdy(SCvR_I0:SCvR_I1, SCvR_J0:SCvR_J1)) ...
            .* SgnCvMat_dy ...
358         ) ./ SgnCvMat_r;
359     SgnCv = accumarray( SgnCvMat_rp1(:) , f4SgnCv(:) );
360
361     SgnCv_Cnt = accumarray( SgnCvMat_rp1(:) , ...
        ones(numel(f4SgnCv),1) );
362     SgnCv_Cnt = SgnCv_Cnt + (SgnCv_Cnt == 0);
363     SgnCv = SgnCv ./ SgnCv_Cnt;
364
365     % Suppress the undesired entries in the sgn. curve
366     % --- Radii that correspond to short arcs
```

```
367     SgnCv = SgnCv .* ( SgnCv_Cnt ≥ (pi/4 * ...
            [0:(numel(SgnCv_Cnt)-1)]') );
368     % -- Radii that are out of the given range
369     SgnCv( 1 : (round(prm_r_range(1))+1) ) = 0;
370     SgnCv( (round(prm_r_range(2))+1) :  end ) = 0;
371
372     % Get rid of the zero radius entry in the array
373     SgnCv = SgnCv(2:end);
374     % Smooth the sgn. curve
375     SgnCv = filtfilt( fltr4SgnCv , [1] , SgnCv );
376
377     % Get the maximum value in the sgn. curve
378     SgnCv_max = max(SgnCv);
379     if SgnCv_max ≤ 0,
380         cirrad(k) = 0;
381         continue;
382     end
383
384     % Find the local maxima in sgn. curve by 1st order ...
            derivatives
385     % -- Mark the ascending edges in the sgn. curve as ...
            1s and
386     % -- descending edges as 0s
387     SgnCv_AscEdg = ( SgnCv(2:end) - SgnCv(1:(end-1)) ) ...
            > 0;
388     % -- Mark the transition (ascending to descending) ...
            regions
389     SgnCv_LMmask = [ 0; 0; SgnCv_AscEdg(1:(end-2)) ] & ...
            (¬SgnCv_AscEdg);
390     SgnCv_LMmask = SgnCv_LMmask & [ ...
            SgnCv_LMmask(2:end) ; 0 ];
391
392     % Incorporate the minimum value requirement
393     SgnCv_LMmask = SgnCv_LMmask & ...
394         ( SgnCv(1:(end-1)) ≥ (prm_multirad * ...
              SgnCv_max) );
395     % Get the positions of the peaks
396     SgnCv_LMPos = sort( find(SgnCv_LMmask) );
397
398     % Save the detected radii
399     if isempty(SgnCv_LMPos),
400         cirrad(k) = 0;
401     else
402         cirrad(k) = SgnCv_LMPos(end);
403         for i_radii = (length(SgnCv_LMPos) - 1) : -1 : 1,
404             circen = [ circen; circen(k,:) ];
405             cirrad = [ cirrad; SgnCv_LMPos(i_radii) ];
406         end
407     end
```

```
408 end
409
410 % Output
411 varargout{1} = circen;
412 varargout{2} = cirrad;
413 if nargout > 3,
414     varargout{3} = dbg_LMmask;
415 end
```

Client.m

```
clear all
close all
clc
%inizializzazione scheda I/O
dispositivo=digitalio('nidaq',1);
dato=addline(dispositivo,0:7,'out');
%cartella in cui sono memorizzati i dati dell'esperimento
source = 'Y:/data/Filippo/clc/nd250/';
%file di log
filename = strcat(source,'log.txt');
stop=0;
ns=0;%numero di commutazioni del segnale
while stop==0
clc;
pause(3);
logfile=fopen(filename,'r');
t=fscanf(logfile,'%f',1);
fclose(logfile);
    if t==0
        pause(4);
        stop=0;
    elseif t==1;
        logfile=fopen(filename,'wt');
        fprintf(logfile,'%f',0);
        fclose(logfile);
        disp('moving syringe up');
        stepperUP(dispositivo,dato);
        ns=ns+1;
        stop=0;
    elseif t==-1;
        logfile=fopen(filename,'wt');
        fprintf(logfile,'%f',0);
        fclose(logfile);
        disp('moving syringe down');
        stepperDOWN(dispositivo,dato);
        ns=ns+1;
        stop=0;
    elseif t==2;
        stop=1;
    end
end
ns=ns-1;
```

Crop1stImage.m

```
function CropRect=Crop1stImage(source,image)
i=imread(strcat(source,image));
[junk, cr]=imcrop(imadjust(i));
CropRect=cr;
clear i cr junk;
end
```

ExperimentLiveCheck.m

```
%This script controls bright-field images to analyze ...
    how an experiment
%is developing
clc
close all
%experiment path
path='y:/data/Filippo/clc/ndxxx/';
index=dir(strcat(path,'*c1.tif'));
figure
for i=1:(length(index)-1)%:120
if i<10
   file1=strcat('t00000',num2str(i),'c1.tif');
   file2=strcat('t00000',num2str(i),'c2.tif');
elseif i>=10&&i<100
   file1=strcat('t0000',num2str(i),'c1.tif');
   file2=strcat('t0000',num2str(i),'c2.tif');
elseif i>=100&&i<1000
   file1=strcat('t000',num2str(i),'c1.tif');
   file2=strcat('t000',num2str(i),'c2.tif');
end
title1=strcat('Image','',file1);
title2=strcat('Image','',file2);
filename1 = strcat(path,file1);
filename2 = strcat(path,file2);
temp1 = imread(filename1);
temp1 = imadjust(temp1);
temp2 = imread(filename2);
temp2 = imadjust(temp2);

subplot(1,2,1)
hold on;
title(title1);
imshow(temp1);
hold off;

subplot(1,2,2)
hold on;
title(title2);
imshow(temp2);
hold off;

pause(.25);
end
```

FindImage.m

```
function t=FindImage(source,filename)
path=dir(source);
s=[];
for i=3:length(path)
    s=[s;cellstr(path(i).name)];
end
f=strcmp(filename,s);
if max(f)==1
    t=1;
else
    t=0;
end
end
```

GFevaluateMean2.m

```
function [GFMean ...
    RFMean]=GFevaluateMean2(BFimgPath,GFimgPath,...
RFimgPath,cropRect,varargin)

    params;

    global origBFImg;
    global ORIG_IMG;
    global decTreeClassifier;
    global efmHash;

    efmHash=containers.Map();

    decTreeClassifier=[];
    optargin = size(varargin,2);

    if optargin≥1
        DEBUG=varargin{1};
    else
        DEBUG=0;
    end

    if optargin==2

        decTreeClassifier= varargin{2};

    end

    disp('Reading...');
    tic;

    BFImg=imread(BFimgPath);
    pause(1);
    GFImg=imread(GFimgPath);
    pause(1);
    RFImg=imread(RFimgPath);

    BFImg=imcrop(BFImg,cropRect);
    GFImg=imcrop(GFImg,cropRect);
    RFImg=imcrop(RFImg,cropRect);

    ORIG_IMG=BFImg;

    h = fspecial('gaussian',GLP_HSIZE,GLP_SIGMA);
```

```
44      BFImg=imfilter(BFImg,h);
45
46      BFImg=imadjust(BFImg);
47
48      origBFImg=BFImg;
49
50      h = fspecial('gaussian', [5, 5],0.7);
51      BFImg=imfilter(BFImg,h);
52
53      if DEBUG
54          hold off;
55          imshow(BFImg);
56          hold on;
57      end
58
59      toc
60
61      disp('Pre processing...');
62      tic
63
64      [BFImg bounding_boxis] =preProcessedImg(BFImg); ...
           %preprocessed image
65
66      toc
67      disp('Object detection/filtering...');
68      tic;
69
70      [circen, cirrad] = findObjs(BFImg,bounding_boxis); ...
           %array containing the cells found in the image
71
72      toc
73
74      if DEBUG
75          text(10,20,BFimgPath,'color','r');
76
77          for k=1:length(circen)
78
79              drawCircle(circen(k,1),circen(k,2),cirrad(k));
80
81          end
82          pause (0.5);
83      end
84
85      disp('creating mask...');
86      tic;
87      mask=maskFromObjs(GFImg,size(GFImg),circen,cirrad);
88      toc;
89
90      disp('evaluating means...');
```

```
91      tic;
92      GF=GFImg.*uint16(mask);
93
94      GFMean=sum(sum(GF))/sum(sum(mask));
95
96      GFinv=GFImg.*uint16(¬mask);
97
98      GFinvMean=sum(sum(GFinv))/sum(sum(¬mask));
99
100     GFMean=GFMean-GFinvMean;
101
102     RF=RFImg.*uint16(¬mask);
103     RFMean=sum(sum(RF))/sum(sum(¬mask));
104     RFinv=RFImg.*uint16(mask);
105     RFinvMean=sum(sum(RFinv))/sum(sum(mask));
106     RFMean=RFMean-RFinvMean;
107
108     toc
109
110
111
112 end
113
114 %Circular Hough transform is used to find circles in ...
       the current frame.
115 %yeast cells present a circle like shape,so this ...
       method  worka quite good in detecting yeast cells ...
       in images.
116 function [circen cirrad] = findObjs(img,bounding_boxis)
117
118 params;
119
120 circen=[];
121 cirrad=[];
122
123 global origBFImg;
124 global ORIG_IMG;
125 global DEBUG;
126
127
128 for k=1:length(bounding_boxis)
129
130     bb=bounding_boxis{k};
131
132     if isempty(bb)
133         continue;
134     end
135
136
```

```
137     [croppedImg cropRect]=imcrop(img,bb);
138
139     if size(croppedImg,1)<32 || size(croppedImg,2)<32
140         continue;
141     end
142
143     if DEBUG
144
145         rectangle('Position',bb);
146     end
147
148     [acc, cc, cr] = CircularHough_Grd(croppedImg,...
149     HOUGH_RADIUS_SEARCH_RANGE,0,8,1);
150
151     for j=1:length(cr)
152
153         x=bb(1)+cc(j,1);
154         y=bb(2)+cc(j,2);
155
156         toAdd=true;
157         for h=1:length(cirrad)
158
159             if norm(circen(h,:)-[x ...
                    y])<MIN_CIRCLE_DISPLACEMENT
160
161                 toAdd=false;
162             end
163         end
164         if toAdd
165
166             circen=[circen ; [x y]];
167             cirrad=[cirrad ; cr(j)];
168         end
169     end
170
171
172 end
173
174 [circen cirrad]=filteredHoughResult(origBFImg,ORIG_IMG,...
175 circen,cirrad);
176
177
178 end
179
180 %plot a circle given center and radius
181 function drawCircle(x,y,r)
182
183 nseg=1200;
184 theta = 0 : (2 * pi / nseg) : (2 * pi);
```

```
185 pline_x = r * cos(theta) + x;
186 pline_y = r * sin(theta) + y;
187 plot(pline_x,pline_y);
188
189 end
190
191
192 %image preprocessing: select image regions where likly ...
       there is some cell
193 %in bright field microscopy images yeast cells show a ...
       halo arond them: we can
194 %take advantage of that. The Hough transform will be ...
       much faster on
195 %preprocessed image when image regions containing ...
       cells are small compared
196 %to the whole image
197 function [I2 bounding_boxis]=preProcessedImg(I)
198
199 bounding_boxis={};
200
201 I=imadjust(I);
202
203 th=graythresh(I);
204
205 BW=im2bw(I,th);
206 BW=imclose(BW,strel('disk',5));
207
208 L=bwlabel(BW);
209 s  = regionprops(L, 'ConvexHull','Area','BoundingBox');
210
211 count=0;
212 BW2=zeros(size(I));
213
214
215 for i=1:length(s)
216
217     if s(i).Area>10^2
218
219         count=count+1;
220
221         polygon=s(i).ConvexHull;
222
223         %plot(polygon(:,1),polygon(:,2),'r');
224
225         mask=poly2mask(polygon(:,1),polygon(:,2),...
226         size(I,1),size(I,2));
227
228         BW2=BW2+(mask);
229
```

```
230         bounding_boxis{i}=s(i).BoundingBox;
231
232
233     end
234 end
235
236 BW2=logical(BW2);
237
238
239 I2=I.*uint16(BW2);
240
241
242
243 end
244
245
246 %try to extract the circles that surround a yeast cell ...
       from circles
247 %detected by the hough transform
248 function [circen ...
       cirrad]=filteredHoughResult(img,origImg,circen,cirrad)
249
250 params;
251 global decTreeClassifier;
252
253 idx=find(cirrad<1);
254 cirrad(idx)=[];
255 circen(idx,:)=[];
256
257
258 toRemove=[];
259
260 if isempty(decTreeClassifier)
261 for h=1:length(cirrad)
262
263     counts=extractFeatures(img,circen(h,1),...
264     circen(h,2),cirrad(h));
265
266
267     [M m_ind]=max(counts);
268
269     if m_ind>MAX_INT_BINS
270
271         toRemove=[toRemove h];
272
273     end
274 end
275
276 else
```

```
277
278
279     for h=1:length(cirrad)
280
281     morphs=extractFeaturesForDt(origImg,circen(h,1),...
282     circen(h,2),cirrad(h));
283
284     cl=decTreeClassifier.eval(morphs);
285
286     if ¬logical(str2num(cl{1}))
287         toRemove=[toRemove h];
288     end
289
290     end
291
292 end
293
294
295 circen(toRemove,:)=[];
296 cirrad(toRemove)=[];
297
298 end
299
300 function morphs = extractFeaturesForDt(I,x,y,r)
301
302
303 croppedI=imcrop(I,[x-r y-r 2*r 2*r]);
304 croppedIad=imadjust(croppedI);
305 bw=im2bw(croppedIad,graythresh(croppedIad));
306
307 nseg=30;
308 theta = 0 : (2 * pi / nseg) : (2 * pi);
309 pline_x = r * cos(theta)+r;
310 pline_y = r * sin(theta)+r;
311
312 m=poly2mask(pline_x,pline_y,size(croppedI,1),...
313 size(croppedI,2));
314
315
316 BPN=sum(sum(¬m)); %black pixel number
317
318 [counts x]=imhist(croppedIad.*uint16(m),10);
319 counts(1)=counts(1)-BPN;
320 counts=counts';
321
322
323
324 bw=imcomplement(bw);
325 bw=imfill(bw,'holes');
```

```
326 L=logical(bw);
327
328
329 s=regionprops(L,croppedIad,'Area','Eccentricity',...
330 'Centroid','Solidity','MeanIntensity');
331
332 Areas=cat(1,s.Area);
333
334 [A ind]=max(Areas);
335
336 if ¬isempty(s)
337     morphs=[A/(2*pi*r^2)  s(ind).MeanIntensity/2^16 ...
            s(ind).Solidity  norm(s(ind).Centroid-[r r])/r];
338 else
339     morphs=[0  0 0 1];
340 end
341 mean= sum(sum(croppedIad.*uint16(m)))/sum(sum(m))/2^16;
342 morphs=[ counts mean morphs];
343
344 end
345
346 function counts = extractFeatures(I,x,y,r)
347
348
349 croppedI=imcrop(I,[x-r y-r 2*r 2*r]);
350 croppedIad=imadjust(croppedI);
351
352 nseg=30;
353 theta = 0 : (2 * pi / nseg) : (2 * pi);
354 pline_x = r * cos(theta)+r;
355 pline_y = r * sin(theta)+r;
356 m=poly2mask(pline_x,pline_y,...
357 size(croppedI,1),size(croppedI,2));
358
359 BPN=sum(sum(¬m)); %black pixel number
360
361 [counts x]=imhist(croppedIad.*uint16(m),10);
362 counts(1)=counts(1)-BPN;
363 counts=counts';
364
365
366 end
367
368
369 function mask=maskFromObjs(GFImg,imSize,circen,cirrad)
370
371
372    mask=false(imSize(1),imSize(2));
373
```

```
374     for ind=1:length(cirrad)
375
376
377             x=circen(ind,1);
378             y=circen(ind,2);
379             r=cirrad(ind);
380
381             nseg=30;
382             theta = 0 : (2 * pi / nseg) : (2 * pi);
383             pline_x = r * cos(theta) + x;
384             pline_y = r * sin(theta) + y;
385             m=poly2mask(pline_x,pline_y,imSize(1),...
386             imSize(2));
387
388             croppedI=imcrop(imadjust(GFImg),[x-r y-r ...
                    2*r 2*r]);
389             croppedIad=imadjust(croppedI);
390             bw=im2bw(croppedIad,graythresh(croppedIad));
391
392             L=bwlabel(bw,4);
393
394             s=regionprops(L,'Area');
395             Areas=cat(1,s.Area);
396             [A ind]=max(Areas);
397
398             clear A;
399
400             bw=L==ind;
401
402             startX=round(y-r);
403             startY=round(x-r);
404             countX=size(bw,1);
405             countY=size(bw,2);
406
407             if startX<=0
408                 startX=1;
409             end
410             if startY<=0
411                 startY=1;
412             end
413
414             m(startX:startX+countX-1,startY:...
415             startY+countY-1)=m(startX:startX...
416             +countX-1,startY:startY+countY-1)&bw;
417
418
419
420             mask=mask | m;
421
```

```
422         end
423
424
425     end
```

IRMA5.m

```
function dy = IRMA5(t,y,Z)
K=[0,0.040432405,1,0.0356,0.0221812275,..
0.0001492861,0.0008827352,0.0372,...
0.0477845334,0.2013609861,0.002970814,...
0.002237756,0.2,0.09375,0.421662678,...
0.00074,0.014695,1.814,0.098045,...
0.167615,0.00061,0.018194148,...
1.814,0.05,9,3,9;];
K(28)= 0.0900;
K(29)= 0.0154;
K(30)= 0.0154;


var=load('galtime.mat');
galtime=var.galtime;
u=1/2*(sign(t+galtime*60)-sign(t-galtime*60));

v=1;
deg=1;
alfa3=1;
deg3=1;
alfa4=1;
deg4=1;
alfa6=1;
deg6=1;
dec=1;
% CBF1 mRNA
dy(1,1)= ((K(1)+ v*K(2)* (Z(3)/( (K(3)+Z(3)) *( ...
    1+(y(5)/K(4)))  ))  -deg* K(5)*y(1)));
% GAL4 mRNA
dy(2,1) = (K(6)+K(7)*(y(1) /(K(8) +y(1)) ) -  K(9) *y(2));
% SWI5 mRNA (note that the values of 3 parameters ...
    change depending on the medium, galactose
% or glucose)
dy(3,1) =(K(11)*alfa3+ (K(12)*(1-u)+u*(K(12)*K(25)) )  ...
    *(y(2).^4./(  (K(14)*(1-u)+u*(K(14)/K(27)) ).^4 ...
    +y(2).^4  .*( 1+(y(4).^4./(   ...
    (K(13)*(1-u)+u*(K(13)*K(26)) ).^4)  ))  )    ) ...
    -deg3*K(15)*y(3));
% GAL80 mRNA
dy(4,1) =(K(16)+K(17)*(y(3)/(K(18)+y(3)) )- K(19) ...
    *y(4) );
% ASH1 mRNA
```

```
37 dy(5,1) =(K(21)*alfa4+ ...
      K(22)*(y(3)/(K(23)+y(3)))-K(24)*deg4*y (5));
38
39 end
```

IRMA5nodelay.m

```
function dy = IRMA5_nodelay(t,y)
%IRMA model with 5 state variables
K=[0,0.040432405,1,0.0356,0.0221812275,..
0.0001492861,0.0008827352,0.0372,...
0.0477845334,0.2013609861,0.002970814,...
0.002237756,0.2,0.09375,0.421662678,...
0.00074,0.014695,1.814,0.098045,...
0.167615,0.00061,0.018194148,...
1.814,0.05,9,3,9;];
var=load('galtime.mat');
galtime=var.galtime;
u=1/2*(sign(t+galtime*60)-sign(t-galtime*60));

v=1;
deg=1;
alfa3=1;
deg3=1;
alfa4=1;
deg4=1;
alfa6=1;
deg6=1;
dec=1;
% CBF1 mRNA
dy(1,1)= ((K(1)+ v*K(2)* (y(3)/( (K(3)+y(3)) *( ...
    1+(y(5)/K(4)))  ))  -deg* K(5)*y(1)));
% GAL4 mRNA
dy(2,1) = (K(6)+K(7)*(y(1) /(K(8) +y(1)) ) -  K(9) *y(2));
% SWI5 mRNA (note that the values of 3 parameters ...
    change depending on the medium, galactose
% or glucose)
dy(3,1) =(K(11)*alfa3+ (K(12)*(1-u)+u*(K(12)*K(25)) )  ...
    *(y(2).^4./(  (K(14)*(1-u)+u*(K(14)/K(27)) ).^4 ...
    +y(2).^4  .*( 1+(y(4).^4./(   ...
    (K(13)*(1-u)+u*(K(13)*K(26)) ).^4)  ))  )    ) ...
    -deg3*K(15)*y(3));
% GAL80 mRNA
dy(4,1) =(K(16)+K(17)*(y(3)/(K(18)+y(3)) )- K(19) ...
    *y(4) );
% ASH1 mRNA
dy(5,1) =(K(21)*alfa4+ ...
    K(22)*(y(3)/(K(23)+y(3)))-K(24)*deg4*y (5));

end
```

LinearModel.m

```
function dy = IRMA5_nodelay(t,y)
%IRMA model with 5 state variables
K=[0,0.040432405,1,0.0356,0.0221812275,..
0.0001492861,0.0008827352,0.0372,...
0.0477845334,0.2013609861,0.002970814,...
0.002237756,0.2,0.09375,0.421662678,...
0.00074,0.014695,1.814,0.098045,...
0.167615,0.00061,0.018194148,...
1.814,0.05,9,3,9;];
var=load('galtime.mat');
galtime=var.galtime;
u=1/2*(sign(t+galtime*60)-sign(t-galtime*60));
u=interp1(t1,u,t);

v=1;
deg=1;
alfa3=1;
deg3=1;
alfa4=1;
deg4=1;
alfa6=1;
deg6=1;
dec=1;
% CBF1 mRNA
dy(1,1)= ((K(1)+ v*K(2)* (y(3)/( (K(3)+y(3)) ...
*( 1+(y(5)/K(4)))  ))  -deg* K(5)*y(1)));
% GAL4 mRNA
dy(2,1) = (K(6)+K(7)*(y(1) /(K(8) +y(1)) ) -  K(9) *y(2));
% SWI5 mRNA (note that the values of 3 parameters...
 change depending on the medium, galactose
% or glucose)
dy(3,1) =(K(11)*alfa3+ (K(12)*(1-u)+u*(K(12)*K(25)) )...
  *(y(2).^4./(  (K(14)*(1-u)+u*(K(14)/K(27)) ).^4 ...
     +y(2).^4...
    .*( 1+(y(4).^4./(   (K(13)*(1-u)+u*(K(13)*K(26)) ...
       ).^4)...
      ))  )    ) -deg3*K(15)*y(3));
% GAL80 mRNA
dy(4,1) =(K(16)+K(17)*(y(3)/(K(18)+y(3)) )- K(19) ...
    *y(4) );
% ASH1 mRNA
dy(5,1) =(K(21)*alfa4+ K(22)*(y(3)/(K(23)+y(3)))...
-K(24)*deg4*y (5));

```

```
42 end
```

params.m

```
1     %SEGMETATION PARAMETERS
2
3
4
5  % The possible minimum and maximum radii of the ...
       circles to be searched
6
7  HOUGH_RADIUS_SEARCH_RANGE = [8 18];
8
9  % Gaussian lowpass filter window size
10
11 GLP_HSIZE=[5 5];
12
13 %Gaussian low pass filter standard deviation
14
15 GLP_SIGMA=0.5;
16
17 % Number of intensity bins from 1 to BIN_NUM having ...
       the largest values of
18 % intensity values
19
20 BIN_NUM=10;
21 MAX_INT_BINS=4;
22
23 % Minumum allowed displacement between tow circles's ...
       centers
24
25 MIN_CIRCLE_DISPLACEMENT=3;
26
27
28    %TRACKING/LINEAGE PARAMETERS
29
30 % cells detected and tracked in less than ...
       MIN_TRAJECTORY_LEN adjacent frames are not considered
31
32 MIN_TRAJECTORY_LEN=30;
33
34
35 % maximum allowed displacement between mother and ...
       doughter's cells circles
36
37 MAX_MOTHER_DOUGHTER_DISPACEMENT=32;
38
```

```
39 % maximum allowed consecutive segmentation misses for ...
       a cell
40
41 MAX_MISSES=3;
```

ReadMeans.m

```
close all
clear all
clc

%cartella in cui sono memorizzati i dati ...
    dell'esperimento in corso
source = 'y:/data/Filippo/clc/ndxxx/';

gmeanspath = strcat(source,'gmeans.txt');
rmeanspath = strcat(source,'rmeans.txt');

varfile = strcat(source,'variables.mat');
load(varfile);
%leggo i valori di GFP da file
disp('Reading GFP values...');
gfile = fopen ( gmeanspath, 'r');
gmeans=fscanf(gfile,'%g',[inf]);
fclose (gfile);

%leggo i valori di RFP da file
disp('Reading RFP values...');
rfile = fopen ( rmeanspath, 'r');
rmeans=fscanf(rfile,'%g',[inf]);
fclose (rfile);

figure
plot(gmeans,'g');

figure
plot(rmeans,'r');
title('measured input');
```

segmentazione.m

```
clc;
%clear all;
path='y:/data/Filippo/clc/ndxxx/';
image='t000001c1.tif';

temp=imread(strcat(path,image));
[junk, CropRect]=imcrop(imadjust(temp));
clear temp;

plantstatehigh=[];
plantinputhigh=[];

for i=1:1:120
    disp(['processando immagine: ',num2str(i)]);
if i<10
   file1=strcat('t00000',num2str(i),'c1.tif');
   file2=strcat('t00000',num2str(i),'c2.tif');
   file3=strcat('t00000',num2str(i),'c3.tif');
elseif i≥10&&i<100
   file1=strcat('t0000',num2str(i),'c1.tif');
   file2=strcat('t0000',num2str(i),'c2.tif');
   file3=strcat('t0000',num2str(i),'c3.tif');
elseif i≥100&&i<1000
   file1=strcat('t000',num2str(i),'c1.tif');
   file2=strcat('t000',num2str(i),'c2.tif');
   file3=strcat('t000',num2str(i),'c3.tif');
end
%definisco i files + il loro path
filename1=strcat(path,file1);
filename2=strcat(path,file2);
filename3=strcat(path,file3);


%Processo l'immagine
[gm rm]=GFevaluateMean2(filename1,filename2,filename3,...
CropRect,0,[]);
plantstatehigh=[plantstatehigh,gm];
plantinputhigh=[plantinputhigh,rm];
end

plantstatelow=[];
plantinputlow=[];
for i=110:1:120
    disp(['processando immagine: ',num2str(i)]);
```

```
45 if i<10
46    file1=strcat('t00000',num2str(i),'c1.tif');
47    file2=strcat('t00000',num2str(i),'c2.tif');
48    file3=strcat('t00000',num2str(i),'c3.tif');
49 elseif i≥10&&i<100
50    file1=strcat('t0000',num2str(i),'c1.tif');
51    file2=strcat('t0000',num2str(i),'c2.tif');
52    file3=strcat('t0000',num2str(i),'c3.tif');
53 elseif i≥100&&i<1000
54    file1=strcat('t000',num2str(i),'c1.tif');
55    file2=strcat('t000',num2str(i),'c2.tif');
56    file3=strcat('t000',num2str(i),'c3.tif');
57 end
58 %definisco i files + il loro path
59 filename1=strcat(path,file1);
60 filename2=strcat(path,file2);
61 filename3=strcat(path,file3);
62
63
64 %Processo l'immagine
65 [gm rm]=GFevaluateMean2(filename1,filename2,filename3,...
66     CropRect,0,[]);
67 plantstatelow=[plantstatelow,gm];
68 plantinputlow=[plantinputlow,rm];
69 end
70
71 highss = mean(plantstatehigh)
72 lowss = mean(plantstatelow)
```

state0_choose_reference.m

```
1  function ...
       [r,exp_time]=state0_choose_reference(upper_bound,...
2  lower_bound)
3  choice=0;
4  r=[];exp_time=0;
5      while choice~=1&&choice~=2&&choice~=3
6          disp('Digita 1 per scegliere come riferimento ...
               una onda triangolare');
7          disp('Digita 2 per scegliere come riferimento ...
               una finestra con fronte di discesa lineare');
8          disp('Digita 3 per scegliere come riferimento ...
               una finestra rettangolare');
9          choice=input('1, 2 o 3?: ');
10     end
11     if choice == 1
12          [r,exp_time] = ...
                onda_triangolare(upper_bound,lower_bound);
13     elseif choice == 2
14          [r,exp_time] = ...
                fin_discesa_lineare(upper_bound,...
15          lower_bound);
16     elseif choice == 3
17          [r,exp_time] = ...
                fin_rettangolare(upper_bound,lower_bound);
18     end
19     clc;
20
21 end
22
23 function ...
       [r,exp_time]=onda_triangolare(upper_bound,lower_bound)
24 clc;
25 percentuale = input('inserisci il picco del ...
       riferimento come percentuale della escursione dello ...
       stato: ');
26 max = ...
       lower_bound+(upper_bound-lower_bound)*(percentuale/100);
27 disp(['il valore massimo del riferimento come onda ...
       triangolare vale: ',num2str(max)]);
28 exp_time = input('inserisci la durata dell esperimento ...
       in minuti: ');
29 rise_time = input('indica il tempo di salita del ...
       riferimento in minuti: ');
30 axis1 = linspace(0,rise_time,rise_time);
```

```
31 axis2 = linspace(rise_time,exp_time,(exp_time-rise_time));
32 reference1 = ...
       lower_bound+((max-lower_bound)/rise_time)*axis1;
33 reference2 = max - ((max-lower_bound)/(exp_time - ...
       rise_time))*(axis2);
34 reference2 = (max-reference2(1)) + reference2;
35
36 r = [reference1,reference2];
37
38
39 end
40
41 function ...
       [r,exp_time]=fin_discesa_lineare(upper_bound,lower_bound)
42 clc;
43 percentuale = input('inserisci il picco del ...
       riferimento come percentuale della escursione dello ...
       stato: ');
44 max = ...
       lower_bound+(upper_bound-lower_bound)*(percentuale/100);
45 disp(['il valore massimo del riferimento vale: ...
       ',num2str(max)]);
46 exp_time = input('inserisci la durata dell esperimento ...
       in minuti: ');
47 rect_time = input('indica la durata della finestra in ...
       minuti: ');
48 axis1 = linspace(0,rect_time,rect_time);
49 axis2 = linspace(rect_time,exp_time,(exp_time-rect_time));
50 reference1 = axis1;
51 reference1(1:end) = max;
52 reference2 = max - ((max-lower_bound)/(exp_time - ...
       rect_time))*(axis2);
53 reference2 = (max-reference2(1)) + reference2;
54
55 r = [reference1,reference2];
56 end
57
58 function ...
       [r,exp_time]=fin_rettangolare(upper_bound,lower_bound)
59 clc;
60 percentuale = input('inserisci il picco del ...
       riferimento come percentuale della escursione dello ...
       stato: ');
61 max = ...
       lower_bound+(upper_bound-lower_bound)*(percentuale/100);
62 disp(['il valore massimo del riferimento vale: ...
       ',num2str(max)]);
63 exp_time = input('inserisci la durata dell esperimento ...
       in minuti: ');
```

```
64 rect_time = input('indica la durata della finestra in ...
      minuti: ');
65 axis1 = linspace(0,rect_time,rect_time);
66 axis2 = linspace(rect_time,exp_time,(exp_time-rect_time));
67 reference1 = axis1;
68 reference1(1:end) = max;
69 reference2 = axis2;
70 reference2(1:end) = lower_bound;
71
72 r = [reference1,reference2];
73 end
```

state0_ModSim.m

```
function [cbf1,y0] = state0_ModSim(tspan)
%STATE0_MODSIM
%   calcola l'andamento dello stato del modello a ...
    partire dallo steady
%   state alto.
%   cbf1 ? l'andamento dello stato della proteina che ...
    ci interessa, y0 ? il
%   vettore di 7 componenti contenente le condizioni ...
    iniziali per la
%   successiva simulazione del modello. tspan è un ...
    intervallo di tempo, ad
%   esempio per simulare 6 ore tspan=[0 360], tspan ...
    viene definito nel main.

%carica le condizioni iniziali
struct=load('gal5.mat');
gal=(struct.gal5)';

%soluzione del modello matematico con ritardo
sol = dde23(@IRMA5,100,gal,tspan);
time=linspace(tspan(1),tspan(end),tspan(end));
Y=deval(sol,time);
tempcbf1=Y(1,:);%estraggo la prima componente del ...
   vettore di stato
%campionamento del vettore ogni 5 minuti.
 j=1;
 cbf1=[];
 for i=1:5:length(tempcbf1)
     cbf1(j)=tempcbf1(i);
     j=j+1;
 end

 %condizioni iniziali per la successiva simulazione.
 y0=Y(:,end);
end
```

state0_ModSimNoDelay.m

```
function [yg,y0g] = state0_ModSimNoDelay(tspan)
%
%Calcola l'andamento dello stato del modello lineare ... durante
%l'inizializzazione del sistema.

%carica le condizioni iniziali
struct=load('gal5.mat');
gal=(struct.gal5)';

%soluzione del modello matematico con ritardo
sol = ode45(@IRMA5_nodelay,tspan,gal);
 time=linspace(0,tspan(2),tspan(2));
 Y=deval(sol,time);
 y0g=Y(:,end);
%tempyg e' il vettore che contiene l'andamento dello stato
 tempyg = Y(1,:);

%campionamento del vettore ogni 5 minuti.
 j=1;
 yg=[];
 for i=1:5:length(tempyg)
     yg(j)=tempyg(i);
     j=j+1;
 end

end
```

state0_PlantInit.m

```
function [f,yp,inp,state,i] = ...
    state0_PlantInit(numbinitpicsgal,numbinitpics,yp,...
inp,state,source,txtname)
f=0;%flag esito init
if state==1
else
    Write2Txt(txtname,1);
    state=1;
end

i=1;%eliminare se inizializzata ad 1 nel main.

while i≤numbinitpicsgal
    e1=1;%flag prima fase init (e1=1 prima fase OK)
    disp(['Processando immagine: ',num2str(i)]);
    [mean,rmean,l] = state0_PlantState(source,i);
        if l==1
            yp=[yp mean];
            inp=[inp rmean];
            i=i+1;


        else
            disp('errore di inizializzazione del plant');
            i=numbinitpicsgal+1;
            e1=0;
        end
end
if e1==0%se prima fase non OK
   e2=0;%non OK anche la seconda fase
else


    Write2Txt(txtname,-1);
    state=0;

    while i≤numbinitpics
    e2=1;%flag seconda fase init
    disp(['Processando immagine: ',num2str(i)]);
    [mean,rmean,l] = state0_PlantState(source,i);
    t=0;%annullo il timer di discesa della siringa(non ...
       occore dalla 2a iterazione)
        if l==1
            yp=[yp mean];
```

```
43              inp=[inp rmean];
44              i=i+1;
45
46
47          else
48              disp('errore di inizializzazione del plant');
49              i=numbinitpics+1;
50              e2=0;
51          end
52      end
53
54  end
55
56  if e2==0
57      f=0;
58  else
59      f=1;%qui se e1=1 e e2=1, allora init OK
60  end
```

state0_PlantState.m

```
function [gm,rm,f] = state0_PlantState(source,i)

load CropRect.mat;
tic;
if i<10
   file1=strcat('t00000',num2str(i),'c1.tif');
   file2=strcat('t00000',num2str(i),'c2.tif');
   file3=strcat('t00000',num2str(i),'c3.tif');
elseif i≥10&&i<100
   file1=strcat('t0000',num2str(i),'c1.tif');
   file2=strcat('t0000',num2str(i),'c2.tif');
   file3=strcat('t0000',num2str(i),'c3.tif');
elseif i≥100&&i<1000
   file1=strcat('t000',num2str(i),'c1.tif');
   file2=strcat('t000',num2str(i),'c2.tif');
   file3=strcat('t000',num2str(i),'c3.tif');
end
%definisco i files + il loro path
filename1=strcat(source,file1);
filename2=strcat(source,file2);
filename3=strcat(source,file3);

s1=0;s2=0;s3=0;

b=toc;
while ((s1==0)||(s2==0)||(s3==0))&&(b)≤500
   s1=FindImage(source,file1);
   s2=FindImage(source,file2);
   s3=FindImage(source,file3);
   b=toc;
end

if (s1==1&&s2==1&&s3==1)
%Processo l'immagine
[gm rm]=GFevaluateMean2(filename1,...
filename2,filename3,CropRect,0,[]);
f=1;%il calcolo della media Ã¨ andato a buon fine
else
    disp('errore di acquisizione immagini');
    f=0;%il calcolo della media non Ã¨ andato a buon fine
    gm=-1;
end
end
```

state0_PlantStateScaling.m

```
function [ yp ] = state0_PlantStateScaling(yp,statebounds)

for i=1:length(yp)
   if yp(i) < statebounds(1)
        yp(i)=statebounds(1);
   elseif yp(i) > statebounds(2)
        yp(i)= statebounds(2);
   end
end
```

state1_calculatingInput.m

```
function [ton,U] = ...
    state1_calculatingInput(uc,controllertime)
Δ=5/length(controllertime);
        j=0;
        for i=1:length(uc)
            if uc(i)==0
                j=j+1;
            end
        end
        toff=Δ*(j-1);
        ton=floor((5-toff)*60);

U=[controllertime,uc];
end
```

state3_MoveSyringes.m

```
function [b,state] = ...
    state3_MoveSyringes(ton,state,txtname)
tic;
if ton < 60 && ton > 0
    disp('ton < di 60 secondi');
    if state==1
      disp('state = 1, la siringa viene abbassata');
      Write2Txt(txtname,-1);
      state=0;
      b=toc;
    else
      state=0;
      b=toc;
    end
elseif ton ≥ 60
    disp('ton > di 60 secondi');
    if state==1
        state=1;
        pause(ton);
        b = toc;
            if b < 300
                Write2Txt(txtname,-1);
                state=0;
                b=toc;
            end
    elseif state==0;
        Write2Txt(txtname,1);
        state=1;
        pause(ton);
        b = toc;
            if b < 300
                Write2Txt(txtname,-1);
                state=0;
                b=toc;
            end
    end

elseif ton==0 && state==1
     Write2Txt(txtname,-1);
     state=0;
     b=toc;

elseif ton==0 && state==0
    state=0;
```

```
44      b=toc;
45
46  end
47
48  end
```

state5_PlantState.m

```
function [mean,r,f] = state5_PlantState(act,source,i)

load CropRect.mat;
tic;

if i<10
   file1=strcat('t00000',num2str(i),'c1.tif');
   file2=strcat('t00000',num2str(i),'c2.tif');
   file3=strcat('t00000',num2str(i),'c3.tif');
elseif i≥10 && i<100
   file1=strcat('t0000',num2str(i),'c1.tif');
   file2=strcat('t0000',num2str(i),'c2.tif');
   file3=strcat('t0000',num2str(i),'c3.tif');
elseif i≥100 && i<1000
   file1=strcat('t000',num2str(i),'c1.tif');
   file2=strcat('t000',num2str(i),'c2.tif');
   file3=strcat('t000',num2str(i),'c3.tif');
elseif i≥1000 && i<10000
   file1=strcat('t00',num2str(i),'c1.tif');
   file2=strcat('t00',num2str(i),'c2.tif');
   file3=strcat('t00',num2str(i),'c3.tif');
elseif i≥10000 && i<100000
   file1=strcat('t0',num2str(i),'c1.tif');
   file2=strcat('t0',num2str(i),'c2.tif');
   file3=strcat('t0',num2str(i),'c3.tif');
elseif i≥100000 && i<1000000
   file1=strcat('t',num2str(i),'c1.tif');
   file2=strcat('t',num2str(i),'c2.tif');
   file3=strcat('t',num2str(i),'c3.tif');
end

filename1=strcat(source,file1);
filename2=strcat(source,file2);
filename3=strcat(source,file3);

s1=0;s2=0;s3=0;

b=toc;
while (s1==0 || s2==0 || s3==0) && (b+act)≤500
    s1=FindImage(source,file1);
    s2=FindImage(source,file2);
    s3=FindImage(source,file3);
    b=toc;
end
```

```
45
46 if s1==1 && s2==1 && s3==1
47
48 f=1;
49 [mean r]=GFevaluateMean2(filename1,...
50 filename2,filename3,CropRect,0,[]);
51 else
52     f=0;
53     disp('errore di acquisizione immagini');
54
55 end
56 end
```

stepperDOWN.m

```
function [e_time,state] = stepperDOWN(dispositivo,dato)
k=375;
tic
    for i=1:k
        pause(0.001);
        putvalue(dato,bin2dec('00100100'));
        pause(0.001);
        pause(0.001);
        putvalue(dato,bin2dec('01000100'));
        pause(0.001);
        pause(0.001);
        putvalue(dato,bin2dec('01000001'));
        pause(0.001);
        pause(0.001);
        putvalue(dato,bin2dec('00100001'));
    end
    e_time=toc;
    state = 0;
end
```

stepperUP.m

```
function [e_time,state] = stepperUP(dispositivo,dato)
k=375;
tic
for i=1:k
        putvalue(dato,bin2dec('00100001'));
        pause(0.001);
        putvalue(dato,bin2dec('01000001'));
        pause(0.001);
        putvalue(dato,bin2dec('01000100'));
        pause(0.001);
        putvalue(dato,bin2dec('00100100'));
        pause(0.001);
end
e_time=toc;
state = 1;
end
```

ton2input.m

```
function u = ton2input(vton)
u=[];
j=1;
k=0;
for i=1:length(vton)
    k=vton(i);
    j=length(u)+1;
    if k < 60
        u(j:j+4)=0;
    elseif k ≥ 60 && k < 300
        minutes = round(k/60);
        u(j:(j+minutes-1)) = 1;
        j=length(u);
        u(j+1:j+(5-minutes)) = 0;
    elseif k == 300
        minutes = 5;
        u(j:j+4) = 1;
    end
end
end
```

Write2Txt.m

```
function [junk] = Write2Txt(txtname,n)

logfile=fopen(txtname,'wt');
fprintf(logfile,'%1.0f',n);
fclose(logfile);
clear logfile;

end
```

WriteGmeans.m

```
function [] = WriteGmeans(gmeanspath,yp)

file=fopen(gmeanspath,'wt');
fprintf(file,'%5.4g\n',yp');
fclose(file);
clear file;

end
```

WriteRmeans.m

```
function [ ] = WriteRmeans( rmeanspath,input )
file=fopen(rmeanspath,'wt');
fprintf(file,'%5.4g\n',input');
fclose(file);
clear file;
end
```

APPENDIX C

Cell growth - Day 0 and Day 1

1. 500 μL Galactose slution (20%) and 500 μL Raffinose slution (20%) are diluted in 4.5mL Synthetic Complete medium;

2. IC18 (P340) colony is picked from the plate and inoculated in the obtained SC+GAL/RAF @2% solution;

3. batch culture Opetical Density is evaluated at interval of 12 hours: dilutions are carried out on Day 0 and Day 1 so as to obtain 0.01 OD_{600} each time.

Preparation of microfluidic components - Day 2

1. Six 2 m and 5 1 m capillaries are cut;

2. Six 60 mL and 5 2 mL syringes are collected;

3. The bigger syringes are labeled from 1 through 5 (this number refers to the port the syringe will be connected to). Two "4" ports

are obtained, one for the cell loading mode and the other for the running mode;

4. Fresh Leur stub adapter are attached to the syringe bodies;

5. A microfluidic line and the metal stub is attached to the end of the line;

6. The syringe bodies are filled as follows:

   (a) Two mL syringes are all completely filled with ddH2O water (paying attention to avoid any residual air bubble within the syringe of the line);

   (b) Syringes labeled from 3 to 5 are filled with 10 mL water

   (c) Syringe 1 is filled with SC+GAL+RAG(2%)+Sulphorhodamine B(0.1%);

   (d) Syringe 2 is filled with SC+GLU(2%);

7. Both syringes and attached lines are carefully inspected for air bubbles. Fluxes are verified by rising and lowering the heights of sytinges with respect to the metal stubs. In case bubbles are found the syringe is gently tapped to remove it;

Microfluidic device setup - Day 2

1. The microfluidic device is secured to the object holde on the moving stage;

2. An exhaustive search for defects is carried out to exclude structural issues in the device (for this activity the 4X objective is usually employed);

3. The five 2 mL syringes are attached to the ports sequentially (starting from port 1 pressure is applied each time a new stub is attached) to remove air from the device;

4. The microfluidic device is further inspected at the microscope to exclude the presence of residual air bubbles withing the channels or the cell trap;

5. Once air has been removed from the device the 5 smaller syringes are disconnected from the device leaving a small water drop on the ports to generate the pressure needed to prevent air from re-entering the channels;

1. Ports 3 and 5 are connected to the bigger syringes first (syringes are fixed at 14.2 and 15 cm from the level of the microfludic device);

2. The device is ispected for the presence of air as an effect of the last step;

3. Port 1 and 2 are secured at the linear rail (90 cm) and connected to the device;

4. The device is ispected for the presence of air as an effect of the last step;

5. Flows are checked against issues by imaging the DAW area of the device and testing that rising and lowering the syringes results in the expected behavior at the mixing section;

Cell loading - Day 2

1. On Day 2 the batch culture is diluted at $OD_{600} = 0.05$ so as to obtain $OD_{600} = 1$ after 4-6 hours;

2. After having completed the flow cross checks, yeasts from the batch culture are poured in the remaining "4" syringe. Both the syringe body and the line are checked against the presence of air bubbles. At this point the metallic stub connected to the line can be atached to port 4 and the syringe body fixed at around 50 cm;

3. The syringe attached to port 5 is rised as well to 49 cm so as to prevent cells to go directly from port 4 to port 5;

4. The microscope's field of view is set on the cell trap (40X magnification) and the line connected to the yeasts suspension filled syringe is gently flicked to ease the cell trapping;

5. Once cell loading is achieved port yeasts' syringe can be replaced with the water filled syringe destined to port 4.

Image acquisition and control algorithm setup

1. The image acquisition routine is set to acquire 3 images (bright field, green fluorescence and red fluorescence) at intervals of 5 min;

2. The control algorithm can be started at this point. Basic information on the communication protocols as well as on the specifications of the experiments (duration, reference signal type etc) are provided. The control algorithm returns a simulation of the control experiment and the experiment can be launched;

BIBLIOGRAPHY

[1] P. A. Iglesias and B. P. Ingalls, *Control theory and systems biology.* MIT Press, 2010.

[2] Borisuk and Tyson, "Bifurcation analysis of a model of mitotic control in frog eggs," *Journal of Theoretical Biology*, vol. 195, pp. 69–85, Nov. 1998. PMID: 9802951.

[3] I. Cantone, L. Marucci, F. Iorio, M. A. Ricci, V. Belcastro, M. Bansal, S. Santini, M. di Bernardo, D. di Bernardo, and M. P. Cosma, "A yeast synthetic network for in vivo assessment of Reverse-Engineering and modeling approaches," *Cell*, vol. 137, pp. 172–181, Apr. 2009.

[4] M. Chaves and J. Gouze, "Exact control of genetic networks in a qualitative framework: The bistable switch example," *Automatica*, vol. In Press, Corrected Proof.

[5] N. Bagheri, J. Stelling, and F. J. Doyle, "Circadian phase resetting via single and multiple control targets," *PLoS Comput Biol*, vol. 4, p. e1000104, July 2008.

[6] G. Vahedi, B. Faryabi, J. F. Chamberland, A. Datta, and E. R.

Dougherty, "Intervention in gene regulatory networks via a stationary mean-first-passage-time control policy," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 10, pp. 2319,2331, 2008.

[7] B. Faryabi, G. Vahedi, A. Datta, J. F. Chamberland, and E. R. Dougherty, "Recent advances in intervention in markovian regulatory networks," *Current Genomics*, vol. 10, no. 7, p. 463â477, 2009.

[8] J. Uhlendorf, S. Bottani, F. Fages, P. Hersen, and G. Batt, "Towards real-time control of gene expression: controlling the HOG signaling cascade," vol. Proc. of PSB 2011.

[9] A. Milias-Argeitis, S. Summers, J. Stewart-Ornstein, I. Zuleta, D. Pincus, H. El-Samad, M. Khammash, and J. Lygeros, "In silico feedback for in vivo regulation of a gene expression circuit," *Nat Biotech*, vol. advance online publication, Nov. 2011.

[10] H. S. Black, "Stabilized Feed-Back amplifiers," *American Institute of Electrical Engineers, Transactions of the*, vol. 53, pp. 114–120, Jan. 1934.

[11] J. Slotine and W. Li, *Applied Nonlinear Control*. Prentice Hall, 1991.

[12] Y. Tu, T. S. Shimizu, and H. C. Berg, "Modeling the chemotactic response of escherichia coli to time-varying stimuli," *Proceedings of the National Academy of Sciences*, vol. 105, no. 39, pp. 14855 –14860, 2008.

[13] T. Ivancic, M. Vodovnik, R. Marinsek-Logar, and D. Stopar, "Conditioning of the membrane fatty acid profile of escherichia coli during periodic temperature cycling," *Microbiology*, vol. 155, pp. 3461 –3463, Oct. 2009.

[14] R. Gunawan and F. J. Doyle, "Phase sensitivity analysis of circadian rhythm entrainment," *Journal of Biological Rhythms*, vol. 22, pp. 180 –194, Apr. 2007.

[15] J. F. Apgar, J. E. Toettcher, D. Endy, F. M. White, and B. Tidor, "Stimulus design for model selection and validation in cell signaling," *PLoS Computational Biology*, vol. 4, p. e30, Feb. 2008.

[16] F. Menolascina, D. Bellomo, T. Maiwald, V. Bevilacqua, C. Ciminelli, A. Paradiso, and S. Tommasi, "Developing optimal input design strategies in cancer systems biology with applications to microfluidic device engineering," *BMC bioinformatics*, vol. 10, no. Suppl 12, p. S4, 2009.

[17] V. Siciliano, F. Menolascina, L. Marucci, C. Fracassi, I. Garzilli, M. N. Moretti, and D. di Bernardo, "Construction and modelling of an inducible positive feedback loop stably integrated in a mammalian cell-line," *PLoS Computational Biology*, vol. 7, p. e1002074, June 2011. PMID: 21765813.

[18] J. C. Locke and M. B. Elowitz, "Using movies to analyse gene circuit dynamics in single cells," *Nature reviews. Microbiology*, vol. 7, pp. 383–392, May 2009. PMID: 19369953 PMCID: 2853934.

[19] D. N. Breslauer, P. J. Lee, and L. P. Lee, "Microfluidics-based systems biology," *Molecular BioSystems*, vol. 2, no. 2, pp. 97–112, 2006.

[20] M. S. Ferry, I. A. Razinkov, and J. Hasty, "Microfluidics for synthetic biology: from design to execution," *Methods in Enzymology*, vol. 497, pp. 295–372, 2011. PMID: 21601093.

[21] G. M. Whitesides, "The origins and the future of microfluidics," *Nature*, vol. 442, pp. 368–373, July 2006.

[22] T. Kalisky and S. R. Quake, "Single-cell genomics," *Nat Meth*, vol. 8, pp. 311–314, Apr. 2011.

[23] M. D. Lazova, T. Ahmed, D. Bellomo, R. Stocker, and T. S. Shimizu, "Response rescaling in bacterial chemotaxis," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, pp. 13870–13875, Aug. 2011. PMID: 21808031.

[24] S. Gulati, V. Rouilly, X. Niu, J. Chappell, R. I. Kitney, J. B. Edel, P. S. Freemont, and A. J. deMello, "Opportunities for microfluidic technologies in synthetic biology," vol. 6, pp. S493–S506, Aug. 2009. PMID: 19474079 PMCID: 2843966.

[25] S. Cookson, N. Ostroff, W. L. Pang, D. Volfson, and J. Hasty, "Monitoring dynamics of single-cell gene expression over multiple cell cycles," *Molecular Systems Biology*, vol. 1, p. 20050024, 2005. PMC1681470.

[26] M. R. Bennett, W. L. Pang, N. A. Ostroff, B. L. Baumgartner, S. Nayak, L. S. Tsimring, and J. Hasty, "Metabolic gene regulation in a dynamically changing environment," *Nature*, vol. 454, no. 7208, pp. 1119–1122, 2008.

[27] U. Alon, *An introduction to systems biology*. CRC Press, 2007.

[28] M. P. Cosma, T. Tanaka, and K. Nasmyth, "Ordered recruitment of transcription and chromatin remodeling factors to a cell cycle and developmentally regulated promoter," *Cell*, vol. 97, pp. 299–311, Apr. 1999.

[29] E. Balleza, E. R. Alvarez-Buylla, A. Chaos, S. Kauffman, I. Shmulevich, and M. Aldana, "Critical dynamics in genetic regulatory networks: Examples from four kingdoms," *PLoS ONE*, vol. 3, p. e2456, June 2008.

[30] P. J. Bhat, *Galactose Regulon of Yeast*. Springer, 2008.

[31] D. W. Griggs and M. Johnston, "Regulated expression of the GAL4 activator gene in yeast provides a sensitive genetic switch for glucose repression," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 88, pp. 8597–8601, Oct. 1991.

[32] D. Lohr, P. Venkov, and J. Zlatanova, "Transcriptional regulation in the yeast GAL gene family: A complex genetic network," *FASEB Journal*, vol. 9, no. 9, pp. 777–787, 1995.

[33] M. G. Mirisola, A. Gallo, and G. D. Leo, "Ras-pathway has a dual role in yeast galactose metabolism," *FEBS Letters*, vol. 581, pp. 2009–2016, May 2007.

[34] A. Sil, S. Alam, P. Xin, L. Ma, M. Morgan, C. Lebo, M. Woods, and J. Hopper, "The Gal3p-Gal80p-Gal4p transcription switch of yeast: Gal3p destabilizes the Gal80p-Gal4p complex in response to galactose and ATP," *Molecular and Cellular Biology*, vol. 19, no. 11, pp. 7828–7840, 1999.

[35] H. de Jong, "Modeling and simulation of genetic regulatory systems: a literature review," *Journal of computational biology*, vol. 9, no. 1, pp. 67–103, 2002.

[36] J. L. Gouzè and T. Sari, "A class of piecewise linear differential equations arising in biological models," *Dynamical systems*, vol. 17, no. 4, pp. 299–316, 2002.

[37] J. Ahmad, G. Bernot, J. P. Comet, D. Lime, and O. Roux, "Hybrid modelling and dynamical analysis of gene regulatory networks with delays," *ComPlexUs*, vol. 3, no. 4, pp. 231–251, 2006.

[38] D. Liberzon, *Switching in systems and control*. Springer, 2003.

[39] F. Menolascina, M. di Bernardo, and D. di Bernardo, "Analysis, design and implementation of a novel scheme for in-vivo control of synthetic gene regulatory networks," *Automatica*, vol. 47, pp. 1265–1270, June 2011.

[40] O. Smith, "Closed control of loops with dead time," *Chem. Eng. Progress*, vol. 57, pp. 217–219, 1957.

[41] R. Majumder, B. Chaudhuri, B. C. Pal, and Q. C. Zhong, "A unified smith predictor approach for power system damping control design using remote signals," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 6, pp. 1063–1068, 2005.

[42] X. Chen and A. Serrani, "Smith predictors in nonlinear systems - application to ISS-based leader/follower trailing control," in *American Control Conference, 2007. ACC '07*, pp. 4506–4511, IEEE, July 2007.

[43] V. Utkin and H. Lee, "Chattering problem in sliding mode control systems," in *International Workshop on Variable Structure Systems, 2006. VSS'06*, pp. 346–350, IEEE, June 2006.

[44] S. M. Williams, R. G. Hoft, U. Sch, and C. A. Monterey, "Adaptive frequency domain control of PWM switched power lineconditioner," *IEEE Transactions on Power Electronics*, vol. 6, no. 4, pp. 665–670, 1991.

[45] J. C. W. Locke, J. W. Young, M. Fontes, M. J. H. Jimenez, and M. B. Elowitz, "Stochastic pulse regulation in bacterial stress response," *Science*, vol. 334, pp. 366 –369, Oct. 2011.

[46] T. Hagglund, *PID Controllers: Theory, Design, and Tuning*. ISA: The Instrumentation, Systems, and Automation Society, 2 sub ed., Jan. 1995.

[47] I. Nagrath, *Control Systems Engineering*. New Age International, Jan. 2006.

[48] T. Casci, "A small leap for adaptation," *Nat Rev Micro*, vol. 4, pp. 322–323, May 2006.

[49] K. Logg, K. Bodvard, A. Blomberg, and M. KÃ ll, "Investigations on light-induced stress in fluorescence microscopy using nuclear localization of the transcription factor msn2p as a reporter," *FEMS Yeast Research*, vol. 9, pp. 875–884, Sept. 2009. PMID: 19566686.

[50] R. A. Hoebe, C. H. Van Oven, T. W. J. Gadella, P. B. Dhonukshe, C. J. F. Van Noorden, and E. M. M. Manders, "Controlled light-exposure microscopy reduces photobleaching and phototoxicity in fluorescence live-cell imaging," *Nat Biotech*, vol. 25, pp. 249–253, Feb. 2007.

[51] T. Das, T. K. Maiti, and S. Chakraborty, "Augmented stress-responsive characteristics of cell lines in narrow confinements," *Integr. Biol.*, vol. 3, pp. 684–695, May 2011.

[52] P. Lee, T. Gaige, and P. Hung, "Microfluidic systems for live cell imaging," *Methods in Cell Biology*, vol. 102, pp. 77–103, 2011. PMID: 21704836.

[53] G. Charvin, C. Oikonomou, and F. Cross, "Long-term imaging in microfluidic devices.," *Methods in molecular biology (Clifton, NJ)*, vol. 591, p. 229, 2010.

[54] T. Squires and S. Quake, "Microfluidics: Fluid physics at the nanoliter scale," *Reviews of Modern Physics*, vol. 77, no. 3, 2005.

[55] J. Bao and D. Jed Harrison, "Measurement of flow in microfluidic networks with micrometer-sized flow restrictors," *AIChE Journal*, vol. 52, pp. 75–85, Jan. 2006.

[56] C. Voigt, *Synthetic Biology, Part A, Volume 497: Methods for Part/Device Characterization and Chassis Engineering.* Academic Press, 1 ed., June 2011.

[57] R. La Brocca, F. Menolascina, D. di Bernardo, and C. Sansone, "Segmentation, tracking and lineage analysis of yeast cells in bright field microscopy images," in *PR PS BB*, (Ravenna, Italy), pp. 45–46, Jan. 2011.

[58] S. Pedersen, "Circular hough transform," *Aalborg University, Vision, Graphics, and Interactive Systems*, 1991.

[59] G. Cuccato, G. D. Gatta, and D. di Bernardo, "Systems and synthetic biology: tackling genetic networks and complex diseases," *Heredity*, vol. 102, pp. 527–532, June 2009. PMID: 19259117.

[60] S. Mangan and U. Alon, "Structure and function of the feed-forward loop network motif.," *PNAS*, vol. 100, no. 21, pp. 11980–11985, 2003.

[61] J. D. Murray, *Mathematical Biology, Vol. 1: An Introduction.* Springer-Verlag, 2008.

[62] O. Purcell, N. J. Savery, C. S. Grierson, and M. di Bernardo, "A comparative analysis of synthetic genetic oscillators," *J R Soc Interface*, vol. 7, pp. 1503 –1524, Nov. 2010.

[63] M. A. Savageau, "Comparison of classical and autogenous systems of regulation in inducible operons," *Nature*, vol. 252, pp. 546–549, Dec. 1974.

[64] M. Gossen and H. Bujard, "Tight control of gene expression in mammalian cells by tetracycline-responsive promoters," *PNAS*, vol. 89, pp. 5547–5551, 1992.

[65] T. Dull, R. Zufferey, M. Kelly, R. Mandel, M. Nguyen, D. Trono, and L. Naldini, "A third-generation lentivirus vector with a conditional packaging system," *J Virol*, vol. 72, pp. 8463–847, 1998.

[66] T. Kafri, H. van Praag, F. H. Gage, and I. M. Verma, "Lentiviral vectors: regulated gene expression," *Mol. Ther.*, vol. 1, pp. 516–521, 2000.

[67] H. Kaufmann, X. Mazur, M. Fussenegger, and J. E. Bailey, "Influence of low temperature on productivity, proteome and protein phosphorylation of CHO cells," *Biotechnol Bioeng*, vol. 63, pp. 573–582, June 1999. PMID: 10397813.

[68] H. De Jong, "Modeling and simulation of genetic regulatory systems: A literature review," *J Comput Biol*, vol. 9, pp. 67–103, 2002.

[69] M. B. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators.," *Nature*, vol. 403, pp. 335–338, January 2000.

[70] T. S. Gardner, C. R. Cantor, and J. J. Collins, "Construction of a genetic toggle switch in Escherichia coli.," *Nature*, vol. 403, no. 6767, pp. 339–342, 2000.

[71] B. P. Kramer, A. U. Viretta, M. Daoud-El-Baba, D. Aubel, W. Weber, and M. Fussenegger, "An engineered epigenetic transgene switch in mammalian cells.," *Nat Biotechnol*, vol. 22, pp. 867–870, July 2004.

[72] M. Tigges, T. T. Marquez-Lago, J. Stelling, and M. Fussenegger, "A tunable synthetic mammalian oscillator," *Nature*, vol. 457, no. 7227, pp. 309–312, 2009.

[73] J. Stricker, S. Cookson, M. R. R. Bennett, W. H. H. Mather, L. S. S. Tsimring, and J. Hasty, "A fast, robust and tunable synthetic gene oscillator," *Nature*, vol. 456, pp. 516–519, October 2008.

[74] S. H. Siegel MR, "Inhibition of protein synthesis in vitro by cycloheximide.," *Nature*, vol. 200, pp. 675–676, 1963.

[75] V.A., "Living colors destabilized ecfp and eyfp vectors.," *Clontechniques*, vol. XIV(3), pp. 14–15, 1999.

[76] T. Maiwald and J. Timmer, "Dynamical modeling and multi-experiment fitting with PottersWheel," *Bioinformatics*, vol. 24, pp. 2037–2043, Sept. 2008. PMID: 18614583 PMCID: 2530888.

[77] P. Bogacki and L. F. Shampine, "A 3(2) pair of runge-kutta formulas," *Appl. Numer. Math.*, vol. 2, pp. 1–9, 1989.

[78] A. Becskei, B. Seraphin, and L. Serrano, "Positive feedback in eukaryotic gene networks: cell differentiation by graded to binary response conversion," *EMBO J*, vol. 20, pp. 2528–2535, May 2001.

[79] N. Yosef and A. Regev, "Impulse control: temporal dynamics in gene transcription," *Cell*, vol. 144, pp. 886–896, Mar. 2011. PMID: 21414481.

[80] N. Rosenfeld, M. B. Elowitz, and U. Alon, "Negative autoregulation speeds the response times of transcription networks," *J Mol Biol*, vol. 323, pp. 785–793, Nov. 2002.

[81] K. R. Allison and J. J. Collins, "Bacteria as control engineers," *Molecular Cell*, vol. 41, no. 1, pp. 4–5, 2011.

[82] Y. Liu, J. Slotine, and A. Barabasi, "Controllability of complex networks," *Nature*, vol. 473, pp. 167–173, May 2011.