Real-time anomaly and pattern recognition in high-resolution optical satellite imagery for future on-board image processing on Earth observation satellites

Dipl.-Phys. Sebastian Beulig

Vollständiger Abdruck der von der Fakultät für Luft- und Raumfahrttechnik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieur
(Dr.-Ing.)

genehmigten Dissertation.

Die Dissertation wurde am 14.05.2014 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 18.06.2014 angenommen. Die mündliche Prüfung fand am 15.12.2014 statt.

Abstract

Tomorrow's optical Earth observation satellites should be intelligent camera systems. They should evaluate the content of an acquired satellite image scene automatically and, in case of an emergency, autonomously change their acquisition mode. Then, additional shots of the affected area can be made. To do this, large volumes of imagery have to be analysed on-board a satellite in a short period of time in which, due to the limited resources on-board a spacecraft, the power consumption of the processing unit has to be minimal. Furthermore, all algorithms on-board have to be executed in a well-defined time span, i.e. they have to be capable of performing in real-time.

In this PhD thesis a programmable hardware chip, a so called Field Programmable Gate Array (FPGA) was used to perform the image processing. With this technology, the posed requirements can be realised. In the process, the image processing algorithms are specially customised to run on the chip in an optimal way.

The algorithms developed for anomaly and pattern recognition provide, especially in case of a flooding event, useful information about the affected region. In a captured scene, water areas, islands, bridges over water and traffic on a bridge can be recognised. High-resolution panchromatic *Proba-1/ HRC* satellite images, as well as *RapidEye* imagery from the near-infrared, the blue and the red spectral region were used as data sources. Additionally, *RapidEye* image data of the 2011 tsunami disaster in Japan were included in the analyses.

In the course of this doctoral thesis, methods were developed that are well suited for implementation on satellites due to their processing speed, real-time capability and relevance in case of catastrophes. Furthermore, they advance the spectrum of common on-board image analysis tools and demonstrate that high-level image processing is a feasible task even on a small FPGA. The obtained results are promising and provide a significant contribution to the aspired autonomy of future satellite missions.

Kurzfassung

Die optischen Erdbeobachtungssatelliten von morgen sollen intelligente Kamerasysteme werden. Sie sollen selbstständig den Inhalt einer aufgenommenen Satellitenbildszene erfassen und im Krisenfall ihren Aufnahmemodus eigenständig wechseln. So können zusätzliche Aufnahmen von dem betroffenen Gebiet gemacht werden. Dazu müssen an Bord eines Satelliten große Mengen an Bilddaten in kurzer Zeit analysiert werden, wobei aufgrund der begrenzten Ressourcen auf einem Raumfahrzeug der Stromverbrauch der Prozessierungseinheit so gering wie möglich sein soll. Zusätzlich sollen die Algorithmen an Bord in einer klar definierten Zeitspanne ablaufen, d.h. sie sollen echtzeitfähig sein.

In der vorliegenden Dissertation wurde ein programmierbarer Hardwarechip, ein sogenanntes Field Programmable Gate Array (FPGA), für die Bildprozessierung verwendet. Mit dieser Technologie können die oben genannten Anforderungen umgesetzt werden. Die Bildverarbeitungsalgorithmen werden dabei speziell angepasst, um optimal auf dem Chip zu funktionieren.

Die entwickelten Algorithmen zur Anomalie- und Mustererkennung liefern besonders im Fall einer Flutkatastrophe nützliche Informationen über die beschädigte Region. In einer aufgenommen Szene können Wasserflächen, Inseln, Brücken über Wasser und Straßenverkehr auf einer Brücke erkannt werden. Die Datengrundlage bilden hochaufgelöste panchromatische *Proba-1/ HRC* Satellitenbilder sowie *RapidEye* Aufnahmen aus dem nahen Infrarot, dem blauen und dem roten Spektrum. Es wurden auch *RapidEye* Bilddaten der Tsunami Flutkatastrophe von 2011 in Japan in die Analysen mit einbezogen.

Im Rahmen dieser Doktorarbeit wurden Methoden entwickelt die aufgrund ihrer Geschwindigkeit, Echtzeitfähigkeit und Relevanz im Katastrophenfall für einen Einsatz an Bord von Satelliten bestens geeignet sind. Darüber hinaus erweitern Sie das Spektrum von bisherigen Bildanalysen an Bord und zeigen, dass auch höhere Bildverarbeitung auf einem kleinen FPGA möglich ist. Die erzielten Ergebnisse sind dabei vielversprechend und leisten einen signifikanten Beitrag zur angestrebten Eigenständigkeit von künftigen Satellitenmissionen.

Contents

Chapter 1

Introduction

1.1 Background

It is more than five decades since the first Earth Observation (EO) satellites were inserted into orbit. The image acquisition chain on-board an EO satellite, however, follows even nowadays a rather strict concept. The acquisition date is scheduled on ground and uplinked to the satellite when it comes into view of one of its ground stations. The satellite executes the tasks according to the timeline. Later, the captured images are handled according to the ‘store-and-forward’ scheme (Yuhaniz and Vladimirova, 2009), i.e. the image data are stored in the on-board mass memory and downlinked to the ground stations afterwards. These steps are executed without regard to the actual image information content. Only when the imagery is analysed on ground can its true value be identified.
A smarter and more time-efficient approach would be to evaluate the captured satellite image data directly on-board. This would not substitute a detailed analysis on ground, but would provide some brief initial information. Corrupted or unusable data could be erased to free valuable memory space (Dawood et al., 2002b). A content related, selective image compression could be used to manage downlink capacities more efficiently (Camarero et al., 2010). And, in case of detected changes, the observation timeline could be rearranged to acquire additional images of the scene (Chien et al., 2003). The supplementary information would support the work of crisis information centres, as in the framework of the *International Charter Space and Major Disasters*. The charter is an association of 15 space agencies worldwide, to provide a fast and non-bureaucratic aid in case of natural disasters and serious technical breakdowns. In such an event, all available satellite image data are gathered and made available to the involved emergency services and governments (Mittelbach, 2013).
There are past and current satellite missions that demonstrate on-board image processing capabilities already. Some of the missions will be reviewed later. But, the main *information extraction* or *change detection* scenarios there covered variations in the occurrence of water areas, fires and clouds. This information is important and presents a basis. In this study, however, some more sophisticated patterns and anomalies were considered, to acquire more useful and desirable information from an image.

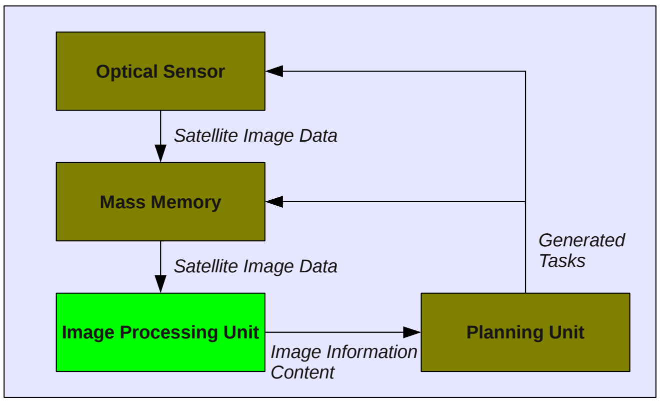


Figure 1.1: diagram of the described data and information flow.

According to an article about the work of the Crisis Information Center (ZKI) at the German Aerospace Center (DLR), reasonable information is:

> *'Which roads and bridges are still passable? How many houses or residential estates have been damaged? Which parts of a river bank need additional flood protection? '* (Danzeglocke, 2013)

So, in case of a flooding, an island detection method can be used to find rooftops where people are waiting for rescue. And a bridge over water recognition can provide information about potential transportation routes. Moreover, moving vehicles are an anomaly in an otherwise static image. Their detection on a recognised bridge gives further evidence on the accessibility of the bridge.
Almost 10 years ago, Zhou et al. (2004) wrote about future intelligent Earth observation satellites: *'...higher-level intelligent image processing like...change detection, image interpretation, pattern recognition...will need several generations of development. It has been demonstrated that full automation of image analysis and image interpretation is quite difficult...'*. So, in addition to unfolding new image processing capabilities for future satellite missions, the methods that will be presented in this thesis shall also demonstrate that high-level image analysis on-board a satellite has become a feasible task.

A basic structure and processing cycle of such an on-board system is visualised in figure 1.1. Optical satellite imagery is captured by the *Sensor* and stored in the *Mass Memory*. A copy of the data is routed through to the *Image Processing Unit*, where the information content is estimated. The box is highlighted in light green, because the present work focuses on this specific part. The extracted information, e.g. in form of a thematic map, is then passed forward to the *Planning Unit*. With regard to the status of the satellite, e.g. the available power and memory space, the *Planning Unit* could create a new timeline. The newly generated tasks, such as the arrangement of a second data-take or tagging an image with a higher downlink priority, could be realised immediately.

In order to afford an instant second data-take, e.g. with a backward-pointing camera (see the *ALOS-3* mission (Suzuki et al., 2012)), the image data have to be evaluated in a specific (small) time span, i.e. the process has to be *real-time*. A proper definition, and the challenges that arise with this requirement, will be discussed in the next section. On-board a satellite, it means large volumes of data have to be processed in a very short time, while the power consumption of the processing hardware shall be minimal. To tackle this discrepancy, the present study follows the approach of using *Field Programmable Gate Arrays* (FPGAs) as processing hardware. Thus, the present work is also a demonstration of the capabilities of FPGAs with regard to future space applications.

Altogether, the desired satellite's autonomy with regard to automatic image processing, a flexible and situational image acquisition and thus optimised on-board data handling, supports mission operations as well as crisis information centres, and greatly raises the benefits of the satellite mission.

1.2 Real-time and On-board Requirements

In the previous section, a proposition was made to evaluate captured satellite image data directly after the acquisition, thus enabling an immediate second data-take. For this purpose, image data processing has to be executed in *real-time*.
In common language, the term real-time processing is often used in the sense of a very fast performance but, actually, the real-time requirement for a computer program is something different. According to the definition and examples of Scholz (2005):

> *'A real-time system is a system that has to establish a correct reaction in a well defined period of time.'*

So, besides the correctness of the outcome, also the point in time when the result is available is of considerable interest. A true result too late is also a failure. Regarding the consequences of a failure, two different classes of real-time systems are distinguished, **hard real-time** systems and **soft real-time** systems:

- For a **hard real-time** system, an exact run-time can be determined, i.e. it is known beforehand at which time the process will terminate and a result is available. Many security related applications, e.g. the flight control system of an aeroplane, are liable to a hard real-time criteria. A violation of this requirement has a serious impact on the whole system.

- In a **soft real-time** system, a program has to be finished in an averaged period of time. A violation of certain deadlines is tolerable and does not have grave effects. An example is a certain frame-rate of a video multi-media system that cannot be held. The outcome would be a flickering image, that is a tolerable flaw.

For the aforementioned purpose of an image evaluation between two data takes, a hard real-time criteria would be appropriate. It has to be known in advance whether such an analysis in that time span is practical or not. However, the run-time of

programs performing high-level image analysis tasks often depends on the actual image content, so a hard real-time criteria cannot be fulfilled unless a worst case is assumed that can be met every time. At times, a worst case scenario cannot be given or processing times are far from being realistic. On the other hand, if the deadline is not met, it would mean missing a second data-take. An omission of a second image acquisition would not be a crucial factor for the whole satellite mission, thus it is tolerable. So the real-time requirement can be relaxed to a soft real-time requirement, which means that the image processing has to be done within a certain time-span.
In order to execute a routine in real-time, at least two approaches are practical. Firstly, a real-time operating system can be used. It can be run, e.g. on a microprocessor, and ensures the completion of programs in real-time. However, in low complex systems, they are avoided due to their necessary additional hardware and no gain in efficiency (Scholz, 2005).
The second approach is a direct implementation of algorithms into hardware. In this case, no operating system is necessary (Ley et al., 2011), it is a dedicated piece of hardware for a special purpose. There are at least two potential hardware devices: *Application Specific Integrated Circuits* (ASICs) and *Field Programmable Gate Arrays* (FPGAs).
ASICs have a lower power consumption, can be smaller and have a higher radiation tolerance than FPGAs, which are important factors for a space application, but they can be configured only once. This is a shortcoming for this study, since the development and testing of algorithms requires an adjustable system, thus an ASIC is not appropriate here. However, the final hardware realisation might be implemented as an ASIC as well.
In contrast to ASICs, FPGAs can be reconfigured many times directly by a user. So, with regard to an extensive testing and restructuring of algorithms during the development phase, they seem to be the better choice. FPGA power consumption depends on the application and has to be estimated (Hauck and DeHon, 2007). All in all, in this thesis FPGAs are employed as real-time capable processing hardware. To strengthen this course even more, two relevant satellite missions will be presented below.

1.3 Review of Relevant Satellite Missions

In the last decade, there have been several missions having the scientific goal to advance autonomous image processing on-board satellites. The *EO-1* mission, operated by the National Aeronautics and Space Administration (NASA), was launched in November 2000 and performed a broad number of on-board science analyses (Chien et al., 2003). Among other things, they conducted an automated flood detection (Ip et al., 2006). The Australian *FedSat* mission launched in December 2002 with the task to perform on-board image processing. As a start, they implemented two standard filter operations on an FPGA (Dawood et al., 2002a). A third example is the autonomous classification of acquired satellite image data on-board the German *BIRD* satellite (Halle et al., 2002).

In the following text, a more recent satellite mission will be presented in detail, since it has some relevant aspects for the current study carried out in this thesis. Afterwards, an overview will be given on a proposed future satellite mission, the *ALOS-3* mission.

1.3.1 The *X-Sat* Mission

Launched in 2011, X-Sat is a micro-satellite mission operated by the Centre for Research in Satellite Technologies (CREST), Singapore. The main payload of the satellite is a multi-spectral camera with spectral bands in the visible and near-infrared region. It captures images of the Earth at a spatial resolution of $10m$.
Besides Earth observation, another mission goal is the analysis of captured image data directly on-board. Therefore, a so-called Parallel Processing Unit (PPU) was established. The core of this processing unit consists of two FPGAs and 20 StrongARM microprocessors (PN_x). A block-diagram is shown in figure 1.2. During operations, the incoming workload is subdivided to the FPGAs and to the micro-processors. The FPGAs have to *'...provide a flexible network topology among the PNs...'* and take over *'...additional processing capabilities for real-time tasks'*. The coupled micro-processors perform *'...an unsupervised ground cover classification to extract thematic maps...[and an] autonomous detection of haze due to fires'*. To achieve this, a parallel alignment of micro-processors is chosen due to the *'...similarity to contemporary multi-processor and cluster systems for ground-based computation of remotely sensed data'*. The diagram, as well as the citations, are taken from the report of Bretschneider et al. (2004).
Due to the desired similarity to ground-based processing architectures, the whole set-up consists of at least 22 hardware devices and has a power consumption of $15-22W$ (X-Sat, 2013). In space, however, a smaller set-up would be favourable. For this reason, the study presented in this thesis will also show that high-level image analysis tasks can be implemented on an FPGA, so that a future image processing unit might consist of a single FPGA, only.

1.3.2 The *ALOS-3* Mission

The Advanced Land Observing Satellite-3 (ALOS-3) is an optical satellite mission initiated by the Japan Aerospace Exploration Agency (JAXA). It is a successor of the ALOS mission that ended in 2011, after a power generation anomaly. One of the main mission goals of ALOS-3 is disaster monitoring of certain areas.
The satellite carries the Panchromatic Remote-sensing Instrument for Stereo Mapping-2 (PRISM-2). This sensor comprises two telescopes with a Ground Sample Distance (GSD) of $0.8m$ at nadir and $1.25m$ backward looking (at nadir footprint) in order to acquire detailed images for a precise damage assessment. In addition, due to the stereoscopic image acquisition capability, Digital Surface Models (DSMs) can be developed. See figure 1.3.
Up to now, it is neither known whether an on-board image evaluation is intended for the mission, nor which is the potential image processing hardware in question.

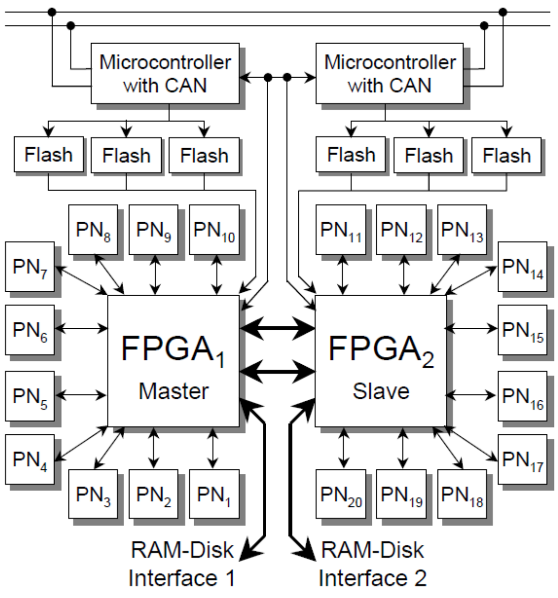

Microcontroller with CAN
Microcontroller with CAN
Flash
Flash
Flash
Flash
Flash
Flash
PN_8 PN_9 PN_{10} PN_{11} PN_{12} PN_{13}
PN_7 PN_6 PN_5 PN_4
PN_{14} PN_{15} PN_{16} PN_{17}
$FPGA_1$ Master
$FPGA_2$ Slave
PN_3 PN_2 PN_1 PN_{20} PN_{19} PN_{18}
RAM-Disk Interface 1
RAM-Disk Interface 2

Figure 1.2: block-diagram of the Parallel Processing Unit on-board the X-Sat satellite.

Nevertheless, the proposed sensor system suggests two possible scenarios which could benefit from an on-board image evaluation:

1. The nadir-pointing camera acquires an image that is immediately evaluated by the on-board image processing unit. If severe changes are discovered, the planning unit could arrange a second data-take with the backward-looking camera. In this case the timing constraints are very important.

2. A slightly less exacting scenario could be that at every data-take an image is collected from each camera. Afterwards, both images are analysed. If no acute changes occurred, at least one image can be erased from the mass memory to gain free memory space and save downlink capacities.

These ideas shall underline the practical aspects of an on-board image evaluation. Similar concepts for small satellites in general were also proposed by McLoughlin and Bretschneider (2010). So far, the launch of ALOS-3 is scheduled for 2015.

1.4 Scope and Scientific Results of this Thesis

This thesis demonstrates and evaluates new image processing capabilities, algorithms, as well as hardware implementations, dedicated for an application on future satellite missions. Reasonable pattern and anomaly detection tasks were selected that enhance current on-board image evaluation abilities and yield additional useful information. Several novel algorithms, partly based on already existing methods, were developed to detect water areas, islands, bridges over water, and moving vehicles in satellite

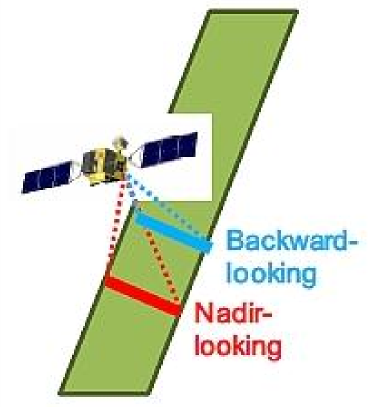


Figure 1.3: view of the image collection concept of PRISM-2 (image credit: JAXA). The image shows the stereo observation concept for a two-line push-broom strip-map imaging. The information and the image are taken from ALOS-3 (2013).

imagery. Most of these algorithms refer to high-level image analysis tasks and a hardware implementation is challenging. So, besides the novelty of the algorithms, also their implementation in a small FPGA presents a new engineering achievement.

The scientific results of this thesis are summarised as follows.

1. A simplified method to derive a water mask from panchromatic or near-infrared satellite image data. The algorithm avoids sophisticated calculations, thus it is well suited for hardware implementation. The obtained results are in good accordance with a ground truth mask and data from literature.

2. An efficient method to recognise bridges over water. The outcome is a map with located bridge center points. This approach needs little computational resources and is very fast. Overall, the accuracy is comparable to literature data.

3. A method to detect islands in a binary water mask. In principle, it follows a standard technique called *hole/ blob detection*, with some optimisations. The results improve the second bridge recognition and allow a more detailed analysis of the site.

4. A second bridge recognition method. The procedure is completely different to the first method and the outcome is a labelling of the whole bridge deck area. With this, additional analyses of the site are possible.

5. A small time-lag in the data acquisition procedure of *RapidEye* (RE) satellite image data permits the detection of moving objects. After a literature review, it seems that in this study the effect is used for the first time on RE imagery in order to get velocity information. In combination with the results of the second bridge recognition method, moving vehicles on a bridge can be detected. Hence,

additional information on the accessibility of a bridge can be provided. For this, a new algorithm was developed.

6. Each of the proposed methods fits into a small FPGA. That means, the algorithms are optimised program code to meet the constraints of computational time and resource consumption.

The methods and results of the *Water Body Extraction* and the *First Bridge Recognition Method* were presented at scientific congresses and are published in Beulig et al. (2012a) and Beulig et al. (2012b). In addition, a publication in a peer-reviewed journal is in progress. The *Second Bridge Recognition Method* was presented at an international conference and is published in Beulig et al. (2013).

1.5 Structure of this Thesis

The thesis is organised into four chapters. In the present introductory chapter, an overview and some background aspects about the thesis are given. The remaining three chapters can be summarised as follows.

Chapter 2: Review of Satellite Imagery and Hardware Set-up

This chapter represents the traditional 'Material and Methods' part of the thesis. In the beginning, the applied satellite imagery is introduced with an overview about the sensor systems and the presentation of an arranged data set that will be used in the experiments later on. Then, the processing hardware, the FPGA, is explained in more detail, so that the reader will be familiar with certain benefits and challenges that will influence the algorithm design. Finally, the current experimental set-up is displayed and explained.

Chapter 3: New Algorithms for Image Analysis On-board Satellites

This chapter is the main part of the thesis. The key findings, five different algorithms for an image analysis on-board satellites, are described and discussed in full detail. The chapter starts with an overview of the algorithms, then each method is presented in a separate section. The sections start with an introduction and a literature review, followed by description of the algorithm. Then, the obtained results are presented. In the end, each section concludes with a discussion and a summary.

Chapter 4: Findings and Prospects

In the final chapter of this thesis, the conducted work is summarised and the chapter concludes with an outlook towards future applications and potential areas of research.

Appendix A: Satellite Imagery Details

Additional information, such as the locations, coordinates and dates of acquisition of the applied satellite imagery, are listed in the appendix. In this study, a total of 60 scenes were analysed.

Chapter 2

Review of Satellite Imagery and Hardware Set-up

2.1 Outline

The spatial resolution of satellite imagery has a significant impact on the potential patterns or anomalies that can be recognised in an image. In principle, the higher the spatial resolution, the higher the probability that smaller objects can be found and larger objects appear more precise. However, optical sensor systems with a very high spatial resolution of $1m$ or less pose a demanding requirement, especially for very small satellites, since the sensor set-up is larger and more complex. Many man-made objects, like bridges over water, are often clearly visible at a spatial resolution of $5m$, already. So, in order to relax the specifications for an optical system, this resolution was chosen and the respective satellite imagery was employed in the current study.
In chapter 1, the ALOS-3 mission was presented as a potential optical system that could benefit from an on-board image evaluation. The stereo-imaging system comprises two panchromatic (PAN) cameras. With regard to this example system, PAN imagery is primarily used in this study. In addition, near-infrared (NIR) imagery is applied as well, since the reflectance values of water are very low in this spectral range (Liu et al., 2010), so that an enhanced separation between water and non-water areas is expected. To exploit the aforementioned time-lag in RapidEye imagery, red band (RED) and blue band (BLUE) RE image data are analysed in this study too. The colors have no actual meaning. The reason for choosing these bands is that the time gap is at a maximum between those two bands. The satellite/ sensor systems that provide these data will be presented in the next section, followed by a presentation of an arranged set of images that will be used in the experiments.
The favoured processing hardware was already introduced in chapter 1. In section 2.3, FPGAs are presented in more detail. Basic facts are given, a programming language is introduced, and some strategies for an optimised FPGA implementation are considered.
The chapter concludes with a description of the development process. The necessity of a software pre-development is discussed, an FPGA development board is presented and the work flow is illustrated, from algorithm coding up to resulting images.

2.2 Satellite Image Data

2.2.1 ***Proba-1/ HRC*** Data

The PRoject for On-Board Autonomy-1 (PROBA-1) is a mini-satellite of the European Space Agency (ESA), launched in 2001. The main purpose is *'...demonstrating the opportunities and benefits of on-board autonomy.'* (Bermyn, 2000). So, the overall ambition is similar to the current work, however, PROBA-1 does not include any high-level image evaluation. One optical sensor payload on-board PROBA-1 is the High Resolution Camera (HRC). It captures panchromatic images of the Earth's surface with a spatial resolution of $5m$. Each image has a dimension of 1026×1026 pixels, where each pixel is stored as an 8 bit grey scale value, i.e. with a value between 0 and 255 (PROBA-1, 2013). The main data characteristics are listed in table 2.1.
These satellite image data are the basis for the conducted experiments on PAN imagery. After an application for HRC images at ESA, free access to a database was granted. A set of images depicting bridges over water was selected. At this point, the author would like to thank ESA for providing the data.

2.2.2 ***RapidEye*** Data

RapidEye is a commercial Earth observation system comprising five identical small satellites. The satellite constellation was launched in 2008 and has an aspired live time of at least seven years. Each of the satellites is equipped with a 5-band multispectral imager. The spectral bands are RED, GREEN, BLUE, Red Edge and NIR. In this study, the NIR, RED and BLUE data are used. The data product is Level 3A, i.e. several pre-processing operations, such as sensor corrections, etc., were applied to the data in advance. For a detailed description, the reader is referred to RapidEye (2012). The RE imagery was delivered as large images, so called *tiles*, with a dimension of 5000×5000 pixels, and with a bit depth of 16 bit. The main data characteristics are listed in table 2.1.
In order to obtain RE data, an application was sent to the RapidEye Science Archive (RESA). According to the standard contract for educational purposes, 11 scenes were granted. The author thanks the German Aerospace Center (DLR), RESA and RapidEye for providing the data.

	HRC	RE
Spectral bands	PAN	NIR, RED, BLUE
Spatial resolution	$5m$	$5m$
Image size (pixel)	1026×1026	5000×5000
Bit depth	8 bit	16 bit

Table 2.1: basic specifications of the applied satellite image data.

2.2.3 Data Set

The list of data specifications presented in table 2.1 reveals significant differences between HRC and RE imagery with regard to image size and bit depth. In the experiments, however, a consistent data processing flow is desired. So, a uniform image size was chosen and the bit depth of all images is (reduced to) 8 bits. To gain some benefits in the later hardware implementation, the size of an image template was set to 1024×1024 pixels. But, in order to avoid errors while scanning the neighbourhood of pixels at the image boundary and to visualise the effective regarded image region, the outer pixels were coloured (in a neutral) white. As a result, the absolute satellite image size is 900×900 pixels. These preparations were carried out with the image manipulation software *GIMP* version 2.6.11 (GIMP, 2013).
In the beginning, a few satellite images were selected for the sole purpose of algorithm development. Later, a data set consisting of 20 HRC and 20 RE scenes was arranged to carry out a statistical assessment of the pattern recognition algorithms. An additional 20 RE scenes, that in eight cases depict the same locations as the first RE set, were selected to test the anomaly detection algorithm. All in all, the selections show a wide variety of different imaging conditions and scenarios. There are images showing rivers and lakes with bridges of all different sizes. The atmospheric, illumination and water conditions vary significantly. And the surroundings were rural and urban areas. Even images from a tsunami disaster were included in the study. The varied mixture is to test and reflect the robustness of the methods.
All data take locations are depicted in figure 2.1. Some regions are covered by HRC (green triangles) or RE (red circles) imagery solely, other regions are captured by both satellite systems (blue squares). A complete list, containing the coordinates and the acquisition dates of all scenes, is given in the appendix.

2.3 Field Programmable Gate Arrays (FPGAs)

2.3.1 Introduction

The first FPGAs were manufactured in the 1980s. Their introduction was somehow revolutionary since they combine the high-performance of ASICs, e.g. in matters of power consumption, with the flexibility of microprocessors (Hauck and DeHon, 2007). In recent years, an increasing effort was put into porting software image analysis algorithms on to FPGAs. The two main reasons are to enable a real-time system or to reduce the size and power consumption of the processing unit (Bailey and Johnston, 2010). Both arguments also hold for the pursued real-time image evaluation on-board satellites. Thus, the choice for an FPGA seems reasonable.
But hardware programming is different from software programming, since the algorithm design is strongly influenced by the employed chip and the connected peripheral devices (Hauck and DeHon, 2007). So, in the beginning of this section, the structure of an FPGA and some general aspects will be given. Then, a 'C-like' hardware description language, called *Handle-C*, is presented. This language allows a rapid development of hardware configurations without expertise in the underlying hardware (Alston and Madahar, 2002), a key factor in developing hardware programming in

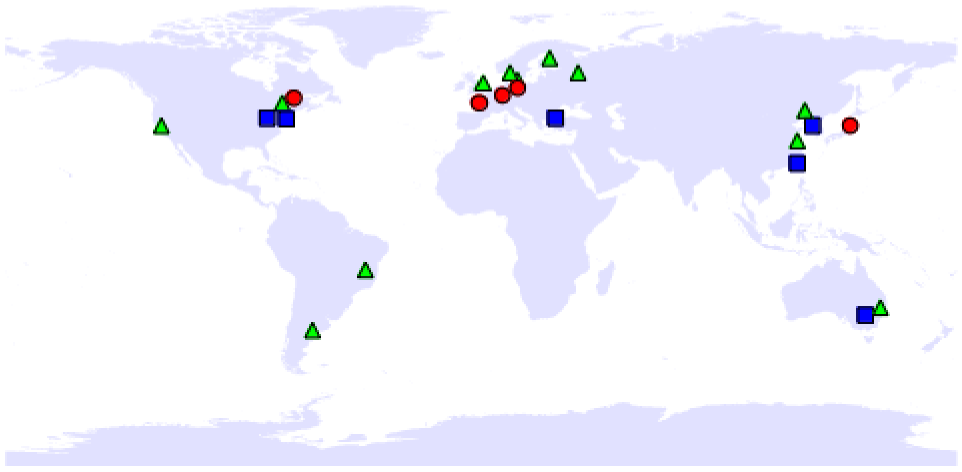


Figure 2.1: map illustrating the locations of the HRC scenes (green triangles), the RE scenes (red circles), and scenes taken by both sensors (blue squares).

general. Finally, the section concludes with a review of common strategies to transfer software code on to hardware in an optimal way.

2.3.2 Fundamentals

The general internal structure of an arbitrary FPGA consists of an array of Logic Blocks and Interconnections. These Logic Blocks are the basic elements of an FPGA. They consist of a Look-Up Table (LUT) and a Flip-Flop (FF). Several Logic Blocks compose a Slice, in which the number of Logic Blocks varies between different FPGA types. The larger or more complex an algorithm, the more Slices are necessary, and the larger a hardware implementation.
The intermediate Routing Structure shuffles signals between different Logic Blocks and additional units, e.g. Input-Output-Blocks which connect the FPGA to the 'outside world'. To this end, it is essential that every signal reaches its destination in one clock cycle. If the path is too long, e.g. due to a complex algorithmic structure, errors will arise and the whole implementation will not run properly. This is a crucial difference to a software program. As a result, there is a maximum clock frequency to every implementation up to which all signal timings can be met. A higher frequency will result in a malfunction. A diagram of an FPGA structure is shown in figure 2.2. It is a basic illustration for visualisation purposes only especially the routing architecture is much more sophisticated in an actual FPGA.
The logic and the routing elements are controlled by programming points. Every logic function or routing choice is controlled by a single memory bit. An FPGA is configured when all programming points are set. This is done via an upload of a

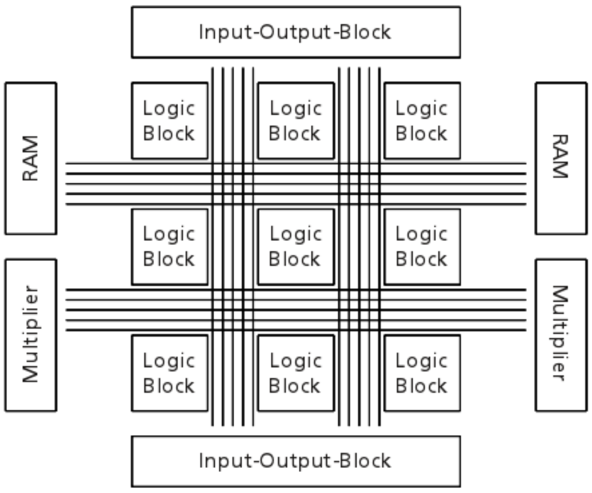


Figure 2.2: diagram of a basic FPGA structure. The array of Logic Blocks is connected by a Routing Structure (lines). In addition, there are units, such as RAMs, Multipliers and Input-Output-Blocks, to take over specific tasks or connect the FPGA to peripheral devices.

configuration file or bit-stream on to the FPGA whenever the chip is re-powered. Due to the fact that the configuration pattern can be modified or changed every time by the user, FPGAs are called Field Programmable.
Besides these basic elements, there are several additional units on modern chips too. They serve to enhance the flexibility and performance, while handling inefficient tasks with specified components on the chip. Examples of such extended logical elements are Multipliers and Random Access Memories (RAMs). A multiplication can be performed much faster and more economically on a Multiplier than within several Slices. Due to a highly complex Routing Structure, signals can be rapidly transferred to the Multiplier, and only the result is then routed to the next Slice. But the number of Multipliers is limited, so it is advantageous to avoid large multiplications whenever possible. RAM cells are very small and can hold large volumes of data. They can be used as Read Only Memory (ROM) when only coefficients have to be read out, or as a RAM in case of a buffering, for example, but their number is limited too. These limitations make hardware programming a challenging task.

2.3.3 The Programming Language *Handel-C*

As mentioned in the previous section, an FPGA is configured by an upload of a configuration file or bit-stream on to the chip. Traditionally, such a hardware configuration is created with hardware description languages such as VHDL or Verilog. But these languages operate at a very low level and require a deep understanding of the employed hardware, so transferring an algorithm to hardware can be a time-consuming task.

An approach to simplify the transferring or development of algorithms is taken by the language *Handel-C* (Bailey and Johnston, 2010). In Handel-C, an algorithm is written in a high-level language and the output is a gate level description, a so-called *Electronic Design Interchange Format* (EDIF) file. The syntax of the language is that of conventional C with several extensions, e.g. the `par` keyword to use parallelism (Alston and Madahar, 2002).
Handel-C implicitly executes routines sequentially, but to exploit the parallelism of hardware, the `par` keyword enables e.g. a parallel assignment of statements, as in the following example (Bowen, 1999).

```
par
{
   x = 1;
   y = 2;
}
```

Here, both variables are assigned in one clock cycle, whereby the same microprocessor implementation (without the `par`) would require two clock cycles. From this it is obvious, the more program parts are running in parallel, the larger the benefit of a hardware implementation.
The source code can be written, debugged and simulated in a graphical development environment, the *DK Design Suite*. When an algorithm is developed and 'finally released' for hardware implementation, the compiler creates an EDIF file. This EDIF file is then transferred to an FPGA vendor-specific software to create the final chip specific 'bit' file (configuration file). At this stage, the FPGA can be configured. The whole work flow will be explained in section 2.4.3.

2.3.4 Strategies for FPGA Implementation

A high-level hardware programming/ descriptive language makes it easier to implement an algorithm on hardware. Nevertheless, it is useful to follow some principal considerations in order to achieve a sound hardware implementation. Some rules are listed in the following text.
Basic calculations, like a multiplication or division, are in general not ideal for a direct FPGA implementation. Thus, dedicated hardware units were added to modern FPGAs to outsource these inefficient operations. However, due to the binary calculus of computational hardware, a division or multiplication by powers of two can be easily performed by a *bit shift* either to the right or left, respectively. So, the programmer can adjust or pose requirements that take advantage of this effect already at this stage of the algorithm design. In this study, the specification for a 1024×1024 pixel image size is such a useful requirement. The first pixel index in the xth row can be selected by adding $(x - 1) \times 1024$ to the first image pixel index. The multiplication is equivalent to a bit shift of $(x - 1)$ 10 times to the left, which is a hardware friendly operation.
Another common practice is the use of Look-Up Tables (LUTs) when dealing with

complex expressions, like square-roots, or multiplications and divisions by arbitrary numbers. Here, pre-calculated values are stored in an LUT and no actual calculation has to be performed on the chip. But this might imply some uncertainties due to rounding errors, and comes at the cost of storage capacities. So, whenever possible, such demanding operations should be avoided, or at least transformed to less complex terms (Johnston et al., 2004).

Eleven general algorithm transformation techniques are presented and discussed in the article of Bailey and Johnston (2010). Some of them will be introduced now because they have been used in this work. The first method is called *loop unrolling.* It means that, the innermost loops are replaced by sequential copies of the operations. This can significantly reduce the complexity of an algorithm.

At times, also the order of operations has an impact on the complexity and processing speed. Here, an *algorithm rearrangement* might result in hardware savings.

In general, any *substitution* of a demanding operation by a simpler one might enhance hardware consumption parameters or processing speed. If this involves approximations, the outcome of the algorithm might differ slightly but may still be acceptable. So, sometimes a trade-off has to be made. Overall, each algorithm should exploit available parallelism whenever possible.

2.4 Experimental Set-up

2.4.1 Software Pre-development

Direct implementation of an algorithm presented in literature is a challenging and often impossible task, even in software. In most cases, exact parameters are missing or crucial steps are not explained sufficiently. Thus, in the present study, several approaches were tried out and a lot of testing was accomplished.

In general, designing a hardware implementation is more time-consuming than a comparable software routine (Hauck and DeHon, 2007), thus the first two algorithms were developed in software at first. The Interactive Data Language (IDL) was used to program a first running version of the *Water Body Extraction* and the *First Bridge Recognition* algorithm, both of which will be presented in chapter 3.

During the hardware development process, severe changes had to be made to the software program code in order to establish a working hardware implementation. Regardless of some optimisations aspects, a running software code does not imply a working hardware routine. If arguments, calculations or loop operations, are too complex and the necessary run-time of signals exceeds a certain limit, then the implementation does not work properly and produces errors. A more detailed discussion will be found in the respective algorithm sections in chapter 3. As a result, all further algorithms were directly designed for an FPGA implementation using Handel-C.

2.4.2 The *Nexys-2* Development Board

An FPGA is a processing unit, but it has to be connected to some additional devices in order to make a proper system. Basic devices, such as a *Clock*, an *External RAM*, and *Input/Output ports*, as well as extras such as a row of *Switches*, are assembled

on the *Nexys-2* development board. It is a commercial-off-the-shelf product and not appropriate for direct application in space. However, it is well suited for the pursued technology demonstration on ground. The board, in combination with a Personal Computer (PC), represents the experimental set-up in this study. An image of the board, as well as indicators to some relevant devices, are shown in figure 2.3.
In the center of the board there is a XILINX Spartan 3E-1200 FPGA. XILINX is the name of the FPGA manufacturer. The FPGA is connected to a clock on the right-hand side that oscillates at a frequency of $50MHz$. Above the FPGA there is an external RAM with a capacity of $16MB$: the image data will be stored there. Due to a slow interconnection between the FPGA and the external RAM, the maximum clock rate of the FPGA has to be reduced to $12.5MHz$, that is $\frac{1}{4}$ of the original clock signal. Later, the processing times will be given at this clock rate, but the reader should keep in mind that higher clock rates are possible. A detailed description of the board is given in Digilent (2011).
The program flow can be controlled by activating certain Switches on the lower side of the board. Each Switch initiates a different routine, e.g. to load data from the PC into the FPGA, to process an image, or to copy the result back to the PC. The necessary image data are sent via a serial connection from the PC to the FPGA and back. The mandatory protocols for the data transfer were defined by my colleagues Kurt Schwenk and Katharina Götz. At this point, the author would like to thank them both for the preparation of the routines.
The power supply of the board is provided via a USB connection to the PC. The 'bit' file, to configure the FPGA, is transferred over the same USB connection. The generation of a 'bit' file from an EDIF file is performed by the *ISE Design Suite*. This software is provided by XILINX and allows the creation of configuration files for XILINX FPGAs. In the creation process, a lot of design-related adjustments can be made, e.g. particular optimisation strategies can be chosen with regard to the processing time or the power consumption of the chip. In the current set-up, a balanced design goal between hardware consumption and processing time was aspired. A subroutine of the Design Suite, called IMPACT, enables a direct upload of the 'bit' file to the connected FPGA. A complete description of the work flow will be given below.

2.4.3 Work Flow

The realisation of a hardware application involves several intermediate steps. An overview shall be given now. For clarity, the work flow is split into two parts: at first, the configuration of an FPGA is described, then the data transfer between FPGA and PC.

1. Write *Handel-C* code in the *DK Design Suite*.
2. Create an EDIF file from the source code.
3. Initiate a project in the *ISE Design Suite*, by specifying the FPGA and possibly choosing some optimisation strategies.
4. Load the EDIF file into the *ISE Design Suite* and create a 'bit' file.

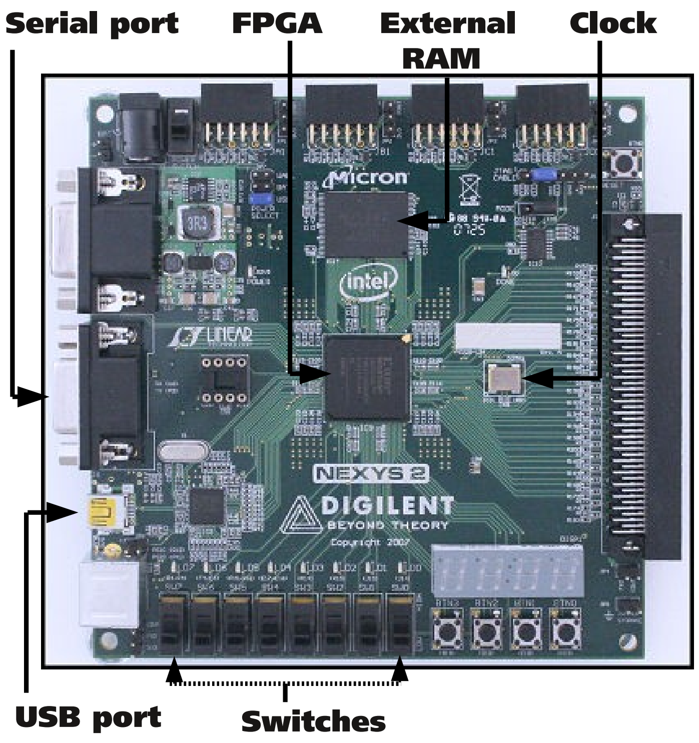


Figure 2.3: the Nexys-2 development board. The FPGA image was taken from Nexys-2 (2013). All relevant devices for the current set-up are named and indicated.

5. Load the 'bit' file on to the FPGA via IMPACT.

Now, the FPGA is configured and the image data can be uploaded to the board to be processed.

1. Set the serial port of the Nexys-2 into 'hearing mode' to get data from the PC into the FPGA.
2. Start the PC program that copies image data to the serial port of the PC, so that the data are uploaded to the Nexys-2, i.e. stored in the external RAM of the board.
3. Run the image processing routines.
4. Initiate the PC program that 'listens' at the serial port of the PC to fetch data from the FPGA and to store them on the hard drive.
5. Start the copy procedure at the Nexys-2 to send the results of the image processing from the external RAM via the FPGA on to the PC.

This work flow description illustrates the more complex development procedure of a hardware implementation when compared to coding a software. However, with regard to a future space application, a hardware realisation might have several significant advantages in the later space-proof verification process. In contrast to the very extensive procedure for a software assessment, with hardware a lot of potential malfunctions can be excluded a-prior and operational behaviour can be characterised more accurately. In the end, this might save time and money.

Chapter 3

New Algorithms for Image Analysis On-board Satellites

3.1 Overview

This chapter contains the main scientific contributions of this thesis, five novel algorithms to detect water areas, islands, bridges over water (two methods) and traffic on a bridge, in an acquired satellite image. All five routines are primarily developed for an application on-board a spacecraft and, for the reasons given in chapter 1, they have to run on an FPGA. So, besides an acceptable accuracy of the detection results, the following three algorithm design criteria have to be fulfilled:

- robust implementation,
- fast performance,
- autonomous processing.

The hardware implementation has to be robust, without any complex algorithm structures: the reasons for this were given in the previous chapter. A fast performance is aspired since large volumes of data have to be processed in a short time. And an automatic processing is necessary for the intended application in space. These requirements influence the algorithm designs and prevent a simple adoption of existing methods. Hence, new developments were necessary.
Each algorithm will be presented in a separate section. The first four methods refer to pattern recognition tasks, and the fifth algorithm is an anomaly detection. With regard to content, they compose a processing chain from basic water mask generation, via bridge over water recognition, up to traffic detection on a bridge. The output of each routine is drawn on a thematic map. On-board a spacecraft, it might be that the results are only stored as table entries that can be easily interpreted by the planning unit.
In principle, each algorithm can operate independently, provided that the necessary input data are available. It is possible to combine the methods in a modular concept or interchange different algorithms. An illustration of the algorithm composition applied in this thesis is depicted in figure 3.1. Both processing chains were executed separately. The processing stream on the left-hand side produces an output after a

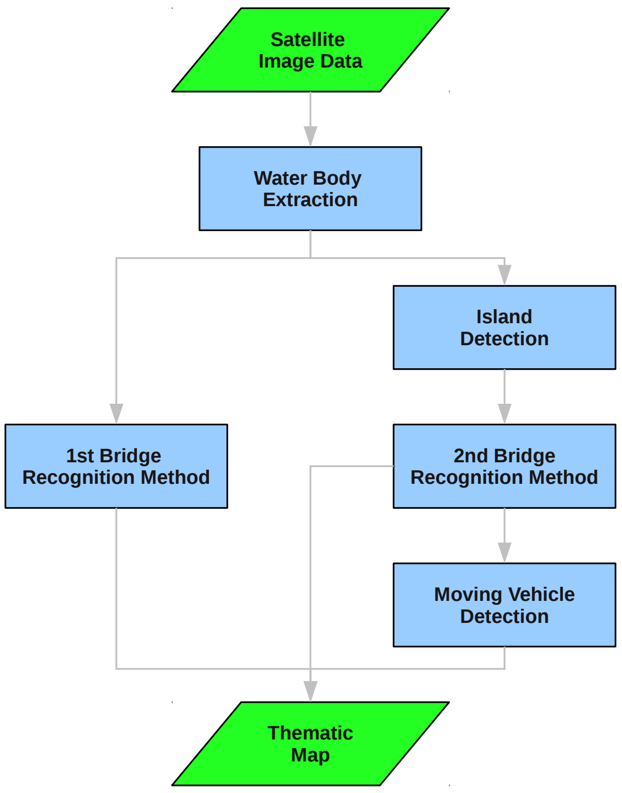


Figure 3.1: flow chart of used algorithm combinations.

short period of time. The resulting thematic map shows water areas and marked bridge center points. A more detailed thematic map can be obtained by following the process flow on the right hand side. Besides detected islands, also the whole bridge deck areas appear in the final thematic map. In case of RE data, a detection of moving vehicles can additionally be carried out.
This set-up provides a flexible program flow that can be adjusted to the actual needs of a future satellite mission.

3.2 Water Body Extraction

3.2.1 Introduction

The first algorithm in this thesis is a method for water body extraction. Every pixel of an image is labelled either as water or as non-water pixel according to certain specific rules. The output is a binary water mask. These rules, especially the respective parameters, are individual for each sensor system and data type. Even though there are already algorithms related to this topic in literature, none of them analysed or presented results for panchromatic HRC or NIR data from RE, thus it is not possible to simply adopt a method to the current case.
Besides the missing parameters, several methods presented in literature use computationally expensive operations to generate a water mask. For the reasons given in the previous section, such routines are not appropriate for a direct hardware implementation. So, a novel approach was required.
The proposed algorithm is, of course, developed on the basis of existing methods, thus in the beginning of this section a review of relevant literature is given. Then, the approach is presented in section 3.2.3, followed by a remark on the hardware and software implementations. The experimental results are presented and the section concludes with a discussion and a summary.

3.2.2 Literature Survey

The following literature survey shall give a brief overview about published approaches, and highlight ideas that are also used in this thesis.
Luo et al. (2007) presented a bridge detection method for IKONOS satellite imagery. The necessary water body extraction was performed with a *Gauss Markov Random Field Support Vector Machine* (GMRF-SVM). To implement and train an SVM is an extensive task and out of scope of this study. In section 3.2.5, there will be a direct comparison between results from their publication and our results.
The approaches of Chen (2008) and Liu et al. (2010) collect sample pixels of water and non-water regions in an image and adjust their classification procedure accordingly. The sampling of pixels is done manually by a human operator, hence the methods are not appropriate for an autonomous application in space.
Han et al. (2007) presented a *self-adapted method* that finds a threshold to separate water and non-water pixels automatically by an analysis of the image histogram. A clear, two-cluster shape in the histogram permits differentiation between darker water pixels and brighter non-water pixels. In the present study, preliminary tests on different satellite images were carried out, but the assumption of a two-cluster shape in the histograms could not be confirmed. Hence, the method does not seem practical.
Other approaches derive thresholds for water separation automatically from an image histogram as in Gu et al. (2011), or use a *fuzzy threshold segmentation* as in Fu et al. (2009). But these procedures involve many multiplications, divisions and square-root operations, thus they are computationally demanding and not appropriate for the aspired hardware implementation.
The review concludes with the publication of Trias-Sanz and Loménie (2003). They

use many parameters, such as the mean value, entropy, energy, variance, skewness and kurtosis of the image histogram, in order to distinguish eight different land-cover types. The computation of all these values is demanding and also does not seem to be advantageous for a hardware solution.
In conclusion, there are different approaches in literature to extract water areas from PAN or NIR satellite imagery. Multiple basic ideas are reasonable and are adopted in the current work as well. For example, the principle structure of the current approach is very similar to the one presented in Luo et al. (2007). The SVM part is, among other things, replaced by two 'simple' thresholds. Due to this, the procedure is more convenient for a hardware implementation. The new approach is presented below.

3.2.3 A Practical Approach

As already mentioned above, the principle structure of this new approach is similar to the one presented in Luo et al. (2007). The work flow consists of three parts: A *pre-processing* part to smooth the input image in the beginning, a *main part* to find the water pixels in the input data and a *post-processing* part to smooth the resulting water mask. The water pixel extraction process used in this thesis is less complex than the originally proposed one, in order to allow its use on an FPGA. Additionally, an extra post-processing process is introduced to enhance the accuracy of the result. A diagram of the work flow is illustrated in figure 3.2. The steps are described as follows.

Pre-processing

There is always noise in image data. The sources can be certain atmospheric conditions or noise directly in the sensor system. So, in general, it is advantageous to reduce the noise in the image data prior to some image analyses. The simplest way to do this is to apply a linear filter, such as a mean filter (Richards, 2012). This filter, with a 3×3 window, can be implemented on hardware easily and processes image data very fast. In case of very noisy data, the filter can be run several times to obtain the desired result. So it seems appropriate for the current purpose, and is applied in this thesis.

Water body extraction

A review of current literature revealed several approaches to extract water pixels from PAN or NIR satellite imagery. Although the applied methods differ, the basic idea is more or less the same: an analysis of the brightness (and texture) characteristics of each image pixel. In the PAN and NIR spectrum, water pixels are in general darker than their surrounding non-water pixels, and especially inland waters have a very smooth surface (Chen, 2008) or (Fu et al., 2009). By estimating the adequate thresholds, e.g. from the image histogram, both classes can be distinguished sufficiently. But as already mentioned, the estimation procedures proposed in literature are not practical for hardware implementation.
In order to simplify the procedure, the parameter estimation process is replaced by two constant thresholds. Both values are chosen very high, so that it is more likely

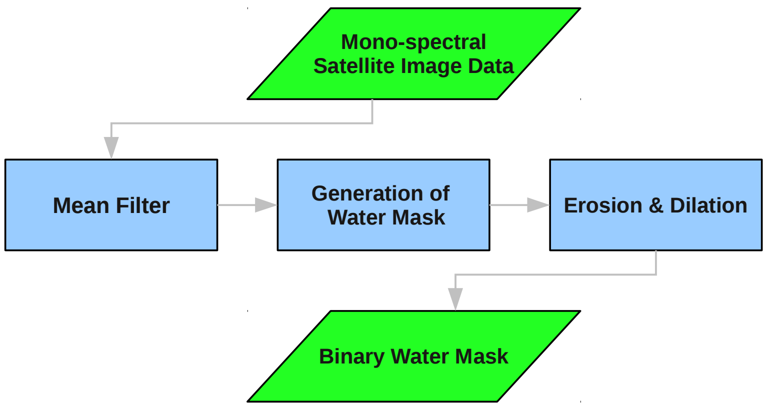


Figure 3.2: flow chart of the water body extraction algorithm.

to falsely classify a pixel as water than to miss a real water pixel, but in a way that the number of falsely classified water pixels is kept as low as possible. As a result, the final water masks often depict more water areas than there are visible in the original satellite images. But the implementation of this method on an FPGA chip is straightforward and a binary water mask is generated within a short period of time. A post-processing step is applied afterwards, to reduce the number of falsely classified water areas.

Post-processing

A post-processing step is executed in Luo et al. (2007) to remove small water regions caused by false positive classifications. The step involves several *erosion* operations applied to the water areas in an image. The same technique is applied in the present study. However, an erosion of all water bodies in an image results in the removal of smaller areas while, at the same time, also the size of larger water areas is reduced. To compensate this effect, the same number of *dilations* is applied to the water bodies after the erosion procedure. A so-called *opening* is performed. The reverse operation, first a dilation and then an erosion, is called *closing*. Here, as a result of the opening, the remaining water areas increase in size again.
Please notice that small water areas are not necessarily false classifications. So, in the erosion step also real water areas like small rivers will vanish. Hence, this step is a trade-off between a reduction of falsely classified water areas and an improper water mask. The accuracy will be evaluated in section 3.2.5.

Additional post-processing

The mean filter, applied in the pre-processing step, is not an edge-preserving filter (Richards, 2012). This means, each water pixel close to the water edge in the original image becomes smeared with adjacent non-water pixels during the filter process. In the end, these blurred water pixels are brighter and more uneven, and hence they are classified as non-water pixels. As a result, the water areas appear smaller and bridges appear broader in the water mask than they are in the original image. To overcome this drawback, it is possible to perform the dilation of water areas in the post-processing step, for example, two times more often than the erosion: the water areas in the water mask become larger and coincide better with the actual visible water areas in the original data. This might lead to an enhanced bridge recognition and so both adjustments (with/without an additional post-processing) are evaluated.

3.2.4 Software and Hardware Implementation

The present algorithm was first implemented with software. The program code was written in the high-level software programming language IDL (see section 2.4.1) and the program was executed on an Intel(R) Core(TM)2Duo T9600 CPU, with two kernels operating at $2.80GHz$.
After initial tests showed satisfactory results, a hardware implementation was realised too. The principle structure of both implementations is similar since the proposed method relies on the use of basic image processing techniques, as *mean filter* or *erosion* and *dilation*, in order to keep the complexity of the algorithm as low as reasonably possible. So, a working hardware implementation was established in a short period of time.

Run-length coding

The result of the actual water extraction process (prior to post-processing) is a binary water mask. In a bit representation, this mask consists of zeros and ones, and the whole bit-stream of the image is composed of sequences of zeros and ones. A sequence of a certain bit is called *run*.
A widespread technique in image processing is a transformation of such a binary image into a so-called *run-length encoded representation*. Due to the fact that the alternating sequence of zeros and ones is known, it can be more efficient to only declare the first bit and then the length of each sequence, than to store the original image representation. It is a loss-less compression technique, i.e. the encoded and then decoded image is identical with the original one, but the required memory space might be smaller in the encoded state. For images with large runs, this method can provide significant memory savings. Short runs, however, do not yield any advantages (Müller et al., 2005). Additionally, morphological operations, such as erosion and dilation (Breuel, 2008), or *connected component labelling*, can be performed on the encoded images as well (Appiah et al., 2008).
In the present study, however, no run-length encoding of the binary water masks was performed mainly for the following reason: the generated water masks are primarily the basis for the later bridge recognition procedures. The bridge recognition methods operate with an image representation of these masks, since the spatial arrangement of the water areas is of importance. This means that if a water mask would be run-length encoded, a decoding would be inevitable, before the bridge recognition can be executed. Water mask encoding and decoding does not seem beneficial, and morphological operations can be performed very efficiently, also on the original image

representation using a pipeline structured algorithm on the FPGA. Anyway, the compression technique might be an option, if only an economical storage of water masks is desired.

Comparison

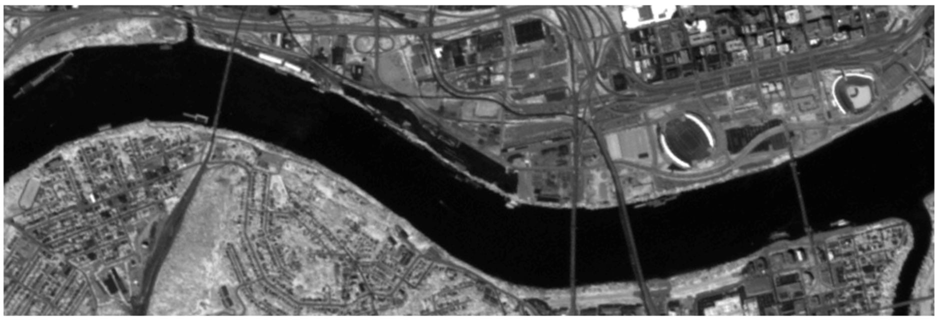
In the field of hardware engineering, it is common to perform a direct comparison between a hardware and software implementation, e.g. with regard to the processing time for an image. However, in this study, this does not seem appropriate for the following two reasons: at first, the software implementation was not optimised. Much more effort was put into the hardware version, so the comparison is not balanced. Secondly, it is outside the scope of this study that a CPU, like the one used in a laptop computer, will soon fly on a satellite mission. For one thing, the energy consumption is far too high. In contrast, the applied Spartan 3E FPGA has a size and power consumption that is affordable on-board current satellites. So, even if the outcome of the comparison would favour the use of a such a powerful CPU, it is not an option for the aspired space application. Nevertheless, it can be stated that the processing time for the software version was within a few seconds.

3.2.5 Experimental Results

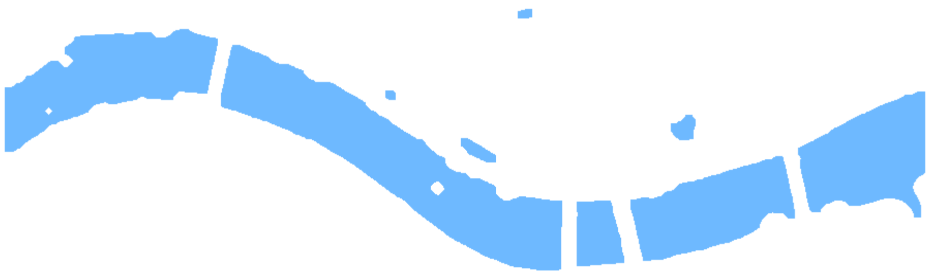
In the course of this study, several tests were carried out to verify and characterise the proposed water body extraction. The obtained results are presented below.

Example image

The principle outcome of the proposed method is a binary water mask. In the present thesis, the water pixels are coloured blue for visualisation purposes only.
Figure 3.3 (a) shows a small part of a scene showing the Ohio river in the city of Cincinnati (USA). For the details of the data take, see appendix A, table A.2, entry RE 2. All further references to the appendix of this thesis are abbreviated by giving the table number and entry name, e.g. in this case, A.2, RE 2. The image is a NIR satellite image taken by RE. It is enhanced in brightness and contrast for better visibility. The scene represents rather good imaging conditions, since there is a strong contrast between the water and non-water areas.
The mean filter was applied to the image data first. Preliminary tests showed good results, with one filter iteration for RE scenes and three iterations for HRC scenes. The thresholds for brightness and surface roughness were set to 20 and 1 for RE imagery, and 150 and 2 for HRC image data, respectively. The final erosion and dilation steps were executed three times on both image types.
The resulting water mask for image 3.3 (a) is depicted in figure 3.3 (b). In the water mask, the river course as well as all four larger bridges, indicated by the gaps between the river segments, are reproduced properly. Some smaller water areas appear on the upper side of the river, too. Not all of them are real water segments, but they are dark and homogeneous, thus classified as water.
The lower part of the small river on the right-hand side is not represented in the water mask at all. This is due to the erosion step in which smaller water bodies are



(a)



(b)

Figure 3.3: (a) an enhanced NIR image of the Ohio river in the city of Cincinnati, USA. ©DLR/RapidEye (b) The resulting water mask.

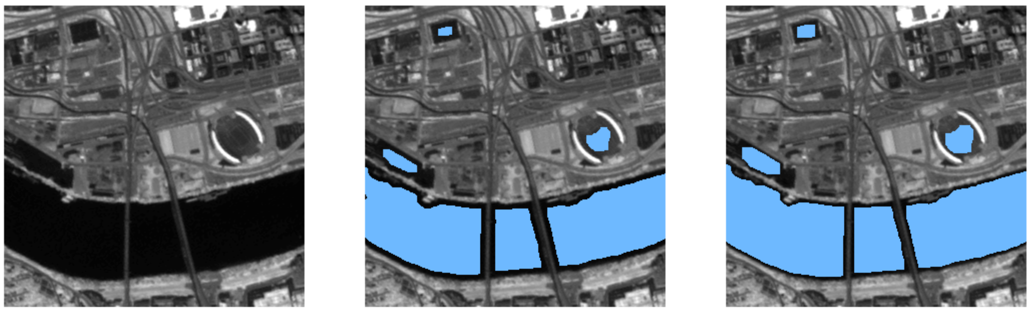


Figure 3.4: (a) a section of figure 3.3. (b) The section combined with the water mask, without additional post-processing (c) The section combined with the water mask, with additional post-processing.

erased from the map. A statistical evaluation was carried out and is presented in subsection *accuracy assessment*.

Additional post-processing

An additional post-processing step was presented in the description of this algorithm. The idea is to compensate the blurring effect of the mean filter and to enhance the accuracy of the water mask by performing additional dilation steps at the end of the water mask generation.
For better visibility, a section of figure 3.3 (a) is displayed in figure 3.4 (a). In figure 3.4 (b) this scene is combined with the result of the water body extraction without extra post-processing. It can be seen that the blue water (mask) areas are smaller than the dark water areas in the original image.
To compensate this drawback, the final dilation was performed two times more often than the erosion, i.e. five times. The result of the water body extraction with additional post-processing is presented in figure 3.4 (c). From the image, it can be seen that all water bodies became larger and the blue river segments from the water mask coincide better with the dark river segments of the original image. But the falsely classified water areas became larger as well.
So, the additional post-processing step is reasonable and yields a better compliance of true water areas in the water mask and the respective areas in the satellite imagery. But it also increases the size of falsely classified regions. A statistical evaluation was carried out and is presented in subsection *accuracy assessment*.

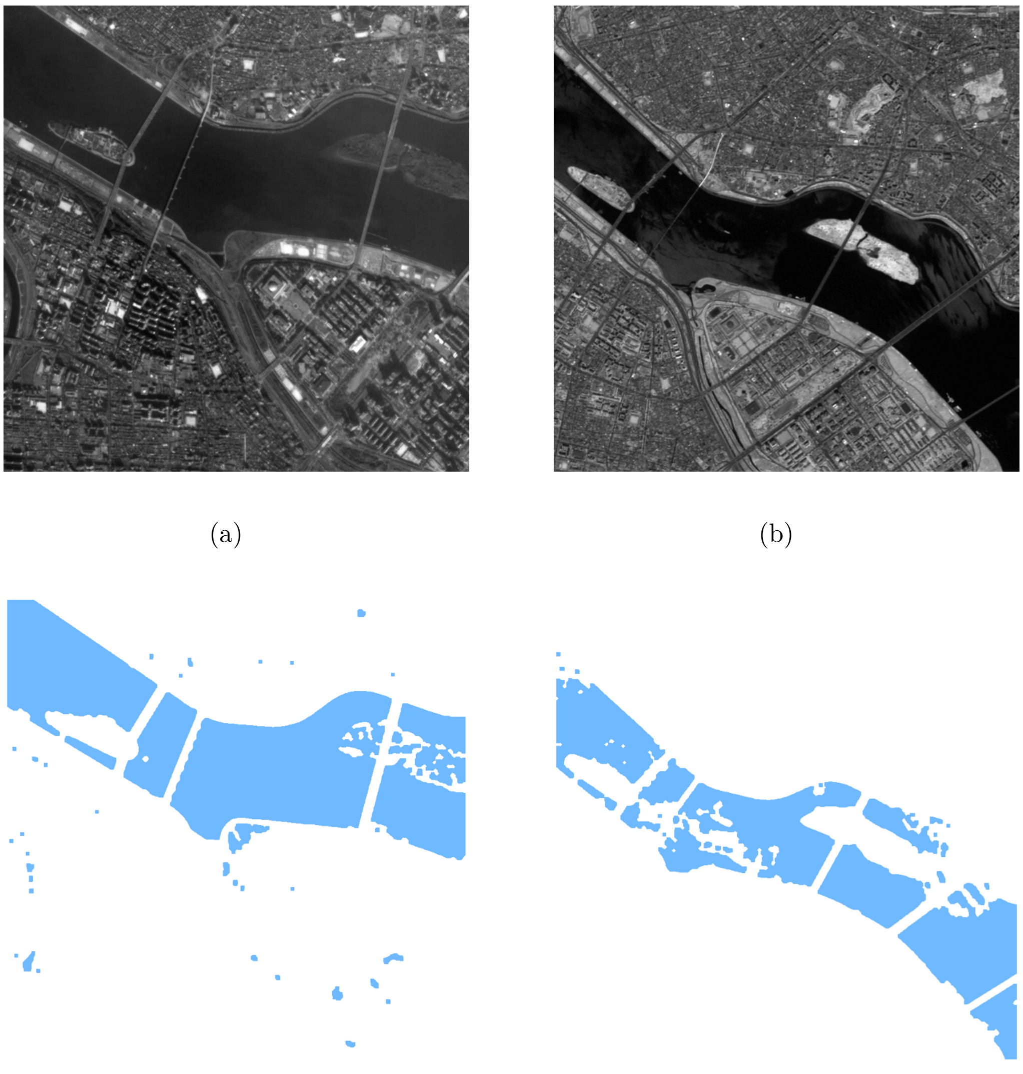
Comparison of ***HRC*** and ***RE*** imagery

The satellite imagery used in the study of this thesis was presented at the beginning of chapter 2. The data set comprises PAN data from the HRC sensor and NIR data from the RE satellites. The use of NIR data was motivated by an assumed increased separability of water and non-water pixels in NIR satellite imagery. This assumption was based on the fact that water has lower reflectance values in the NIR spectral

region than in the visible (PAN) spectrum (Liu et al., 2010). In this subsection, the influence of the input data on the water body extraction shall be appraised.
To gain a proper comparison between HRC and RE data, almost the same scene was selected from the data set and is depicted in figures 3.5 (a) and (b). The images show the Han river in the city of Seoul (KOR). Details of the data-takes can be obtained in appendix A.1, HRC 6 and A.2, RE 15. From the images, it can be seen that the contrast between water and non-water pixels is much stronger in the NIR image (fig. 3.5 (b)) than in the PAN image (fig. 3.5 (a)), as expected.
The respective water masks are displayed in figures 3.5 (c) and (d). The river course is visible in both masks, although the classification is not perfect. In figure 3.5 (c), a lot of additional smaller water bodies appear around the river, and the island on the upper right side of the image is partially marked as water. In contrast, almost no false classifications appear in figure 3.5 (d). But the water extraction failed in some parts of the river due to bright spots in the water. In the end, it seems that the actual appearance of water has a greater impact on the extraction than the applied sensor data. An accuracy assessment is presented below.

Accuracy assessment

Three individual examples were presented in the previous subsections: in this subsection, the statistical evaluation of the whole data set is given.
The procedure for the conducted accuracy assessment is adopted from Yuhaniz and Vladimirova (2009) and Liu et al. (2010). A so-called *ground truth* was generated manually by a human operator. The ground truth is water masks derived from satellite imagery by visual inspection. They shall represent a rough-and-fast labelling of all water bodies in an image. No pixel-precise analysis is demanded.
Both maps, the ground truth and the automatically generated water mask, were semi-transparently put on top of each other by the software GIMP. Discrepancies became clearly visible due to a different colouring. Additionally, in order to get a more accurate view of the results, the obtained error was distinguished into two different classes. According to Richards (2012) there is the *error of omission* and the *error of commission*. The first one represents here water pixels that are not detected by the algorithm. The second error describes water pixels in the generated water mask that are non-water pixels in the ground truth. The errors shall give an indication of whether the water body extraction under- or overestimates the actual water areas.
At first, there is an accuracy assessment of the water masks derived by the method **without** additional post-processing. The results are displayed in table 3.1. An overall accuracy of $79 - 86\%$ was achieved. As already mentioned, it is expected that water areas appear smaller in the water masks than they are in the original satellite images. This effect is reflected in the relatively high ($14 - 21\%$) error of omission. The error of commission is $8 - 10\%$. So, around one tenth of all water pixels in a water mask are too many. Overall, the performance of this method is similar for both sensor systems.
The results obtained by the method **with** additional post-processing step are presented in table 3.2. As expected, the extracted water areas coincide better with the ground truth, so the overall accuracy increased to $85 - 92\%$. In contrast, the error of omission falls to $8 - 15\%$. Due to the enlargement of all water areas, also the falsely



(a) (b) (c) (d)

Figure 3.5: a scene of the Han river in the city of Seoul, KOR. (a) A panchromatic image taken by the *HRC*. ©ESA (b) An enhanced NIR image taken by *RE*. ©DLR/RapidEye The respective water masks are depicted in figures (c) and (d).

Data	Commission error [%]	Omission error [%]	Overall accuracy [%]
HRC	9.5	14.4	85.6
RE	8.7	20.9	79.1

Table 3.1: accuracy of water body extraction **without** extra post-processing

Data	Commission error [%]	Omission error [%]	Overall accuracy [%]
HRC	13.6	8.9	91.1
RE	12.8	14.6	85.4

Table 3.2: accuracy of water body extraction **with** extra post-processing

classified ones were expanded and hence the error of commission rised to $12 - 14\%$. Again, the method works similarly for both data types. A discussion on the results can be found in section 3.2.6.

Comparison with literature data

A statistical evaluation of the accuracy of a method strongly depends on the applied satellite image data. So it is not quite balanced to judge the performance of a method only by comparing the statistics. For this reason, a direct comparison was also performed with results from literature. The comparison is not absolutely accurate since different satellite imagery was used. However, an analogy of two methods can be appraised by visual inspection.
As was already mentioned in section 3.2.2, a comparison with the method of Luo et al. (2007) was conducted. For this purpose, satellite imagery of an identical location taken by the HRC, the RE satellites and the IKONOS satellite was selected. Please note that the spatial resolution of the IKONOS image is $1m$. However, besides the differences in spatial resolution and sensor characteristics, it seems a reasonable evaluation of the quality of the method proposed in this study against a more sophisticated approach from literature.
In figures 3.6 (a), (c) and (e), an almost identical part of the city of Taipei (TPE) is depicted. For details on the data takes, see appendix A.1, HRC 5 and A.2, RE 3. All three images show the courses of the Danshui and the Jilong rivers. Several bridges of different size cross the rivers. The resulting water masks are displayed in figures 3.6 (b), (d) and (f). Since the water areas were coloured black in the literature, here all water segments are coloured black too. The water masks presented in figure 3.6 (b) and (d) were generated by the method proposed in this thesis. The water mask depicted in figure 3.6 (f) was derived from the IKONOS satellite image (fig. 3.6 (e)) using a GMRF-SVM. The IKONOS scene, as well as the respective water mask, were taken with permission from Luo et al. (2007).
Obviously, the IKONOS scene has the strongest contrast between water and non-water areas and the outcome is a well-classified water mask. However, the water

masks derived from our own experiments depict the river courses as well. In the end, it is difficult to say whether the differences in the classification results arise due to the applied method or the imagery. A discussion will follow in section 3.2.6.

Case study of a natural hazard: the 2011 tsunami in Japan

The methods provided in this thesis shall, among other things, support the work of crisis information centres. So, in order to assess the performance of the developed water body extraction under real disaster conditions, satellite image data from the 2011 tsunami in Japan was analysed.
On 11th March 2011, an earthquake off the north-east coast of Japan caused a tsunami that struck the coast of Japan with a widespread flooding. In some areas, the maximum wave height was about $19.5m$, and it reached $4.5 - 5km$ inland (Mori et al., 2011) and (Goto et al., 2011).
RapidEye imagery captured on 19th March 2011, 8 days after the tsunami, is used in the study of this thesis and one of these images is presented in figure 3.7 (a). The image shows the river mouth of the Natori river and a part of the coastal region. Details of the data take can be obtained from A.2, RE 17. Brightness and contrast of this NIR image are again enhanced for better visibility.
A water mask was derived from this image and is presented in figure 3.7 (b). Besides the river and the coastal waters, a lot of land areas are classified as water because their appearance is homogeneous and dark too. This is probably due to the still wet ground caused by the flooding. So, the additional water areas are not necessarily false classifications. The area is affected by a flooding event and the water mask reflects this. The result is discussed in section 3.2.6.

Hardware configurations

The experimental set-up was presented in chapter 2. At this point, some algorithm-specific details of the hardware implementation shall be revealed. The realisation of water body extraction in the hardware used needs 2842 slices, that is approx. 32% of the chip's total number of slices. No BlockRAM is needed and the power consumption is specified in the ISE design suite at approx. $0.5W$. The maximum clock frequency up to which an algorithm can be operated on an FPGA is limited by the signal run-times of the implementation. For water body extraction, a maximum clock rate of $98MHz$ was estimated by the ISE design suite. This frequency is obtained for a *balanced design goal* and might be increased by a *timing optimised design goal*. But, timing optimisation might eventually lead to a higher resource consumption.
In the current state, with a clock rate of $12.5MHz$, a water mask is derived from a 1024×1024 pixel image in $6 - 8s$. The times vary with the size and number of water segments in an image. If additional post-processing is performed, the processing time increases to $14 - 18s$. This non-linear rise in time is due to the non-linear enlargement of the water segments. However, considering an around eight times higher maximum clock frequency, the processing time will decrease accordingly. So the times represent just a basic indication.

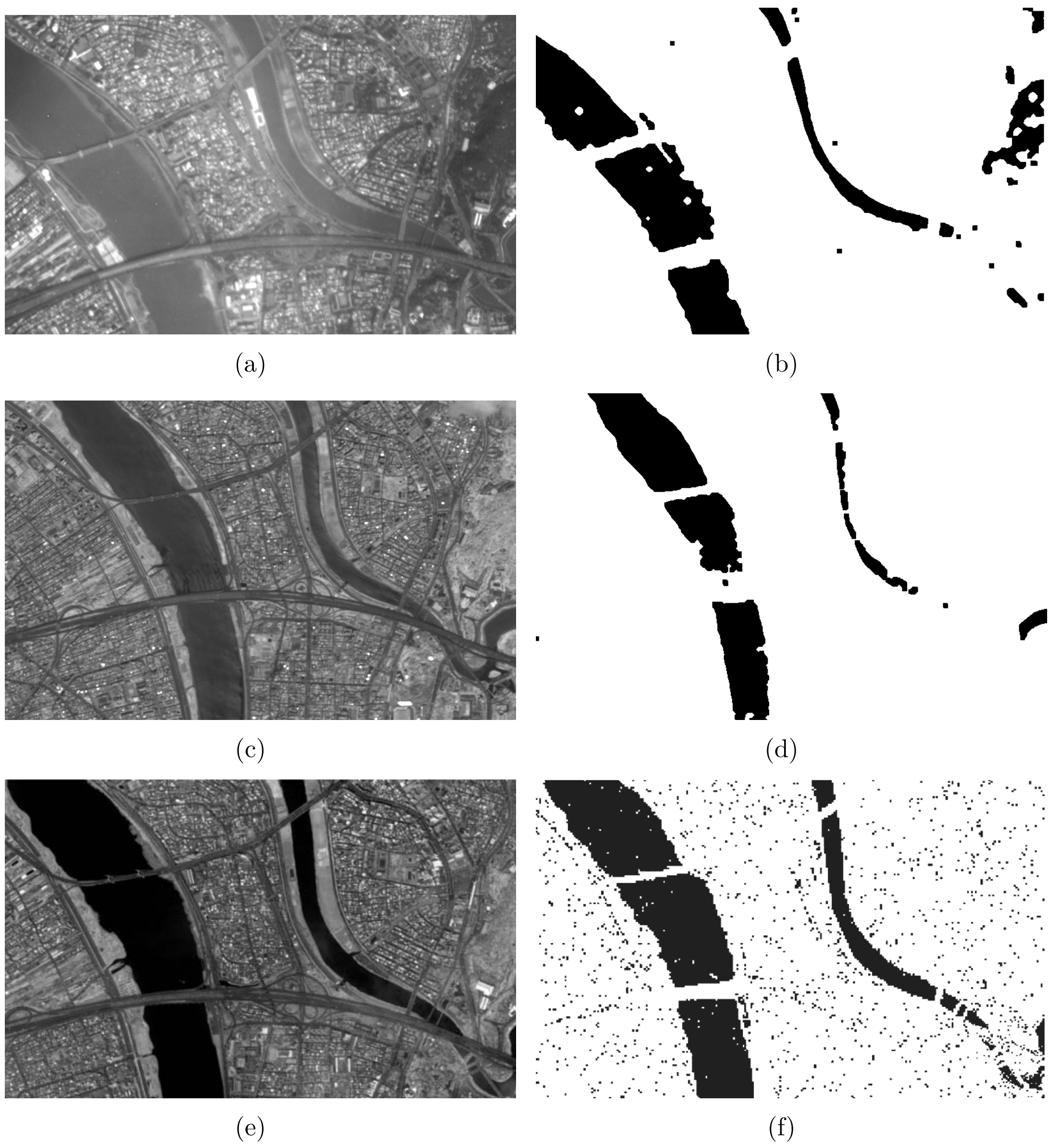


Figure 3.6: comparison with data taken from literature. Displayed is a part of the city of Taipei, TPE. (a) The PAN image taken by HRC. ©ESA (c) The enhanced NIR image from RE. ©DLR/RapidEye (e) The PAN image acquired by the IKONOS satellite taken out of the article by Luo et al. (2007). Figures (b), (d) and (f) depict the respective water masks.

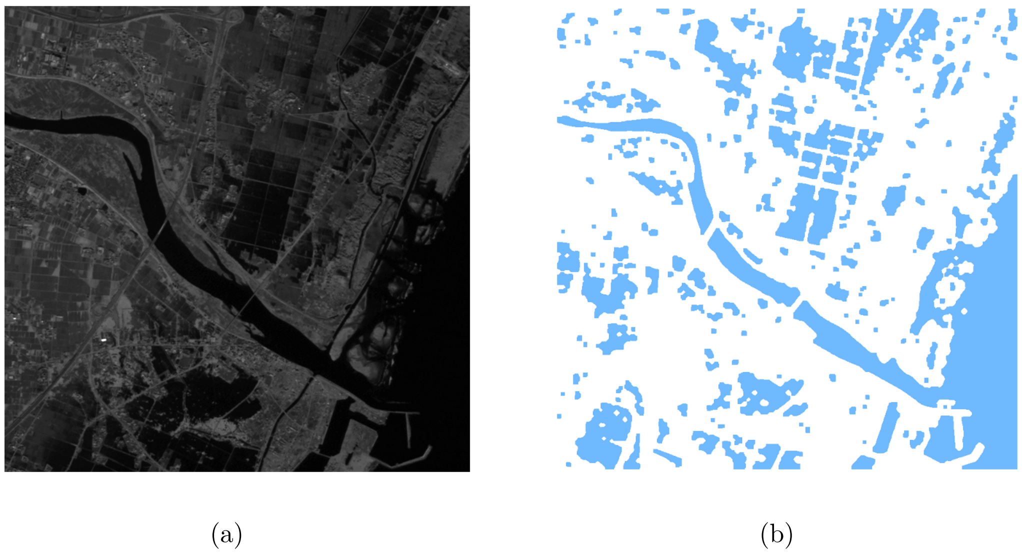


Figure 3.7: (a) an enhanced NIR image of the river mouth of the Natori river in Japan taken by RapidEye. ©DLR/RapidEye (b) The derived water mask.

Slices	2842 (32%)
BlockRAM	0
Max. Clock Frequency	$98MHz$
Power	$0.5W$

Table 3.3: *Nexys-2* implementation parameters

3.2.6 Discussion and Summary

The water body extraction proposed in this thesis is a simplified approach based on existing ideas for water classification in satellite imagery. The quest is for a method that is well suited for an FPGA implementation, while providing acceptable results in a short time.
Under good imaging conditions, i.e. a sharp contrast between dark water pixels and bright non-water pixels, satisfactory results can be obtained. Overestimations of water areas arise due to additional dark and homogeneous areas, such as forests or shadows of houses. Bright and turbid waters lead to a false classification of water pixels as non-water pixels. These issues have been addressed in Liu et al. (2010), Gao et al. (2011) or Gu et al. (2011) before, but no solution was presented there, so these situations remain a challenge.
The fact that water areas appear smaller in the water masks than they are in the original satellite image is compensated by applying additional post-processing. The outcome yields the desired result, but the processing time of an image increases significantly. Additionally, the small increase in the overall accuracy involves a rise in the error of commission at the same time. However, it might be the case that a water

mask with additional post-processing will lead to an enhanced bridge recognition. Thus, both bridge recognition methods will be tested on these water masks as well.
The introduced comparison with literature data is not perfectly balanced, since the IKONOS data has a five times higher spatial resolution and a much stronger contrast between water and non-water pixels than the used satellite images from the HRC and RE. But the water masks generated with the method presented in this thesis reflect the river courses sufficiently and give an impression of the water areas in that scene. So, even under more demanding imaging conditions, it is possible to achieve reasonable results with the proposed method.
The water mask obtained from the tsunami disaster satellite image looks disturbing at first, since there is a large number of additional water bodies visible. However, these areas are not necessarily false classifications but probably still wet regions after the flooding. A change detection procedure would pay attention to these areas, and a first survey of the affected areas could be provided.
In conclusion, the processing time of only a few seconds for a 1024×1024 pixel image and an overall accuracy of more than 79% is certainly appropriate for the current objective.

Summary

In this section, a simplified water body extraction process is presented. The method can derive a binary water mask from PAN or NIR satellite image data and is implemented on an FPGA. The image data are processed in a few seconds, and under normal imaging conditions all major water bodies are reflected reliably in the mask. Even under difficult illumination and water conditions in an image, sufficient results can be obtained. Thus, the generated water masks are the basis for all further algorithms in this thesis.

3.3 First Bridge Recognition Approach

3.3.1 Preface

An automatically derived water mask provides, in case of a flooding, a first survey of the affected areas. Yet, besides a knowledge on the extent of a flood, there is also an interest in the accessible infrastructure, such as roads or bridges (Mittelbach, 2013) and (Danzeglocke, 2013). From the perspective of a satellite, bridges over water are adjacent to water areas, thus an analysis of water masks might provide information about potential transportation routes. So, to advance the capabilities of an on-board image analysis further, a novel method to locate bridges over water in satellite imagery is presented in this section.
The proposed method is implemented in an FPGA. So in addition, it is a demonstration that also high-level image analysis tasks can be realised directly in hardware and no parallel microprocessor set-up is necessarily required.
In the section below, a review of existing methods for bridge detection will be given. Then, the new algorithm will be presented, as well as its implementations in software and hardware. After that, the results are displayed. The section concludes with a discussion and a summary.

3.3.2 Literature Review

The autonomous detection of bridges over water is a rather particular topic in the field of remote sensing. Nevertheless, there has been a slight increase in the number of publications over the last few years. A selection of existing methods for panchromatic satellite imagery is given in this section.
The article of Loménie et al. (2003) aims for a detection of all kinds of bridges. This includes bridges over water as well as bridges over roads and railways. At first, they conduct an advanced terrain classification and separate eight different land cover types. Then they search the classified image for parallel segments, the sides of a bridge. Their assumption is that the two sides of a bridge have almost the same length and orientation in an image. This sounds reasonable, however, own initial tests produced a lot of false positives and showed that this is a rather challenging approach. Additionally, the computation of angles is not a hardware-friendly task, and the comparison of different orientations is not unique. The detection rate given in literature of less than 42% is not promising either, hence this method does not seem suitable for the current purpose.
The approach of Luo et al. (2007) does seem relevant to the present study. They reduce the appearance of a bridge to a single line, perform a validation, and get as a result the medial axis of a bridge. Unfortunately, the method requires a knowledge of the width of each bridge in an image. This is not practical on-board a spacecraft, so the method cannot be applied here. The same drawback holds for the method presented in Fu et al. (2009). However, some intermediate steps look similar to stages of the method presented in this thesis.
The bridge detection technique presented in Gu et al. (2011) evaluates, among other things, the orientation of potential bridge deck areas and adjacent water segments in order to verify a bridge. Several multiplication, division and square-root operations

in the algorithm, however, prevent a fast hardware realisation.
In conclusion, there exist different approaches to locate bridges over water in satellite imagery. The direct neighbourhood of a bridge to two (larger) water segments, in combination with a search for parallel lines, is an often applied strategy in several publications on bridge recognition. This principle idea is also adopted to the method presented in this thesis. The main goal of this novel development, however, is to establish a simplified algorithm that is suitable for direct implementation into hardware. The new approach is presented below.

3.3.3 A Novel Method

The first bridge recognition method presented in this thesis was developed with the ambition to locate as fast as possible all bridges over water in a captured satellite image. The main difference to existing methods is the simple algorithm structure that enables direct implementation into a small FPGA. The input data is a binary water mask, thus the procedure is independent of the actual sensor data.
The idea behind this approach is a successive reduction of relevant pixels in the water mask, until only the center points of the located bridges remain. This yields a very fast algorithm performance and provides the answer, whether a bridge was found or not. A visualisation of all the processing steps is given in fig. 3.8. Each stage is listed on the left-hand side. The corresponding (intermediate) results, obtained from a real bridge image, are displayed on the right-hand side. Active pixels are coloured black, and the dots in the lower three images are enlarged for a better visualisation. In conformance with the later thematic maps, the final bridge center point is coloured light green. A description of the algorithm is as follows:

1. To reduce the required memory space and to minimize the computational effort, only the water rim is particularly labelled and examined in the next steps.

2. All directly adjacent water rim pixels are clustered to segments. Each segment is allocated a unique label. Additionally, the size of each segment is determined. If it is below a certain threshold, the water rim is erased and the segment is neglected from further consideration.

3. The image is scanned for regions which presumably form a bridge. All pixels of one water rim segment are marked which have one or more pixels from another water rim segment within a predefined vicinity. If a bridge is clearly broken, then both water segments will melt together, forming one water segment in the water mask, and no bridge will be recognised. In case of a proper bridge, the result is a pair of more or less U-shaped water rim segments on either side of the bridge. This looks similar to the result published in Fu et al. (2009). However, in the present work, the mentioned ‘vicinity’ is a fixed value that is independent of any bridge size. So, in contrast to literature, no a-priori knowledge of the bridge or river width is necessary.

4. To reduce the number of relevant pixels further, only the end-points of the U-shaped segments are labelled. These four points frame the bridge and the goal is to extract the center of them. However, because up to this step the

algorithm does not 'know' that these four points belong together, some ancillary intermediate steps are necessary.

5. There are two end-points, on one side of a bridge, that have the same segment number. They are selected and their center point, the so-called *support point*, is calculated. This linear interpolation prohibits the recognition of curved bridges, at least, but it is the simplest way to implement the method. Then, these support points are marked on both sides of a bridge.

6. A predefined surrounding area around each support point is scanned for the appearance of another support point. If another one is found, the center point of both points is derived and marked. This is the final center point of the bridge.

The algorithm was implemented in software and hardware and a comparison of both realisations is given below.

3.3.4 Comparison of the Software and Hardware Realisation

As for the water body extraction, also the first bridge recognition was programmed in software at first in order to validate the algorithm's performance. Since the proposed method implies several strong simplifications compared to existing methods, initial tests should ensure that the novel approach yields proper results as well. Yet, there are disparities between the software and the hardware versions, and a comment on both implementations is given in the following text.
Again, it shall be noted that the usual comparison between the software and hardware implementation is omitted in this study. The reasons for this were already given in section 3.2.4. Especially the stages of optimisation differ significantly between the two implementations. The processing time of the software version, however, is also in the range of several seconds.

Reduction to the water rim

The procedure is straightforward: all water pixels in the water mask are checked for a direct connection to a non-water pixel. If a water pixel is adjacent to a non-water pixel, this water pixel is assigned as a water rim pixel. The step was executed in the software version after the water segment labelling. However, for the hardware implementation, it is crucial to perform it in the beginning, since the reduction of relevant pixels drastically reduces the amount of necessary memory space in the labelling process.

Labelling the water rim segments

As was mentioned in the literature review, a bridge is in general between two different water segments. So, each water segment gets its own label, i.e. a number, to distinguish it. The applied labelling technique is called *region growing* and was employed on an FPGA in the article of Pedrino et al. (2010). During the label process, extra

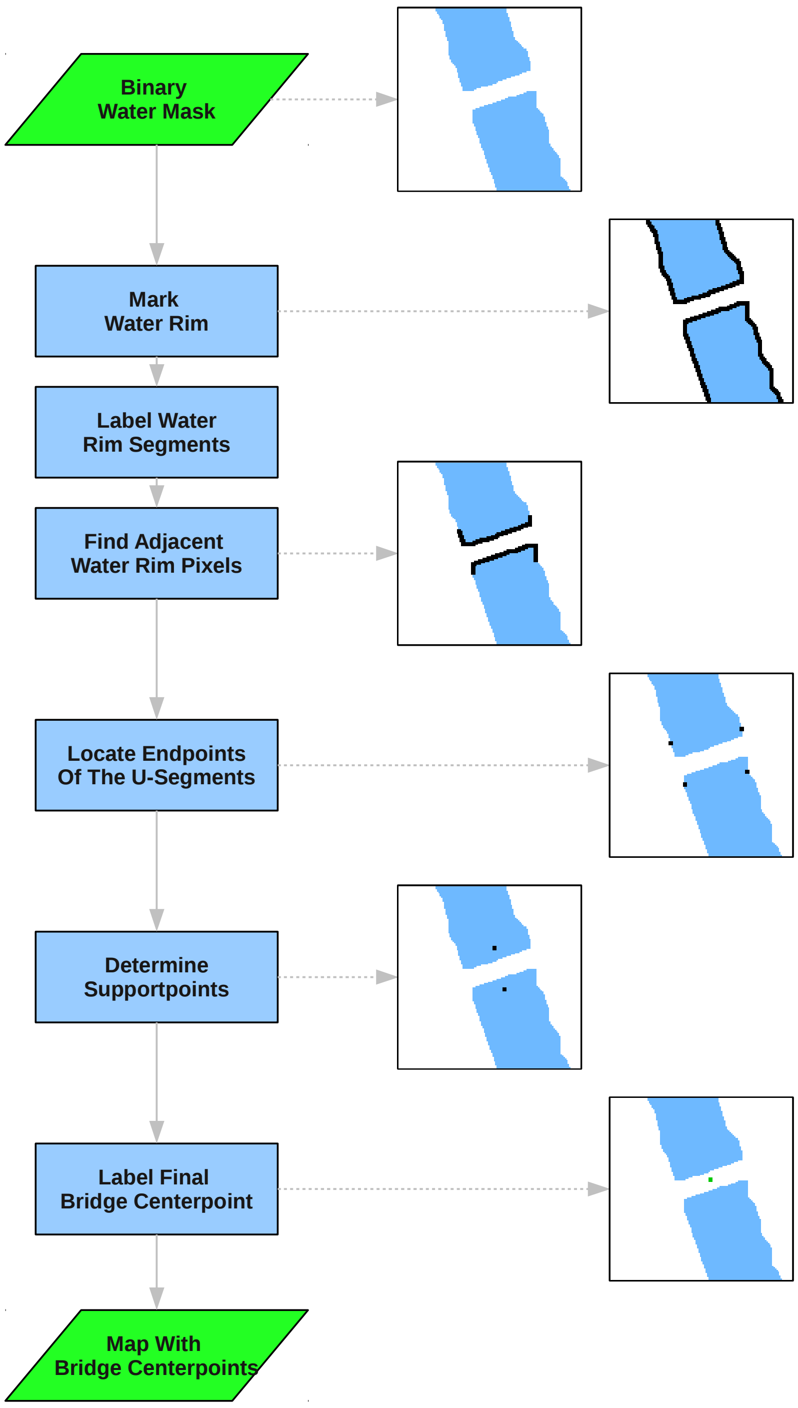


Figure 3.8: flow chart of the first bridge recognition algorithm.

memory space is needed to store some intermediate values. In the current implementation, the FPGA-internal BlockRAM is used. The required amount of memory space has to be defined in advance, so the usual segment size has to be estimated beforehand. If large segments are anticipated, a lot of allocated memory space is required. At the same time, the size of each segment is determined. If a water rim segment is shorter than 150 pixels (value determined empirically), it is not considered in the process any more.

Identification of adjacent water rim pixels

In order to find the water rim pixels of another water rim segment, a squared 'ring' template in a distance of 20 pixels for RE images and 31 pixels for HRC images around each currently regarded water rim pixel is searched for another water rim pixel from a different segment. A diagram of the template is illustrated in figure 3.9 (a). If at least one pixel is found, the current water rim pixel is assigned as a *potential bridge pixel*. The 'radius' of 20 pixels is derived from the statement that the widest bridge on earth is the *Blue Bridge* in St. Petersburg (RUS) with a width of $97.3m$ (Blue-Bridge, 2013). Every bridge is assumed to be smaller than $100m$, thus the distance between two sides of a bridge, in the applied data here with a spatial resolution of $5m$, is less than 20 pixels. The different radii between the template for RE and HRC imagery result from test experience. In general, the gaps between two water segments, indicating a bridge, are larger in the HRC imagery, thus to compensate this effect the search radius is enlarged.

Location of the end points of a U-segment and the support points

Adjacent potential bridge pixels are clustered to segments and the cluster size is checked. If a cluster is smaller than 50 pixels, the segment is discarded. This step is necessary to reduce false positives, but it is a challenge for the detection of smaller bridges too.
A significant difference between the software and the hardware version comes with the determination of the support points. In general, there is one support point on each side of a bridge and they are particularly labelled. The calculation of a support point, i.e. the computation of the center of a segment, involves the summation of and the division by the number of potential bridge pixels of one segment. Generally, this is an unknown number of pixels and the calculations are not recommended for hardware implementation. Thus, this algorithm part was changed in the hardware version and only the two outer pixels of each segment are considered. The two points constitute a fix number in the calculations and a division by two can be executed easily by a *bit-shift* operation. Finally, their center point (support point) is determined. But these simplifications make the algorithm also vulnerable since the recognition now strongly depends on the location of these end points.

The final bridge center point

The determination of the final bridge center point is done in the same way as the adjacent water rim pixels were found. Around every support point, a template area

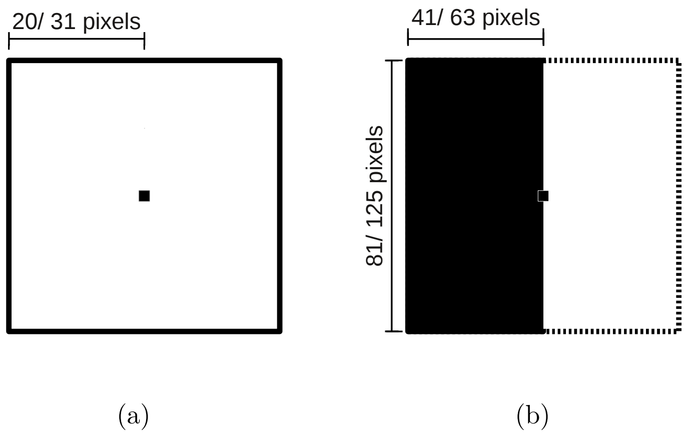


Figure 3.9: visualisation of the applied scan pattern. (a) The 'ring' template (black line) around the current pixel (black dot). (b) The semi-square area template on the left-hand side of the current pixel in the center.

of 41 × 81 pixels for RE images and 63 × 125 pixels for HRC images is searched for another support point. The size of the areas is derived from the following consideration: an assumption is made that there is a bridge with a width (not length!) of only one pixel. In this case, both support points have a distance between each other of around twice the search radius for adjacent water rim pixels. That means, that each support point has to scan an area to the extent of twice this search radius. Since the water rim pixel search radii differ for HRC and RE imagery, also the search radii for two adjacent support points differ accordingly. An illustration of the template can be seen in figure 3.9 (b). The entire area is scanned since the distance between two points is unknown. In addition, each template is only half of a squared area, it is somewhat the 'backward view' to each support point. This reduces the size of the regarded area and ensures that each pixel pair is only marked once. In the present study, the two quadrants on the left-hand side of the central pixel are chosen. Of course, it is also possible to chose the two quadrants above the central pixel to achieve the same result. After screening, the center point of each pair of support points is calculated. A last check is carried out to validate that the determined center point is a non-water pixel, then it is marked.

In conclusion, all aforementioned processing steps depend on the image content or the extracted image information of previous steps, thus only less strict real-time criteria can be fulfilled. The comparison also highlights several implementation strategies that were recommended for transferring software code on to hardware (see section 2.3.4). In some cases, necessary simplifications or optimisations were challenging, thus all further algorithms in this thesis were realised directly in Handel-C. The overall results of the software and hardware implementation are similar, so no detailed analyses were performed on the effects of some differences. An approach to overcome some limitations of this procedure in general is given with the *Second Bridge Recognition* method in section 3.5. The obtained results are presented in the section below.

3.3.5 Survey Results

Example image

The example image is the same scene as for the water body extraction in section 3.2, it shows a part of the Ohio river in the city of Cincinnati (USA). Figure 3.10 (a) is the enhanced NIR satellite image combined with the results of the first bridge recognition. Located bridge center points are illustrated as green dots. From that image it can be seen that all four major bridges were identified. Only the small bridge in the lower right corner was not recognised. To obtain an explanation for the failed detection, the water mask, that is the basis for the image analysis, is presented in figure 3.10 (b) as well. Also this image is combined with the bridge detection results. In the water mask, the lower part of the small river is missing, thus even for a human interpreter it is impossible to recognise any bridge there. So the failed recognition originates from an improper water mask and not from the present method. The additional false positive water areas caused no problems since they are too small.

Statistical assessment

A statistical assessment of the first bridge recognition method was carried out on 40 satellite images. In total, 178 bridges were visible in the images. In the conducted experiments, it turned out that the length of a bridge has a significant impact on its recognition. So at first, three bridge classes were defined relative to their lengths l. There are *Short Bridges*, with a length of 13 pixels or less. At the given spatial resolution of $5m$, this corresponds to a length of $65m$. *Medium Bridges*, with a length larger than 13 and less than or equal to 40 pixels, here $200m$. And there are *Long Bridges*, with a length of more than 40 pixels. In the definition used in this study, only that part of a bridge which spans the river is considered a ‘bridge’. Hence, the length of a bridge could only be determined by manual measurement. A template was put on top of the original satellite image and, by visual inspection, the assignment of each bridge to one of these three classes was performed. A listing of the classes is given in table 3.4.

Short Bridges	$l \leq 13$
Medium Bridges	$13 < l \leq 40$
Long Bridges	$40 < l$

Table 3.4: defined bridge classes (in pixels)

The classification of bridges provides a more accurate view of the obtained results. Not every recognition error is a shortcoming of the method. ‘Short bridges’ cannot be detected if the associated rivers were erased in the erosion step of the water mask generation. This is not a failure of the bridge recognition, and thus shall be considered separately. On the other hand, very long bridges are in general clearly visible in an image, thus they have to be detected by any method. So they are grouped in the class of ‘long bridges’ in this study. All remaining bridges that have a length in

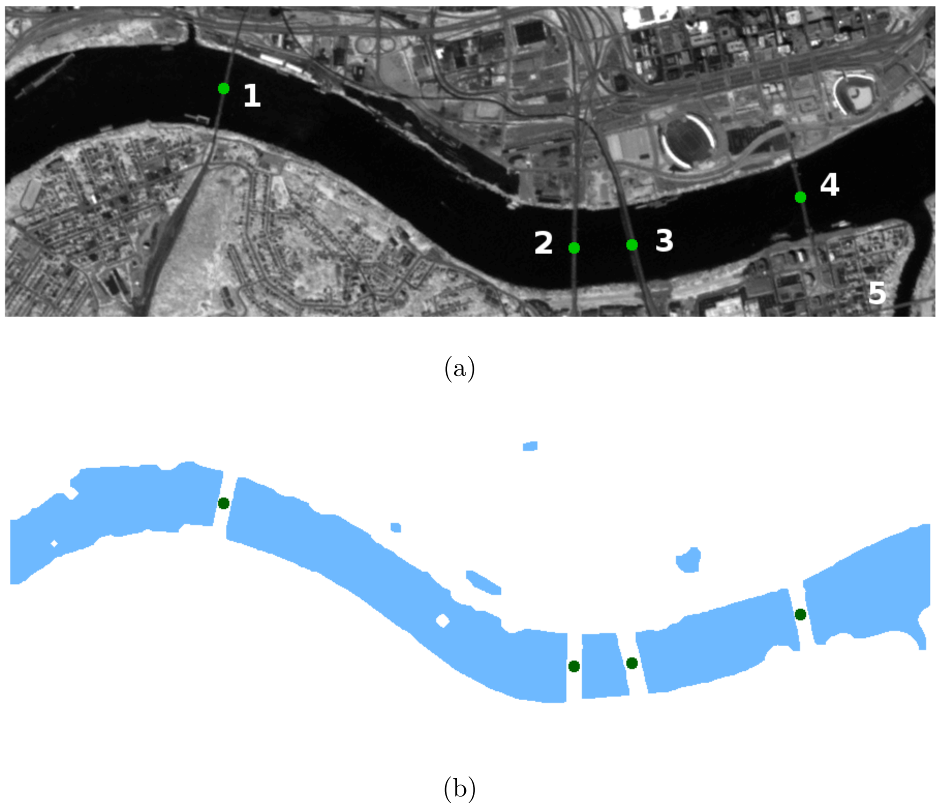


Figure 3.10: (a) an enhanced NIR scene of the city of Cincinnati (USA) combined with the result of the first bridge recognition. Located bridges are marked by a green dot. The five visible bridges are numbered. (b) The result of the first bridge detection (green dots) put on top of the corresponding water mask. Please note that the bridge center points are coloured in different shades of green in figure (a) and (b) (and on all further related images) for visualisation purposes only.

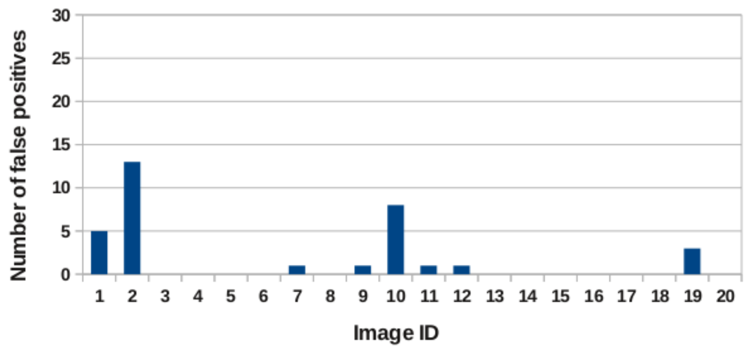
Data	Medium Bridges		Long Bridges		False Positives
	visible	found	visible	found	
HRC	24	11	42	33	33 (8 images)
RE	21	7	60	47	38 (4 images)
Total	45	18	102	80	71 (12 images)

Table 3.5: number of visible and recognised bridges from water masks **without** additional post-processing.

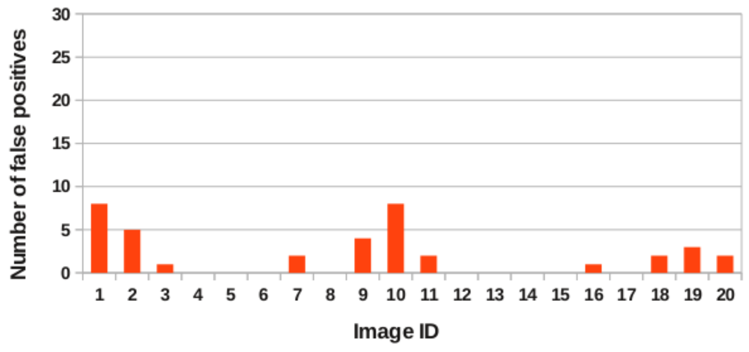
Data	Medium Bridges		Long Bridges		False Positives
	visible	found	visible	found	
HRC	24	13	42	32	38 (11 images)
RE	21	8	60	46	37 (6 images)
Total	45	21	102	78	75 (17 images)

Table 3.6: number of visible and recognised bridges from water masks **with** additional post-processing.

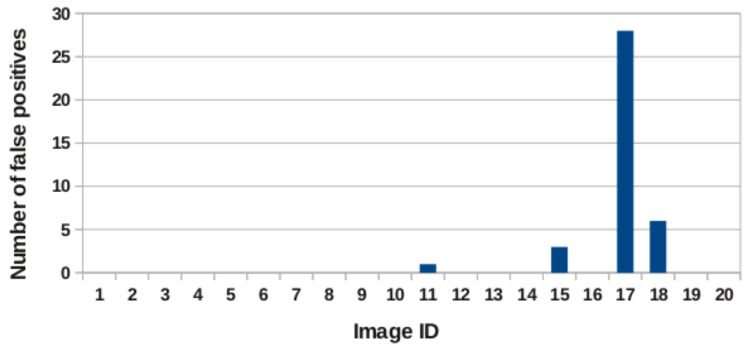
between these two classes are grouped into the class of ‘medium bridges’.
Such an in-depth analysis is carried out here for the first time. None of these considerations were made in literature, so a direct comparison between literature results an our results should be made with care.
The water mask upon which bridge recognition is performed has a significant impact on the detection result. In general, the more accurate the water mask, the higher the probability that a bridge is found and that false positives are suppressed. So, all water masks **without** and **with** additional post-processing are analysed.
At first, the results obtained from the water masks **without** extra post-processing are presented in table 3.5. For the reasons given above, the 31 ‘short bridges’ are not listed in the table. The detection rate of ‘medium bridges’ is about 40% and ‘long bridges’ were recognised with a rate of around 78%.
The results obtained with water masks **with** additional post-processing are displayed in table 3.6. Here, ‘medium bridges’ were detected with a rate of 46% and ‘long bridges’ with an average rate of 76%. So, a small influence of the different water masks can be observed. The effect is opposite for ‘medium’ and ‘long bridges’, so no general improvement can be determined. The underlying satellite image data seem to have no significant impact.
Yet, the large number of false positives is noticeable. The histogram illustration, displayed in figure 3.11, shall provide a better insight. It can be seen that the false positives occur in 12 (17) of 40 images, and in most cases there are less than 10 in one image. A considerable deviation appears in image 17 of the RE data set. In this image, a maximum of 28 false positives showed up.



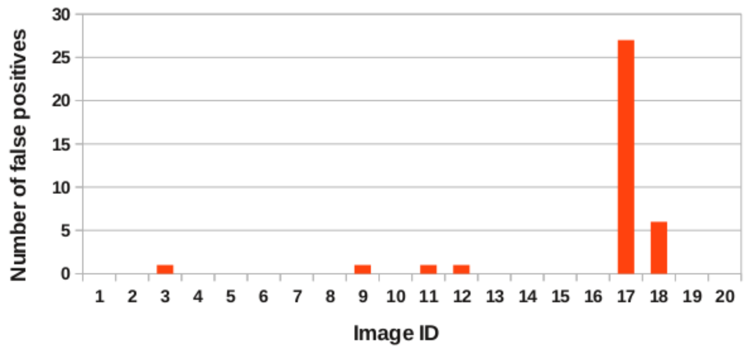
(a)



(b)



(c)



(d)

Figure 3.11: distribution of false positives in the first bridge recognition for each evaluated image. The HRC data set and a water mask **without** extra post-processing (a) and **with** extra post-processing (b). The RE data set and a water mask **without** extra post-processing (c) and **with** extra post-processing (d).

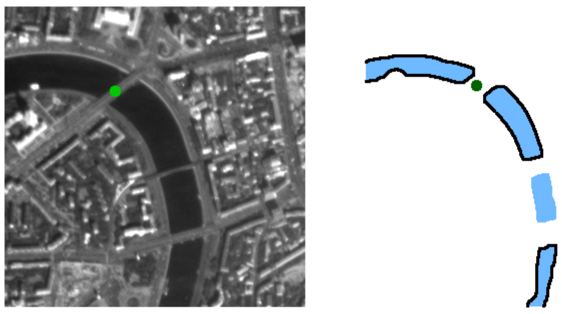


Figure 3.12: the first special case. (a) The PAN image combined with the result of the first bridge recognition. The detected bridge is marked by a green dot. (b) The result of the first bridge recognition (the green dot) put on top of the corresponding water mask. Large water segments are framed by a black rim.

But this image, as well as image number 18, is a scene from the tsunami disaster region in Japan that was already mentioned in section 3.2.5. There, a lot of additional water areas appeared and thus a lot of 'additional bridges' are found in between these water bodies. The image will be analysed separately in section 3.3.5. In the histogram, the slight influence of the different water masks is visible too. The additional post-processing, illustrated by the red histograms, reduces the number of false positives in some images, while at the same time their number increases in other images. A thorough discussion will be given in section 3.3.6. Four special cases that reflect individual situations at certain bridges are presented below.

Special case 1: small river segments

The first special case describes the situation that a river is broad enough to be represented in the final water mask, but a segment of the river is too small to be considered as a valid water segment. In figure 3.12 (a), a part of the city of Moscow (RUS) is displayed. For details of the data-take, see appendix A.1, HRC 3. Three bridges are visible and there is a strong contrast between water and non-water areas. The corresponding water mask is presented in figure 3.12 (b) and depicts the river course well. Nevertheless, only one bridge is marked by a green dot. An explanation can be given by the water mask. Three of the water segments are framed by a black rim, i.e. their size is above the threshold, thus they are considered in the bridge recognition process. The water segment without a black rim is too small, so it is 'invisible' for the remaining bridge detection steps. Hence, the two bridges on each side of this water segment cannot be detected. So, the size threshold is a problem, especially for 'medium bridges', but it is a useful filter to prevent false positives.

Special case 2: a bridge to an island

Another special case is a bridge leading to an island. In figure 3.13 (a), a part of the New York harbour area is given. It shows Ellis island, which is connected to the

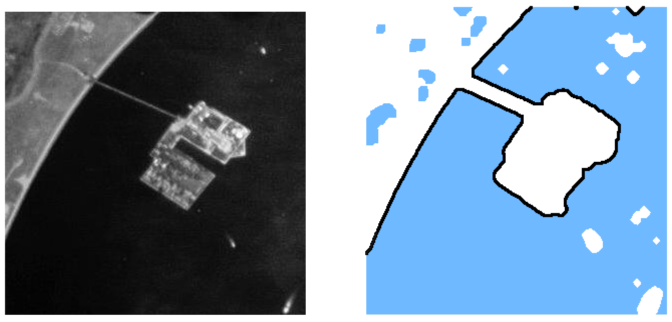


Figure 3.13: the second special case. (a) The original PAN image. (b) The related water mask. The large water segment is framed by a black rim. Since it is one big water segment, no bridge is detected.

mainland by a bridge. For details of the data-take, see appendix A.1, HRC 17. The bridge is not marked by a green dot and the reason lies in the labelling of the water rim. On both sides of the bridge, the water rim has the same label (number) since it is one continuous line. According to the definition, a bridge normally lies between two different water segments. Thus the processing stops at this point. Again, this is a precautionary measure to exclude false positives.

Special case 3: a curved bridge

Now and then it happens that a bridge has a curved shape, as depicted in figure 3.14 (a). Details of the data-take can be obtained from A.2, RE 13. Although the proposed method is in principle independent of the shape of a bridge, some simplifications within the algorithm prevent the detection of convoluted structures. In detail, it is the mentioned linear interpolation of the end points of each U-shaped segment on each side of a bridge. In figure 3.14 (b) the related water mask is depicted. The black lines illustrate the potential bridge segments. The support points on each side of the potential bridge segments are displayed as well (black dots). Due to the curvature, both points lie on one side of the bridge, hence there is no bridge in between these two points, so the bridge is not detected at all. To enable the detection of curved bridges as well, the *Second Bridge Recognition* method will be introduced in section 3.5.

Special case 4: a ship below a bridge

A rather exceptional case is the situation when a ship is cruising under a bridge right at the moment the satellite image is acquired. Nevertheless, it occurred a few times in the data set, hence it shall be noted here.
In figure 3.15 (a), an enhanced NIR image of the city of Cologne (GER) is displayed. For details on the data-take, see appendix A.2, RE 4. Both visible bridges were found by the algorithm, and are denoted by green dots. The upper bridge is marked by one green dot, as expected, whereas the lower bridge is marked by three green dots. For

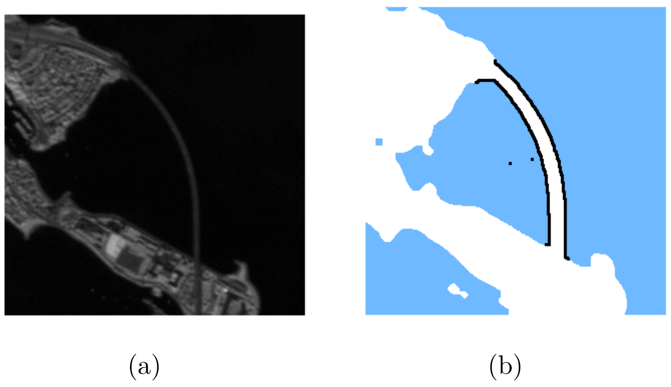


Figure 3.14: the third special case. (a) The enhanced NIR image. (b) The corresponding water mask. The black lines outline the potential bridge region. Due to the curvature, both support points (black dots) lie on one side of the bridge. Thus the bridge is not detected.

a better visualisation, the RGB composite image of that scene is depicted in figure 3.15 (b). From that, it can be seen that there is a ship below the lower bridge that is barely visible in the RGB image and almost invisible in the NIR image. Nevertheless, the corresponding water mask, displayed in figure 3.15 (c), represents the ship as an unusual cavity on the lower side of this bridge. The support points on each side of the bridge are depicted as well. Since there are three points now within a certain vicinity, the final center point of each pair of points is composed. And because all three final center points are non-water pixels, they are accepted in the recognition process as valid bridge center points. Thus this bridge is represented by three center points. The outcome is correct, and reflects that there is an intact bridge with an unusual bridge shape. A discussion will follow in section 3.3.6.

Comment on broken bridges

The aim of the present method is to provide information on accessible bridges. Broken bridges require special treatment and probably a-priori knowledge on their existence. Nevertheless, a comment shall be given on them, too.
As already mentioned in the description of the algorithm in section 3.3.3, a clearly broken bridge would result in a combination of adjacent water segments, thus no bridge can be recognised. The indirect information, that there is no accessible bridge, is true. However, it might be the case that the broken part of the bridge is only very small, so that the ‘river’ between the bridge sections is very thin. In this case, the erosion step in the water mask generation would simply erase this small ‘river’ and ‘repair’ the shape of the bridge, more or less. This is visualised in the simulated water masks in figure 3.16 (a) - (c). The normal case is depicted in figure (a), where the bridge is clearly indicated by the gap between the two water segments. Then, a simulated broken bridge is displayed in figure 3.16 (b). After the opening of the water segments, the water mask looks as in figure 3.16 (c). In this case, the bridge would be identified by this method, even though it is actually broken. Satellite image

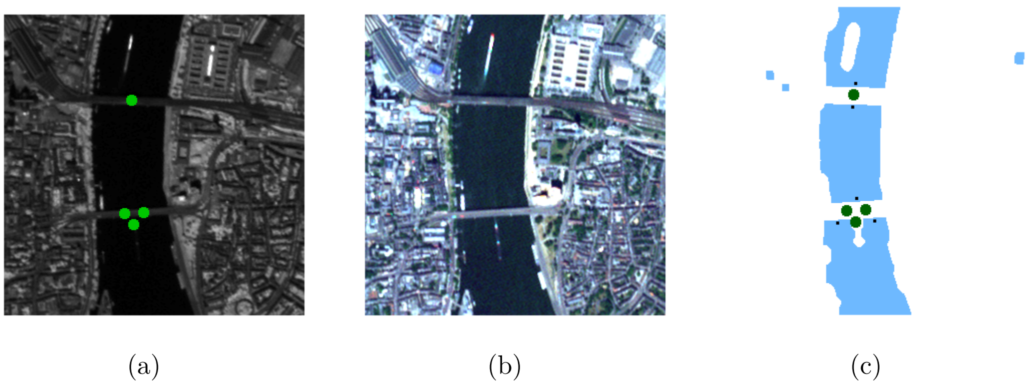


Figure 3.15: the fourth special case. (a) The enhanced NIR image combined with the result of the first bridge detection (green dots). (b) The respective RGB composite image of the scene. (c) The associated water mask combined with the results (green dots). To explain the three green dots on the lower bridge, the support points (black dots) are depicted as well.

data containing such a situation were not available for the current study, therefore this case could not be evaluated properly.
Again, it should be pointed out that the intention of this approach is a fast and brief survey of an acquired scene. For a more detailed analysis, other methods, e.g. the *Second Bridge Recognition* method, should be considered.

Case study of a natural hazard: the 2011 tsunami in Japan

To evaluate the performance of the algorithm on a real disaster scenario, satellite image data from the 2011 tsunami in Japan were evaluated. In figure 3.17 (a), the same part of Japan's coastline is shown as in the water body extraction section. This time it is combined with the result of the first bridge recognition. Located bridges are marked by green dots. As can be seen, a lot of markers appear over the entire image, though most of them are false positives. The large number of additional water areas that are not necessarily false classifications, since this is a region after flooding, arouse these false positives. The water mask in figure 3.17 (b) illustrates that in the spaces between these water bodies, numerous regions are categorised as bridges.
The two 'long bridges' (no. 1 and 2), in the center of the image, are recognised by the proposed method. The two 'short bridges' (no. 3 and 4) are not detected for the given reasons. And the 'medium bridge' (no. 5) remains undetected, because the water segment on the lower side is too small. A discussion follows in section 3.3.6.

Hardware resources

The hardware use of the first bridge recognition method is comparable with that of the water body extraction. Around 31% of the available Slices on the Spartan-3E FPGA are utilised. Additionally, three BlockRAM cells are in use. In the current experimental set-up, the algorithm runs at $12.5MHz$, thus processing one 1024×1024

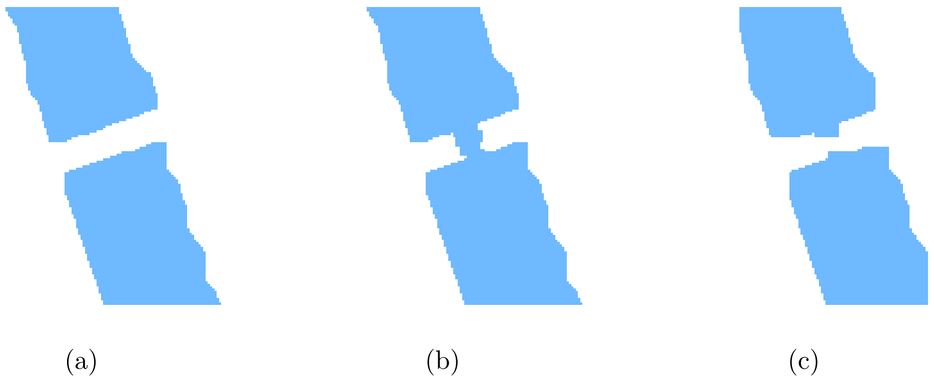


Figure 3.16: a simulated broken bridge scenario. (a) A water mask under normal circumstances. (b) A water mask containing a broken bridge. (c) The appearance of the broken bridge after the erosion and dilation step.

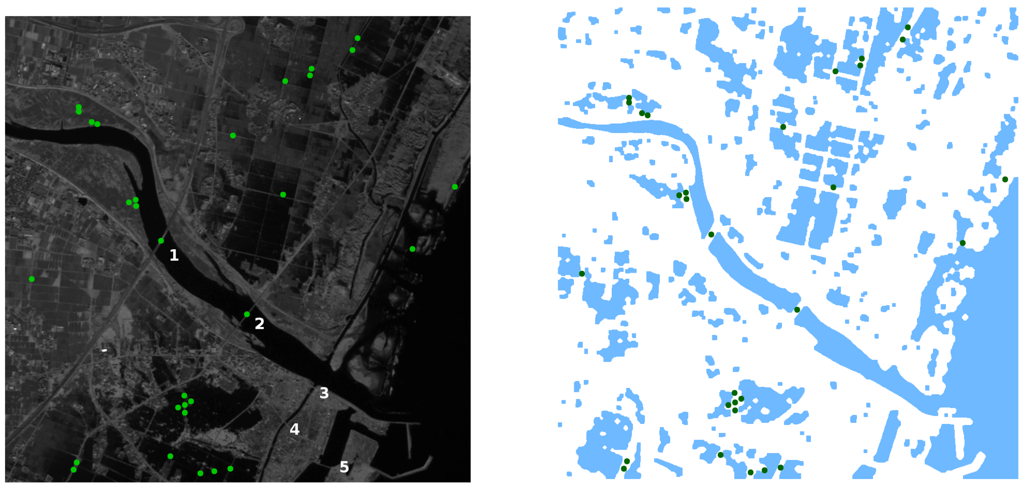
Slices	2724 (31%)
BlockRAM	3 (10%)
Max. Clock Frequency	$77MHz$
Power	$0.8W$

Table 3.7: *Nexys-2* implementation parameters

pixel image takes approx. $2s$. The maximum clock frequency at which the algorithm still operates, is estimated by the ISE design suite to be $77MHz$. This is a six times higher processing speed. With this clock rate, results can be obtained in less than a second. The power consumption is again below $1W$. The details of the Nexys-2 implementation are listed in table 3.7. A discussion regarding the results as well as the implementation is given below.

3.3.6 Discussion and Summary

In this section of the thesis, a novel method is presented that automatically recognises bridges over water in satellite images. The main difference of this approach to already existing ones is the optimised algorithm structure that enables implementation on a small FPGA.
The preliminary software program was written in IDL to assess the performance of the proposed method. This program was modified and optimised in the hardware version to meet the constraints of the used FPGA. Mathematically sophisticated operations and complex loop procedures were omitted or substituted by low level operations. In the end, the outcome of both implementations is comparable, but the algorithm structure differs in several parts. In conclusion, all further algorithms that will be presented in the remaining sections are written directly in Handel-C.
A binary water mask is the sole basis for the bridge recognition method and, in princi-



(a) (b)

Figure 3.17: (a) the enhanced NIR image combined with the result of the first bridge recognition (green dots). The five visible bridges are numbered. (b) The corresponding water mask combined with the result of the first bridge recognition.

ple, the more accurate the water mask, the better the results of the bridge detection. In the current water masks, this implies that it is impossible to find 'short bridges' since the associated rivers are erased in the water mask generation procedure. A possibility to overcome this drawback could be the use of satellite imagery with a higher spatial resolution e.g. as the IKONOS imagery used in Luo et al. (2007). There, small rivers are represented by a larger number of pixels, and the assumption is that the erosion procedure may be adjusted more precisely to preserve the true water bodies while still excluding false positives.
The statistical evaluation revealed a recognition rate for 'long bridges' between 76% and 78%. Compared to statistical assessments in literature with detection rates of 80.95% in Zhang and Sun (2011) or 89.61% in Chen (2008) this seems a quite good result, especially keeping in mind the less complex structure of the algorithm and the possibility to implement it on an FPGA. The lower detection rate of 'medium bridges' might be related to the fact that the employed water masks are not very precise at the water area boundaries. A misclassification in these regions, however, results in an inexact representation of the bridges in the water masks. In combination with the size filter for bridge segments, this effect is stronger for shorter bridges than for longer ones, hence it is more difficult to recognise 'medium bridges' properly. Again, it shall be noticed that the classification of bridges relating to their length is performed here for the first time, so that comparison with literature data must be done with care (see section 3.3.5).
The additional post-processing step in the water mask generation was implemented to enhance the accuracy of the water masks. However, the statistical evaluation of the first bridge recognition results did not reveal any significant increase in the detection rate when using these water masks. Regarding the clearly longer processing

times in the water mask generation step, it does not seem advantageous to perform additional post-processing.
Four special cases were considered separately. They show situations that happen now and then in real satellite imagery and characterise the performance of the proposed algorithm in more detail. Some cases, e.g. the curved bridge, provide arguments for the development of the *Second Bridge Recognition* method, because it would be nice to detect these bridges as well. Other situations, like the ship under the bridge, show that the method is capable of detecting even small discrepancies from the normal state.
The results of the bridge recognition obtained from the tsunami disaster satellite image show a lot of markers and, at first glance, most of them seem to be false positives. However, besides the information that there are two (intact) bridges that span the river, the thematic map also provides a first useful survey on the infrastructure in the whole area. As already mentioned, it might be the case that these additional water areas are indeed water areas at that time, and the location of 'bridges' between these water zones provides information on potential transportation routes through this affected area. In combination with a road map, this might be valuable information.
In conclusion, processing one 1024×1024 pixel image in around $2s$, and a minimum detection rate of 76%, make the method well acceptable for the current purpose.

Summary

The first bridge recognition method requires a binary water mask as sole input data and is implemented on a small FPGA. No a-priori knowledge on the location of a bridge is needed, so even shortly constructed pontoon bridges can be detected. All in all, the method provides a fast survey of bridges over water in a captured scene. The robust implementation, as well as the fast and autonomous image processing, make this approach a reasonable tool for implementation on future spacecraft.

3.4 Island Detection

3.4.1 Purpose

The idea of image evaluation on-board satellites is especially reasonable and useful in situations were rapid changes occur on the Earth's surface and large areas are affected. As already mentioned, one such event is flooding, and prompt information on the extent of the disaster and the situation of buildings and infrastructure is desired. Additional relevant information pose potential places where refugees might wait for rescue, e.g. on rooftops of houses.
In this thesis, the term *island* refers to all non-water areas that are completely surrounded by water. In case of flooding, this description also holds for rooftops. So, in this section of the thesis, a method will be presented that extracts all islands in a captured satellite image and provides the desired information.
Some background knowledge, the algorithm and its hardware implementation are presented in the following two sections. Then, the results are visualised and a conclusion is given.

3.4.2 Background

The use of the term 'island' originates from the fact that remote sensing images of the Earth's surface are the primarily input data in this study, thus an area completely surrounded by water is called an 'island'. However, it means that the nature of an 'island' is not limited to natural islands like rocks or sandy beaches, but also implies artificial islands, such as rooftops or ships. Furthermore, a natural island that is connected to the mainland via a bridge will not be regarded as an island here. The reason is that the proposed method takes a binary water mask as input and from that such an island is just an extended part of the mainland (see e.g. Ellis Island in figure 3.13).
So, in a binary water mask, islands are holes or spots in the water segments. The filling of holes or spots in a binary image is a well-known task in image processing. Yet, in contrast to the standard procedure to erase these holes, here they are particularly labelled.
The principle work flow of the algorithm is depicted in figure 3.18. Besides the original binary water mask, a second auxiliary memory space is allocated that has the same dimension as the water mask. A binary segmentation is performed on that auxiliary image, i.e. all non-water pixels that are connected to the image boundary are assigned the value 1, and the remaining pixels are labelled with the value 0. After the entire image is processed, both images (the original and the auxiliary) are compared. All non-water pixels in the water mask assigned with the value 0 in the auxiliary image are labelled as islands. Using this approach, all islands in a water mask can be found quickly.

3.4.3 Hardware Implementation

As mentioned above, the detection of spots in a binary image has been addressed in literature before. An implementation of this technique in an FPGA is presented in

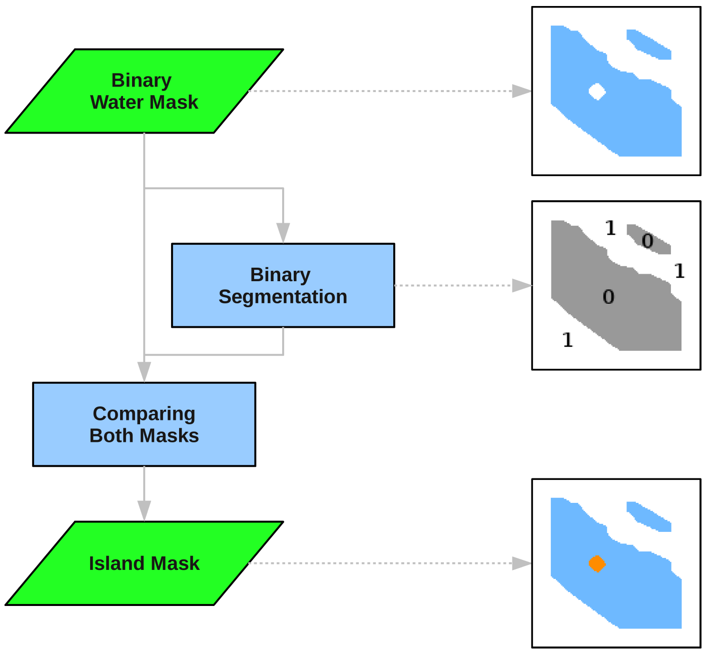


Figure 3.18: flow chart of the island detection algorithm.

Pedrino et al. (2010). The approach used in this thesis is similar, although there are some differences in the hardware realisation.
Both methods require the original binary image and a secondary image of the same dimension. Then, the connection of all background pixels (here: non-water pixels) to the image boundary is analysed. Background pixels with no connection are labelled as islands. During the initialisation of the process, the literature approach marks the entire image rim of the auxiliary image, whereas the present approach only marks the first pixel therein as a starting point. This is a valid procedure, since every background pixel connected to the image boundary is automatically connected to the 'first pixel' due to the white (background) rim around each satellite image. The whole image is then scanned according to the eight different scan-line paths depicted in figure 3.19. A description of the routine can be given as follows:

1. Initialise the whole auxiliary image with the value 0.
2. Set the first pixel in the auxiliary image to 1.
3. Choose a scan-line path (choose the starting point and the direction)
4. Pick every pixel that is labelled 'background' in the original image and that has the value 0 in the auxiliary image.
5. Take the two previous pixels of the current pixel (the precursor in the same line and the pixel in the line above/below) in the auxiliary image and check if at least one of them has the value 1.

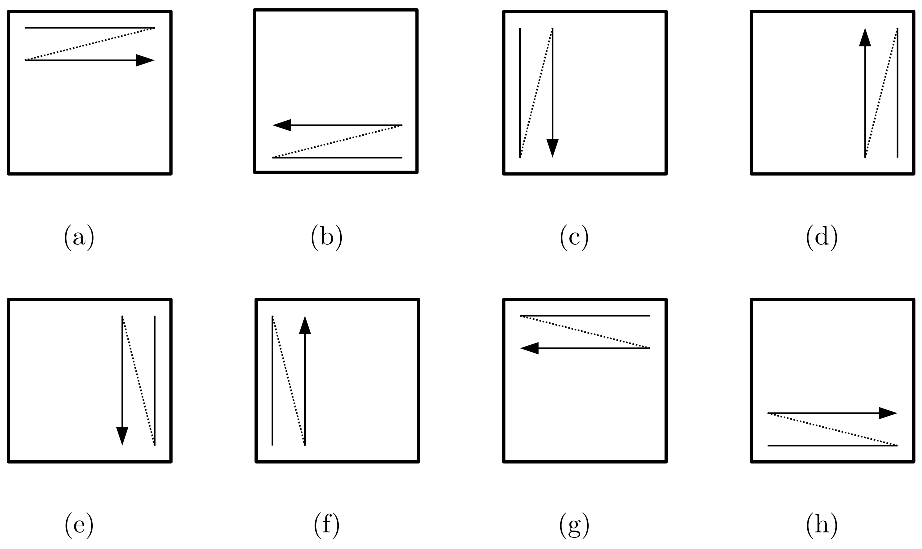


Figure 3.19: illustration of the eight scan-line paths.

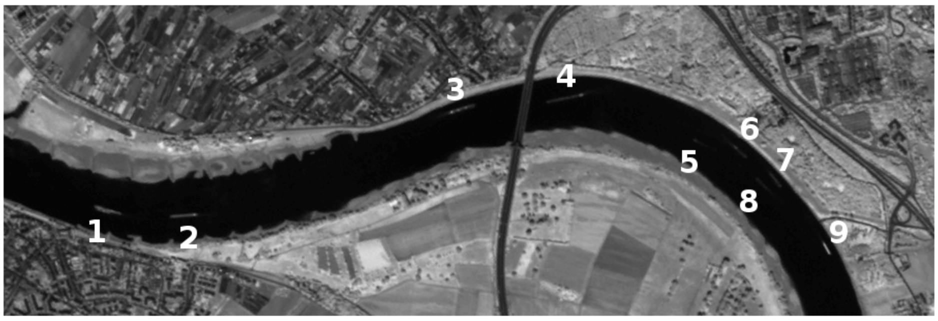
6. If the test is positive, then assign the value 1 also to the current pixel in the auxiliary image. If not, the pixel value remains 0.
7. When all image pixels are processed, go back to point 2, choose the next scan-line path and repeat the procedure until all eight scan-line paths are processed.

In applying this technique, every background pixel connected to the image boundary is excluded, and only the island pixels remain particularly labelled. A result is presented in the section below.

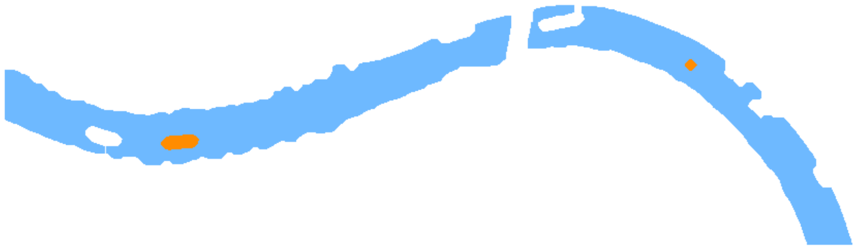
3.4.4 Results

Example image

In this section, a new example image is chosen to represent the results of this method. In figure 3.20 (a), a part of the river Rhine near the city of Düsseldorf (GER) is displayed. For details on the data-take, see appendix A.3, RE 26. The image shows a bridge that crosses the river in the center of the scene, and some ships cruising the river. Again, this NIR image was enhanced in brightness and contrast for visualisation purposes only.
The result of the 'island detection' is shown in figure 3.20 (b). Water areas are blue and located islands are orange. It can be seen that two ships (no. 2 and 5) are labelled as islands. According to the current definition of an island, the result is correct. A further two ships (no. 1 and 4) are represented in the water mask as well, but they are not labelled as islands since they have a (small) connection to the background (white region). And five ships (no. 3 and 6-9) are very close to the river boundaries, so they only appear as ragged shapes on the riversides.



(a)



(b)

Figure 3.20: example image for island detection. (a) An enhanced NIR image of a scene near the city of Düsseldorf (GER) ©DLR/RapidEye. The ships are numbered. (b) The derived thematic map. Water areas are blue, islands are orange.

Overall, in the used data set, the complete labelling of all islands was achieved with the proposed technique. A statistical evaluation is omitted since no false classifications occurred.

Hardware characteristics

The 'island detection' is the first algorithm that was directly developed and implemented in Handel-C. The realisation was straightforward, as described in section 3.4.3. The required hardware and some parameters describing the FPGA implementation are listed in table 3.8. The used logic adds up to 13% of the Spartan-3E's total capacity, and no BlockRAM is needed. The ISE design suite determines the maximum clock frequency for this algorithm to be $87MHz$. In the current set-up, with a clock rate of $12.5MHz$, processing one 1024×1024 pixel image takes around $4s$. The power consumption is again well below $1W$.

Slices	1191 (13%)
BlockRAM	0
Max. Clock Frequency	$87MHz$
Power	$0.2W$

Table 3.8: *Nexys-2* implementation parameters

3.4.5 Conclusion

The 'island detection' presented in this section of the thesis is capable of very quickly finding all islands in a binary water mask. The method is again implemented in a small FPGA. The information on islands in a certain area can provide knowledge on potential human whereabouts in case of flooding. In combination with a detailed analysis of the islands, it is also possible to locate ships or offshore wind parks. Due to the chess pattern-like appearance of offshore wind farms, they can be recognised and might provide an additional orientation point in an image. Finally, it should be mentioned that the particular labelling of islands is advantageous for the *Second Bridge Recognition* method, since it increases the detection rate significantly. All in all, the 'island detection' is a useful tool for future on-board image evaluation capabilities.

3.5 Second Bridge Recognition Approach

3.5.1 Preliminaries

The first bridge recognition method presented in this thesis provides information on the location of bridges over water in a captured satellite image. The extracted positions, the center points of detected bridges, are illustrated as large green dots in the corresponding thematic map. A single point, however, does not contain any information on the structure of a bridge, nor does it enable a detailed analysis of the site to be performed. With regard to the aforementioned situation of a broken bridge, such an in-depth examination would be helpful to verify the accessibility of a bridge. In addition, the recognition of curvilinear bridges is a desired ability for a new technique too. This is the reason why a second bridge recognition method was developed that labels the whole bridge deck area.
The second approach differs significantly from the first bridge recognition method, yet it is also designed for an implementation on hardware. Since the results look similar to the outcomes of already existing methods described in literature, a short review will be given in the next section. Then, the new algorithm is presented, as well as its implementation on an FPGA. Afterwards, the findings are visualised. A discussion, a comparison with the first method and a summary, conclude this section.

3.5.2 Survey of Articles

The conducted literature survey revealed three articles that are relevant for the second bridge recognition method in this thesis, since all methods have a very similar outcome: a labelled bridge deck area. In the following, each of the literature approaches is presented.
The article of Gu et al. (2011) starts with an extraction of a binary water mask from aerial images. After all major water bodies are found, adjacent water segments separated by a bridge are aligned again. Potential bridge regions are recognised by a comparison of the original water mask and the new one. Finally, a verification step is performed on the bridge area. This verification is performed under the assumption that a bridge is a '*narrow and rectangular structure with big ratio of length to width*'. A rectangular shape, however, prohibits the detection of curved bridges. Yet, this is a desired qualification of the second bridge detection method. Furthermore, a big ratio of length to width implies that it is very likely to exclude smaller bridges from detection, since they only have a small length to width ratio. So, this approach does not seem an improvement, hence it is not applied in the study of this thesis.
In Abraham and Sasikumar (2012), a water mask is derived from multi-spectral satellite images using a fuzzy threshold segmentation. A water area size filter is then applied to exclude small false positive water regions, so that only the river course remains in an image. In the resulting water mask, a bridge region is found by the assumption that there is a gap between two water segments no greater than 10% of the image width. The principle idea of closing a gap between two adjacent water segments is also used in the current approach, though redefined.
A further technique to label the deck area of a bridge is presented in Gedik et al. (2012). The authors use very high-resolution panchromatic and multi-spectral satel-

lite imagery from IKONOS and GEOEYE to automatically locate bridges over water. At first, all water bodies are extracted and examined whether they are part of a river or a canal. The associated segments are assumed to be thin, long and continuous structures. Then, distinctive points at the edges of each water segment are located. The four points framing a bridge are connected by straight lines and the inner area is labelled, thus the bridge deck is found. To prevent false positives, two verification strategies were presented. The first one uses the orientation of each detected bridge segment. If the orientation of a potential bridge segment is higher than 70° with regard to the adjacent water segments, it is assumed to be a real bridge, otherwise it is discarded. The second method uses an automatically derived road map (see Karaman et al. (2012)). Only potential bridge segments that are also marked in the road mask are verified bridge segments. The limitation to thin rivers, however, would exclude the detection of many bridges, e.g. the Golden Gate Bridge in San Francisco, since this bridge is between two large water regions. Yet, the strategy of a stepwise detection and verification process seems reasonable and is also applied in the study of this thesis.
In conclusion, the methods presented in literature have certain shortcomings, so they cannot be fully adopted for the given purpose. However, some practical ideas will be used here as well. The new approach is presented in the section below.

3.5.3 An Advanced Method

The second bridge recognition method is based upon a thematic map where all major water bodies and islands are depicted. Thus, the method is, like the first one, independent of the original satellite image data. The algorithm consists of two parts: firstly, potential bridge deck areas are located and, secondly, true bridge areas are verified. The final result, the verified bridge deck areas, are drawn as green segments on a thematic map.

Potential bridge deck areas

The location of potential bridge deck areas is achieved in two steps through applying morphological operations on the thematic map. In principle, it is a *closing* of the water areas. An illustration is given in figures 3.21 (a)-(c). In figure 3.21 (a), a small section of a thematic map is displayed. The white passage between the two blue water segments represents a bridge, while the concavity in the right water segment is a ship on the river that was unfortunately connected to the main land in the erosion step of the water mask generation. The procedure is as follows:

1. The **extension** of water areas. To achieve this, the water rim of every water body is labelled and the size of each water rim segment is determined. Small segments are again excluded from further processing. Then, the water rim segments of the remaining water segments are extended by a certain number of pixels, so that all water areas are enlarged. The area between the original water rim and the new water rim is particularly labelled (in figure 3.21 the pixels are coloured red). As can be seen from figure 3.21 (b), the bridge area as well as

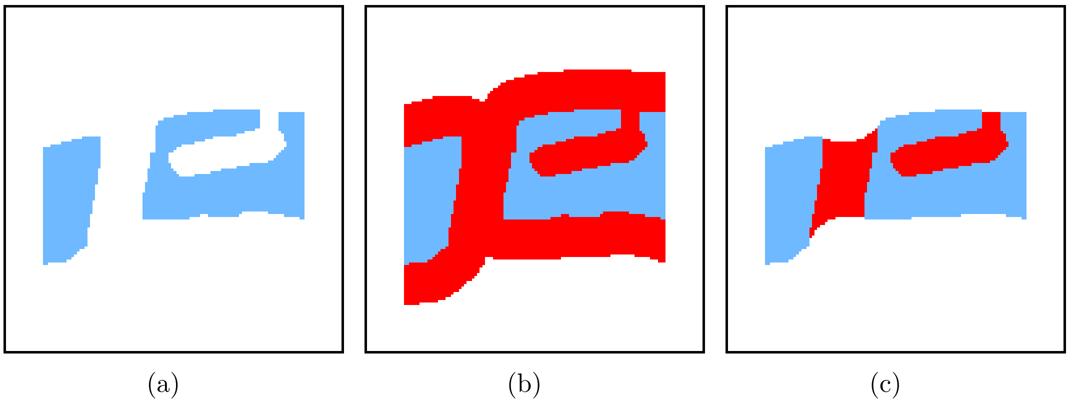


Figure 3.21: main steps of the potential bridge deck area detection. (a) A section of a water mask. (b) The water mask with extended water areas coloured red. (c) The resulting thematic map with extracted potential bridge deck areas coloured red.

the concavity are completely filled by red pixels. If islands were not excluded beforehand, they would also be filled with red pixels to the extent of the new water rim, i.e. islands with a diameter of less than twice the 'extent' would be completely filled, whereas larger islands would still have a non-water area in their center.

2. The **reduction** of water areas. At first, the outer rim of each enlarged water body is labelled. Then, the water rims are reduced by the same number of pixels as in the extension step, and the red pixels in between are erased. In an ideal case, e.g. a spherical water area, the outcome of these two steps applied to the water area would be the original water area again. In a situation like the one depicted in figure 3.21 (b), however, several pixels on the bridge and in the concavity have a larger distance to the outer rim than the 'extension' was, so they remain red. The result is illustrated in figure 3.21 (c). Larger islands would also have a new water rim at the boundary to the inner non-water pixels, and all red pixels in the vicinity of this water rim would be erased. Now and then, a larger island is near a bridge and some red pixels on the bridge would be erased too. This is a drawback to bridge recognition. To avoid this, islands are marked prior to the second bridge detection and excluded from the morphological operations.

In the example image in figure 3.21 (c), the whole bridge deck area is labelled, but also the concavity that is obviously not a bridge. To distinguish between a real bridge and a false one, an additional processing step is performed afterwards.

Bridge verification

The previous processing step is performed 'blindly' on the whole thematic map. Besides real bridges, a lot of unwanted areas are declared as potential bridge areas. So, the next part of the algorithm should drastically reduce the number of false bridge areas, although it will increase the processing time.

The procedure is similar to the proposed technique for 'island detection' in section 3.4. Unfortunately, one pass has to be performed **before** and one **after** the bridge deck area detection, so it is not possible to combine them and reduce the number of passes. The procedure is illustrated by the three images in figures 3.22 (a)-(c) and the algorithm works as follows.

1. An auxiliary memory space is allocated that has the same dimension as the thematic map. It is completely blank except for some black lines that represent all (blue) water pixels in image 3.21 (c) that have a direct contact to at least one red pixel. These black lines are slightly enlarged in figure 3.22 (a) for visualisation purposes only. Initially, all 'blank' pixels are labelled with the number zero.

2. Each of the four corner pixels of an image is a white background pixel due to the white rim around the satellite image. Beginning in the upper left corner of the image, the label of the first pixel is raised by one. Then, the labelling goes line by line from the top to the bottom of the image, indicated by the yellow arrow in figure 3.22 (b). All white pixels receive the higher label number if they have the higher label number in their direct 4-neighbourhood (top, bottom, left, right).

3. The process is repeated in the backward direction, from the bottom to the top of the image. The label number of the first pixel (in the lower right corner) is raised by one and all other white pixels receive the higher label number if they fulfil the neighbourhood requirement.

4. In the process, the black lines act as barriers, so coming from below, the backward labelling process cannot reach the pixels inside the concavity, thus these pixels remain with the lower label number. Here, this is indicated by the yellow pixels in figure 3.22 (c).

5. In order to analyse bridges and concavities with different orientations in an image, the image is in principle rotated 3 times by 90 degree and scanned from the left to the right and vice versa after each rotation. For sound hardware implementation, the rotation and the screening of the image is 'realised' by changing the starting point and direction of the scan-line path according to figure 3.19.

6. After the 2nd and the 4th forward and backward screening of the image (in this case the highest label number is 4 or 8 respectively), the auxiliary image is compared with the (intermediate) thematic map in figure 3.21 (c). All potential bridge pixels with the label number 4 or 8 in the auxiliary image are verified bridge deck areas and labelled green in the resulting thematic map (see figure 3.22 (d)). False bridge areas remain red.

The sequence of the scan-line paths and the additional check after the 2nd forward and backward screening over the entire image is customised, so that also bridges that are diagonal in an image can be detected and verified as true bridges. In applying this technique, the majority of false bridge deck areas, that often appear as cavities at the water's edge, can be distinguished from real bridges.

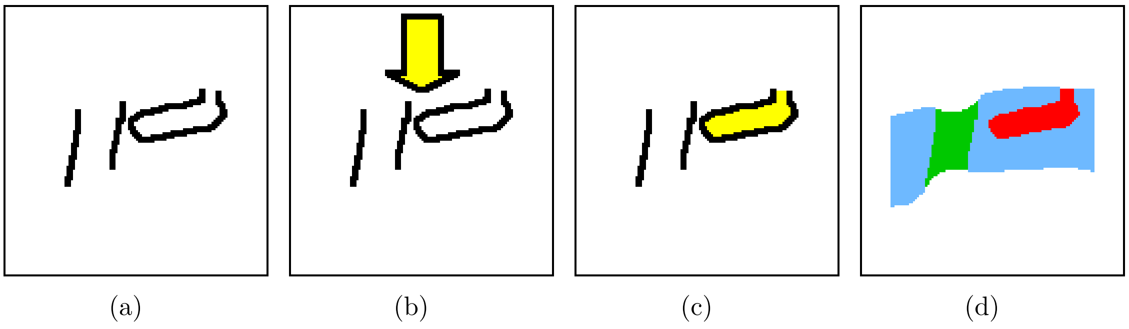


Figure 3.22: illustration of the bridge verification process. (a) A secondary memory space with delimiting black lines. (b) Visualisation of the 'yellow' labelling of the image. (c) The outcome of the procedure with remaining yellow pixels in the cavity. (d) The final result of the whole bridge detection algorithm. The verified bridge is labelled green, whereas the false bridge area stays red.

3.5.4 Remark on the Hardware Realisation

The second bridge recognition method was realised in an FPGA according to the description in the section above. In the experiments, however, it turned out that the choice of the template for the morphological operations has a significant influence on the final results. Thus, two common types of templates will be discussed below.
The maximum width of any bridge in the world was already discussed in section 3.3.4 and is always smaller than $100m$. Taking the given spatial resolution of $5m$, this corresponds to a maximum width of 20 pixels. This gap between two adjacent water segments has to be influenced by the morphological operations. So each water segment has to be enlarged by 10 pixels, and then reduced by the same number, to obtain the desired result. However, it turned out in the conducted experiments that an extension of 16 pixels yields more stable results. The template for the closing procedure can be chosen arbitrarily. Two common shapes, a *quadratic structuring element* and a *circular structuring element*, are examined in the following text, since their form has an impact on the recognition result.
At the beginning of the experiments, a *quadratic structuring element* of 32 x 32 pixels was used. So each currently regarded water rim pixel could influence at least 16 pixels in each direction. In figure 3.23 (a) a section of a thematic map is depicted. Two water rim pixels are marked by a cross and their respective template area is illustrated by the dotted squares. The black lines represent, albeit not absolutely, the final outer water rim after the water body extension. When it comes to the water body reduction step, each pixel of the outer rim is analysed. Two of these pixels are denoted by a cross in figure 3.23 (b), and their corresponding template area is again illustrated by a dotted square. Since the river is slightly diagonal in the image, the templates on both sides of the bridge have a different impact on the number of erased pixels on the bridge. So the final outcome, presented in figure 3.23 (c), shows an irregular recognition of the bridge deck area. This effect grows with an increasing skewness of the river.

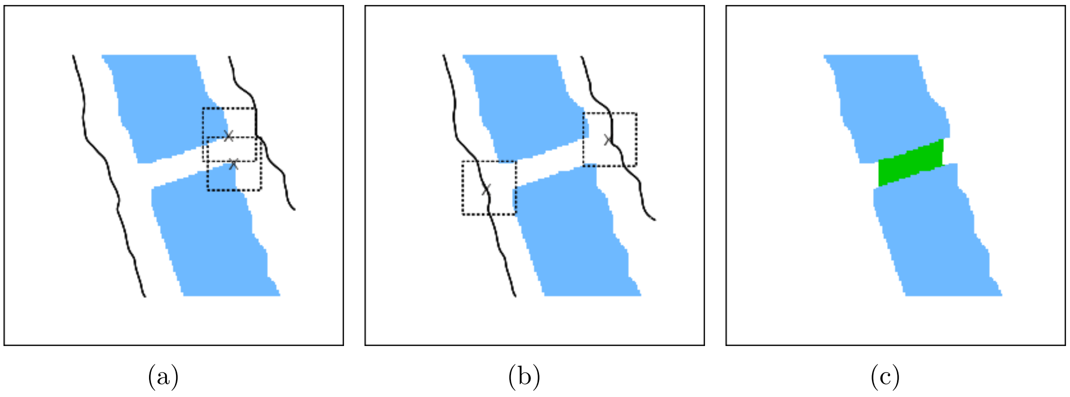


Figure 3.23: illustration of a *quadratic structuring element* for the morphological operations. (a) The dilation of the water areas. (b) The erosion of the water areas. (c) The final thematic map with an uneven bridge deck area coloured green.

To overcome this disadvantage, a symmetric *circular structuring element* was chosen with a radius of 16 pixels, disregarding some discretisation inaccuracies. The shape was 'stored' in a `static ROM` array composed of entries with 0s and 1s, resulting in a large circle of 1s. As before, a *quadratic structuring element* of pixels around each currently regarded pixel was chosen. This time, additional matching with the *circular structuring element* was performed, in order to ensure that the currently selected (template) pixel is within the circle. In figures 3.24 (a)-(c), the same steps are shown as in figures 3.23 (a)-(c), this time with a *circular structuring element*. The final result, depicted in figure 3.24 (c), is a consistent recognition of the bridge deck area on both sides of the bridge. But, please note, as can be seen from figure 3.24 (b), the *circular structuring element* reaches several pixels on the bridge area and erases the pixels there. This happens due to the fact that at the bridge gap, the templates from both sides of the bridge do not overlap completely at the outer rim. So, the extended water rim at the bridge gap has a small dent and is less than 16 pixels away from the bridge head. Thus, in the reduction process, the template is closer to the water areas at the gap and removes several bridge pixels: this is an issue. Yet, especially for 'medium bridges', the uniform detection has a positive influence on their recognition rate, so the *circular structuring element* was finally used in the experiments.

3.5.5 Experimental Results

Example image

At the beginning of this section, an example image shall illustrate the principle outcome of the second bridge recognition method. The same image as in section 3.4 is chosen and depicted in figure 3.25 (a). It is a NIR image of an area near the city of Düsseldorf (GER). Brightness and contrast of the image were enhanced for visualisation purposes only. Additionally, the result of the second bridge detection method is superimposed on the image. The identified bridge deck area is illustrated by the green segment. The coloured segment is broader than the actual visible bridge in the original image, since the bridge region appears broader in the water mask (see fig.

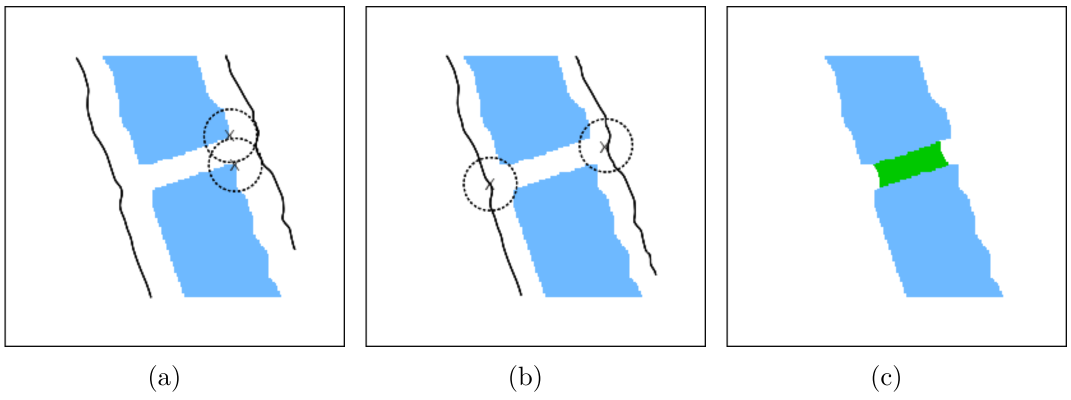
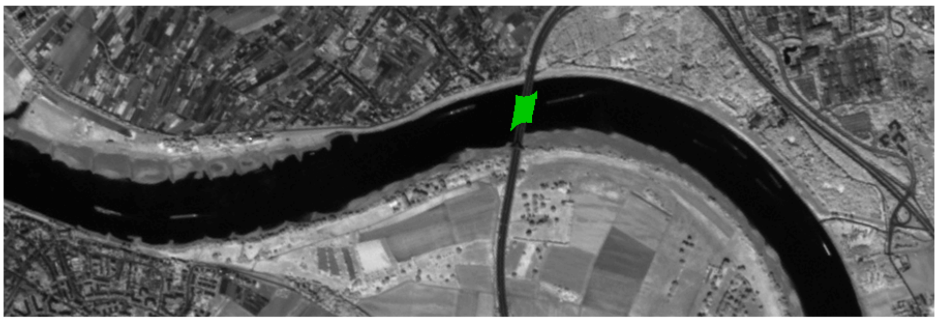


Figure 3.24: illustration of a *circular structuring element* for the morphological operations. (a) The dilation of the water areas. (b) The erosion of the water areas. (c) The final thematic map with a symmetric bridge deck area coloured green.

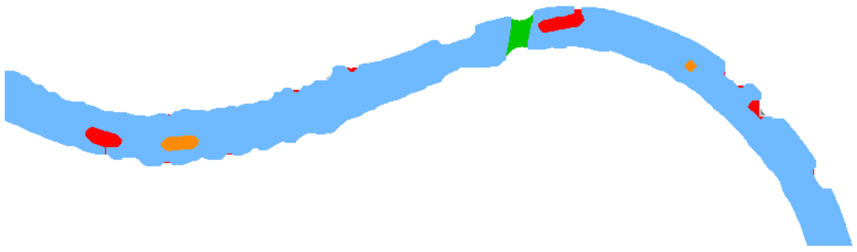
3.25 (b)). Furthermore, the bridge heads have a clearly visible rounded shape, a result of the chosen *circular structuring element*. From the corresponding thematic map, it can be seen that several false bridge deck areas (red segments) were suppressed using the proposed technique.
In conclusion, recognised bridges appear broader and shorter than they are in the original satellite image, yet the labelling of the whole bridge deck area works sufficiently and enables a detailed analysis of the site.

Statistical assessment

The statistical evaluation of the applied data set was carried out in the same way as for the first bridge recognition, i.e. all bridges were classified into three groups relating to their length. So there are 'short bridges' that have a length of 13 pixels or less, 'medium bridges' with a length of more than 13, and less or equal to 40 pixels, and there are 'long bridges' that have a length of more than 40 pixels. Furthermore, all water masks **without** and **with** additional post-processing were analysed, since the different water masks might have now an impact on the detection result.
The results obtained from the water masks **without** extra post-processing are displayed in table 3.9. 'Short bridges' are again not listed in the table, since they cannot be detected at all. The reasons were given in section 3.3.5. 'Medium bridges' were found with a rate of around 22%, and 'long bridges' were identified with a rate of around 85%. For the water masks **with** additional post-processing, the following results were obtained. 'Medium bridges' were recognised with a rate of around 37% and 'long bridges' were found with a rate of around 88%.
So, a small but significant influence of the diverse post-processing procedures can be observed in the water masks' generation. The additional post-processing increases the number of detected bridges, while at the same time the number of false detections rises, too. So, this equalises in a way the enhanced results obtained with the extra post-processing. The number of false positives, i.e. verified bridge areas that are not real bridges in the original satellite images, is very high too. A detailed analysis can



(a)



(b)

Figure 3.25: (a) an enhanced NIR image of an area near the city of Düsseldorf (GER). The located bridge deck area is coloured green. (b) The corresponding thematic map. Water pixels are coloured blue, islands are orange, false positives are red and the verified bridge is green.

Data	Medium Bridges visible	Medium Bridges found	Long Bridges visible	Long Bridges found	False Positives
HRC	24	7	42	34	32 (11 images)
RE	21	3	60	53	32 (7 images)
Total	45	10	102	87	64 (18 images)

Table 3.9: number of visible and recognised bridges from water masks **without** additional post-processing.

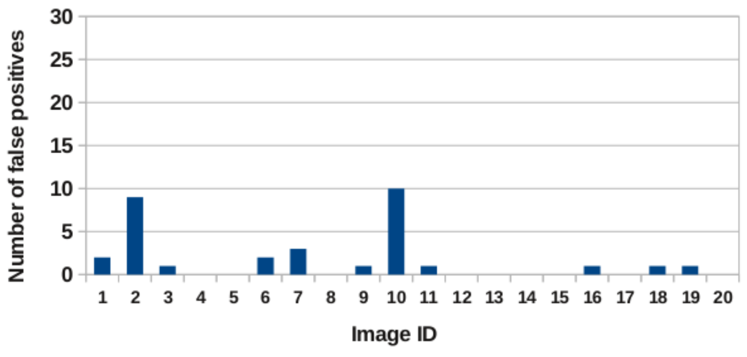
Data	Medium Bridges visible	Medium Bridges found	Long Bridges visible	Long Bridges found	False Positives
HRC	24	11	42	37	34 (10 images)
RE	21	6	60	53	43 (8 images)
Total	45	17	102	90	77 (18 images)

Table 3.10: number of visible and recognised bridges from water masks **with** additional post-processing.

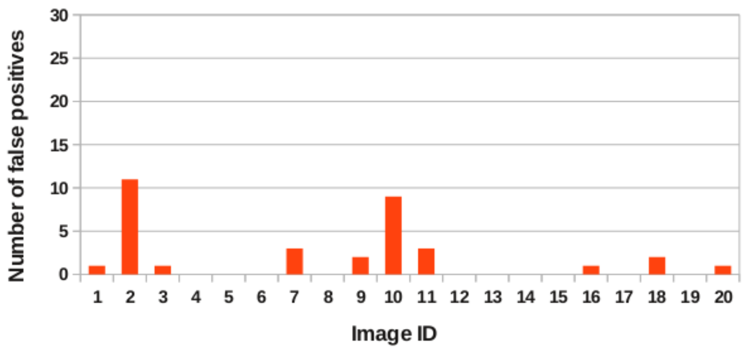
be obtained from the histograms in figure 3.26. In most cases, there are less than ten false positives in an image. A significant cluster of false positives can again be observed in image RE 17. But, as already mentioned in section 3.3.5, this image shows the tsunami disaster area in Japan. The scene will be presented in detail in section 3.5.5. An explicit effect of the different original satellite image data on the results cannot be observed. A thorough discussion on the results, as well as a comparison with the results of the first bridge recognition method, will be given in section 3.5.6. In the following paragraphs, the four special cases presented in section 3.3.5 are considered again.

Special case 1: small river segments

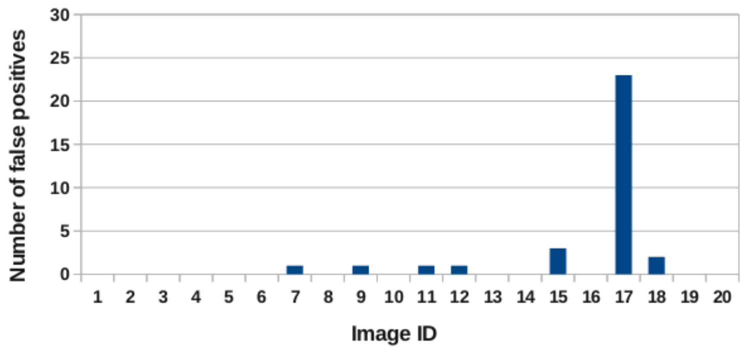
The situation of a small river segment is examined first. In figure 3.27 (a), a small part of a PAN image shows an area in the city of Moscow (RUS). The result of the second bridge detection method is laid on top. Only one bridge is recognised by the algorithm, illustrated by the green segment on the upper bridge. The respective thematic map is presented in figure 3.27 (b). All larger water bodies are framed there by a black rim. The water segment without such a rim is too small and thus not considered in the bridge detection process, so the two adjacent bridges, above and below that segment, cannot be found.
Another problem can be seen on the recognised bridge. There, the *circular structuring element* erases a significant part of the bridge heads. In the current situation of a 'medium bridge', this means that only a fraction of the bridge deck area in the center of the bridge is detected and labelled as a bridge. An additional size filter of the



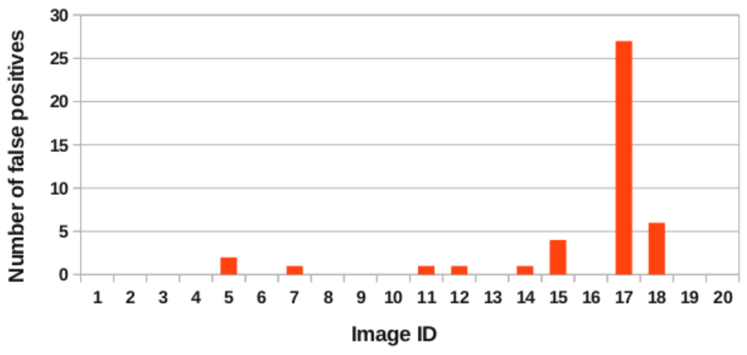
(a)



(b)



(c)



(d)

Figure 3.26: distribution of false positives in the second bridge recognition for each evaluated image. The HRC data set and a water mask **without** extra post-processing (a) and **with** extra post-processing (b). The RE data set and a water mask **without** extra post-processing (c) and **with** extra post processing (d).

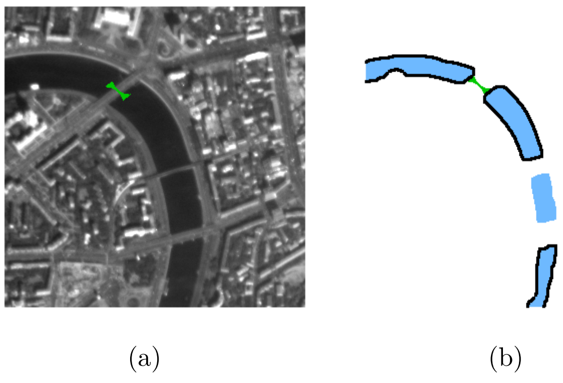


Figure 3.27: the first special case. (a) A PAN image of the city of Moscow (RUS), combined with the result of the 2nd bridge recognition method. The green segment illustrates the located bridge deck. (b) The corresponding thematic map. Large water segments are framed by a black rim.

bridge deck area reduces the detection rate of smaller 'medium bridges'. All in all, these filter operations are a challenge, especially for 'medium bridges', however, they proved to be useful in reducing the number of false positives.

Special case 2: a bridge to an island

The second special case is a bridge leading to a natural island. Figure 3.28 (a) shows Ellis island in the New York harbour area. The image is combined with the result of the second bridge recognition method. The bridge was detected by the algorithm and is labelled as a green segment. The corresponding water mask is presented in figure 3.28 (b). Please note that Ellis island has a diameter of more than twice the dimension of the water region extension (16 pixels), so after closing the water regions, the island is a non-water area again. If the island would be smaller, with a radius of less than 16 pixels, it would be completely filled in the water extension process, but the extended water would not be removed in the reduction step, since the new water rim is only on the mainland and cannot reach into the island. As a result, the bridge and the small island would be regarded as a cavity and excluded. In the present situation, however, the island is large enough, so that the bridge between the island and the mainland is detected.
Additionally, in contrast to the first bridge recognition method, this time no check is performed on whether the bridge lies between two different water segments or not, so the bridge deck area is accepted as valid bridge. The situation demonstrates the trade-off that has to be made at some point between suppression of false positives and detection of special bridge structures.

Special case 3: a curved bridge

Curved bridges represent a minority, yet it is desirable to include their detection in a bridge recognition method as well. In figure 3.29 (a), the enhanced NIR image of a coastal area near the city of New York (USA) is displayed. The result of the

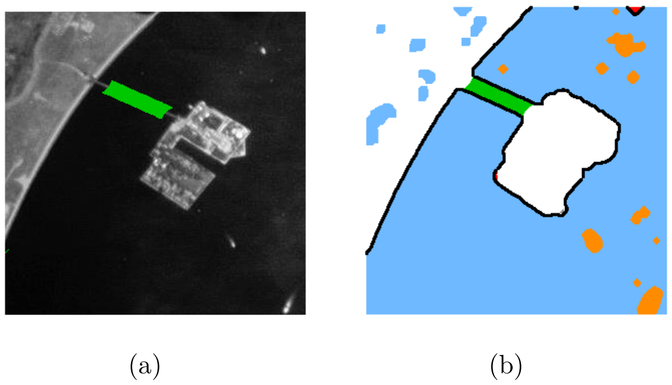


Figure 3.28: the second special case. (a) A PAN image of Ellis Island in the New York harbour area (USA), with a labelled bridge deck area in green. (b) The corresponding thematic map. Water areas are blue, islands are orange, false bridge deck areas are red and the verified bridge is green.

second bridge recognition is overlaid on the satellite image and the located bridge is indicated by the green segment. From the scene, it can be seen that the present approach is capable of detecting curved bridges. Theoretical considerations, however, revealed that the degree of curvature and the orientation of the bridge in an image might have an impact on its recognition. Due to the fact that, in the data set used in the study of this thesis, all curved bridges were found, a detailed analysis was omitted. A discussion will be given in section 3.5.6.

Special case 4: a ship below a bridge

The particular situation that a satellite image is taken at the moment a ship is cruising under a bridge shall be reviewed too. Again, the NIR image and the corresponding RGB image of a part of the city of Cologne (GER) are depicted in figures 3.30 (a) and (b). The result of the bridge recognition is superimposed on the NIR image. Both recognised bridges are indicated correctly by the green segments, although the ship under the lower bridge is corrupting the bridge's shape in the water mask (see fig. 3.30 (c)). Here, the bridge verification step identifies the ship as a cavity in the water segment, and so this part is excluded from the remaining bridge deck area. The reliable representation of a bridge deck area is a significant advantage of the second bridge detection method.

Case study of a natural hazard: the 2011 tsunami in Japan

The performance of the second bridge recognition method shall be appraised under real disaster conditions as well. The same scene is analysed as in section 3.3.5: an area around the river mouth of the Natori river. In figure 3.31 (a), the enhanced NIR image is displayed, combined with the result of the second bridge recognition method. The green segments illustrate located bridge deck areas. Due to the fact that this is a region after flooding, many additional water areas appear in the water mask (see fig. 3.31 (b)), and between these water regions a lot of 'additional bridges'

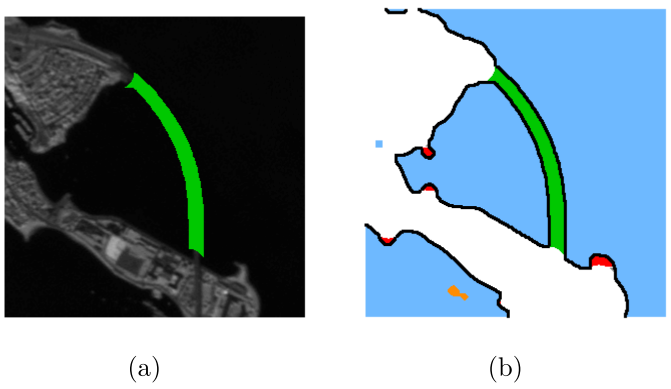


Figure 3.29: the third special case. (a) An enhanced NIR image of an area near the city of New York (USA). The located bridge deck area is labelled green. (b) The corresponding thematic map. Water areas are blue, islands are orange, false bridge areas are red and the verified bridge deck area is green.

were found by the algorithm. The two 'long bridges' (no. 1 and 2) spanning the river have been detected. The bridges no. 3, 4 and 5 remained undetected due to the erased (or too small) adjacent water bodies. The outcome will be discussed in section 3.5.6.

Hardware resources

The hardware requirement of the current implementation is around one third of the available Slices on the Spartan-3E FPGA. Additionally, two BlockRAM cells are occupied. The estimated maximum clock frequency is $43MHz$. In the current set-up with a clock rate of $12.5MHz$, a 1024×1024 pixel image is processed in about $14s$. So, the processing time is significantly longer than in the first bridge recognition method. The power consumption is clearly below $1W$. All details are listed in table 3.11 and a discussion will be given in the next section.

Slices	3097 (35%)
BlockRAM	2 (7%)
Max. Clock Frequency	$43MHz$
Power	$0.2W$

Table 3.11: *Nexys-2* implementation parameters

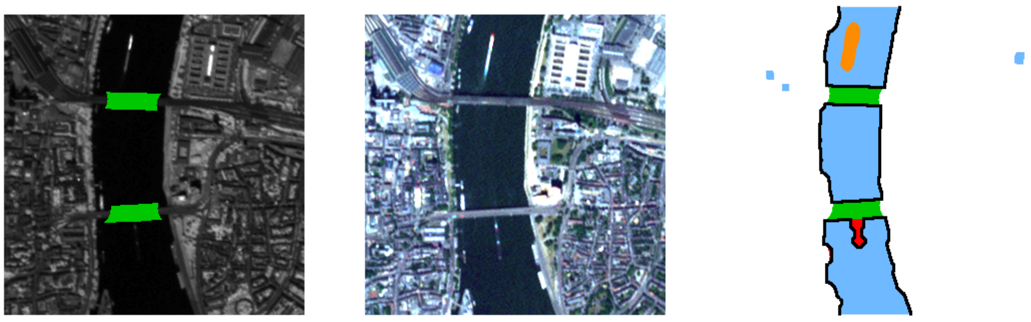


Figure 3.30: the fourth special case. (a) An enhanced NIR image of the city of Cologne (GER) combined with the result of the bridge detection. (b) An RGB composite image of the same scene. (c) The associated thematic map. Water bodies are blue, islands are orange, false bridge deck areas are red, and verified bridges are green.

3.5.6 Discussion, Comparison and Summary

Discussion

The second bridge recognition method was developed to overcome shortcomings with the first bridge recognition method and to enable a detailed analysis of a bridge. The method differs significantly from the first and the outcome is a thematic map where each located bridge deck area is marked.
In contrast to the first method, the algorithm was implemented directly into hardware. No preparatory software version was established. In retrospect, the algorithm development and implementation functioned well.
The input data are thematic maps depicting all larger water bodies and islands. No a-priori knowledge of a scene is necessary. The output for a 1024×1024 pixel image is obtained in around $14s$, whereby the processing times vary with the number and size of the water and bridge segments.
The statistical assessment revealed a detection rate between 85% and 88% for 'long bridges‘. In comparison with literature data of 80.95% in Zhang and Sun (2011) and 89.61% in Chen (2008), this is a thoroughly satisfactory result. Please note that the evaluated bridges in the literature and their size/ appearance in an image are unknown, so this comparison is made with reservation. 'Medium bridges‘ were recognised with a far lower detection rate of 22% to 37%. The main reasons are size filters for the water segments as well as for the verified bridge segments, and the erasure of bridge pixels at the bridge heads due to the *circular structuring element*. Further analyses showed that variations in the template size or additional morphological operations do not have an advantageous influence, so a detailed consideration of the applied morphological operations is advisable. The additional post-processing step in the water mask generation yields slightly better detection results, but it also increases the number of false positives. Overall, water masks with extra post-processing do not seem advantageous.

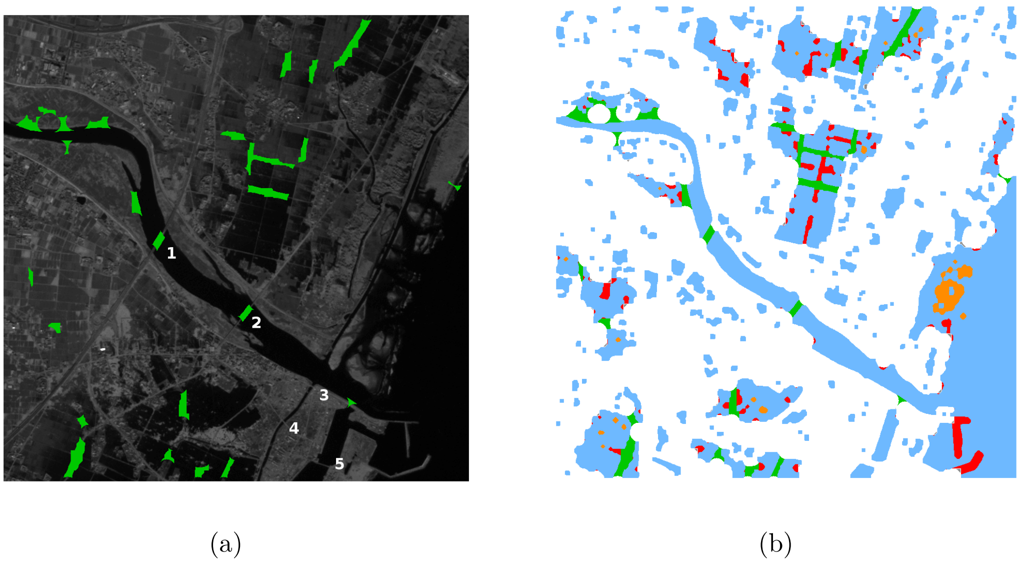


Figure 3.31: (a) an enhanced NIR image of the river mouth of the Natori river combined with the result of the bridge recognition (green segments). The five visible bridges are numbered. (b) The corresponding thematic map. Water bodies are blue, islands are orange, false bridge deck areas are red, and verified bridges are green.

The special cases considered showed some characteristics of the algorithm in more detail. Still, small river segments pose a challenge, and it is impossible to detect adjacent bridges if the river segment is not regarded as a true water segment. Instead of a size filter, another characteristic of water should be used to separate true water areas from false ones. The assumption that a bridge is between two different water segments was not used in the current version of the second bridge detection algorithm, thus the bridge to Ellis island was found. This check could be implemented as an additional processing step. As a result, the bridge to Ellis island would not be accepted as a valid bridge any more, but false positives would vanish, too. The third special case showed that curved bridges can be found with the proposed method. In theory, however, the same bridge might not be found if it is slightly rotated so that it forms a U-shaped segment directly in one of the main directions (top, bottom, left, right). Then, it might be the case that the bridge verification process prohibits its detection. A detailed analysis of this situation would involve a lot of additional work and since all curved bridges in the current data set were found, the analyses were omitted at this point. With the fourth special case, it can be shown that by using the second bridge recognition method it is possible to distinguish between the real bridge structure and a ship beneath. This is an advantage since such a situation does not alter the recognition result and enables a separate analysis of the bridge and the ship.
The case study of the tsunami disaster region in Japan revealed a similar result as with the first method. Besides the two detected (true) 'long bridges', a lot of additional bridges were marked in the thematic map. As already mentioned, they appear in the spaces between the extra water regions caused by the flooding. Their shape is

often akin to the appearance of true bridges, so they were not excluded in the bridge verification process. However, with the second bridge detection method, a detailed analysis of the recognised bridge deck areas is possible, so a subsequent test might exclude these as well. On the other hand, as already mentioned in section 3.3.6, these additional bridges represent potential transportation routes through the affected area so they might be useful as well. One technique to get additional information about a bridge and determine its accessibility more reliably will be presented in section 3.6.
In conclusion, the second bridge recognition method needs significantly more processing time, but it fulfils the posed requirements: the detection of curved bridges and the possibility to evaluate detected bridges in detail. All in all, the method differs from the first one but is not necessarily superior. Both methods are summarised in short.

Comparison of the first and the second bridge recognition method

The object of both methods developed in the course of this thesis was to locate and label bridges over water in a captured satellite image, yet the outcomes differ clearly. The first method labels a located bridge with a single dot at the center point of the bridge, whereas the second method marks the whole bridge deck area. According to this, the processing time for a 1024×1024 pixel image is $2s$ with the first method and $14s$ with the second. The hardware requirement, i.e. the number of used Slices and BlockRAM, is almost identical in both realisations, but the maximum clock frequency differs significantly. The first algorithm can be run at a maximum clock rate of $77MHz$, the second at only $43MHz$ at most. So, the already faster routine of the first method can be operated much faster in addition. However, the power consumption of the second method is only $\frac{1}{4}$ of the necessary power for the first implementation.
In conclusion, if it is sufficient to know whether there are bridges located in a certain area or not, then the first method can provide this knowledge in a very short period of time. If additional information about the site is desired, then the second method has to be used. So, the preference for one method or the other depends on the actual mission goal and the (processing time) requirements.

Summary

The second bridge recognition method was developed to improve the detection results of the first method, and to enable an examination of located bridges. As for the first method, it is implemented on a small FPGA chip and processes an image in a few seconds. No supplementary information on a scene is necessary. In addition, the outcomes might provide a basis for further investigations of an image. All in all, the presented method would seem to be a reasonable tool for integration on a future satellite mission.

3.6 Moving Vehicle Detection

3.6.1 Introduction

The final method presented in this thesis is a contribution to anomaly detection. An anomaly, in general, is an irregularity or deviation from an (expected) normal case. In satellite imagery, an anomaly is the appearance of an object at different positions in one colour image.
The outcome of the second bridge recognition method, the labelled bridge deck areas, can be used to examine in detail located bridges. In the analysis, moving vehicles can be found on a bridge, since they appear as coloured streaks in a colour image. The detection of moving vehicles is an indicator for a good condition of the bridge, because road traffic will only appear on an intact bridge. With this information, a reliable conclusion could be made as to whether a bridge is accessible or not. Of course, the converse argument, if no traffic is detected then the bridge is not accessible, does not apply. Nevertheless, any ancillary knowledge is welcome that gives further information on the situation on ground. In order to identify moving objects in a single satellite image, the already mentioned time-lag in the RE data collection procedure is used.
In the section below, this effect is explained followed by a literature survey. Then, a new approach is presented to detect moving vehicles, together with its hardware implementation. At the end, the findings and a conclusion are given.

3.6.2 Time-Lag in *RapidEye* Image Collection

A small time-lag in the data acquisition of different sensor units is a known characteristic of Very High Resolution (VHR) sensors e.g. on-board the *QuickBird* satellite (Pesaresi et al., 2008). Several publications use this effect to estimate moving objects in a single satellite image. A short review will be given in section 3.6.3.
A similar time-lag also occurs in the data collection procedure of RE imagery. Due to the internal set-up of the sensor, each spectral band is collected by a separate CCD array, and there is a time difference in imaging the same point on the ground between each band. The maximum time-lag is around three seconds between the blue and the red band. The data products of Level 1B and Level 3A are co-registered using a Digital Elevation Model (DEM) to correlate all bands to the red-edge band (reference band). A final alignment is performed using an auto-correlation approach between the bands (RapidEye, 2012). The outcome of this procedure is a matching of all bands, so that static objects have the same location in each band. Moving objects, however, still have a different position in each band after the matching process. This can be observed for clouds, aeroplanes and vehicles as well.

> *'Bright vehicles moving on the ground will also look like coloured streaks due to the image time differences.'* (RapidEye, 2012)

This effect will be used in the study of this thesis to detect moving vehicles on a bridge. However, a remark shall be given on two things: firstly, due to the spatial resolution of $5m$, only larger vehicles such as trucks or vans might be visible as bright spots on a road or bridge. Secondly, besides the size of an object also the colour has a

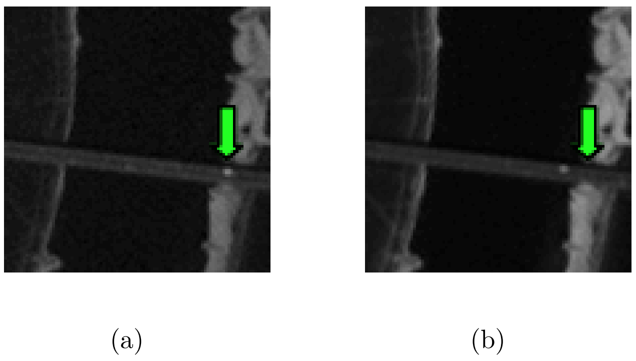


Figure 3.32: visualisation of the time-lag in RE imagery (a) The blue band of a scene. The green arrow points to a moving object (bright spot). (b) The red band of the same scene. The green arrow is still at the same position as in (a), but not the object.

significant impact on its recognition. White vehicles are visible as bright spots in both spectral bands, whereas otherwise-coloured cars might be well visible in one spectral band but not detectable in the other band at all. A blue van, for example, will appear as a bright spot in the blue band but not in the red. This spectral dependence is a problem to be solved.
In figure 3.32, a bridge near the city of Düsseldorf is displayed. Therein, a bright spot is visible on the road in the blue and red band (fig. 3.32 (a) and (b) resp.). A green arrow points to the location of the object in the blue band and is still at the same position in the red band. It can be seen that the bright spot is at a different position in the red band. The displacement and the direction (leftward on the upper side of the bridge) are in accordance with regular traffic in Germany. The green spectral band is left out, since two bands are sufficient to detect moving objects and the time gap is at its peak between BLUE and RED. If the green band would be included, a bright dot would be visible between the bright dots of the red and the blue band, since the data take of the green band is in-between the two. In a colour composite image (RGB), the result would be a coloured streak from blue via green to red. Taking the deduction above and the reasonable appearance of the anomalies, it is assumed that the bright spots in the individual bands represent a moving vehicle. The detection of moving vehicles on a bridge will be used to provide additional information on the accessibility of a bridge.

3.6.3 Literature

VHR sensor data are often applied in vehicle detection studies due to their very high spatial resolution of $1m$ (IKONOS) or better (QuickBird, WorldView-2) (Larsen et al., 2009). In addition, five articles use time-lag to estimate the velocity and direction of identified vehicles. They are reviewed below.
In Pesaresi et al. (2008), QuickBird imagery is used to detect moving targets from a single optical satellite image. The optical system on-board the QuickBird satellite consists of two sensors, a panchromatic sensor (PAN) with a spatial resolution of

0.6m, and a multi-spectral sensor (MS) with a spatial resolution of 2.4m. The time-lag between the data-takes is approximately 0.2s. The vehicles were found in both bands and a velocity was estimated using the distance between the two points and the known time difference. A similar approach was presented in Xiong and Zhang (2008). However, in both studies, the target vehicles were extracted manually from the images and this is not an appropriate procedure for an automated on-board application.
Satellite imagery from QuickBird was also used in the studies of Leitloff et al. (2010) and Liu et al. (2011). Both approaches automatically detect vehicles in the PAN image and extract a small region as a template. The corresponding vehicle in the MS image is found by a correlation of the template with the MS image in Liu et al. (2011), and by comparison of gradient directions in both images, in Leitloff et al. (2010). Both matching techniques are computationally expensive and therefore not considered as an appropriate choice for an FPGA implementation.
An article from Salehi et al. (2012) proposed an automatic extraction of moving vehicles from single WorldView-2 imagery. The introduced approach consists of three steps: (a) an automatic road extraction, (b) a moving vehicle detection using a Principle Component Analysis (PCA) based change detection and (c) an estimation of the target's velocity. Here in this study, the road extraction can be simplified by using the result of the second bridge recognition method, and the velocity estimation is out of scope due to the relatively coarse spatial resolution of the RE data. The PCA is an advanced image processing technique that is too complex for a brief image evaluation task, so it is omitted here too.
So far, no application of RE data has been found in literature addressing this topic. The reasons might be the coarser spatial resolution and the spectral dependence. Nevertheless, in this thesis an approach is pursued to use this effect for the current purpose and it will be introduced in the section below.

3.6.4 A New Approach

The review of current literature revealed a keen interest in the detection of moving vehicles and the estimation of their velocities and their directions. So far, satellite image data from VHR sensors was applied in such studies. The aim of the present work is, however, just to provide information on whether there are moving vehicles on a bridge or not. No accurate detection and description of single objects is necessary. Thus, it is assumed that the mentioned time-lag in the data collection procedure of RE imagery might provide sufficient information. A conservative approach was chosen, so that it is more likely to miss traffic than to falsely locate moving objects. All in all, it is just ancillary knowledge that might support the decision whether a bridge is still accessible or not.
The principle procedure of this basic approach is depicted in figure 3.33. The main processing steps are listed on the left-hand side. The intermediate results are illustrated on the right-hand side. Please note that the small images represent approximations of the corresponding steps. The final thematic map is not an exact representation of the result that is obtained from the depicted satellite images in the first row, it is just a visualisation of the key points.

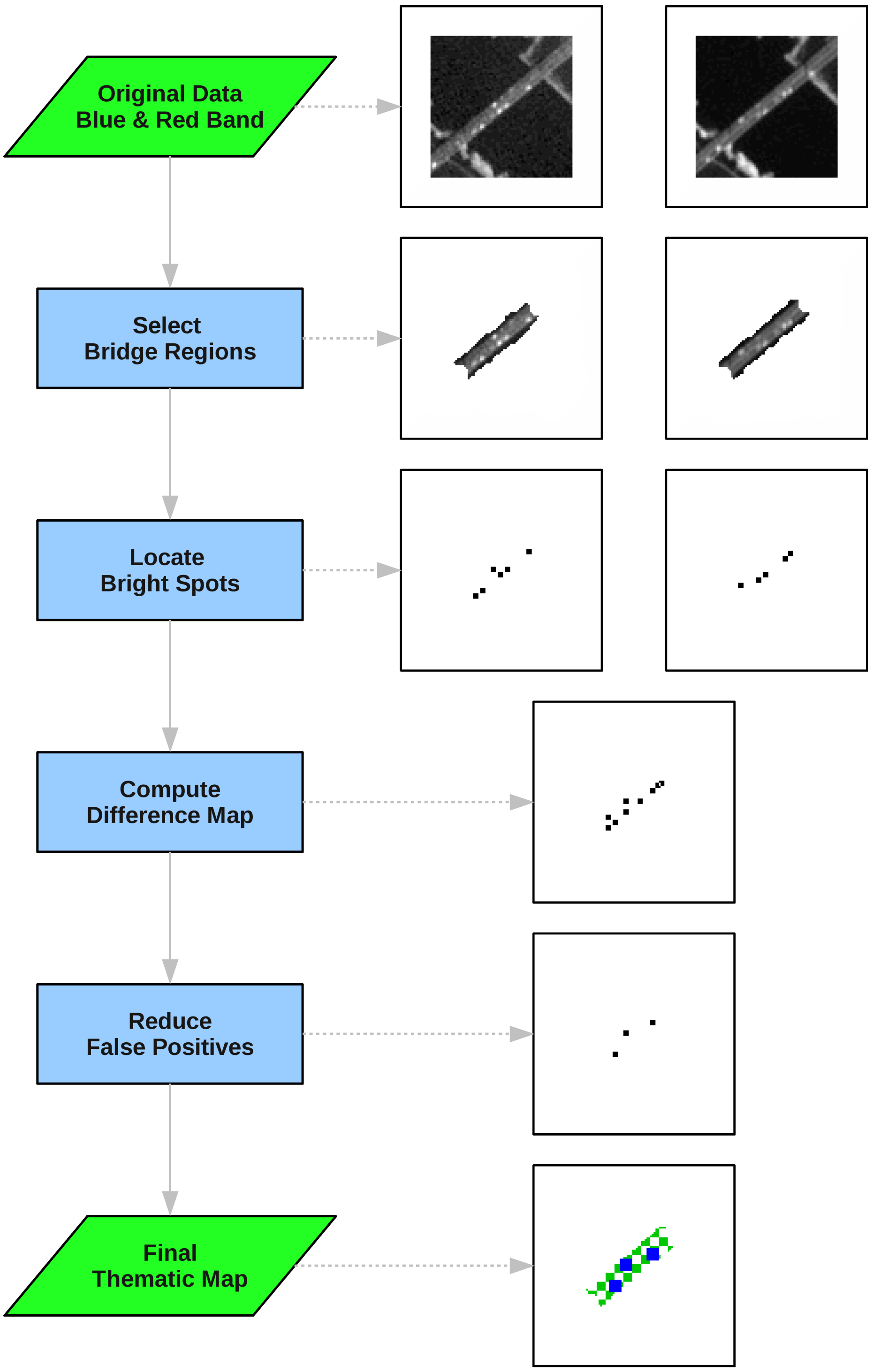


Figure 3.33: flow chart of the moving vehicle detection algorithm.

A description of the procedure can be given as follows:

1. Examine the bridge deck areas in the blue and red spectral band. They are selected via a bridge template, the result of the second bridge recognition method.
2. Search each area for bright pixels that potentially represent vehicles. Since the spatial resolution of the imagery is $5m$, only larger vehicles such as trucks or vans will be spotted. Furthermore, white vehicles are more likely to be detected, since they are clearly visible in both spectral bands.
3. Label all bright spots that occur uniquely in one band. If a bright pixel appears at the same position in the blue and the red band, it is assumed to be a static object, and is thus discarded from further considerations.
4. Reduce the number of false positives and draw the final outcome on a thematic map.

The big blue dots in the small image at the bottom of figure 3.33 represent located moving objects, and the chessboard pattern on the bridge area indicates that a human operator found some moving vehicles on that bridge too. The exact hardware realisation is described below.

3.6.5 Hardware Routine

The proposed algorithm uses RE satellite imagery of the blue and the red band, as well as the corresponding thematic map as input data. In order to keep the hardware procedure similar to the previous implementations, an extract from the blue band, the red band and the thematic map is selected to form a composite image, with almost the same size (1024×1020 pixels) as the former input images. As a result, each small scene depicts the same location, has a dimension of 1024×300 pixels, and is framed by a white rim of 20 pixels at the bottom and the top to separate the images. The traffic detection procedure is as follows:

1. Set all pixels in the blue and red band to background pixels (white) that are not bridge deck pixels (green) in the thematic map.
2. Find all bright pixels in the blue and the red band that have a higher value than the mean of the eight adjacent pixels. The difference must be greater than 4.
3. Remove all bright pixels that have the same location in the blue and in the red band to exclude static bright spots.
4. Remove single bright pixels (since they are likely to be false positives).
5. Review an area of 61×61 pixels in the red band that has a bright pixel at its center in the blue band. If at least one bright pixel is detected in the red band, label the pixels in both bands as a 'verified anomaly'.

6. Remove all single bright spots.

The value for the difference in brightness was determined empirically. The search area of 61×61 pixels was derived from the assumption that trucks and vans do not drive faster than $180 \frac{Km}{h}$ that is, at the given spatial resolution of $5m$ and a time-lag of $\sim 3s$, a distance of 30 pixels in one direction. The results obtained from an analysis of real satellite image data are presented in the following section.

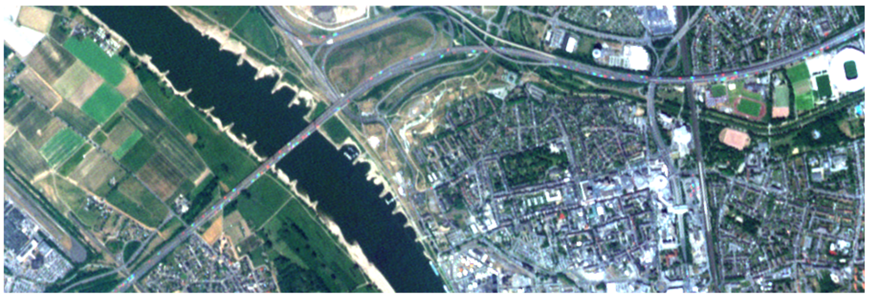
3.6.6 Experimental Results

Example image

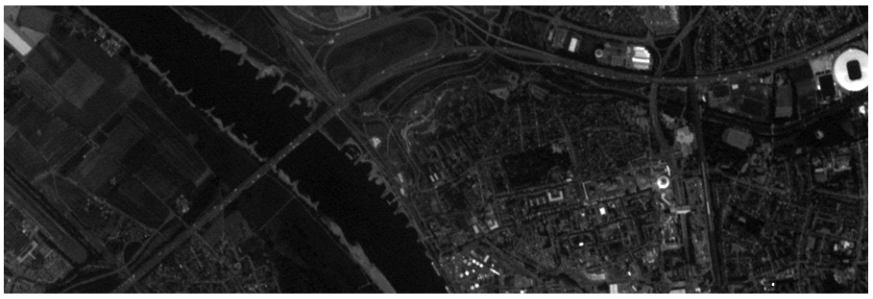
Figure 3.34 (a) is an RGB composite image of a scene near the city of Leverkusen (GER). For details on the data-take, see appendix A.3, RE 24. A motorway bridge crosses the river Rhine and several bright dots on the bridge and on the continuative motorway indicate a lot of traffic. The same bright spots are visible in the blue and red band images (fig. 3.34 (b) and (c), respectively). A closer look reveals a different position of the numerable dots in the blue and red band images due to the mentioned time-lag in image collection. The corresponding thematic map is illustrated in figure 3.34 (d). A chessboard pattern on the bridge deck area now indicates the traffic discovered by a human operator. In the current version of the algorithm, it is not possible to tell exactly which two anomalies (one in the blue and one in the red band) belong to the same vehicle on ground, but this is of minor relevance here, since only information as to whether there is traffic or not is desired. The big blue dots display the result of the moving vehicle detection, and highlight the located objects.

Statistical evaluation

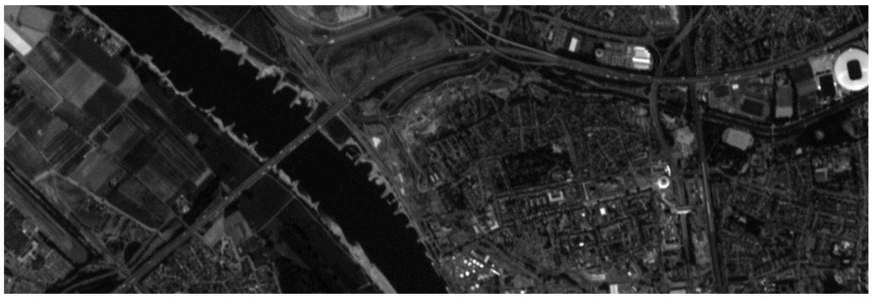
A statistical evaluation was carried out on 20 RE scenes. For details on the data-takes, see appendix A.3, RE 21-40. In the images, a total of 35 bridges is visible. On 27 bridges, no moving objects were observed, while on eight bridges several moving vehicles were spotted by a human operator. The statistical outcome is displayed in table 3.12. Moving objects were detected on four of the eight bridges with traffic. The reason for the misclassification in four cases is the faint visibility of the bright spots. The human observer was able to extract even slightly brighter points on a noisy bridge, whereas the algorithm suppressed such small signals to avoid false detections. Besides two false classifications, bridges without traffic were identified correctly, which reflects the conservative approach. The false detections arose in one case on a very shiny bridge, and in a second case due to a small moving boat near the bridge. In the first case, no traffic was detected by the human observer due to a high reflectance of the bridge's surface. So even if the algorithm worked correctly, it could not be verified, thus the result was declared as a false detection. In the second situation, a boat was in the vicinity of a bridge, thus this non-water area was merged with the bridge area. The detection of a moving object on the bridge (area) was therefore correct, but because it was a boat and not road traffic, the outcome was anyway a false detection. Overall, 29 of 35 bridges were evaluated correctly.



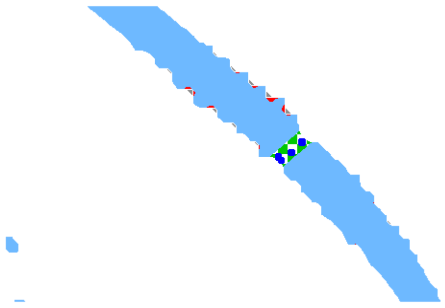
(a)



(b)



(c)



(d)

Figure 3.34: a scene near the city of Leverkusen (GER). (a) An RGB composite image. (b) The blue band image. (c) The red band image. (d) The thematic map. The chessboard pattern on the bridge indicates moving objects found by a human operator. The blue dots illustrate moving vehicles detected by the proposed algorithm.

Recognition	Bridges **with** traffic	Bridges **without** traffic
Correct	4	25
Wrong	4	2
Total	8	27

Table 3.12: the number of bridges **with** and **without** traffic and their recognition by the proposed method.

Slices	1475 (17%)
BlockRAM	0
Max. Clock Frequency	$112MHz$
Power	$0.2W$

Table 3.13: *Nexys-2* implementation parameters

Hardware characteristics

The hardware requirement of this implementation is very small. Only 17% of the available Slices are used, and no BlockRAM is needed. The maximum clock frequency is estimated by the ISE design suite to be more than $100MHz$. So an image evaluation of a 1024×340 pixel scene, that takes less than two seconds in the current set-up with $12.5MHz$, can be performed in a fraction of a second with a much higher clock rate. The power consumption is again far below $1W$. Table 3.13 lists the main characteristics of the implementation.

3.6.7 Conclusion

In order to provide additional information about the accessibility of a bridge, a novel approach has been proposed. Under the assumption that moving vehicles on a bridge suggest that a bridge is still in good condition, a time-lag in the data collection of RE imagery was used to identify moving objects. Such an effect was noticed on VHR imagery before and several publications employed this effect for moving target detection, but so far it was never used on RE image data.
For FPGA implementation, multiple low-level image processing operations were combined to recognise moving traffic on a bridge. The final hardware realisation is small and performs an image analysis of a 1024×340 pixel scene in less than two seconds. The run time depends on the image content, but the processing time averaged at $2s$, so a relaxed real-time criteria is fulfilled.
A statistical evaluation revealed a traffic recognition rate of 50%, whereby eight bridges with traffic do not really represent a statistical measure, but moving vehicles were hard to find in the data set. Several moving vehicles were not detected, mainly

due to their very weak appearance. Overall, the algorithm pursued a conservative approach to avoid false positives. The classification was correct in around 83% of all bridges (with and without traffic).
The work presented in this study is a technology demonstration on ground and Level 3A data were used. So pre-processing, e.g. co-registration of the bands, was already performed. In case of an application on-board a satellite, these pre-processing steps have to be taken into account. But, they might already be conducted in data pre-processing inside modern sensor systems.
All in all, the proposed method is an additional tool for image evaluation on-board satellites. Further similar applications might include the detection of moving ships, aeroplanes, or other moving objects of interest.

Chapter 4

Findings and Prospects

4.1 Findings

This final chapter of the present PhD thesis summarises the key findings of the conducted work and provides an outlook towards future areas of research.
The idea of this thesis is to support mission operations on future satellite missions. The image acquisition procedure in space shall become more flexible and adjusted to the content of the currently acquired optical satellite imagery. Therefore, captured scenes shall be evaluated directly on-board a spacecraft, and a brief overview of the information content shall be provided (to the on-board planning unit). In the event of severe changes on ground, this might enable a prompt re-scheduling of the timeline and initialisation of appropriate actions, e.g. a second data-take of the affected area. Due to the autonomy of the satellite, the currently necessary direct interaction of the ground segment can be reduced or omitted. Further benefits would be a faster supply of the data to terrestrial data centres and additional valuable information on a certain region.
In the course of this doctoral thesis, five different algorithms were designed to extract useful information from optical satellite imagery. The level of obtained knowledge spans from a primal classification of water bodies up to a detection of moving objects on a recognised bridge. Considering the real-time requirements and power constraints on-board a satellite, Field Programmable Gate Arrays (FPGAs) were chosen as a reasonable processing unit. Hence, all algorithms were specially tailored for an FPGA implementation. A résumé of the thesis can be given as follows:

The thesis starts in **Chapter 1** with a proposal for on-board image evaluation. Reasons were given to explain the additional work of image processing in space compared to a terrestrial implementation. But also the benefits were highlighted: an adjusted image acquisition and a faster response in case of an emergency. Practical scenarios were visualised with regard to current or near-future satellite missions. In order to cope with some real-time requirements and the constraints on-board a spacecraft, Field Programmable Gate Arrays (FPGAs) were introduced as an adequate processing unit. The chapter concludes with a listing of the scientific contributions and the structure of this thesis.

At the beginning of **Chapter 2**, the satellite imagery used in the study of this thesis was presented. Panchromatic satellite image data from the HRC on-board the satellite Proba-1 was chosen with regard to the future ALOS-3 satellite mission, to demonstrate the applicability of the proposed methods to this kind of imagery. And satellite imagery from the RapidEye (RE) satellites was used to show the direct utility of the developed methods, since RE satellite image data are a part of the German contribution in case the International Charter Space and Major Disasters is invoked. Both data types have a spatial resolution of $5m$. A set of 20 HRC and 40 RE scenes was selected, including a real disaster situation, the 2011 tsunami in Japan. The data-take locations are distributed all over the world to ensure a variety of different scenarios. Finally, the performance of the algorithms was evaluated on this data set.
A reason for using an FPGA as a real-time image processing unit on-board satellites was already given in chapter 1. In chapter 2, some basic knowledge on this technology was provided and the challenges that come along with hardware programming were discussed. A high-level hardware description language, called Handel-C, was brought forward that suits the need for a rapid development of hardware algorithms. The whole experimental set-up was illustrated, with the employed Nexys-2 development board and its connection to a PC. The chapter concludes with a description of the overall experimental work flow.

In **Chapter 3**, the main part of this thesis, five different FPGA-based image processing algorithms were presented. At the beginning, an overview was given on the potential combinations of the algorithms. Due to the fact that each algorithm works independently of the other, and provided that the necessary input data are available, the algorithms can be arranged in a modular concept. So the output and the overall processing time can be adjusted to the satellite mission goal or requirements.
At first, a **water body extraction** was presented. With the proposed method, all major water bodies can be recognised in a PAN or NIR satellite image. The result is drawn after $6 - 8s$ (without extra post-processing) onto a binary water mask.
A statistical assessment was conducted by comparing the automatically generated water masks with manually derived ones. The overall agreement was about 80% as a minimum. A comparison with data from literature was established by analysing a similar scene of the city of Taipei (TPE). The results are comparable, albeit not definite, yet all larger water areas were represented in the water masks. An analysis of the tsunami disaster site was performed as well, to verify the performance of the proposed method under real disaster conditions. Besides the expected water regions, many additional water bodies were found due to the flooding in that area. So, the proposed method reflected the flooding in the corresponding water mask. The hardware implementation is characterised by the use of 32% of the available Slices on a Spartan-3E FPGA, a maximum clock rate of around $100MHz$, and a power consumption of $0.5W$.
In conclusion, the water body extraction gives a first, fast, and rough evaluation of the distribution of water areas in a scene. It was the basis for all further algorithms in this thesis.

The **first bridge recognition** algorithm can find bridges over water in a satellite image. The routine screens a binary water mask and labels all located bridges by a dot at the bridge center point. A result can be obtained after $2s$ in a 1024×1024 pixel image and the output is drawn on to a thematic map.
A statistical assessment revealed a recognition rate of around $76 - 78\%$ for 'long bridges'. Four special cases were considered separately. They showed, among other things, that small river segments pose a difficulty for the detection rate, and that curved bridges cannot be found with the proposed method. Furthermore, an analysis of the tsunami disaster region in Japan was performed. To the best of our knowledge, this is the first time that a bridge recognition method was evaluated on real disaster images. The outcome showed the reliable detection of 'long bridges' and might provide knowledge on potential transportation routes through an affected area. The transfer of the pre-developed software code into working hardware was challenging and the code had to be verified significantly. In the end, the hardware implementation needs around 31% of the available Slices on the Spartan-3E FPGA, has a maximum clock rate of $77MHz$, and a power consumption of $0.8W$.
The proposed method is independent of (potentially outdated) map data and can provide up-to-date information, e.g. on recently constructed pontoon bridges, in an area. It is a very fast and completely autonomous bridge recognition method.

The **island detection** algorithm was intended to find rooftops in case of a flooding, since these spots might pose a shelter for people waiting for help. In the current study, the term 'island' refers to all non-water areas that are completely surrounded by water in a water mask. This includes natural islands, as well as artificial islands, such as rooftops or ships.
The applied technique is similar to a proposed method from literature that was also implemented on an FPGA. Yet some modifications were made in the present algorithm to optimise the procedure. In the conducted experiments, all islands were found and the average processing time for a 1024×1024 pixel image is approx. $4s$. The hardware implementation needs 13% of the available Slices of the Spartan-3E FPGA, the maximum clock rate is $87MHz$, and the power consumption is only $0.2W$.
Besides the detection of rooftops, an a-priori masking of all islands in a water mask raises the detection rate for a second bridge recognition algorithm. And a detailed analysis of the islands might present a novel field of research (see section 'prospects' below).

The **second bridge recognition** method was developed to enhance the detection results of the first method, and to provide the possibility for a detailed analysis of the site. A completely different approach was chosen and the outcome is a labelling of the complete bridge deck area of located bridges.
A statistical evaluation showed a detection rate of $85 - 88\%$ for 'long bridges'. The same four special cases that were considered in the first bridge recognition, were regarded again. Still, small river segments pose a challenge to bridge detection, however, in contrast to the first method, the present one is able to detect curved bridges and yields robust results in case a ship is cruising under a bridge in an image. The analysis of the tsunami disaster site in Japan showed, as in the first method, a reli-

able detection of 'long bridges'. The algorithm was directly written in Handle-C, no (preliminary) software version was developed, and the FPGA implementation functioned well. The processing time for a 1024×1024 pixel image is around $14s$. The hardware realisation uses about 35% of the available Slices on the Spartan-3E FPGA, the maximum clock rate is $43MHz$, and the power consumption is $0.2W$.
As with the first method, also the second bridge recognition method does not require any additional information on the scene, and it operates completely autonomously.

In this thesis, a small time-lag in the data acquisition procedure of RE imagery was used to perform **moving vehicle detection** on recognised bridges. The small time gap in the data-takes allows the detection of moving vans or trucks. The conclusion is that if moving traffic is found on a bridge, this bridge is still accessible. To the best of our knowledge, it is the first time that this effect of RE satellite imagery was used for a detection of moving objects.
A statistical evaluation produced a correct evaluation (in cases with and without traffic on a bridge) in about 83%. The processing of a 1024×300 pixel scene is performed in around $2s$, and the hardware implementation is characterised by the use of 17% of the available Slices on the Spartan-3E FPGA, a maximum clock rate of $112MHz$, and a power consumption of $0.2W$.
This is supplementary information, providing more precise knowledge on the condition of a bridge. Yet, in addition, it is a demonstration that also satellite imagery with a relatively coarse spatial resolution of $5m$ can provide such useful information.

Please note that in the current experimental set-up the FPGA was operated at a clock rate of $12.5MHz$. The given processing times and power consumption specifications refer to this clock rate. If higher clock rates, e.g. maximum possible clock rates are used, the processing times will decrease but the power consumption might increase.

In summary, the five methods presented in the course of this thesis represent novel image analysis capabilities. The algorithms operate completely autonomously and process an image in a few seconds. Furthermore, they are implemented in a small FPGA, which enables real-time performance and affords a power consumption of less than $1W$. All in all, the developed methods would seem to be reasonable tools for on-board application on future Earth observation satellites.

4.2 Prospects

In this PhD thesis, five novel methods for image evaluation on-board future satellite missions were presented and their field of application was described. An outlook on potential future research activities, based on the proposed methods, shall complete the study of this thesis.

A first step could be the employment of satellite imagery with a spatial resolution of $1m$ or better. The satellite imagery used, with a spatial resolution of $5m$, is adequate for the proposed methods, however, it can be observed, e.g. with regard to the

ALOS-3 mission, that future optical systems will provide PAN imagery in a very high spatial resolution. This might be beneficial for object recognition tasks but might also include some challenges. In order to show that the current methods are also applicable to VHR satellite imagery, a demonstration using these data would make sense.

Furthermore, it would be reasonable to include more useful algorithms in an image analysis and represent the results on the thematic map too. An **automatic road extraction** would be beneficial to present potential transportation routes, and would suit a combination with the presented bridge recognition algorithms. In addition, a detection of **urban areas** would increase the information content of a thematic map and could potentially guide a second data-take to regions where it is more likely to find people waiting for help.

Another prospect might be the detection of **moving ships**. Since, '*Ship detection from remote sensing imagery is a crucial application for maritime security...*' (Corbane et al., 2010), several publications have addressed this topic already. In Zhu et al. (2010) and Corbane et al. (2010), methods for ship detection in $5m$ panchromatic *SPOT-5* imagery were presented, and in Saur et al. (2011), the green band of RE imagery was used to detect ships in general.
A novelty would be the separate detection of *moving* ships. This information would provide supplementary knowledge on the situation at sea. For that purpose, the labelled 'islands', the results of island detection, can be examined in detail to detect signs of the 'time-lag effect' in RE imagery. If a ship is moving, then a small part at the front side (in the direction of motion) is visible at a specific position in the red band, while there is water at this exact position in the blue band. The rear part of a ship is at a specific position in the blue band, while there is water at this position in the red band. Additionally, the water appears white due to the foaming at the back side of a ship. The result is an exposure of a bright red spot at the front side of a ship and a light blue spot at the back side of a ship in an RGB composite image. Thus, moving ships can be distinguished from non-moving ones, where the coloured spots do not occur.

Finally, the scope shall be widened a little to another potential field of application. In this thesis, the developed algorithms were primarily dedicated to an evaluation of optical satellite images. However, in principle, it is possible to adopt them to optical aerial imagery as well. The outcome is still a thematic map of the captured scene, including landmarks, such as bridges over water, off-shore wind parks, or particularly shaped islands. Such a map might be used in a **terrain aided navigation system** for airborne vehicles. Recognised features could be compared with known features from previous data-takes and thereby the position of a vehicle could be determined (Kim and Sukkarieh, 2004).

Appendix A

Satellite Imagery Details

The following pages list the locations, coordinates and dates of acquisition for the evaluated satellite image scenes. The first table shows details of the PROBA-1/ HRC data, the second and third table list details of the RapidEye image data.

Image ID	Geogr. Location	Center coordinates (approx.)	Date of acquisition
HRC 1	London, GBR	51°31’N 0°08’W	2005-11-07
HRC 2	Rostock, GER	54°11’N 12°08’E	2005-10-07
HRC 3	Moscow, RUS	55°45’N 37°34’E	2005-08-26
HRC 4	Canberra, AUS	35°18’S 149°08’E	2007-06-23
HRC 5	Taipei, TPE	25°05’N 121°31’E	2004-02-26
HRC 6	Seoul, KOR	37°32’N 126°55’E	2004-10-27
HRC 7	Montreal, CAN	45°25’N 73°38’W	2006-11-01
HRC 8	Istanbul, TUR	41°02’N 28°59’E	2008-05-08
HRC 9	Moscow, RUS	55°42’N 37°39’E	2010-10-07
HRC 10	Viedma, ARG	40°48’S 62°59’W	2010-06-24
HRC 11	Tres-Marias, BRA	18°12’S 45°14’W	2010-07-16
HRC 12	Cincinnati, USA	39°06’N 84°31’W	2010-09-09
HRC 13	Shanghai, CHN	31°16’N 121°33’E	2010-09-21
HRC 14	Moscow, RUS	55°43’N 37°39’E	2010-10-07
HRC 15	Shenyang, CHN	41°45’N 123°28’E	2010-12-11
HRC 16	Copenhagen, DEN	55°41’N 12°35’E	2003-03-31
HRC 17	New York, USA	40°42’N 74°03’W	2006-08-01
HRC 18	San Francisco, USA	37°49’N 122°29’W	2004-05-02
HRC 19	Sydney, AUS	33°51’S 151°13’E	2002-11-04
HRC 20	Helsinki, FIN	60°10’N 24°54’E	2011-06-29

Table A.1: details of the evaluated PROBA-1/ HRC data

Image ID	Geogr. Location	Center coordinates (approx.)	Date of acquisition
RE 1	Cincinnati, USA	39°06'N 84°29'W	2011-06-30
RE 2	Cincinnati, USA	39°06'N 84°32'W	2011-06-30
RE 3	Taipei, TPE	25°05'N 121°31'E	2010-09-03
RE 4	Cologne, GER	50°56'N 6°59'E	2011-05-25
RE 5	Düsseldorf, GER	51°13'N 6°45'E	2011-05-30
RE 6	Bonn, GER	50°45'N 7°07'E	2011-05-25
RE 7	Quebec, CAN	46°53'N 71°09'W	2010-06-21
RE 8	Quebec, CAN	46°45'N 71°17'W	2010-08-19
RE 9	Bordeaux, FRA	44°58'N 0°28'W	2011-05-25
RE 10	Bordeaux, FRA	44°49'N 0°33'W	2011-05-25
RE 11	New York, USA	40°43'N 74°00'W	2010-09-14
RE 12	New York, USA	40°46'N 73°56'W	2010-09-14
RE 13	New York, USA	40°48'N 73°49'W	2010-09-14
RE 14	Canberra, AUS	35°18'S 149°08'E	2010-02-20
RE 15	Seoul, KOR	37°32'N 126°55'E	2010-06-23
RE 16	Magdeburg, GER	52°08'N 11°39'E	2010-06-24
RE 17	Sendai, JPN	38°11'N 140°57'E	2011-03-19
RE 18	Iwanuma, JPN	38°05'N 140°53'E	2011-03-19
RE 19	Istanbul, TUR	41°03'N 29°02'E	2009-06-03
RE 20	Istanbul, TUR	41°02'N 28°57'E	2009-08-22

Table A.2: details of the evaluated RapidEye data for pattern recognition

Image ID	Geogr. Location	Center coordinates (approx.)	Date of acquisition
RE 21	Cincinnati, USA	39°03’30”N 84°25’00”W	2011-06-30
RE 22	Cincinnati, USA	39°05’40”N 84°31’20”W	2011-06-30
RE 23	Taipei, TPE	25°05’00”N 121°29’20”E	2010-09-03
RE 24	Leverkusen, GER	51°02’10”N 6°58’50”E	2011-05-25
RE 25	Cologne, GER	50°53’50”N 6°60’20”E	2011-05-25
RE 26	Düsseldorf, GER	51°11’10”N 6°46’30”E	2011-05-30
RE 27	Montreal, CAN	45°28’20”N 73°31’30”W	2011-06-15
RE 28	Bordeaux, FRA	44°57’20”N 0°27’50”W	2011-05-25
RE 29	New York, USA	40°36’20”N 74°02’20”W	2010-09-14
RE 30	New York, USA	40°42’40”N 73°59’30”W	2010-09-14
RE 31	New York, USA	40°34’20”N 73°53’50”W	2010-09-14
RE 32	New York, USA	40°38’30”N 74°07’50”W	2010-09-14
RE 33	New York, USA	40°51’10”N 73°56’50”W	2010-09-14
RE 34	New York, USA	40°47’00”N 73°53’20”W	2010-09-14
RE 35	New York, USA	40°48’40”N 73°49’00”W	2010-09-14
RE 36	New York, USA	40°48’00”N 73°48’40”W	2010-09-14
RE 37	Düsseldorf, GER	51°14’50”N 6°45’40”E	2011-05-30
RE 38	Düsseldorf, GER	51°13’40”N 6°45’40”E	2011-05-30
RE 39	Düsseldorf, GER	51°12’10”N 6°44’20”E	2011-05-30
RE 40	Taipei, TPE	25°07’20”N 121°28’30”E	2010-09-03

Table A.3: details of the evaluated RapidEye data for anomaly detection

List of Figures

List of Tables

List of Abbreviations

ALOS-3	Advanced Land Observing Satellite - 3
ASIC	Application Specific Integrated Circuit
BIRD	Bispectral InfraRed Detection
BLUE	Blue band
CREST	Centre for REsearch in Satellite Technologies
DLR	German Aerospace Center
DSM	Digital Surface Model
EDIF	Electronic Design Interchange Format
EO	Earth Observation
ESA	European Space Agency
FF	Flip-Flop
FPGA	Field Programmable Gate Array
GIMP	GNU Image Manipulation Program
GMRF	Gauss Markov Random Field
GREEN	Green band
GSD	Ground Sample Distance
HRC	High Resolution Camera
IDL	Interactive Data Language
JAXA	Japan Aerospace Exploration Agency
LUT	Look-Up Table
MS	Multi-Spectral
NASA	National Aeronautics and Space Administration
NIR	Near Infra-Red
PAN	Panchromatic
PC	Personal Computer
PCA	Principle Component Analysis
PPU	Parallel Processing Unit
PRISM-2	Panchromatic Remote-sensing Instrument for Stereo Mapping-2
PROBA-1	PRoject for On-Board Autonomy-1
RAM	Random Access Memory
RE	RapidEye
RED	Red band
RESA	RapidEye Science Archive
RGB	Red, Green, Blue
ROM	Read Only Memory
SVM	Support Vector Machine

Bibliography

Abraham, L. and Sasikumar, M. (2012). A fuzzy based automatic bridge detection technique for satellite images. In Venugopal, K. and Patnaik, L., editors, *Wireless Networks and Computational Intelligence*, volume 292 of *Communications in Computer and Information Science*, pages 213–219. Springer Berlin Heidelberg.

ALOS-3 (2013). The ALOS-3 mission. See www.eoportal.org.

Alston, I. and Madahar, B. (2002). From C to netlists: hardware engineering for software engineers? *Electronics & Communication Engineering Journal*, 14(4):165–173.

Appiah, K., Hunter, A., Dickinson, P., and Owens, J. (2008). A run-length based connected component algorithm for FPGA implementation. In *ICECE Technology, 2008. FPT 2008. International Conference on*, pages 177–184. IEEE.

Bailey, D. G. and Johnston, C. T. (2010). Algorithm transformation for FPGA implementation. In *Electronic Design, Test and Application, 2010. DELTA'10. Fifth IEEE International Symposium on*, pages 77–81. IEEE.

Bermyn, J. (2000). Proba-project for on-board autonomy. *Air & Space Europe*, 2(1):70–76.

Beulig, S., Schönermark, M., and Huber, F. (2013). An FPGA implemented bridge over water recognition for an image evaluation on-board of satellites. In *SPIE Optical Engineering+ Applications*, pages 88710I–88710I. International Society for Optics and Photonics.

Beulig, S., von Schönermark, M., and Huber, F. (2012a). A FPGA-based automatic bridge over water recognition in high-resolution satellite images. In *SPIE Remote Sensing*, pages 853713–853713. International Society for Optics and Photonics.

Beulig, S., von Schönermark, M., and Huber, F. (2012b). Mustererkennung in Echtzeit anhand hochaufgelöster multispektraler Satellitenbilder. In *RapidEye Science Archive (RESA) - Vom Algorithmus zum Produkt.*

Blue-Bridge (2013). The Blue Bridge in St. Petersburg. See https://en.wikipedia.org/wiki/Blue_Bridge_(Saint_Petersburg).

Bowen, M. (1999). Handel-C Language Reference Manual. *Embedded Solutions Ltd*, Version 2.1.

Bretschneider, T., Ramesh, B., Gupta, V., and McLoughlin, I. (2004). Low-cost space-borne processing on a reconfigurable parallel architecture. In *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms*, pages 93–99.

Breuel, T. (2008). Binary Morphology and Related Operations on Run-Length Representations. In *VISAPP*, pages 159–166.

Camarero, R., Thiebaut, C., Dejean, P., and Speciel, A. (2010). CNES studies for on-board implementation via HLS tools of a cloud-detection module for selective compression. In *SPIE Optical Engineering+ Applications*, pages 781004–781004. International Society for Optics and Photonics.

Chen, A. (2008). Novel approach for recognizing bridges over water in large remote sensing images. In *Computational Intelligence and Design, 2008. ISCID'08. International Symposium on*, volume 2, pages 47–51. IEEE.

Chien, S., Sherwood, R., Tran, D., Castano, R., Cichy, B., Davies, A., Rabideau, G., Tang, N., Burl, M., Mandl, D., et al. (2003). Autonomous science on the EO-1 mission.

Corbane, C., Najman, L., Pecoul, E., Demagistri, L., and Petit, M. (2010). A complete processing chain for ship detection using optical satellite imagery. *International Journal of Remote Sensing*, 31(22):5837–5854.

Danzeglocke, J. (2013). Flutkatastrophe. *DLR Newsletter Countdown*, 22:16–19.

Dawood, A., Visser, S., and Williams, J. (2002a). Reconfigurable FPGAs for real time image processing in space. In *Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on*, volume 2, pages 845–848. IEEE.

Dawood, A., Williams, J., and Visser, S. (2002b). On-board satellite image compression using reconfigurable FPGAs. In *Field-Programmable Technology, 2002. (FPT). Proceedings. 2002 IEEE International Conference on*, pages 306 – 310.

Digilent (2011). *Digilent Nexys-2 Board Reference Manual*. Digilent, Inc.

Fu, Y., Xing, K., Huang, Y., and Xiao, Y. (2009). Recognition of bridge over water in high-resolution remote sensing images. In *Computer Science and Information Engineering, 2009 WRI World Congress on*, volume 2, pages 621–625. IEEE.

Gao, F., Hu, L., and He, Z. (2011). Bridge extraction based on on constrained delaunay triangulation for panchromatic image. In *Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International*, pages 1429 –1432.

Gedik, E., Cinar, U., Karaman, E., Yardimci, Y., Halici, U., and Pakin, K. (2012). A new robust method for bridge detection from high resolution electro-optic satellite images. *Proc of the 4th GEOBIA*, 298.

GIMP (2013). The GNU Image Manipulation Program. See www.gimp.org.

Goto, K., Chague-Goff, C., Fujino, S., Goff, J., Jaffe, B., Nishimura, Y., Richmond, B., Sugawara, D., Szczucinski, W., Tappin, D., Witter, R., and Yulianto, E. (2011). New insights of tsunami hazard from the 2011 tohoku-oki event. *Marine Geology*, 290:46 – 50.

Gu, D., Zhu, C., Shen, H., Hu, J., and Chang, H. (2011). Automatic bridge extraction for optical images. In *Image and Graphics (ICIG), 2011 Sixth International Conference on*, pages 446–451. IEEE.

Halle, W., Brie, K., Schlicker, M., Skrbek, W., and Venus, H. (2002). Autonomous onboard classification experiment for the satellite BIRD. *International archives of photogrammetry, remote sensing and spatial information sciences*, 34(1):63–68.

Han, Y., Zheng, H., Cao, Q., and Wang, Y. (2007). An effective method for bridge detection from satellite imagery. In *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pages 2753–2757. IEEE.

Hauck, S. and DeHon, A. (2007). *Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Ip, F., Dohm, J., Baker, V., Doggett, T., Davies, A., Castano, R., Chien, S., Cichy, B., Greeley, R., Sherwood, R., et al. (2006). Flood detection and monitoring with the autonomous sciencecraft experiment onboard EO-1. *Remote Sensing of Environment*, 101(4):463–481.

Johnston, C., Gribbon, K., and Bailey, D. (2004). Implementing image processing algorithms on FPGAs. In *Proceedings of the Eleventh Electronics New Zealand Conference, ENZCon'04*, pages 118–123.

Karaman, E., Çinar, U., Gedik, E., Yardimci, Y., and Halici, U. (2012). Automatic road network extraction from multispectral satellite images. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pages 1–4. IEEE.

Kim, J. and Sukkarieh, S. (2004). Autonomous airborne navigation in unknown terrain environments. *Aerospace and Electronic Systems, IEEE Transactions on*, 40(3):1031–1045.

Larsen, S., Koren, H., and Solberg, R. (2009). Traffic monitoring using very high resolution satellite imagery. *Photogrammetric Engineering & Remote Sensing*, 75(7):859–869.

Leitloff, J., Hinz, S., and Stilla, U. (2010). Vehicle detection in very high resolution satellite images of city areas. *IEEE Transactions on geoscience and remote sensing*, 48(7):2795–2806.

Ley, W., Wittmann, K., and Hallmann, W. (2011). *Handbuch der Raumfahrttechnik*. Carl Hanser Verlag München.

Liu, J., Currit, N., and Meng, X. (2010). Extraction of water bodies from remotely sensed images. In *Intelligent Signal Processing and Communication Systems (ISPACS), 2010 International Symposium on*, pages 1–4. IEEE.

Liu, W., Yamazaki, F., and Vu, T. (2011). Automated vehicle extraction and speed determination from quickbird satellite images. *IEEE Journal of selected topics in applied earth observations and remote sensing*, 4(1):75–82.

Loménie, N., Barbeau, J., and Trias-Sanz, R. (2003). Integrating textural and geometric information for an automatic bridge detection system. In *Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings. 2003 IEEE International*, volume 6, pages 3952–3954. IEEE.

Luo, J., Ming, D., Liu, W., Shen, Z., Wang, M., and Sheng, H. (2007). Extraction of bridges over water from ikonos panchromatic data. *International Journal of Remote Sensing*, 28(16):3633–3648.

McLoughlin, I. V. and Bretschneider, T. R. (2010). Reliability through redundant parallelism for micro-satellite computing. *ACM Transactions on Embedded Computing Systems (TECS)*, 9(3):26.

Mittelbach, E. (2013). Satellitenbilder für den Notfall. *DLR Magazin*, 137:11–15.

Mori, N., Takahashi, T., Yasuda, T., and Yanagisawa, H. (2011). Survey of 2011 tohoku earthquake tsunami inundation and run-up. *Geophysical Research Letters*, 38(7).

Müller, T., Käser, H., Gübeli, R., and Klaus, R. (2005). *Technische Informatik 1.* vdf Hochschulverlag.

Nexys-2 (2013). The Nexys-2 development board. See www.digilentinc.com.

Pedrino, E., Saito, J., Senger, H., Roda, V., and Marinho, M. (2010). An fpga-based region-growing architecture for binary images. *17th International Conference on Systems, Signals and Image Processing.*

Pesaresi, M., Gutjahr, K., and Pagot, E. (2008). Estimating the velocity and direction of moving targets using a single optical vhr satellite sensor image. *International Journal of Remote Sensing*, 29(4):1221–1228.

PROBA-1 (2013). The PROBA-1 mission. See www.eoportal.org.

RapidEye (2012). *Satellite Imagery Product Specifications.* RapidEye AG, Germany. Version 3.3.

Richards, J. (2012). *Remote Sensing Digital Image Analysis: An Introduction.* Springer.

Salehi, B., Zhang, Y., and Zhong, M. (2012). Automatic moving vehicles information extraction from single-pass worldview-2 imagery. *IEEE Journal of selected topics in applied earth observations and remote sensing*, 5(1):135–145.

Saur, G., Estable, S., Zielinski, K., Knabe, S., Teutsch, M., and Gabel, M. (2011). Detection and classification of man-made offshore objects in TerraSAR-X and RapidEye imagery: Selected results of the DeMarine-DEKO project. In *OCEANS, 2011 IEEE-Spain*, pages 1–10. IEEE.

Scholz, P. (2005). *Softwareentwicklung eingebetteter Systeme: Grundlagen, Modellierung, Qualitätssicherung*. Springer.

Suzuki, S., Kankaku, Y., Imai, H., and Osawa, Y. (2012). Overview of ALOS-2 and ALOS-3. In *SPIE Asia-Pacific Remote Sensing*, pages 852811–852811. International Society for Optics and Photonics.

Trias-Sanz, R. and Loménie, N. (2003). Automatic bridge detection in high-resolution satellite images. *Computer Vision Systems*, pages 172–181.

X-Sat (2013). The X-Sat mission. See www.eoportal.org.

Xiong, Z. and Zhang, Y. (2008). An initial study on vehicle information extraction from single pass quickbird satellite imagery. *Photogrammetric Engineering & Remote Sensing*, 74(11):1401–1411.

Yuhaniz, S. and Vladimirova, T. (2009). An onboard automatic change detection system for disaster monitoring. *International Journal of Remote Sensing*, 30(23):6121–6139.

Zhang, J. and Sun, G. (2011). Recognition of bridge over water in remote sensing image using discrete hopfield neural network. In *Transportation, Mechanical, and Electrical Engineering (TMEE), 2011 International Conference on*, pages 439 –442.

Zhou, G., Baysal, O., Kaye, J., Habib, S., and Wang, C. (2004). Concept design of future intelligent earth observing satellites. *International Journal of Remote Sensing*, 25(14):2667–2685.

Zhu, C., Zhou, H., Wang, R., and Guo, J. (2010). A novel hierarchical method of ship detection from spaceborne optical image based on shape and texture features. *Geoscience and Remote Sensing, IEEE Transactions on*, 48(9):3446–3456.

Acknowledgements

During the last 4.5 years I had the great opportunity to realise my PhD thesis at the German Aerospace Center (DLR), Oberpfaffenhofen. I am grateful for the support and advice of numerous people, who have made this thesis become reality. In particular, I would like to thank:

At first, Prof. Dr.-Ing. Felix Huber for giving me the opportunity to become a PhD candidate at DLR. I was able to work on this thesis in an inspiring and fascinating environment. He provided a motivating atmosphere and supervised my research offering valuable feedback and advice.

Then, Prof. Dr.-Ing. Helmut Mayer for being the second reviewer of my thesis and his support of my work with valuable feedback, comments and advice. Furthermore, I would like to thank Prof. Dr. Christian Kähler for the organisation and the conduct of the defence of my thesis.

Then, my supervisor Dr. Maria von Schönermark for giving my ideas room, so that I could investigate and realise a lot of own thoughts and concepts in my work.

Then, Prof. Dr.-Ing. Roger Förstner and his group at Bundeswehr University Munich for the pleasant and motivating seminars and the lively exchange of ideas.

Furthermore, I would like to thank the DLR_Graduate_Program for offering a variety of superb seminars and the possibility to exchange experience with other PhD students at different DLR sites/ institutes.

I also want to thank the RB-RFT group for the pleasant environment. Additional thanks to Dr.-Ing. Toralf Boge, Heike Benninghoff and Florian Rems for the welcoming atmosphere. And thanks to Florian Paul for his help and the stimulating discussions.

Moreover, I want to thank my colleagues Katharina Götz, Kurt Schwenk and Matthias Eder. You enriched my work with fruitful discussions and new ideas and I am grateful to have worked with you.

Last but not least, I would like to thank my friend Dr.-Ing. Andreas Fieselmann, my girlfriend Ina Kutscher and of course my family for their great patience, support and encouragement throughout all these years.

Thank you!

Sebastian Beulig