Image registration methods for reconstructing a gene expression atlas of early zebrafish embryogenesis

Master Thesis
by
Evangelia Balanou

Patras, 2010
Image registration methods for reconstructing a gene expression atlas of early zebrafish embryogenesis

Evangelia Balanou
Electrical and Computer Engineer N.T.U.A.
Biomedical Engineer

Carried out at
DEPARTMENT OF ELECTRONIC ENGINEERING
TECHNICAL SCHOOL OF TELECOMMUNICATIONS ENGINEERING
TECHNICAL UNIVERSITY OF MADRID (ETSIT-UPM)

Tutor
Miguel Ángel Luengo Oroz
Research Engineer
Department of Electronic Engineering
Technical University of Madrid
(ETSIT-UPM)
Spain

Examining Committee

<table>
<thead>
<tr>
<th>Nicolas Pallikarakis</th>
<th>Andrés Santos</th>
<th>Dionysios Synetos</th>
<th>Panagiotis Katsoris</th>
</tr>
</thead>
<tbody>
<tr>
<td>Professor</td>
<td>Professor</td>
<td>Professor</td>
<td>Associate Professor</td>
</tr>
<tr>
<td>Department of Medicine</td>
<td>Department of Electronic Engineering</td>
<td>Department of Medicine</td>
<td>Department of Biology</td>
</tr>
<tr>
<td>University of Patras</td>
<td>Universidad Politécnica de Madrid</td>
<td>University of Patras</td>
<td>University of Patras</td>
</tr>
<tr>
<td>(supervisor)</td>
<td>(co-supervisor)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Patras, 2010
ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΣΤΗΝ
ΒΙΟΪΑΤΡΙΚΗ ΤΕΧΝΟΛΟΓΙΑ

Μέθοδοι αντιστοίχισης εικόνων για την
ανακατασκευή άτλαντα γονιδιακής
έκφρασης κατά τα αρχικά στάδια της
εμβρυογένεσης των ζεβρόψαρων

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
της
ΕΥΑΓΓΕΛΙΑΣ ΜΠΑΛΑΝΟΥ

Πάτρα, 2010
Μέθοδοι αντιστοίχισης εικόνων για την ανακατασκευή άτλαντα γονιδιακής έκφρασης κατά τα αρχικά στάδια της εμβρυογένεσης των ζεβρόψαρων

Ευαγγελία Μπαλάνου
Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.
Μηχανικός Βιοϊατρικής Τεχνολογίας

Διεξάγεται στο
DEPARTMENT OF ELECTRONIC ENGINEERING
TECHNICAL SCHOOL OF TELECOMMUNICATION ENGINEERING
TECHNICAL UNIVERSITY OF MADRID (ETSIT-UPM)

Σύμβουλος
Miguel Ángel Luengo Oroz
Research Engineer
Department of Electronic Engineering
Technical University of Madrid
(ETSIT-UPM)
Spain

Εξεταστική Επιτροπή

Νικόλαος Παλληκαράκης
Καθηγητής
Ιατρικού Τμήματος
Πανεπιστημίου Πάτρας
(επιβλέπει)

Andrés Santos
Καθηγητής
Department of Electronic Engineering
Universidad Politécnica de Madrid
(συνεπιβλέπει)

Διονύσιος Συνετός
Καθηγητής
Ιατρικού Τμήματος
Πανεπιστημίου Πάτρας

Παναγιώτης Κατσώρης
Αναπλ. Καθηγητής
Τμήματος Βιολογίας
Πανεπιστημίου Πάτρας

Πάτρα, 2010
Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.
Acknowledgements

Looking back, this Master Thesis has been for me a journey, both literally and metaphorically. This written work is just a part of all the things that I have learnt during this adventure. I hope that when others set out for such a journey, they receive the same support as I have.

A number of great people have contributed in various ways to its completion.

First of all I am grateful for the motivation and generous support I have received from Prof. Nicolas Pallikarakis, coordinator of the Postgraduate Program on Biomedical Engineering in Patras. I am proud to have been his student.

The Postgraduate Program would not be the same without Eleni Panoutsopoulou. I appreciate especially her patience and willingness to help every student in the program. I am grateful to be one of them.

I thank Prof. Andrés de Santos Lleó for accepting me to carry out my thesis with him. This offered me the opportunity to participate in the Erasmus Program and gain a vast experience from working with his research group and living in Spain.

For the accomplishment of this work I deeply thank my tutor, Miguel Ángel Luengo Oroz. Without his fruitful advice and ideas this thesis would never have been completed.

I am grateful for the support I have received from the International Relations Offices, both in the University of Patras and the Politécnica de Madrid. I thank them for their assistance in everything that came up along the way.

I would like to express my sincere gratitude to my tutor in my diploma thesis, Yiannis Katsanis, for his words of advice then moulded my way of thinking.

I thank my family for their unconditional support during the years of my studies and for teaching me to stand on my own feet.

This journey would not be possible without Kosta. Firstly, I acknowledge that his comments and editing assistance in writing this thesis were invaluable. Secondly and most importantly, I thank him for being by my side.

I consider myself lucky to have friends who share my joy. Among them, I would especially like to thank Tzina and Manto for caring and believing in me.

To all of these people and many more I owe the satisfaction of completing successfully another chapter of my life. Thank you all.
Abstract

The process of embryogenesis is governed by the expressions of groups of genes which are acting in a coordinate way. Uncovering how the expressions of these genes control the development of a multicellular organism is fundamental for developmental biology.

Gene expression atlases could capture quantitative spatio-temporal information of all genes expressed in a developing embryo. In other words, they could reveal the underlying genetic activity during the embryogenetic processes by providing information about, apart from how much and which, where and when the genes are expressed at cellular level and the interactions between them. The extraction of relative gene expression data and their simultaneous study in animal models, such as zebrafish, would be greatly facilitated by the existence of such atlases.

This Master Thesis focuses on the early zebrafish embryogenesis and its goal is to design and implement an image processing framework that will provide the means to gather the expression patterns of different genes from different embryos at a given developmental stage into a common template. The framework should work with image-based data from different embryos, each fluorescently stained to label nuclear DNA and the expression patterns of a reference gene and another gene of interest. The volumetric data from each fluorescent label are contained in different channels. Therefore the crux of the framework lies in its ability to combine appropriately the information from the different channels and deal with a three-dimensional image registration problem.

The implemented framework works with datasets of two embryos, one that serves as a template and depicts a whole embryo and another that has to be aligned with the first and depicts part of the other embryo. It is composed of different steps, responsible for preprocessing the channels, coarsely positioning the partial embryo view in the three dimensional template’s space and determining the geometrical transformation that finally aligns it with the template using as reference the gene expression pattern common to all labelled embryos. The resulting transformation is used to map the second expression patterns, thus producing their spatial expression atlas. The algorithm developed is based on the Insight Segmentation and Registration Toolkit.

The framework was evaluated with data from six embryos at the same developmental stage and four different registration methods were compared in terms of performance. Visual inspection of the results identified the combination of the correlation coefficient, as a similarity measure function between two images, with the gradient descent optimization algorithm as the most appropriate method for this specific application. The final results obtained showed that the framework achieves its goal of integrating several gene expression patterns into a common template.

This framework is ready to be used in order to construct a gene expression atlas integrating a large number of gene expression data of the zebrafish embryogenesis. In the near future, this atlas should be validated with known genetic interactions and used to unravel new ones, so that gene regulatory network models can become a reality.

Keywords
Gene expression atlas, early zebrafish embryogenesis, image registration, Insight Segmentation and Registration Toolkit (ITK)
Περίληψη

Η διαδικασία ανάπτυξης των εμβρύων καθορίζεται από τις συγχρονισμένες εκφράσεις ορισμένων γονίδιων. Το πώς αυτές ελέγχουν την δημιουργία ενός πολυκύτταρου οργανισμού από ένα κύτταρο αποτελεί αντικείμενο μελέτης της Αναπτυξιακής Βιολογίας. Ένας χάρτης γονιδιακής έκφρασης θα μπορούσε να συνδυάζει χρονικές, χωρικές και ποσοτικές πληροφορίες σχετικά με την έκφραση των γονίδιων που ενορχηστρώνουν την διαδικασία ανάπτυξης ενός εμβρύου. Σε έναν τέτοιο χάρτη η γενετική δραστηριότητα θα αναλυόταν σε συνιστώσες που αφορούν το ποια, πότε, πού και πόσο εκφράζονται τα γονίδια σε κυτταρικό επίπεδο. Η ύπαρξη τέτοιων χαρτών για οργανισμούς, όπως το ζεβρόψαρο, που πρωταγωνιστεί σε μελέτες που αφορούν σπονδυλώτα, θα διευκολύνει την ταυτόχρονη μελέτη των εκφράσεων και των μεταξύ τους αλληλεπιδράσεων.

Η παρούσα διπλωματική εστιάζεται στον σχεδιασμό και στην υλοποίηση μιας διαδικασίας επεξεργασίας εικόνας που θα είναι ικανή να συγκεντρώσει τις γονιδιακές εκφράσεις ενός συγκεκριμένου σταδίου ανάπτυξης του ζεβρόψαρου σε έναν τρισδιάστατο άτλαντο. Τα δεδομένα που πρέπει να επεξεργαστεί είναι τρισδιάστατες απεικονίσεις διαφορετικών εμβρύων, κάθε ένα από τα οποία έχει σημανθεί με φθορίουσες ουσίες με στόχο το DNA των πυρήνων και τις εκφράσεις δύο γονίδιων, εκ των οποίων η μία είναι κοινή για όλα τα έμβρυα. Η πληροφορία που αποκαλύπτει κάθε ουσία βρίσκεται σε διαφορετικό κανάλι της κάθε εικόνας. Συνεπώς η διαδικασία θα πρέπει να συνδυάζει κατάλληλα τις πληροφορίες των διαφορετικών καναλιών και να ευθυγραμμίζει τρισδιάστατες εικόνες.

Η υλοποιημένη διαδικασία δέχεται σειρές δεδομένων από δύο έμβρυα. Η μία απεικονίζει πλήρως ένα έμβρυο, το οποίο αποτελεί το έμβρυο αναφοράς και η άλλη μέρος του άλλου εμβρύου, το οποίο θα πρέπει να ευθυγραμμισθεί με το πρώτο. Η διαδικασία αποτελείται από διάφορα βήματα, τα οποία προσεξεργάζονται σε δεδομένα, μεταχειρίζοντας την εικόνα του μερικούς απεικονιζόμενου εμβρύου ώστε να αποκτήσει μια κατάλληλη αρχική θέση στον χώρο που ορίζεται από την εικόνα του ολόκληρου εμβρύου και τέλος ευθυγραμμίζουν την πρώτη εικόνα στην δεύτερη χρησιμοποιώντας ως αναφορά την γονιδιακή έκφραση που είναι κοινή για όλα τα σημασμένα έμβρυα. Οι παράμετροι που προκύπτουν από την ευθυγραμμίση χρησιμοποιούνται για την αντιστοίχιση των άλλων σημασμένων γονιδιακών εκφράσεων, δημιουργώντας έτσι ένα τρισδιάστατο χάρτη εκφρασέων. Η υλοποίηση βασίζεται στο Insight Segmentation and Registration Toolkit.

Η διαδικασία δοκιμάστηκε με δεδομένα από έξι έμβρυα του ίδιου σταδίου ανάπτυξης και συγκρίθηκαν οι επιδόσεις τεσσάρων μεθόδων αντιστοίχισης. Οπτική αξιολόγηση των αποτελεσμάτων ανέδειξε τον συνδυασμό της συνάρτησης του συντελεστή συσχέτισης με τον αλγόριθμο βελτιστοποίησης gradient descent ως την πιο κατάλληλη μέθοδο για την συγκεκριμένη εφαρμογή. Τα τελικά αποτελέσματα απέδειξαν ότι η διαδικασία επιτυγχάνει τον στόχο της.

Η διαδικασία που υλοποιήθηκε στη παρούσα διπλωματική είναι έτοιμη προς χρήση για την δημιουργία ενός χάρτη γονιδιακής έκφρασης του ζεβρόψαρου. Αντικείμενο μελλοντικής μελέτης αποτελεί η επικύρωση του χάρτη με ήδη γνωστές γονιδιακές αλληλεπιδράσεις και κατόπιν η χρήση του για την εύρεση νέων.

Λέξεις κλειδιά
Χάρτης γονιδιακής έκφρασης, πρώιμη ανάπτυξη ζεβρόψαρου, μέθοδοι αντιστοίχισης εικόνων, τεχνικές ευθυγράμμισης εικόνων, Insight Segmentation and Registration Toolkit (ITK)

xiii
Chapter 1

Introduction

This introductory chapter gives a general overview of the present thesis. Firstly the motives that drove this work are described. Then the objective of the thesis is analyzed and the framework, within which the thesis was realized, is presented. The chapter closes with the general layout of the thesis.

1.1. Motivation

In the animals’ life, complex forms alternate with simple ones. An individual develops from a single cell, a zygote, which bears no resemblance to the complex structure and pattern displayed in the adult form. The process of embryonic development, with its highly ordered increase in complexity accompanied by perfect reproducibility, is controlled by a subset of the animal’s genes. Animals have a large number of genes. The exact number is not known for any multicellular organism, nor is it known how many and which are required for the development of complexity, pattern and shape during embryogenesis. To identify these genes and to understand their functions is a major issue in biological research [1].

Animal embryos can be thought of as dynamic three dimensional arrays of cells expressing gene products in intricate spatial and temporal patterns that determine cellular differentiation and morphogenesis [2]. To fully understand animal transcription networks, automated methods that measure the spatial and temporal expression patterns at cellular resolution are required.

Microarrays provide huge amounts of quantitative information about gene expression profiling. However microarray analysis misses the complexity of gene expression patterns and the relationship between them, as their degree of spatial information is very limited. Nowadays, it remains an open challenge to connect the spatio-temporal pattern of those genes that play a role in the developmental events to the transcription network that regulates development. Recent approaches have addressed this issue in invertebrate animal models, such as fruit-flies (Drosophila melanogaster) [2, 3] and roundworms (Caenorhabditis elegans) [4].

To address the need for sufficient quantitative data about gene expression in time and space at cellular resolution, Fluorescence In Situ Hybridization techniques (FISH) in combination with laser scanning microscopy have been introduced. This way gene expression patterns can be visualized in space and time. However, FISH and microscopy limitations do not permit to reveal more than four or five fluorescent gene expression patterns at a time [5, 6]. The challenge is therefore to gather these data into a three dimensional gene expression atlas so that their simultaneous analysis can be possible.

Our focus is on the zebrafish embryo that provides an interesting model system for the vertebrate development [see Chapter 2]. Specifically the goal is to construct a
spatiotemporal gene expression atlas of the early zebrafish embryogenesis, overcoming the limitations in the detection of multiple gene expression patterns simultaneously. The construction of such an atlas can make possible the simultaneous comparison of a large number of gene expression patterns and open the way for modelling the underlying regulatory networks. The general strategy followed is depicted in Figure 1-1. Details of the data are given in Chapter 3.

The general strategy involves two key components, gene expression quantification and spatial registration. The gene expression quantification process aims to obtain quantitative measures about gene expression in each cell of individual embryos. The aim of the registration process is to define the spatial geometrical transformations that map the genes expression patterns on a common template. Both results, quantitative measurements and the spatial geometrical transformations, are combined to provide the quantitative atlas with the spatial pattern expression of several genes at a given stage in the early zebrafish embryogenesis. This procedure repeated over different developmental stages will result in tracking the temporal dynamics of gene expression.

**1.2. Objective**

The goal of this thesis is to design and implement a framework that provides the means to map expressions of different genes from different zebrafish embryos at the same developmental stage onto a common template. In other words, this framework is responsible for the registration process depicted in Figure 1-1.

Different zebrafish embryos were fluorescently stained to label nuclear DNA and the expression patterns of two genes, one of the two common to all labelled embryos, and then imaged by laser scanning microscopy. For each embryo, the information from each fluorescent label is captured in a stack of optical slices (image channel) and all its channels together comprise a dataset. The nature of these experimental data formulated the objectives of the framework, which this thesis aims to implement, as follows:
The framework should be able to combine the information given by the reference gene and the nuclei in each dataset.

The framework should provide a way to sufficiently overlap the combined volumetric data from two datasets so that their registration is initialized.

The framework should be able to deal with a three dimensional image registration problem, including different registration components so that the most appropriate method can be identified.

Finally, once the reference gene expression pattern of each dataset has been registered onto a common template, the framework should use the corresponding transformation parameters to map on the same template the gene expression patterns co-stained with the reference gene.

For the implementation the National Library of Medicine Insight Segmentation and Registration Toolkit (ITK) is used, which provides advanced algorithms for filtering, segmentation, and registration of volumetric data.

The framework’s performance and robustness will be evaluated with data from a specific developmental stage of the zebrafish embryogenesis (shield, 6.00 hours post fertilization).

### 1.3. Organization

This thesis is part of the MORPHONET: *Reconstructing Genetic and Cellular Dynamics in Early Zebrafish Embryogenesis*, which is a Spain-France integrated research project [HF2007-0074]. Its multidisciplinary character requires the interaction between different groups, each one specializing in a specific field. The groups in charge of the project are:

- **DEPSN** (Development Evolution Plasticity of the Nervous System), France. This centre is in charge of the biological procedures and image acquisition.
- **BIT** (Biomedical Image Technologies), ETSIT, UPM, Madrid, Spain. This research group is specialized in biomedical image processing.
- **CREA** (Centre de Recherche en Épistémologie Appliquée, École Polytechnique), France. It is responsible for the reconstruction of the genetic network and its modelling.

![Diagram](image-url)

*Figure 1-2 The groups involved in the Morphonet project, their major tasks and interactions*
1.4. **Structure**

The present thesis is organized in seven chapters, as follows:

1. Chapter 1 (the present one) gives a general overview of the thesis. Firstly the motives that drove this work and its objectives are presented. Then the chapter continues with an overview of the international context, in which it has been developed, and closes with this general layout of the thesis.

2. Chapter 2 provides the biological background of the thesis. More specifically the advantages of the zebrafish, as a research model, are highlighted, an overview of its development is presented and the role of the goosecoid gene during its development is discussed.

3. In Chapter 3 there is a description of the techniques and the instrumentation used to acquire our experimental data. In particular, the technique described is the Fluorescence In Situ Hybridization (FISH) and the instrumentation concerns the laser scanning microscopy.

4. Chapter 4 provides the necessary background concerning medical image registration with emphasis on the rigid, intensity-based registration methods and their components.

5. Chapter 5 gives an extent presentation of the implemented framework and of the steps it is composed of. Firstly, the general strategy upon which the framework is based is presented. Then a brief overview of the overall process is given and finally each step of the framework is analyzed, in terms of its concept and implementation.

6. In Chapter 6 the results obtained by processing the experimental data according to the implemented framework are presented. Furthermore, a comparison of the different approaches allowed within the framework is given, deducing the one that yields the best results.

7. Chapter 7 draws the final conclusions of the evaluation and discusses future goals.

Additionally, in the Appendices a glossary is provided for the biological terms used in Chapter 2. Moreover there is a short overview of the basic concepts in image processing and a presentation of the tools used in this thesis. Finally, a user’s manual is given to run the algorithms implemented within the presented framework as well as a full list of the parameters’ values that were used to obtain the result presented in Chapter 6.
Chapter 2

Biological Background

In this chapter a small introduction is given concerning the biological context involved in the embryonic development of zebrafish. Firstly the advantages of zebrafish as a model organism are discussed (section 2.1), an overview of its developmental stages is given (section 2.2) and the role of a specific gene, the goosecoid, throughout its embryonic development is presented (section 2.3).

A glossary for the biological terms mentioned in the description of the zebrafish embryogenesis is provided in Appendix A.

2.1. Zebrafish - A vertebrate model organism

As a model organism for genetical studies of development, Drosophila melanogaster (a.k.a. fruit fly) has a long tradition and is by far the most well established organism available. However, flies are different from the vertebrates in many aspects. In order to identify genes affecting the vertebrate development it is necessary to study a vertebrate organism. Therefore, several laboratories have established methods to use the zebrafish as a model organism to analyze the genetic control of embryonic development in a vertebrate.

The zebrafish, Danio rerio, was selected as a model with potential for genetical research by George Streisinger (1981). Since then a number of useful methods have been developed for the breeding and genetic analysis, as well as for the investigation of embryonic development of the zebrafish.

The zebrafish is a small tropical fish that has many characteristics that are ideal for studying gene function during embryonic development, such as fast development, short generation time (3-4 months), large number of offspring, small size (4-5 cm in length), ease of culture, amenability to genetic manipulation and optical transparency during the embryonic stages [1].

Large numbers of mutations that disrupt embryonic development have been isolated in the zebrafish, many of which may serve as models for human diseases. These mutants will help to understand the genetic network controlling the development of vertebrates, including humans.
Zebrafish have 25 chromosomes and their genome consists of about $1.5 \times 10^9$ basepairs, comparable to mammalian genome sizes of about $3 \times 10^9$ basepairs [7]. An online database of zebrafish genetic, genomic and developmental information, the Zebrafish Information Network (ZFIN), has been established [8]. To facilitate use of the zebrafish as a model of human biology, ZFIN links these data to corresponding information in other model organism (e.g., mouse) and human disease databases. Gene product, gene expression and phenotype data are annotated with terms from biomedical ontologies.

2.2. Overview of the zebrafish development

Embryonic development in zebrafish (from fertilization to hatching) extends for 48 h. Within 10 hours post fertilization (hpf), (which is equal to about 21% of the total embryonic period), the basic vertebrate body plan of the zebrafish embryo is established, the anterior-posterior and dorsal-ventral axes are formed and the three germ layers (ectoderm, mesoderm and endoderm) are developed. The segmentation and pharyngula stages are completed in about 14 and 24 h, respectively [9].

According to Kimmel et al. [10] seven broad periods of the embryogenesis are defined: the zygote, cleavage, blastula, gastrula, segmentation, pharyngula and hatching periods, which highlight the changing spectrum of major developmental processes that occur during the first 3 days after fertilization. The stages, which subdivide the above mentioned periods, and their names are based on morphological features of the embryo.

The following brief description of zebrafish development is derived predominantly from Kimmel et al.

**ZYGOTE PERIOD (0 - ¾ h)**

The newly fertilized egg is in the zygote period until the first cleavage occurs (Figure 2-2), about 40 minutes after fertilization. Fertilization activates cytoplasmic movements. Nonyolk cytoplasm begins to stream toward the animal pole, segregating the blastodisc from the clearer yolk granule-rich vegetal cytoplasm.

![Figure 2-2 The zygote period. Extracted from [10].](image)

**CLEAVAGE PERIOD (¾ - 2 ¾ h)**

After the first cleavage the cells, or blastomeres, divide at about 15-minute intervals, resulting in a three-dimensional mass of cells. The cytoplasmic divisions are meroblastic. During cleavage the volume of the embryo remains essentially constant, so that the divisions produce a larger number of smaller cells. The six cleavages that comprise this period frequently occur at regular orientations. The cells in the blastoderm form the embryo
proper, whereas the yolk is an extraembryonic structure. The blastomeres at the margin of the blastoderm have a unique fate. They lie against the yolk and remain cytoplasmically connected to it throughout cleavage.

**Figure 2-3** The cleavage period. The stages are represented by schematic drawing of mid-sagittal sections of the embryos. Extracted from [10].

**BLASTULA PERIOD (2 ¼ to 5 ¼ h)**

This is the period when the blastodisc begins to look ball-like, at the 128-cell stage (8th zygotic cell cycle) until the time of onset of gastrulation (14th cycle). Important processes occur during this blastula period; the embryo enters midblastula transition (MBT), the yolk syncytial layer (YSL) forms, and epiboly begins. Epiboly continues during the gastrulation period.

The MBT begins at the 512-cell stage and marks the time when zygotic transcription begins (although a few genes may be transcribed prior to the MBT), so that the zygotic genome begins to govern embryonic development.

The blastomeres at the margin of the blastoderm during the blastula stage release their cytoplasm and nuclei together into the immediately adjoining cytoplasm of the yolk cell, producing the yolk syncytial layer (YSL), a thin, multinucleate structure at the interface of the blastoderm and the yolk. Beginning in the late blastula stage, the YSL and the blastodisc spread over the yolk cell, in a process known as epiboly. In the process of epiboly, cells intercalate radially, thereby thinning the blastoderm and spreading over the yolk. The extent of yolk cell coverage (measured as “percent epiboly”) provides a convenient way to determine an embryo’s developmental stage.

**Figure 2-4** The blastula period. Extracted from [10].
**GASTRULA PERIOD (5 ¼ - 10 h)**

Within the first 5 hours post fertilization (hpf), cleavage and blastula are completed, and the process of gastrulation takes another 5h. Gastrulation is a central process in the early development during which the three germ layers (ectoderm, mesoderm and endoderm) are formed and involves the morphogenetic cell movements of involution, convergence, and extension, producing the embryonic axis.

The onset of gastrulation occurs at 50% epiboly. At this time a thickened marginal region, termed the germ ring, appears around the blastoderm rim. The germ ring is formed by a folding of the blastoderm back upon itself (involution). Hence, within the germ ring there are two germ layers. The upper layer (the epiblast) continues to feed cells into the lower (the hypoblast) throughout gastrulation. The cells remaining in the epiblast when gastrulation ends correspond to the ectoderm and will give rise to such tissues as epidermis, the central nervous system, neural crest, and sensory placodes. The hypoblast gives rise to both the mesoderm and endoderm, although it is unclear how this layer subdivides into endoderm and mesoderm.

By 6 hpf convergence and extension movements have begun, resulting in the dorsal accumulation of cells at one position along the germ ring, the so-called embryonic shield, moving from lateral and ventral regions of the blastoderm (convergence). Convergence of cells toward the dorsal side of the embryo marks the first clearly apparent break in radial symmetry. Concomitantly, converging cells intercalate with dorsal blastomeres, spreading
them along the animal-vegetal axis, leading to a lengthening of the anterior-posterior axis (extension).

Hence, when the shield forms one can reliably distinguish the orientation of the embryo's eventual dorsal - ventral axis. Cells involuting at the shield form the so called axial hypoblast. Expression of the gene goosecoid (gsc) is a reliable marker of where the shield will form, and appears to label the earliest cells to involute in the axial hypoblast.

Eventually, at the end of gastrulation, the yolk cell is completely surrounded by the spreading YSL and blastodisc. The blastodisc becomes considerably thinner in the process.

\[\text{Figure 2-7 Gastrulation movements. (a) Dome stage. Cells intercalate radially, contributing to epiboly. (b) Shield stage. Cells at the margin internalize and migrate toward the animal pole. Cells converge dorsally, with lateral mesodermal cells starting convergence at later stages than cells closer to the shield (c) 90\% epiboly stage. Epiboly, internalization, convergence and extension continue. Extracted from} \[12].\]

**SEGMENTATION PERIOD (10 - 24 h)**

A variety of morphogenetic movements now occur, the somites develop, the rudiments of the primary organs become visible, the tail bud becomes more prominent and the embryo elongates. The first cells differentiate morphologically and the first body movements appear.

\[\text{Figure 2-8 The segmentation period. Extracted from} \[10].\]

**PHARYNGULA PERIOD (24 - 48 h)**

The pharyngeal arches develop rapidly during this second day. The head straightens out and the fins begin to form. Pigment cells differentiate, the circulatory system forms and tactile sensitivity appears.

**HATCHING PERIOD (48-72 h)**

During the hatching period the embryo continues to grow at about the same rate as earlier. Morphogenesis of many of the organ rudiments is now rather complete and slows down considerably, with some notable exceptions including the gut and its associated organs.
Chapter 2 – Biological Background

**EARLY LARVAL PERIOD**

By day 3 the hatched larva has completed most of its morphogenesis, and it continues to grow rapidly. Prominent changes during the next day include the inflation of the swim bladder and the continued anterior-dorsal protrusion of the mouth. The gut tube drops more ventrally and the yolk extension nearly empties. Whereas during the hatching period the embryo is usually at rest, the early larva gradually begins to swim about actively, and moves its jaws, opercular flaps, pectoral fins, and eyes. These developments produce swift escape responses and herald respiration, the seeking of prey, and feeding.

2.3. The goosecoid gene in zebrafish development

Several genes have been suggested to play a role in gastrulation and in the establishment of the vertebrate body axis in zebrafish. Among these the goosecoid has been demonstrated in functional studies to be particularly important and to have the properties to be considered as a dorsal organizer [12].

The first zygotic gsc transcripts can be detected between 3 and 4 hours of development, i.e. at blastula stages prior to the onset of epiboly. Maximal amounts of transcripts are present during gastrulation, between 50% and 100% epiboly stages. During early somitogenesis, the level of gsc transcript decreases markedly. No message can be detected between the 5-somite stage and 24 hours of development (end of somitogenesis). However, starting from 36 hours there exists a second, independent, phase of gsc expression which lasts at least until day five.

At the blastula stage there is a clear gradient of gsc transcripts along the dorsal-ventral axis with the maximum at one side of the embryo. This gradient persists only for a short period of time, after which expression of gsc is restricted exclusively to the dorsal-most cells of the future germ ring. With the onset of involution and the concomitant formation of the embryonic shield, the localization of these cells is at the dorsal side of the early gastrula and the gsc-expressing cells are among the first to involute under the margin. As gastrulation proceeds, gsc-expressing cells migrate towards the animal pole of the embryo (75% epiboly) [13].

The expression of the goosecoid gene plays an important role in the construction of the gene expression atlas, as it was selected in the data acquisition to serve as the reference position with respect to which all the other genes can be mapped.
Chapter 3

Data Acquisition

Fluorescence In Situ Hybridization (FISH) in combination with laser scanning microscopy provide an excellent tool to generate three-dimensional images of gene expression patterns. A brief overview of the FISH technique (section 3.1), the underlying physical principles (section 3.2), the imaging instrumentation (section 3.3) and a presentation of our data, acquired by the combination of the above techniques, (section 3.4) follow in the next sections.

3.1. Fluorescence In Situ Hybridization (FISH)

A valuable tool for studying spatial characteristics of patterns of gene expression is In Situ Hybridization (ISH), the power of which can be extended by the use of fluorescent probes (FISH). In principle the FISH technique is quite straightforward. Selected probes (sequences of DNA or RNA) are labelled either directly with a reporter molecule (fluorophore) or indirectly with a hapten (e.g. digoxigenin). Then the labelled probes bind to the target via hybridization. If the probe has been labelled indirectly, an extra step is required for the visualization of the non-fluorescent hapten using an enzymatic or immunological detection system. Whereas FISH is faster with directly labelled probes, indirect labelling offers the advantage of signal amplification by using several layers of antibodies (e.g. Tyramide Signal Amplification), and it might therefore produce a signal that is brighter compared with background levels.

Figure 3-1 Principles of FISH. (a) The basic elements: a probe and a target. (b) Either indirect (left panel) or direct labelling (middle panel) can be used. (c) Both labelled probe and target are denatured. (d) Hybridization. (e) Detection of the non-fluorescent hapten in the case of indirect labelling. Adapted from [14].
Chapter 3 – Data Acquisition

The specimen is then screened for the reporter molecule by fluorescence microscopy, revealing the spatial location of the target. Therefore staining the transcripts of a gene with a labelled probe via FISH reveals the expression of the gene.

3.2. Fluorescence - Physical principles

Fluorescence is a phenomenon by which a molecule, upon illumination at a specific wavelength, reemits light at another (typically longer) wavelength. A molecule that has the ability to fluoresce is called a fluorophore or fluorochrome. A molecule can exist in a variety of energetic states, which, for the most part, are determined by the configuration of its electrons and the vibrational agitation of its atomic nuclei. If a photon with sufficient energy is absorbed by a fluorophore, the latter moves from its ground state to an excited electronic state (Figure 3-2). Fluorescence occurs when the excited molecule returns to the ground state by releasing energy through emission of a photon. Because some of the energy gained during excitation is converted to heat, the emitted photon has a lower energy than the absorbed one. This explains the difference in wavelength mentioned earlier, which is also known as the Stokes shift.

A related phenomenon is that of multiphoton excitation. A fluorophore can also be excited by the simultaneous absorption of two or more photons, given that the combined energy of the photons corresponds to the energy required for single-photon excitation [15].

The phenomenon of two-photon excitation arises from the simultaneous absorption of two photons in a single quantitized event. Since the energy of a photon is inversely proportional to its wavelength, the two absorbed photons must have a wavelength about twice that required for one-photon excitation. For example, a fluorophore that normally absorbs ultraviolet light (~350nm) can also be excited by two photons of near-infrared light (~700nm) if both reach the fluorophore at the same time (Figure 3-2). In this case, "the same time" means within an interval of about $10^{-18}$ seconds.

![Two-Photon Jablonski Energy Diagram](image)

Figure 3-2 Jablonski diagram representing the energy level transitions involved in the fluorescence of a single and a two-photon excitation. Thick lines represent electronic energy levels, thin ones are associated vibrational energy levels. (a) Upon absorption of a photon at a specific wavelength (purple), the molecule moves from the ground state $S_0$ to the excited state $S_1$. Vibrational energies are immediately converted into heat in a process called vibrational relaxation. When the molecule returns to the ground state, the remaining energy is released via emission of a new photon at a longer wavelength (blue). (b) In the case of two-photon excitation, the excitation wavelength (red) is longer than the emission wavelength (blue). Extracted from [16].
Because two-photon excitation depends on simultaneous absorption, the resulting fluorescence emission varies with the square of the excitation intensity. This quadratic relationship between excitation and emission gives rise to many of the significant advantages associated with two-photon excitation microscopy. In order to produce a significant number of two-photon absorption events (in which both photons interact with the fluorophore at the same time), the photon density must be approximately one million times that required to generate the same number of one-photon absorptions. The consequence is that extremely high laser power is required to generate significant two-photon-excited fluorescence. Thus, fluorescent emission following two-photon excitation is exactly the same as emission generated in normal one-photon excitation [17].

3.3. Microscopy

We now turn our attention to the instrumentation for fluorescence imaging. **Laser Scanning Microscopy (LSM)** is currently the most promising method for imaging embryos with single cell resolution. LSM generates images by raster scanning a laser over a specimen usually through the use of fluorescence. An important advantage of LSM over traditional light microscopy is its ability to eliminate out of focus light so that only a single focal plane in a thick specimen is imaged to produce what is called an ‘optical section’. Stacks of optical sections called Z stacks can be collected across a range of focal planes. The series of two-dimensional (2-D) images (the ‘optical sections’) in the stack can be reconstructed to generate a 3-D image that captures the depth of the specimen [18].

![Figure 3-3 Optical sections and reconstruction of a three dimensional image. Extracted from [19]](image)

LSM eliminates out of focus light from fluorescent specimens using one of two principles. In confocal LSM (single photon excitation), a pinhole aperture located in an optically conjugated position to the focal plane blocks out of focus light. In two-photon LSM, fluorescent markers are excited only at the focal plane through the use of two-photon excitation, which is generated by focusing a single pulsed laser through the microscope optics. This is illustrated in Figure 3-4.

As the laser beam is focused, the photons become more crowded (their spatial density increases), and the probability of two of them interacting simultaneously with a single
fluorophore increases. The laser focal point is the only location along the optical path where the photons are crowded enough to generate significant occurrence of two-photon excitation. The probability of two-photon excitation drops off by the fourth power of the distance from the focal plane so there is very little excitation outside the focal plane.

Although the image resolution obtained with two-photon excitation is not better than that achieved in a well-aligned confocal microscope, the first technique has significant advantages over the second due to the narrow localization of two-photon excitation to the illumination focal point.

Firstly, in a confocal microscope, although fluorescence is excited throughout the specimen illuminated volume, only signal originating in the focal plane passes through the confocal pinhole, allowing background-free data to be collected. By contrast, two-photon excitation only generates fluorescence at the focal plane, and since no background fluorescence is produced, a pinhole is not required (Figure 3-5).

Secondly, in a confocal microscope, excitation photons are absorbed by any fluorophores that are encountered along the excitation light path. The consequence is that fewer photons reach the focal plane, thus decreasing the signal generated. This effect becomes more pronounced if the specimen contains fluorophores throughout. By contrast, in two-photon excitation, since there is no absorption in out-of-focus specimen areas, more of the excitation light penetrates through the specimen to the plane of focus. Figure 3-6 illustrates the dramatic difference between confocal and two-photon excitation, in which the intensity is relatively constant with penetration depth.

Thirdly, photobleaching of the fluorophore and phototoxicity to the specimen are reduced by the use of two-photon excitation, in comparison to confocal microscopy, as the excitation intensity is effectively concentrated at the focal spot [17].
Chapter 3 – Data Acquisition

Figure 3-5 Schematic of a confocal and a two photon microscope, showing their main components.

In a confocal microscope, the excitation light (blue) is focused into the specimen and the fluorescence (green) from that focal spot is captured by the objective lens, passes cleanly through the pinhole, and reaches the detector. Some of fluorescence light can be scattered as it passes back through the specimen. These losses reduce the detected fluorescence signal. As the excitation light passes through the specimen, it may be absorbed or scattered before it reaches the focus. If it is absorbed, it can generate fluorescence. A small portion of this out-of-focus fluorescence can be scattered into the pinhole and be detected creating a background fog and reducing the image contrast. Likewise, the scattered excitation can generate fluorescence, and this fluorescence can also contribute to the background fog.

In the two-photon excitation method, the excitation photons (red) can be scattered as in the confocal system. However, the probability of two photons being scattered simultaneously to the same specimen location is essentially zero, and consequently, the background fog is not generated. In addition, a greater proportion of the excitation light reaches the focal plane due to the reduced out-of-focus absorption and the decreased scattering of the longer-wavelength two-photon excitation light. The generated fluorescence (green), even if scattered, has an increased likelihood of being detected by the photomultiplier tube because no pinhole is present to block it. This insensitivity to scattering effects and absence of out-of-focus absorption allow for the preservation of the full image contrast from considerable depth within specimens. Extracted from [16].

Figure 3-6 Single and two photon scanning profiles. The excitation intensity is relatively constant with penetration depth in the two-photon excitation, whereas in the one-photon the excitation intensity is decreasing with depth. Adapted from [17].
3.4. Experimental Data

As already mentioned, the general goal of the MORPHONET project, to the achievement of which the present thesis plays an important part, is the construction of a 3-D gene expression atlas of the early zebrafish embryogenesis. Specifically the present thesis focuses on the spatial alignment of different gene expression patterns into a common template at a given developmental stage by using volumetric data from different embryos. This is an image registration problem and as such, in order to be solved, common features in the data must be employed.

These features are the embryos’ geometrical shape, which is given by staining the nucleus of every cell, and the expression of the goosecoid gene (gsc), which is given by staining its transcripts. The location of the expression of the gsc serves also as reference location, with respect to which other gene expressions can be mapped within the same template.

In order to reveal two gene expression patterns, the zebrafish embryos were treated for double fluorescent in situ hybridization (FISH) procedures. In addition, the embryos’ nuclei were counter-stained to allow proper detection of every single cell. The nuclei were stained with DAPI (which appears blue/cyan) to reveal the geometrical structure of each embryo in space. The transcripts of the gsc gene were labelled with red fluorescent cyanine dye Cy5, while the transcripts of another gene, whose expression will be spatially located with respect to the location of the gsc’s expression, were labelled with the green fluorescent fluorescein dye FITC. The embryos were then imaged by two-photon laser scanning microscopy resulting in high-resolution 3-D digital images throughout different time stages. Being constrained by the cellular resolution, these images do not comprise the whole embryo but only cover views restricted to the stained region (partial views).

The partial views contain information in three different channels: The first one holds the embryo’s 3-D nuclei images at the moment of acquisition. The second one always contains the gsc’s expression pattern, as it is used to guide the spatial registration, along with the nuclei, and serves as reference to other gene expression patterns. The third one contains the expression pattern of one out of several other genes. An example is given in Figure 3-7.

![Figure 3-7](image-url)

Figure 3-7 Transverse slice of a partial view of a zebrafish embryo at single cell resolution in the shield stage. Information is contained in three channels: (a) goosecoid gene expression channel, (b) nuclei channel and (c) snail gene expression channel.
In order to register the partial views of a developmental stage into a common template, where simultaneous comparison of a large number of gene products can be achieved, a view that depicts a whole embryo at the same stage and also contains the gsc’s expression, is necessary. To this complete embryo view the partial views are registered, employing the reference gene’s expression pattern along with the embryo’s geometrical shape. The complete template contains information in two channels: The first one holds the embryo’s nuclei and the second the gsc’s expression pattern. Figure 3-8 is an example of a complete template.

![Transverse slice of a whole zebrafish embryo view in the shield stage. Information is contained in two channels: (a) goosecoid gene expression channel and (b) nuclei channel.](image)

Image acquisition was performed with a Leica TCS SP5 laser scanning microscope equipped with a Leica objective HCX APO L U-V-I 20x/0.5NA W for the partial views and with a 20x/1NA W objective for the whole embryo view. All views were acquired in the same direction in the ‘z’ axis, that is, from the top of the embryo towards the yolk.
Chapter 4

Medical Image Registration

The focus of this chapter is on medical image registration methods. Their application can be extended to biomedical images, such as in our case. From the existing methods the rigid, intensity based method is analyzed, since this one was selected for the implemented framework.

This chapter gives the definition of registration (section 4.1) and reviews a classification of the existing registration methods (section 4.2). According to this classification, a particular method, the intensity based registration, is further analyzed (section 4.3), since this is the one that the implemented framework is based on. Its analysis includes all the different elements, into which the intensity based method can be decomposed (sections 4.4 – 4.8).

4.1. Definitions

The term ‘medical image’ covers a wide variety of types of images, with very different underlying physical principles and very different applications. The sort of images used in healthcare and medical research vary from microscopic images to video images used for remote consultation, and from images of the eye taken with a fundus camera to whole body radioisotope images [20].

Registration is the determination of a geometrical transformation that aligns points in one view of an object with corresponding points in another view of that object or another object. The term “view” includes a three dimensional image, a two-dimensional image, or the physical arrangement of an object in space. In all cases, we are concerned with digital images stored as discrete arrays of intensity values.

From an operational view, the inputs of registration are the two views to be registered; the output is a geometrical transformation, which is merely a mathematical mapping from points in one view to points in the second. To the extent that corresponding points are mapped together, the registration is successful. The determination of the correspondence is a problem specific to the domain of objects being imaged. Registration is merely the determination of that relationship. The goal of registration is thus simply to produce as output a geometrical transformation that aligns corresponding points [21].

4.2. Classification of registration methods

There are many image registration methods and a number of surveys exist concerning the classification of medical image registration in particular [22-25]. A categorization model based on nine criteria has been suggested by Maintz and Viergever [25], which by Fitzpartick et al. [21] have been condensed into eight categories: image dimensionality, registration
basis, geometrical transformation, degree of interaction, optimization procedure, modalities, subject, and object.

1. “Image dimensionality” refers to the number of geometrical dimensions of the image spaces involved, which in medical applications are typically three-dimensional but can also be two-dimensional sometimes.

2. The “registration basis” is the aspect of the two views used to affect the registration. For example, the registration might be based on a given set of point pairs that are known to correspond (point based) or the basis might be a set of corresponding surface pairs (surface based). Alternatively, there may be no known correspondences as input. In that case, intensity patterns in the two views will be matched (intensity based).

3. The category, “geometrical transformation”, is a combination of two of the categories mentioned in [25], the “nature of transformation” and the “domain of transformation”. It refers to the mathematical form of the geometrical mapping used to align points in one space with those in the other. The transformations are discussed in section 4.5.

4. “Degree of interaction” refers to the control exerted by a human operator over the registration algorithm. The interaction may consist simply of the initialization of certain parameters or it may involve adjustments throughout the registration process in response to visual assessment of the alignment or to other measures of intermediate registration success. The ideal situation is obviously the fully automatic algorithm, which requires no interaction.

5. “Optimization procedure” refers to the standard approach in algorithmic registration in which the quality of the registration is estimated continually during the registration procedure in terms of some function of the images and the mapping between them. The optimization procedure is the method, possibly including some degree of interaction, by which that function is maximized or minimized through an iterative search.

6. “Modalities” refers to the means by which the images to be registered are acquired. Registration between the same modalities is called “intramodal” or “monomodal” registration; registration between differing modalities is called “intermodal” or “multimodal” registration.

7. “Subject” refers to patient involvement and comprises three subcategories: intrapatient, interpatient, and atlas, the latter category comprising registrations between patients and atlases, which are themselves typically derived from patient images.

8. “Object” refers to the particular region of anatomy to be registered.

To build a classification hierarchy one of the above criteria must be placed at the top level. Therefore based on the registration basis we can firstly classify our implemented framework as intensity based. Then the geometrical transformation criterion follows classifying it as global rigid. Our framework works with three dimensional images taken from the same modality (laser scanning microscopy) and depicting different subjects (different embryos). It involves the adjustment of certain parameters from the user. The transformation parameters desired are sought by iterative optimization algorithms over a similarity function
defined on the parameter space. Since we are dealing with zebrafish embryos no anatomy can be defined.

4.3. Intensity-Based Registration

The term intensity refers to the scalar values in image pixels or voxels. The physical meaning of the pixel or voxel value depends on the modalities being registered. Intensity-based registration involves calculating a transformation between two images using the pixel or voxel values alone. Since we are dealing with 3-D views, the term voxels will be used from now and on.

The registration transformation is determined by iteratively optimizing some “similarity measure” calculated from all voxel values. Many intensity-based registration algorithms use only a subset of voxels and require some sort of preprocessing. For example, the algorithm may run faster if only a subset of voxels is used. It is normal in these circumstances to blur the images before sampling to avoid aliasing in the subsampled images, and the amount of blurring used may be application dependent. Alternatively, an algorithm may work reliably only if the similarity measure is calculated from the voxels in a defined region of interest in the image, rather than all voxels. In this case, some sort of pre segmentation of the images is required. In some other intensity-based algorithms, the similarity measures work on derived image parameters such as image gradients or principal axes, rather than the original voxel values.

In the following sections the similarity measures used in our implemented framework are reviewed (section 4.4), the geometrical transformations, with emphasis on the rigid transformation, are described (section 4.5), the issue of the direction of the transformation mapping and interpolation are discussed (sections 4.6 and 4.7 respectively) and finally, the algorithms used to iteratively determine the transformation by optimizing the similarity measure are analyzed (section 4.8).

4.4. Similarity measures

A similarity measure is a function that indicates how well the two images match each other. A large number of similarity measures exist. Since the list is long, only the ones implemented in our framework are presented here, namely the correlation coefficient and the mutual information measures.

Correlation Coefficient

If the intensities in images \( I_1 \) and \( I_2 \) being registered are linearly related, then the correlation coefficient \( \text{CC} \) can be shown to be the ideal similarity measure.

\[
\text{CC} = \frac{\sum_{i=1}^{N} (I_{1i} - \bar{I}_1)(I_{2i} - \bar{I}_2)}{\sqrt{\sum_{i=1}^{N} (I_{1i} - \bar{I}_1)^2 \sum_{i=1}^{N} (I_{2i} - \bar{I}_2)^2}}
\]  

(4-1)
where $I_{1i}$ is the i-th voxel value and $\overline{I}_1$ the mean voxel value of image $I_1$, $I_{2i}$ is the i-th voxel value and $\overline{I}_2$ the mean voxel value of image $I_2$ and $N$ is the number of voxels considered.

The CC can be thought of as a normalized version of the widely used cross correlation measure. The CC as it is written in equation 4.1 has to be maximized. Multiplying equation 4.1 by minus one (-1) the optimization problem is reversed to minimization. Then the optimal value of the metric is minus one. Misalignment between the images results in small measure values (the values’ range is [-1, 1]).

**Mutual Information**

Before analyzing the mutual information measure, a reference to the information theory techniques is appropriate. According to these techniques, we can think of registration as trying to maximize the amount of information shared between two images, or trying to minimize the amount of information present in the combined image. When the two images are perfectly aligned all the corresponding structures will overlap, eliminating any duplicate elements that result from misalignment. Thus, registration works based upon a measure of information.

The most commonly used measure of information in signal and image processing is the Shannon–Wiener entropy measure $H$,

$$H = -\sum_i p_i \log p_i$$  \hspace{1cm} (4-2)

$H$ is the average information supplied by a set of $n$ symbols whose probabilities are given by $p_1, p_2, p_3, ..., p_n$. $H$ reaches its maximum value when all probabilities are equal, so any change in the data that tends to equalize the probabilities of the different symbols increases the entropy.

The distribution of the probabilities $p_i$ can be represented as a histogram. Histograms indicate the probability for a voxel to have a certain intensity, i.e. the number of voxels having a certain intensity divided by the total number of voxels. Similarly, joint probability distributions are computed by counting the occurrences of pairs of intensities $(a, b)$, i.e. the number of voxel positions $(i, j, k)$ where $I_1(i, j, k) = a$ and $I_2(i, j, k) = b$, which generates a two-dimensional histogram, or joint histogram.

Joint entropy measures the amount of information we have in the two images combined. The joint entropy $H(I_1, I_2)$ is given by

$$H (I_1, I_2) = \sum_a \sum_b p_{I_1I_2}(a, b) \log p_{I_1I_2}(a, b)$$  \hspace{1cm} (4-3)

with $p_{I_1I_2}$ being the probability distribution function which can be visualized as a joint histogram as described above. The number of elements in the probability distribution function can either be determined by the range of intensity values in the two images, or from a partitioning of the intensity space into ‘bins’. In the above equation $a$ and $b$ either represent the original image intensities or the selected intensity bins.
If $I_1$ and $I_2$ are totally unrelated, then the joint entropy will be the sum of the entropies of the individual images. The more similar (i.e. less independent) the images are, the lower the joint entropy compared with the sum of the individual entropies

$$H(I_1, I_2) \leq H(I_1) + H(I_2) = H(I_1)$$

$$= \sum_{\alpha} p_{I_1}(\alpha) \log p_{I_1}(\alpha) + \sum_{\alpha} p_{I_1}(\alpha) \log p_{I_1}(\alpha)$$

(4-4)

The idea behind mutual information is now to combine the calculation of the individual and the joint entropies.

Mutual information (MI) measures how much information one random variable (image intensity in one image) tells about another random variable (image intensity in the other image). The major advantage of using MI is that the actual form of the dependency does not have to be specified. Therefore, complex mapping between two images can be modelled. For two images $I_1$ and $I_2$, mutual information $MI(I_1, I_2)$ can be defined as

$$MI(I_1, I_2) = H(I_1) + H(I_2) - H(I_1, I_2)$$

(4-5)

In the case where $I_1$ and $I_2$ are completely unrelated, MI reaches its minimum value of 0. For identical images, $H(I_1) = H(I_2)$, so $MI(I_1, I_2) = H(I_2) \leq \log n$, where $n$ is the number of histogram bins. More details about Mutual Information and its theoretical background can be found in [26].

### 4.5. Geometrical Transformations

Each view that is involved in a registration will be referred to a coordinate system, which defines a space for that view. Our definition of registration is based on geometrical transformations, which are mappings of points from the space $X$ of one view to the space $Y$ of a second view. The transformation $T$ applied to a point in $X$ represented by the column vector $x$ produces a transformed point $x'$.

$$x' = T(x)$$

(4-6)

If the point $y$ in $Y$ corresponds to $x$, then a successful registration will make $x'$ equal, or approximately equal, to $y$.

The set of all possible $T$ may be partitioned into rigid and non-rigid transformations with the latter transformations further divided into many subsets. It should be noted here that in this partition as non-rigid transformation we consider those that do not belong to the rigid in the strict mathematical sense. This notation is important since in literature non-rigid transformation is frequently synonymous to curved (or elastic) transformation.

An image coordinate transformation is called rigid, when only translations and rotations are allowed. If the transformation maps parallel lines onto parallel lines it is called affine. If it maps lines onto lines, it is called projective. Finally, if it maps lines onto curves, it is called curved or elastic. Each type of transformation contains as special cases the ones described before it, e.g., the rigid transformation is a special kind of affine transformation. A composition of more than one transformation can be categorized as a single transformation.
of the most complex type in the composition, e.g., a composition of a projective and an affine transformation is a projective transformation, and a composition of rigid transformations is again a rigid transformation.

According to the domain of the transformation, a transformation is called *global* if it applies to the entire image, and *local* if subsections of the image each have their own transformations defined. Figure 4-1 shows examples of all transformation types mentioned.

Figure 4-1 Classification of image registration methods by nature and domain of the transformation. Extracted from [25].

A rigid or affine 3-D transformation can be represented by means of *homogeneous coordinates* by using a single constant 4x4 matrix \( A' \), where the three dimensional translation vector \( t = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \) and the 3x3 rotation matrix \( A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \) are folded into. To accomplish this augmented vectors \( u \) and \( u' \) are used for which \( u_i = x_i \) and \( x'_i = u'_i \) for \( i = 1,2,3 \) and \( u_4 = u'_4 = 1 \).

\[
\begin{align*}
  u' &= \begin{pmatrix} u'_1 \\ u'_2 \\ u'_3 \\ 1 \end{pmatrix} = A'u = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ a_{31} & a_{32} & a_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ 1 \end{pmatrix} \\
  & \quad (4-7)
\end{align*}
\]

In the affine transformation no restrictions on the elements \( a_{ij} \) exist since scaling and shearing are included, whereas the restrictions for the rigid transformation are presented in section 4.5.1.

In the projective case, the transformations have the form

\[
\begin{align*}
  u' &= \begin{pmatrix} u'_1 \\ u'_2 \\ u'_3 \\ u'_4 \end{pmatrix} = A'u = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ a_{31} & a_{32} & a_{33} & t_3 \\ p_1 & p_2 & p_3 & a \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ 1 \end{pmatrix} \\
  & \quad (4-8)
\end{align*}
\]

Here \( u'_4 \) is no longer equal to 1 and \( x'_i = u'_i/u'_4 \) for \( i = 1,2,3 \).

Curved transformations cannot in general be represented using constant matrices. The next section is focused on rigid transformations, as the implemented framework was based on such transformations. Details on the non-rigid transformations can be found in [27], [28] and [29].
4.5.1. Rigid transformations

Rigid transformations, or rigid mappings, are defined as geometrical transformations that preserve all distances. These transformations also preserve the straightness of lines (and the planarity of surfaces) and all nonzero angles between straight lines. Registration problems that are limited to rigid transformations are called rigid registration problems. Rigid transformations are simple to specify, and there are several methods of doing so. In each method, there are two components to the specification, a translation and a rotation. The translation is a three-dimensional vector \( t \) that may be specified by giving its three coordinates \( t_x, t_y, t_z \) relative to a set of \( x,y,z \) Cartesian axes or by giving its length and two angles to specify its direction in polar spherical coordinates. There are many ways of specifying the rotational component, among them Euler angles and quaternions. We will utilize orthogonal matrices. With this approach, if \( T \) is rigid, then

\[
x' = A(x) + t
\]

where \( A \) is a 3x3 orthogonal matrix, meaning \( A^t A = AA^t = I \) (the identity matrix). Thus \( A^t = A^{-1} \).

This class of matrices includes both the proper rotations (preserve orientation), which describe physical transformations of rigid objects, and improper rotations (reverse orientation), which do not. These latter transformations both rotate and reflect rigid objects. Improper rotations can be eliminated by requiring the determinant of \( A \) to be equal to one, that is \( \det(A) = +1 \).

Proper rotations can be parameterized in terms of three angles of rotation \( \theta_x, \theta_y, \theta_z \) about the respective Cartesian axes, the so-called "Euler angles." The rotation angle about a given axis is, with rare exception, considered positive if the rotation about the axis appears clockwise as viewed from the origin while looking in the positive direction along the axis. The rotation of an object (as opposed to the coordinate system to which it is referred) about the \( x, y \) and \( z \) axes, in that order leads to

\[
A = \begin{pmatrix}
\cos \theta_z & -\sin \theta_z & 0 \\
\sin \theta_z & \cos \theta_z & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_y & 0 & \sin \theta_y \\
0 & 1 & 0 \\
-\sin \theta_y & 0 & \cos \theta_y
\end{pmatrix}
\begin{pmatrix}
1 & 0 & 0 \\
0 & \cos \theta_x & -\sin \theta_x \\
0 & \sin \theta_x & \cos \theta_x
\end{pmatrix}
\]

\[
= R_z(\theta_z) \cdot R_y(\theta_y) \cdot R_x(\theta_x)
\]

(4-10)

with the three matrices \( R_z(\theta_z), R_y(\theta_y) \) and \( R_x(\theta_x) \) representing the rotations about \( z, y \) and \( x \) respectively (in reverse order because they are applied from right to left). Other angular parameterizations are sometimes used, including all permutations of the order of \( R_z(\theta_z), R_y(\theta_y) \) and \( R_x(\theta_x) \).

Proper rotations can also be parameterized in terms of a rotation axis and an angle, through the use of quaternions. The quaternion is a vector \( q \) consisting of four elements, \( q_0, q_1, q_2, q_3 \), and obeying special rules for multiplication. A quaternion of unit length (versor) can be used to represent a rotation, where the elements are equal respectively to \( \cos \theta/2, \omega_x \sin \theta/2, \omega_y \sin \theta/2 \) and \( \omega_z \sin \theta/2 \), where \( \omega_x, \omega_y \) and \( \omega_z \) are the components of the rotational axis and \( \theta \) the angle of rotation about that axis. In this case the matrix \( A \) is given by
Chapter 4 – Medical Image Registration

The latter parameterization is used in the implemented framework of this thesis. The choice was based on the fact that, since we have a rotation centre different from the coordinates' origin, it is simpler to understand the rotation in terms of a rotation axis passing through that centre and the rotation angle.

4.6. Direction of transformation

The transformation can be realized in a forward or backward (inverse) manner. In order to understand that, let’s imagine we have an image (input image in Figure 4-2) and with an estimated transformation we want to get a transformed image (output image in Figure 4-2). Each voxel from the input image can be directly transformed using the estimated transformation. This approach, called forward mapping as shown in Figure 4-2, has the problem that two or more voxels in the input image can be transformed to the same location in the output image raising the question of how to combine multiple output values into a single output voxel. Besides, it is possible that some output locations may not be assigned a value at all. As a result from the above, it can produce holes and/or overlaps in the output image. Hence, the backward approach is usually chosen using the inverse of the estimated transformation. In this way neither holes nor overlaps can occur in the output image because the value of the output voxel intensity is computed by interpolation in the input image.

\[
A = \begin{pmatrix}
q_x^2 + q_y^2 - q_z^2 - q_x^2 & 2q_xq_y - 2q_0q_z & 2q_xq_z + 2q_0q_y \\
2q_xq_y + 2q_0q_z & q_x^2 - q_z^2 + q_y^2 - q_x^2 & 2q_yq_z - 2q_0q_x \\
2q_xq_z - 2q_0q_y & 2q_yq_z + 2q_0q_x & q_y^2 - q_z^2 - q_x^2 + q_z^2
\end{pmatrix}
\]

(4-11)

The left image shows the forward mapping, where the input image pixel is mapped onto the output image. In this case some output pixels might be “hit” more than once (overlapping) or might not be “hit” at all (holes). The right image shows the inverse mapping, where the output image pixels are mapped back onto the input image. In this case output pixel value must be interpolated from a neighbourhood in the input image. Extracted from [30].

4.7. Interpolation

Interpolation is the process by which an intensity value at a location between voxels is estimated. In inverse mapping, as already mentioned, interpolation in the input image deals with the assignment of values to voxels in the transformed image (output image).

Interpolation is necessary for any realistic transformation, since scalar values exist only on the grid points of the dataset. The nearest neighbour function, trilinear functions, cubic
Bsplines, higher-order B-splines and truncated sinc functions belong to the most commonly used functions for interpolation. Here the nearest-neighbour and the trilinear interpolation are described.

Nearest-neighbour interpolation is the simplest but least accurate method of interpolation. The voxel closest to interpolation point amongst eight neighbouring voxels is identified and its intensity value is assigned to the interpolation point. This interpolation is preferred when the image to be transformed contains low number of intensities and we do not want to introduce 'synthetic' gray levels/colours by higher order interpolation.

Smother results are obtained at the cost of higher computational complexity with trilinear interpolation. Trilinear interpolation assumes the intensity varies linearly with the distance between the grid points along each direction. It considers all the contributions to interpolation point from the eight neighbouring voxels.

![Figure 4-3 The importance of interpolation. The interpolated images show a 50x50x50 sphere. On the left, nearest neighbour interpolation was used, and the underlying voxel structure of the dataset is clearly visible. On the right, trilinear interpolation is used, and the sphere appears smooth even in the close-up image. Extracted from [30].](image)

Intensity based methods are iterative, so low computational cost interpolators must be used during the registration, such as the ones described above. When transforming points from one image to another, interpolation is usually required to estimate the gray value of the resulting point. Because the interpolation is not perfect, it alters the intensity histogram. Trilinear interpolation, for example, causes a low-pass filtering of the images, whose interaction with edges will tend to smooth the histogram, and in spatially smooth areas will reduce noise and hence contribute to a sharpening of the histogram. The consequence of this effect can be an interpolation-dependent change in the value of the similarity measure, which can cause fluctuations of the similarity measure with the period of the voxel separation. The interpolation artefacts are greatest for high spatial frequency features in the images. One solution to the problem is, therefore, to blur the images prior to carrying out registration, thereby reducing the artefacts. The effect is also reduced if the starting estimate of the transformation includes a rotational component, or if the voxel dimensions in the images being registered are different.

Taking all the above into consideration trilinear interpolation was used in our implemented framework.
4.8. Optimization

In order to register two images using intensity based methods, it is necessary to find the optimal value of the similarity measure over a parameter space with dimensionality defined by the number of degrees of freedom of the transformation. There are two classes of optimization algorithms that can be used: those that use derivative information and those that do not.

As already mentioned, intensity based methods require an iterative approach, in which an initial estimate of the transformation is gradually refined by trial and error. In each iteration the current estimate of the transformation is used to calculate a similarity measure. The optimization algorithm then makes another (hopefully better) estimate of the transformation, evaluates the similarity measure again, and continues until the algorithm converges, at which point no transformation can be found that results in a better value of the similarity measure, to within a preset tolerance. Where derivatives can be calculated efficiently, they should be used to speed up optimization. A detailed discussion of optimization techniques is outside the scope of this chapter. A review of optimization techniques is given in [25]. Here some characteristics of the optimization methods are analyzed and the ones implemented in our framework are presented, namely the gradient descent and the differential evolution algorithm.

It is sometimes useful to consider the parameter space of values of the similarity measure. For rigid body registration there are six degrees of freedom, giving a six-dimensional parameter space. Each point in the parameter space corresponds to a different estimate of the transformation. The parameter space can be thought of as a high-dimensionality image in which the intensity at each location corresponds to the value of the similarity measure for that transformation estimate. If we consider dark intensities as good values of similarity, and high intensities as poor ones, an ideal parameter space image would contain a sharp low intensity optimum with monotonically increasing intensity with distance away from the optimum position. The job of the optimization algorithm would then be to find the optimum location given any possible starting estimate.

In many optimization problems it is desirable to determine the globally optimal solution. For image registration, however, the desired optimum is frequently a local rather than global extremum of the similarity measure. The reason that the global optimum is not desirable can be understood by careful consideration of the behaviour of the similarity measures for large transformations. For example, when transforming one image to such an extent that only an area of background overlaps for the two images, the similarity measure CC, in the case that this is multiplied by minus one (-1), will give a low value, as in the case where the two images are correctly aligned. Likewise, in the case of mutual information, a very tight cluster in that area of the histogram is observed, corresponding to background intensities in both images. This transformation will produce lower joint entropy and higher mutual information, than the correct alignment.

The solution to this problem is to start the algorithm within the ‘capture range’ of the correct optimum that is within the portion of the parameter space in which the algorithm is more likely to converge to the correct optimum than the incorrect global one. In practical terms, this requires that the starting estimate of the registration transformation is
reasonably close to the correct solution. The size of the capture range depends on the features in the images, and cannot be known \textit{a priori}, so it is difficult to know in advance whether the starting estimate is sufficiently good.

It is essential when registering images using these algorithms to inspect the solution visually. A solution that lies outside the capture range is immediately recognizable as incorrect, and a better solution can be found by re-starting the algorithm with a user-provided starting estimate that is nearer to the correct solution.

**Gradient descent**

In the context of registration, the transformation parameters define the search space for optimizers. That is, the goal of the optimization is to find the set of parameters defining a transformation that results in the best possible value of a similarity measure.

In the case of the gradient descent algorithm, the derivative of similarity measure is required. In order to compute the measure's derivative, the knowledge of the Jacobian is required. The Jacobian is a matrix whose elements are the partial derivatives of the output point with respect to the transformation parameters:

$$J = \begin{bmatrix}
\frac{\partial x'_1}{\partial p_1} & \frac{\partial x'_1}{\partial p_2} & \cdots & \frac{\partial x'_1}{\partial p_m} \\
\frac{\partial x'_2}{\partial p_1} & \frac{\partial x'_2}{\partial p_2} & \cdots & \frac{\partial x'_2}{\partial p_m} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial x'_n}{\partial p_1} & \frac{\partial x'_n}{\partial p_2} & \cdots & \frac{\partial x'_n}{\partial p_m}
\end{bmatrix}$$

where $p_i$ are the transformation parameters and $x'_i$ are the coordinates of the output point. The Jacobian can be interpreted as a matrix that indicates for a point in the input space how much its mapping on the output space will change as a response to a small variation in one of the transformation parameters. Note that the values of the Jacobian matrix depend on the point in the input space. So actually the Jacobian can be noted as $J(x'_i)$.

The similarity measure's derivative can then be computed:

$$\frac{\partial S(p \mid I_1, I_2, T)}{\partial p_j} = \sum_j \frac{\partial S(p \mid I_1, I_2, T)}{\partial x'_j} \cdot \frac{\partial x'_j}{\partial p_j}$$

where $I_1$ and $I_2$ are the two images to be registered and $T$ the transformation.

The gradient descent algorithm, in each iteration, updates the current position according to

$$p_{n+1} = p_n + \frac{\partial S}{\partial p_n} \lambda$$

where $\lambda$ is the learning rate of algorithm, $\frac{\partial S}{\partial p_n}$ the step length, $n$ is the iteration number, $p_n$ is a vector of the transformation parameters at the current iteration and $S$ the similarity measure. The influence the learning rate has during the optimization is shown in Figure 4-4.
It should be noted that gradient search based algorithms are easily attracted by local minima, as shown in Figure 4-5.

Differential Evolution

Differential Evolution (DE) is an evolutionary strategy introduced by Storn and Price [32]. In the method, a population (NP) of random D-dimensional parameter vectors $x_{i0} =$
\((x_{1,G}, x_{2,G}, \ldots, x_{D,G})\) is generated and in every generation a set of new vectors \(x_{i,G+1}\) is constructed from already existing elements by

\[
x_{i,G+1} = x_{r_1,G} + F \cdot (x_{r_2,G} - x_{r_3,G}), \quad i = 1, 2, \ldots, NP
\]  

\((4-15)\)

where \(x_{i,G+1}\) is a new mutant vector, \(x_{i,G}\) is a population vector, \(r_1, r_2, r_3\) are random indices \(\in \{1, 2, \ldots, NP\}\), \(G\) is the generation number, \(F\) is a weighting factor and \(NP\) is the population size. At the beginning of the optimization (generation zero), the initial parameter values for the \(x_{i,G}\) are drawn randomly from the initial parameter range (IPR) that are defined according to the problem to be solved. In other words, for each parameter \(x_{j,0}\) the following equation holds:

\[
x_j^l \leq x_{j,0} \leq x_j^u, \quad \text{IPR: } x_j \in [x_j^l, x_j^u] \text{ and } j = 1, 2, \ldots, D
\]  

\((4-16)\)

\(F\) is a real and constant factor \(\in [0, 2]\) which controls the amplification of the differential variation \((x_{r_2,G} - x_{r_3,G})\). Note that the smaller the differences between parameters of parent \(r_2\) and \(r_3\), the smaller the difference vector and therefore the perturbation. That means if the population gets close to the optimum, the step length is automatically decreased.

In order to increase the diversity of the perturbed parameter vectors, crossover (CR) is introduced. To this end, the trial vector is formed:

\[
x_{i,G+1} = (u_{1,G+1}, u_{2,G+1}, \ldots, u_{D,G+1})
\]  

\((4-17)\)

where

\[
u_{i,G+1} = \begin{cases} 
    v_{i,G+1} & \text{if } (randb(j) \leq CR) \text{ or } j = \text{rnbr}(i) \\
    x_{i,G} & \text{if } (randb(j) > CR) \text{ or } j \neq \text{rnbr}(i)
\end{cases}, \quad j = 1, 2, \ldots, D
\]  

\((4-18)\)

In \(4-18\) \(randb(j)\) is the \(j\)-th evaluation of a uniform random number generator with outcome \(\in [0, 1]\). \(CR\) is the crossover constant \(\in [0, 1]\), which has to be determined by the user. \(CR=0\) means no crossover. \(rnbr(i)\) is a randomly chosen index \(\in 1, 2, \ldots, D\) which ensures that \(u_{i,G+1}\) gets at least one parameter from \(v_{i,G+1}\). Otherwise no new parent vector would be produced and the population would not alter.

The resulting trial vector \(u_{i,G+1}\) is compared to the target vector \(x_{i,G}\) using the "greedy" criterion. The one that yields the best value of the objective function is retained for the next generation. The algorithm is repeated a selected number of generations \(G\) (there is no stopping condition).

The above scheme is not the only variant of DE. In order to classify the different variants, the notation: \(DE/x/y/z\) where \(x\) specifies the vector to be mutated, \(y\) the number of difference vectors used, and \(z\) denotes the crossover scheme. ‘x’ can be ‘rand’ (a randomly chosen population vector) or ‘best’ (the vector with the best value from the current population). ‘z’ can be exponential or binomial crossover scheme. The case of the binomial scheme is explained above. For exponential crossover, the starting position of crossover is chosen randomly from \(1, \ldots, D\), and \(L\) consecutive elements (counted in circular manner) are taken from the mutant vector \(v_{i,G+1}\). Probability of replacing the \(k\)-th element in the sequence \(1, 2, \ldots, L, L \leq D\), decreases exponentially with increasing \(k\) [33].
Using the notation, the basic DE-strategy can be written as $ DE/rand/1/bin $. Another possibility for example is the method $ DE/best/2/bin $, where the mutant vector is

$$
V_{i,G+1} = \bar{x}_{best,G} + F \cdot (x_{r1,G} - x_{r2,G} + x_{r3,G} - x_{r4,G}), \quad i = 1, \ldots, NP \tag{4-19}
$$

More details about DE and its theoretic background can be found in [34].

Differential evolution is a global optimization algorithm. Figure 4-7 illustrates an example of the differential evolution algorithm over Ackley’s function, which is given by:

$$
f(x) = 20 + \exp(1) - 20 \exp\left(-0.02 \cdot \sqrt{D^{-1} \cdot \sum_{j=1}^{D} x_j^2}\right) - \exp\left(D^{-1} \cdot \sum_{j=1}^{D} \cos(2\pi x_j)\right) \tag{4-20}
$$

In the example illustrated it is $ x = (x_1, x_2), \ D = 2, \ IPR: x_j \in [-5, 5] $. The algorithm has to search for a $ x^* \in [-5, 5] $ such that $ f(x^*) \leq f(x) \ \forall \ x \in [-5, 5] $ with parameters $ NP = 10, F = 0.5 $ and $ CR = 0.1 $. The optimum solution is $ x^* = (0, 0) $ that gives $ f(x^*) = 0 $.

Figure 4-6 Crossover schemes. In the case of the binomial crossover, once an initial, randomly chosen parameter is inherited from the mutant $ V_{i,G} $ (e.g. $ j_{\text{rand}} = 3 $), D-1 independent trials are conducted to determine the source of the remaining parameters. If $ \text{rand}(j) \leq CR $, the mutant donates a parameter value; otherwise parameters are copied from the target vector $ G_{i,G} $. In the case of the exponential crossover, starting at the randomly chosen parameter index $ (j_{\text{rand}} = 3) $, trial parameters are inherited from the mutant $ V_{i,G} $ as long as $ \text{rand}(j) \leq CR $ (e.g. $ j=4, 5 $). The first time that $ \text{rand}(j) > CR $, all the remaining trial parameters (e.g. $ j=6,7,0,1,2 $) are inherited from the target vector $ G_{i,G} $. In both cases $ D=8 $. Extracted from [34].
Chapter 4 – Medical Image Registration

Figure 4-7 Differential evolution algorithm (DE) with parameters NP = 10, F = 0.5 and CR = 0.1 over Ackley’s function for 2 variables. In (a) Ackley’s function is illustrated, in (b), (c), (d), (e), (f), (g), (h), and (i) the generations 0, 1, 3, 7, 10, 15, 20 and 30 of the DE are shown respectively. Extracted from [35].
Chapter 5

Implemented Framework

This chapter presents the framework that is implemented in order to define the spatial geometrical transformations that map the expression patterns of the genes co-stained with the reference gene within a common template. The concept and overview of the framework are described in sections 5.1 and 5.2 respectively. The chapter continues with the implementation details of every step the framework is composed of (sections 5.3 and 5.4). In each step the process is first described in general terms and then the specific components that are used are given. To visualize the outcome of each step the reader is referred to Chapter 6, which presents the data used and the results of every step.

The basic image processing concepts that are used in the framework can be found in Appendix B. The implementation is done in C++ and is based on the National Library of Medicine Insight Segmentation and Registration Toolkit (ITK) (Appendices C and D). A User’s Manual for the implemented programs is provided in Appendix E.

5.1. Concept

The general concept, on which the implemented framework is based, is that a complete view of a zebrafish instance can serve as the template, into which partial views containing information on gene expressions patterns can be registered. The registration is guided by the common reference gene (gsc) and the embryo’s geometrical shape given by nuclei staining. An important assumption is that the embryos are similar in size and shape at the same developmental stage.

The idea behind the framework’s design is based on the very nature of our data. As already mentioned, our data include one view of a whole zebrafish embryo and partial views of other embryos, taken from the same developmental stage. Each view is composed of different channels. The whole embryo view is composed of two channels, one that contains the nuclei and the other the expression pattern of the gsc gene. Each partial view is composed of three channels. The first two contain the same information as the whole view. The third contains the expression pattern of another gene, the one that was co-stained with the gsc.

Our problem is the mapping of all the available gene expression patterns on the whole embryo view. This mapping is an iterative procedure that involves the template and a different partial view each time. In each partial embryo view the relative location of the expressions of the two genes is known. To be able to map these expressions on the template a reference position is necessary and that is provided by the expression of the gsc gene given by the corresponding channel in both template and partial views.

So, our initial problem is now even more focused: what needs to be done first is to bring into alignment the expressions of the reference gene of the whole embryo and the partial view.
Chapter 5 – Implemented Framework

This is an image registration problem whose solution is the transformation, which when applied on the partial view, brings it into spatial alignment with the template. This transformation is the answer to our initial problem. Since we know how to map the expression of the reference gene depicted in a partial view on the template, we know also how to map the expression of the co-stained gene contained in the same partial view. This is done by applying the same transformation on the third channel of the partial view; hence the relative location between the two expressions of the reference and co-stained gene is kept.

As mentioned this process can be repeated with the same template and partial views that contain expressions of different genes in their third channel. This way, for a given developmental stage, the mapping of different gene expression on a template can be accomplished. This process can be taken one step further by extending it to other developmental stages as well.

5.2. Framework overview

The implemented framework can be broken up into two parts: one is the registration pipeline, which provides the spatial transformation, and the other is the atlas construction pipeline, which applies that transformation on the channels that contain the expressions of the co-stained genes. The different steps involved in the above are shown in Figure 5-2.
Chapter 5 – Implemented Framework

The registration pipeline works with two sets of images: one that will remain fixed (static) throughout the procedure and another that will be spatially mapped to align with the first. In our case every time the framework is run, one of the partial views acts as the moving image and the complete view always acts as the fixed image, as the common template for all the partial images to be registered. Since the process is guided by the embryo’s geometrical shape and the location of the gsc expression pattern, the corresponding channels of our data are employed: the channel that contains the nuclei information and therefore gives details about the anatomical constrains and the channel that contains the gsc expression pattern, which provides the universal reference about where the other, co-stained gene expression patterns are to be placed.

Prior to the registration step itself, both channels of both the complete and the partial view to be registered are preprocessed (preprocessing step) and then combined into a single image keeping both the nuclei structural information and the gsc’s expression details (addition step). This way the fixed and the moving image, that take part in the registration procedure, are created. These images could be fed directly to the registration step. However the better the images overlap before the registration procedure, the more likely it is to obtain a successful alignment. Bearing that in mind, an initialization step was implemented to coarsely place the moving image we have each time onto the template. After that, the fixed and the initialized moving image are passed onto the registration step, the output of which are the transformation parameters, that bring into alignment the two images, and the registered image, which is the initialized moving image transformed by these transformation parameters.

Once the desired transformation parameters are found, they are given to the atlas construction pipeline to use them in order to map the third channel of the partial view that took part in the registration onto the template.

The implemented framework repeated with the same template and other partial views, that contain different gene expression patterns in their third channel, can lead to the construction of a gene expression atlas for the given developmental stage.
Over the following sections each step's implementation is described, along with the corresponding ITK components.

5.3. Registration pipeline

The goal of the first pipeline of the implemented framework is to determine these transformation parameters that bring into spatial alignment the template and one of the partial views. The steps required to achieve this is the preprocessing step, the addition step, the initialization step and the step that performs the actual registration. The role they play in the framework, along with the implementation’s details, are described in the following sections.

5.3.1. Preprocessing Step

The first building block of the registration pipeline is the preprocessing step, which, as its name implies, processes both channels of the two views that will take part in the registration, the whole embryo view (template) and the partial view. Its purpose is to remove any noise, if present, to perform blurring and downsampling, thereby reducing computational time, and to apply thresholding, which is necessary when a binary mask is needed. This step works on each channel separately and produces a preprocessed version of it that depends on the user’s choices. Its pipeline is given in Figure 5-3.

The preprocessing data pipeline is given one input, the channel to be processed. Bearing in mind that preprocessing is not a standard procedure for all channels but depends highly on the kind of noise they have, the foreground and background’s intensity values and their size, a number of choices is given to the user for parameterizing this step’s execution. For example, if there are light details in the background and they have to be removed, then opening should be applied. If the size of the image results in large computational time, then downsampling is necessary. The downsampling process requires the data to be preprocessed with a smoothing filter to blur the image, thereby avoiding the occurrence of aliasing effects due to overlap of the spectrum in the frequency domain. Then the input can be turned into a binary image by thresholding it with an intensity value. Depending on what part of the image the user wishes to segment, a threshold intensity value can be defined after visual inspection. If the user is interested only on the geometry of that segmented part, closing is available to fill in any holes.

All of these choices can be seen in Figure 5-4, which shows the flowchart of the algorithm implemented for this step, along with the corresponding ITK components used and the parameters that the user can provide, before running the corresponding program (see Appendix E). A short overview of the used ITK components follows.
Components

The input to the algorithm is given by implementing the *ImageFileReader* class. This class is responsible for reading images and is located at the beginning of any data processing pipeline. ITK does not enforce any particular file format; however it recommends the use of file formats that combine a text header file and a raw binary file, like MetaImage-format. MetaImages are raw files with an additional text header file (MetaHeader) containing the required data to read an image (e.g. datatype, size, origin, spacing). MetaHeaders have the extension .mhd and the raw files .raw. This file format is the one that our data are converted to.

Morphological opening on a grayscale image is performed by the *GrayscaleMorphologicalOpeningImageFilter* class. The user can provide the size of the structuring element.

Figure 5-4 Overview of the preprocessing step. The first column is the flowchart of the implemented program. The second column shows the ITK components implemented for the corresponding process. The third column shows the parameters that the user can provide.
Smoothing is performed by the *MeanImageFilter* class, given that it provides a convenient run-time performance. This class is an averaging filter; it computes the value of each output voxel by finding the statistical mean of the neighbourhood of the corresponding input voxel. This way the high spatial frequencies in their spectrum are attenuated. The user can choose the size of the neighbourhood over which the mean is computed.

The smoothed images can be downsized either by the *ShrinkImageFilter* or the *ResampleImageFilter* class. The only difference in their usage lies on whether the user wants to keep the spatial position of the image’s origin, when that differs from the origin of the physical coordinates. In both cases, the user provides the downsampling factor in each dimension. In the *ResampleImageFilter* the physical coordinates of the origin are preserved, whereas as in the *ShrinkImageFilter*, they are divided by the downsampling factors.

The *BinaryThresholdImageFilter* class converts the input into a binary image by changing the voxel values according to the rule illustrated in Figure 5-5. For each voxel in the input image, the value of the voxel is compared with the lower and upper thresholds. In our implementation the InsideValue is set to one, the OutsideValue is set to zero, the upper threshold is set to 255 and the lower threshold is chosen by the user. In the case that a threshold value is not chosen but yet thresholding is desired, then the *OtsuThresholdImageFilter* is used instead of the *BinaryThresholdImageFilter*. This filter separates the image into foreground and background components by computing the Otsu's threshold [37].

For closing there are two options. In the case that the image is thresholded, then the *BinaryMorphologicalClosingImageFilter* performs a binary morphological closing. If the image is not thresholded but still closing is desired, then the *GrayscaleMorphologicalClosingImageFilter* performs a grayscale morphological closing. Again the size of the structuring element for the closing can be defined by the user.

Finally the class responsible for writing images, the *ImageFileWriter* is located at the end of the data processing pipeline.

![Figure 5-5 Transfer function of the BinaryThresholdImageFilter. Extracted from [36].](image-url)
5.3.2. Addition Step

The second building block of the registration pipeline is the addition step. Its purpose is to combine the information given by the nuclei and the gsc channels separately into a single combined image. This process is performed on each view, template and partial, separately. It takes as input the nuclei and gsc channels and produces as output a combined image. The resulting image from the template view is the fixed image in our registration scheme and the resulting image of the partial view the moving image. The pipeline of this step is shown in Figure 5-6.

![Figure 5-6 Pipeline of the addition step](image)

This step is necessary because of the fact that the nuclei channel gives details about the anatomical constraints while the gsc channel provides the feature that refines the registration. In order to combine the information given by the different channels we had to form a general equation, based on which we could design the step. This general equation is given by:

\[
\text{combined image} = \text{preprocessed gsc channel} + \alpha \times \text{preprocessed nuclei channel}
\]  

(5-1)

The parameter alpha (\(\alpha\)) balances the weight each channel has in guiding the registration process. Whether this parameter plays a crucial role in the results of the registration is evaluated with data in Chapter 6.

From the previous preprocessing step either both channels can be kept grayscale or the nuclei channel can be turned into a binary mask, thus keeping the generic information of the zebrafish shape, which does not outline specific geometric variations across specimens. Depending on the above different addition methods are implied, as the estimation of \(\alpha\) is different.

In the first case (method one) where both channels are kept grayscale, \(\alpha\) is estimated by the equation:

\[
\alpha = \frac{\sum_{i=1}^{n} l_i}{\sum_{j=1}^{m} l_j \cdot \text{factor}}
\]

(5-2)

where \(l_i\) is the intensity value of each voxel that belongs to gsc expression region in the gsc channel, \(n\) is the total number of voxels of this region, \(l_j\) is the intensity value of each voxel that belongs to the nuclei region in the nuclei channel, \(m\) is the total number of voxels of this region and \(\text{factor}\) is just a constant, chosen by the user.

In the second case (method two) \(\alpha\) is estimated by the equation:
where $I_i$ is the intensity value of each voxel that belongs to gsc expression region in the gsc channel, $n$ is the total number of voxels of this region and $factor$ is just a constant, again chosen by the user.

The flowchart of the implemented program for this step, as well as the used ITK components and parameters that can be provided by the user, before running the corresponding program (see Appendix E), are shown in Figure 5-7. A presentation of the used ITK components follows, bearing in mind that both addition methods use the same components.

**Components**

At the beginning and at the end of the data processing pipeline the classes responsible for reading (`ImageFileReader`) and writing images (`ImageFileWriter`) are located respectively.

---

**Figure 5-7** Overview of the addition step. The first column is the flowchart of the implemented program. The second column shows the ITK components implemented for the corresponding process. The third column shows the parameters that the user can provide.
In order to have an indication of the parameter alpha (\(\alpha\)) equation 5-2 or equation 5-3 is used, depending on the method chosen. In the first case the user must provide the thresholds to isolate the nuclei and the \textit{gsc} expression regions in the corresponding channels. Once the average intensity of each region is computed, their ratio is divided by a constant (eq. 5-2). In the second case the user must provide only one threshold to isolate the \textit{gsc} expression region, since the nuclei channel here is a binary mask. Then the region’s average intensity is computed and divided by a constant (eq. 5-3). The division factors can be provided by the user. All the above computations are accomplished with the help of the \textit{ImageRegionIteratorWithIndex} class.

Depending on the method chosen, the following two processes, the multiplication of the nuclei channel and the casting of both channels, are interchanged. Casting is performed to overcome the problem of overflow. Each voxel value in our data is represented by 8 bits. By implementing a \textit{CastImageFilter}, casting is performed on every voxel in our data and the value’s representation is extended to 16bits/voxel. The multiplication of the nuclei channel, grayscale or binary mask, by the estimated parameter alpha (\(\alpha\)) is performed voxel by voxel by the \textit{MultiplyByConstantImageFilter}.

The actual addition of the \textit{gsc} channel and the multiplied nuclei channel, is performed by the \textit{ConstrainedValueAdditionImageFilter}.

Before writing the combined image, a \textit{RescaleIntensityImageFilter} is implemented in order to linearly scale the voxel values so that the minimum and maximum values of the combined image are mapped to minimum and maximum values of the dynamic range of 8bits/voxel. The linear transformation applied by this filter can be expressed as

\[
\text{output voxel} = (\text{input voxel} - \text{inputMin}) \times \frac{\text{outputMax} - \text{outputMin}}{\text{inputMax} - \text{inputMin}} + \text{outputMin} \quad (5-4)
\]

### 5.3.3. Initialization Step

The third building block of the registration pipeline is the initialization step. Its purpose is to coarsely place the moving image in the fixed image’s space as an initial positioning, by calculating a geometrical transformation. The better the images overlap before the registration process, the less misalignment the registration algorithm has to recover and more are the chances to obtain a successful alignment. The input of this step is a set of six images: the four that form the output of the preprocessing step and the two output images of the addition step. The four first images, that are the preprocessed nuclei and \textit{gsc} channels of the template and partial view, are used for the calculation of the geometrical transformation. An important detail here is that the binary masks of the nuclei channels are needed. This step by applying the calculated transformation to the moving image produces the initialized moving image as its output. Also, the rotation centre that is used in this step is passed as an output to the next. Figure 5-8 shows the pipeline of the initialization step.

The importance of this step lies in the fact that if the initialization does not provide sufficient overlapping, the registration will fail. Classes that perform initialization are provided by ITK; however when using them registration fails. So we understood that our problem requires a
different approach as to how an initial positioning of the moving image in the fixed image’s space should be determined. This different approach is performed by this initialization step.

![Figure 5-8 Pipeline of the initialization step](image)

The algorithm implemented is based on the nature of our data. First of all from data acquisition it is known that all views were taken with the same direction in the ‘z’ axis. This fact ensures the same curvature in all views. Secondly, taking advantage of the fact that the nuclei and gsc channels (before the addition) of each view reveal different kind of information, two different centres of mass (gravity centres) based on the intensity values in each view can be calculated. The first centre of mass is calculated from the nuclei channel and it represents the gravity centre of the nuclei, whereas the second is calculated from the gsc channel and represents the gravity centre of the gsc expression. Threshold values are needed for the calculation of the centres of mass for the gsc expression so as to determine the gene expression regions in the gsc channel for both views. The centres of mass for the nuclei are calculated from the mask, so there is no need for a threshold value. The coordinates of the centres are given by:

\[
[c_x, c_y, c_z] = \left[ \frac{\sum_{x=0}^{X} l \cdot x}{\text{mass}}, \frac{\sum_{y=0}^{Y} l \cdot y}{\text{mass}}, \frac{\sum_{z=0}^{Z} l \cdot z}{\text{mass}} \right]
\]  

(5-5)

where \( l \) is the intensity value of each voxel belonging to the specific region (nuclei or gsc expression), \( x, y, z \) are the physical positions of the voxel in the \( x, y, z \) axes and \( X, Y, Z \) are the physical sizes of the view along the axes. The mass is the sum of all the intensity values of all the voxels in the region of interest, that is \( \text{mass} = \sum_{i=1}^{n} l_i \), where \( l_i \) is the intensity value of each voxel that belongs to the region and \( n \) is the total number of voxels in that region.

Using equation 5-5 four gravity centres are determined, one from each preprocessed channel of each view: the gravity centre of the nuclei (nfixed) and of the gsc (gscfixed) for the complete template and the gravity centre of the nuclei (nmoving) and of the gsc (gscmoving) for the partial view. These four points define a spatial transformation that applying it on the moving image provides an appropriate initial positioning to our problem.
Chapter 5 - Implemented Framework

The transformation, which is illustrated in Figure 5-9, firstly translates and then rotates the moving image. The translation part is a vector defined by the gsc gravity centres of both views. That means that firstly the moving image is translated so that the two gsc gravity centres coincide. Then the moving image is rotated. The angle and the rotation axis are determined by two vectors, vF and vM. Both vectors start from the gsc gravity centre of the fixed image (gscfixed). vF ends at the nuclei gravity centre of the fixed (nfixed) and vM ends at the translated nuclei gravity centre of the moving (translated nmoving). The centre of rotation is the gsc gravity centre of the fixed (gscfixed). The rotation angle is the angle defined by the vectors and the rotation axis is defined by their cross product.

This kind of initialization solves a critical issue in the registration problem: the embryo curvature has the same sense in both fixed and initialized moving image. In addition, the initialized moving image is roughly aligned to the fixed, facilitating the registration process to reach a successful alignment. It is notable that the calculation of the geometrical transformation to be applied on the moving image is based only on the preprocessed channels of each view. Therefore, the calculated transformation is independent of the method that produces the fixed and the moving images in the addition step.

The initialization step’s flowchart, as well as the used ITK components and parameters that can be provided by the user, before running the corresponding program (see Appendix E), are shown in Figure 5-10. An overview of the used ITK components follows.

![Diagram](image)

Figure 5-9 Initialization procedure based on the gravity centre of nuclei (nfixed) and gsc expression (gscfixed) of the template fixed image, and the gravity centre of nuclei (nmoving) and gsc expression (gscmoving) of the moving partial image. First, translation is performed, marked by (1) and rotation is following, marked by (2).
Chapter 5 – Implemented Framework

Components

The data processing pipeline starts and ends with the classes responsible for reading (ImageFileReader) and writing images (ImageFileWriter) respectively.

For the computation of the gravity centres the ImageRegionIteratorWithIndex class was implemented (also seen in section 5.3.2). Here threshold values only for the gsc channels need to be defined, since the nuclei channels are given as binary masks. The class returns the physical coordinates of every centre calculated according to equation 5-5.

The transformation defined from these centres is applied on the moving image with the help of the VersorRigid3DTransform class. This class implements a rigid transformation where rotation is represented by versors and translation along each dimension by vectors. This kind of parameterization has already been presented in section 4.5.1. The transformation calculated, according to the procedure described in Figure 5-9, is composed firstly of the translation and then by the rotation. Therefore attention should be paid on the fact that the way the transformation here is set up is equivalent to having two separate transformations, one that only translates and another that only rotates. The translation is defined by the translation vector while the rotation is defined by a versor and the rotation centre. The versor itself is defined by the rotation axis and angle.

Figure 5-10 Overview of the initialization step. The first column is the flowchart of the implemented program. The second column shows the ITK components used for the corresponding process. The third column shows the parameters that the user can provide.
The transformation is passed as input to the ResampleImageFilter, along with the moving image and the LinearInterpolateImageFunction. The ResampleImageFilter maps the moving image into the discrete grid of the fixed image. The nature of the resampling process is such that an algorithm must go through every voxel of the fixed image and compute the intensity that should be assigned to this voxel from the mapping of the moving image. This computation involves taking the integral coordinates of the voxel in the fixed image grid, mapping them into the physical space of the fixed image (transformation T1), mapping those physical coordinates into the physical space of the moving image (the inverse translation and rotation calculated by the four gravity centres), then mapping the physical coordinates of the moving image in to the integral coordinates of the discrete grid of the moving image (transformation T2), where the value of the voxel intensity will be computed by interpolation. This procedure in two dimensions is shown Figure 5-11.

![Figure 5-11 Different coordinate systems involved in the resampling of the moving image to the grid of the fixed. Extracted from [36].](image)

### 5.3.4. Registration Step

The fourth building block of the registration pipeline is the registration step. Its purpose is to register the initialized moving image (produced by the initialization step) to the fixed (produced by the addition step). This step is given as input the two images, along with the rotation centre, which is also provided by the initialization step. From this registration step we obtain the transformation parameters, which optimize the similarity measure between the two images, and the registered image, that is produced when these parameters are applied on the initialized moving image. The corresponding pipeline is shown in Figure 5-12.

![Figure 5-12 Pipeline of the registration step](image)
The registered image in our case is not an end cause, as in other image registration applications, but the mean by which we can decide whether the transformation parameters that produced it were the desired ones. Our interest lies on the transformation parameters. These transformation parameters, that successfully map the gsc’s expression pattern of the partial view on the template, can be applied to map the expression pattern of the other gene co-stained with the gsc.

Registration is treated as an optimization problem with the goal of finding the spatial mapping that will bring the initialized moving image into alignment with the fixed image. This problem can be decomposed in a set of basic components, as seen in Figure 5-13, and is based on the registration framework proposed by ITK.

The transformation component represents the spatial mapping of points from the fixed image space to points in the initialized moving image space. Note that the transformation is realized in the inverse (backward) manner, as explained in section 4.6. Interpolation is used to evaluate the initialized moving image intensities at non-grid positions. The similarity measure provides a measure of how well the fixed image matches the transformed initialized moving image. This measure forms the quantitative criterion to be optimized over the search space defined by the parameters of the transformation (similarity measure function). Starting from an initial set of parameters, the optimization procedure iteratively searches for the optimal solution by evaluating the similarity measure at different position of the transformation parameter search space. In other words it searches for the optimal solution of the similarity measure function. Once the iterations stop due to the fulfilment of a stopping criterion, then the last transformation parameters are used to transform the initialized moving image and produce the registered image. The latter can be loaded to a visualization program to assess whether the registration was successful or not.

The above components and specific implementations of each one are already discussed in Chapter 4. By selecting a specific implementation of each of these components a particular registration method is defined. This selection must be based on the nature of the registration problem in order to define the method that provides a successful solution.

![Diagram of the registration step’s components.](image)

In our implementation of this step we selected for the transformation component a rigid transformation, assuming that the embryos are similar in size and shape at the same
developmental stage, and for the interpolation component the trilinear interpolation, since it provides a good run-time performance. As far as the similarity measure component is concerned, two different implementations are used, the correlation coefficient and the mutual information. Two different implementations are also used in the optimization component, the gradient descent and the differential evolution algorithm. Later, with the help of our data, we will evaluate which combination of the above is most suitable for our problem.

Figure 5-14 shows the implemented ITK components in the registration step and Figure 5-15 shows the parameters that the user can provide before running the corresponding implemented program (see Appendix E). Each component will be analyzed separately.

**Components**

**Transformation:** In order to determine the position of the initialized moving image in the fixed image’s space a rigid registration approach is needed. The transformation, chosen among the variety of components ITK provides, is the VersorRigid3DTransform, which has already been presented in the initialization step (section 5.3.3). As already explained, this transformation represents a rotation and a translation in three dimensions with six parameters. The first three correspond to the three independent components that define a versor. It is reminded that a versor is a unit quaternion so its fourth component can be computed so that the quaternion is of unit length. The other three parameters represent the translation vector, one parameter along each dimension. Moreover this class allows the centre of rotation to be specified. In order to restrict the transformation to be optimized to the domain where the gsc is expressed, the centre of rotation is set to be the gsc gravity centre of the fixed view. The initial set of parameters given to the transformation is an arbitrary rotation axis \([0.1, 0.1, 0.1]\), which is passing from the specified centre of rotation, a zero rotation angle and a translation vector \([0, 0, 0]\).

**Interpolation:** For the estimation of the intensities of the initialized moving image at non-grid positions as well as for the resampling of the initialized moving image, once the
optimum transformation parameters are found, trilinear interpolation is used, with the help of the *LinearInterpolateImageFunction*.

**Similarity measure:** The similarity measure is perhaps the most critical element of a registration problem, as it defines the goal of the process. Unfortunately, there are no clear-cut rules about how to select a similarity measure, other than trying some of them in different conditions. As mentioned above, two similarity measures to be optimized over the search space defined by the parameters of the transformation were employed, the correlation coefficient (normalized cross-correlation) and as an alternative, the mutual information.

Normalized cross-correlation allows registering views whose intensity values are linearly related. A variant is implemented in ITK by the *NormalizedCorrelationImageToImageMetric* class, which is insensitive to multiplicative factors between the two images. This similarity measure produces a function to be minimized over the transformation parameters’ space (cost function).

Mutual information (MI) is a good measure to detect nonlinear correlations between voxel’s intensities, hence adequate for images from different modalities or different resolutions. Among the several implementations ITK provides, the *MattesMutualInformationImageToImageMetric* class was chosen, which follows the method specified by Mattes *et al.* [38]. This class computes the negative mutual information, thereby producing a function to be minimized over the transformation parameters’ space (cost function).

The *MattesMutualInformationImageToImageMetric* class requires a few parameters to be set before computation. This is because to compute the MI the marginal and joint probability densities (PDF) are needed, as described in section 4.4. The densities are estimated at bins, uniformly spread within the dynamic range of the images, using a set of intensity samples drawn from the images. Entropy values are then computed by summing over the bins. The number of samples depends on the content of the images. The more smoothed an image is, the less samples it needs. In our implementation the number of spatial samples is set to 50% of the total voxels of each image. This percentage can be decreased once all the other parameters of the registration have been fine tuned. The number of bins has a dramatic effect on the performance of the optimization algorithm; the smaller their number is, the more likely for the algorithm to get trapped early in local minima. In our application, the number of bins is set to 50.

Mutual information can result more expensive than normalized cross-correlation in computational time.

**Optimization:** While the similarity measure is the most critical component, the most complex is the optimization algorithm. Its selection requires deep understanding of the function to be optimized and its use of the dynamic range of the transformation parameters.

Plotting our cost functions over the transformation parameters’ space would greatly enhance our understanding on how each function describes our problem and on the behaviour of the optimization algorithms. Unfortunately this is not an easy task, as we have a space of six dimensions. So a descriptive attempt follows as to how we expect these functions to “look” over our six-dimensional space.
Our similarity measures describe our problem as a cost function over the six-dimensional space. We understand that each resulting function has many local minima. Furthermore, it is likely that the global minimum of each one doesn’t correspond to a successful registration. This is because since we are dealing with views of two different embryos, the global minimum might just as well correspond to the overlapping of the background in the two views. The reason for the many local minima is the symmetries in the views under study. As long as the nuclei of the two views overlap, the resulting function’s value will correspond to a local minimum. Considering the above we suspect that the transformation parameters, which result in a successful registration, correspond to one of the local minima.

As already mentioned two different optimization algorithms are used: the gradient descent and the differential evolution. Both have already been presented in section 4.8.

The gradient descent algorithm is “myopic” so the initial parameters are especially important for its behaviour. However we don’t know a priori an appropriate set of parameters that can ensure a successful optimization. Here is where the importance of the initialization step is revealed and how its output affects the behaviour of this algorithm. If the initialization step manages to overlap the two images sufficiently then the starting point of the optimization should be close enough to the desired minimum so that the gradient descent algorithm can find it.

The differential evolution is a global optimization algorithm. However as explained above, it is possible that the solution to our problem is not the global minimum. So here it is especially important to constrain the algorithm over the function’s area where the desired minimum is expected. Again the role of the initialization step is critical. The better it manages to overlap the two images, the more constrained the space over which the algorithm optimizes the function.

For the gradient descent algorithm the VersorRigid3DTranformOptimizer class from ITK is used. This specialized version of the gradient descent algorithm is necessary as the transformation parameters are composed of versor and vector components. It advances the parameters in the direction of the gradient where a bipartition scheme is used to compute the step length (see equation 4-14). Whenever the direction of movement changes in the parametric space, the step length is reduced. The rate at which it is reduced is controlled by a relaxation factor. The higher the value of the relaxation factor, the lower the rate the step size is reduced. The relaxation factor, the initial and minimum step size and number of iterations are parameters to be tuned according to application. To these the transformation parameters’ scales should be added. This scaling is necessary in order to compensate for their dynamic numerical range. The parameters should be scaled with respect to the expected misalignment and to each other. The minimum step size and number of iterations act as stopping criteria for the optimization.

In our implementation we set the relaxation factor to 0.8 in order to prevent the premature shrinkage of the step size, the maximum and minimum step length to 1.0 and 0.01 respectively, the number of iterations to 100 and the scales to 1.0 for the versor space and to $10^{-4}$ for the vector space. That way we “tell” the algorithm that we expect larger steps in translation with respect to rotation. It should be noted that the program lets the user define values other than the above, since they were defined by non-exhaustive search.
For the differential evolution algorithm an implementation that was developed within the BIT research group was employed. This implementation follows the ITK development guidelines. The algorithm’s parameters to be tuned are the population size, the crossover probability, the weighting factor and the initial numerical range of parameters. The algorithm is repeated a selected number of generations (there is no stopping condition). In our implementation, taking into account the suggestion provided by Gämperle et al. [39], we set the population size to 15, the crossover probability to 0.8, the weighting factor to 0.6, the number of iterations to 100 and the initial numerical range of parameters to $10^{-5}$ for rotation and $10^{-4}$ for the translation. Again the user is free to define other values for the parameters, since the above values were defined by non-exhaustive search. The strategy used is the DE/best/2/bin.

It should be noted that the differential evolution algorithm is more expensive than the gradient descent in computational time.

**Resampling:** The final transformation, that is found when the optimization’s iterations stop, is passed as input to the *ResampleImageFilter*, along with the initialized moving image and the *LinearInterpolateImageFunction*. The *ResampleImageFilter* maps the initialized moving image into the discrete grid of the fixed image. The resampling process has been described in section 5.3.3. It should be noted here regarding Figure 5-11. The transformation optimized by the registration step is the one that maps the physical coordinated of the fixed image into the physical space of the initialized moving image.

Figure 5-15 shows the components implemented in this registration step as well as the parameters that the user can provide before running the corresponding program (see Appendix E).

![Implemented Registration step](image)

**Figure 5-15** Parameters the user can set in the implemented program. Apart from the rotation centre, which must be provided, the rest of the parameters have default values.

## 5.4. Atlas construction pipeline

Once the registration is successful, its output, that is, the transformation parameters, is used to transform the third channel of the partial view. This is done in order to map the expression pattern of the gene contained in that channel on the template. This part of the
framework is responsible for this transformation and involves three steps, shown in Figure 5-16. The first two steps have already been presented in sections 5.3.1 and 5.3.3 and their corresponding programs can be found in Appendix E. Here, only one new step has been implemented, the transformation step. The pipeline that constructs the atlas is shown in Figure 5-16.

First comes the preprocessing step that smooths and downsamples the image. Its output is given to the initialization step so that it transforms the third channel with the same transformation used to initialize the corresponding partial image in the registration pipeline, since the transformation parameters given by the registration pipeline are with respect to this initial position. This step works with the same set of images as before (section 5.3.3) with the difference that in the place of the moving image in Figure 5-8 is the downsampled version of the third channel, as this is the image to be transformed.

The third step performs the second and final transformation of the image using the transformation parameters found by the registration. This step is given the output of the initialization step along with the fixed image, the rotation centre and the final transformation parameters. Its output is the third channel mapped on the template. Figure 5-17 shows the flowchart of this step, as well as the implemented ITK components and the parameters that should be given by the user, before running the corresponding program (see Appendix E).

**Components**

As in all steps mentioned previously, the classes `ImageFileReader` and `ImageFileWriter` are at the beginning and end of the data pipeline respectively.

The transformation is again performed by the `VersorRigid3DTransform`. This class is the only logical choice since the transformation parameters, which are provided by the registration process, correspond to the parameters that describe this class. They are six parameters in total: the first three are the versor components and the other three the translation along each dimension. These are found by the registration with respect to a rotation centre; therefore the same rotation centre must be provided here as well.

Finally, the image is mapped into the discrete grid of the fixed image with the help of the `ResampleImageFilter` class and the `LinearInterpolateImageFunction`. 

---

**Figure 5-16 Atlas construction pipeline**

Gravity centre from initialization of homologous moving image

Fixed image's grid

Rotation centre

Partial embryo view, third channel

preprocessing

initialization

transformation

Transformation Parameters

Rotation centre

Fixed image

Third channel mapped on template
Figure 5-17 Overview of the transformation step. The first column is the flowchart of the implemented program. The second column shows the ITK components used for the corresponding process. The third column shows the parameters that the user can provide.
Chapter 6

Results

The performance of the implemented framework, presented in Chapter 5, is evaluated with six experimental datasets, which belong to the same developmental stage (6 hpf) and include one dataset that depicts a whole zebrafish embryo and five datasets that depict partial views of different embryos. These input data are presented in section 6.1. The chapter continues with illustrating the output of each framework’s step separately. In section 6.2, firstly, the effects of preprocessing and addition are shown on slices of the data. Then, the four different combinations composed of the similarity measures and the optimization algorithms in the registration step are compared in terms of performance. Moreover the effect of the addition methods on the outcome of the registration is evaluated. Finally, in section 6.3, a specific combination of a similarity measure and optimization algorithm, that presents a coherent and sufficient performance, is chosen to map the third channels of the partial views on the template. The parameters’ values chosen for each step, in order to get the results presented, can be found in Appendix F.

6.1. Input data

The framework’s input data are a subset of the experimental data that have been acquired as described in section 3.4. One template and five partial views, belonging to the shield stage (6.00 hpf), were chosen to evaluate the implemented framework developed in the present thesis. It is reminded that the template view depicts a whole embryo and is composed of two channels, one containing the nuclei and the other the gsc’s expression pattern, while each partial view depicts part of the embryo and has not only the above channels but also an additional one containing the expression of another gene of interest. The characteristics of the input data are given in Table 6-1.

<table>
<thead>
<tr>
<th>Views</th>
<th>Resolution</th>
<th>Voxel size (μm)</th>
<th>3rd channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template</td>
<td>512 x 512 x 465</td>
<td>1.517 x 1.517 x 1.509</td>
<td>-</td>
</tr>
<tr>
<td>Partial view 1</td>
<td>1024 x 1024 x 141</td>
<td>0.586 x 0.586 x 1.167</td>
<td>goosecoid (gsc)</td>
</tr>
<tr>
<td>Partial view 2</td>
<td>1024 x 1024 x 130</td>
<td>0.581 x 0.581 x 1.167</td>
<td>goosecoid (gsc)</td>
</tr>
<tr>
<td>Partial view 3</td>
<td>1024 x 1024 x 156</td>
<td>0.586 x 0.586 x 1.167</td>
<td>snail</td>
</tr>
<tr>
<td>Partial view 4</td>
<td>1024 x 1024 x 157</td>
<td>0.586 x 0.586 x 1.167</td>
<td>chordin (chd)</td>
</tr>
<tr>
<td>Partial view 5</td>
<td>1024 x 1024 x 118</td>
<td>0.586 x 0.586 x 1.167</td>
<td>spadetail (spt)</td>
</tr>
</tbody>
</table>

Table 6-1 Characteristics of our input data
Chapter 6 – Results

It should be noted that two of the partial views contain again the *gsc* expression in their third channel, as they were used for the quantification process mentioned in the MORPHONET’s overall strategy. These were only used to assess the registration pipeline, as no other different gene expression is available in the third channel to use with the atlas construction pipeline. The template and the other three partial views are shown throughout Figure 6-1 to Figure 6-4. All our data are grayscale images in the range 0-255 and the colours that can be seen in the figures of our data are “introduced” by changing the colourmap in the visualization program used each time. By doing that it is easier for the reader to distinguish the different channels.

Figure 6-1 Transverse slice of the whole zebrafish embryo view in the shield stage that will be used as a template. Information is contained in two channels: (a) goosecoid gene expression channel and (b) nuclei channel.

Figure 6-2 Transverse slice of the partial view 3 of a zebrafish embryo at single cell resolution in the shield stage. Information is contained in three channels: (a) goosecoid gene expression channel, (b) nuclei channel and (c) snail gene expression channel.
6.2. Registration pipeline’s results

Firstly, the results of the registration pipeline are presented. This part of the framework is the critical one, since the atlas construction can only take place only when the registration produces satisfactory results. The outcome of each step in the registration pipeline is presented separately with the exception of the first two, the preprocessing and addition steps, which are presented together.

6.2.1. Preprocessing and addition steps

Both nuclei and gsc channels in every view were preprocessed prior to addition. The choices in preprocessing are dependent on the existence of noise, resolution, intensity distribution and computational time. For the addition, the second method (described in section 5.3.2) with a division factor of three was chosen for all data. This way the enhancement of the gsc channel in comparison to the nuclei channel is threefold. Since the physical space they
occupy is not of importance in these steps, their outcome is presented on slices, taken from our 3-D data with the help of Amide (see Appendix C).

In particular, the original channels of the template presented bright spikes in the background; hence opening was first applied. Then the channels were blurred and downsampled. Furthermore the nuclei channel was turned into a binary mask by thresholding, using a value that was chosen with respect to its intensity distribution. Closing was then applied to fill in any holes between the nuclei. Both preprocessed channels were then fed into the addition step, which produced one single, combined image. Likewise, the nuclei and gsc channels of the partial views were preprocessed and through the addition step a single combined image for each pair was acquired. The results of the above steps are illustrated in Figure 6-5 to Figure 6-10.

Figure 6-5 Transverse slices of the template embryo. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.

Figure 6-6 Transverse slices of the first partial view. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.
Figure 6-7 Transverse slices of the second partial view. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.

Figure 6-8 Transverse slices of the third partial view. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.
Chapter 6 – Results

Figure 6-9 Transverse slices of the fourth partial view. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.

Figure 6-10 Transverse slices of the fifth partial view. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.
6.2.2. Initialization step

The goal of the initialization step is to position each combined partial view to the physical space of template embryo in such a way that a successful registration can be achieved. Therefore here it is important to present the template embryo in space (Figure 6-11) as well as the relative spatial location of the other embryos, from which the partial views were acquired, before and after the initialization step (Figure 6-12 to Figure 6-16). Volume rendering was done with the help of Amira® (see Appendix C).

Figure 6-11 Template embryo in space. The gravity centres are illustrated with the spheres. The yellow sphere represents the nuclei gravity centre and the blue the gsc gravity centre. These centres correspond to the nfixed and gsccfixed points, presented in Figure 5-9, respectively.

Figure 6-12 The first partial view in space. In (a) the original location in space is presented along with the gravity centres. In (b) the relative location before and after initialization is shown along with the centres, where the nuclei gravity centre and gsc gravity centre correspond to nfixed and gsccfixed respectively, as notated in Figure 5-9. In (c) the location of the first partial view with respect to the template is presented.
Figure 6-13 The second partial view in space. In (a) the original location in space is presented along with the gravity centres. In (b) the relative location before and after initialization is shown along with the centres, where the nuclei gravity centre and gsc gravity centre correspond to nmoving and gscmoving respectively, as notated in Figure 5-9. In (c) the location of the second partial view with respect to the template is presented.

Figure 6-14 The third partial view in space. In (a) the original location in space is presented along with the gravity centres. In (b) the relative location before and after initialization is shown along with the centres, where the nuclei gravity centre and gsc gravity centre correspond to nmoving and gscmoving respectively, as notated in Figure 5-9. In (c) the location of the third partial view with respect to the template is presented.
Figure 6-15 The fourth partial view in space. In (a) the original location in space is presented along with the gravity centres. In (b) the relative location before and after initialization is shown along with the centres, where the nuclei gravity centre and gsc gravity centre correspond to nmoving and gscmoving respectively, as notated in Figure 5-9. In (c) the location of the fourth partial view with respect to the template is presented.

Figure 6-16 The fifth partial view in space. In (a) the original location in space is presented along with the gravity centres. In (b) the relative location before and after initialization is shown along with the centres, where the nuclei gravity centre and gsc gravity centre correspond to nmoving and gscmoving respectively, as notated in Figure 5-9. In (c) the location of the fifth partial view with respect to the template is presented.
6.2.3. **Registration step**

The results of this step should help in finding the answers in two important questions:

(a) What is the combination of similarity measure and optimization algorithm that results in a successful registration?

(b) How do the choices in the addition step affect the results of the registration?

**Defining the combination**

It is difficult to come up with a simple answer to the first question, primarily because the comparison between two optimization algorithms is not an easy task. To be able to decide which one performs better, both of them have to be running with optimized parameters and that means that an exhaustive search for parameter tuning for both of them has to be conducted. In our case the parameter tuning was not exhaustive. The parameters were chosen judging from the outcome each time. Then these parameter values were applied to all registration procedures and coherence in performance was evaluated.

It is important here to note that the desired transformation parameters are not known a priori, neither the desired minimum value of the similarity measure. Therefore the assessment whether the registration is successful or not relies solely on visual inspection. The results for each partial view are shown in Table 6-2 to Table 6-6, along with the final value of the similarity measure.
<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
<th>Partial view 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation Coefficient</td>
<td>Gradient descent</td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td>Differential evolution</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td>Mutual Information</td>
<td>Gradient descent</td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td>Differential evolution</td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Table 6-2 Results of the four combinations of the registration step for the first partial view.
Chapter 6 – Results

<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gradient descent</td>
</tr>
<tr>
<td>Correlation Coefficient</td>
<td>-0.4844</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mutual Information</td>
<td>-0.0527</td>
</tr>
</tbody>
</table>

Table 6-3 Results of the four combinations of the registration step for the second partial view.
<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
<th>Grad. descent</th>
<th>Diff. evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation Coefficient</td>
<td>Gradient descent</td>
<td><img src="image1.png" alt="Image" /></td>
<td>-0.6560</td>
</tr>
<tr>
<td>Mutual Information</td>
<td>Differential evolution</td>
<td><img src="image3.png" alt="Image" /></td>
<td>-0.1285</td>
</tr>
</tbody>
</table>

Table 6-4 Results of the four combinations of the registration step for the third partial view.
Table 6-5 Results of the four combinations of the registration step for the fourth partial view.

<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
<th>Partial view 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>Gradient descent</em></td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td><em>Differential evolution</em></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td>Correlation Coefficient</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mutual Information</td>
<td><img src="image3.png" alt="Image" /></td>
<td>-0.0945</td>
</tr>
<tr>
<td></td>
<td><img src="image4.png" alt="Image" /></td>
<td>-0.1161</td>
</tr>
</tbody>
</table>
# Partial view 5

<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
<th>Gradient descent</th>
<th>Differential evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation Coefficient</td>
<td></td>
<td>-0.6253</td>
<td>-0.5831</td>
</tr>
<tr>
<td>Mutual Information</td>
<td></td>
<td>-0.0128</td>
<td>-0.1289</td>
</tr>
</tbody>
</table>

Table 6-6 Results of the four combinations of the registration step for the fifth partial view.
Chapter 6 – Results

From the results shown in Tables 6-2 to 6-6 we can draw conclusions on the performance of each combination:

- The mutual information measure combined with the differential evolution optimization algorithm produced in most cases the worst results, although the final values are better (that is, lower) than the ones found with the gradient descent algorithm.
- Between the mutual information and the correlation coefficient measures we cannot compare values. However it seems that the correlation coefficient describes better our problem. This conclusion agrees with the fact that the correlation coefficient is the ideal similarity measure in a monomodal case where the images’ intensities are linearly related, which is the case we deal with.
- Finally, the correlation coefficient measure optimized by the gradient descent algorithm managed to produce satisfactory results every time in comparison to the optimization by the differential algorithm.

Determining the effect of addition

Now to define how the choices in the addition step affect the registration results across the four combinations, a partial view with little variability between the registration’s results was selected, the third partial view. Firstly we examined the effects of the choice of method and then the effects of the division factor.

To determine whether the choice of method plays a crucial role in the results, the alternative addition method (the first one in section 5.3.2) was applied, keeping the enhancement of the gsc channel threefold with respect to the nuclei channel.

It is reminded that this method does not require the nuclei channel to be turned into a binary mask. Furthermore the initialization step is not affected in any way from the addition step. The outcome of the first addition method on the template embryo and the chosen partial view are shown in Figure 6-17 and Figure 6-18.
Chapter 6 – Results

Figure 6-17 Transverse slices of the template embryo. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.

Figure 6-18 Transverse slices of the third partial view. In (a) and (b) the original channels are shown. (c) and (d) are the outcomes of the preprocessing step. In (e) the combined view of both channels is presented.

As the initialization step is independent of the addition step, the combined partial view was initialized the same way as before (Figure 6-14). Then both initialized partial view and template were given to the registration step. The results are presented in Table 6-7.
Table 6-7 Results of the four combinations of the registration step for the third partial view. Method one was used in the addition step prior to registration.

From the results shown in Table 6-7 we can deduce that by keeping the enhancement of the gsc channel threefold with respect to the nuclei channel the first addition method produces the same results as the second addition method.

From the conclusion drawn above we understand that the effect of the division factor is independent of the choice of method. Since the results of the registration step were evaluated using the second addition method, we examined the influence of the division factor using again the same method and the third partial view as before.

In the results shown in Table 6-2 to Table 6-6 the division factor was set to three (3). So, in order to determine its effects on the results of the registration step, the registration procedure with the four combinations was performed twice, once setting the division factor to one (1) and then to six (6).

Firstly, the results with the division factor set to one (1) are presented. Figure 6-19 shows the outcome of the addition step using the second addition method with a division factor of one (1).
Figure 6-19 Transverse slices of the template and partial views. In (a) and (b) and (d) and (e) the preprocessed channels of the template and the partial view are shown respectively. In (c) and (f) the combined image of the template and the partial view are presented respectively, using the second addition method with a division factor of one (1).

The combined partial view was initialized in the same way as before (Figure 6-14) and passed to the registration step along with the combined template view. The results are shown in Table 6-8.

<table>
<thead>
<tr>
<th>Partial view 3</th>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Correlation Coefficient</td>
<td>Gradient descent</td>
</tr>
<tr>
<td></td>
<td>-0.5999</td>
<td>-0.6245</td>
</tr>
<tr>
<td></td>
<td>Mutual Information</td>
<td>-0.1183</td>
</tr>
</tbody>
</table>

Table 6-8 Results of the four combinations of the registration step for the third partial view. The second addition method was used in the addition step prior to registration with the division factor set to one (1).
The same procedure was repeated setting this time the division factor to six (6). Figure 6-20 presents the outcome of the addition step and Table 6-9 the results of the registration step.

Figure 6-20 Transverse slices of the template and partial views. In (a) and (b) and (d) and (e) the preprocessed channels of the template and the partial view are shown respectively. In (c) and (f) the combined image of the template and the partial view are presented respectively, using the second addition method with a division factor of six (6).

<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>Optimization algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation Coefficient</td>
<td>Gradient descent</td>
</tr>
<tr>
<td></td>
<td>-0.6862</td>
</tr>
<tr>
<td>Mutual Information</td>
<td>-0.1315</td>
</tr>
</tbody>
</table>

Table 6-9 Results of the four combinations of the registration step for the third partial view. The second addition method was used in the addition step prior to registration with the division factor set to six (6).
Considering the results shown in Table 6-8 and Table 6-9, we come to the conclusion that the division factor, which is used by the addition step when calculating the parameter alpha (α), plays a crucial role in all the combinations in the registration step apart from the one where the correlation coefficient measure is optimized by the gradient descent algorithm.

Overall, the combination of the correlation coefficient with the gradient descent algorithm presented a coherent and sufficient performance, when the division factor in the addition step was in the [1–6] range. Therefore, the transformation parameters that were given by this combination were used to transform the third channels of the partial views.

6.3. Illustration of the atlas construction

The final results of the implemented framework are obtained by transforming the third channels of the partial views. Following the pipeline described in section 5.4, the third channels of the partial views were first smoothed and downsampled through the preprocessing step, then transformed through the initialization step to obtain the same position in space as their corresponding moving images, and finally transformed for a second time through the transformation step using the transformation parameters given from the registration step. The transformation parameters used were the ones given by the combination of the correlation coefficient with the gradient descent algorithm.

This last transformation maps the expression patterns of the genes co-stained with the gsc gene on the template. The results of the final transformation are presented in Figure 6-21.
Figure 6-21 In (a) the template embryo is shown along with the segmented gsc’s expression (yellow region). In (b), (c) and (d) the relative location of the gsc and the co-stained gene’s expressions are presented for every view before the transformation and after. Finally in (e) all the gene expressions are mapped onto the template.
Chapter 7

Conclusions

This chapter gives an overall evaluation of the implemented framework, which was described in Chapter 5 and its results were presented in Chapter 6, and furthermore, analyzes its advantages (section 7.1). As last words, the future work is discussed in section 7.2.

7.1. Discussion

In this thesis the problem of registering the gene expression patterns of partial zebrafish embryo views onto a template embryo was successfully addressed by implementing a framework that includes all the processes necessary for such a task.

The framework was evaluated with a template and five partial views. Its key points are the initial positioning of the partial views on the template prior to registration and the specification of the most suitable registration method for our problem. From the results in Chapter 6, the following general conclusions can be drawn for these crucial points:

- The implemented algorithm that deals with the initial positioning problem has proven to yield satisfactory results, since it manages to sufficiently overlap the gsc expressions of both template and partial views, while keeping the direction of the curvature of the embryo.

- The implemented algorithm that deals with the registration problem was designed so that the combinations of two similarity measures with two optimization algorithms could be evaluated in terms of coherence in performance. The correlation coefficient measure optimized by the gradient descent algorithm has proven to yield successful results, even when the influence of the addition method was examined.

The transformation given by the above combination for every partial view was then applied to map the co-stained gene’s expression on the template and the construction of a three-dimensional gene expression atlas was illustrated (Figure 6-21).

The implementation of the framework was based on the Insight Segmentation and Registration Toolkit (ITK), an open-source software library. The framework implemented has the following advantages:

- **Modularity**: Each step of the registration process was implemented as a separate module, thus limiting the complexity of the overall software. This way necessary future changes or improvements can be made only at a specific part of the process, making the framework extensible and reusable. Furthermore, by following the ITK programming principles, the “plugging” of more processing components within each step is facilitated, thereby accommodating non-anticipated cases of input data.
Chapter 7 – Conclusions

- **Configurability**: The big amount of experimental data on which the framework will be applied and their diversity in terms of noise, resolution, etc., made necessary the implementation of a framework that can be configured on a case-by-case basis. Not only each component in each processing step can be configured with different parameters, but also the selection of different components for the same task is available through command-line configuration without the need of altering the source code.

7.2. **Future work**

In the future the results of the implemented algorithm should be confirmed with more experimental data. Future work also includes the mapping of additional gene expression patterns on the template for the specific developmental stage for constructing its 3-D gene expression atlas. The extension of this procedure to other developmental stages will introduce the time dimension and lead to the construction of a spatial-temporal gene expression atlas.

Combining the above results with the results of the gene expression quantification process, which obtains quantitative measures about gene expression at cellular resolution, will complete the spatial-temporal atlas with the quantitative information of the genes’ expressions. This atlas must first be validated with known gene-regulatory interactions. Once validated, it can be employed to other studies, such as research in unknown gene-regulatory interactions and comparative studies with mutants.
Appendices

A. Glossary

This glossary contains the biological terms that are found in Chapter 2.

animal pole. location on the egg where the polar bodies emerge, corresponding to the point of fertilization in fish like zebrafish, just below where the sperm penetrates the chorion through the micropile (passageway).

animal-vegetal axis. a line passing through the animal and vegetal poles of the embryo before the end of epiboly.

anterior-posterior axis. the principal axis of the embryo, the line from head to tail.

blastoderm. cellular part of the embryo, excluding the yolk cell, derived from the blastodisc by early morphogenesis; refers particularly to the time when the cell array is sheet-like, between 30%-epiboly and the end of gastrulation.

blastodisc (-disk). dome of cytoplasm that segregates from the yolk toward the animal pole during and after the one cell-stage, and which undergoes cleavage.

blastomere. a cell arising during cleavage; the term encompasses the partially cleaved, incomplete “cells” at the blastodisc margin before they collectively form the yolk syncytial layer in the midblastula.

blastula. classically the single-layered hollow ball of cells formed by cleavage in organisms that show this developmental style; in Chapter 2 it is used to mean a stereoblastula, not hollowed-out, and as a descriptor for the period of development when the blastodisc begins to look ball-like, at the 128-cell stage through the time of onset of gastrulation.

cleavage. an early mitotic cell division occurring in the blastodisc, special in that the cell cycles are short in length, are not accompanied by cell growth during interphase, and occur synchronously or metasynchronously with other cleavages of the same number; in the staging series, the cleavage period refers to the period of development encompassing the first six zygotic cell cycles.

convergence. deep cell movement toward the dorsal side of the embryo during the gastrula and early segmentation periods.

dorsal. toward the back (the side opposite to the belly)

dorsal-ventral (dorsoventral) axis. axis passing from the back to the belly; within a sagittal plane and at right angles to the anterior-posterior axis.

ectoderm. the outer germ layer, which gives rise to epidermis, nervous system and sense organs

epiboly. the thinning and spreading of both the yolk syncytial layer (YSL) and the blastoderm over and across the yolk cell, eventually encompassing the yolk cell completely; epiboly begins at dome
Appendix A

stage, converts the blastodisc to the blastoderm, and is considered to be over when the yolk plug closes over (at 100%-epiboly).

**endoderm.** the inner germ layer, which gives rise to the gastrointestinal tract and associated structures.

**gastrula.** classically a postblastula stage in which an archenteron (primitive gut or gastrocoele) forms by invagination or involution of cells through a blastopore and when the germ layers and embryonic axis appear; the zebrafish forms neither an archenteron nor a blastopore, and here the term refers to the roughly equivalent period of development, beginning at the onset of involution (at the 50%-epiboly stage) that produces the two primary germ layers, the epiblast and hypoblast, and during which the definitive embryonic axis forms by convergence and extension movements.

**gastrulation.** morphogenesis during the gastrula period; the process by which blastoderm cells are specified and move to generate an embryo with three germ layers and anterior-posterior and dorsal-ventral polarity.

**involution.** deep cell movement at the blastoderm margin in which the deep cell multilayer (DEL) folds inward and back upon itself, producing the germ ring and its two primary germ layers, the epiblast and hypoblast.

**meroblastic (incomplete) cleavage.** cell division in which sister cells are only partially separated from one another by cytokinesis; includes all of the early cleavages, and also occurs along the blastodisc margin in the late cleavage and early blastula periods.

**mesoderm.** the middle germ layer, which gives rise to bone, muscle, connective tissue, urogenital and circulatory system.

**midblastula transition (MBT).** the time of increase in cell cycle length above the rapid rate characterizing the cleavage period, beginning at the tenth zygotic cycle; at the MBT cells also begin to divide less synchronously, and motility and zygotic transcription are first observed.

**pharyngeal (visceral) arch.** a segment of the lateral wall of the pharynx that will form jaw structures (anterior two arches) or gill structures (posterior five arches).

**pharyngula.** generally, a vertebrate embryo that has developed to the phylotypic stage; in the series used as a period name to describe the 2nd of the 3 days of embryonic development.

**somite.** undifferentiated mesodermal component of an early trunk or tail segment or metamere, derived from paraxial mesoderm.

**vegetal pole.** location on the egg opposite to the animal pole, corresponding later to the point on the yolk cell furthest from the developing blastodisc.
B. Basic concepts of Image Processing

Image processing techniques are frequently used in order to make images more suitable for a specific application. In our case, concepts and methodologies belonging to the field of digital image processing are used as a preprocessing step prior to registration. The techniques described in this section are primarily the ones that have been implemented in our work. More specifically in section B.1 it is reminded to the reader how a grayscale image is represented, in section B.2 image sampling is presented, image enhancement approaches are discussed in section B.3 and finally the basic morphological operators are given in section B.4. Their analysis is based on 2-D images. However the same principles apply to 3-D volumetric data.

B.1. Grayscale image

In image processing a grayscale image (2-D) can be represented by a function \( D_1 \rightarrow I \), where \( D_1 \) is a subset of \( Z^2 \) (\( Z \) denotes the set of real integers) and \( L = \{ l_{\text{min}}, \ldots, l_{\text{max}} \} \) is an ordered set of gray levels. If the image format being used supports 8bit/pixel, then each pixel can be one of the 256 different gray values which vary from 0 (black) to 255 (white).

![Pixel coordinates used to represent digital images](image)

B.2. Image Sampling

The Shannon sampling theorem states that if the function is sampled at a rate equal to or greater than twice its highest frequency, it is possible to recover completely the original function from its samples. If the function is undersampled, then aliasing corrupts the sampled image. The corruption is in the form of additional frequency components being introduced into the sampled function (aliased frequencies). The sampling rate in images is the number of samples (in all dimensions) taken per unit distance. The principal approach for reducing the aliasing effects on an image is to reduce its high-frequency components by blurring the image prior to sampling.

The process of shrinking an image can be viewed as undersampling. Therefore to reduce possible aliasing effects, it is a good idea to blur an image slightly before shrinking it.
Appendix B

B.3. Image Enhancement

Image enhancement approaches fall into two broad categories: frequency domain methods and spatial domain methods. Frequency domain processing techniques are based on modifying the Fourier transform of an image. Spatial domain methods are procedures that operate directly on the pixels. The rest of the section is focused on spatial domain enhancement. Spatial domain processes will be denoted by the expression

\[ P(i, j) = T[I(i, j)] \]  

where \( I(i, j) \) is the input image, \( P(i, j) \) is the processed image, and \( T \) is an operator on \( I(i, j) \), defined over some neighborhood of \( (i, j) \). In addition, \( T \) can operate on a set of input images. A note that should be made here for the sake of clarity is that the operator \( T \) is just an intensity mapping and should not be confused with \( T \) mentioned in Chapter 4, which is a spatial transformation.

B.3.1. Point Processing

The simplest form of \( T \) is when the neighborhood is of size 1x1 (the size of a single pixel). In this case, \( P \) depends only on the value of \( I \) at \( (i, j) \), and \( T \) becomes a gray-level (also called an intensity or mapping) transformation function of the form

\[ s = T(r) \]  

where \( r \) and \( s \) are variables denoting, respectively, the gray level of \( I(i, j) \), and \( P(i, j) \) at any point \( (i, j) \). Three basic types of \( T \) exist: linear, logarithmic, and power-law functions.

When \( T \) operates on a set of input images pixel by pixel, arithmetic or logic operations can be performed. Some examples of this category are discussed in the following paragraphs.

(i) Contrast stretching

The idea behind contrast stretching is to increase the dynamic range of the gray levels in the image being processed (e.g. to span the full range of pixel values that the image type concerned allows.). Figure B-2 (a) shows a piecewise linear transformation used for contrast stretching. The locations of points \((r_1, s_1)\) and \((r_2, s_2)\) control the shape of the transformation function. Each pixel is scaled by

\[ T(r) = \frac{s_2 - s_1}{r_2 - r_1} (r - r_1) + s_1 \]  

![Figure B-2](image-url)  

Figure B-2 Graylevel transformation functions for contrast enhancement. (a) Form of transformation function for contrast stretching (b) form of transformation function for global thresholding. Extracted from [40].
In general, \( r_1 \leq r_2 \) and \( s_1 \leq s_2 \) is assumed so the order of gray levels is preserved. If the image format being used supports 8bit/pixel, the full range of pixel values is 0 to 255. If the desired stretching is to span the full range, then \( s_1 \) is selected 0 and \( s_2 \) 255. Usually \( r_1 \) and \( r_2 \) are the lowest and highest pixel values present in the input image. However if there is a single outlying pixel with either very high or very low value, this can lead to very unrepresentative scaling. This can be solved by choosing other values for \( r_1 \) and \( r_2 \). Contrast stretching can also be used when converting from one image type to another (floating point value to 8bit integer pixel value).

(ii) Thresholding

In the above equation if \( r_1 = r_2 \), \( s_1 = 0 \) and \( s_2 = L-1 \) (where \( L \) is the number of gray-levels in the output image), the transformation becomes a thresholding function that creates a binary image. Thresholding is usually performed when segmentation is needed on the basis of the different intensities of the regions that we are interested in.

Thresholding, in its simplest form (global thresholding), is described by

\[
s = \begin{cases} 
L - 1, & r > k \\
0, & r \leq k 
\end{cases}
\]  

Each pixel in the input image is compared to a given threshold \( k \) (intensity threshold). If the pixel's value is below the threshold \( k \), the pixel is set to 0 (black) in the output; otherwise it is set to \( L - 1 \) (white). The key objective is to merely create a binary image, so \( L - 1 \) can just be equal to 1. In addition the relationship black-white can be reversed.

(iii) Arithmetic/Logic Operations

Arithmetic/logic operations involving images are performed on a pixel-by-pixel basis between two or more images (this excludes the logic operation NOT, which is performed on a single image) and have many uses as preliminary steps in more complex operations. Image arithmetic is the implementation of standard arithmetic operations, such as addition, subtraction, multiplication and division.

In the case of adding two images (2-D), the addition is carried out pixel-wise, that is the first pixel of the resulting image is the sum of the first pixel in one image and the first pixel in the other. One of the input images may be a constant value, when a constant offset to an image needs to be added. Generally the output pixel values are given by

\[
P(i, j) = I_1(i, j) + I_2(i, j)
\]  

where \( I_1 \) and \( I_2 \) are the two input images and \( P \) is the output image. If the image format being used supports 8bit/pixel then it is easy for the result of the addition to be greater than the maximum allowed pixel value. The effect of this depends upon implementation. The overflowing pixel values might just be set to the maximum allowed value (saturation) or might wrap around from zero again.

The same apply for multiplication. The output pixel values are given by

\[
P(i, j) = I_1(i, j) \times I_2(i, j)
\]  

One of the input images may be a constant value. If that constant value is greater than one, then the image brightens, if it is lower than one, it darkens. Multiplication by a constant is useful prior to other image arithmetic in order to prevent pixel values going out of range.
In the case of blending the value of each pixel in the output image is a linear combination of the corresponding pixel values in the input images. The coefficients define the ratio by which to scale each image before combining them. These proportions are applied so that the output pixel values do not exceed the maximum pixel value. Here again, one of the input images may be a constant value.

\[ P(i,j) = aI_1(i,j) \times (1-a)I_2(i,j) \]  
(B-7)

The factor \( \alpha \) can be a constant factor for all pixels or can be determined for each pixel separately using a mask.

### B.3.2. Spatial Filtering

As mentioned in section B.3, \( T \) is an operator on \( I(i,j) \) defined over some neighbourhood of \( (i,j) \). When the neighbourhood is of larger size than 1x1, the technique is referred to as mask processing or filtering.

The principal approach in defining a neighbourhood about a point \( (i,j) \) is to use a square (or other neighbourhood shapes) subimage area centred at \( (i,j) \). The centre of the subimage is moved from pixel to pixel and the operator \( T \) is applied at each location \( (i,j) \) to yield the output, \( P \), at that location. The operator \( T \) works with the pixel values in the area of the image spanned by the neighbourhood and the corresponding values of a subimage that has the same dimensions as the neighbourhood. The subimage is called a filter, mask, kernel, template or window. The values in a filter subimage are referred to as coefficients and determine the nature of the process. Filters can be linear or nonlinear.

The response of a linear filter of size \( m \times n \) at each point is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask:

\[ P(i,j) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)I(i+s,j+t) \]  
(B-8)

where \( a = \frac{m-1}{2} \) and \( b = \frac{n-1}{2} \). To generate a complete filtered image of size \( M \times N \) this equation must be applied for \( i = 0,1,\ldots,M-1 \) and \( j = 0,1,\ldots,N-1 \). The process of linear filtering given in equation B-8 is similar to a frequency domain concept called convolution.

Equation B-8 can be simplified when interest lies on the response \( R \) of the filter:

\[ R = \sum_{i=1}^{mn} w_i z_i \]  
(B-9)

where \( w_i \) are the masks coefficients and \( z_i \) are the values of the image pixels and \( mn \) is the total number of coefficients in the mask.

![Figure B-3 Representation of a general 3x3 spatial filter mask. Extracted from [40].](image)
Nonlinear spatial filters also operate on neighbourhoods, and the mechanics of sliding a mask past an image are the same as in the case of linear filters. For example, noise reduction can be achieved effectively with a nonlinear filter whose basic function is to compute the median gray-level value in the neighbourhood in which the filter is located. Computation of the median is a nonlinear operation.

A subset of the linear spatial filters is the smoothing filters, which are used for blurring and for noise reduction. Blurring is used in preprocessing steps, such as removal of small details from an image prior to (large) object extraction, and bridging of small gaps in lines or curves. Noise reduction can be accomplished by blurring with both linear and nonlinear filtering.

In the case of a linear smoothing spatial filter, the output (response) is simply the average of the pixels contained in the neighbourhood of the filter mask. These filters sometimes are called averaging filters (or low-pass filters).

Smoothing filters work by replacing the value of every pixel in an image by the average of the gray levels in the neighbourhood defined by the filter mask. The mask’s size and shape establish the relative size and shape of the objects that will be blended with the background.

The simplest case is that of the arithmetic mean filter. A mean filter of size \( m \times n \) centred at \((i,j)\) computes the average value of the image in the area defined by the mask by:

\[
P(i,j) = \frac{1}{mn} \sum_{s=a}^{a} \sum_{t=b}^{b} I(s,t)
\]

A mean filter smoothes local variations in an image, thus blurs, and as a result noise is reduced. Figure B-4 shows an example of a 3x3 mean filter. Substituting the coefficients of the mask into equation B-9 results in:

\[
R = \frac{1}{9} \sum_{t=1}^{9} z_t
\]

which is the average intensity of the pixels in the 3x3 neighbourhood defined by the mask.

The mask in Figure B-4 computes a weighted average, where pixels are multiplied by different coefficients (weights).

Nonlinear spatial filters are the order-statistics filters, whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the centre pixel with the value determined by the ranking result. A typical example of this category is the median filter, which replaces the value of a pixel by the median of the intensities in the neighbourhood of that pixel. Median filters are particularly effective in the presence of impulse noise, also called salt-and-pepper noise.

![Figure B-4 Two 3x3 smoothing (averaging) filter masks.](image)
**B.4. Mathematical Morphology**

The basic morphological operators are dilation and erosion. Their combination constitutes new operations—opening and closing. Both opening and closing, tend to smooth the image. The structuring element $b(i, j)$ is itself a subimage function.

(i) **Dilation**

Gray-scale dilation of $I(i, j)$ by $b$, denoted $I \oplus b$, is defined as

$$\left( I \oplus b \right)(s, t) = \max\{I(s - i, t - j) + b(i, j) | (s - i, t - j) \in D_I; (i, j) \in D_b\}$$  \hspace{1cm} (B-12)

where $D_I$ and $D_b$ are the domains of $I$ and $b$, respectively. The general effect of performing dilation on a gray-scale image is twofold: (1) if all values of the structuring element are positive, the output image tends to be brighter than the input and (2) dark details are either reduced or eliminated, depending on how their values and shapes relate to the structuring element.

(ii) **Erosion**

Gray-scale erosion, denoted $I \ominus b$, is defined as

$$\left( I \ominus b \right)(s, t) = \min\{I(s + i, t + j) - b(i, j) | (s + i, t + j) \in D_I; (i, j) \in D_b\}$$  \hspace{1cm} (B-13)

where $D_I$ and $D_b$ are the domains of $f$ and $b$, respectively. The general effect is twofold: (1) if all elements of the structuring element are positive, the output image tends to be darker than the input image and (2) the effect of the bright details in the input image, that are smaller in an area than the structuring element is reduced, with the degree of reduction being determined by the gray values surrounding the bright detail and by the shape and amplitude values of the structuring element itself.

(iii) **Opening**

Gray-scale opening, denoted $I \circ b$, is defined as

$$I \circ b = \left( I \ominus b \right) \oplus b$$  \hspace{1cm} (B-14)

Opening operations usually are applied to remove small (with respect to the size of the structuring element) light details, while leaving the overall gray levels and larger brighter features relatively undisturbed.

(iv) **Closing**

Gray-scale closing, denoted $I \bullet b$, is defined as

$$I \bullet b = \left( I \oplus b \right) \ominus b$$  \hspace{1cm} (B-15)

Closing is generally used to remove dark details from an image, while leaving bright features relatively undisturbed.
C. Tools

This thesis benefited from the use of the Insight Segmentation and Registration Toolkit (ITK), an open-source software library (BSD licensed) developed as an initiative of the U.S. National Library of Medicine and available at www.itk.org. ITK is an object-oriented software library for image processing, segmentation, and registration, implemented in C++. ITK’s mathematical library is based on the VXL/VNL software package. VXL (Vision * Library) is a collection of C++ libraries, useful for building computer vision applications and maintained as free, open source software under SourceForge, http://sourceforge.net/. VNL is the VXL’s numerics library, which provides convenient C++ interfaces for linear algebra, polynomials, transformations and linear and non-linear optimization.

ITK is cross-platform, using a build environment called CMake to manage the compilation process in a platform-independent way. CMake was partially funded by the U.S. National Library of Medicine as part of the Insight Toolkit project. CMake is an open-source system, which generates native makefiles and workspaces that can be used in the compiler environment of the user’s choice and it is freely available at www.cmake.org. The development environment used was Microsoft Visual Studio 2008.

The ITK does not provide any interface for visualization operations of the results. For this purpose two viewing tools were used, AMIDE and Amira. AMIDE is a free tool for viewing, analyzing, and registering volumetric medical imaging data sets, available at http://amide.sourceforge.net/. Amira® is a 3-D visualization and modelling system with an interactive 3-D viewer. Originally, Amira® was designed and developed by the Visualization and Data Analysis Group at Zuse Institute Berlin (ZIB). Today it is available as a commercial product and distributed by the companies Visage Imaging, Berlin, and VSG - Visualization Sciences Group, France. While Amira® is distributed by Visage Imaging, VSG distributes a derivative of Amira® under the name Avizo®.

Complete installation guides for the above tools can be found in their official websites. The operating system used in the present thesis is Microsoft Windows XP Professional (service pack 3). The implemented algorithms were performed by a 2.40 GHz Intel® Core™2. All of the above have been summarized in Table C-1.

| Tools used for developing the algorithms: | • Insight Segmentation and Registration Toolkit  
• Vision * Library  
• CMake  
• Microsoft Visual Studio 2008. |
| Tools used for visualization: | • AMIDE  
• Amira |

Table C-1. Overview of the tools used in this Master Thesis.
D. Image in ITK

Bearing in mind that the registration process is done in physical space and not in the grid space, the representation of an image in ITK has to be taken into account, so as to understand the geometrical concepts it involves. In particular the physical spacing between pixels/voxels and the image’s position in space with respect to some world coordinate system are extremely important. Figure D-1 illustrates the main geometrical concepts associated with an image in the Insight Toolkit in two dimensions, where circles are used to represent the centre of pixels.

![Figure D-1: Geometrical concepts associated with the ITK image. Extracted from [36].](image)

The value of the pixel is assumed to exist as a Dirac Delta Function located at the pixel centre. Physical spacing is measured between the pixel centres and can be different along each dimension. The image origin is associated with the coordinates of the first pixel in the image (with index (0, 0)) with respect to an arbitrary reference system in space. A pixel is considered to be the rectangular region surrounding the pixel centre holding the data value.

For example, the pixel of index \( I = (2, 3) \) in the image in Figure D-1 with origin \( O = (60.0, 70.0) \) and pixel spacing \( S = (20.0, 30.0) \) corresponds to the spatial (physical) position

\[
P[I] = I[I] \times S[I] + O[I]
\]

leading to \( P = (2 \times 20.0 + 60.0, 3 \times 30.0 + 70.0) = (100.0, 160.0) \).
E. User’s Manual

This section provides the user with the essential information in order to compile or/and execute the implemented programs. In the following paragraphs an overview of the framework is given (E.1), where in every step the name of the corresponding program has taken its place, and for each program separately the parameters are presented (E.2 - E.6). The parameters’ values used to obtain the results shown in Chapter 6 are presented in Appendix F.

E.1. Basics

In order to compile the five implemented programs, the user must have installed ITK, CMake and Microsoft Visual Studio (or a development environment supported by CMake). For the execution of the compiled programs only the installation of ITK is necessary.

For the compilation only the source file of the program is needed. The source file contains the necessary code and a text file, called CMakelists.txt, which provides the information needed for CMake to generate the workspace. In our case a Microsoft Visual Studio project was generated. Once this is done, the project has to be built so that the executable file can be created.

From the command line the user can run each executable file, providing the necessary arguments. If the name of the program is typed along with -h, or -- help, then a list of the required parameters appears. The required parameters for each program are presented in the following paragraphs.

Figure E-1 Overview of the framework showing the name of the implemented program for each step.
Appendix E

E.2. **Preprocessing Program**

The executable file is under the name Preprocess.exe. By typing its name and -h or --help in the command line the following appears on the screen:

```
USAGE:
Preprocess.exe  [--alreadybin]  [--open <integer>]  [--close <integer>]
    [--threshold <integer (0-255)>]  [--binary]  [--resampler]
    [--factorZ <double>]  [--factorY <double>]  [--factorX
double>]  [--downsample]  [--smooth]  [--radiusZ
<integer>]  [--radiusY <integer>]  [--radiusX <integer>]
    [-o <string>]  [--]  [-v]  [-h]  <string>
```

This brief usage is followed by a list, where each argument is explained. By typing –v or --version the version of the program is displayed. The argument -- or --ignore_rest can be used to ignore the rest of the labelled arguments following it. Table E-1 provides a description of the arguments and their values.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Value</th>
<th>Default value</th>
<th>Req.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input image</td>
<td>The name of the image to be processed. It must be a 3-D image with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>–</td>
<td>✓</td>
</tr>
<tr>
<td>-o / --output</td>
<td>The name of the output image. It is a string with extension '.mhd'.</td>
<td>string</td>
<td>'preprocessed. mhd'</td>
<td></td>
</tr>
<tr>
<td>--open</td>
<td>The structuring element’s size if opening is desired to be performed on the input image.</td>
<td>integer</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>--smooth</td>
<td>A flag that if set, smoothing is applied.</td>
<td>boolean</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>--radiusX</td>
<td>Kernel size used for smoothing along x.</td>
<td>integer</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>--radiusY</td>
<td>Kernel size used for smoothing along y.</td>
<td>integer</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>--radiusZ</td>
<td>Kernel size used for smoothing along z.</td>
<td>integer</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>--downsample</td>
<td>A flag that if set, downsampling is performed using the ShrinkFilter.</td>
<td>boolean</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>--resampler</td>
<td>A flag that if set, the ResamplerFilter (with trilinear interpolation and identity transform) is used, instead of the ShrinkFilter</td>
<td>boolean</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>--factorX</td>
<td>Downsampling factor along x.</td>
<td>double</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>--factorY</td>
<td>Downsampling factor along y.</td>
<td>double</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>--factorZ</td>
<td>Downsampling factor along z.</td>
<td>double</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>--binary</td>
<td>A flag that if set, the image is turned to binary.</td>
<td>boolean</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>--threshold</td>
<td>Lower threshold value used in converting the image from grayscale to binary</td>
<td>integer</td>
<td>(0-255)</td>
<td></td>
</tr>
<tr>
<td>--close</td>
<td>The structuring element’s size if closing is desired.</td>
<td>integer</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>--alreadybin</td>
<td>A flag that has to be set only if the input image is already a binary and closing is desired to be applied.</td>
<td>boolean</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Table E-1 Parameters for the Preprocessing Program.

At runtime information are printed on the screen regarding the process of the program.
Appendix E

E.3. Addition Program

The executable file is under the name Add.exe. By typing its name and -h or -- help in the command line the following appears on the screen:

**USAGE:**

```
Add.exe [--thresnuclei <integer (0-255)>] [--thresgene <integer (0-255)>] [--division <integer>] [--method <integer (1,2)>] [-o <string>] [--] [-v] [-h] <string> <string>
```

This brief usage is followed by a list, where each argument is explained. By typing --v or --version the version of the program is displayed. The argument -- or -ignore_rest can be used when we want to ignore the rest of the labelled arguments following it. Table E-2 provides a description of the arguments and their values.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Value</th>
<th>Default value</th>
<th>Req.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nuclei image name</td>
<td>The name of the nuclei channel image with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Gene image name</td>
<td>The name of the gsc channel image with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>-o / --output</td>
<td>The name of the output image. It is a string with extension '.mhd'.</td>
<td>string</td>
<td>'added.mhd'</td>
<td></td>
</tr>
<tr>
<td>--method</td>
<td>1. Method One (both channels are grayscale) 2. Method Two (nuclei channel is turned into a binary mask)</td>
<td>integer</td>
<td>(1,2)</td>
<td>✓</td>
</tr>
<tr>
<td>--thresgene</td>
<td>Threshold to calculate the average intensity of voxels showing the gene expression.</td>
<td>integer</td>
<td>(0-255)</td>
<td>✓</td>
</tr>
<tr>
<td>--thresnuclei</td>
<td>Threshold to calculate the average intensity of voxels showing the nuclei (only for method one).</td>
<td>integer</td>
<td>(0-255)</td>
<td>✓ Method One</td>
</tr>
<tr>
<td>--division</td>
<td>The constant by which the weight is divided</td>
<td>integer</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Table E-2 Parameters for the Addition Program.

Attention should be paid to the order in which the input images are written. The name of the nuclei channel should be written before the name of the gsc channel. At runtime information about the process are printed on the screen.

E.4. Initialization Program

The executable file is under the name Initialize.exe. By typing its name and -h or -- help in the command line the following appears on the screen:

**USAGE:**

```
Initialize.exe [--threspartialgene <integer (0-255)>]
[--thretemplategene <integer (0-255)>] [-o <string>] [--] [-v] [-h] <string> <string> <string> <string> <string> <string>
```

91
This brief usage is followed by a list, where each argument is explained. By typing --v or --version the version of the program is displayed. The argument -- or --ignore_rest can be used when we want to ignore the rest of the labelled arguments following it. Table E-3 provides a description of the arguments and their values.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Value</th>
<th>Default value</th>
<th>Req.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed image</td>
<td>The name of the fixed image with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Moving image</td>
<td>The name of the moving image with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Template nuclei channel mask</td>
<td>The name of the template nuclei channel mask with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Template gsc channel</td>
<td>The name of the preprocessed template gsc channel with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Partial nuclei channel mask</td>
<td>The name of the partial nuclei channel mask with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Partial gsc channel</td>
<td>The name of the preprocessed partial gsc channel with extension '.mhd'. In the same directory the file '.raw' must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>--o / --output</td>
<td>The name of the output image. It is a string with extension '.mhd'.</td>
<td>string</td>
<td>'Initialized_moving.mhd'</td>
<td></td>
</tr>
<tr>
<td>--threstemplategene</td>
<td>Threshold for calculating the fixed image's gene gravity centre</td>
<td>integer</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--threspartialgene</td>
<td>Threshold for calculating the moving image's gene gravity centre</td>
<td>integer</td>
<td>0</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table E-3 Parameters for the Initialization Program.

Attention should be paid to the order in which the input images are written. The name of the fixed image should be written first, followed by the name of the moving image, the name of the template nuclei mask, the name of the template gsc channel, the name of the partial nuclei mask and lastly the name of the partial gsc channel.

Among the information printed on the screen at runtime is the calculated gravity centre for the goosecoid’s expression of the template view. These coordinates are important for the registration program, as they are passed as the rotation centre.

**E.5. Registration Program**

The executable file is under the name Register.exe. By typing its name and -h or --help in the command line the following appears on the screen:
This brief usage is followed by a list, where each argument is explained. By typing \texttt{--version} the version of the program is displayed. The argument \texttt{--ignore_rest} can be used when we want to ignore the rest of the labelled arguments following it. Table E-4 provides a description of the arguments and their values.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Value</th>
<th>Default value</th>
<th>Req.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed image</td>
<td>The name of the fixed image with extension <code>.mhd</code>. In the same directory the file <code>.raw</code> must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Initialized moving image</td>
<td>The name of the initialized moving image with extension <code>.mhd</code>. In the same directory the file <code>.raw</code> must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>-o / --output</td>
<td>The name of the output image. It is a string with extension <code>.mhd</code>.</td>
<td>string</td>
<td><code>registered.mhd</code></td>
<td></td>
</tr>
<tr>
<td>--centre</td>
<td>A flag that when set, the rotation centre should be given by the user. Otherwise it's the (0,0,0).</td>
<td>boolean</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>--rotX</td>
<td>Physical coordinate in x of the rotation centre</td>
<td>double</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>--rotY</td>
<td>Physical coordinate in y of the rotation centre</td>
<td>double</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>--rotZ</td>
<td>Physical coordinate in z of the rotation centre</td>
<td>double</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>--metric</td>
<td>The similarity measure to be used : 1.Normalized Correlation</td>
<td>integer</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>2.MattesMutualInformation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--optimizer</td>
<td>The optimization algorithm to be used: 1.Regular Step Gradient (R.S.G), 2.Differential Evolution (DE) (BEST/2/BIN strategy)</td>
<td>integer</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>-i / --niter</td>
<td>Number of iterations</td>
<td>integer</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>--maxstep</td>
<td>Maximum step size for the R.S.G</td>
<td>double</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>--minstep</td>
<td>Minimum step size for the R.S.G</td>
<td>double</td>
<td>0.01</td>
<td></td>
</tr>
<tr>
<td>--relax</td>
<td>Relaxation factor for the R.S.G</td>
<td>double</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>--tscale</td>
<td>Translation parameters’ scale for the R.S.G</td>
<td>double</td>
<td>$10^{-4}$</td>
<td></td>
</tr>
<tr>
<td>--rscale</td>
<td>Rotation parameters’ scale for the R.S.G.</td>
<td>double</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>--pop</td>
<td>Population size for the D.E.</td>
<td>integer</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>--cross</td>
<td>Crossover probability for the D.E.</td>
<td>double</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>--weight</td>
<td>Weight factor for the D.E.</td>
<td>double</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>--rdelta</td>
<td>Rotation range for the D.E.</td>
<td>double</td>
<td>$10^{-5}$</td>
<td></td>
</tr>
<tr>
<td>--tdelta</td>
<td>Translation range for the D.E.</td>
<td>double</td>
<td>$10^{-3}$</td>
<td></td>
</tr>
</tbody>
</table>

Table E-4 Parameters for the Registration Program.
When passing values, the fixed image’s name should be written before that of the moving. Among the information printed on the screen at runtime are the rotation centre, the final transformation parameters and the corresponding similarity measure’s value.

E.6. Transformation Program

The executable file is under the name Transform.exe. By typing its name and -h or -- help in the command line the following appears on the screen:

```
USAGE:
```

This brief usage is followed by a list, where each argument is explained. By typing --v or --version the version of the program is displayed. The argument -- or --ignore_rest can be used when we want to ignore the rest of the labelled arguments following it. Table E-5 provides a description of the arguments and their values.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Value</th>
<th>Default value</th>
<th>Req.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed image</td>
<td>The name of the fixed image with extension ‘.mhd’. In the same directory the file ‘.raw’ must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Partial third channel</td>
<td>The name of the third channel with extension ‘.mhd’. In the same directory the file ‘.raw’ must be included.</td>
<td>string</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>-o /--output</td>
<td>The name of the output image. It is a string with extension ‘.mhd’.</td>
<td>string</td>
<td>‘final.mhd’</td>
<td></td>
</tr>
<tr>
<td>--centerX</td>
<td>Physical coordinate in x of the rotation centre</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--centerY</td>
<td>Physical coordinate in y of the rotation centre</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--centerZ</td>
<td>Physical coordinate in z of the rotation centre</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--versorX</td>
<td>The first of the six transformation parameters given as output from the registration program.</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--versorY</td>
<td>The second of the six transformation parameters given as output from the registration program.</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--versorZ</td>
<td>The third of the six transformation parameters given as output from the registration program.</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--transX</td>
<td>The fourth of the six transformation parameters given as output from the registration program.</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--transY</td>
<td>The fifth of the six transformation parameters given as output from the registration program.</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>--transZ</td>
<td>The last of the six transformation parameters given as output from the registration program.</td>
<td>double</td>
<td>0</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table E-5 Parameters for the Transformation Program.

When passing values, the name of the fixed image should be written before that of the partial third channel. The rotation centre and the transformation parameters are taken from the execution of the registration program.
F. Parameters' values

In this section the parameters’ values for each program, which were used to get the results shown in Chapter 6, are presented, so their numerical range can be understood and the reproducibility of the results can be ensured with the same data. Each program’s parameters have already been presented in Appendix E. The images’ names are not included in the given values.

F.1. Preprocessing and Addition Programs

The arguments of the two executables files, the Preprocess.exe and the Add.exe, have already been presented in Tables E-1 and E-2 respectively. The values of these parameters that were used to get the results shown in section 6.2.1 (Figure 6-5 to Figure 6-10) are given in Tables F-1 and F-2.

<table>
<thead>
<tr>
<th>Preprocess.exe Parameters</th>
<th>Template</th>
<th>Partial view 1</th>
<th>Partial view 2</th>
<th>Partial view 3</th>
<th>Partial view 4</th>
<th>Partial view 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>--open</td>
<td>3 3 - - -</td>
<td>3 3 - - -</td>
<td>3 3 - - -</td>
<td>3 3 - - -</td>
<td>3 3 - - -</td>
<td>3 3 - - -</td>
</tr>
<tr>
<td>--smooth</td>
<td>true true true true true true true true true true true true true</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--radiusX</td>
<td>3 3 3 3 3 3 3 3 3 3 3 3 3 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--radiusY</td>
<td>3 3 3 3 3 3 3 3 3 3 3 3 3 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--downsample</td>
<td>true true true true true true true true true true true true true</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--resampler</td>
<td>false false false false false false false false false false false false</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--factorX</td>
<td>4 4 2 2 2 2 2 2 2 2 2 2 2 2 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--factorY</td>
<td>4 4 2 2 2 2 2 2 2 2 2 2 2 2 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--factorZ</td>
<td>4 4 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--binary</td>
<td>true false true false true false true false true false true false true false true</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--threshold</td>
<td>10 10 10 10 10 10 10 10 10 10 10 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--close</td>
<td>3 - 3 - 3 - 3 - 3 - 3 - 3 -</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--alreadybin</td>
<td>false false false false false false false false false false false false false false</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table F-1 Parameters’ values used when running the executable file Preprocess.exe for Figures 6-5 to 6-10

<table>
<thead>
<tr>
<th>Add.exe Parameters</th>
<th>Template</th>
<th>Partial view 1</th>
<th>Partial view 2</th>
<th>Partial view 3</th>
<th>Partial view 4</th>
<th>Partial view 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>--method</td>
<td>2 2 2 2 2 2 2 2 2 2 2 2 2 2 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--thresgene</td>
<td>70 100 40 120 80 70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--division</td>
<td>3 3 3 3 3 3 3 3 3 3 3 3 3 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table F-2 Parameters’ values used when running the executable file Add.exe for Figures 6-5 to 6-10
For the results shown in Figure 6-17 and Figure 6-18, the values in Tables F-3 and F-4 were used.

<table>
<thead>
<tr>
<th>Preprocess.exe Parameters</th>
<th>Template embryo</th>
<th>Partial view 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>--open</td>
<td>3</td>
<td>-</td>
</tr>
<tr>
<td>--smooth</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>--radiusX</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>--radiusY</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>--radiusZ</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>--downsample</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>--resample</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>--factorX</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>--factorY</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>--factorZ</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>--binary</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>--threshold</td>
<td>-</td>
<td>10</td>
</tr>
<tr>
<td>--close</td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>--alreadybin</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Table F-3 Parameters’ values used in running the executable Preprocess.exe for Figures 6-17 and 6-18

<table>
<thead>
<tr>
<th>Add.exe Parameters</th>
<th>Template embryo</th>
<th>Partial view 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>--method</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>--thresgene</td>
<td>70</td>
<td>120</td>
</tr>
<tr>
<td>--thresnuclei</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>--division</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Table F-4 Parameters’ values used when running the executable Add.exe for Figures 6-17 and 6-18

For the results shown in Figure 6-19 and Figure 6-20, firstly preprocessing was performed with the values given in Table F-1. For the addition program the values that were used are presented in Tables F-5 and F-6 respectively.

<table>
<thead>
<tr>
<th>Add.exe Parameters</th>
<th>Template embryo</th>
<th>Partial view 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>--method</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>--thresgene</td>
<td>70</td>
<td>120</td>
</tr>
<tr>
<td>--division</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table F-5 Parameters’ values used when running the executable file Add.exe for Figure 6-19
Appendix F

<table>
<thead>
<tr>
<th>Add.exe Parameters</th>
<th>Template embryo</th>
<th>Partial view 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>--method</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>--thresgene</td>
<td>70</td>
<td>120</td>
</tr>
<tr>
<td>--division</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

Table F-6 Parameters’ values used when running the executable file Add.exe for Figure 6-20

F.2. Initialization Program

The initialization program’s parameters, Initialize.exe, have already been presented in Table E-3. Table F-7 gives the values that were used to get the results shown in section 6.2.2 (Figure 6-12 to Figure 6-16).

<table>
<thead>
<tr>
<th>Initialize.exe Parameters</th>
<th>Partial view 1</th>
<th>Partial view 2</th>
<th>Partial view 3</th>
<th>Partial view 4</th>
<th>Partial view 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>--thretemplatogene</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>--threpartialgene</td>
<td>100</td>
<td>40</td>
<td>120</td>
<td>80</td>
<td>70</td>
</tr>
</tbody>
</table>

Table F-7 Parameters’ values used when running the executable file Initialize.exe

The output of this program is not only the initialized moving image, but also the gravity centre of the gsc's expression in the template image that is used in the registration program as the rotation centre.

The values presented in Table F-7, regarding the third partial view, were used for all the initialization steps prior to registration.
F.3. Registration Program

As mentioned above, the coordinates of the rotation centre were taken from the initialization program. In Table F-8 the values used for the results in Tables 6-2 to 6-9 are presented.

<table>
<thead>
<tr>
<th>Register.exe Parameters</th>
<th>For all partial views</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Combination 1</td>
</tr>
<tr>
<td>--centre</td>
<td>true</td>
</tr>
<tr>
<td>--rotX</td>
<td>154.467</td>
</tr>
<tr>
<td>--rotY</td>
<td>222.227</td>
</tr>
<tr>
<td>--rotZ</td>
<td>179.83</td>
</tr>
<tr>
<td>--metric</td>
<td>1</td>
</tr>
<tr>
<td>--optimizer</td>
<td>1</td>
</tr>
<tr>
<td>--maxstep</td>
<td>1.0</td>
</tr>
<tr>
<td>--minstep</td>
<td>0.01</td>
</tr>
<tr>
<td>--relax</td>
<td>0.5</td>
</tr>
<tr>
<td>--tscale</td>
<td>1.0/10000.0</td>
</tr>
<tr>
<td>--rscale</td>
<td>1.0</td>
</tr>
<tr>
<td>--pop</td>
<td>-</td>
</tr>
<tr>
<td>--cross</td>
<td>-</td>
</tr>
<tr>
<td>--weight</td>
<td>-</td>
</tr>
<tr>
<td>--rdelta</td>
<td>-</td>
</tr>
<tr>
<td>--tdelta</td>
<td>-</td>
</tr>
<tr>
<td>--niter</td>
<td>100</td>
</tr>
</tbody>
</table>

Table F-8 Parameters’ values used when running the executable file Register.exe

The registration program gives as output not only the registered image, but also the transformation parameters that were applied on that image.

F.4. Transformation Program

As mentioned in the framework’s overview the third channels of the partial views are first preprocessed and then transformed to obtain the same initial position as the their corresponding moving images. The Preprocess.exe program was just used to downsample the channels and the Initialize.exe was run with the same parameters’ values as in Table F-7 (apart from the fact that now the image to be transformed is the third channel instead of the moving image). Finally the images are given one by one to the transformation program, with the values given in Table F-9, yielding Figure 6-21. The rotation centre was taken from the initialization program and the versor and translation components from the registration program that was run under the combination of the Correlation Coefficient measure and the Gradient Descent Algorithm.
### Table F-9 Parameters’ values used when running the executable file Transform.exe

<table>
<thead>
<tr>
<th>Transform.exe Parameters</th>
<th>Partial view 3</th>
<th>Partial view 4</th>
<th>Partial view 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>--centerX</td>
<td>154.467</td>
<td>154.467</td>
<td>154.467</td>
</tr>
<tr>
<td>--centerY</td>
<td>222.227</td>
<td>222.227</td>
<td>222.227</td>
</tr>
<tr>
<td>--centerZ</td>
<td>179.83</td>
<td>179.83</td>
<td>179.83</td>
</tr>
<tr>
<td>--versorX</td>
<td>0.204854</td>
<td>-0.01397</td>
<td>-0.479892</td>
</tr>
<tr>
<td>--versorY</td>
<td>-0.0988401</td>
<td>-0.221808</td>
<td>-0.229993</td>
</tr>
<tr>
<td>--versorZ</td>
<td>0.156107</td>
<td>0.156655</td>
<td>0.26755</td>
</tr>
<tr>
<td>--transX</td>
<td>-14.8894</td>
<td>-4.47066</td>
<td>-2.43941</td>
</tr>
<tr>
<td>--transY</td>
<td>-6.10306</td>
<td>3.20003</td>
<td>7.35626</td>
</tr>
<tr>
<td>--transZ</td>
<td>-5.61534</td>
<td>7.80313</td>
<td>4.2131</td>
</tr>
</tbody>
</table>
Bibliography


   Available at: http://www.zf-models.org/ (Accessed 18/9/2009)

[8] ZFIN:The Zebrafish Model Organism Database.
   Available at: http://zfin.org/ (Accessed 18/9/2009)


Available at: http://www.cs.rpi.edu/academics/courses/spring04/imagereg/ (Accessed 18/9/2009)


