University of Potsdam
Institute of Computer Science

Zuse Institute Berlin

Master's thesis

Registration-based Tracking of Regions Defined by Level Sets of Time-dependent Scalar Fields

Raphael Badel

March 3, 2021

Priv.-Doz. Dr. Henning Bordihn (University of Potsdam)
Dr. Daniel Baum (Zuse Institute Berlin)

Kurzzusammenfassung

Diese Arbeit befasst sich mit dem weiten Forschungsfeld der Merkmalsverfolgung in zeitabhängigen Daten. Für eine Verfolgung verschiedenster Merkmale in skalaren Daten, die in Form von diskreten Zeitschritten vorliegen, existiert bereits eine Vielzahl von Lösungen. Auf dem Gebiet der Meteorologie geben die aktuellen Niederschlagsdaten der COSMO-REA2 Reanalysen Anlass zur Untersuchung der zeitlichen Entwicklung von konvektivem Niederschlag. Dabei sollen Niederschlagszellen im zeitlichen Verlauf verfolgt werden. Eine vorangegangene Studie ergab, dass eine Merkmalsverfolgung, die auf dem häufig verwendeten Kriterium der räumlichen Überlappung basiert, keine adäquaten Ergebnisse für die Reanalysedaten liefert. Basierend auf neuartigen Anforderungen wird im Verlauf dieser Arbeit ein neuer Ansatz zur Verfolgung von Regionen in zeitabhängigen Skalarfeldern entwickelt und in einer prototypischen Studie auf Beispieldatensätze der COSMO-REA2 Reanalysen angewendet. Neben der konkreten Motivation, die entwickelte Methode auch für nachfolgende Untersuchungen von Niederschlag nutzbar zu machen, ist der Ansatz so konzipiert, dass er für beliebige skalare Größen anwendbar ist, die konzeptionell durch uniforme Gitter beliebiger Dimension diskretisiert sein können. Auf Basis einer detaillierten Beschreibung der verwendeten Methoden wird in dieser Arbeit eine neuartige Lösung zur Merkmalsverfolgung vorgestellt, im Rahmen derer Korrespondenzen anhand von Bildregistrierung zwischen diskreten Zeitschritten identifiziert werden und unabhängig von den eigentlichen Merkmalen vorausberechnet werden können. In Kombination mit einer hierarchischen Segmentierung von Merkmalen durch die Wasserscheidentransformation ermöglicht die vorgestellte Implementation eine effiziente Lösung für die Verfolgung von Merkmalen sowie eine schnelle Generierung von Ergebnissen.

Abstract

This thesis is concerned with the wide field of feature tracking in time-dependent data. Many solutions already exist for the tracking of various features in scalar fields that are given as discrete time steps. In the field of meteorology, recently published precipitation data of the COSMO-REA2 reanalysis system gave rise to the analysis of precipitation at a convective scale for which a tracking of precipitation cells over time is desired. A previous study indicated that a tracking based on the widely used overlap criterion does not perform well for the reanalysis data. Based on a novel set of requirements, a new approach to the tracking of regions in time-dependent scalar fields is developed in the course of this thesis and applied in a prototypical study to example datasets of the COSMO-REA2 system. Despite the concrete motivation of using the developed method for subsequent studies of precipitation, the tracking approach is designed to be applicable for arbitrary scalar quantities that can conceptually be given on uniform grids of arbitrary dimensions. Based on a detailed description of the utilized methods, this thesis presents a novel tracking solution whose correspondence identification is based on image registration of successive time steps in combination with a hierarchical watershed segmentation by means of which features are extracted. The proposed implementation allows for an efficient generation of tracking results under the premise that the registration-based correspondence information has been precomputed.

Acknowledgements

This research has been funded by Deutsche Forschungsgemeinschaft (DFG) through grant CRC 1114 “Scaling Cascades in Complex Systems”, Projects A01 “Coupling a multiscale stochastic precipitation model to large scale atmospheric flow dynamics” and C06 “Multiscale structure of atmospheric vortices”.

I would like to express my gratitude to my supervisor Dr. Daniel Baum for his invaluable advice and continuous support during my graduate research at Zuse Institute Berlin. Likewise, I wish to express my appreciation to Hon.-Prof. Hans-Christian Hege for his guidance and vital ideas that gave rise to this thesis in the first place. I would like to thank my supervisor from the University of Potsdam, Priv.-Doz. Dr. Henning Bordin, who supported my thesis proposal right from the beginning.

In addition, I would like to acknowledge the cooperation partners of the Institute of Meteorology of the Free University of Berlin, Univ.-Prof. Dr. Henning Rust, Priv.-Doz. Dr. Peter Névir and Dr. Annette Müller, whose contributions and comments have been essential for achieving my goal.

Finally, I must express my very profound gratitude to my wife who took care of so many things allowing me to focus on this work. Really, I am thankful to everyone who helped me on the way to my graduation!

Declaration of Authorship

I hereby declare that this thesis has been composed by me and is based on my own work, unless stated otherwise. No other person's work has been used without due acknowledgement in this thesis. All references and verbatim extracts have been quoted, and all sources of information have been specifically acknowledged.

Date: Signature:

Contents

1 Introduction

In many scientific fields, physical quantities are studied that both depend on spatial location and time. In practice, such quantities are either measured or the outcome of a mathematical modelling process e.g. by means of a computer simulation. In both cases, the results are usually available in form of datasets that store data values over a discretized spatial domain that is recorded for separate time steps. While a simulation can theoretically produce results in any resolution, real life scenarios in which data is actually measured are most likely subject to conditions that constrain the maximal resolution.

This thesis focusses on scalar data that typically represents the intensity of a physical quantity such as e.g. temperature, pressure or phase. Some criteria then define individual features that can be extracted for each time step. This allows for the analysis of spatial properties of features such as e.g. their size and position in each time step. However, to account for the temporal dimension of a time-dependent quantity, an obvious application is to analyse how features and their properties develop over time. For this purpose, correspondences between features in successive time steps is needed to obtain feature tracks. On the one hand, the spatio-temporal perspective gives rise to temporal properties, such as e.g. the life time or length of feature tracks. On the other hand, the spatio-temporal dynamic of features belonging to one track can be examined. This includes the joining and splitting of features over time. This thesis deals with the problem of feature tracking which includes three mentioned subproblems: (i) feature definition and extraction, (ii) correspondence identification, and (iii) the representation of feature tracks. For all three components, different strategies have been developed and implemented, resulting in various tracking solutions.

Precipitation In meteorology, the study of precipitation as a time-dependent quantity is of particular interest. Its temporal development is influenced by many aspects, such as e.g. other physical processes as well as the underlying topographic terrain. A common way to examine precipitation is based on cells which are defined as spatially connected areas that exceed a certain level of precipitation. Precipitation events can be grouped according to different spatial and temporal scales. When examining the influence of precipitation on extreme weather scenarios such as floods, two forms are of particular interest [106]. *Convective precipitation* occurs from convective clouds in form of rather small precipitation cells that only exist for a short amount of time. The precipitation intensity is typically much higher compared to *stratiform precipitation*. This form results from the interferences of large-scale high- and low-pressure areas and the concomitant hot and cold fronts. Those evolve over a long period of time during which the intensity can be characterized as steady but at a lower level. *Mesoscale precipitation* describes a mix of both of the aforenamed forms [39, 90]. Floods can be the result of an extreme manifestation of either convective or stratiform precipitation. To be able to better foresee such scenarios and anticipate changing trends of precipitation due to the changing climate, stochastic precipitation models are employed [90]. They are based on statistical properties of precipitation such as the frequency and intensity measured or simulated at individual locations [156]. It is the subject of current research to integrate time-aggregate statistics of precipitation cells and a detailed analysis of their spatio-temporal dynamic to be able to improve these models in the future. Comprehensive statistics on the number, area, mean and maximal intensities of precipitation cells as well as comparing these properties to the life-time cycle of cells yields further insights into precipitation as a time-dependent quantity [39].

Datasets from the reanalysis system COSMO-REA2 [149] have recently become publicly available including precipitation as a meteorological quantity. The underlying COSMO model offers a spatial resolution of approximately 2 km giving rise to the analysis of precipitation at a convective scale. Moreover, the system offers an hourly temporal resolution spanning the years from 2007 to 2013. Consequently, the new datasets provide the possibility to gain new information

about quantitative and qualitative characteristics of precipitation at different scales. Such information can be obtained by the tracking of precipitation cells.

1.1 Related Work

As of now, there exists a multitude of tracking solutions that have either been applied for specific applications, e.g. in the field of meteorology, or were developed for universal purposes. It should be noted that the following overview only comprises those publications which, from a methodological point of view, can be applied to any kind of scalar data, even though they might have been motivated by a concrete application. In order to approach the large amount of tracking solution, they are divided into two groups.

The first group is based on the strategy of extracting features in every time step and subsequently identifying correspondences between these features in successive time steps. On the one hand, the notion of features can be purely geometrical like connected regions that are defined by a threshold [2, 33, 61, 80, 100, 110, 115, 116, 128, 132, 154]. On the other hand, the definition of the very same kind of features can be motivated by a topological point of view namely Morse theory. In Morse theory, the threshold is understood as a level in the context of level sets. The connected regions mentioned before are then considered connected components of the sub- or superlevel set whose common boundary is the level set [17, 88, 126, 146, 155]. The topological point of view also offers the possibility to define features as connected components of the level set itself [135] which are known as contours. Further topological features include critical points [134], persistence pairs [136], nodes and arcs of (topological) graph structures [103] or the Morse-Smale complex [82]. Since Morse theory can also be thought of as a theoretically sound foundation of the watershed transformation [31], which originated in the context of digital pictures, features defined by watershed regions [81] are also considered topological. Defining features by means of Morse theory provides elegant and efficient solutions when it comes to the extraction of features in actual datasets.

The abovementioned features are mostly regional ones. Moreover, non-regional features such as contours, critical points or graph entities can also be associated with regions which is enabled by the topological point of view. Therefore, the identification of correspondences between features in this group is predominantly based on geometrical properties of or between regions. This comprises distance proximity and/or attribute similarity regarding the size of features as well as other geometric attributes [33, 61, 80–82, 110, 115, 128, 154]. A particular geometric property that is widely used is the spatial overlap between regional features of successive time steps [2, 17, 81, 88, 100, 116, 126, 132, 134, 135, 146, 154, 155]. In addition to overlap, regions are matched based on their intensity distributions [126]. After identifying an initial set of correspondences, thresholds for the used criteria are usually applied to filter out matches that are unlikely. Alternatively, solutions are proposed that minimize a cost function that includes the distance between centroids as well as the difference in size [33] or the distance between centroids in combination with spatial overlap [81]. Corresponding persistence pairs are identified based on the assignment of pairs according to the Wasserstein distance between persistence diagrams [136]. Nodes and arcs of graph structures that represent the topological relations among connected components of either the sub- or superlevel set for each time step are considered with respect to their position within the graph. Correspondences between graph entities are then identified based on graph transformations [103].

In the majority of publications, the representation of feature tracks is described as directed acyclic graphs (DAGs). Vertices represent the features in each time steps and edges symbolize correspondence between features.

In the second group of tracking solutions, time-dependent data are not regarded as separate time steps that are processed independently of each other. Instead, time is considered an additional dimension of a spatio-temporal domain. This perspective gives rise to the definition of contours defined in 4D space-time [16, 59, 60, 150]. As a consequence, features in each time step can be extracted from slicing the space-time contour [59, 60]. The resulting contours in each time step can also serve as the domain for the definition of more

detailed features such as Morse-Smale complexes on these contours [16, 150]. In both cases, the underlying contours in 4D space-time determine correspondence implicitly.

The representation of feature tracks is once more based on DAGs. The temporal coordinates of vertices are either 'snapped' to the original discrete time steps [59, 60] or resemble the assumption of continuous time [16, 150]. A representation of the latter is gained from considering time as a Morse function over each of the space-time contours and computing the respective Reeb graphs.

All tracking solutions that have been proposed in the context of meteorology, fall into the first group. To improve the correspondence identification, additional information about the estimated motion between time steps is injected into the tracking approach. The level of detail at which the motion is estimated varies strongly across different approaches. In the prominent SCIT algorithm, Johnson et al. [61] use the already identified tracks in past time steps to predict the position of each feature in the next time step. Weusthoff & Hauf [154] compute a single motion vector based on the cross correlation between features of successive time steps to estimate the average motion and predict the motion of each feature. Peleg & Morin [110] attempt to compute a more robust average motion by means of a number of slight rotations of the time steps considered in the cross correlation analysis. Moseley et al. [100] compute a coarse set of motion vectors representing the average motion in local sub-regions of the domain by means of downsampling. A similar approach was used by Kyznarová & Novák [80] but for multiple sampling stages. The information is gathered to generate a dense motion field. Finally, Valsangkar et al. [146] use optical flow [56] to estimate the detailed motion of features.

1.2 Scope of Thesis

Common to all existing approaches is that the information being used for correspondence identification is restricted to the features that ought to correspond. This thesis presents a new approach to tracking that does not rely on correspondences between features but computes detailed information about

how one time step in its entirety transitions into the next one. In turn, this information is the basis for the identification of correspondences between features. The approach is motivated by physical quantities such as precipitation, the temporal development of which is influenced by interactions with other physical processes. Even if these processes are known, data describing those processes cannot be assumed to be available. However, the (unknown) interactions generally take place at different scales and determine how precipitation as a whole develops over time. Therefore, the identification of correspondence should be determined by all information of two successive time steps. More precisely, the intensities of two time steps are used for a fine-grained and dense motion estimation that describes the transformation from one time step to the next one. The fundamental assumption underlying this thesis is that the estimated motion appropriately represents the temporal development of the quantity at all scales such that a tracking based on that universal information produces adequate results. Since the motion estimation between time steps is carried out independently of the actual definition or extraction of features, the correspondence information can be precomputed.

Furthermore, an independent correspondence identification is also freed from implicit assumptions that are part of criteria such as shape similarity, distance proximity or even overlap. Shape similarity assumes that a feature's shape does not change significantly from one time step to the next one. Distance proximity assumes that no 'false' features appear within a radius equal to the velocity of the feature in question. Overlap entails even more assumptions than distance proximity since the shape and extent of features in close proximity can lead to wrong correspondences as well. From a temporal perspective, all three criteria assume that the temporal resolution is sufficient, such that potential disadvantages do not take effect. In the case of the precipitation data of the COSMO-REA2 system, the temporal resolution of one hour is relatively low compared to other data sources that are typically consulted when analyzing precipitation. Moreover, a prior study by Fischer [39] approaching a tracking solution for the same datasets struggled with the fact that there are in general many small features in close proximity.

As mentioned before, in the field of meteorology, many tracking solutions already exist that consider motion estimation in addition to the aforementioned criteria with the particular aim to improve the tracking results. However, the level of detail at which motion is estimated is rather coarse. The approach that stands out is the work of Valsangkar et al. [146] in which optical flow [56] is employed to compute a dense motion estimation for each feature evaluating the underlying intensities. The approach proposed in this thesis is similar in the sense that dense motion is estimated as well. However, in contrast to Valsangkar et al. [146], the motion is estimated for the entire domain based on all intensities. Moreover, instead of relying on optical flow whose motion estimation is by definition limited to short distances an image registration method is employed.

When it comes to the definition of features, a topological approach based on level sets seems reasonable since it enables an efficient extraction of features for different levels. This generally benefits an explorative analysis of the temporal development of precipitation which is typically characterized by inspecting the tracking results while varying the level. At first, a definition by means of level sets, more precisely, connected components of the superlevel set, seems appropriate. In two dimensions, this meets the common understanding of what precipitation cells are. However, the tracking approach should support a feature extraction of precipitation cells with respect to the different precipitation scales. Large frontal cells that exhibit a fairly homegeneous distribution of average intensities exist alongside small convective cells with high maximal intensities. In particular, precipitation at a mesoscale is characterized by both kind of cells such that convective cells are nested in frontal cells. Regarding the definition and extraction of features, a more appropriate representation of features might result from splitting precipitation cells into sub-cells each of which resembles a dominant local maximum. Therefore, features are defined as by means of a hierarchical watershed segmentation which allows to control what is considered locally dominant. For the definition of storm cells, a similar approach but with far-reaching extensions was proposed by Lakshmanan et al. [81].

The representation of feature tracks should be such that properties of precipitation cells like their number, size, mean and maximal intensities in each time step as well as their lifetime and distance travelled can be accessed easily. The spatio-temporal dynamic should be represented such that join and split events that might occur during the life-time of a feature track can be detected.

The purpose of this thesis is to develop a tracking approach for time-dependent scalar fields with an exemplary focus on the tracking of precipitation cells. The question should be answered under which methodological conditions and to which extent regarding the aforementioned requirements precipitation cells can be tracked in the available COSMO-REA2 datasets. The outcomes of this thesis should enable a subsequent and systematical analysis of precipitation obtained from the tracking of precipitation cells carried out by domain experts. At the same time, the tracking approach developed here should be universal since, in the long run, it should also be used for other meteorologic datasets that might share additional requirements. This also concerns quantities that are resolved on a three-dimensional domain in each time step. Hence, the theoretical results of this thesis are not restricted to 2D+time but applicable to nD+time. In particular, the following contributions are made in this thesis:

- Correspondence identification among features that builds upon a dense registration-based motion estimation.
- Definition of a spatio-temporal digital space that allows for feature extraction with built-in correspondence information.
- Parametrization for the registation of precipitation data that enables adequate tracking results in a prototypical and qualitative study.

2 Time-dependent Data

For the course of this thesis, time-dependent data is given as finite sets of scalar values over a discretized two-dimensional spatial domain that is recorded for separate time steps.

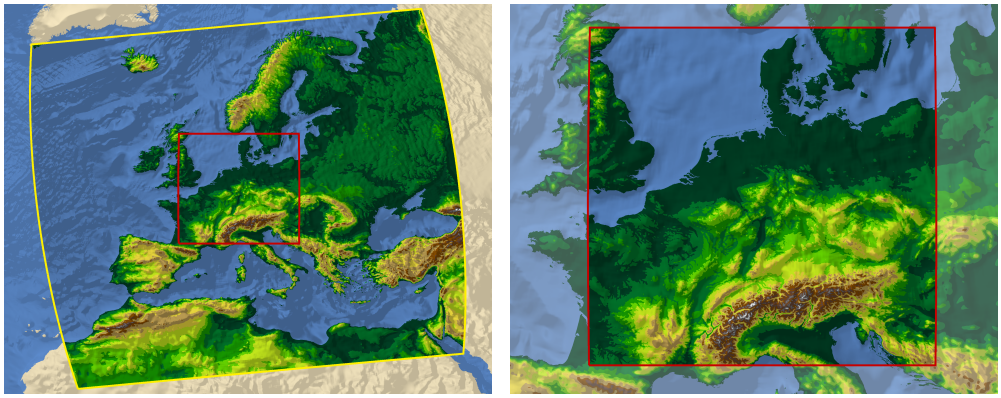
2.1 Precipitation Data

Precipitation is a real-valued non-negative meteorological scalar quantity that – for the course of this thesis – is available as one variable among others from an atmospheric reanalysis system called COSMO-REA2.

Meteorological reanalyses generate physical consistent and realistic state estimates of the atmospheric system for a past time span ranging from years to decades. The states of an atmospheric system are determined by a multitude of numerical variables. A reanalysis provides a synthesis of heterogeneous observational systems and various model simulations using a physical numerical weather prediction (NWP) model in combination with a data assimilation scheme. The output of the model are spatiotemporal meteorological fields in discretized form according to a spatial and temporal resolution. Due to its physical formulation, the model is able to estimate states between observations in a physically consistent fashion. In this way, the model is kept as close to the observed atmospheric states as possible [4, 13, 149].

The regional reanalysis system COSMO-REA6 for Europe is based on the NWP model COSMO by the German Meteorological Service (DWD)[1] and features a spatial resolution of 0.055°, which is approximately 6 km [13]. Recently, the regional reanalysis system COSMOA-REA2 was developed by the Hans

[1] Deutscher Wetterdienst



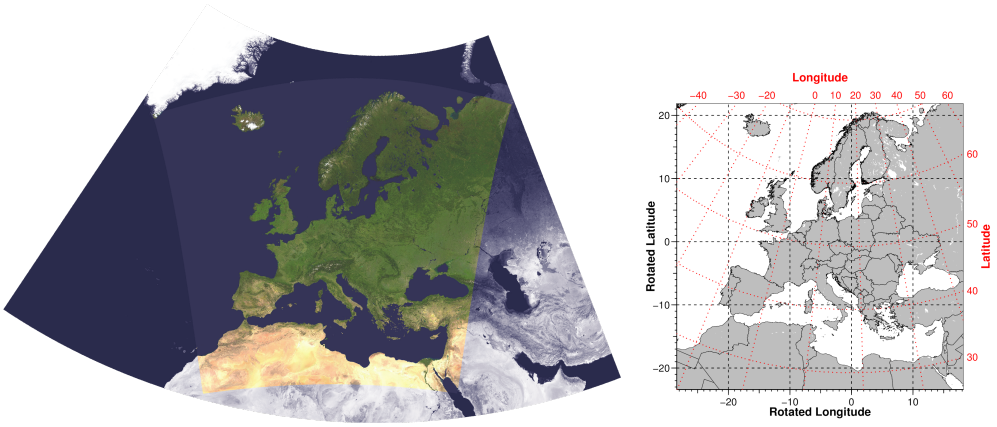
(a) (b)

Figure 1: The yellow border in (a) highlights the spatial domain of COSMO-REA6. In comparison, the domain of COSMO-REA2 is framed by the red border in (a) and (b). Images reproduced from Wahl et. al [149].

Ertel Centre for Weather Research (HErZ)[2] for Central Europe. Due to a higher spatial resolution of 0.018°, which is about 2 km, COSMO-REA2 enables an improved representation of precipitation at a convective-scale [149]. Figure 1 shows the different domains of COSMO-REA6 and -REA2. The COSMO-REA2 system spans the years from 2007 to 2013 with the years 2014 to 2016 being currently in production. A subset of often requested parameters is publicly available provided by the Meteorological Institute of the University of Bonn.[3]

The reanalysis is carried out over a subdomain of the earth's surface. Instead of the earth's intrinsic coordinate system of latitudes and longitudes, the reanalysis makes use of spherical coordinates with a rotated pole. The position of the pole is chosen such that the equator runs through the middle of the subdomain in order to minimize the convergence of the meridians. Under the assumption of a small subdomain, hence a negligible impact of the curvature of the earth's surface, the domain can be parametrized by a Cartesian coordinate system [12, 13]. Figure 2 illustrates this situation for the COSMO-REA6 model.

[2] Hans-Ertel-Zentrum für Wetterforschung
[3] `https://reanalysis.meteo.uni-bonn.de/?Download_Data___COSMO-REA2`



(a) (b)

Figure 2: The highlighted area in (a) shows the spatial domain of COSMO-REA6. In (b), the domain is shown in a rotated Cartesian coordinate system with the unrotated spherical coordinates in red. Images adapted from Bollmeyer [12] and Bollmeyer et al. [13].

2.2 COSMO Grid

The spatial domain is discretized by a two-dimensional grid consisting of vertices and edges. While vertices specify points in two-dimensional Euclidean space, edges can be thought of as straight line segments each of which connects two vertices. The grid has a regular structure and partitions the domain into rectangular cells. When all cells are equal in size, the grid is called a *uniform grid* (see Figure 3b) because it has uniform spacings between the vertices in each respective dimension [83]. The dotted black lines in Figure 2b can be regarded as the edges of a grid that is exemplary for reanalysis systems that use the COSMO model. The grid of the COSMO-REA2 system has a spatial resolution of 0.018° in both dimensions [149]. For each hourly[4] recording of the domain, the vertices of the grid are associated with a precipitation value enabling a discretization of the meteorological field in space in time.

[4]There exists a temporal discretization of precipitation at a 15-minute resolution as well. However, this data is not directly available to the public without further ado.

3 Methodology

In this chapter, the methodological background is provided that is necessary for the development of the tracking approach with respect to the requirements identified in the introduction. Throughout this chapter, the set of the first k non-negative integers $\{x \in \mathbb{Z} \mid 0 \leq x < k\}$ is denoted as $\mathbb{N}_k$.

3.1 Graph Theory

A simple, undirected graph is defined by the pair $G = (V, E)$ with vertex set V and edge set $E \subset [V]^2$. Edges are 2-element subsets of V and determine the adjacency between two vertices. The vertex set of a graph G is referred to as $V(G)$, its edge set as $E(G)$. The number of vertices of a graph G is denoted as $|G|$ with $|G| = |V(G)|$ [14, 32, 50].
For the course of this thesis, G is assumed to be finite with $|G| = k$.

An alternative specification of the edge set E is by a binary adjacency relation ε on V that is irreflexive and symmetric. In contrast to the edge set $E \subset [V]^2$, $\varepsilon = \{(u, v) \mid u, v \in V; u \neq v\}$ is a subset of the Cartesian product $V \times V$ and each 2-element subset $\{u, v\} \in E$ corresponds to two ordered pairs $(u, v), (v, u) \in \varepsilon$. If $(u, v) \in \varepsilon$, u is said to be adjacent *to* v and v is said to be adjacent *from* u [55]. When the edges of a graph are specified as ordered pairs, a graph is called a directed graph (or digraph). A simple undirected graph G, whose edge set is specified by an irreflexive and symmetric adjacency relation, is referred to as a simple symmetric digraph [50, 105].

Cartesian Product of Graphs

The Cartesian product of two graphs $G \square H$ is defined as a graph on $V(G) \times V(H)$, i.e. the set $\{(g, h) \mid g \in V(G), h \in V(H)\}$ [46, 50, 58, 125]. The edge set $E(G \square H)$ is defined as

$$E(G \square H) = (E(G) \times V(H)) \cup (V(G) \times E(H))$$

or alternatively as all pairs $[(g_1, h_1), (g_2, h_2)]$ of vertices with $[g_1, g_2] \in E(G)$ and $h_1 = h_2$ or $[h_1, h_2] \in E(H)$ and $g_1 = g_2$ [58].

Since the Cartesian product of graphs is commutative and associative [46, 58, 125], the product of n graphs can be written as $G_1 \square G_2 \square ... \square G_n$ and is a graph on n-tuples $(v_1, v_2, ..., v_n)$ where $v_i \in V(G_i)$. Two n-tuples, $(u_1, u_2, ..., u_n)$ and $(v_1, v_2, ..., v_n)$, are adjacent if there exists an index j such that $\{u_j, v_j\} \in E(G_j)$ and $u_i = v_i$ for $i \neq j$.

Path Graphs

Path graphs are a family of simple, undirected and 1-connected graphs on m vertices and with $m - 1$ edges denoted as P_m for $m > 1$ [46]. In distinction from trees, two vertices of a path graph are of degree one while the other $m - 2$ vertices are of degree two. In other words, a path graph can be drawn so that all of its vertices and edges lie on a single straight line [46]. Due to the linearity, $V(P_m)$ can be considered a sequence $\langle v_i \rangle_{i \in \mathbb{N}_m}$ ordered in such a way that

$$E(P_m) = \{\{v_i, v_{i+1}\} \mid i \in \mathbb{N}_{m-1}\}.$$

Formally, $\langle v_i \rangle_{i \in \mathbb{N}_m}$ is the image of an injective function $f_v : I \to V(P_m)$, where $I = \mathbb{N}_m$, that maps elements i of the index set I to elements of the indexed set $V(P_m)$ [49].

Grid Graphs

Grid graphs are a family of simple, undirected, n-connected and n-partite graphs that are also referred to as n-dimensional k_1-k_2-...-k_n-grids [19] or k_1-k_2-...-k_n-

meshes [46]. An n-dimensional grid graph[5] is defined as the Cartesian product of n path graphs

$$G = P_{k_1} \square P_{k_2} \square ... \square P_{k_n}$$

resulting in a graph on k vertices with $k = \prod_{i=1}^{n} k_i$. As mentioned earlier, each vertex is an n-tuple, which is henceforth denoted by a bold letter. Similar to path graphs, $V(G)$ can be considered an indexed family $\langle \boldsymbol{v}_{p_1,p_2,...,p_n} \rangle_{p_i \in \mathbb{N}_{k_i}}$ ordered in such a way that

$$E(G) = \{\{\boldsymbol{v}_{p_1,p_2,...,p_n}, \boldsymbol{v}_{q_1,q_2,...,q_n}\} \mid \textstyle\sum_{i=1}^{n} |p_i - q_i| = 1\}. \tag{3.1}$$

Formally, $\langle \boldsymbol{v}_{p_1,p_2,...,p_n} \rangle_{p_i \in \mathbb{N}_{k_i}}$ is the image of an injective function $f_{\boldsymbol{v}} : I \to V(G)$, where $I = \mathbb{N}_{k_1} \times \mathbb{N}_{k_2} \times ... \times \mathbb{N}_{k_n}$, that maps *index tuples* $(p_1, p_2, ..., p_n)$ of the index set I to elements $\boldsymbol{v}$ of the indexed set $V(G)$ [49]. To avoid additional definitions, the vertex set is confused with the indexed family throughout this thesis and both are referred to as $V(G)$. Hence, in situation when the indexation of vertices is not relevant, elements of $V(G)$ are conveniently written without indicies and distinct elements are referred to with different letters, e.g. $\boldsymbol{u}, \boldsymbol{v} \in V(G)$.

Spatial Representation

In general, the perspective on graphs is abstract and they are regarded as combinatorial structures that capture the adjacency between elements of a given set. Their specification is independent of any spatial properties such as the positions of vertices or edge crossings [46].

A *spatial graph* [6] is a spatial representation of a graph in n-dimensional Euclidean space. Vertices are specified by n-dimensional coordinate vectors and edges represent arcs between two vertices. For the course of this thesis, edges of a spatial graph are assumed to be straight line segments that span the Euclidean distance between its two endpoints.

[5] Imrich et al. [58] mention *grid graphs* only for the Cartesian product of two path graphs.

3.2 Grid Data

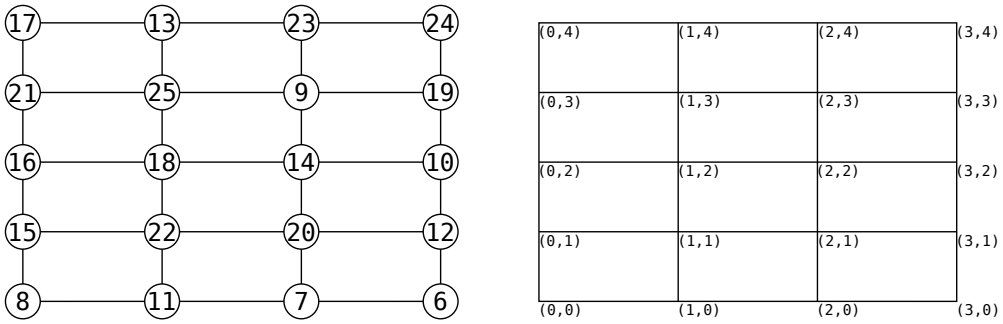
A real-valued scalar quantity that is given over a manifold can mathematically be represented by a scalar field. Let $f : \mathscr{D} \to \mathbb{R}$ be an at least continuous scalar field that assigns a real number to each point in $\mathscr{D}$. In the context of this thesis, the domain $\mathscr{D} \subset \mathbb{R}^n$ is assumed to be a simply connected subset of the n-dimensional Euclidean space, i.e. an n-manifold with boundary.

Usually, the original function f is unknown but instead represented in discretized form as grid data. For the extend of this thesis, such a discrete representation is defined by the pair[6] (U, F), i.e. an n-dimensional uniform grid U and an accompanying finite set of real data values F. A uniform grid is considered a simple, undirected, connected spatial graph whose edges are pairwise oriented either orthogonal or parallel to each other and whose vertices specify points in n-dimensional Euclidean space that are uniformly spaced in each dimension of the grid. For a discrete representation of f, the vertices form a subset of $\mathscr{D}$ and are each associated with a data value in F. A two-dimensional example of grid data is illustrated in Figure 3a.

Uniform Grid Formally, an n-dimensional uniform grid $U = (V, E)$ can be defined as a spatial representation of an n-dimensional grid graph $U = P_{k_1} \square P_{k_2} \square ... \square P_{k_n}$, i.e. the Cartesian product of n path graphs. As a result, the vertices $\boldsymbol{v}_{p_1,p_2,...,p_n} \in V$, indexed by tuples of the index set $I_U = \mathbb{N}_{k_1} \times \mathbb{N}_{k_2} \times ... \times \mathbb{N}_{k_n}$, can be parametrized by their indices $p_1, p_2, ..., p_n$ and real-valued constants $s_1, s_2, ..., s_n$ that determine the uniform spacing between vertices.

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}_{p_1,p_2,...,p_n} = \begin{pmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_n \end{pmatrix} \times \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{pmatrix} \tag{3.2}$$

[6] The pair constituting grid data is referred to as a *height graph* by Carr et al. [21] and more general as a *scalar graph* by Wenger [153].

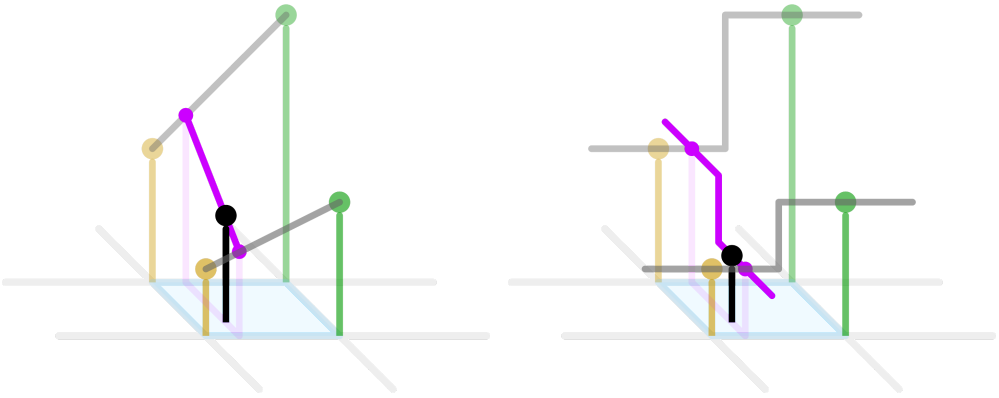


(a) (b)

Figure 3: A two-dimensional example of grid data that will recur throughout this thesis is illustrated in (a). The vertices of the uniform grid are mapped to circles and anotated by integer values. In (b), only the uniform grid with vertex indices is shown. When it comes to the visualization of a grid, vertices are usually not plotted. Instead, they can be identified indirectly as the intersection points of edges.

For the sake of simplicity, the parametrization assumes that edges are oriented either orthogonal or parallel to the Cartesian coordinate axes and that $\boldsymbol{v}_{0,0,...,0}$ is located at the origin. The number of vertices $k_1, k_2, ..., k_n$ in each dimension and their spacings $s_1, s_2, ..., s_n$ determine the resolution as well as the extent of the grid. The indexation of

Figure 3b shows an example of a uniform grid which is a spatial representation of $P_4 \,\square\, P_5$. It becomes immediately apparent how a 2-dimensional uniform grid indicates rectangular subspaces in $\mathbb{R}^2$ that are all equal in size. For an arbitrary n, the spatial arrangement of an n-dimensional uniform grid can be used to determine subspaces in $\mathbb{R}^n$ called *n-orthotopes* [29] which is the generalization of rectangles also known as n-rectangles [75] or hyperrectangles [153]. The edge lengths of n-orthotopes are determined by the uniform spacings $s_1, s_2, ..., s_n$. Although the partition of the domain into subspaces is not intrinsic to the definition of a grid or more specifically to a spatial graph, they will be referred to as *grid cells*. For this thesis, an intuitive notion of those cells is sufficient and a formal definition is omitted.



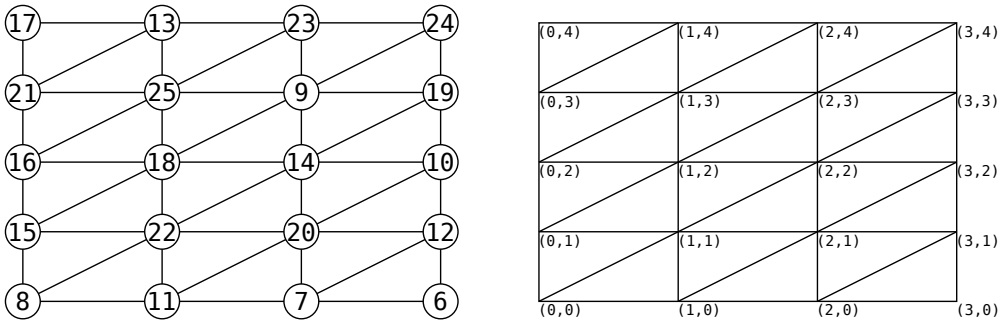
(a) Bilinear (b) Nearest-Neighbour

Figure 4: Multivariate interpolation on a cell of a 2-dimensional uniform grid. The position of the black bar within the blue cell indicates the point being interpolated and its height corresponds to the interpolated value. Yellow and green bars represent vertices and their height corresponds to the associated data values. Illustration adapted from CMG Lee under the CC BY-SA 4.0 license.

A data value associated with a vertex $\boldsymbol{v}_{p_1,p_2,\ldots,p_n} \in V$ is denoted as $f_{p_1,p_2,\ldots,p_n}$, hence $F = \{f_{p_1,p_2,\ldots,p_n} \mid (p_1, p_2, \ldots, p_n) \in I_U\}$. For a spatial representation of a grid graph with vertices parametrized as in Equation 3.2, each index p_i references a spatial dimension in n-dimensional Euclidean space.

3.2.1 Grid Cell Interpolation

When grid data (U, F) is given, it can be thought of as a *sampling* of the unknown continuous scalar field f assuming that $f(\boldsymbol{v}_{p_1,p_2,\ldots,p_n}) = f_{p_1,p_2,\ldots,p_n}$. Since numerous functions can exhibit the same values located at the grid vertices, f is obviously not uniquely defined by grid data [153]. However, another scalar field $\tilde{f} : \mathcal{D} \to \mathbb{R}$, where $\mathcal{D} \subseteq \mathscr{D}$ is the subset of n-dimensional Euclidean space that is spanned by the grid, can be build based on multivariate interpolation on the n-orthotope grid cells [92, 153]. For $n = 2$, two cell interpolants are shown in Figure 4. When applied to all grid cells, data values can be queried not only at vertex positions but for the entire continuous domain $\mathcal{D}$.



(a) (b)

Figure 5: In (a), the same data values as in Figure 3a are given over a simplicial grid. The grid in (b) results from subdividing all rectangular cells of the original uniform grid in Figure 3b following a regular pattern.

Continuous Scalar Fields

A common approach to reconstruct a continuous scalar field is based on multilinear interpolation [5, 86, 153] denoted as $\tilde{f}_{ML}$. The term *multilinear* refers to the product of n linear interpolations for functions of n variables. For an n-orthotope cell, a multilinear interpolant is in general n-polynomial for arbitrary paths through the cell. However, it is linear along edge-aligned cross-sections. Hence, it is linear in the values to be interpolated which means that the range of the interpolant is bounded by the data values associated with the cell vertices. A multilinear interpolant for $n = 2$ is called *bilinear* (see Figure 4a) and for $n = 3$ *trilinear* [152].

Another possibility is to subdivide the grid cells into n-simplices [34, 92] for which an interpolant that is linear over the entire cell exists [152]. A certain class of subdivisions, known as the *Coxeter-Freudenthal-Kuhn triangulations* [28, 42, 78], can be applied implicitly when the original uniform grid or more specifically the vertices of each n-orthotope are processed following a regular pattern [11, 97]. A simplicial grid resulting from one particular triangulation is shown in Figure 5b.

Reconstructions that make use of linear interpolation on simplicial grids are continuous and *piecewise linear* scalar fields denoted as $\tilde{f}_{PL}$. Note, that in general the topology of $\tilde{f}_{ML}$ is different from $\tilde{f}_{PL}$ [108, 158]. This is due to

the subdivision of n-orthotopes into n-simplices being ambiguous [158]. For $n = 2$, simply flipping all diagonal edges in Figure 5b would result in a different simplicial grid. Any subdivision implies a new vertex connectivity which in turn causes some vertices of the original n-orthotope to be separated and grouped into different n-simplices. As mentioned earlier, $\tilde{f}_{ML}$ is in general n-polynomial within each cell which admits the occurrence of saddle points [108]. In case a saddle point is present, any subdivision of that cell will induce a topology that is immanent in the resulting piecewise linear n-simplices and distinct from that of the original n-orthotope [158]. If the topology of $\tilde{f}_{ML}$ should be retained, switching to a simplicial grid is not an option [108].

Discontinuous Scalar Fields

Yet another option is to build a discontinuous scalar field from a nearest-neighbour interpolant that is applied to each grid cell. The result is a piecewise constant scalar field denoted as $\tilde{f}_{NN}$. The nearest-neighbour interpolant assigns the data value associated with the nearest neighbouring vertex to the points within a grid cell. The case of $n = 2$ is illustrated in Figure 4b. Applied to all n-orthotopes of the grid, the domain $\mathcal{D}$ can be partitioned into n-dimensional intervals for which $\tilde{f}_{NN}$ is constant. Those intervals are exactly the Voronoi regions of grid vertices within which $\tilde{f}_{NN}$ takes the value associated with the corresponding vertex. A Voronoi region of a vertex is the set of all points in $\mathcal{D}$ which are at least as close to the vertex as to any other points [72]. For a uniform grid, those regions have the shape of n-orthotopes as well. While a continuous scalar field like $\tilde{f}_{ML}$ can be thought of as a *reconstruction* of the original but unknown continuous scalar field f, $\tilde{f}_{NN}$ reflects the discrete nature of grid data on its continuous domain.

3.2.2 Digital Picture

In the last section, grid data (U, F) was interpreted as a *sampling* of a continuous scalar field f. For this purpose, the n-dimensional uniform grid U was utilized for its geometric aspects that allowed for reconstructions $\tilde{f}$ by means of various cell interpolants.

A conceptually different approach is to interpret grid data as a *digital picture* [55, 70]. On the one hand, grid data can result from an actual digitization of an analogue signal: The domain of a continuous scalar field f is discretized by a uniform grid and the data values associated with vertices represent function values of f within the Voronoi regions of each vertex [55, 70, 144]. The generated grid data is eventually stored on a digital medium with finite precision [70, 109, 119]. On the other hand, when grid data is given in digital form, it can be interpreted as a digital picture even if it is a sampling of a continuous scalar field f. In this case, data values in F are simply reinterpreted from function values sampled at vertex positions to representatives of function values within Voronoi region of vertices. In resemblance to $\tilde{f}_{NN}$, a digital picture is a function $\hat{f}$ that assigns data values $f_{p_1,p_2,\ldots,p_n} \in F$ to the Voronoi regions of vertices $\boldsymbol{v}_{p_1,p_2,\ldots,p_n} \in V$ [55, 70]. However, the difference between both functions is in their conceptual backgrounds: $\tilde{f}_{NN}$ is defined as the result of a nearest neighbour interpolant applied to each grid cell indepently. It solely *appears* that vertices are conceptually associated with their Voronoi regions. On the contrary, those Voronoi regions are in fact the basic units of $\hat{f}$ [55, 70, 72].

An everyday encounter of the aforementioned Voronoi regions, that most people are well acquainted with, is when scalar data given on an equally spaced two-dimensional uniform grid is interpreted as a digital picture and visualized on a screen. The two-dimensional Voronoi regions are known as *pixels*[7]. In three dimensions, the Voronoi regions are called *voxels*[8] in analogy to pixels [41].

When grid data is interpreted as a digital picture, the grid is no longer considered a spatial graph but a graph whose vertices represent continuous subspaces of $\mathbb{R}^n$ which are the Voronoi regions of vertices $\boldsymbol{v}_{p_1,p_2,\ldots,p_n} \in V$ [55]. For an n-dimensional uniform grid, the Voronoi regions are defined independently of the construction of a Voronoi diagram. Instead, they are regarded as a set of uniform n-orthotopes, henceworth referred to as *spels*[9] [51, 144]. Spels are space-filling elements that are centered at vertices and a generalization of pixels

[7] The term *pixel* is a portmanteau of 'pix' (for picture) and 'el' (for element) [41].
[8] The term *voxel* is a portmanteau of 'vox' (for volume) and 'el' (for element) [41].
[9] The term *spel* is a portmanteau of 'space' and 'element' [144].

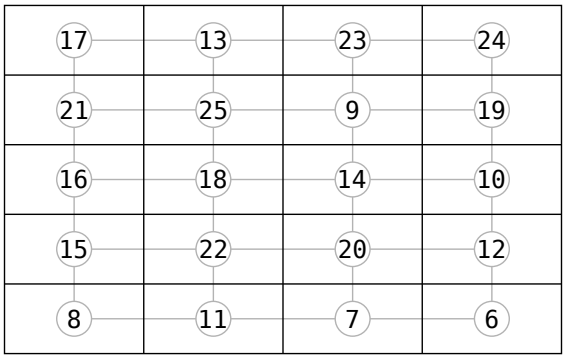


Figure 6: When the two-dimensional example of grid data is interpreted as a digital picture, vertices represent spels which are shown by the rectangles with black boundaries. In two dimensions, spels are commonly known as pixels.

and voxels for arbitrary dimensions [3, 43, 51, 84, 85, 109, 122, 137, 145]. A spel corresponding to a vertex $(v_1, v_2, ..., v_n)_{p_1,p_2,...,p_n} \in V$ is defined as the closed subset $P_{p_1,p_2,...,p_n}$ [84, 137]

$$P_{p_1,p_2,...,p_n} = \{(x_1, x_2, ..., x_n) \in \mathbb{R}^n \mid v_i - \frac{s_i}{2} \leq x_i \leq v_i + \frac{s_i}{2}\}$$

The digital counterpart of the vertex set V is the set of all spels S.

$$S = \{P_{p_1,p_2,...,p_n} \mid (p_1, p_2, ..., p_n) \in I_U\} \tag{3.3}$$

Naturally, the digital counterpart of the edge set E is the set W of 2-element subsets of S.

$$W = \{\{P_{p_1,p_2,...,p_n}, P_{q_1,q_2,...,q_n}\} \mid \textstyle\sum_{i=1}^{n} |p_i - q_i| = 1\}.$$

In summary, the digital counterpart of the n-dimensional uniform grid $U = (V, E)$ is an n-dimensional grid graph (S, W). Since each spel in S is a continuous subspace of $\mathbb{R}^n$, each edge in I defines two spels to be adjacent if and only if they have an intersection in continuous space which is $(n-1)$-dimensional [55].

As stated in Section 3.1, the edge set W can alternatively be specified as an irreflexive and symmetric adjacency relation ω which yields an orientation of those intersections [55].

$$\omega = \{(P_{p_1,p_2,\ldots,p_n}, P_{q_1,q_2,\ldots,q_n}) \mid \textstyle\sum_{i=1}^{n} |p_i - q_i| = 1\}$$

According to Herman [52–55], the pair (S, ω) defines a *digital space*. The adjacency relation ω is called *proto-adjacency* [55] and elements of ω are referred to as *surfels*[10] [51]. As stated by the same author [52, 53, 55], a digital space is essentially a connected symmetric digraph in which a vertex is adjacent *to* another vertex if and only if it is also adjacent *from* the other vertex [50]. Hence, the term 'surfel' is simply a synonym for a directed edge. It should be noted that the definition of a digital space is much more general than it is necessary for the definition of digital pictures. Basically, a digital space is a pair consisting of an arbitrary nonempty set and a symmetric adjacency relation on that set such that it is connected. Even though a digital space was introduced with a geometrical interpretation in mind, its graph-based definition can be regarded independently from any such interpretation. From a mathematical point of view $(V, V \times V)$ would be a legitimate example of a digital space as well [55].

However, the concrete digital space (S, ω) implies a geometrical interpretation indeed. The union of its spels defines a subset of n-dimensional Euclidean space, denoted as $\mathcal{D}$, to which a digital picture assigns values. Formally, a *digital picture* is defined by the triple $(S, \omega, \hat{f})$, where $\hat{f} : S \to \mathbb{R}$ with $\hat{f}(P_{p_1,p_2,\ldots,p_n}) = f_{p_1,p_2,\ldots,p_n}$ [55]. The interpretation of grid data given the concept of a digital picture is shown exemplarily for two dimensions in Figure 6.

3.3 Image Registration

Image registration is the process of finding a spatial relation between two images [44, 67, 94, 129]. For the course of this thesis, an image is the interpretation of grid data (U, F) with $U = (V, E)$ as a digital picture $(S, \omega, \hat{f})$. Conceptually, image registration can be applied to digital pictures of arbitrary

[10] The term *surfel* is a portmanteau of 'surface' and 'element' [51].

(a) (b) (c)

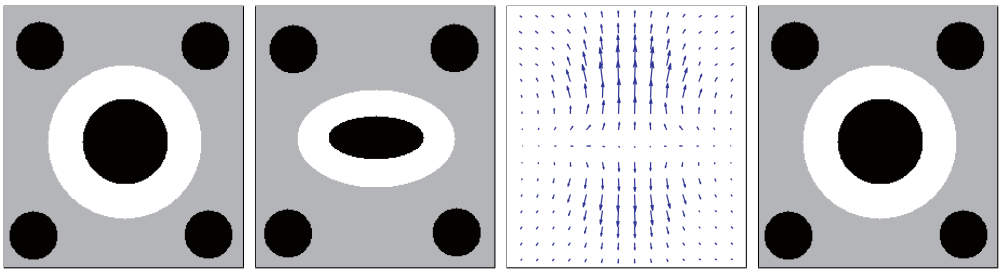
Figure 7: The image in (b) is spatially aligned to the image in (a). The result is a third image shown in (c). Images adapted from Modersitzki [94].

dimensions.

Registration is commonly used for images of a scene that come from different perspectives. These could be different points of view and/or different sensors with which the scene is observed [44, 94]. If the spatial relation is known, it is possible to compare the images as if they were taken from a common point of view. However, registration can also be applied to images of a scene from the same perspective but at different time steps [94]. In this case, the spatial relation gives information about how the scene changes between the time steps at which the images are obtained [44]. Either way, the spatial relation can be used to align one image to the other as shown in Figure 7. A prominent application of image registration is the field of medical imaging, where it is typically used to compare images from different devices that due to technical reasons produce images with different modalities. At the same time, it offers the possibility to compensate for the inevitable movements of patients when image data is obtained in a series of scans [94, 129].

3.3.1 Directed Deformation

The registration of two images is a directed process. While one image is the *fixed image* I_f, the other one is the *moving image* I_m that is deformed to fit the



(a) Fixed image I_f (b) Moving image I_m (c) Vector field $\boldsymbol{d}$ (d) I'_m

Figure 8: Academic example of image registration in which the moving image in (b) is deformed to fit the fixed image in (a). The vector field in (c) constitutes a transformation that maps pixel center of the fixed image domain to positions in the moving image domain. The resulting deformed moving image in (d) is defined over the fixed image domain and is expected to be spatially aligned to the original fixed image. In this constructed example, an exact alignment is achieved. Images adapted from Schwarz [129].

fixed image. The deformation is a mapping from the fixed image to the moving image. It can be defined by means of a vector field $\boldsymbol{d} : V_f \to \mathbb{R}^n$ that maps *spel centers*[11] of the fixed image domain to positions in the moving image domain.[12] The deformed moving image I'_m is then defined as $I'_m(\boldsymbol{v}) = I_m(\boldsymbol{v} + \boldsymbol{d}(\boldsymbol{v}))$ for all $\boldsymbol{v} \in V_f$. During the registration process, the deformation is considered a transformation $\boldsymbol{T} : V_f \to \mathbb{R}^n$, where $\boldsymbol{T}(\boldsymbol{v}) = \boldsymbol{v} + \boldsymbol{d}(\boldsymbol{v})$. The deformed moving image I'_m is equivalently defined as $I'_m(\boldsymbol{v}) = I_m(\boldsymbol{T}(\boldsymbol{v}))$ for all $\boldsymbol{v} \in V_f$. Hence, registration is the problem of finding a transformation $\boldsymbol{T}$ such that I'_m is optimally aligned to I_f. Figure 8 shows an academic example of image registration in which the deformed moving image fits the fixed image exactly.

The direction of the deformation might be confusing since a 'deformation of the moving image' is intuitively thought of as 'moving' spels from its coordinates in the moving image to new coordinates in the fixed image. The problem with this intuitive notion is that not every spel in the deformed image will be

[11] According to the definition of spels in Section 6, spel centers are exactly those points in n-dimensional Euclidean space that are specified by vertices.

[12] From a technical point of view it should be noted that the vector field may also point to positions outside of the domain of the moving image.

necessarily assigned a value and some spels can be assigned several times. To overcome this issue, the deformation is indeed defined in the opposed direction, as stated above, from fixed to moving image. Instead of moving spels to new coordinates, the opposed direction can be thought of as follows: For every spel center $\boldsymbol{v} \in V_f$ of the fixed image, the transformation defines a position $\boldsymbol{x} = \boldsymbol{T}(\boldsymbol{v})$ in the moving image, where the intensity value 'originates from'. Since $\boldsymbol{x} \in \mathbb{R}^n$ is a point in continuous space, the intensity value at $\boldsymbol{x}$ in the moving image is obtained using interpolation [67, 94, 129].

3.3.2 Overview

The following section introduces the mathematical formulation of the registration process and gives an overview of the components of which a universal registration framework consists.

Already mentioned components are the fixed image I_f and the moving image I_m of which the latter is deformed by a transformation $\boldsymbol{T}$. It should be noted that the registration process does not result in arbitrary deformations. This is due to the number of possible deformations being constraint since the transformation is actually assumed to be a *parametrized transformation* $\boldsymbol{T}_{\boldsymbol{\mu}}$. The vector $\boldsymbol{\mu}$ contains the parameters that are immanent in a transformation model. For example, when the transformation is modelled as a two-dimensional rigid transformation, $\boldsymbol{\mu}$ comprises three parameter, one for the rotation angle and two for the translations in x and y direction [66].

The quality of the alignment is defined by a *distance measure* D. In the context of image registration, a distance measure is a function $D(I_f, I_m, \boldsymbol{T}_{\boldsymbol{\mu}})$ that takes as inputs both images as well as the parametrized transformation and computes a numerical value that quantifies the extent to which the fixed image I_f and the deformed moving image I'_m are dissimilar. It should be noted that a distance measure is the inverse of a similarity measure. The latter might be a more suitable term for those concrete examples that have been proposed originally to quantify similarity. However, every similarity measure can simply be turned into a distance measure by multiplication with -1 [62]. A distance

function decreases as the alignment of both images is improved, and reaches a global minimum if I'_m is aligned to I_f in the *best possible* way [129].

In general, the problem of finding such an alignment is ill-posed. On the one hand, small changes of the input images can lead to completely different registration results. Furthermore, it can be shown that for simple academic examples the solution is not unique [94]. On the other hand, not all types of deformations allowed by a chosen transformation model are physically plausible. Depending on the domain of application of the images to be registered, only certain deformations are likely or desired to occur [129]. To circumvent the problems related to ill-posedness and privilege more likely solutions, a *regularization term* R is usually added to the distance measure [94]. The regularization term is a function $R(\boldsymbol{T}_{\boldsymbol{\mu}})$ that takes as input the parametrized transformation and penalizes undesired deformations [38]. As a result, the alignment is constraint such that the function

$$g(\boldsymbol{\mu}) = w_0 D(I_f, I_m, \boldsymbol{T}_{\boldsymbol{\mu}}) + w_1 R(\boldsymbol{T}_{\boldsymbol{\mu}})$$

reaches a minimum if the parameter vector $\boldsymbol{\mu}$ defines a transformation such that I'_m is aligned to I_f in the *optimal* way [67]. The coefficients w_0 and w_1 are fixed parameters that weight the influence of distance measure and regularization term on the constraint alignment. Commonly, the registration problem is formalized as an optimisation problem $\bar{\boldsymbol{\mu}} = \operatorname{argmin} g(\boldsymbol{\mu})$ in which g is the objective function to be minimized w.r.t. the parameters in $\boldsymbol{\mu}$.

3.3.3 Components

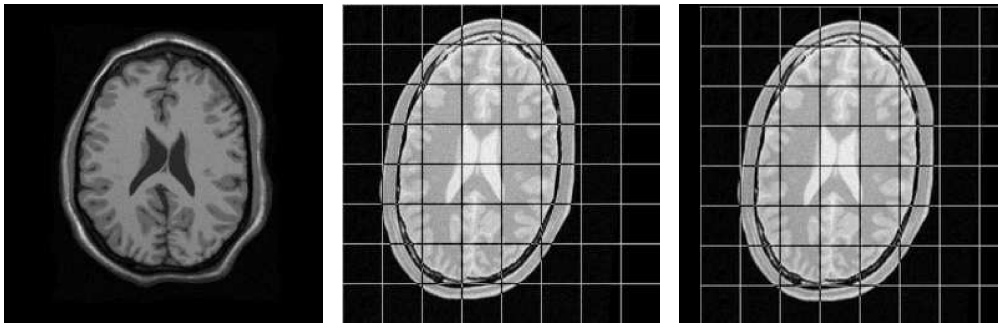
The abovementioned components of a registration framework can be considered the backbone of any implementation of a registration method. However, when a concrete image registration method should be applied to given data, a lot of choices have to be made regarding the parameters of the chosen method. Some of them belong to the universal components presented before, others are immanent in the concrete implementation and might control additional features or technical details.

For the course of this thesis, the *elastix registration method* [67, 130] was used. It can be considered a sophisticated implementation that offers many choices regarding the abovementioned components. Moreover, it provides many options to control the registration process in terms of peformance and offers many parameters for technical aspects. Besides its rich support of features, elastix' implementation progressed over a long period of time (since 2003) and proved useful in many publications. Lastly, it is still under active development and the latest release to date is version 5.0.1 released on October 5^{th} 2020.

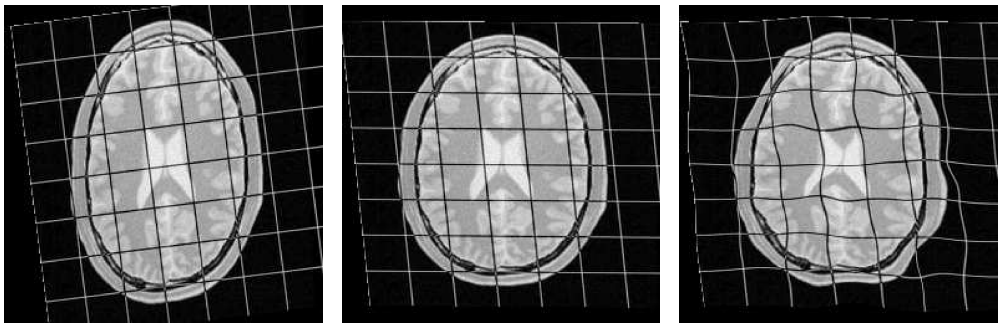
In the following, well-known examples of already mentioned components like transformation model, similarity measure and regularizer are presented all of which are supported by elastix. An hierarchical scheme as an additional feature provided by elastix is presented as well. For a concrete application of a registration framework, many more parameters need to be set. However, the majority of them is important from a technical and performance point of view and do *not* affect the overall registration result fundamentally. Some of them can even be estimated automatically by the registration method. However, choices regarding the following components need to be set explicitly and do have a fundamental influence on the registration result which is why a detailed look at the available options is presented hereinafter.

Transformation Models

A global approach is to model the deformation by means of affine transformations such as scaling, translation, rotation and shearing [18]. Global transformation models are parametrized only by a few variables which has the advantage of fast computation times [139]. However, this is only advantageous if the field of application allows for the assumption that the deformation of the entire image can be defined globally. In many physical contexts, local deformations or local deviations from a global deformation are explicitly desired, which cannot be captured by such models [94, 129]. As a consequence, local transformation models are needed where deformations can manifest itself in subregions of the image, and thus, offer even more flexibility [124, 139]. This gain of flexibility comes at the cost of increased complexity due to significantly more parameters [67,



(a) Fixed image (b) Moving image (c) Translation



(d) Rigid transformation (e) Affine transformation (f) B-spline transformation

Figure 9: Illustrations of the characteristics of different transformation models. Images adapted from Klein & Staring [66].

129]. In the following, three global and one local transformation models are presented in order of increasing flexibility. Their different characteristics are depicted qualitatively in Figure 9.

Translation The simplest transformation model only allows an image to be *translated* (see Figure 9c) defined by the translation vector $\boldsymbol{t}$ between two images, and thus, $\boldsymbol{\mu} = \boldsymbol{t}$.

$$\boldsymbol{T}_{\boldsymbol{\mu}}(\boldsymbol{v}) = \boldsymbol{v} + \boldsymbol{t}$$

Rigid Transformation In case of a rigid transformation model the image is treated as a rigid-body that can be *translated* and *rotated* (see Figure 9d).

$$\boldsymbol{T}_{\boldsymbol{\mu}}(\boldsymbol{v}) = \boldsymbol{R}(\boldsymbol{v} - \boldsymbol{c}) + \boldsymbol{c} + \boldsymbol{t}$$

The rotation is defined by the rotation matrix $\boldsymbol{R}$ as well as the center of rotation $\boldsymbol{c}$. The latter is either set by the user or by default is the geometric center of the fixed image. In 2D, the paramter vector $\boldsymbol{\mu} = (t_x, t_y, \theta_z)$ comprises the translation vector and the angle of rotation around the axis normal to the image.

Affine Transformation The affine transformation model differs from the rigid one by the matrix $\boldsymbol{A}$ having no restriction in contrast to the rotation matrix $\boldsymbol{R}$.

$$\boldsymbol{T}_{\boldsymbol{\mu}}(\boldsymbol{v}) = \boldsymbol{A}(\boldsymbol{v} - \boldsymbol{c}) + \boldsymbol{c} + \boldsymbol{t}$$

As a result, the image can be *translated*, *rotated*, *scaled* and *sheared* (see Figure 9e). The parameter vector $\boldsymbol{\mu}$ contains the translation vector and the matrix elements $a_{ij} \in \boldsymbol{A}$. In 2D, this gives $\boldsymbol{\mu} = (t_x, t_y, a_{11}, a_{12}, a_{21}, a_{22})$.

B-Spline Transformation A well-known type of transformation describing local deformations, also known as free-form-deformations [124, 129], is enabled by the B-spline transformation model (see Figure 9f). In 2D, it can be stated as follows:

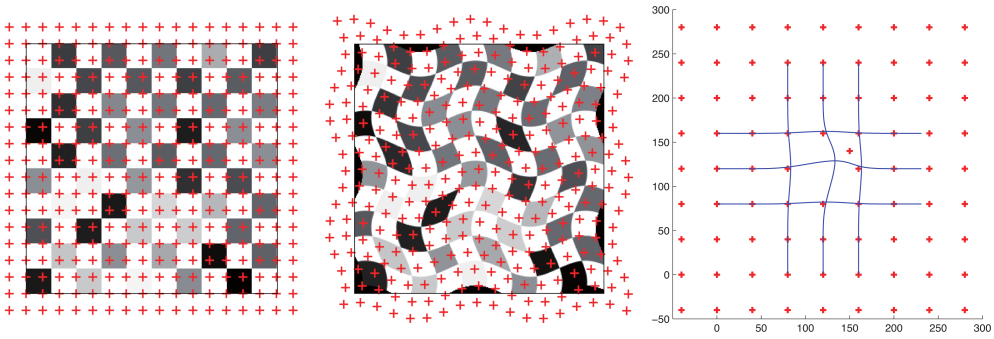
$$\boldsymbol{T}_{\boldsymbol{\mu}}(v_1, v_2) = \sum_{i=1}^{N} \sum_{j=1}^{M} B_{i,d}(v_1) B_{j,d}(v_2) \cdot \boldsymbol{c}_{i,j} \tag{3.4}$$

The general idea is to deform the moving image by manipulating a regular grid of control points $\boldsymbol{c}_{ij} \in \mathbb{R}^2$ that are distributed across the fixed image at an arbitrary resolution (see Figure 10a) [129]. The control points can be moved and are effectively interpreted as positions in the moving image. The influence of the displacement of each control point on the transformation of an individual spel center (v_1, v_2) is modelled by means of B-spline basis functions, $B_{i,d}$ and

$B_{j,d}$. Although the B-spline transformation is only queried for spel centers, the right hand side of Equation 3.4 can be evaluated for all points of the continuous domain that is spanned by a two-dimensional image. Hence, the B-spline transformation actually describes the deformation of a continuous domain (see Figure 10b). This continuous perspective on the transformation can also be used to visualize the transformation by means of the deformation of orthogonal curves that originally ran through the domain as straight lines (see Figure 10c).

In the two-dimensional case, the two (univariate) basis function, $B_{i,d}$ and $B_{j,d}$, corresponding to a control point $\boldsymbol{c}_{ij}$, form a single bivariate B-spline basis function [129, 143]. Each bivariate function can be thought of as a circular region of influence centered at control point $\boldsymbol{c}_{ij}$. Outside the circle, the function is zero and the control point has no influence (local support) [139]. Within the circle, the influence is defined by a polynomial function of degree d which monotonically increases as the radius decreases. The range of the local support is determined by the degree d of the B-spline polynomials, which is a parameter of the transformation model. However, a choice of $d = 3$ can be considered a general recommendation, since cubic B-splines offer a good compromise between local flexibility and a reasonable computational efficiency. Additionally, cubic B-splines provide the property of C^2-smoothness for the deformation due to the differentiability of the transformation model [66, 129]. In case of $d = 3$, the transformation of each spel is determined only by a small number of surrounding control points in its neighbourhood [139]. This neighbourhood consists of e.g. 16 control points in the 2D case [129].

Equation 3.4 can be regarded as a weighted sum of B-spline basis functions [67], where the weighting is determined by the distance between the coordinate of the spel in question and each control point. To account for the local support, B-spline basis functions can be rephrased, such that an equivalent definition of Equation 3.4 can be found that only loops over the control points in the neighbourhood of a spel [129]. Although this enables fast computation, the main computational complexity is determined by the resolution of the control point grid which is reflected by the parameter vector $\boldsymbol{\mu}$ that comprises the positions of control points. Varying the spacing between control points offers a flexible way of controlling the precision of the deformation [129].



(a) (b) (c)

Figure 10: The B-spline transformation is determined by the displacement of control points (red crosses). In (a), the original image with the initial control point grid is shown. After the displacement of control points in (b), the underlying image is transformed. In (c), curves that originally ran through the domain as horizontal and vertical straight lines are used to indicate the local impact of a single control point that is displaced. Images adapted from Schwarz [129].

Distance Measure

While the transformation model defines the parameter space during the registration process, the distance measure is the characteristic part of the objective function over the parameter space. The optimal parameter vector is a point in the parameter space and estimates a global minimum of the objective function.[13] Hence, the choice of distance measure is fundamental, since it determines which solutions can potentially be found by the optimization. Popular choices of distance measures to be discussed in the subsequent sections are based on image intensities, the statistical relationship of image intensities or even more general the amount of information shared between images. The choice of an appropriate distance measure depends on the context of the images to be registered. However, distance measure can be ordered according to their strictness regarding what is considered similar. In the following, the measures are presented in order of decreasing strictness.

[13] An estimate of a global minimum is most likely a point of the parameter space that is close to a local minimum.

Mean Squared Distance A straightforward approach relies on the assumption that similar structures share similar intensity values in the two images to be registered. The distance between both images is quantified based on the intensity differences at each spel center [129]. This leads to a simple distance measure called *mean squared distance* (MSD) that is well-known from many other fields of applications under the name *mean squared error*.

$$D_{MSD}(I_f, I_m, \boldsymbol{T_\mu}) = \frac{1}{|V_f|} \sum_{\boldsymbol{v} \in V_f} (I_f(\boldsymbol{v}) - I_m(\boldsymbol{T_\mu}(\boldsymbol{v})))^2$$

Normalized Cross Correlation If the assumption that similar structures have similar intensities is not valid, the field of statistics provides a similarity measure that is based on the correlation between image intensities instead. The *normalized cross-correlation* (NCC) (which is a *Pearson correlation coefficient* [44]) is sensitive to a linear relationship between two random variables. The *negated* NCC can be used as a distance measure

$$D_{NCC}(I_f, I_m, \boldsymbol{T_\mu}) = (-1) \cdot \frac{\sum\limits_{\boldsymbol{v} \in V_f} (I_f(\boldsymbol{v}) - \overline{I_f})(I_m(\boldsymbol{T_\mu}(\boldsymbol{v})) - \overline{I'_m})}{\sqrt{\sum\limits_{\boldsymbol{v} \in V_f} (I_f(\boldsymbol{v}) - \overline{I_f})^2} \sqrt{\sum\limits_{\boldsymbol{v} \in V_f} (I_m(\boldsymbol{T_\mu}(\boldsymbol{v})) - \overline{I'_m})^2}}$$

with mean intensities $\overline{I_f} = \frac{1}{|V_f|} \sum_{\boldsymbol{v} \in V_f} I_f(\boldsymbol{v})$ and $\overline{I'_m} = \frac{1}{|V_f|} \sum_{\boldsymbol{v} \in V_f} I_m(\boldsymbol{T_\mu}(\boldsymbol{v}))$. Dividing the numerator and denominator by $|V_f|$ unveils that the NCC is effectively the covariance over the product of standard deviations [44].

The NCC is dimensionless and varies between -1 and $+1$. While a value of $+1$ indicates a perfect positive linear relationship (*correlation*), a value of -1 signifies a perfect negative linear relationship (*anticorrelation*). The closer the coefficient is to zero, the less two random variables are correlated. In case of NCC $= 0$, two random variables are totally uncorrelated with respect to a linear relationship [35].

If two images of the same scene are obtained under different lighting conditions such that corresponding intensities in the images are almost linearly related, a low value of D_{NCC} will indicate a high correlation between the intensity distributions of both images. In contrast, if the intensities are nonlinearly

related, the same two images may not produce a sufficiently low distance measure [44].

Mutual Information Another example of a similarity measure that can be used to represent a statistical relation between two images is *mutual information* (MI). It was first defined by Shannon [131] as the 'rate of transmission' in his seminal paper that constituted the field of information theory. Mutual information is an entropy-based measure and quantifies the amount of information obtained from a random variable through observing another one [94]. Multiplication with -1 results in a distance measure

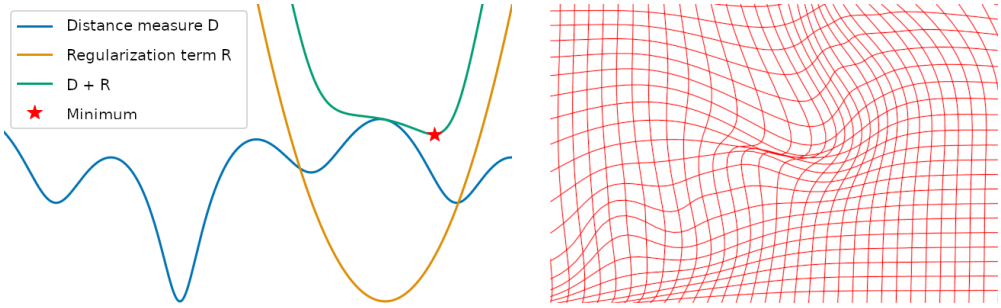
$$D_{MI}(I_f, I_m, \boldsymbol{T_\mu}) = (-1) \cdot \Big(\mathrm{H}(I_f) + \mathrm{H}(I_m, \boldsymbol{T_\mu}) - \mathrm{H}(I_f, I_m, \boldsymbol{T_\mu}) \Big)$$

where $H(I_f)$ and $H(I_m, \boldsymbol{T_\mu})$ denote the entropies of the fixed image I_f and the deformed moving image I'_m, and $H(I_f, I_m, \boldsymbol{T_\mu})$ denotes their joint entropy. The application of MI in the context of image registration was first proposed independently by Collignon [25] and Viola [148] for medical images and has been used since then by many authors [94]. The reason for its success is due to MI neither relying on the values of intensity nor on a linear relationship between their distributions [129]. Hence, MI can be used to determine the similarity between images of different modalities [44].

While entropy refers to the average amount of information held in a random variable, the joint entropy measures the average amount of information associated with the combination of two random variables. If two random variables are totally unrelated, their joint entropy is equal to the sum of individual entropies. The more similar two random variables are, the lower is their joint entropy compared to the sum of individual entropies [48].

$$\mathrm{H}(I_f, I_m, \boldsymbol{T_\mu}) \leq \mathrm{H}(I_f) + \mathrm{H}(I_m, \boldsymbol{T_\mu})$$

In the context of images, where the random variables are image intensities, registration can be thought of as reducing the joint entropy of the two images about to be aligned. The addition of individual entropies $\mathrm{H}(I_f)$ and $\mathrm{H}(I_m, \boldsymbol{T_\mu})$



(a) (b)

Figure 11: A similarity measure is prone to many local minima. The addition of a regularization term as shown in (a) should eliminate small minima that represent irrelevant solutions and at the same time promote those solutions that are more likely to occur. In (b), a local deformation results in a fold, which is considered implausible in many physical contexts. By means of e.g. a bending energy regularizer, deformation like the one that is depicted should be prohibited. Image in (a) adapted from: `https://www.inference.vc/notes-on-imaml-meta-learning-without-differentiating-through`. Image in (b) reproduced from Ozeré [104].

is necessary to compensate for the dependence of the joint entropy $\mathrm{H}(I_f, I_m, \boldsymbol{T_\mu})$ on the alignment which is known to be an important limitation [48]. Mutual information can qualitatively be thought of as a measure of how much information is shared between two images. The lower the value of D_{MI} is, the better is the alignment of two images with respect to their shared information.

Regularization Term

As mentioned before, finding the best alignment between two images by means of a distance measure alone is an ill-posed problem. An objective function that is solely defined by a distance measure is characterized by many local minima since the input data can be expected to be noisy in general. The regularization term penalizes certain properties of the transformation and since it *only* depends on the transformation, it is generally considered to exhibit less local minima compared to the similarity measure. On the one hand, the regularization term

is added to the similarity measure in order to 'smooth away' a number of small local minima, which is illustrated in Figure 11a, and to pick out solution that are more likely. On the other hand, regularization can be considered another possibility to inject domain knowledge into the registration process [94]. This applies particularly to the B-spline transformation model which allows for local deformations that may exhibit sharp deviations such as strong compression with a nearby high expansion [38, 117]. Despite the local deformation being generally favoured in many applications, the aforenamed extreme manifestations as shown in Figure 11b are disregarded since they are considered physically implausible in most of the cases [94, 129].

A commonly used regularizer that penalizes the aforenamed sharp deviations of local deformations is known as the *bending energy regularizer* [117]. In 2D, it can be stated as

$$S(\boldsymbol{T}_{\boldsymbol{\mu}}) = \frac{1}{N \cdot M} \sum_{i}^{N \cdot M} \sum_{j=1}^{2} \Big(\frac{\partial^2 T_j}{\partial x_1^2}(\tilde{\boldsymbol{c}}_i) \Big)^2 + 2 \Big(\frac{\partial^2 T_j}{\partial x_1 \partial x_2}(\tilde{\boldsymbol{c}}_i) \Big)^2 + \Big(\frac{\partial^2 T_j}{\partial x_2^2}(\tilde{\boldsymbol{c}}_i) \Big)^2$$

where N and M are number of control points along each dimension and $\tilde{\boldsymbol{c}}_i$ denotes the displacement of a control point in relation to its original position [66].

Optimization Strategy

The optimal parameter vector $\bar{\boldsymbol{\mu}} = \operatorname{argmin} g(\boldsymbol{\mu})$ is usually estimated by means of an iterative optimization strategy. In every iteration k, the current parameter vector $\boldsymbol{\mu}_k$ is updated by taking a step in the search direction $\boldsymbol{d}_k$ where $a_k \in \mathbb{R}$ is a real-valued scalar that determines the step size along the search direction [67, 94].

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k - a_k \boldsymbol{d}_k, \qquad k = 0, 1, 2, ...$$

A wide range of optimization methods can be formulated in this way, each having different definitions of a_k and $\boldsymbol{d}_k$ [67]. In principle, any minimization technique can be used for the computation of the optimal parameter vector $\bar{\boldsymbol{\mu}}$. However, Newton-type methods based on second order derivatives are not stable for real-life applications. This is due to the derivatives of images that need to be

approximated from the discrete data. Since image data is typically corrupted by noise, the estimation of derivates is a delicate matter [94]. Therefore, a common choice for the search direction $\boldsymbol{d}_k$ is the derivative of the objective function $\frac{\partial g}{\partial \boldsymbol{\mu}}$ evaluated at the current position $\boldsymbol{\mu}_k$. In this case, the iterative optimization strategy is equivalent to a gradient descent method. To enable a reasonable computation time, experimental results indicate that a stochastic gradient descent method is a good choice for many applications. It reduces the computation time of the derivatives of the objective function g by using only a subset of spels. For the stochastic method to converge, new samples must be selected randomly in each iteration [67, 68]. Despite ensuring that the optimization converges, many additional paramters needs to be set for an iterative strategy. In the case of a (stochastic) gradient descent optimization, solutions exists to estimate the accompanying parameters automatically [65, 113].

Hierarchical Scheme

In order to be able to capture structures at different scales in the images to be registered and at the same time to improve the convergence of the optimization, two hierarchical strategies are supported by elastix. What is common to both strategies is the approach to iterate the full registration process at different resolution levels. A hierarchical scheme starts the registration at the coarsest resolution level and progresses to higher levels until the finest resolution is reached. After each iteration, the registration results are propagated to the next level and used as initial conditions. This scheme ensures that structures at larger scales can be recovered early at a coarse resolution and details at smaller scales are accounted for at increasingly finer resolutions [94, 129]. A hierarchical scheme as mentioned above can be used to consider the data of both time steps at different resolutions and/or consider different degrees of freedom of the transformation models [67].

Data resolution Different resolutions of the data result from smoothing with or without downsampling. A series of images with increasing amount of

smoothing is called a scale space. If the images are not only smoothed, but also downsampled, the data is not only less complex, but the amount of data is actually reduced, which is known as a Gaussian pyramid.

Transformation resolution Another possibility is to consider different degrees of freedom within the transformation model. For the B-spline transformation, this means to start the registration with a coarse control point grid, only capable of modelling deformations between large-scale structures. In subsequent resolutions, the B-spline grid is gradually refined, thereby introducing the capability to match smaller structures.

3.4 Level Sets & Regions

As mentioned in the beginning of the last section, the fundamental assumption of grid data is that it is a discrete representation of an unknown continuous scalar field $f : \mathscr{D} \to \mathbb{R}$. A direct approach to defining regions of such a function is by means of level sets. Given a value $h \in \mathbb{R}$, the level set of f is the set $\mathscr{L}_h = \{\boldsymbol{x} \mid f(\boldsymbol{x}) = h\}$ also denoted as $f^{-1}(h)$. A level set separates its complement $\mathscr{D} \setminus \mathscr{L}_h$ into two open subsets, the *strict sublevel set* $\mathscr{L}_h^- = \{\boldsymbol{x} \mid f(\boldsymbol{x}) < c\}$ and the *strict superlevel set* $\mathscr{L}_h^+ = \{\boldsymbol{x} \mid f(\boldsymbol{x}) > c\}$.

A connected component of some set L is a maximally connected subset of L, i.e. a connected subset of L that is not contained in any other connected subset of L [153]. Connected components of $\mathscr{L}_h$ are referred to as *contours* [20, 108, 153]. In this thesis, connected components of $\mathscr{L}_h^-$ and $\mathscr{L}_h^+$ are called *strict sub-components* and *strict super-components*, respectively.

For now, contours are assumed to be regular, i.e. they do not contain critial points [153].[14] Then, each contour λ is an $(n-1)$-manifold that divides its complement $\mathscr{D} \setminus \lambda$ into two connected components [153] such that every path from a point in the one component to a point in the other component must pass through λ [20]. It should be noted that contours may be cut at the boundary of $\mathscr{D}$, and therefore, a contour can be an $(n-1)$-manifold with boundary [24].

[14] Critical points are introduced in Section 3.5.

Moreover, each contour defines the boundary between a strict sub-component and a strict super-component.

When it comes to the actual definition of regions, the closure of strict sub-components and strict super-components is usually considered, which is expressed by omitting the term 'strict'. Technically, *sub-components* and *super-components* are connected components of the *sublevel set* $\mathscr{L}_h^- \cup \mathscr{L}_h = \{\boldsymbol{x} \mid f(\boldsymbol{x}) \leq h\}$ and the *superlevel set* $\mathscr{L}_h^+ \cup \mathscr{L}_h = \{\boldsymbol{x} \mid f(\boldsymbol{x}) \geq h\}$. It is a common approach to define regions either as sub- or as super-components.

However, the original scalar field f is unknown and only represented in discretized form as grid data. In analogy to the two interpretations of grid data, there are two approaches how to derive level sets from the discrete representation. On the one hand, when grid data is assumed as a sampling of f, level sets can be *approximated* by a set of $(n-1)$-dimensional simplices embedded in the continuous domain that is spanned by the grid [153]. On the other hand, when interpreted as a digital picture $(S, \omega, \hat{f})$, digital level sets are defined by subsets of ω [55].

3.4.1 Approximations of Level Sets

A well-known algorithm that approximates level sets [11, 86, 153] given a three-dimensional uniform grid is Marching Cubes [87]. Although the algorithm operates on the uniform grid alone and does not depend on a reconstructed scalar field, it shows resemblance to the multilinear interpolant. This is due to computing intersection points on grid edges where a linear interpolant takes on the *isovalue* h. For each cell of the grid, the intersection points on its edges are connected to form one or more triangles. The key aspect of Marching Cubes – more precisely, after it was revised by Montani et al. [96] and Zhou et al. [157] – is that the possible combinations of a cell's edge intersections can be reduced to a finite number of configurations that determine the generation of triangles within the cell. Triangles that share an edge are stitched together and eventually build piecewise linear 2-manifolds that are commonly referred to as *isosurfaces*.

For two-dimensional uniform grids, a similar algorithm is known as MARCHING SQUARES. It proceeds just as its three-dimensional equivalent but has less and simpler configurations for the generation of line segments within each cell. In two dimensions, this results in piecewise linear 1-manifolds referred to as *isolines*. The geometry of both, isolines and isosurfaces are determined by a finite number of piecewise linear elements. Since it is impossible to interpolate linearly within the original cells of a uniform grid, isocurves and isosurfaces are approximations of a level set $\tilde{f}^{-1}(h)$ of any continuous scalar field $\tilde{f}$.

Both MARCHING SQUARES and MARCHING CUBES make use of a precomputed lookup table, that stores different configurations of how dummy intersection points on edges of a generic cell can be triangulated. For each actual grid cell, actual edge intersections are computed by means of linear interpolation and subsequently triangulated according to the lookup table. Bhaniramka et al. [9, 10] proposed an algorithmic approach to the generation of lookup tables for n-orthotopes that specify the generation of $(n-1)$-simplices. Their work generalizes the hand-made lookup tables of MARCHING SQUARES and MARCHING CUBES which can be reproduced by their algorithm for $n = 2$ respectively $n = 3$. Later, Wenger [153] extended the algorithm to n-polytopes. The final set of $(n-1)$-simplices represents the approximated level set whose connected components are piecewise linear $(n-1)$-manifolds called *isocontours*.

Topological Correctness of Approximations

Despite isocontours being piecewise linear manifolds, they are not necessarily *topologically correct* with regard to a specific interpolant. As mentioned earlier, the choice of cell interpolant determines the scalar field $\tilde{f}$ that is reconstructed on the basis of the given grid. An approximated level set $\mathcal{L}_h$ is topologically correct if its topology coincides exactly with the topology of the level set $\tilde{f}^{-1}(c)$ [23], or more formally, if it is homeomorphic to $\tilde{f}^{-1}(c)$ [30, 153].

Topological correctness for isocurves in 2D, regarding the bilinear interpolant, can be achieved easily by incorporating an ASYMPTOTIC DECIDER that was originally proposed by Nielson & Hamann [102] in 3D as an extension of the original MARCHING CUBES algorithm [153]. In 3D, due to more complex

triangulations within a cell, it took 26 years after the initial publication of MARCHING CUBES [87] to achieve topological correctness with regard to the trilinear interpolant. Key publications of this progress are the ASYMPTOTIC DECIDER algorithm [102], the MARCHING CUBES 33[15] algorithm [23] and at long last the work of Custiodio et al. [30], who revised MARCHING CUBES 33 to be finally topologically correct.[16]

3.4.2 Digital Topology and Geometry

As mentioned in the last subsection, MARCHING CUBES and its variants process the grid on the basis of linear interpolation along grid edges. The results are piecewise linear $(n-1)$-manifolds composed of $(n-1)$-simplices whose union defines a subset of the continuous domain that is spanned by the grid.

When grid data is interpreted as a digital picture $(S, \omega, \hat{f})$, the domain of $\hat{f}$ is not a continuous subspace of $\mathbb{R}^n$, but a discrete set of spels S each of which is associated with its own continuous subspace of $\mathbb{R}^n$ (see Figure 12a) [55]. To be able to relate topological concepts of Euclidean space to the domain of a digital picture, the set of spels S is considered with respect to the proto-adjacency ω. As stated in Section 6, the pair (S, ω) defines a digital space which is essentially a connected digraph. Therefore, digital versions of continuous concepts like connectedness, connected components and level sets make use of graph-theoretic concepts. The definition of topological concepts for digital pictures falls within a scope called *digital topology and geometry* [55, 70, 84] and this section presents a *graph-based approach*[17] towards it [74, 84]. Prominent contributions were made by Rosenfeld et al. [70, 72, 73, 98, 120, 121, 123]. However, their work only considers the cases when $n = 2$ and $n = 3$, but Herman [55] generalized the basic ideas to arbitrary dimensions. Moreover, he proposed an edge-based definition of boundaries in digital pictures that serves as a basis for the thesis at hand.

[15] The name MARCHING CUBES 33 refers to the extended number of 33 different configurations for the generation of triangles, instead of 15 as with MARCHING CUBES.

[16] A related work was published earlier by Weber et al. [151].

[17] Other approaches towards digital topology and geometry such as the work of Khalimsky [63, 64], Kovalesvsky [76] and Kong [71] involve the definition of topologies based on cell complexes [1, 69] and are called *axiomatic* or *topological approaches* [74, 84].

Spel Adjacencies and Connectedness

For a graph-based approach towards digital topology and geometry as proposed by Herman [55], a notion of connectedness in a digital space is necessary in the first place. Based on the proto-adjacency ω, *spel adjacencies* can be defined. An irreflexive and symmetric adjacency relation ρ on S such that $\rho \supseteq \omega$ (ρ contains ω) is called a spel adjacency. Trivially, ω itself can be regarded as a spel adjacency, which considers two spels to be adjacent if their intersection in $\mathscr{D}$ is $(n-1)$-dimensional [55].

$$\omega = \{(P_{p_1,p_2,\ldots,p_n}, P_{q_1,q_2,\ldots,q_n}) \mid \textstyle\sum_{i=1}^{n} |p_i - q_i| = 1\} \tag{3.5}$$

For $n = 2$, ω-adjacency is the well-known 4-adjacency [118], where two pixels are adjacent if they share an edge. The equivalence for $n = 3$ is the classical 6-adjacency [120], where two voxels are adjacent if they share a face. Another spel adjacency α can be defined that considers two spels to be adjacent, if their intersection in $\mathscr{D}$ is at least 0-dimensional, i.e. if they at least share a point.

$$\alpha = \{(P_{p_1,p_2,\ldots,p_n}, P_{q_1,q_2,\ldots,q_n}) \mid |p_i - q_i| \leq 1\}. \tag{3.6}$$

For $n = 2$, α-adjacency is the well known 8-adjacency [118] and for $n = 3$ it is the classic 26-adjacency [120].

Let ρ be a spel adjacency on S and let $K \subseteq S$ be a subset of S. If $(\boldsymbol{u}, \boldsymbol{v}) \in \rho$, $\boldsymbol{u}$ is said to be ρ-adjacent to $\boldsymbol{v}$ and $\boldsymbol{v}$ is said to be ρ-adjacent from $\boldsymbol{u}$. A sequence $\langle \boldsymbol{v}^{(0)}, \ldots, \boldsymbol{v}^{(n)} \rangle$ of elements of K such that $\boldsymbol{v}^{(i)}$ is ρ-adjacent to $\boldsymbol{v}^{(i-1)}$, for $1 \leq i \leq n$, is called a ρ-path in K. In such a case, $\boldsymbol{v}^{(0)}$ and $\boldsymbol{v}^{(n)}$ are said to be ρ-connected in K. Evidently, ρ-connectedness is symmetric, because the reversal of a ρ-path from $\boldsymbol{u}$ to $\boldsymbol{v}$ is a ρ-path from $\boldsymbol{v}$ to $\boldsymbol{u}$. It is also transitive, because if there are paths from $\boldsymbol{u}$ to $\boldsymbol{v}$ and from $\boldsymbol{v}$ to $\boldsymbol{w}$, a ρ-path from $\boldsymbol{u}$ to $\boldsymbol{w}$ can be obtained by concatenating the two individual ρ-paths. A vertex is also said to be trivially connected to itself by a path of length zero. Hence, ρ-connectedness in K is an equivalence relation[18] defined as the reflexive, transitive closure of ρ applied to the subset K. A subset of K is ρ-connected in K if and only if every

[18] An equivalence relation is a binary relation that is reflexive, symmetric and transitive.

pair of vertices is ρ-connected in K. Maximally ρ-connected subsets of K define (connected) components of K with respect to ρ. K is said to be ρ-connected, if it is composed of only one component [55, 70].

Digital Level Sets

Due to the discrete nature of a digital space, the concept of level sets does not translate directly to digital pictures: A set of spels that *separates* the rest of a digital space does not exist in general for every possible level. However, given a threshold at level h, S can always be *partitioned* into two 'clopen' subsets[19], the *strict lower level set* $\mathcal{SL}_h = \{P_{p_1,p_2,\ldots,p_n} \mid \hat{f}(P_{p_1,p_2,\ldots,p_n}) < h\}$ and the *upper level set* $\mathcal{U}_h = \{P_{p_1,p_2,\ldots,p_n} \mid \hat{f}(P_{p_1,p_2,\ldots,p_n}) \geq h\}$. Alternatively, S can be partitioned into the *lower level set* $\mathcal{L}_h = \{P_{p_1,p_2,\ldots,p_n} \mid \hat{f}(P_{p_1,p_2,\ldots,p_n}) \leq h\}$ and the *strict upper level set* $\mathcal{SU}_h = \{P_{p_1,p_2,\ldots,p_n} \mid \hat{f}(P_{p_1,p_2,\ldots,p_n}) > h\}$ [57, 95]. Connected components of $\mathcal{SL}_h$ and $\mathcal{L}_h$ are called (strict) lower-components and connected components of $\mathcal{SU}_h$ and $\mathcal{L}_U$ are referred to as (strict) upper-components.

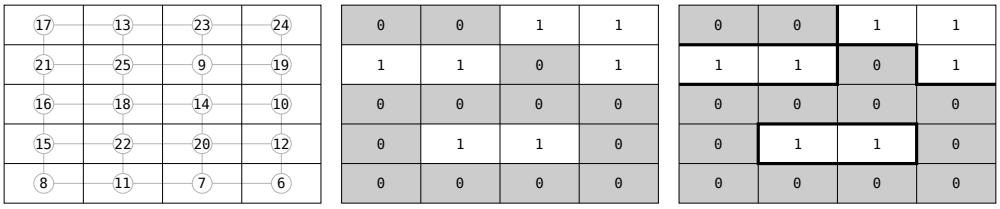
For the purpose of defining digital level sets and contours, the partition of S into the sets $\mathcal{SL}_h$ and $\mathcal{U}_h$ is henceforth assumed. Intuitively, a binary picture can be employed to represents a partition into two disjoint-sets. To be precise, a level h gives rise to a thresholded picture, which is a binary picture $(S, \omega, \hat{f}_h)$ [55].

$$\hat{f}_h(P_{p_1,p_2,\ldots,p_n}) = \begin{cases} 0, & \text{if } P_{p_1,p_2,\ldots,p_n} \in \mathcal{SL}_h \\ 1, & \text{if } P_{p_1,p_2,\ldots,p_n} \in \mathcal{U}_h \end{cases} \tag{3.7}$$

An example of thresholding a digital picture as defined in Equation 3.7 is illustrated in Figure 12a and 12b.

In contrast to a spel-based definition, a digital level set can be defined by a set of surfels $(P_{p_1,p_2,\ldots,p_n}, P_{q_1,q_2,\ldots,q_n}) \in \omega$ such that $P_{p_1,p_2,\ldots,p_n} \in \mathcal{U}_h$ and $P_{q_1,q_2,\ldots,q_n} \in \mathcal{SL}_h$. Transferred to the continuous domain $\mathcal{D}$ that is spanned by a digital picture, a digital level corresponds to a set of $(n-1)$-dimensional intersections

[19] The term *clopen* is a portmanteau of 'closed' and 'open' and a subset that is both open and closed. In a digital space, which is a discrete topological space, every subset is clopen.



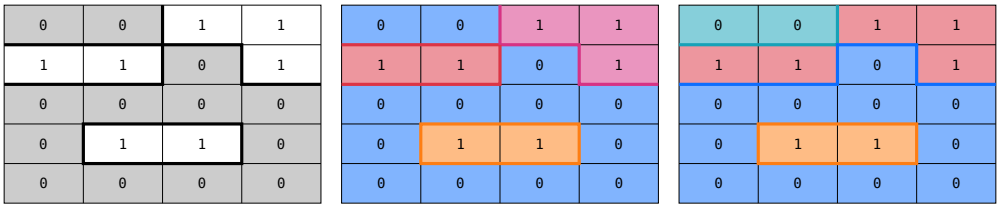
(a) Digital picture (b) Binary picture ($h = 19$) (c) Digital level set ($h = 19$)

Figure 12: In (a), the digital picture of Figure 6 is shown without the original grid. Thresholding at level $h = 19$ results in the binary picture in (b). The bold line segments in (c) indicate the corresponding digital level set.

of spels as exemplarily illustrated in Figure 12c. The aforementioned condition ensures the orientation of those intersections to be consistent.[20]

When it comes to the identification of individual digital contours, the concept of connected components of a digital level set does once again not translate directly. A digital level set is formally a set of directed edges for which there is no straightforward notion of connectivity. However, the digital counterpart of contours can be defined subsequently to the identification of *strict lower-components* and *upper-components*. In the context of binary pictures, Rosenfeld & Pfaltz [123] as well as Duda et al. [36] postulate the necessity of considering different adjacency relations for connected components of 0s and 1s. More specifically, this requires α-adjacency to be taken into account for strict lower-components and ω-adjacency for upper-components or vice versa [55]. It should be noted that each of the two options induces a different composition of components which is illustrated for the recurring two-dimensional example in Figure 13. Given a level h, $\mathcal{SL}_h^i$ denotes the i^{th} (strict) lower-component such that $\mathcal{SL}_h^i$ is α-connected (ω-connected) and $\mathcal{U}_h^j$ denotes the j^{th} upper-component such that $\mathcal{U}_h^j$ is ω-connected (α-connected). A digital contour is a *digital boundary* between a strict lower-component $\mathcal{SL}_h^j$ and an upper-component $\mathcal{U}_h^j$ defined by all surfels $(P_{p_1,p_2,\ldots,p_n}, P_{q_1,q_2,\ldots,q_n}) \in \omega$ such that $P_{p_1,p_2,\ldots,p_n} \in \mathcal{U}_h^j$ and $P_{q_1,q_2,\ldots,q_n} \in \mathcal{SL}_h^i$ [55, 93]. The digital concept of a boundary is defined

[20] The 'inverted' condition, that $P_{p_1,p_2,\ldots,p_n} \in \mathcal{SL}_h$ and $P_{q_1,q_2,\ldots,q_n} \in \mathcal{U}_h$ leads to a consistent orientation as well.



(a) Digital level set ($h = 19$) (b) (c)

Figure 13: Different compositions of components induced by considering α-adjacency for strict lower-components and ω-adjacency for upper-components in (b) or vice versa in (c). The colouring of spels indicates different components whereas the colouring of bold line segments gives information on the concomitant digital contours. In two dimensions, α- and ω-adjacency are commonly referred to as 8- and 4-adjacency.

thoroughly by Herman [55]. Essentially, the combination of α- and ω-adjacencies is called a *Jordan pair* of spel adjacencies that defines properties of digital boundaries that resemble properties of boundaries in a continuous setting. Transferred to $\mathcal{D}$, a digital contour corresponds to an *oriented* $(n-1)$-dimensional surface between a strict lower-component and an upper-component. Since either strict lower- or upper-components are α-connected, digital contours may touch themselves. Hence, they are not generally $(n-1)$-manifolds in contrast to the output of MARCHING CUBES and its variants. The number of digital contours a digital level set is composed of depends on the level h as well as the spel adjacencies considered for strict lower- and upper-components.

Digital Regions Similiar to the definition of continuous regions as the closure of strict sub- or super-components, digital regions are commonly defined either as *lower-components* or as *upper-components*. However, for continuous regions, approximated level sets define the extent of sub- and super-components, while in a digital picture, a level set solely defines the lower level set or the upper level set. The actual digital regions are determined by the choice of spel adjacencies with respect to which lower- and upper-components are defined. For the sake of simplicity, α-connected lower-components and ω-connected upper-components

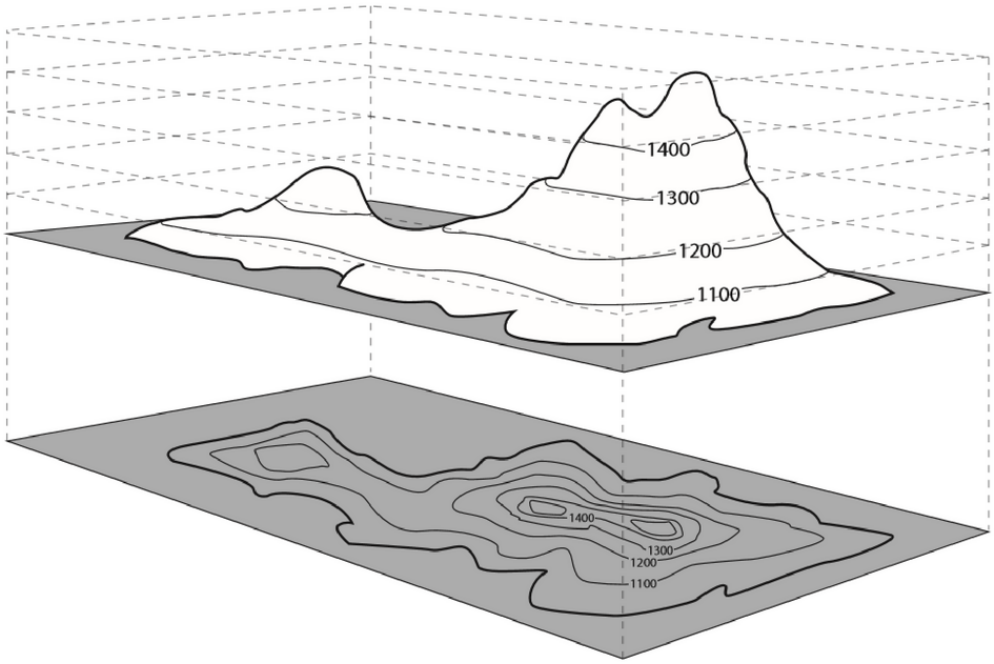


Figure 14: The schematic of a terrain (*top*) is shown with cross-sections of constant height. The topography of the terrain is visualized by means of contour lines in a topographic map (*bottom*). Image reproduced from: `https://www.greenbelly.co/pages/contour-lines`.

are assumed for the rest of this thesis. Data structures that are able to capture the topological relations among lower- and upper-components, respectively, of all possible levels will be presented in the next section.

3.5 Contour Tree

A familiar two-dimensional application of level sets can be found in topographic maps where several height levels visualize the topography of the corresponding terrain (see Figure 14). A single level set can be conceived of as a cross-section of the topographic surface at constant height. In cartography, connected components of a level set are called *contour lines*. The evolution of contour lines can be thought of intuitively by means of a variation of the height level:

sweep of the cross-section from high to low height levels, which is illustrated in Figure 15a and 15b: While the height level is varied, contour lines appear at peaks, disappear at pits and may join and/or split in between at passes. At these points, topological changes of level sets occur that allow for a compact description of the topography of the terrain. In fact, computer cartography is the discipline in which algorithms for the computation of *contour trees* were developed for the first time. The following section outlines the history of these developments by means of key publications and thereby introduces the contour tree as a graph, as shown in Figure 15c, that represents the evolution of contours.

Historic Overview

Originally, the contour tree was considered to be a graph whose nodes represent the contours of a finite set of height levels and whose arcs represent the adjacency relations among them.[21] The earliest publication is the work of Boyell & Ruston [15], who refer to the contour tree as an *enclosure tree*[22], followed by the publications of Morse [99], who uses the term *contour map graph*. Contours are computed independently from discrete elevation data in a preprocessing step [133] which is why the height levels have not been chosen necessarily with a full topological analysis of level sets in mind. Kweon & Kanade [79] propose to compute contours from discrete elevation data defined by a constant increment between height levels. The resulting contour tree is called *topographic change tree* and is used to extract topographic features of the terrain. Still, relying on a finite set of height levels does not guarantee that all topological changes of level sets are captured.

[21] The entities of the contour tree are referred to as nodes and arcs to not confuse them with the graph-based definition of the grid, where the corresponding entities are referred to as vertices and edges. Furthermore, it is assumed that each node of the contour tree is associated with the parameter value $h \in \mathbb{R}$ of the corresponding level set $f^{-1}(h)$ to be able to refer back to the level of the contour.

[22] The notion of enclosure relations assume that contours are nested inside each other. This poses additional constraints on contours. In particular, the nesting property requires all contours to be closed manifolds such that the 'inside' and 'outside' of a contour is well-defined [20]. In other words, contours would not be allowed to be cut at the boundary of the domain. Since contours are explicitly allowed to be manifolds with boundary as stated in Section 3.4, the notion of adjacency relations among contours is favoured in this thesis.

Mark [89] presents a new concept of the contour tree, called the *surface tree*, that utilizes the *surface network* by Pfaltz [111] to yield all height levels of a smooth[23] 2-dimensional scalar field at which level sets undergo topological changes. This gain of topological information is due to the surface network whose definition is based on the identification of *critical points* [89]. At critical points, the gradient of a function vanishes. According to Morse theory, critical points[24] of smooth scalar fields are the points where the topology of level sets changes [91].[25] The function values associated with critical points are called critical values [24, 108]. Under the assumption that a critical value is swept across from a higher to a lower level, critical points can be characterized as follows [20, 24, 107]:

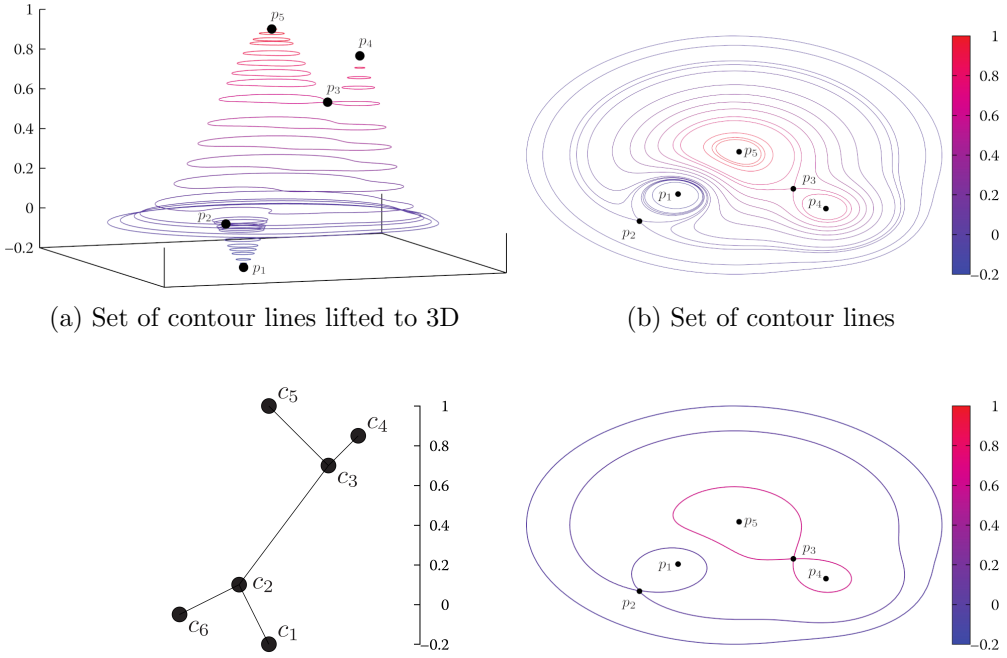
- At a maximum, a contour appears.
- At a minimum, a contour disappears.
- At a saddle, two (or more) contours join and/or a contour splits into two (or more) contours.

Takahashi et al. [140] relate the new approach to the contour tree to the more general concept of the Reeb graph [114]. The Reeb graph is a tool in Morse theory that reflects the evolution of contours of smooth scalar fields on arbitrary manifolds [20]. When the domain of such a function is simply connected, the Reeb graph is a tree [20, 140]. A formal definition of the contour tree by means of Morse Theory that generalizes the ideas of Mark [89] for arbitrary dimensions is presented by Carr et al. [22] based on equivalence classes of contours. In essence, nodes of the contour tree represent critical points of a function $f : \mathscr{D} \to \mathbb{R}$ and arcs represent f-monotone paths of points in $\mathscr{D}$ such that contours passing through these points do not contain any critical point [20].

[23] For the course of this thesis, a function is considered smooth if it is at least twice differentiable.

[24] Morse theory requires that the critical points are isolated, i.e. that they occur at distinct points and values [20].

[25] Morse theory not only approaches topological changes of level sets but also of individual contours (changes of topological genus) based on a superset of critical points called *Morse critical points* [20, 107].



(a) Set of contour lines lifted to 3D

(b) Set of contour lines

(c) Contour Tree

(d) Critical contour lines

Figure 15: In (a) and (b), the evolution of contour lines is illustrated while the level is varied. The contour lines in (a) are lifted to 3D to support the idea of cross-sections of a topographic surface. The critical points are also shown such that contours degenerated to or passing through critical points can be identified easily as nodes in the contour tree shown in (c). These critical contours are shown separately in (d). Images adapted from Wenger [153].

Intuitive Perspective

For the course of this thesis, a more intuitive perspective on the 'new' approach to the contour tree is presented that builds on the original conception where nodes represent contours and arcs represent the adjacency relations among them. Although not limited to two dimensions, this perspective can initially be thought of as the limit of the contour tree of Kweon & Kanade [79] as the increment between levels approaches zero. The result is a graph whose nodes represent the contours of all levels including those that are function values of

critical points. More specifically, critial points are represented in the tree by nodes whose corresponding contours contain the critial points. These nodes can be identified based on their number of incident arcs. Taking up the intuitive description of a sweep from high to low levels, the arcs between nodes are assumed to be directed from nodes corresponding to contours of higher levels to those of lower levels. Subsequently, nodes can be grouped according to their in-degree d_{in} and out-degree d_{out}. The in-degree of a node is the number of incoming arcs. The out-degree of a node is the number of outgoing arcs.

- Nodes with $d_{in} + d_{out} = 1$ correspond to contours that are degenerated to local maxima or local minima. While $d_{in} = 0$ indicates the appearance of a contour at a maximum, $d_{out} = 0$ indicates the disappearance of a contour at a minimum.

- Nodes with $d_{in} > 1$ or $d_{out} > 1$ correspond to contours that pass through saddles. While $d_{in} > 1$ indicates the join of d_{in} contours, $d_{out} > 1$ indicates the split into d_{out} contours. It should be noted that splitting and joining is not necessarily mutually exclusive, such that contours can join and at the same time split into contours at a saddle.

- Nodes with $d_{in} = d_{out} = 1$ correspond to contours that evolve without being involved in any of the aforenamed topological changes of level sets.

As already mentioned in Section 3.4, a contour that does not contain a critical point is called a *regular contour* and divides its complement into two connected components [153]. In contrast, the complement of contours degenerated to extrema or passing through saddles exhibit either one or more than two connected components. Therefore, they are referred to as *critical contours* [153] (see Figure 15d). Analogously, nodes that fall in the first two categories are called *critical* and those of the third category are called *regular* [20]. The parameter values associated with critical nodes are the levels at which topological changes of level sets occur. If all regular nodes are removed from the

tree by a series of homeomorphic contractions[26], the reduced tree is equivalent to the contour tree defined by Carr et al. [21] by means of Morse theory [89].

A visualization of the contour tree as in Figure 15c is usually a planar straight-line graph[27] such that the vertical positions of nodes reflect the associated levels. If the tree is then cut horizontally, the intersections correspond to the contours that belong to a particular level, which is illustrated for two exemplary cuts in Figure 16a and 16b.

3.5.1 Join and Split Trees

As stated in Section 3.4, the definition of regions is inherently related to contours as contours are connected components of the level set $\mathscr{L}_h$ and regions are connected components of either the sublevel set $\mathscr{L}_h^- \cup \mathscr{L}_h$ or the superlevel set $\mathscr{L}_h^+ \cup \mathscr{L}_h$. However, while the level is varied, topological changes of the sublevel set and the superlevel set occur each at different subsets of critical points [20, 24]. More intuitively, given a level set $\mathscr{L}_h$, the number of sub- and super-components is generally not equal to the number of contours. Imagine two horizontal cuts through the contour tree as indicated by the two dashed lines in Figure 16a. At the magenta level, there are two intersections with the tree corresponding to the two magenta contours highlighted in Figure 16b. As shown in Figure 16c, there are in fact two super-components but only one sub-component. The opposite situation occurs at the purple level. Again, there are two intersections with the tree corresponding to the two purple contours highlighted in Figure 16b. However, there is only one super-component albeit there are two sub-components as shown in Figure 16d.

When the focus is not on contours but on regions, a representation similar to the contour tree but for the evolution of sub- and super-components would be handy. Fortunately, the necessary information is contained within the

[26] When removing a regular node, the connectivity of the tree is preserved by contracting the two incident arcs into a single one. A *homeomorphic contraction* can be regarded as a reduction to distinguish it from the simple removal of a vertex [22].

[27] A planar straight-line graph is an embedding of a planar graph in the plane such that its edges are straight line segments [112].

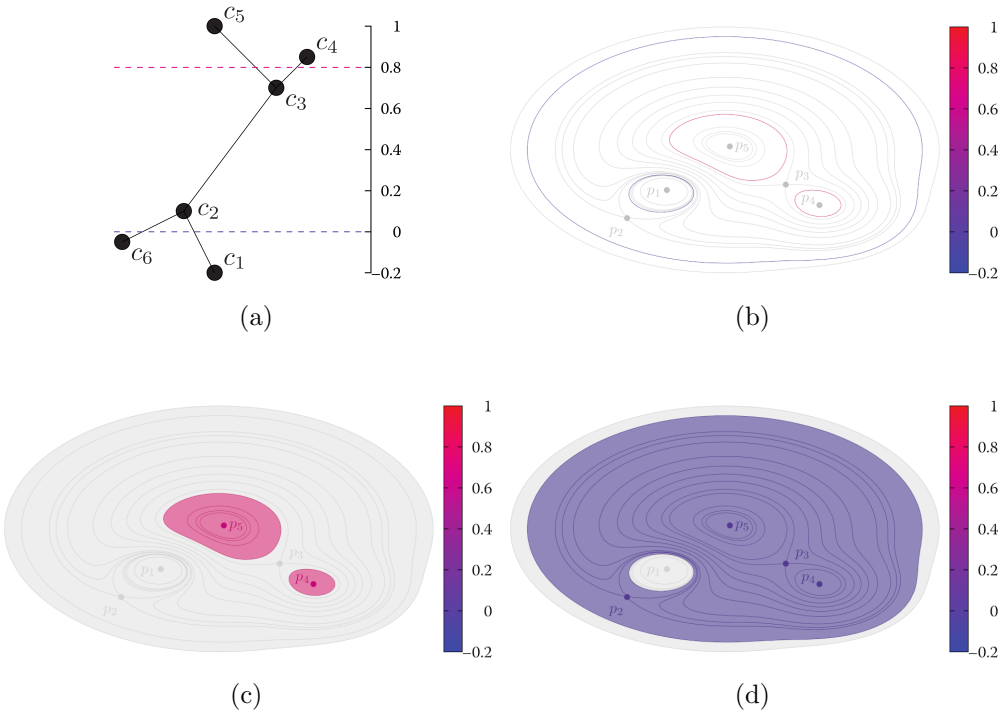
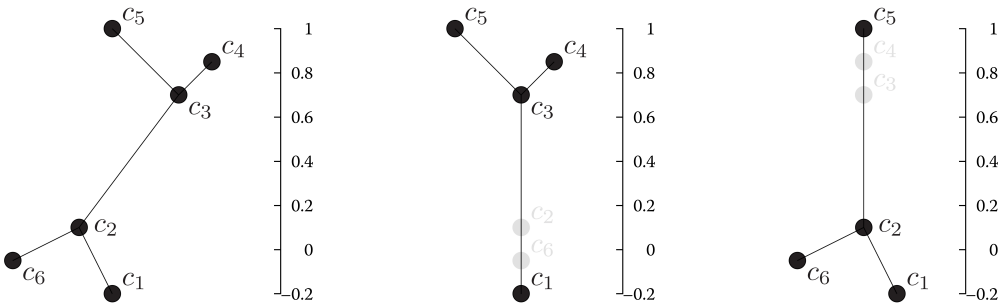


Figure 16: In (a), two horizontal cuts through the contour tree are indicated by the two dashed lines in magenta and purple. The corresponding contours are highlighted in (b). The coloured regions in (c) and (d) are the super-components of the magenta and violet level respectively, whereas the strict sub-components are shown in grey in both cases. Images adapted from Wenger [153].

contour tree and can be extracted easily. In essence, the contour tree can be decomposed into a join and a split tree. Conversely, the join and split trees can be merged to reconstruct the contour tree. In case of the join tree, the splitting of contours is disregarded. Symmetrically, the joining of contours is disregarded in the split tree. In Figure 17, the contour tree is shown with its corresponding join and split trees. The different subsets of topological changes captured by both trees actually represent the evolution of super-components and sub-components. In particular, the join tree represents the appearances and joinings of super-components and the split tree represents the splittings and



(a) Contour tree (b) Join tree (c) Split tree

Figure 17: The contour tree in (a) can be decomposed into a join tree in (b) and a split tree in (c). The grey nodes are *not* part of the respective trees since they do not represent topological changes that are captured each. Still, they are included to indicate which topological changes are disregarded compared to those contained in the contour tree.

disappearances of sub-components while sweeping from high to low levels [20, 24, 153]. The root node takes a separate role, since it does not necessarily correspond to a topological change at an extremum. Instead, it generally marks the level for which a single sub- or super-component covers the entire domain under consideration. It should be noted that in contrast to the contour tree, the join and split tree actually represent a nesting relations among sub- and super-components respectively.

3.5.2 Contour Tree for Discrete Data

The intuitive perspective of the contour tree in Section 3.5 has the advantage that it is very close to actual algorithms that compute the contour tree from discrete scalar data. It is assumed that such discrete data is given in the form of grid data (see Section 3.2), allowing for reconstructions of continuous scalar fields or its interpretation as a digital picture. In both cases, algorithms rely on an ordered processing of vertices according to their associated data values. However, since the data values are not guaranteed to be unique, the order of vertices is ambiguous. A solution to this problem is to simulate unique data values by means of perturbation [37]. For the course of this thesis, a simple

symbolic perturbation is employed that defines a lexicographic order [20, 24, 108, 153]: If two vertices $\boldsymbol{v}_{p_1,p_2,...,p_n}, \boldsymbol{v}_{q_1,q_2,...,q_n} \in V$ have equal scalar values $f_{p_1,p_2,...,p_n} = f_{q_1,q_2,...,q_n}$, the indexation of both vertices breaks the tie. Therefore each index tuple $(p_1, p_2, ..., p_n)$ is converted into a linear index

$$p_{linear}(p_1, p_2, ..., p_n) = p_1 + k_1 \cdot (p_2 + k_2 \cdot (... + n_{n-1} \cdot p_n)...)$$

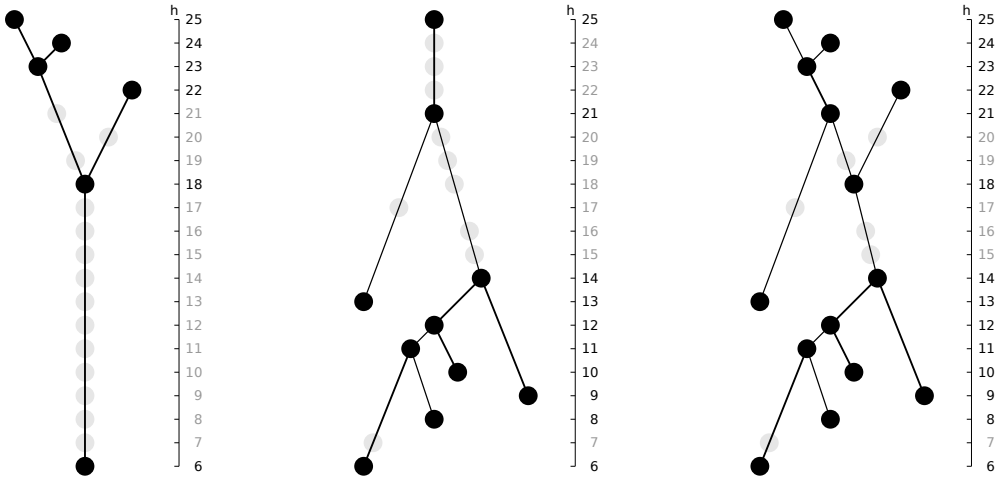
which is guaranteed to be unique among all vertices. Hence, $\boldsymbol{v}_{p_1,p_2,...,p_n} < \boldsymbol{v}_{q_1,q_2,...,q_n}$ if $p_{linear}(p_1, p_2, ..., p_n) < p_{linear}(q_1, q_2, ..., q_n)$. A processing of vertices according to their lexicographic order is equivalent to an actual slight variation of all data values which otherwise would have been necessary as a preprocessing step.

Contour Tree for Piecewise Linear Scalar Fields

Under the assumption of a simplicial grid (see Figure 5), the critical points of a reconstructed piecewise linear scalar field $\tilde{f}_{PL}$ are located at the vertices of the grid. This is due to the cell interpolant being linear across each grid cell which excludes the occurence of critical points in its interior [108, 153].[28] Consequently, levels at which the topology of level sets changes are among the data values associated with the vertices of the grid. In the intervals between those values, all contours are guaranteed to evolve linearly. Hence, a processing of vertices is sufficient, since all topological changes are guaranteed to be captured [20, 108]. Moreover, a processing of vertices in descending lexicographic order reflects the aforementioned sweep from high to low levels in a discrete sense.

Algorithm by Carr et al. In fact, the algorithm presented by Carr et al. [21] computes the contour tree by means of two sweeps through the vertex set. One sweep processes the vertices in descending lexicographic order and constructs a preliminary join tree (see Figure 18a) which is described by Algorithm 1. Another sweep processes the vertices in ascending lexicographic order and

[28] Morse theory requires all critical points to be isolated regarding their position as well as their values. When a piecewise linear scalar field is reconstructed from a simplicial grid, an isolation regarding the position is implicitly given. Due to the simulation of unique data values, critical points are virtually isolated regarding their values as well.



(a) Join Tree (b) Split Tree (c) Contour Tree

Figure 18: The (augmented) join and split trees for the piecewise linear scalar field induced by data values given over a simplicial grid as in Figure 5 are depicted in (a) and (b). Critical nodes in black are augmented by regular nodes in grey. Merging both trees yields the (augmented) contour tree in (c).

constructs a preliminary split tree (see Figure 18b) in a completely symmetric procedure.[29] Each preliminary tree comprises nodes for all grid vertices. In case of the join tree, nodes that correspond to topological changes of the superlevel set are critical ones. Symmetrically, critical nodes in the split tree correspond to topological changes of the sublevel set. The remaining nodes in each tree are regular ones. The preliminary trees are called the *augmented join* and *split tree* since critical nodes are augmented by regular nodes. Eventually, both trees are merged to form the *augmented contour tree* as depicted in Figure 18c. Its nodes represent all contours that pass through the vertices of the grid which includes critical as well as regular contours. The final (unaugmented) contour

[29] It should be noted that the name *split tree* is tied to the idea of sweeping from high to low levels while capturing splittings and disappearances of sub-components. When constructed algorithmically by processing the vertices in ascending lexicographic order, sub-components actually appear and join instead. In fact, from an algorithmic point of view, join and split trees both capture appearances and joinings of connected components. Due to the opposite directions in which the vertices are processed, connected components correspond to sub-components in the split tree and to super-components in the join tree.

Algorithm 1 Construction of the Join Tree

Input: Grid data with vertices sorted in lexicographic order $\boldsymbol{v}_0, \boldsymbol{v}_1, ..., \boldsymbol{v}_{k-1}$
Output: Join tree

```
1: initialize disjoint-set structure SC   // keeps track of super-components
2: initialize array LowestNodeOfComp of length n
3: initialize graph JoinTree with nodes w_0, w_1, ..., w_{k-1}
4: for i ← k − 1 to 0 do
5:    SC.createSet(i)
6:    LowestNodeOfComp[i] ← w_i
7:    for each vertex v_j adjacent to v_i in any order do
8:       if j < i or SC.find(j) = SC.find(i) then
9:          skip iteration
10:      else
11:         SC.merge(i, j)
12:         add edge between LowestNodeOfComp[j] and w_i to JoinTree
13:         LowestNodeOfComp[j] ← w_i
14:      end if
15:   end for
16: end for
```

tree results from the removal of all regular nodes by a series of homeomorphic contractions. The augmented join and split trees can be reduced in the same way to yield their unaugmented versions.

The assumption of a simplicial grid manifests itself during the computation of the join and split tree. More precisely, in line 7 of Algorithm 1, the loop iterates over all vertices that are adjacent to the currently processed vertex. Despite the adjacent vertices that are inherent in the uniform grid, additional vertices are considered adjacent, which reflects the implicit *Coxeter-Freudenthal-Kuhn triangulation* [28, 42, 78] (see Section 3.2.1).

While the vertices are processed, a disjoint-set data structure [142] is utilized to keep track of connected components of vertices. Therefore, it maintains the partition of an abstract set into disjoint subsets by means of storing different labels for each subset. In the context at hand, the labels come from the range of indices that determine the lexicographic order of vertices. The data structure provides methods for creating new sets (createSet, see line 5), merging the

sets of two elements (MERGE, see line 11) and finding the label of a set containing a given element (FIND, see line 8). The last operation allows for an efficient query if two elements are in the same set [26, 142]. CREATESET adds a new element to the partition. More precisely, the element is placed into a new elementary set[30] and this set is added to the data structure. The label of an elementary set is its only element. MERGE replaces two sets by their union. For the sake of simplicity, the label repesenting the merged sets is assumed to be the label of the set passed to the method as the second argument.

The relationship between the disjoint-set data structure, the connected components of vertices as well as the evolution of super-components – which only holds under the assumption of a simplicial grid – can be explained by how often the else-branch is entered during the for-each loop in Algorithm 1.

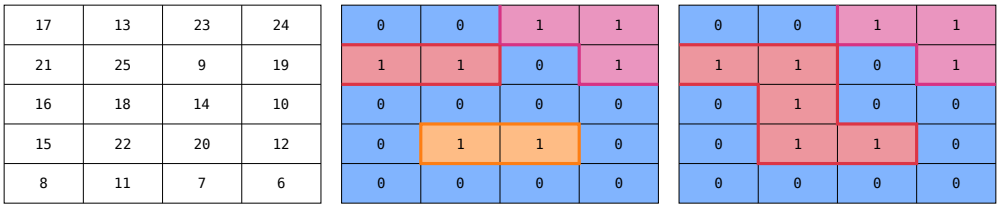
- When the else-branch is not entered for any of the adjacent vertices, the current vertex represents a maximum and a new set is created priorly due to line 5. The corresponding node in the join tree is destined to be a critical node, more precisely a leaf, that represents the appearance of a new super-component.

- When the else-branch is entered only once for all adjacent vertices, this means that an existing set is extended by one element. In this case, the corresponding node in the join tree is a regular node.

- When the else-branch is reached for two (or more) adjacent vertices, this means that the current vertex is the joining element between two priorly disjoint-sets. The corresponding node is a critical node that represents the joining of two (or more) super-components.

At the end of the outer loop, all vertices are processed and likewise merged into one set. The join tree can be thought of as a visualization of different snapshots of the disjoint-set data structure, and thus, represents the evolution of connected components of vertices. The equivalence to the evolution of super-components only holds under the assumption of a simplicial grid [20, 24, 108, 153].

[30] An elementary set is a set that only contains a single element.



(a) Digital picture (b) $h = 19$ (c) $h = 18$

Figure 19: Between the two levels $h = 19$ in (b) and $h = 18$ in (c) a topological change of the upper level set as well as the digital level set occurs. The red and orange coloured upper components and digital contours undergo a critical transition, i.e. a join when imagining a sweep from high to low levels.

Prior Work Prior to the work of Carr et al. [21], de Berg & van Kreveld [7] were the first who published a computation of the 'new' approach to the contour tree based on Morse theory directly from discrete elevation data without the interim computation of the surface network as in the work of Takahashi et al. [140]. De Berg & van Kreveld state a runtime in 2D with $\mathcal{O}(N \log N)$, where N is the number of grid cells. A subsequent publication by van Kreveld et al. [77] simplified the algorithm for 2D while keeping the overall runtime. More importantly, they proposed an algorithm that can work with discrete data in arbitrary dimensions which enables the contour tree to be used for many other applications that go beyond the two-dimensional case. For arbitrary dimensions, their algorithms performed with a runtime of $\mathcal{O}(N^2)$. Tarasov & Vyalyi [141] improved the runtime in 3D to $\mathcal{O}(N \log N)$. Finally, a universal and elegant solution to the computation of contour trees in arbitrary dimensions was presented by Carr et al. [22] that serves as the reference for this thesis. The authors state a runtime of $\mathcal{O}(n \log n + N)$ in arbitrary dimensions, where n is the number of vertices and N is the number of grid cells.

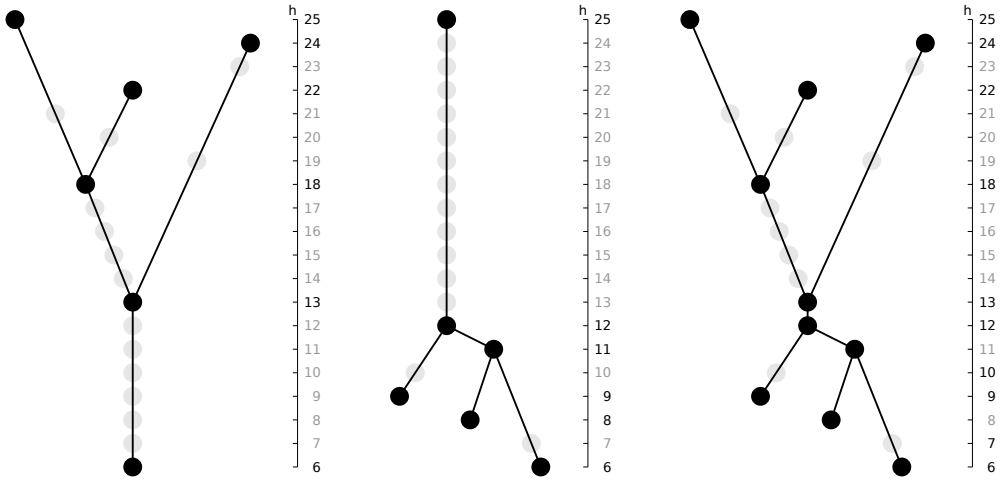
Contour Tree for Digital Pictures

When grid data is interpreted as a digital picture, no assumption regarding interpolation has to be made. Instead, vertices are understood as spels, which are defined in Section 6 as uniform space-filling elements centered at the vertex

positions (see Figure 19a). Given a partition of a digital space into a strict lower and an upper level set, a digital level set represents the oriented intersections of spels from both sets whereas digital contours correspond to $(n-1)$-dimensional surfaces between strict lower- and upper-components. In Section 3.4.2, a binary picture as defined in Equation 3.7 is employed for an intuitive idea of digital level sets. All possible states of digital level sets can then be described by a finite number of binary pictures each of which defined by a threshold that induces a different composition of 0s and 1s.

In contrast to a continuous setting, digital contours as well as (strict) lower-components and (strict) upper-components do not evolve continuously while the level is varied. Instead they undergo discontinuous transitions whenever the level sweeps across a data value that is associated with a spel. Hence, topological changes of the respective set manifest itself in form of those transitions before and after which its topology differs [27, 93]. For the sake of consistency, such transitions are referred to as *critical transitions* whereas the others are called *regular transitions*. As described in the last sections, topological changes of level sets are intrinsically linked to topological changes of the sublevel set and/or the superlevel set. Likewise, critical transitions of the strict lower and/or the upper level set induce critical transitions of digital level sets. An example of a critical transition is depicted in Figure 19b and 19c. Under the assumption that a data value involved in a critical transition is swept across from a higher to a lower level, *critical spels* in digital pictures can be characterized as follows:

- A digital maximum is a spel that constitutes the appearance of an upper-component as well as a digital contour.
- A digital minimum is a spel that constitutes the disappearance of a lower-component as well as a digital contour.
- A digital saddle is a spel that constitutes the join of two (or more) upper-components and/or the split of a lower-component into two (or more) lower-component. Likewise, two (or more) digital contours join and/or a digital contour splits into two (or more) digital contours.



(a) Join Tree (b) Split Tree (c) Contour Tree

Figure 20: In (a), (b) and (c), the 'digital' versions of the trees in Figure 18 are shown. They result from the algorithm of Carr et al. [22] applied to the digital picture shown in 19a.

Algorithm Mizuta & Matsuda [93] define the necessary digital counterparts of those concepts in Morse theory that are necessary to apply the algorithm of Carr et al. [21] to digital pictures.[31] Besides the interpretation of vertices as spels, the only modification of the algorithm concerns the aforementioned assumption of α-connected lower-components and ω-connected upper-components. As stated in Section 3.5.1, the join and split trees represent the evolution of sub- and super-components. Transferred to digital pictures, the 'digital' join and split trees represent the evolution of lower- and upper-components. Hence, the 'digital' version of the algorithm of Carr et al. [21] operates under the premise of using α-adjacency during the construction of the augmented join tree (see line 7 of Algorithm 2) and ω-adjacency during the construction of the augmented split

[31] Prior to the work of Mizuta & Matsuda [93], Cox et al. [27] presented a similar but more extensive approach to a digital equivalent of Morse theory under the name *Digital Morse Theory* (DMT). It should be noted that from the perspective of DMT, data values are not required to be unique. However, to keep a straightforward analogy to the continuous setting and at the same time avoid those tedious cases that arise from the admission of duplicates, unique data values are assumed still.

Algorithm 2 Construction of the 'Digital' Join Tree

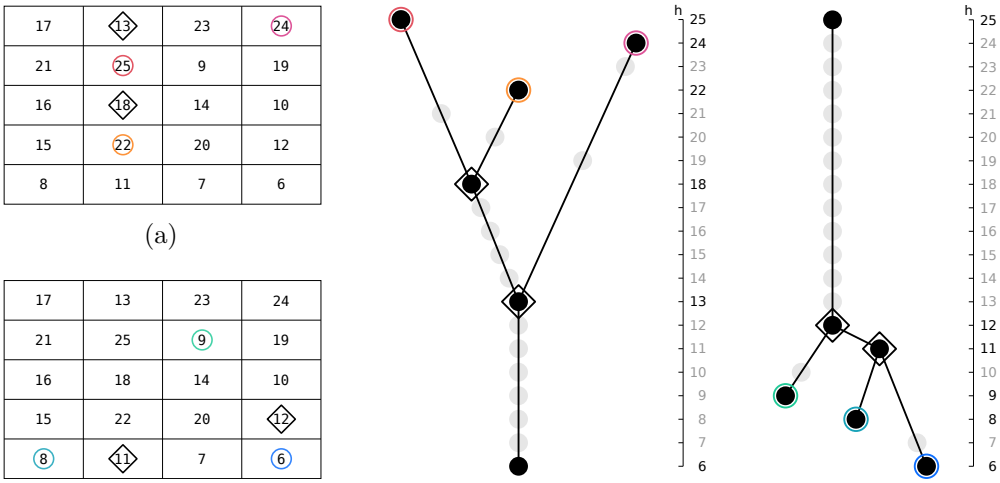
Input: Grid data with vertices sorted in lexicographic order $\boldsymbol{v}_0, \boldsymbol{v}_1, ..., \boldsymbol{v}_{k-1}$
Output: Digital join tree

```
1: initialize disjoint-set structure UC   // keeps track of upper-components
2: initialize array LowestNodeOfComp of length n
3: initialize graph JoinTree with nodes w_0, w_1, ..., w_{k-1}
4: for i ← k − 1 to 0 do
5:    UC.createSet(i)
6:    LowestNodeOfComp[i] ← w_i
7:    for each spel P_j ω-adjacent to P_i in any order do
8:       if j < i or UC.find(j) = UC.find(i) then
9:          skip iteration
10:      else
11:         UC.merge(i, j)
12:         add edge between LowestNodeOfComp[j] and w_i to JoinTree
13:         LowestNodeOfComp[j] ← w_i
14:      end if
15:   end for
16: end for
```

tree [93]. Figure 20 depicts the (augmented) join, split and contour trees for the exemplary digital picture shown in Figure 19a.

As stated for the algorithm by Carr et al. [21], the augmented join and split trees each comprise nodes for all vertices. When grid data is interpreted as a digital picture, this means that each node corresponds to a particular spel. In case of the digital join tree, nodes that correspond to critical spels that are involved in a critical transition of the upper level set are critical ones. Symmetrically, critical nodes in the digital split tree correspond to critical spels that are involved in a critical transition of the lower level set. The remaining nodes in each digital tree are regular ones corresponding to regular transitions of the respective set. In Figure 21, critical nodes in the respective trees and the corresponding critical spels in the digital picture are highlighted.

In the field of digital image processing, the augmented join and split trees are already known independently from their contribution to the contour tree as *min* and *max trees* [127]: Although each node in a 'digital' augmented tree



(a) (b) (c) Join tree (d) Split tree

Figure 21: In (a), spels corresponding to critial nodes of the join tree (excluding the root node) are highlighted as *maxima* (○) and *saddles* (◇). In (b), spels corresponding to critial nodes of the split tree (excluding the root node) are highlighted as *minima* (○) and *saddles* (◇).

represents a single spel, each node can be thought of to logically represent a lower- or upper-component. Starting from a node, the component its spel is connected to grows out of the recursive union of spels represented by child nodes [57].

3.6 Segmentation of Regions

For the further course of this thesis, the interpretation of grid data as a digital picture is assumed. Hence, regions are defined by levels that either determine the extent of lower-components or upper-components. In the context of digital pictures, segmentation is regarded as the assignment of numerical labels to the spels of a digital space. Under this premise, a binary picture is a simple example of a segmentation only comprising two different labels. Usually, the assignment of 0s and 1s is chosen such that the subset of spels labelled as 0 defines the background while the subset labelled as 1 defines the foreground. However, in

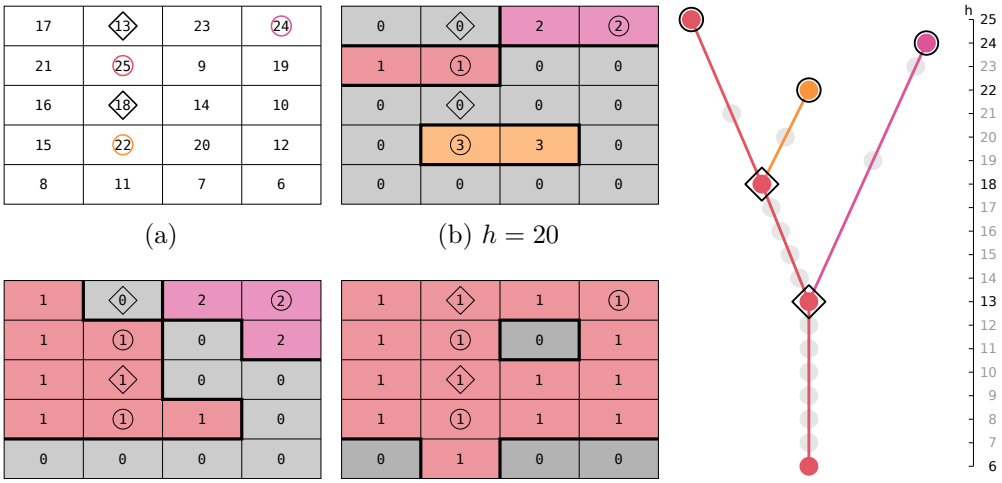


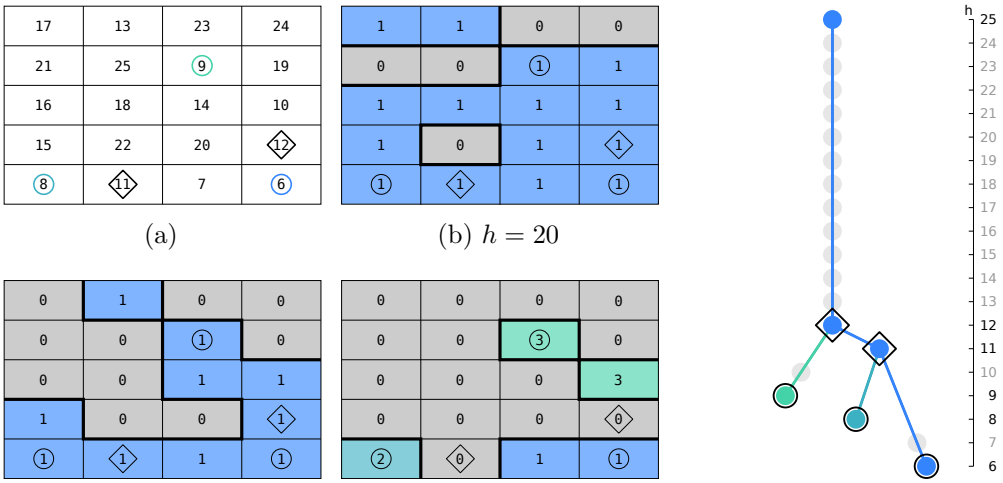
Figure 22: CCLs of the upper level set in (b)–(c) for three different levels. In (a), spels are highlighted as in Figure 21a. Moreover, the colouring of the join tree in (e) intuitively represents how labels of upper-components evolve over CCLs for all possible level sets.

many situations, a labelling of spels that expresses more information than fore- and background is used.

Connected Components Labelling (CCL) A more advanced segmentation, given a threshold at some level h, results from a labelling either of lower-components or upper-components. For this task, multiple algorithmic strategies exist that all fall under the term *connected component labelling* (CCL). Figure 23 and 22 show CCLs of the upper and lower level set, each for the same three levels. In Figure 23e and 22e, the unaugmented trees are coloured to intuitively represent how labels of upper- and lower-components evolve.

3.6.1 Watershed Segmentation

Another option for a segmentation of a digital picture is to additionally incorporate the concept of *watersheds* that will result in a finer segmentation of regions into subregions. In the context of a *watershed segmentation* (WS) those

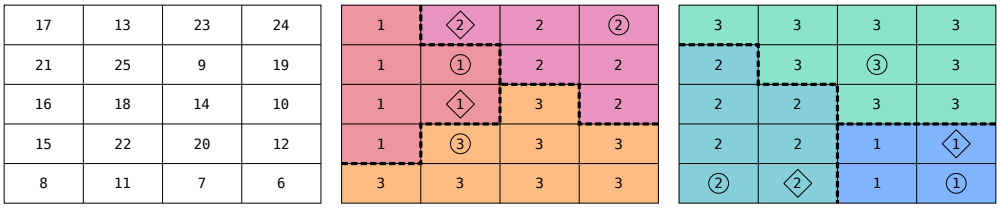


(a) (b) $h = 20$ (c) $h = 15$ (d) $h = 10$ (e) Split tree

Figure 23: CCLs of the lower level set in (b)–(c) for three different levels. In (a), spels are highlighted as in Figure 21b. Moreover, the colouring of the split tree in (e) intuitively represents how labels of lower-components evolve over CCLs for all possible level sets.

subregions are called catchment basins. For an intuitive understanding of a WS, the field of topography can be taken up once more. A two-dimensional digital picture can be thought of as a terrain that is constituted by rectangular plateaus of constant altitudes. For the purpose of illustration, holes are supposed to be pierced in this topographic surface at the locations of each minimum. The algorithm simulates the effect of gradually immersing the surface into a bath of water: Starting from the minima of lowest altitude, the water will progressively fill up the catchment basins corresponding to different minima. Wherever water from two adjacent catchment basins would mix, a dam is erected to keep the basins separate. At the end of this immersion process, the whole surface is flooded and each minimum can be associated with its own catchment basin. The dams constitute the dividing lines of water also known as watersheds.

In the well-known publication by Vincent & Soille [147], that paved the way for many WS-based applications as well as implementations, watersheds are introduced as a subset of pixels. The result of a WS is then a segmentation of



(a) Digital picture (b) IWS (c) WS

Figure 24: A digital picture in (a) with the corresponding full IWS in (b) and full WS in (c). Extrema and saddles are highlighted as in the previous figures.

catchment basins as well as watersheds by means of different labels assigned to pixels. However, in the same publication, the authors introduce the use case of partitioning the entire digital space into catchment basins. Transferred to the idea of a digital picture describing a terrain that is constituted by plateaus of constant altitude and immersed into water, a mixing of waters from two basins would take place by flooding certain pixels as a whole and at once. Instead of labelling each of these pixels as being part of a watershed, but to actually get a partition of all pixels into regions, the pixel in question is labelled according to either of the two catchment basins involved. Different strategies to optimally decide to which of the two regions the vertex is assigned are feasible. Since the different segmentations resulting from different strategies do not determine the resulting segmentation conceptually, the different options are not covered in this thesis. For the sake of simplicity, it is henceforth assumed that a pixel is assigned to the regions of the adjacent vertex with the highest data value. The crucial aspect is that two regions are kept separate due to the concept of watersheds instead of merging them and therefore becoming one connected component. Based on the terminology of spels and surfels and at the same time generalizing the WS to arbitrary dimensions, watersheds can be defined by sets of surfels that determine a partition of spels into regions. It should be noted that under the assumption of ω-connected (α-connected) regions, a single watershed is a digital boundary between a region and α-connected (ω-connected) components of a region's complement.

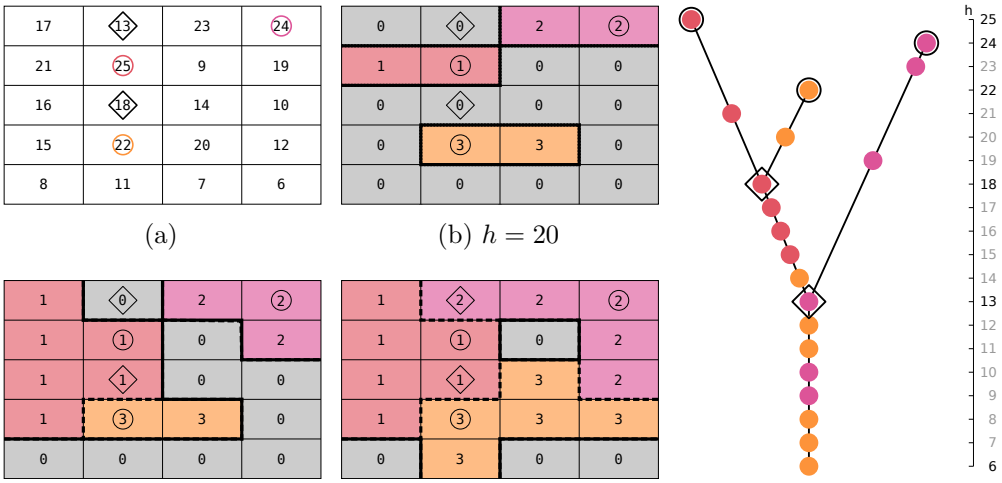
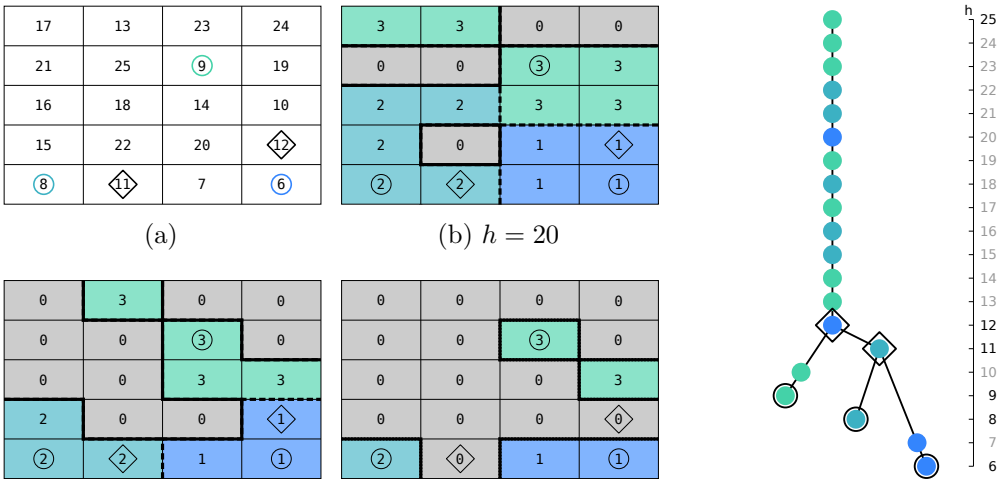


Figure 25: IWS in (b)–(c) for three different levels. In (a), spels are highlighted as in Figure 21a. The nodes of the augmented join tree in (e) are coloured according to their label assigned by the full IWS shown in Figure 24b.

The WS as explained above implies a progressive labelling of spels from the lower level set while the level is varied from low to high levels. Despite the topographic analogy of a rising water level not applying anymore, it is from an algorithmic point of view unproblematic to define an *inverse watershed segmentation* (IWS) that progressively labels spels from the upper level set while the level is varied from high to low levels. The non-topographic analogy can be resolved as the IWS can be imagined as if the water level would flood a topographic surface that results from the multiplication of the original data values with -1 such that peaks become pits and vice versa. Moreover, the explanation above implies a partition of the whole domain into regions which is depicted for both IWS and WS in Figure 24. Usually, IWSs and WSs are computed with respect to a threshold at some level within the range of data values such that some parts of the digital space are left unsegmented. Taking up the metaphor of immersing a topographic surface into a bath of waters, this



(a) (b) $h = 20$ (c) $h = 15$ (d) $h = 10$ (e) Split tree

Figure 26: WS in (b)–(c) for three different levels. In (a), spels are highlighted as in Figure 21b. The nodes of the augmented split tree in (e) are coloured according to their label assigned by the full WS shown in Figure 24c.

would mean that the water rises only to a certain level. Figures 25 and 26 show the result of an IWS and a WS, each for three different levels.

It should be noted how for the exemplary two-dimensional grid data, IWS and WS in Figures 25 and 26 generally result in a segmentation of lower and upper level sets into additional subregions in constrast to the CCLs in Figures 22 and 23. Consequently, the watersheds for a given level comprise additional boundaries between spels compared to the digital level set which is indicated by dashed lines in Figures 25 and 26.

The main purpose of Figures 25e and 26e is to illustrate that the join and split trees do *not* represent the evolution of catchment basins. In contrast to a CCL, two upper-components do not join at a digital saddle. Hence, the common edge in the join tree after two edges joined at a saddle does not represent a single upper-component but the parallel evolution of multiple catchment basins that are kept separate by means of watersheds. The parallel evolution is indicated

by the colouring of nodes in the augmented join and split trees in contrast to the colouring of the unaugmented join tree in Figures 22e and 23e.

Hierarchical Watershed Segmentation A typical problem of a classical watershed segmentation is that the resulting segmentation is too fine, which is called oversegmentation. Obviously this is due to the data typically comprising noise, such that many small local extrema exist, each of which will get its own associated regions in the final segmentation. To overcome this problem, many strategies exist to allow for a coarser partition of regions. A particular group of strategies falls under the name *hierarchical watershed segmentation* [8, 40, 101].

A segmentation is hierarchical if a criterion defines different detail levels in which the segmentations at coarser levels can be produced from simple merges of (adjacent) regions from segmentations at finer levels. Attributes of each two adjacent regions can be evaluated to decide on the merge. Examples are geometrical attributes [45] such as e.g. the size of regions [81] or related to the intensities belonging to regions [45, 101]. The labelling at a coarser level results from the merge of adjacent regions that afterwards share the same label. Therefore, the regions at finer levels are nested in relation to those at coarser levels [47]. In that sense does a hierarchical watershed segmentation not prevent an oversegmentation but instead supresses irrelevant watersheds [8]. One can imagine the hierarchical watershed segmentation as if dams would be tiered down such that waters from adjacent catchment basins can mix.

3.6.2 Hierarchical Implementation

In this thesis, the focus is on an algorithmic solution for the hierarchical WS and IWS that is based on the 'digital' join and split trees. Note that the algorithms for the hierarchical WS and IWS are completely symmetrical since the computation of join and split trees is symmetrical as well.[32] Therefore, only the algorithm and hierarchical IWS is presented in the following.

[32] From the perspective of an actual implementation, a single algorithm can be implemented that is parametrized by a flag variable that decides whether the result is a WS or an IWS.

Algorithm 3 Inverse Watershed Segmentation (IWS)

Input: Grid data with vertices sorted in lexicographic order $\boldsymbol{v}_0, \boldsymbol{v}_1, ..., \boldsymbol{v}_{k-1}$
Output: IWS by means of disjoint-set structure

```
 1: initialize disjoint-set structure CB   // keeps track of catchment basins
 2: for i ← k − 1 to 0 do
 3:    CB.makeSet(i)
 4:    for each spel P_j ω-adjacent to P_i in descending order do
 5:       if j < i then
 6:          skip iteration
 7:       else
 8:          CB.merge(i, j)
 9:          leave inner loop
10:       end if
11:    end for
12: end for
```

Referring to Algorithm 2, in particular the case where the else-branch is reached for two (or more) adjacent vertices, the vertex currently processed is the joining element between two (or more) upper-components. Transferred to the idea of a watershed segmentation, this leads to a mixing of waters from two catchment basins taking place at the earliest possibility. In fact, with respect to the idea of watersheds separating catchment basins, Algorithm 2 results in a segmentation where no catchment basins are ever separated by watersheds. Accordingly, at the end of the algorithm the disjoint-set data structure only comprises a single set representing all spels. If the purpose is not to construct the 'digital' join tree but instead to respect the concept of watersheds during updates of the disjoint-set data structure, this means particularly that two or more merges during the inner for-loop are prohibited. A modified version of Algorithm 2 that gives rise to an IWS is presented in Algorithm 3.[33]

As mentioned before, a hierarchical segmentation constructs segmentations at coarser detail level depending on some criterion that controls the merging of regions from segmentation at finer levels. A fast implementation of a hierarchical

[33] It should be noted that for the purpose of visualization, the segmentation shown in Figure 24b results from an alternative order in which adjacent spels are processed compared to what is assumed in line 4 of Algorithm 3.

Algorithm 4 Prerequisites for a Fast IWS

Input: Grid data with vertices sorted in lexicographic order $\boldsymbol{v}_0, \boldsymbol{v}_1, ..., \boldsymbol{v}_{k-1}$
Output: Prerequisites for a fast IWS

```
1: initialize disjoint-set structure UC   // keeps track of upper-components
2: initialize disjoint-set structure CB   // keeps track of catchment basins
3: initialize array LowestNodeOfComp of length k
4: initialize array MaxOfBasin of length k
5: initialize graph JoinTree with nodes w_0, w_1, ..., w_{k-1}
6: for i ← k − 1 to 0 do
7:    UC.createSet(i)
8:    LowestNodeOfComp[i] ← w_i
9:    CB.createSet(i)
10:   MaxOfBasin[i] ← f(P_i)
11:   assignedToBasin ← false
12:   for each spel P_j ω-adjacent to P_i in descending order do
13:      if j < i or UC.find(j) = UC.find(i) then
14:         skip iteration
15:      else
16:         UC.merge(i, j)
17:         add edge between LowestNodeOfComp[j] and w_i to JoinTree
18:         LowestNodeOfComp[j] ← w_i
19:         if not assignedToBasin then
20:            CB.merge(i, j)
21:            assignedToBasin ← true
22:         end if
23:      end if
24:   end for
25: end for
```

IWS is based on the precomputation of the segmentation at the finest detail level as produced by Algorithm 3 as well as the construction of the join tree by means of Algorithm 2. The place where the waters from two basins would mix at the earliest possibility is a saddle of the digital space [101]. Hence, it is sufficient to evaluate attributes of adjacent regions at a saddle nodes of the 'digital' join tree to decide if both regions are merged. A processing of saddle nodes in descending lexicographic order (or ascending lexicographic order for a WS) guarantees that the segmentations at different detail levels are in fact nested inside of

Algorithm 5 Fast IWS

Input: Threshold p defining detail level of hierarchical IWS
Output: Fast IWS by means of disjoint-set data structure

```
1: create local copy of CB
2: for each saddle node w_i in JoinTree in descending order do
3:   for each spel P_j ω-adjacent to P_i in descending order do
4:     if j < i or CB.find(j) = CB.find(i) then
5:       skip iteration
6:     else
7:       saddleBasin ← CB.find(i)
8:       saddleDepth ← MaxOfBasin[saddleBasin] − f(P_i)
9:       adjacentBasin ← CB.find(j)
10:      adjacentDepth ← MaxOfBasin[adjacentBasin] − f(P_i)
11:      if min(saddleDepth, adjacentDepth) < p then
12:        if adjacentDepth < saddleDepth then
13:          CB.merge(i, j)
14:        else
15:          CB.merge(j, i)
16:        end if
17:      end if
18:    end if
19:  end for
20: end for
```

each other. As a criterion that controls the merging at saddles, the absolute difference between the maximal intensity value within a region and the intensity value associated with a saddle is used. More intuitively, this criterion can be thought of as the *depth* of a catchment basin. A threshold p then determines to merge adjacent basins if either of its depths is below the threshold. Fortunately, all the information that is necessary can be precomputed within a loop over all vertices as presented in Algorithm 4 which is basically a combination of Algorithm 2 and 3. The fast IWS making use of the precomputed information is presented in Algorithm 5.

The essential aspect of Algorithm 5 is that – given a watershed segmentation at the finest detail level – it is sufficient to revisit the saddle nodes of the join tree, which correspond to those spels at which two upper-components could

have been joined at the earliest possibility. The conditions in line 4 ensure that only distinct regions that have emerged at higher levels are considered adjacent. According to line 11, a merge depends on the depth of either of the two regions which must fall under a certain threshold p defining the level of detail.

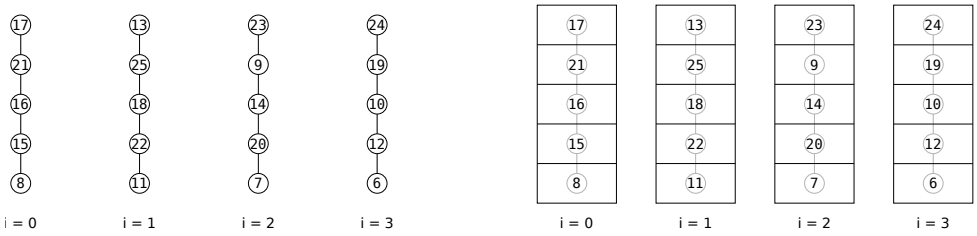
Fast IWSs that are constrained by thresholds at different levels as in Figure 25b–25d can be achieved by iterating the outer for-loop in Algorithm 5 only for saddle nodes whose associated data values are below the specified level. Finally, only the labels associated with spels whose data values exceed the level are exported to the final output image while the others are set to zero. A fast IWS with respect to a given threshold and for the coarsest detail level becomes a CCL.

4 Tracking Approach

In this chapter, the individual requirements that were identified in the introduction are adressed and it is shown how the components of a tracking method, which are (i) feature definition and extraction, (ii) correspondence identification, and (iii) the representation of feature tracks are implemented. As introduced in the beginning, time-dependent scalar data is given as temporal recordings of a spatial domain that is discretized by an n-dimensional uniform grid $U = (V, E)$. For each time step $i \in \mathbb{N}_t$, a finite set of scalar values F_i is associated with the vertices of the grid. The combination of the uniform grid with each set of data values (U, F_i) defines each time step in the form of grid data as described in Section 3.2. In the following, the tracking approach is described for arbitrary dimensions but exemplified for the case of 1D+time. Therefore, the two-dimensional example of grid data that repeatedly served as an illustration, is now altered for the purpose of representing four separate time steps of grid data each of which being one-dimensional as shown in Figure 27a.

4.1 Correspondence Identification

As pointed out in the introduction, the correspondence should be determined by all information of two successive time steps. In particular, this information should be used to estimate a fine-grained and dense motion between time steps that can be used universally for the identification of corresponding features at different levels. In that sense, the identification of correspondence is related to the approach of Valsangkar et al. [146]. However, in contrast to their approach, the motion of the entire domain is estimated independently of the actual extraction of features allowing for the motion estimation to be precomputed. Therefore, a fundamental assumption under which this tracking approach will



(a) Time steps of grid data

(b) Time steps of digital pictures

Figure 27: The former two-dimensional example of grid data is now used to represent four separate time steps of grid data in (a). Its interpretation as four one-dimensional digital pictures with a two-dimensional visualization of spels is shown in (b).

be presented is that the grid data of each time step (U, F_i) is interpreted as a digital picture $(S, \omega, \hat{f}_i)$ (see Section 6). Under this premise, spels – which are the basic units of a digital space – are the building blocks of which regional features in digital pictures are composed. Although the grid data in Figure 27a is one-dimensional, the spels in Figure 27b are visualized intentionally as if the digital picture would be two-dimensional with the length of the second dimension set to one. The extent in the second dimension is chosen such that it mimics the former two-dimensional example for the purpose of a consistent visual impression throughout this thesis.

4.1.1 Use Case for Image Registration

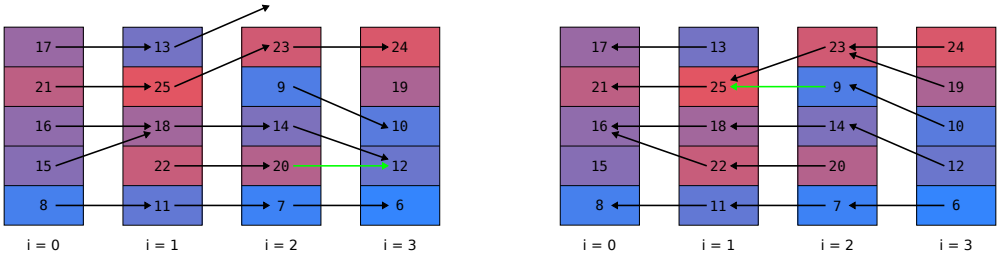
Under the assumption of interpreting grid data as a digital picture, an image registration method (see Section 3.3) can be employed to yield the spatial relations between each pair of successive digital pictures, $(S, \omega, \hat{f}_i)$ and $(S, \omega, \hat{f}_{i+1})$ for $i \in \mathbb{N}_{t-1}$. It should be noted that the two registration processes for each pair are independent of each other such that deformation fields $\boldsymbol{d}_i^{forw}$ and $\boldsymbol{d}_{i+1}^{back}$ are independet as well. For the purpose of tracking, the actual deformed moving image is not relevant but instead the deformation itself, which "is the focal point of any registration algorithm" [129]. The general idea underlying this thesis is that the deformation between two successive digital pictures, in form of a vector field henceforth referred to as *deformation field*, serves as an appropriate

motion estimation. Since image registration is a directed process, an unbiased motion estimation results from the combination of deformation fields in both directions. More specifically, for each pair of digital images, the registration process needs to be carried out twice to yield one deformation field pointing forward in time and another one pointing backwards in time. This can simply be achieved by switching the roles of the moving and fixed image in a second registration process. Hence, the registration of each pair results in $2 * (t - 1)$ deformation fields, $\boldsymbol{d}_i^{forw}$ and $\boldsymbol{d}_{i+1}^{back}$ for $i \in \mathbb{N}_{t-1}$.

Contrary to the interpretation of the deformation during registration as pointing towards positions in the moving image where the intensity values originate from, for the purpose of tracking, the forward and backward deformation fields are utilized to estimate the motion of each spel, i.e. the position where a spel moves to in the next and previous time step, respectively. Therefore, the deformation fields are snapped to the nearest neighbouring spel centers. Effectively, this turns the deformations into mappings between vertex centers which is utilized to establish correspondences between spels of successive time steps. In Figure 28, possible deformation fields for the exemplary four time steps are shown that are already snapped to pixel centers. In turn, the correspondences between spels are used as a basis for identifying correspondences among regional features which – in the context of digital pictures – are constituted by sets of spels.

Tracking-related Properties of the Deformation

As mentioned in Section 3.3, the deformation might also point to positions outside of the domain that is spanned by a picture, which is depicted in Figure 28a. Since these local manifestations of the deformation field do not result in any correspondences between spels, they are simply ignored for the purpose of tracking. Although the exemplary deformation fields in Figure 28 can be regarded as overall sufficient with respect to an intuitive notion of which spels 'belong' together, cases such as those indicated by the green arrows can nevertheless arise. The reason for such unintuitive correspondences is simple: The deformation fields resulting from image registration are dense by



(a) Forward deformation fields

(b) Backward deformation fields

Figure 28: Possible deformation fields for the four time steps of digital pictures that are snapped to the spel centers. Effectively, the forward deformations in (a) and the backward deformations in (b) describe correspondences between spel of successive time steps.

definition and represent the spatial relation between two images that minimizes an objective function for the entire image. For each spel center, a vector will be computed that points into the domain of the other image involed in the registration process. The spatial relation described in this way is optimal with respect to the entire image, yet it allows for local manifestations of the deformation field that are unintuitive if considered individually. This also applies to local deformations as enabled by the B-spline transformation model which is supposed to be the underlying model of the deformation fields in Figure 28. While local deformations are immanent in a B-spline transformation, the model also ensures a certain degree of smoothness of the entire deformation depending on the degree of B-spline basis functions. Additionally, a B-spline transformation is usually constraint by a regularization term that penalizes strong compression and stretching. Overall, the deformation of an individual spel is influenced by the deformation of spels in its surrounding. As a result, unintuitive deformations as those represented by the green arrows can simply considered to be natural phenomena arising from image registration.

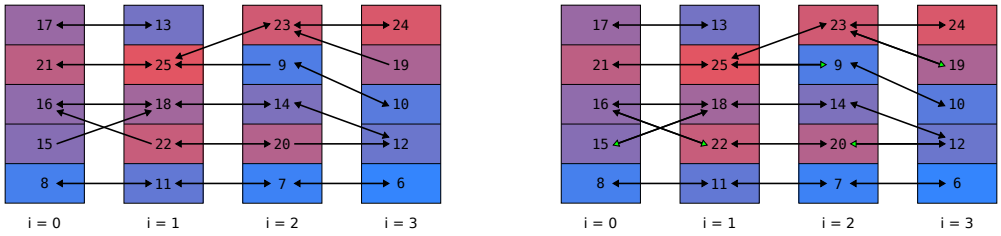
4.2 Feature Definition & Extraction

To account for the fact that features belonging to different scales might be nested inside of each other, features are defined and extracted as connected regions by means of a fast IWS. For the sake of simplicity, a presentation of the alternative approach defining and extracting features by means of a WS is ommitted since this case is completely symmetric to the one expounded hereinafter. As stated in Section 3.6.1, in the context of a watershed segmentation, connected regions are called catchment basins. To determine the scale up to which features are considered separate in contrast to merging them, the depth of catchment basins is used as a criterion to control the detail level of the segmentation. As a consequence, thresholding the criterion yields a hierarchical IWS which can be implemented efficiently as proposed in Section 3.6.1. In that sense, the definition and extraction of features is similar to the approach of Lakshmanan et al. [81], who described storm cells by means of regions related to catchment basins.

However, in the approach at hand, the concept of the watershed segmentation is combined with the strategy to consider time-dependent data not as separate time steps of grid data each of which interpreted as an individual digital picture but as 'one piece' of grid data that is interpreted as a spatio-temporal digital picture that can be used conveniently for the purpose of tracking. More specifically, the underlying spatio-temporal digital space is defined in such a way that the correspondence information resulting from image registration is used to define the adjacency among spels in the temporal dimension. As a result, features of different levels can be extracted and implicitly tracked simply by computing a fast IWS for the spatio-temporal digital picture. The assignment of labels to the spels of the spatio-temporal digital space then defines regions in each time step as well as feature tracks over time by means of regions from different time steps sharing the same spatio-temporal label.

4.2.1 Spatio-Temporal Digital Picture

As a general requirement for the construction of a spatio-temporal digital space, time-dependent data given as separate time steps each of which in the form



(a) (b)

Figure 29: Combined visualization of forward and backward deformation fields. In (a), symmetric correspondences between spels are indicated by double-headed arrows while asymmetric correspondences are represented by normal arrows. In (b), heads are added to the normal arrows highlighted in green to indicate the gain if all correspondences would be symmetric.

of n-dimensional grid data, (U, F_i) where $i \in \mathbb{N}_t$, is considered 'one piece' of $(n+1)$-dimensional grid data (U', F'). The data values in F' associated with the vertices of the $(n+1)$-dimensional uniform grid $U' = P_{k_1} \square P_{k_2} \square ... \square P_{k_n} \square P_t$ represent the data not as separate recordings of an n-dimensional Euclidean space but as recordings over an $(n+1)$-dimensional Euclidean space whose last dimension actually represents time. The number of time steps t and the interval s_{n+1} between time steps determine the resolution as well as the extent of the temporal dimension of the grid. Therefore, the neighbourhood of each vertex can be considered to contain vertices that are adjacent not only in space but also in time. For the purpose of tracking, the interpretation of the $(n+1)$-dimensional grid data as a digital picture slightly deviates from Section 6: In the spatial dimensions of the spatio-temporal digital space, the adjacency of spels coincides with the original ω-adjacency. In the newly added temporal dimension, the results of the precomputed registrations are integrated.

Formally, the spatio-temporal space that serves for the purpose of tracking is defined by the pair (S', ω') with the set of spels S' defined as stated in Equation 3.3.

$$S' = \{P_{p_1,p_2,...,p_n,p_{n+1}} \mid (p_1, p_2, ..., p_n) \in I_{U'}\}$$

The adjacency relation of spels in spatial dimensions resembles the definition of ω in Equation 3.5.

$$\omega_{space} = \{(P_{p_1,p_2,\ldots,p_n,p_{n+1}}, P_{q_1,q_2,\ldots,q_n,q_{n+1}}) \mid \sum_{i=1}^{n} |p_i - q_i| = 1\}$$

To determine the adjacency among spels in the temporal dimension, the correspondences between spels are utilized. Therefore, the forward and backward deformations must be considered with respect to the digital space (S, ω) that served as the basis for the digital pictures in each time step. Since the deformation fields $\boldsymbol{d}_i^{forw}$ and $\boldsymbol{d}_{i+1}^{back}$ are assumed to be snapped to spel centers, they can be effectively regarded as binary relations δ_i^{forw} and δ_{i+1}^{back} on the index set I_U (see Section 3.1) for $i \in \mathbb{N}_{t-1}$. The union of all binary relations resulting from registration is yet another binary relation R.

$$R = \bigcup_{i=0}^{n-2} \delta_i^{forw} \cup \delta_{i+1}^{back}$$

However, the relation R is not symmetric. A spel in time step i corresponding to a spel in time step $i+1$ by means of a forward deformation does not imply that the same spels correspond by means of the opposed backward deformation. Many such 'asymmetries' can be identified in Figure 29a. In a digital space however, the adjacency relation is symmetric by definition which means that a spel is adjacent to another spel if it is also adjacent from the other spel. Therefore, every asymmetric correspondence must be transformed into a symmetric adjacency by means of the symmetric closure C of R.

$$C = R \cup \{\ \big((q_1, q_2, ..., q_n), (p_1, p_2, ..., p_n)\big) \mid \big((p_1, p_2, ..., p_n), (q_1, q_2, ..., q_n)\big) \in R\}$$

In Figure 29b, the additional correspondences gained from the symmetric closure of R are indicated by green arrow heads. Finally, the adjacency between spels in the temporal dimensions, τ, can be defined by means of the symmetric relation C.

$$\tau = \{(P_{p_1,p_2,\ldots,p_n,p_{n+1}}, P_{q_1,q_2,\ldots,q_n,q_{n+1}}) \mid \big((p_1, p_2, ..., p_n), (q_1, q_2, ..., q_n)\big) \in C\}$$

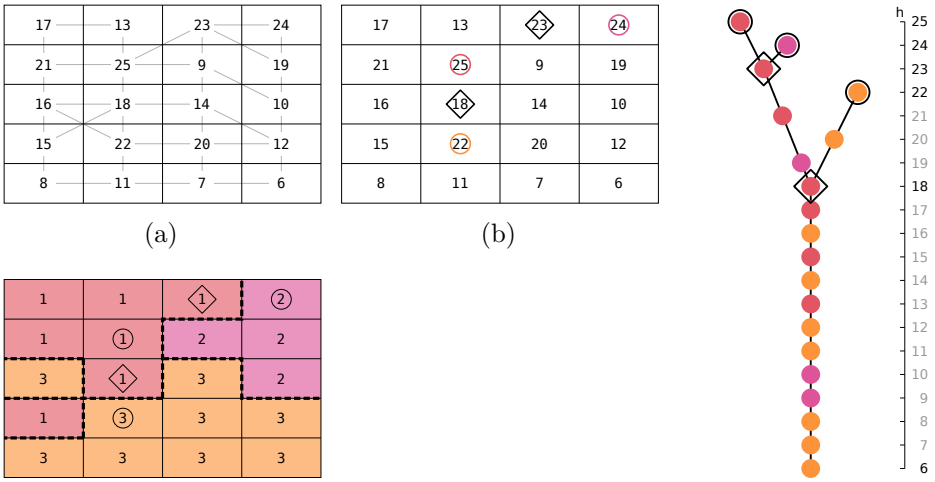


Figure 30: In (a), the spatio-temporal digital picture is shown with the underlying proto-adjacency ω' indicated in grey. Since the two-dimensional spatio-temporal digital picture exhibits a different adjacency relation compared to the former two-dimensional example, the corresponding join tree in (d) differs as well. In (b), spels corresponding to critial nodes of the join tree (excluding the root node) are highlighted once more as *maxima* ($\circ$) and *saddles* ($\diamond$). A spatio-temporal watershed segmentation is shown in (c).

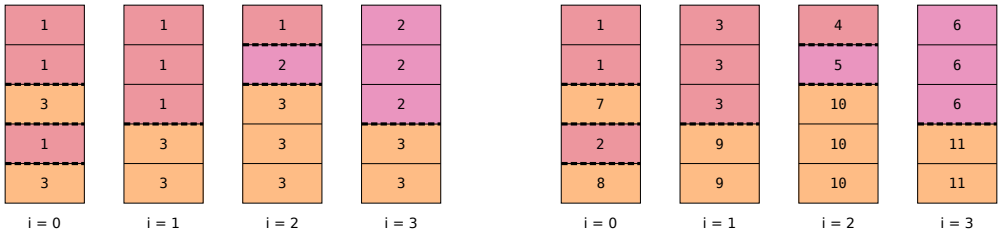
The proto-adjacency ω' constituting the spatio-temporal digital space then results from the union of the spatial and temporal adjacency.

$$\omega' = \omega_{space} \cup \tau$$

The triple $(S', \omega', \hat{f}')$ then defines a spatio-temporal digital picture which is shown for the recurring example in Figure 30a.

4.2.2 Spatio-Temporal Watershed Segmentation

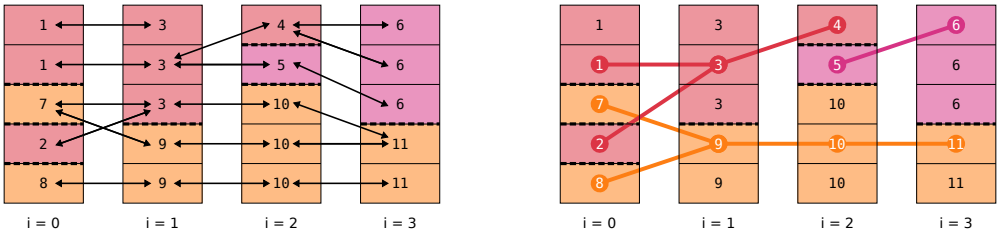
A fast IWS applied to the spatio-temporal digital picture $(S', \omega', \hat{f}')$ operates based on using ω' as a spel adjacency (see Section 3.4.2) during Algorithm 4 and 5. Hence, regions that have been assigned a common label by the segmentation are



(a) (b)

Figure 31: In (a), the spatio-temporal labels resulting from a IWS are considered in each time step separately. The first time step can be taken as an example showing that the spatio-temporal labels cannot be used to extract ω-components in each time step. Therefore, subsequent CCLs must be applied to those spels in each time step that share the same spatio-temporal label. The results are shown in (b) by means of a new labelling. At the same time, the colouring of spels represents the original spatio-temporal labels.

either connected with respect to the ω_{space}-adjacency in the spatial dimensions, or 'connected' regarding the τ-adjacency in the temporal dimension. Figure 30c shows an IWS at the finest detail level for the spatio-temporal digital picture in Figure 30a. If the spatio-temporal labels are then considered separately for each time step, the labelling only represents which spels belong to a common feature track (see Figure 31a). To extract the actual regional features in each time step, subsequent CCLs with respect to ω-connectedness must be applied to those spels in each time step that share the same spatio-temporal label. These components are henceforth referred to as *track components*. In Figure 31b, track components can be identified by a new labelling while the colouring of spels still indicates which track components share the same spatio-temporal label. When the watersheds, indicated by the dashed lines in Figure 30c, are considered in each time step separately, they are in fact digital boundaries between an ω-connected track components and α-connected components of a track-component's complement. Unfortunately, in the case of $n = 1$ used for the illustrations, ω- and α-adjacency coincide as simple 2-adjacencies. Without giving proof it is claimed that, for arbitrary dimensions, the track components in each time step are bounded by digital boundaries as defined by Herman [55].



(a) (b)

Figure 32: In (a), the labelling of track components from Figure 31b is shown in combination with the symmetric correspondences between spels. Based on this information, the resulting tracking graph with a connected component for each spatio-temporal label is illustrated in (b).

4.3 Representation of Feature Tracks

The labelling of track components as shown in Figure 31b can be used directly to examine properties such as the number, area, mean and maximal intensities of features. Furthermore, due to the spatio-temporal labels resulting directly from the IWS, properties such as those mentioned before can be aggregated and compared over the life-cycle of feature tracks. In addition, the spatio-temporal dynamic can be described by a spatial tracking graph embedded in the spatio-temporal digital space that represents the evolution of features and captures changes in their topological structure. Each track-component is represented by a single node whose position is chosen to be the spel center of a track-component's maximum. The nodes of two track components that share a common spatio-temporal label are connected by an arc if the track components are connected by means of correspondences among their spels (see Figure 32). The resulting connected components of the tracking graph indicate the appearances and disappearances of features as well as their spatio-temporal dynamic during their lifetime exhibiting joins and splits. Particular topological events can be identified based on the degree of tracking graph nodes.

5 Results

The tracking method developed in this thesis and described in Chapter 4 is applied to the precipitation data of the COSMO-REA2 system introduced in Chapter 2.1. In this chapter, the results are presented. As pointed out in the last chapter, the tracking results are determined fundamentally by the deformation fields which in turn result from image registration. In fact, finding a sufficient set of parameters for the registration of precipitation data can be considered a task on its own which is independent of the tracking approach in the first place. This independence is also stressed by the fact, that the registration-based motion estimations are expected to be precomputed. They are intended to be universally usable within the tracking method with regard to feature tracking at different levels.

Therefore the results are twofold: First, the parametrization of elastix that was found to be sufficient for the considered datasets is presented in Section 5.1. Subsequently, the results of the tracking approach are adressed in Section 5.2 with respect to the requirements that were mentioned in the introduction.

Datasets

The first dataset considered is publicly available with an hourly temporal resolution and spans the entire month March 2007 in form of 744 time steps (dataset A). Originally, the spatial extent covers the entire domain of the COSMO-REA2 system discretized by a uniform grid with 724 x 780 vertices. Preliminary studies by the Institute of Meteorology of the Free University of Berlin identified strong artifacts of the precipitation intensities appearing consistently near the border of the domain. Therefore, the uniform grid is cropped by 20 vertices at each side, resulting in 684 x 740 vertices representing a slightly smaller subdomain. The second dataset features a 15-minute resolution

and covers the period from March 25, 2007 at 6:30 to March 26, 2007 at 24:00 in a total of 167 time steps (dataset B). The spatial extent represents only a subset of the original domain which is mostly Germany discretized by a uniform grid with 291 x 321 vertices. This dataset was provided by the meteorologists of the Free University of Berlin.

5.1 Registration Results

The first step of the tracking approach presented in Section 4 is the registration of each pair of successive time steps of the aforenamed datasets. As mentioned before, the tracking results depend heavily on the outcomes of these registration processes. Therefore, a sufficient set of parameters must be found to yield deformation fields that represent an appropriate motion estimation. It should be noted that there is no universal way of finding a best set of parameters automatically for a given dataset. Instead, knowledge about the concrete field of application as well as experience with image registration and its components itself enable the identification of parameters that suit the underlying purpose which in the case at hand are adequate tracking results based on motion estimation.

5.1.1 Empirical Parameter Estimation

The parametrization presented in the following is determined empirically for the considered datasets. In fact, a single set of parameters that was found to be sufficient is identified for both hourly as well as 15-minute resolutions. The process of the empirical parameter estimation is based on a manual identification of sections in the datasets where the deformations react sensitively to a variation of the parameters in question. Among these sections, a subjective selection is made for which a ground truth is assumed to be identifiable without any additional domain knowledge. Subsequently, the tracking results are investigated manually for different parameter choices. The parameters presented hereafter are found to produce adequate tracking results in the majority of cases. However, they are not claimed to be the best-possible choice for precipitation data

in general. Instead, the parametrization should rather be considered a first suggestion for the concrete case of precipitation.

Transformation Model To account for the fact that precipitation as a physical quantity does not evolve globally but is obviously affected by local interactions, a B-spline transformation model is used. To allow for fine-grained local transformations, the resolution of the control point grid (at the finest level) is defined by a spacing of 4 pixels in both dimensions. The default setting of B-spline basis functions of degree 3 is adopted.

Distance Measure On the one hand, the extents as well as the shapes of precipitation cells vary greatly. On the other hand, a variation of the intensities within cells over time is a natural phenomenon. Therefore, the process of finding a spatial relation between two images should be flexible such that structures can be aligned even if they exhibit different intensities. To be independent of the assumption of a linear relationship between intensities as well, mutual information (MI) is chosen as a similarity measure.

Regularization Term In addition to the degree of smoothness that is inherent in the B-spline transformation, the transformation is constrained as it is generally recommended for local deformation models. More specifically, the bending energy regularizer is used to privilege deformations that do not exhibit strong compression accompanied by high expansion in close proximity.

Weights The weights w_0 and w_1 control the influence of the distance measure and the regularization term. This weighting is largely responsible to which extent local deformation will manifest itself in the final deformation. A desired pair of weights is expected to allow only for local deformations that can still be considered physically plausible. At the same time, the weights have a direct impact on which spels are adjacent in the temporal dimension of the spatio-temporal digital space utilized for the purpose of tracking. Despite the fact that a weighting with $w_0 = 0.8$ and $w_1 = 0.2$ is found to be appropriate for the datasets considered, the weighting parameter should generally get special

attention during a parameter estimation, even across datasets of the same quantity.

Optimization Strategy As an optimization strategy, an adaptive stochastic gradient descent optimizer is chosen. On the one hand, gradient descent is a strategy that is well-understood. To overcome the performance issues that are immanent in the computation of the gradient over a high-dimensional parameter space, elastix offers sophisticated solutions in form of a stochastic variant. For the number of random pixels drawn in each iteration, 1% of the overall number of pixels was found to be a compromise between computation time and accuracy of the final deformation. For dataset A, this results in 5100 samples while for dataset B the correspondingly smaller number is 1200 samples. Moreover, to prevent the optimal solution from being trapped in a local minimum of a single optimization run, the entire optimization is repeated 500 times and the final solution is the minimum of all 500 runs.

Hierarchical Scheme The parametrization so far describes the fundamental components of a registration method. In addition, elastix provides the possibility to iterate the whole process at different resolution levels while propagating the (intermediate) registration results from coarse levels up the the finest level. This hierarchical scheme is used for the datasets in form of four resolution levels regarding the data in each iteration as well as the transformation model. In particular, in each resolution level $i \in [0, 1, 2, 3]$, where level 0 represents the finest resolution and level 3 the coarsest resolution, the images are smoothed with a Gaussian blur with $\sigma = \frac{2^i}{2}$ and the resolution of the B-spline transformation's control point grid is defined by a spacing of 4×2^i in both dimensions.

5.1.2 Example Parameter File

In the following, the actual parameter file used for the registration process of each pair of successive images is presented.[34] If not set explicitly in the following, parameters are left at their defaults.

[34] Since the number of random pixels drawn in each iteration of the optimization process must be specified as an absolute number, the listed parameter `NumberOfSpatialSamples` refers specifically to dataset A.

```
//  GENERAL
(MovingImageDimension 2)
(FixedImageDimension 2)
(MovingInternalImagePixelType "float")
(FixedInternalImagePixelType "float")
(ResultImagePixelType "float")
(ResultImageFormat "mha")

//  HIERARCHICAL SCHEME
(Registration "MultiMetricMultiResolutionRegistration")
(NumberOfResolutions 4)
(MovingImagePyramid "MovingSmoothingImagePyramid")
(FixedImagePyramid "FixedSmoothingImagePyramid")

//  TRANSFORMATION MODEL
(Transform "RecursiveBSplineTransform")
(BSplineTransformSplineOrder 3)
(FinalGridSpacingInVoxels 4.0 4.0)

//  SIMILARITY MEASURE & REGULARIZATION TERM
(Metric "AdvancedMattesMutualInformation"
        "TransformBendingEnergyPenalty")
(Metric0Weight 0.8)
(Metric1Weight 0.2)

//  OPTIMIZER
(Optimizer "AdaptiveStochasticGradientDescent")
(MaximumNumberOfIterations 500)
(ASGDParameterEstimationMethod "Original")
(ImageSampler "Random")
(NumberOfSpatialSamples 5100)
(NewSamplesEveryIteration "true")
```

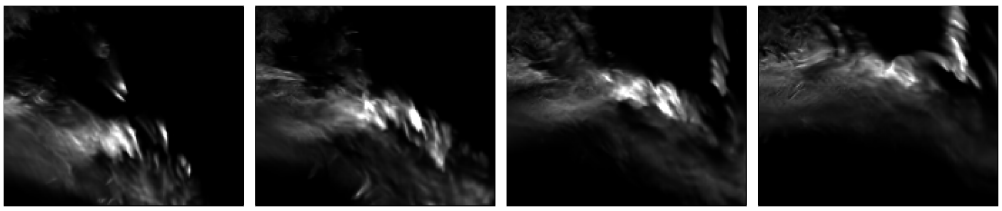
5.2 Tracking Results

Based on the parametrization of the registration method presented in the previous section, the tracking results are shown for an example section of dataset A. On the one hand, the example is used to convey a qualitative notion of hourly precipitation as well as the corresponding deformation fields. On the other hand, the purpose is to get a qualitative impression of how the tracking approach presented in Section 4 (method A) performs for a quantity such as precipitation. To be able to assess the possible gain enabled by the novel correspondence identification developed in this thesis, the results of method A are shown alongside the results of a second tracking solution, method B, that uses spatial overlap to identify correspondence.

The actual implementation of the developed tracking method, as well as its variation based on overlap, was implemented within the visualization software *Amira* [138]. Additionally, the entire tracking pipeline as well as the visualizations depicted hereinafter are based on the possibilities provided by Amira.

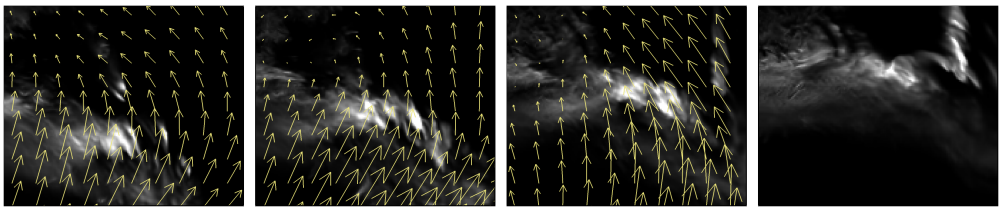
Tracking Example

An example section by means of which the tracking results are shown is chosen that comprises many aspects of the tracking approach. Hence, it is claimed to be representative on a qualitative basis for the datasets considered in this thesis. The example covers a small section in the south-east of the COSMO-REA2 domain on March 27, 2007 between 18:00 and 21:00 in the form of four time steps. While Figure 33 shows the precipitation intensities, Figures 34 and 35 additionally show the corresponding forward and backward deformation fields. It should be noted that the deformation fields are sampled such that the visualization is not cluttered by an extensive number of vector arrows. In all following figures where the intensities are visualized, a linear mapping from black to white is used such that the smallest intensity value of 0 is mapped to black and intensity values of 20 or higher are mapped to white. Values in between are interpolated linearly.



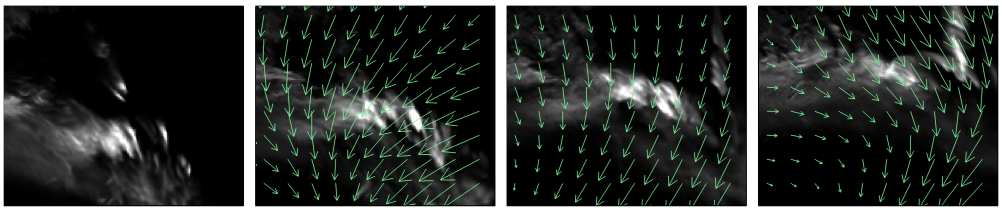
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 33: Precipitation data



(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 34: Forward deformation fields



(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 35: Backward deformation fields

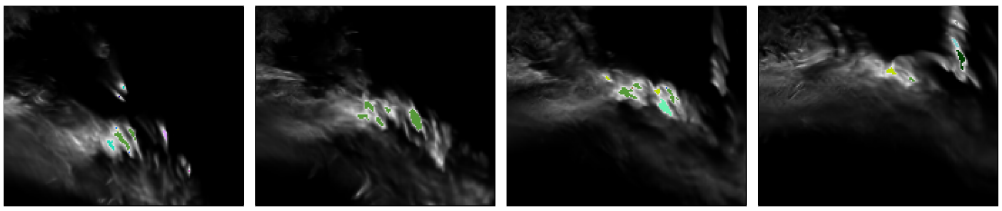
The depicted time steps in Figure 33 show small structures of high precipitation in the center as well as more extensive structures of average and low precipitation that either comprehend the central high-precipitation structures or evolve independently on the left. The dominant apparent motion seems to point towards the top. The corresponding deformation fields in Figures 34 and 35 visualize the spatial relations between successive time steps. As mentioned in Section 4.1.1, the registration processes resulting in the forward and backward deformations for a pair of time steps are independent of each other. This becomes

apparent by comparing the three forward deformation fields in Figure 34 each with the corresponding and opposed backward deformation field in Figure 35. The middle pair of deformations (Figures 34b and 35c) describe relations that are 'almost symmetrical'. The right pair of deformations (Figures 34c and 35d) exhibit strong differences in the lower left area while the relations in the other areas can still be described as 'almost symmetrical'. The left pair (Figures 34a and 35b) shows strong differences between both relations due to local manifestations describing an expansion to the right in the forward deformation and an expansion to the left in the backward deformation.

Segmentation & Tracking Graph

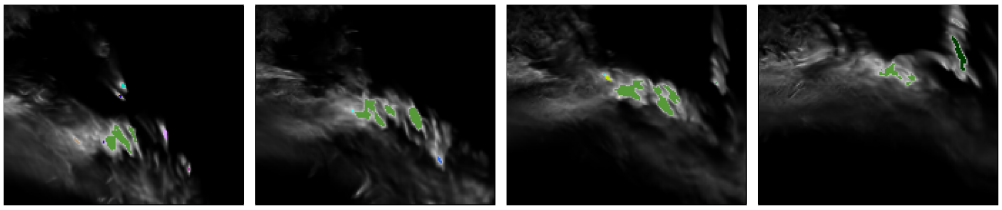
In the following, the results of both considered tracking methods are applied to the aforementioned example section in order to assess qualitatively how the registration-based tracking approach (method A) performs for a quantity such as precipitation compared to a tracking solution that builds upon overlap (method B). The spatio-temporal IWS of method A is carried out for three precipitation levels, $h \in [18, 15, 12]$, at the coarsest detail level, $p = max$, such that all adjacent regions that arise during the spatio-temporal IWS are merged. Effectively, this results in a CCL of the spatio-temporal digital picture for the three specified levels. In Figures 36–38, the resulting segmentation is shown separately in each time step together with the underlying precipitation data. In Figure 39, the same segmentation but for the entire three-dimensional spatio-temporal digital space is shown by means of an orthographic projection along the third dimension together with the corresponding tracking graph. The same visualizations are generated for method B in which spatial overlap is used to identify correspondences between features instead (see Figures 40–42 and Figure 43).

For the first level, $h = 18$, as shown in Figures 39a and 43a, there are two overlapping components of successive time steps such that method B produces a single feature track. Method A identifies many corresponding components even for large distances resulting in two feature tracks. At the same time, components that are close are not necessarily found to be corresponding. In



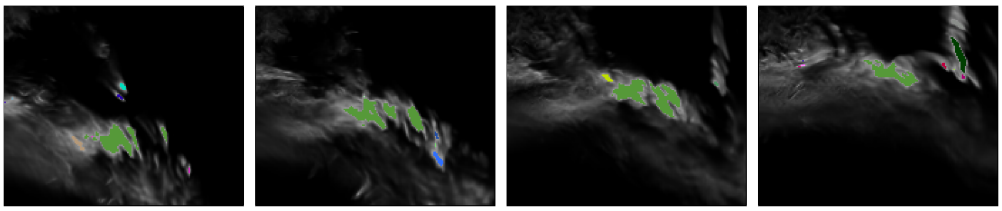
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 36: Method A; spatio-temporal IWS with $h = 18$ and $p = max$



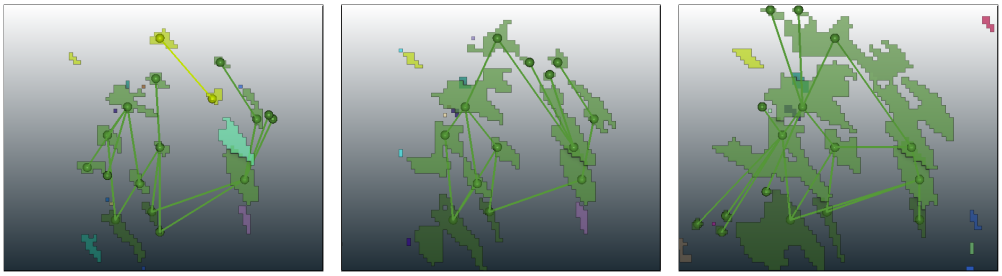
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 37: Method A; spatio-temporal IWS with $h = 15$ and $p = max$



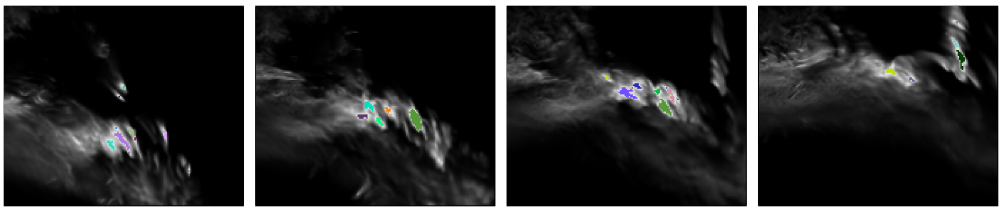
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 38: Method A; spatio-temporal IWS with $h = 12$ and $p = max$



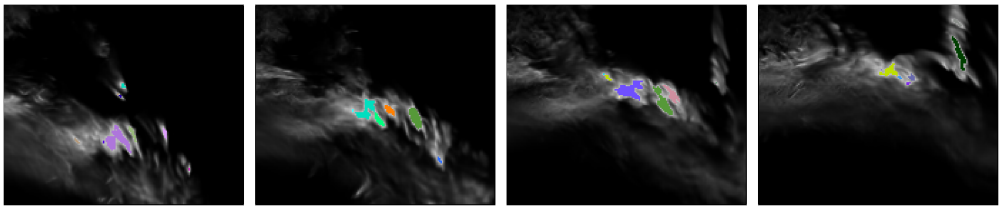
(a) $h = 18$, $p = max$ (b) $h = 15$, $p = max$ (c) $h = 12$, $p = max$

Figure 39: Method A; spatio-temporal IWS shown as an orthographic projection along the third dimension overlayed with the corresponding tracking graph.



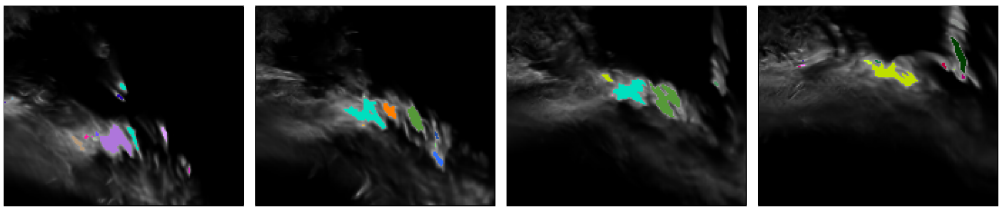
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 40: Method B; spatio-temporal IWS with $h = 18$ and $p = max$



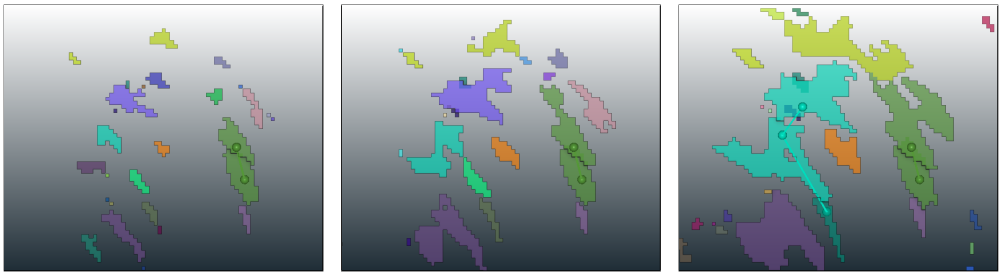
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 41: Method B; spatio-temporal IWS with $h = 15$ and $p = max$



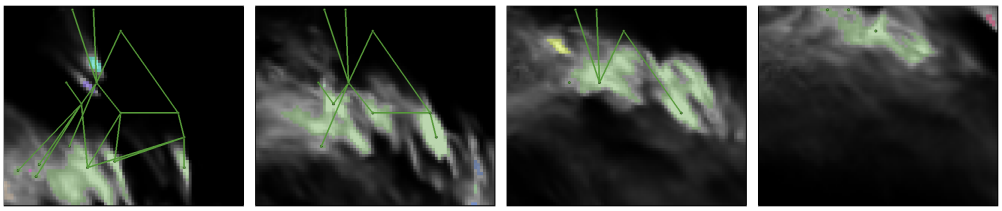
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 42: Method B; spatio-temporal IWS with $h = 12$ and $p = max$



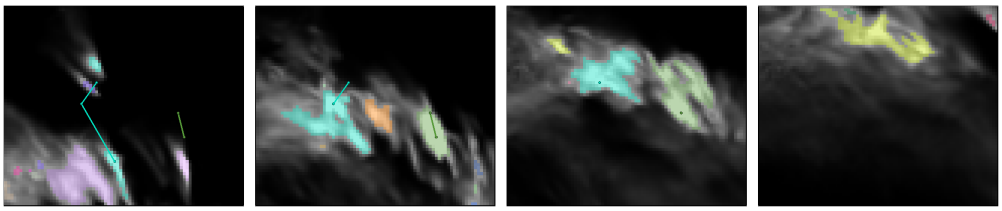
(a) $h = 18$, $p = max$ (b) $h = 15$, $p = max$ (c) $h = 12$, $p = max$

Figure 43: Method B; spatio-temporal IWS shown as an orthographic projection along the third dimension overlayed with the corresponding tracking graph.



(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 44: Method A; plausibility visualization for $h = 12$ and $p = max$



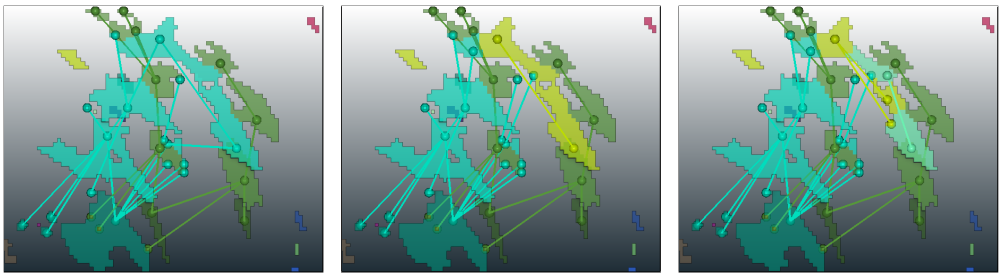
(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 45: Method A; plausibility visualization for $h = 12$ and $p = max$

particular, the two components that overlap do not correspond. Even for a high precipitation level, method A produces a complex tracking graph. The main reason for this is that a single pair of corresponding spels is sufficient to generate an edge between two components in the final tracking graph.

For the second level, $h = 15$, the components of the segmentation in Figures 39b and 43b obviously increase in size. However, this does not lead to a higher number of overlapping components such that the number of feature tracks found by method B does not change and only the tracked area increases slightly. In contrast, due to the already high number of correspondences identified by method A, the tracked area scales well. In addition, the now larger components give rise to new correspondences between spels such that new corresponding components are found. However, since some components that have been distinct for $h = 18$ have merged for $h = 15$, some arcs of the tracking graph collapse.

For the lowest level, $h = 12$, a second feature track is identified by method B as shown in Figure 43c. Although the same track components are also found to belong to a common feature track by method A in Figure 39c, the tracking



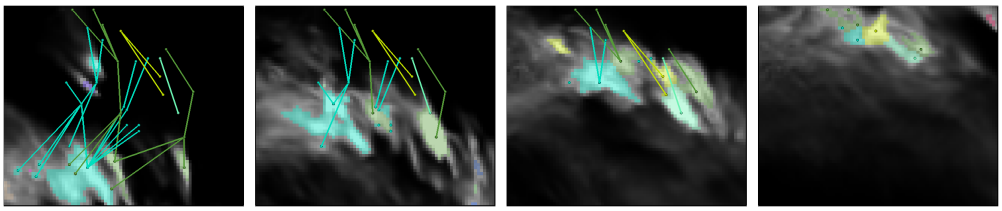
(a) $h = 12$, $p = 14$ (b) $h = 12$, $p = 13$ (c) $h = 12$, $p = 12$

Figure 46: Method A; hierarchical spatio-temporal IWS shown as an orthographic projection along the third dimension overlayed with the corresponding tracking graph.

graphs of method A and method B show that the particular correspondences between track components in Figure 43c are not present in Figure 39c. Overall, method A identifies almost all depicted components (except a few smaller ones) to belong to a common feature track. Therefore, almost the entire visible area is tracked over time. In contrast, method B identifies correspondences only among five components, and hence, tracks a significantly lower area over time. What becomes noticeable once more, is that many arcs are generated in the tracking graph of method A due to small track components. The fact that track components are assumed to be ω-connected, which resolves to a 4-connectedness in two dimensions, potentially supports the generation of additional nodes and arcs which can be seen in the top area of Figure 39c. In Figures 44 and 45, the previous visualizations are combined to yield a plausibility visualization for method A and method B for the precipitation level of $h = 12$.

Hierarchical Tracking

So far, the tracking results are shown at the coarsest detail level $p = max$. In the following, the spatio-temporal IWS of method A is carried out once more for the lowest precipitation level, $h = 12$, but for three detail levels, $p \in [14, 13, 12]$ as shown in Figure 46. In Section 3.6.1, the hierarchical watershed segmentation was introduced from the perspective of the finest detail level



(a) Time step 1 (b) Time step 2 (c) Time step 3 (d) Time step 4

Figure 47: Method A; plausibility visualization for $h = 12$ and $p = 12$

with the segmentations at coarser levels resulting from merges of regions from segmentations at finer levels. For real datasets such as the precipitation example, the segmentation of the finest detail level would imply an extensive number of feature tracks due to the vast number of local maxima. Therefore, the hierarchical tracking for $h = 12$ is carried out from coarse to fine detail levels.

For $p = 14$, the single feature track shown in Figure 39c splits in two as depicted in Figure 46a. The two new feature tracks represent subregions of the spatio-temporal IWS that correspond to the two most dominant maxima. Contrary to intuition, the two new connected components of the tracking graph are not subgraphs of the tracking graph just before the split in Figure 39c. The reason for this is that the generation of the tracking graph is based on subsequent CCLs with respect to ω-connectedness, i.e. 4-connectedness in two dimensions, applied to the pixels in each time step that share the same spatio-temporal label. As shown in Figure 46a, this fact is also prone to the generation of additional nodes and arcs due to potentially many small track components. The new feature track that emerged in Figure 46a splits once more into two feature tracks for $p = 13$ as shown in Figure 46b. The same procedure repeats for $p = 12$ as shown in Figure 46c. It should be noted that the hierarchical segmentation for $p = 12$ locally resembles the segmentation in Figure 39a for $h = 18$ and $p = max$ but with respect to the lower threshold. Figure 47 shows a plausibility visualization for method A for the precipitation level $h = 12$ and detail level $p = 12$. It can clearly be seen how the track components on the right hand side represent local maxima in each time step, while on the left hand side, the regions of local maxima already merged for the specified detail level.

6 Discussion

In the last chapter, the results of the tracking approach that has been developed in this thesis are presented for the case of precipitation data. The development is motivated by concrete requirements mentioned in the introduction. These requirements can be summarized as follows:

Requirement A The correspondence identification between features should be determined by all information of two successive time steps in form of dense motion estimations that are expected to be usable for the tracking of features at different levels in a universal way.

Requirement B Features should be able to be split into sub-features depending on dominant local extrema such that a tracking of sub-structures is supported.

Requirement C The representation of feature tracks is expected to enable the extraction of properties of features in each time step as well as over the life-time of each feature track. Moreover, the analysis of the spatio-temporal dynamic of feature tracks regarding join and split events should be supported.

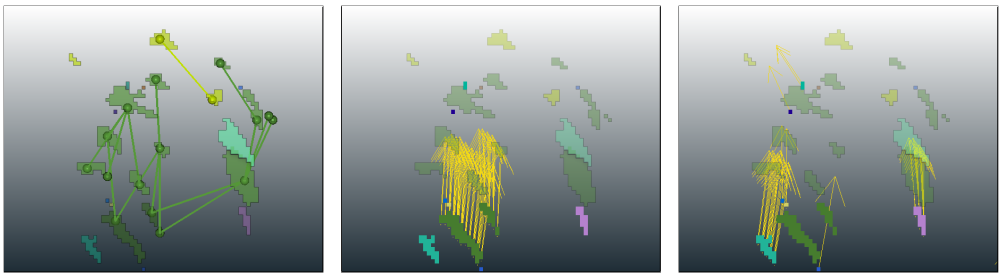
In the following discussion, the abovementioned requirements are adressed in order to indicate in which way and to which extent the developed tracking approach can be used for a subsequent systematical analysis of precipitation as well as other (meteorologic) quantities.

Requirement A

The registration-based motion estimation by means of elastix offers many possibilities regarding the parametrization of the registration process. The

presented set of parameters results in deformation fields that exhibit local manifestations and at the same time do not lead to implausibilities in the form of e.g. folds of the domain due to regularization. Hence, an estimation of motion that represents the temporal development of precipitation being affected by local interactions is basically enabled. It should be noted that the parameters that determine the accuracy of the local deformation (`FinalGridSpacingInVoxels`, `MaximumNumberOfIterations` and `NumberOfSpatialSamples`) are chosen such that a practical tradeoff between computation time and accuracy is achieved that allowed for a reasonable investigation of tracking results in the course of this thesis. The depicted deformation fields in Figures 34 and 35 indicate an estimated motion that can mostly be considered plausible. However, the opposed deformation fields in Figures 34a and 35b show strong differences. Intuitively, the two registration processes for each pair of time steps are expected to result in deformation fields that look almost symmetrical. However, the deformations represent spatial relations between time steps in their entireties. If the abovementioned forward and backward deformations are considered with respect to the intensities in the entire image, one can find arguments for both deformation fields to be appropriate. When the registration-based motion estimations are utilized within the tracking method, forward and backward deformations are incorporated in an unbiased way such that, even if there is a strong difference between a forward and a backward deformation, both are considered equally.

While the feature tracks in Figure 43 obviously depend on the chosen level due to the dependence on overlap, the feature tracks depicted in Figure 39 are first and foremost independent of the actual level which is exactly intented by the identification of correspondence by means of a motion estimation being independent of the extraction of features. The estimated motion spans distances that exceed the spatial extent of precipitation cells at high levels by a multitude. As a result, feature tracks are identified that simply cannot be found by means of overlap purely from a conceptional point of view. The plausibility visualization in Figure 44 indicates that the incorporation of registration-based motion estimations within a tracking approach produces adequate tracking results while



(a) (b) (c)

Figure 48: Detailed visualization of the forward deformation for the tracking results of method A for $h = 18$ and $p = max$. The same visualization as in Figure 39a is show in (a). The forward deformation field of the first time step is shown for track components in (b) and for components that are not tracked over time in (c).

the overlap-based tracking identifies the majority of precipitation cells to be isolated without belonging to any feature track.

Besides the direction of the estimated motion, that is decisive for which features in each time step can correspond at all, the length of the deformation is crucial when considering the tracking results at different levels. This fact is visualized in Figure 48, where the tracking results of method A for $h = 18$ and $p = max$ are shown once more in combination with the corresponding dense forward deformation restricted to the segmentation in the first time step. While Figure 48b shows the forward deformation for the identified track components, the forward deformation is shown for the components that have not been tracked in Figure 48c. It can be seen that the forward deformation of the pixels of the turquoise and pink components – as well as of the blue and yellow 1-pixel components – actually point in a direction such that they tend to contribute to a feature track. However, due to the combination of the size of components being too small and the forward deformation reaching too far, the abovementioned components are not part of a feature track for the shown level at $h = 18$. In lower levels, due to the components of the segmentation increasing in size, the components in question are in fact corresponding as

shown in Figure 39. This example shows how the dense motion estimation failed to be used universally for feature tracking at different levels. However, due to the accuracy of the B-spline transformation that is parametrized with respect to a reasonable computation time, the situation depicted above might be resolved by carrying out the registration with highest accuracy. Another possibility is to use less resolution levels of the hierarchical scheme to focus the registration process on structures at smaller scales. However, this would apply to the registration results overall which is prone to produce undesired results in other regions of the domain. Instead, subsequent studies regarding the tracking of precipitation cells are expected to yield more insights into the parametrization of the registration method for the concrete application of precipitation data, such that the aforementioned problem can be resolved.

Requirement B

Since the tracking approach builds upon a segmentation of the spatio-temporal digital space by means of a hierarchical watershed segmentation, thresholding the criterion that defines different detail levels gives rise to a hierarchy of feature tracks. The criterion that was employed for the developed tracking method was described in Section 3.6.1 as the 'depth of catchment basins'. An alternative formulation of the depth is the spanned distance between the minimal and maximal intensities belonging to a region. The results for the detail levels shown in Figure 46 seem plausible for the depicted example. However, it is difficult to comprehend from a visual inspection which intensities were decisive for the split of a feature track and for which parts of a feature track the next splits in the hierarchy can be expected. Although this is clearly defined from the definition of the criterion, it is desired to be able to better foresee the hierarchical progression when varying the detail level during a visual exploration of a dataset. Instead of relying on the intensity ranges of regions which are only defined by the minimal and the maximal intensity, additional criteria are feasible that might consider all intensity values of a region such as e.g. the intensity distribution or have recourse to domain-specific concepts that are well-known in the field of application a given dataset is associated with.

Requirement C

The tracking results are first and foremost based on a segmentation of the underlying digital space to identify regions that belong to common feature tracks by means of spatio-temporal labels. In a subsequent step, tracking-components for each spatio-temporal label are identified in each time step. Based on this additional labelling in each time step, the tracking graph is generated. Due to the combination of spatio-temporal labels and the labelling of track components, properties of precipitation cells in each time step as well as over the life-time of feature-tracks can be accessed. The corresponding tracking graph yields the spatio-temporal dynamic of feature tracks such that topologic events such as the birth, death, join and merge of track components can be identified based on the degree of tracking graph nodes.

A particular property of feature tracks is the spatial distance that was travelled over the course of the track. If a feature track neither splits nor joins over its life-time, its distance travelled could be estimated by the sum of projected edges that connect track components in successive time steps. However, as indicated by the multi-branched connected components of the tracking graphs in Figure 46, feature tracks in general exhibit a high dynamic regarding splitting and joining track components. Two strategies are feasible to estimate the length of such tracks. On the one hand, the spatial arrangement of feature tracks can be analysed to compute a length that minimizes a certain error with respect to the entire feature track. On the other hand, each connected components of the tracking graph can be transformed to be linear, and hence, allowing for the aforementioned estimation. The second option was considered in particular during the development of the tracking approach. However, a sufficient solution that generates linear connected components of a tracking graph that appropriately represents the original multi-branched components in general has not been found during this thesis. Although the developed solutions were satisfying for tracking graphs whose connected component are almost linear, the approaches failed clearly for complex graph components. Overall, the knowledge gained is that the strategy to transform the connected components of the tracking graph into linear components is not a viable option and that the

first option should be considered instead. Unfortunately, an alternative solution could not be realised in the course of this thesis.

6.1 Conclusion

In this thesis, a new approach to the tracking of regions in time-dependent scalar fields is presented. The development is motivated by the interest in analysing precipitation data of the COSMO-REA2 reanalysis system over time as well as in particular a previous study of the very same data for which an overlap-based tracking solution did not produce satisfying results. Based on a set of requirements and the methodological conditions that arise as a consequence thereof, the design of a tracking method is presented that can be applied to arbitrary scalar quantities on uniform grids of arbitrary dimensions. For the course of this thesis, the new tracking method is implemented for the case of 2D+time although it is conceptually applicable to nD+time. The tracking approach is based on the assumption of interpreting the discrete time steps of time-dependent grid data as digital pictures. Only this assumption allows for the computation of a dense motion estimation of the entire digital space by means of image registration between successive time steps. Based on the resulting correspondences between pixels, a spatio-temporal digital space is defined over which the time-dependent data can be labelled according to the concept of a watershed segmentation. The labelling for different thresholds and detail levels can be generated fast due to the implementation of a hierarchical watershed segmentation that employs the topological information captured by digital join and split trees, respectively. Due to the incorporation of the temporal dimension within the segmentation process, the resulting spatio-temporal labels give rise to a tracking over time. In combination with a subsequent identification of track components in each time step as well as the final generation of the tracking graph, a representation of feature tracks is enabled that gives access to tracking-related properties as well as their spatio-temporal dynamic.

Finally, the tracking method is applied to the concrete case of precipitation data by means of example datasets of the COSMO-REA2 reanalysis system. The qualitative study of tracking results showed promising results regarding

almost all requirements that have been identified beforehand. In fact, the tracking method developed in this thesis is already used by the Institute of Meteorology of the Free University of Berlin with great satisfaction. Despite subsequent and more detailed case studies regarding precipitation, the tracking method is also used to track regions that are defined by thresholding the integrated vertical updraft. This quantity shows quite different characteristics compared to precipitation but first preliminary studies indicate that an adapted parametrization of the registration method yields promising tracking results as well. What is appreaciated in general, is the fast generating of tracking results for different thresholds and detail levels.

6.2 Outlook

Due to the various components of the developed tracking method, which are image registration, the watershed segmentation, join and split trees as well as the final visualization of the tracking results by means of a spatial graph, there are many starting points for further improvements that will be considered in the future. First and foremost, the elastix framework is under active development and offers – besides new functionality – new possibilities to accelerate the overall registration process. In particular, various modifications of the adaptive stochastic gradient descent are available which are planned to be investigated with the aim to increase the accuracy of the final deformations without additional computation time. In general, the feedback from subsequent studies of the meteorologists are expected to yield further insights towards an optimal parametrization of the registration method for the concrete case of precipitation. As already mentioned in the discussion, additional criteria defining the detail level of the hierarchical watershed segmentation are desired. In particular, domain-specific criteria can be used to inject additional knowledge of the concrete field of application into the hierarchical segmentation process. Furthermore, it is planned to investigate to which degree the final tracking results can be improved by setting a threshold on the minimal size of regions during the watershed segmentation. The developed tracking method makes use of the topological information captured by the saddle nodes of the join and

split trees to decide if two adjacent regions should be merged or not. A further reaching idea is to use a visual representation of the join and split trees to decide interactively if two regions should merge or if a merge should be reverted. At long last, this thesis did not succeed to estimate the travelled distance of complex feature tracks. Therefore, the development of a consistent solution for both almost linear as well as highly complex feature tracks will be adressed directly.

Bibliography

[1] P. S. Alexandroff and H. Hopf, *Topologie, Erster Band*, ser. Grundlehren der mathematischen Wissenschaften. Springer, 1935, vol. 45.

[2] Y. Arnaud, M. Desbois, and J. Maizi, "Automatic tracking and characterization of african convective systems on meteosat pictures", *Journal of Applied Meteorology*, vol. 31, no. 5, pp. 443–453, 1992.

[3] E. Artzy, G. Frieder, and G. T. Herman, "The theory, design, implementation and evaluation of a three-dimensional surface detection algorithm", *Computer Graphics and Image Processing*, vol. 15, no. 1, pp. 1–24, 1981.

[4] L. Bach, C. Schraff, J. D. Keller, and A. Hense, "Towards a probabilistic regional reanalysis system for Europe: Evaluation of precipitation from experiments", *Tellus A: Dynamic Meteorology and Oceanography*, vol. 68, no. 1, 2016.

[5] C. L. Bajaj, V. Pascucci, and D. R. Schikore, "Seed sets and search structures for optimal isocontour extraction", Texas Institute of Computational and Applied Mathematics (TICAM), University of Texas at Austin, Tech. Rep. 99-35, 1999.

[6] M. Barthélemy, "Spatial networks", *Physics Reports*, vol. 499, pp. 1–101, 2011.

[7] M. de Berg and M. van Kreveld, "Trekking in the alps without freezing or getting tired", *Algorithmica*, vol. 18, pp. 306–323, 1997.

[8] S. Beucher and F. Meyer, "The morphological approach to segmentation: The watershed transformation", in *Mathematical Morphology Image Processing*, CRC Press, 1992, pp. 433–481.

[9] P. Bhaniramka, R. Wenger, and R. Crawfis, "Isosurfacing in higher dimensions", in *Proceedings of the 11th Conference on Visualization (VIS)*, IEEE Computer Society, 2000, pp. 267–273.

[10] ——, "Isosurface construction in any dimension using convex hulls", *Transactions on Visualization and Computer Graphics*, vol. 10, no. 2, pp. 130–141, 2004.

[11] J.-D. Boissonnat and M. Wintraecken, "The topological correctness of PL-approximations of isomanifolds", in *Proceedings of the 36th Symposium on Computational Geometry (SoCG)*, ser. Leibniz International Proceedings in Informatics (LIPICS), vol. 164, Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2020, 20:1–20:18.

[12] C. Bollmeyer, "A high-resolution regional reanalysis for europe and germany", Ph.D. dissertation, Faculty of Mathematical and Natural Sciences, Rheinische Friedrich-Wilhelms-Universität Bonn, 2015.

[13] C. Bollmeyer, J. D. Keller, C. Ohlwein, S. Wahl, S. Crewell, P. Friederichs, A. Hense, J. Keune, S. Kneifel, I. Pscheidt, S. Redl, and S. Steinke, "Towards a high-resolution regional reanalysis for the european cordex domain", *Quarterly Journal of the Royal Meteorological Society*, vol. 141, no. 686, pp. 1–15, 2015.

[14] B. Bollobás, *Modern Graph Theory*, ser. Graduate Texts in Mathematics. Springer, 1998, vol. 184.

[15] R. L. Boyell and H. Ruston, "Hybrid techniques for real-time radar simulation", in *Proceedings of the 2nd Fall Joint Computer Conference*, AFIPS – American Federation of Information Processing Societies, 1963, pp. 445–458.

[16] P.-T. Bremer, G. H. Weber, V. Pascucci, M. S. Day, and J. B. Bell, "Analyzing and tracking burning structures in lean premixed hydrogen

flames", *Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 248–260, 2010.

[17] P.-T. Bremer, G. H. Weber, J. Tierny, V. Pascucci, M. S. Day, and J. B. Bell, "Interactive exploration and analysis of large-scale simulations using topology-based data segmentation", *Transactions on Visualization and Computer Graphics*, vol. 17, no. 9, pp. 1307–1324, 2010.

[18] L. G. Brown, "A survey of image registration techniques", *ACM Computing Surveys*, vol. 24, no. 4, pp. 325–376, 1992.

[19] G. Burosch and J.-M. Laborde, "Characterization of grid graphs", *Discrete Mathematics*, vol. 87, no. 1, pp. 85–88, 1991.

[20] H. Carr, "Topological manipulation of isosurfaces", Ph.D. dissertation, Department of Computer Science, University of British Columbia, 2004.

[21] H. Carr and J. Snoeyink, "Path seeds and flexible isosurfaces – using topology for exploratory visualization", in *Proceedings of the 5th EG/IEEE Symposium on Visualization (VisSym)*, Eurographics, 2003, pp. 49–58.

[22] H. Carr, J. Snoeyink, and U. Axen, "Computing contour trees in all dimensions", *Computational Geometry*, vol. 24, no. 2, pp. 75–94, 2003.

[23] E. V. Chernyaev, "Marching Cubes 33: Construction of topologically correct isosurfaces", CERN, Tech. Rep. CN-95-17, 1995.

[24] Y.-J. Chiang, T. Lenz, X. Lu, and G. Rote, "Simple and optimal output-sensitive construction of contour trees using monotone paths", *Computational Geometry*, vol. 30, no. 2, pp. 165–195, 2005.

[25] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens, and G. Marchal, "Automated multi-modality image registration based on information theory", in *Information Processing in Medical Imaging*, vol. 3, Springer, 1995, pp. 263–274.

[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3^{rd} Edition. MIT Press, 2009.

[27] J. Cox, D. B. Karron, and N. Ferdous, "Topological zone organization of scalar volume data", *Journal of Mathematical Imaging and Vision*, vol. 18, no. 2, pp. 95–117, 2003.

[28] H. S. M. Coxeter, "Discrete groups generated by reflections", *Annals of Mathematics*, vol. 35, no. 3, pp. 588–621, 1934.

[29] ——, *Regular Polytopes*, 3^{rd} Edition. Dover, 1973.

[30] L. Custodio, T. Etiene, S. Pesco, and C. Silva, "Practical considerations on marching cubes 33 topological correctness", *Computers & Graphics*, vol. 37, no. 7, pp. 840–850, 2013.

[31] L. De Floriani, F. Iuricich, P. Magillo, and P. Simari, "Discrete morse versus watershed decompositions of tessellated manifolds", in *Proceedings of the* 17^{th} *International Conference on Image Analysis and Processing (ICIAR), Part II*, ser. Lecture Notes in Computer Science, vol. 8157, Springer, 2013, pp. 339–348.

[32] R. Diestel, *Graph Theory*, 5^{th} Edition, ser. Graduate Texts in Mathematics. Springer, 2017, vol. 173.

[33] M. Dixon and G. Wiener, "TITAN: Thunderstorm identification, tracking, analysis, and nowcasting—A radar-based methodology", *Journal of Atmospheric and Oceanic Technology*, vol. 10, no. 6, pp. 785–797, 1993.

[34] A. Doi and A. Koide, "An efficient method of triangulating equi-valued surfaces by using tetrahedral cells", *Transactions on Information and Systems*, vol. 74, no. 1, pp. 214–224, 1991.

[35] S. Dowdy, S. Wearden, and D. Chilko, *Statistics for Research*, 3^{rd} Edition, ser. Wiley Series in Probability and Statistics. Wiley, 1983.

[36] R. O. Duda, P. E. Hart, and J. H. Munson, "Graphical-data-processing research study and experimental investigation", Stanford Research Institute, Stanford University, Tech. Rep. ECOM-01901-26, 1967.

[37] H. Edelsbrunner and E. P. Mücke, "Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms", *Transactions on Graphics*, vol. 9, no. 1, pp. 66–104, 1990.

[38] B. Fischer and J. Modersitzki, "A unified approach to fast image registration and a new curvature based registration technique", *Linear Algebra and its Applications*, vol. 380, pp. 107–124, 2004.

[39] J. Fischer, "Von konvektiver zu stratiformer Struktur: Eigenschaften von Niederschlag in COSMO-REA2", M.S. thesis, Institute of Meteorology, Freie Universität Berlin, 2018.

[40] M. Fisher and R. Aldridge, "Hierarchical image segmentation using a watershed scale-space tree", in *Proceedings of the 7th International Conference on Image Processing and its Applications (IPA)*, IEE – Institution of Electrical Engineers, 1999.

[41] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*, ser. The Systems Programming Series. Addison-Wesley, 1990.

[42] H. Freudenthal, "Simplizialzerlegungen von beschränkter Flachheit", *Annals of Mathematics*, vol. 43, no. 3, pp. 580–582, 1942.

[43] D. Gordon and J. K. Udupa, "Fast surface tracking in three-dimensional binary images", *Computer Vision, Graphics, and Image Processing*, vol. 45, no. 2, pp. 196–214, 1989.

[44] A. A. Goshtasby, *Theory and Applications of Image Registration*. John Wiley & Sons, 2017.

[45] M. Grimaud, "New measure of contrast: The dynamics", in *Image Algebra and Morphological Image Processing III*, vol. 1769, SPIE – International Society for Optics and Photonics, 1992, pp. 292–305.

[46] J. L. Gross, J. Yellen, and M. Anderson, *Graph Theory and Its Applications*, 3^{rd} Edition, ser. Textbooks in Mathematics. CRC Press, 2018.

[47] S. J. F. Guimarães, J. Cousty, Y. Kenmochi, and L. Najman, "A hierarchical image segmentation algorithm based on an observation scale", in *Structural, Syntactic, and Statistical Pattern Recognition – Proceedings of the Joint IAPR International Workshops on Statistical Techniques in*

Pattern Recognition and Structural and Syntactic Pattern Recognition SSPR /SPR, ser. Lecture Notes in Computer Science, vol. 7626, Springer, 2012, pp. 116–125.

[48] J. V. Hajnal, D. L. G. Hill, and D. J. Hawkes, *Medical Image Registration*, ser. Biomedical Engineering. CRC Press, 2001.

[49] P. R. Halmos, *Naive Set Theory*, ser. University Series in Undergraduate Mathematics. Van Nostrand, 1960.

[50] F. Harary, *Graph Theory*. Addison-Wesley, 1969.

[51] G. T. Herman, "Discrete multidimensional jordan surfaces", *Graphical Models and Image Processing*, vol. 54, no. 6, pp. 507–515, 1992.

[52] ——, "Oriented surfaces in digital spaces", *CVGIP: Graphical Models and Image Processing*, vol. 55, no. 5, pp. 381–396, 1993.

[53] ——, "Jordan pairs of adjacency relations in digital spaces", *Annals of the New York Academy of Sciences*, vol. 728, no. 1, pp. 154–169, 1994.

[54] ——, "Boundaries in digital spaces: Basic theory", in *Topological Algorithms for Digital Image Processing*, ser. Machine Intelligence and Pattern Recognition, vol. 19, Elsevier, 1996, pp. 233–261.

[55] ——, *Geometry of Digital Spaces*, ser. Applied and Numerical Harmonic Analysis. Birkhäuser, 1998.

[56] B. K. P. Horn and B. G. Schunck, "Determining optical flow", *Artificial Intelligence*, vol. 17, no. 1–3, pp. 185–203, 1981.

[57] X. Huang, M. Fisher, and Y. Zhu, "From min tree to watershed lake tree: Theory and implementation", in *Proceedings of the 1st International Conference on Image Analysis and Recognition (ICIAR)*, ser. Lecture Notes in Computer Science (LNCS), vol. 3656, Springer, 2004, pp. 848–857.

[58] W. Imrich, S. Klavžar, and D. F. Rall, *Topics in Graph Theory: Graphs and Their Cartesian Product*. A K Peters, 2008.

[59] G. Ji and H.-W. Shen, "Efficient isosurface tracking using precomputed correspondence table", in *Proceedings of the 6th EG/IEEE Symposium on Visualization (VisSym)*, Eurographics, 2004, pp. 283–292.

[60] G. Ji, H.-W. Shen, and R. Wenger, "Volume tracking using higher dimensional isosurfacing", in *Proceedings of the 14th Conference on Visualization (VIS)*, IEEE Computer Society, 2003, pp. 209–216.

[61] J. T. Johnson, P. L. MacKeen, A. Witt, E. D. W. Mitchell, G. J. Stumpf, M. D. Eilts, and K. W. Thomas, "The storm cell identification and tracking algorithm: An enhanced WSR-88D algorithm", *Weather and Forecasting*, vol. 13, no. 2, pp. 263–276, 1998.

[62] A. P. Keszei, B. Berkels, and T. M. Deserno, "Survey of non-rigid registration tools in medicine", *Journal of Digital Imaging*, vol. 30, no. 1, pp. 102–116, 2017.

[63] E. D. Khalimsky, "The topologies of generalized segments", in *Soviet Mathematics Doklady*, vol. 10, American Mathematical Society, 1969, pp. 1508–1511.

[64] E. D. Khalimsky, R. Kopperman, and P. R. Meyer, "Computer graphics and connected topologies on finite ordered sets", *Topology and its Applications*, vol. 36, no. 1, pp. 1–17, 1990.

[65] S. Klein, J. P. W. Pluim, M. Staring, and M. A. Viergever, "Adaptive stochastic gradient descent optimisation for image registration", *International Journal of Computer Vision*, vol. 81, no. 3, pp. 227–239, 2009.

[66] S. Klein and M. Staring, *Elastix: The manual*, `https://github.com/SuperElastix/elastix/releases/download/5.0.0/elastix-5.0.0-manual.pdf`, Oct. 2019.

[67] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. W. Pluim, "Elastix: A toolbox for intensity-based medical image registration", *Transactions on Medical Imaging*, vol. 29, no. 1, pp. 196–205, 2010.

[68] S. Klein, M. Staring, and J. P. W. Pluim, "Evaluation of optimization methods for nonrigid medical image registration using mutual information and b-splines", *Transactions on Image Processing*, vol. 16, no. 12, pp. 2879–2890, 2007.

[69] R. Klette, "Cell complexes through time", in *Vision Geometry IX – Proceedings of the International Symposium on Optical Science and Technology*, vol. 4117, SPIE – International Society for Optics and Photonics, 2000, pp. 134–145.

[70] R. Klette and A. Rosenfeld, *Digital Geometry: Geometric Methods for Digital Picture Analysis*, ser. The Morgan Kaufmann Series in Computer Graphics. Elsevier, 2004.

[71] T. Y. Kong, R. Kopperman, and P. R. Meyer, "A topological approach to digital topology", *The American Mathematical Monthly*, vol. 98, no. 10, pp. 901–917, 1991.

[72] T. Y. Kong, A. W. Roscoe, and A. Rosenfeld, "Concepts of digital topology", *Topology and its Applications*, vol. 46, no. 3, pp. 219–262, 1992.

[73] T. Y. Kong and A. Rosenfeld, "Digital topology: Introduction and survey", *Computer Vision, Graphics, and Image Processing*, vol. 48, no. 3, pp. 357–393, 1989.

[74] ——, "Digital topology: A comparison of the graph-based and topological approaches", in *Topology and Category Theory in Computer Science*, Oxford University Press, 1991, pp. 273–289.

[75] T. Kosicki, "Graph representation of n-dimensional space", *Advances in Manufacturing*, vol. 2, pp. 54–60, 2014.

[76] V. A. Kovalevsky, "Finite topology as applied to image analysis", *Computer Vision, Graphics, and Image Processing*, vol. 46, no. 2, pp. 141–161, 1989.

[77] M. van Kreveld, R. van Oostrum, C. L. Bajaj, V. Pascucci, and D. R. Schikore, "Contour trees and small seed sets for isosurface

traversal", in *Proceedings of the 13th Symposium on Computational Geometry (SoCG)*, ACM, 1997, pp. 212–220.

[78] H. W. Kuhn, "Some combinatorial lemmas in topology", *Journal of Research and Development*, vol. 4, no. 5, pp. 518–524, 1960.

[79] I. S. Kweon and T. Kanade, "Extracting topographic terrain features from elevation maps", *CVGIP: Image Understanding*, vol. 59, no. 2, pp. 171–182, 1994.

[80] H. Kyznarová and P. Novák, "Celltrack — convective cell tracking algorithm and its use for deriving life cycle characteristics", *Atmospheric Research*, vol. 93, no. 1–3, pp. 317–327, 2009.

[81] V. Lakshmanan, K. Hondl, and R. Rabin, "An efficient, general-purpose technique for identifying storm cells in geospatial images", *Journal of Atmospheric and Oceanic Technology*, vol. 26, no. 3, pp. 523–537, 2009.

[82] D. E. Laney, P.-T. Bremer, A. Mascarenhas, P. L. Miller, and V. Pascucci, "Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities", *Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1053–1060, 2006.

[83] U. Lang and M. Grave, "Data structures in scientific visualization", in *Focus on Scientific Visualization*, ser. Computer Graphics: Systems and Applications, vol. 4, Springer, 1993, pp. 85–102.

[84] L. J. Latecki, *Discrete Representation of Spatial Objects in Computer Vision*, ser. Computational Imaging and Vision. Springer, 1998, vol. 11.

[85] H. K. Liu, "Two- and three-dimensional boundary detection", *Computer Graphics and Image Processing*, vol. 6, no. 2, pp. 123–134, 1977.

[86] A. Lopes and K. Brodlie, "Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing", *Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 16–29, 2003.

[87] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm", *Computer Graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[88] J. Lukasczyk, G. H. Weber, R. Maciejewski, C. Garth, and H. Leitte, "Nested tracking graphs", *Computer Graphics Forum*, vol. 36, no. 3, pp. 12–22, 2017.

[89] D. M. Mark, "Topological properties of geographic surfaces: Applications in computer cartography", in *Proceedings of the 1st Advanced Study Symposium on Topological Data Structures for Geographic Information Systems*, Harvard University, 1977.

[90] P. Markowski and Y. Richardson, *Mesoscale meteorology in midlatitudes*, ser. Advancing Weather and Climate Science. John Wiley & Sons, 2011, vol. 2.

[91] Y. Matsumoto, *An Introduction to Morse Theory*, ser. Translations of Mathematical Monographs. American Mathematical Society, 2002, vol. 208.

[92] C. Min, "Simplicial isosurfacing in arbitrary dimension and codimension", *Journal of Computational Physics*, vol. 190, no. 1, pp. 295–310, 2003.

[93] S. Mizuta and T. Matsuda, "Description of digital images by region-based contour trees", in *Proceedings of the 2nd International Conference on Image Analysis and Recognition (ICIAR)*, ser. Lecture Notes in Computer Science (LNCS), vol. 3211, Springer, 2005, pp. 549–558.

[94] J. Modersitzki, *Numerical Methods for Image Registration*. Oxford University Press, 2004.

[95] P. Monasse and F. Guichard, "Scale-space from a level lines tree", *Journal of Visual Communication and Image Representation*, vol. 11, no. 2, pp. 224–236, 2000.

[96] C. Montani, R. Scateni, and R. Scopigno, "A modified look-up table for implicit disambiguation of marching cubes", *The Visual Computer*, vol. 10, pp. 353–355, 1994.

[97] D. W. Moore, "Simplical mesh generation with applications", Ph.D. dissertation, Faculty of Graduate School, Cornell University, 1992.

[98] D. G. Morgenthaler and A. Rosenfeld, "Surfaces in three-dimensional digital images", Computer Vision Laboratory, University of Maryland, Tech. Rep. TR-940, 1980.

[99] S. P. Morse, "A topological approach to the problem of searching on a contour map", New York University, Tech. Rep. 400-129, 1966.

[100] C. Moseley, P. Berg, and J. O. Haerter, "Probing the precipitation life cycle by iterative rain cell tracking", *Journal of Geophysical Research: Atmospheres*, vol. 118, no. 24, 2013.

[101] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation", *Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 12, pp. 1163–1173, 1996.

[102] G. M. Nielson and B. Hamann, "The asymptotic decider: Resolving the ambiguity in marching cubes", in *Proceedings of the 2nd Conference on Visualization (VIS)*, IEEE Computer Society, 1991, pp. 83–91.

[103] P. Oesterling, C. Heine, G. H. Weber, D. Morozov, and G. Scheuermann, "Computing and visualizing time-varying merge trees for high-dimensional data", in *Topological Methods in Data Analysis and Visualization IV*, ser. Mathematics and Visualization, Springer, 2017, pp. 87–101.

[104] S. Ozeré, "Deformation field correction to preserve topology for image registration", *ESAIM: Proceedings and Surveys*, vol. 45, pp. 512–522, 2014.

[105] P. J. Pahl and R. Damrath, *Mathematical Foundations of Computational Engineering: A Handbook*. Springer, 2012.

[106] F. Pappenberger, H. L. Cloke, D. J. Parker, F. Wetterhall, D. S. Richardson, and J. Thielen, "The monetary benefit of early flood warnings in Europe", *Environmental Science & Policy*, vol. 51, pp. 278–291, 2015.

[107] V. Pascucci, "On the topology of the level sets of a scalar field", in *Proceedings of the 12th Canadian Conference on Computational Geometry (CCCG)*, 2001, pp. 141–144.

[108] V. Pascucci and K. Cole-McLaughlin, "Efficient computation of the topology of level sets", in *Proceedings of the 13th Conference on Visualization (VIS)*, IEEE Computer Society, 2002, pp. 187–194.

[109] T. Pavlidis, *Algorithms for Graphics and Image Processing*. Springer, 1982.

[110] N. Peleg and E. Morin, "Convective rain cells: Radar-derived spatiotemporal characteristics and synoptic patterns over the eastern mediterranean", *Journal of Geophysical Research: Atmospheres*, vol. 117, no. D15, 2012.

[111] J. L. Pfaltz, "Surface networks", *Geographical Analysis*, vol. 8, no. 1, pp. 77–93, 1976.

[112] F. P. Preparata and M. I. Shamos, *Computational Geometry – An Introduction*, ser. Monographs in Computer Science. Springer, 1985.

[113] Y. Qiao, B. van Lew, B. P. F. Lelieveldt, and M. Staring, "Fast automatic step size estimation for gradient descent optimization of image registration", *Transactions on Medical Imaging*, vol. 35, no. 2, pp. 391–403, 2015.

[114] G. Reeb, "Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique [On the singular points of a completely integrable Pfaff form or of a numerical function]", *Comptes rendus de l'Académie des sciences*, vol. 222, pp. 847–849, 1946.

[115] F. Reinders, F. H. Post, and H. J. Spoelder, "Visualization of time-dependent data using feature tracking", 1, vol. 17, Springer, 1999, pp. 55–71.

[116] T. Rigo, N. Pineda, and J. Bech, "Analysis of warm season thunderstorms using an object-oriented tracking method based on radar and total lightning data", *Natural Hazards and Earth System Sciences*, vol. 10, no. 9, pp. 1881–1893, 2010.

[117] T. Rohlfing, C. R. Maurer, D. A. Bluemke, and M. A. Jacobs, "Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint", *Transactions on Medical Imaging*, vol. 22, no. 6, pp. 730–741, 2003.

[118] A. Rosenfeld, "Adjacency in digital pictures", *Information and Control*, vol. 26, no. 1, pp. 24–33, 1974.

[119] ——, "Digital topology", *The American Mathematical Monthly*, vol. 86, no. 8, pp. 621–630, 1979.

[120] ——, "Three-dimensional digital topology", Computer Vision Labratory, University of Maryland, Tech. Rep. TR-936, 1980.

[121] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*. Academic Press, 1976.

[122] A. Rosenfeld, T. Y. Kong, and A. Y. Wu, "Digital surfaces", *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 4, pp. 305–312, 1991.

[123] A. Rosenfeld and J. L. Pfaltz, "Sequential operations in digital picture processing", *Journal of the ACM*, vol. 13, no. 4, pp. 471–494, 1966.

[124] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes, "Nonrigid registration using free-form deformations: Application to breast MR images", *Transactions on Medical Imaging*, vol. 18, no. 8, pp. 712–721, 1999.

[125] G. Sabidussi, "Graph multiplication", *Mathematische Zeitschrift*, vol. 72, no. 1, pp. 446–457, 1959.

[126] H. Saikia and T. Weinkauf, "Global feature tracking and similarity estimation in time-dependent scalar fields", *Computer Graphics Forum*, vol. 36, no. 3, pp. 1–11, 2017.

[127] P. Salembier, A. Oliveras, and L. Garrido, "Antiextensive connected operators for image and sequence processing", *Transactions on Image Processing*, vol. 7, no. 4, pp. 555–570, 1998.

[128] R. Samtaney, D. Silver, N. Zabusky, and J. Cao, "Visualizing features and tracking their evolution", *Computer*, vol. 27, no. 7, pp. 20–27, 1994.

[129] L. A. Schwarz, "Non-rigid registration using free-form deformations", Ph.D. dissertation, Department of Informatics, Technische Universität München, 2007.

[130] D. P. Shamonin, E. E. Bron, B. P. F. Lelieveldt, M. Smits, S. Klein, and M. Staring, "Fast parallel image registration on cpu and gpu for diagnostic classification of alzheimer's disease", *Frontiers in Neuroinformatics*, vol. 7, p. 50, 2014.

[131] C. E. Shannon, "A mathematical theory of communication", *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[132] D. Silver and X. Wang, "Tracking and visualizing turbulent 3D features", *Transactions on Visualization and Computer Graphics*, vol. 3, no. 2, pp. 129–141, 1997.

[133] J. K. Sircar and J. A. Cebrian, "An automated approach for labeling raster digitized contour maps", *Photogrammetric Engineering and Remote Sensing*, vol. 57, no. 7, pp. 965–971, 1991.

[134] P. Skraba and B. Wang, "Interpreting feature tracking through the lens of robustness", in *Topological Methods in Data Analysis and Visualization III*, ser. Mathematics and Visualization, Springer, 2014, pp. 19–37.

[135] B.-S. Sohn and C. L. Bajaj, "Time-varying contour topology", *Transactions on Visualization and Computer Graphics*, vol. 12, no. 1, pp. 14–25, 2005.

[136] M. Soler, M. Plainchault, B. Conche, and J. Tierny, "Lifted wasserstein matcher for fast and robust topology tracking", in *Proceedings of the 8th Symposium on Large Data Analysis and Visualization (LDAV)*, IEEE Computer Society, 2018, pp. 23–33.

[137] S. N. Srihari, "Representation of three-dimensional digital images", *Computing Surveys*, vol. 13, no. 4, pp. 399–424, 1981.

[138] D. Stalling, M. Westerhoff, and H.-C. Hege, "Amira: A highly interactive system for visual data analysis", in *The Visualization Handbook*, Elsevier, 2005, pp. 749–767.

[139] R. Szeliski and J. Coughlan, "Spline-based image registration", *International Journal of Computer Vision*, vol. 22, no. 3, pp. 199–218, 1997.

[140] S. Takahashi, T. Ikeda, Y. Shinagawa, T. L. Kunii, and M. Ueda, "Algorithms for extracting correct critical points and constructing topological graphs from discrete geographical elevation data", in *Computer Graphics Forum*, Eurographics, vol. 14, Wiley, 1995, pp. 181–192.

[141] S. P. Tarasov and M. N. Vyalyi, "Construction of contour trees in 3D in O(n log n) steps", in *Proceedings of the 14th Symposium on Computational Geometry (SoCG)*, ACM, 1998, pp. 68–75.

[142] R. E. Tarjan, "Efficiency of a good but not linear set union algorithm", *Journal of the ACM*, vol. 22, no. 2, pp. 215–225, 1975.

[143] N. J. Tustison, B. B. Avants, T. A. Sundaram, J. T. Duda, and J. C. Gee, "A generalization of free-form deformation image registration within the ITK finite element framework", in *Proceedings of the 3rd Workshop on Biomedical Image Registration (WBIR)*, ser. Lecture Notes in Computer Science (LNCS), vol. 4057, Springer, 2006, pp. 238–246.

[144] J. K. Udupa, "Multidimensional digital boundaries", *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 4, pp. 311–323, 1994.

[145] J. K. Udupa, S. N. Srihari, and G. T. Herman, "Boundary detection in multidimensions", *Transactions on Pattern Analysis and Machine Intelligence*, no. 1, pp. 41–50, 1982.

[146] A. A. Valsangkar, J. M. Monteiro, V. Narayanan, I. Hotz, and V. Natarajan, "An exploratory framework for cyclone identification and tracking", *Transactions on Visualization and Computer Graphics*, vol. 25, no. 3, pp. 1460–1473, 2018.

[147] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations", *Transactions on Pattern Analysis & Machine Intelligence*, no. 6, pp. 583–598, 1991.

[148] P. A. Viola, "Alignment by maximization of mutual information", Ph.D. dissertation, Massachusetts Institute of Technology, 1995.

[149] S. Wahl, C. Bollmeyer, S. Crewell, C. Figura, P. Friederichs, A. Hense, J. D. Keller, and C. Ohlwein, "A novel convective-scale regional reanalysis COSMO-REA2: Improving the representation of precipitation", *Meteorologische Zeitschrift*, vol. 26, no. 4, pp. 345–361, 2017.

[150] G. H. Weber, P.-T. Bremer, M. S. Day, J. B. Bell, and V. Pascucci, "Feature tracking using reeb graphs", in *Topological Methods in Data Analysis and Visualization*, ser. Mathematics and Visualization, Springer, 2011, pp. 241–253.

[151] G. H. Weber, G. Scheuermann, H. Hagen, and B. Hamann, "Exploring scalar fields using critical isovalues", in *Proceedings of the 13th Conference on Visualization (VIS)*, IEEE Computer Society, 2002, pp. 171–178.

[152] A. Weiser and S. E. Zarantonello, "A note on piecewise linear and multilinear table interpolation in many dimensions", *Mathematics of Computation*, vol. 50, no. 181, pp. 189–196, 1988.

[153] R. Wenger, *Isosurfaces: Geometry, Topology, and Algorithms*. CRC Press, 2013.

[154] T. Weusthoff and T. Hauf, "The life cycle of convective-shower cells under post-frontal conditions", *Quarterly Journal of the Royal Meteorological Society*, vol. 134, no. 633, pp. 841–857, 2008.

[155] W. Widanagamaachchi, C. Christensen, V. Pascucci, and P.-T. Bremer, "Interactive exploration of large-scale time-varying data using dynamic tracking graphs", in *Proceedings of the 2nd Symposium on Large Data Analysis and Visualization (LDAV)*, IEEE Computer Society, 2012, pp. 9–17.

[156] D. S. Wilks and R. L. Wilby, "The weather generation game: A review of stochastic weather models", *Progress in Physical Geography: Earth and Environment*, vol. 23, no. 3, pp. 329–357, 1999.

[157] C. Zhou, R. Shu, and M. S. Kankanhalli, "Handling small features in isosurface generation using marching cubes", *Computers & Graphics*, vol. 18, no. 6, pp. 845–848, 1994.

[158] Y. Zhou, W. Chen, and Z. Tang, "An elaborate ambiguity detection method for constructing isosurfaces within tetrahedral meshes", *Computers & Graphics*, vol. 19, no. 3, pp. 355–364, 1995.