Registration and Content Recognition of Nature Pictures

Eleni Kanellidi

Department of Electronic and Computer Engineering

Technical University of Crete

Dissertation Thesis Committee:

Stavros Christodoulakis, Professor (Supervisor)

Michael Zervakis, Professor

Euripides Petrakis, Associate Professor

Chania 2011

To my family

ACKNOWLEDGEMENTS

I owe my gratitude to all those people who have made this dissertation possible and because of whom my graduate experience has been one that I will cherish forever.

I would like to express my deepest gratitude to my advisor, Professor Stavros Christodoulakis, for his excellent guidance, caring, patience, and providing me with an excellent atmosphere for doing research. I would also like to thank him for giving me the opportunity to work on this very interesting field of research.

Also, I would like to thank Professor Michael Zervakis and Professor Euripides Petrakis, who agreed to participate in my dissertation committee and evaluate my diploma thesis.

My thanks must go also to Michael Foukarakis. Without his help and support this thesis would not have been completed. I also thank Chrisa Tsinaraki for helping me and providing an experienced ear for all my anxieties about writing a thesis.

I would like to thank Ioannis K., who as a good friend, was always willing to help and give his best suggestions. It would have been a lonely lab without him. Many thanks also to Manolis M. and Giorgos S. for offering me not only a position to their office but also a friendly environment to work in.

The days would have passed far more slowly without the support of my friends, Xenia, Giorgos and Alex. I thank them all for putting up with my idiosyncrasies and for providing such a rich source of conversation, education and entertainment. Special thanks to my friend Xenia for understanding me better than anyone and for being always willing to help me. Many thanks also to my friends Sofia and Giannis for their support and care. I deeply appreciate their belief in me.

And finally never enough thanks to my parents and my brother for their love and care. I thank them for reliably providing all kinds of support during all of my academic education. Without them I would never have enjoyed so many opportunities.

ABSTRACT

This thesis is based on the SPIM framework and introduces the SPIM+ framework. The SPIM framework is used for semantic spatial information processing. The framework exploits the modern digital camera's potential for capturing contextual parameters through the use of sophisticated sensor devices, information found in specially annotated semantic maps and industrial standards. It processes images with embedded positional and directional information. The objective is to effectively manage and associate the information and semantic objects contained in both the semantic maps and the images. In addition, the framework employs the use of image processing and other algorithms to enable the automatic annotation of the images.

Although SPIM achieves its objectives with considerable success, there are still some aspects that can be improved. SPIM+ is based on the SPIM implementation and uses its principal components in order to add functionality to the initial system and improve it.

The SPIM framework registers images of nature with the model which is produced by the 3D land maps. A number of experiments under various conditions (time of day, season, weather condition, *etc.*) were performed and it was observed that on many occasions, the SPIM failed to registered the images correctly. The aim was to improve its performance. To do that not only were improvements made to the original algorithm but also alternative detection objectives (shore lines, island lines *etc.*) were employed, and the position of the sun was exploited by taking into account the time and season. The results were incorporated into the SPIM+ main algorithm. The SPIM+ main algorithm was then tested in a new set of images and the quality of its results was evaluated and compared with the original SPIM algorithm. SPIM+ significantly improves the registration performance of the original algorithm.

Finally, the initial system was extended in order to process images which have only positional and no directional information embedded in them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Nowadays, a digital camera is regarded as a common good. This is evidenced by the fact that most of us own a digital camera and even if not, we do own a mobile phone with an integrated digital camera. Apparently, the use of digital cameras is widespread and since the task of capturing, storing and sharing digital images has been simplified, the size of the personal photo collections of the users has been growing dramatically. This increase has made quite urgent the need to better organize, annotate, index and browse the photos.

Digital cameras not only have been popular; they have also evolved. Their capabilities are no longer restricted to just capturing digital media. The technological advancements related to the digital cameras have provided them with effective features. Digital cameras are able to interoperate with internal or external sensors and integrate the additional information in the digital media. Thus, the digital cameras acquire additional capabilities for capturing and understanding the image taking context. For this thesis the context of the images includes geographical features such as mountains, sea areas, islands etc.

To take advantage of these new capabilities and manage the ever-increasing media content in an efficient way, the SPIM (Spatial Image Management) Framework has already been implemented. The framework exploits the potential of the digital cameras for capturing contextual parameters through the use of sensor devices, information found in specially annotated semantic maps and industrial standards. The aim is to manage and associate the information and semantic objects which are included in both the semantic maps and the pictures. Moreover, the framework uses image processing and other algorithms to enable the automatic annotation of the photos.

Briefly, the SPIM implementation provides a picture database that allows users to store and view their photos. Along with the photos, the users may view personalized semantic maps, annotated with semantic objects. Photos with position and direction information can be visualized on top of maps and be associated with semantic objects. Finally, the implementation allows interactive exploration of the picture contents.

Although SPIM achieves its objectives with considerable success, there are still some aspects that can be improved. This thesis introduces the SPIM+ Framework. SPIM+ is based on the SPIM implementation and uses its principal components in order to add functionality to the initial system and ameliorate it.

The SPIM Framework registers the nature images with the model which produced by the 3D land maps. The transformation that produces and applies to the model image in order to fit the reference photo (user photo) properly is generated only by the matching of ridgelines. We performed a number of experiments under various conditions (time of day, season, weather condition, etc) and we observed that in many occasions the algorithm failed to registered the pictures correctly. Our aim was to improve the original algorithm. To do that we experimented with improvements on the original algorithm but also with alternative detection objectives (shore lines, island lines etc), as well as exploiting the sun's position by taking into account the time and season. Our results were incorporated into the SPIM+ algorithm. We then tested the SPIM+ algorithm in a new set of pictures and we evaluate the quality of its results. We compared the results with the old SPIM algorithm. SPIM+ improves significantly the registration performance of the original algorithm.

1.1 THE SPIM FRAMEWORK

The SPIM Framework transforms images into interactive windows to the outside world. Its result is a set of images that allow the interactive exploration of their contents.

The procedure for converting an image into an interactive one is divided into three phases: *(a) Viewshed Calculation*; *(b) Image Segmentation*; and *(c) Image Registration* [1].

In the first phase (*Viewshed Calculation*), the visible areas and the semantic objects are determined from the spatial view of an image, utilizing the elevation data and the position and direction parameters of the image. The result of the algorithm is a 2D representation (model) of the landscape and the individuals that are included in it, from the point of view and the direction of the image.

The second phase (*Image Segmentation*) uses a segmentation algorithm and partitions the image into disjoint and homogeneous regions. Afterwards, additional heuristic algorithms are set off in order to detect the regions that correspond to specific geographic individuals and features.

The third phase (*Image Registration*) of the procedure produces the final result. In the previous phases, the ridgelines have been detected and this phase takes them as input. The algorithm replaces the sequences of the points representing the ridgelines, from both the image and the 2D model with a connected series of line segments. Then it computes the optimal matching between the line segments from the model and the image and generates the transformation which is applied to the model so that the model successfully registers with the image.

Superimposing the transformed model onto the image places the individuals of the model at their proper position in the image. Then, the user may select them and see the relevant information just like interacting with a geographic semantic map.

Figure 1 depicts the aforementioned procedure which is followed in order to convert a photo into a semantic map. Here the ridgelines, namely Mountain Lines, are extracted from the model and the image and are used as inputs in the matching algorithm. The numbers 1, 2, 3 refer to the three phases of the procedure, Viewshed Calculation, Image Segmentation and Image Registration correspondingly.

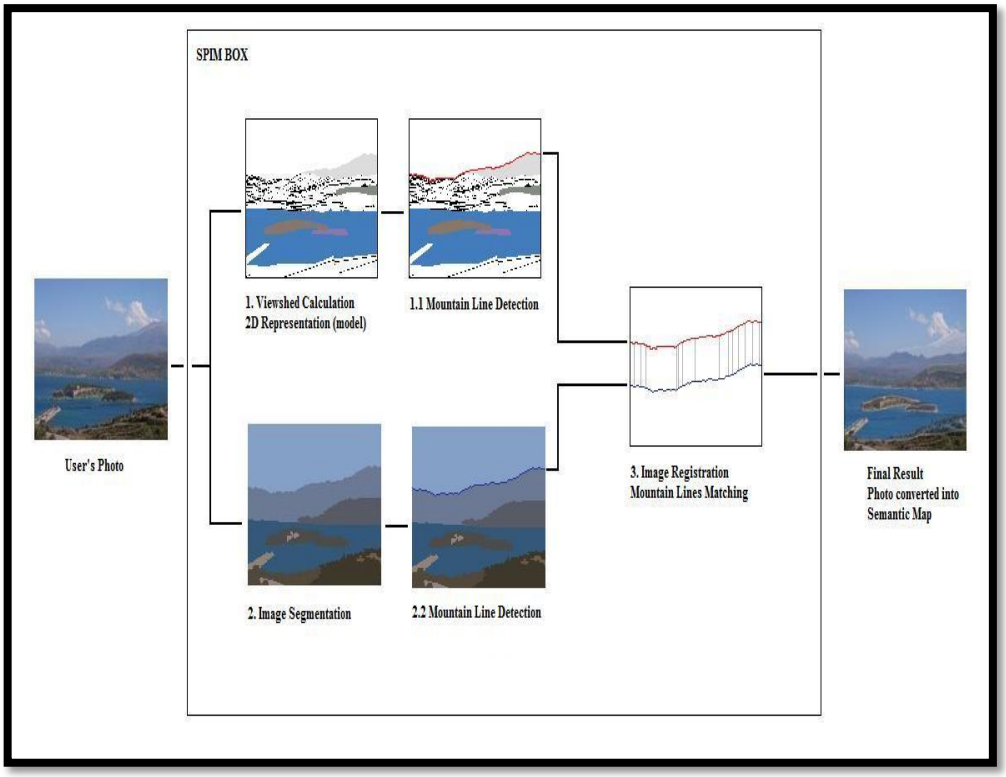


FIGURE 1: PHOTO TO SEMANTIC MAP CONVERSION

1.2 THE SPIM+ FRAMEWORK

SPIM+ Objectives

SPIM + aims not only to improve the performance of the SPIM Framework, but also to add functionality to it.

Regarding the SPIM Framework performance improvement, the SPIM+ Framework provides the following enhancements:

- It generates more data inputs (lines as coastline, after the sea coastline, island line) for the image registration algorithm.
- It provides the image registration algorithm with the most reliable and consistent data inputs (lines) among the available ones.
- It adjusts the sensitivity of the image registration algorithm.

The addition of functionality in the SPIM+ Framework includes:

- The calculation and exploitation of the Sun Position at the time the photos were taken.
- Support for inexpensive cameras (cameras with only a GPS receiver and no Compass).

SPIM+ Implementation

Several algorithms have been implemented in order to achieve the SPIM+ objectives, including:

- Algorithms for detecting and extracting the boundaries of any individual included in the 2D-model image of a photo, not only from the 2D-model image but also from a segmentation image have been developed. These are the algorithms which generate more data inputs (lines) for the image registration algorithm.
- An algorithm for the calculation of the Sun Position. The algorithm takes as inputs the date and the time that a photo was taken as well as the location from which it was taken and it calculates the position of the Sun at the given date-time and location. The outcome is then used for further calculations in order to understand better the image context.
- An algorithm that incorporates the conclusions reached by all the experiments we conducted in order to improve the performance of the original SPIM algorithm. SPIM+ algorithm, compared to the SPIM original algorithm, manages to register a wider range of nature images.
- An algorithm that supports inexpensive cameras has been developed. It designs an image that depicts what we see from the given location (GPS data) by looking simultaneously towards all the possible directions and then it finds through several processes which part of this panoramic image is depicted in the current image.

1.3 THESIS CONTRIBUTION

The SPIM+ Framework is not only an improved version but also an extension of SPIM.

As an improved version, SPIM+ allows transforming a wider range of nature images to interactive maps. Images with more indistinguishable content due to heavy clouds in the sky, fog in the atmosphere, high humidity or strong solar reflection, now can be processed and successfully transformed to interactive ones. All the SPIM+ algorithms have been used in order to extract from these “difficult” images as much consistent and reliable data as possible and provide the algorithms creating interactive images with them as input.

As an extension, SPIM+ ensures the conversion of nature images without direction information to interactive ones. This contribution is of great importance. Smart phones have already flooded the market and a typical capability of theirs is the production of geotagged images. In general, these are images with position information and without direction information. Since SPIM+ manages to process these images and transform them to interactive ones, it becomes a useful tool for the wider consumer public.

1.4 THESIS STRUCTURE

The rest of the thesis is organized as follows. Chapter 2 presents the related to this thesis research and chapter 3 presents the related technologies. The calculation of the Sun position and the significance of this information are described in detail in Chapter 4. Chapter 5 contains the algorithms used for generating more data inputs for the image registration algorithms. Chapter 6 describes the image registration algorithm. The procedure followed for supporting inexpensive cameras is explained in Chapter 7. Chapter 8 shows the experiments testing the performance of the SPIM original algorithm and introduces us the SPIM+ Main algorithm. Chapter 9 presents the evaluation results of the SPIM+ algorithms. Finally, the thesis conclusions as well as the future research issues are discussed in Chapter 10.

2. RELATED RESEARCH

This thesis expands upon various fields of research which will be described in this chapter. In particular, section 2.1 provides a short background on image annotation categories, section 2.2 deals with general purpose image annotation as well as annotations for personal photography and section 2.3 presents some map-based applications which are also relevant to this thesis. Finally, section 2.4 presents relevant to the light information.

2.1 IMAGE ANNOTATION CATEGORIES

The first major topic for this thesis is image annotation. Image annotation and metadata creation are essential for managing the increasing amount of personal collections of images and facilitating image retrieval. Before listing the related work, the main methods of image annotation will be presented, starting from low-level metadata and finishing with label (tag) annotations.

2.1.1 USING LOW-LEVEL METADATA

Low-level metadata are produced during the image taking procedure and include information about the camera (model, manufacturer), its status at the time of capturing the image (flash fired, focal length), the time and date, comments and more, including possibly the position and direction of the shooting.
The widely used standard describing these metadata is Exif[2]. Most modern digital cameras automatically store the image taking parameters in the Exif header of the digital image. The majority of the parameters provide just technical information, which is not directly useful to the users. Subsequently, processing the metadata in relation with other sources is essential in order to produce richer image annotations. More information about Exif and its parameters (also known as Exif tags) is presented in Chapter 3.
Risto Sarvas [3] addresses the usefulness problem of image capturing metadata and introduces the concept of social metadata suited for personal use. However, as seen from this thesis and other similar works, even raw image metadata proves to be rather important for personal photography use.

2.1.2 USING TAGS

A tag is a non-hierarchical keyword or term assigned to a piece of information (such as an internet bookmark, digital image, or computer file). This kind of metadata helps describe an item and allows it to be found again by browsing or searching. Tags are chosen informally and personally by the item's creator or by its viewer, depending on the system.
Assigning tags (also known as tagging) a picture is a very simple way of describing the picture's subject, location, persons and general content. Apart from simplicity, another advantage of using tags is its mainstream use in Web 2.0 and the usefulness of tag clouds in various websites such as Flickr[4]. Flickr has been widely used as a source of (possibly geo-coded) images for different kinds of research using tags and image metadata [5][6][7][8][9] [10]. The related research will be presented later in this chapter.

While using tags is flexible and easy, tagging is not without its drawbacks. Typically there is no information about the meaning or semantics of a tag. For example, the tag "apple" may refer to the fruit, Apple Inc., the Beatles' music label, or Gwyneth Paltrow's baby. This lack of semantic distinction in tags can lead to inappropriate connections between items. Additionally, selection of "tag terms" is highly individualistic. Different people use drastically different terms to describe the same concept: for example, items related to a version of Apple Computer's operating system might be tagged both "Mac OS X" and "Leopard", and possibly many other terms. Users of tagging systems must make judgments, based on the number of connections and the choices of "tag terms", whether possible connections between items are valid for their interests.

2.2 THE SEMANTIC GAP

A common problem concerning image retrieval is the so-called semantic gap. Smeulders et al. [11] describes the semantic gap as "the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation". As presented in Hare et al [12], there are two main methods of "attacking" the gap:

- Using the raw metadata and processing it to achieve at least identification of the objects in an image (bottom-up approach), usually in an automatic manner.
- Using ontologies as structured knowledge representations (top-down approach).

The bottom-up approach is closely related to automatic image annotation using raw metadata and image characteristics. The techniques for automatic image annotation include:

- Segmenting an image into regions and trying to annotate each relevant segment (spatial context).
- Utilizing global information for scene classification.
- Taking into account the image capture condition context to derive richer metadata and annotations. The contextual parameters can come from different sources (sensors) or can be derived from other parameters.
- Taking advantage of the information contained in neighboring images (temporal context).
- Exploring and exploiting the plethora of information contained in large collections of labeled (tagged) images.
- Using training sets of images to classify images based on similarity detection.
- A combination of the above.

The following two sections deal with the bottom-up approach, followed by a subsection describing top-down approaches using ontologies.

2.2.1 GENERAL PURPOSE AUTOMATIC IMAGE ANNOTATION

One of the first attempts on automatic image annotation was by Mori et al [13]. In this work, a co-occurrence model was applied to keywords and low-level features of rectangular image regions.
The following researchers focused primarily on image segmentation methods for automatic annotation. They are similar to a small part of this work that includes image segmentation for determining specific geographic and other features of the images. The main differences are that the annotation proposed by these authors is

usually just keyword-based and that most approaches are general and not focused on specific image categories or domains. Duygulu et al. [14] used a model inspired by machine translation to map between keyword annotations and a discrete vocabulary of clustered regions. The proposed data-set (Corel) has been widely used in annotation systems in the literature. Jeon et al. [15] demonstrated that probabilistic annotation provides better ranking in the results in contrast to hard annotations thus improving on the results of Duygulu et al. [14] by using a different relevance model. This, however, was outerperformed by the relevance model proposed by Lavrenko et al. [16] which built continuous probability density functions for describing the generating region features process. Hare et al. [17] also avoided hard annotations and proposed an approach where annotation is performed implicitly in a soft manner. An inference network approach to connect regions and their annotations has been used by Metzler and Manmatha [18]. The resulting model allows rich queries with structured operators and term weights to be evaluated for combinations of terms and images. Monay and Gatica-Perez [19] applied and compared two commonly used in text analysis and simple latent space models for image annotation, namely Latent Semantic Analysis (LSA) and Probabilistic LSA (PLSA) and discuss annotation strategies for each one. Feng et al. [20] proposed a relevance model based on a multiple Bernoulli distribution and used rectangular image regions. Jeon et al. [21] also used rectangular image regions; however their approach used Maximum Entropy, a statistical technique which allows one to predict the probability of a label given test data. Blei and Jordan [22] provided a clean probabilistic model based on Latent Dirichlet Allocation [23] which could generate keywords for images, image regions and facilitate text-based image retrieval. Barnard et al. [24] extended this work to multi-modal data.

Turning to scene-oriented approaches, Oliva and Torralba [25][26] used a selection of low-level global filters to apply basic scene annotations such as 'buildings' or 'streets' with success. They also showed [27] how the presence or absence of objects in a scene can be inferred by using simple image statistics. Using even simpler global features and the technique of kernel smoothing, Yavlinsky et al. [28] proposed a modeling framework based on nonparametric density estimation. The results shown by this research were comparable with the inference network [21] and the model by Lavrenko et al. [16]. Important work has also been done in the identification of specific parts of an image, such as the blue sky, by Galagher et al. [29] and Platt [30]. Real-time automatic annotation has been recently attempted by Li and Wang [7], using a training set of tagged images and a probabilistic method to assign tags to unseen images according to their similarity to the training set. Finally, Vailaya et al. [31] exploited domain semantics for the classification of pictures to broad categories like indoor and outdoor pictures and their subcategories. In contrast with these scene-oriented approaches, the work proposed in this thesis can accurately classify pictures containing mountainous landscape, bodies of water and other geographic features as long as the pictures contain various parameters such as position and direction.

2.2.2 Automatic Image Annotation for Personal Photography

In this subsection, research more closely related to personal photography will be presented. Recently there has been strong interest in semantically annotating and sharing personal photographic collections. Combining the context in which a photograph was captured with information from other external resources and services for the purpose of image annotation has been the subject of some projects. Most of them take advantage of location and time information in order to derive richer information. Other approaches, including this work, take advantage of more

parameters such as the camera direction. There is also related research examining the context around the user, using sources such as calendars and social networks.

The research by O'Hare et al. [32] combined context-based information and content based analysis to facilitate image classification and retrieval. In his thesis, Matthew Boutell [33] categorized the context related to images into spatial (e.g. sky, buildings etc. and their relations), temporal (relation between images in a collection – e.g. time elapsed between them) and image capture condition context (captured metadata). He proposed a graphical model for each category for the purpose of scene classification. The part of his work that is closer to this thesis is presented in Boutell et al. [34]. In it, the captured metadata are processed and used as classifiers for indoor/outdoor scene discrimination and sunset or manmade/natural scene detection. This work also takes advantage of the captured metadata (which are more rich thanks to the additional sensors) and uses them for image annotation.

Another interesting dissertation is by Mor Naaman [35]. Naaman did many studies using GPS, weather, and user-created tags for automatically organizing georeferenced photo collections and other tasks. Like this work, Naaman also emphasized the use of time and location metadata. The fact that position and temporal information can be used for inferring the picture contents is another similarity between the two efforts. The developed system (PhotoCompas) provided a textual interface for browsing and retrieval that categorized photo collections using information such as location, weather conditions, light status (day/night) and events. The description for each picture included an event/location pair. Events in particular are one of the major content categories of the proposed framework of this thesis. A major difference is that Naaman tried to avoid the use of geographic maps and focused only on textual descriptions of location information. He also did not take into account camera direction. Finally, the descriptions offered by PhotoCompas were textual, interactive exploration of the image contents was not available and ontology based annotations were not considered. However, the proposed system was solid and provided useful insight on the organization of photo collections into events using context-based information.

Sarvas et al. [36] proposed a metadata creation system for mobile images. This approach used a camera system connected to external sources (a remote server) and in that regard it is similar to this work. Another similarity is the context capturing process. The location of the mobile phone is provided by the GSM network characteristics (instead of GPS). The captured images were uploaded to the remote server which located similar photos taken at the same location and calculated the appropriate metadata for image annotation. The remote server compared the uploaded picture with other pictures that depicted the same object and were located in close proximity. It then created relevant metadata and sent them back to the mobile device. A significant downside was that the phone model was difficult to use and the connection to the remote server proved to be slow and frustrating. Another difference from this work is that the process was semi-automatic – it required user input on the validity of the proposed metadata. The next attempt to combine image capturing context with external sensors was by Volgin et al. [37], which correlated XML-based metadata with the images. Volgin et al. provided a more consistent sensor model for communication with smart phones. They used external environmental sensors known as motes [38] to capture context metadata and provide them to the mobile devices at the moment of image capture or at a later time. The sensor concept does not differ very much from the proposed model, however they did not use map information to annotate the image contents and camera direction was also not taken into account.

An interesting sensor that can be integrated into modern digital cameras and most researchers have ignored so far is the digital compass. An early attempt was by Smith et al. [39], where a digital camera was augmented with both position and direction sensors and the metadata captured was used to retrieve historical images from a database. There are many similarities between this attempt and the current

work. Smith et al. mainly focused on educational applications of the camera and database system, by encouraging students to take pictures of their local community. The naïve system acquired the position and direction parameters of the pictures taken by the camera and retrieved historical photos of the same places as the ones depicted in the pictures. The students could also manually write ontological descriptions for each picture according to their observations and the system was able to provide additional retrieval capabilities based on these annotations. The system had a simple map application for visualizing the local area and some places of interest. The algorithm for determining the contents of the picture was primitive and did not take into account the angle of view of the camera and the elevation of the area. This thesis expands upon the concept of using orientation information in addition to position and provides more focused visualizations and retrieval capabilities using semantic maps.

The metadata that users are mostly interested in revolve around places, people and events as emphasized by Risto Sarvas [3]. Research directed at this kind of "social" metadata has been performed by many researchers. Cao et al. [40] tried to annotate collections of images rather than single images, using hierarchical event and scene models, while Gallagher et al. [41] utilized user calendars as a source for annotations. Using a trained set of images containing events, a freely available GIS database and tags, Joshi et al. [8] proposed a method for inferring generic activities and events. Rattenbury et al. [9] attempted to automatically extract event and place semantics from a large collection of tagged images, based on tag usage distribution. Finally, Yu and Luo [10] experimented with probabilistic season and location context models to prove that even when the contextual data is not precise, accurate scene understanding can be inferred. Joshi et al., Rattenbury et al. and Yu and Luo all took advantage of the large image database and social photography site Flickr [4], which also includes geo-referenced images.

2.3 Map Applications and Photos

The second major topic for this thesis is (semantic) maps and visualizations on top of maps. One of the very early relevant applications in culture and tourism was by Christodoulakis et al. [42]. In the system proposed, a laptop/GPS combination allowed the user to find their position on top of a map and according to that, useful information about nearby points of interest was shown. Christodoulakis et al. [43] also developed tools that provide geographic information functionalities and utilize map-based information and combine the location and topological relations of spatial objects. The present thesis was based on many of the principles of these applications and tools. The second most relevant to this thesis work (mentioned before) was by Smith et al. [39], due to the integration of both GPS and digital compass in their camera system. The additional information provided by the digital compass greatly enhances the ability to determine the contents of a picture with assistance from map data.

The map system by Diomidis Spinellis [44] utilized the GPS track log and pictures and created trip log based map visualizations in a web page. The points where the pictures were taken were visible on a non-interactive map. A disadvantage of the system, other than the non-interactive maps, is that it does not support hierarchical navigation by event or location.

The Campiello project [45] aimed at using innovative communication and information technology for developing new relationships between local communities and visitors of historical cities of culture and art. The project's objectives were to connect local inhabitants of historical places better, to make them active participants in the construction of cultural information and to support new and improved connections with cultural managers and tourists. The system included a recommender module, a search module, and a shared data space. The similarity with

this work is that they both focus on personalization of the provided information. In Campiello, the users can view and submit comments on objects of cultural heritage. The feedback collected by the system enables it to provide personalized information to different users, according to criteria or proactively. In the system implemented in this work, personalization is achieved by selecting the desirable content (which can be from any domain, not just cultural heritage) and maps containing only objects of interest will be returned by the system.
Perhaps the most important contribution in organizing geo-referenced photos is attributed to Toyama et al. [46] and the World Wide Media eXchange (WWMX) project. Photos were indexed by the WWMX database according to time and location and the WWMX browser visualized them using a map interface and provided retrieval functionality. Different kinds of acquiring location tags, browsing images and visualizing them on a map were presented. The ability to put georeferenced photos on top of a map is a common characteristic of WWMX and this system. Toyama et al. [46] put a lot of effort in trying to collect a large amount of photos, finding the best methods for their visualization on top of maps and dealing with performance issues for large queries. This work focuses more on providing better annotations for the images using ontologies and takes advantage of more than location and time information to achieve this purpose. The ontological approach also allows for richer retrieval queries.

2.4 LIGHT

2.4.1 LIGHT DIRECTION

The direction from which we view a light source has a profound effect on our perception of it, and on how the objects in a scene will appear. Knowing which direction the main light is coming from is one of the most important information since it will have a great impact on how a scene will appear.

Front lighting:

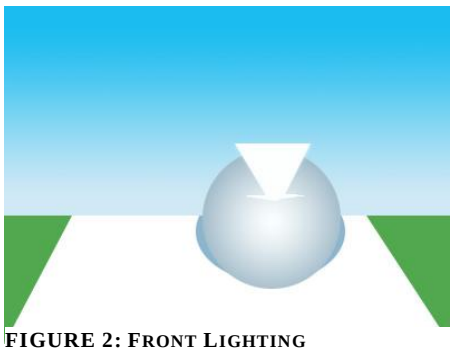


FIGURE 2: FRONT LIGHTING

This is the case when the light source is directly behind the viewer's point of view. It is most commonly seen in flash photography and is often fairly unappealing if the light source is hard. There are, though, exceptions and in some situations very attractive images result from soft frontal lighting.

Front lighting does little to reveal the form or texture since the shadows are mostly hidden from the view, as a result it can make things look flat.

Side Lighting:

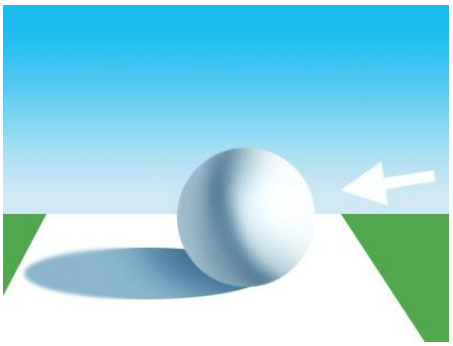


FIGURE 3: SIDE LIGHTING

Side lighting is very good for showing the form and texture and lends a three-dimensional quality to objects. As a result, the shadows are prominent and the contrast can be high. Side lighting is generally attractive and it is often used to great effect: it is the kind of lighting encountered in the beginning and the end of the day and as such it is often seen in photographs.

Potential drawbacks of using side lighting are that areas of the image can be lost in the shadow, and it can reveal imperfections.

Back lighting:

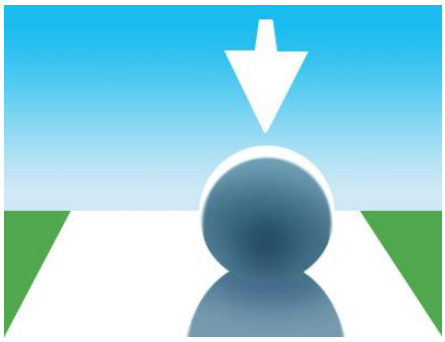


FIGURE 4: BACK LIGHTING

Back lighting is the case when the viewer is looking in the light source, and the objects have their lit sides facing away from us to appear either as silhouettes or darkly lit by the fill light. It is usually a high contrast situation. If the light source is at a slight angle relative to our point of view, the objects will have a rim of light defining one or more of their edges, the harder the light the more pronounced this rim will be. Backlit scenes usually contain a lot of shadow unless the light source is very soft. Most of the time the image will be predominantly dark with dramatic pools of light. The rim lighting that occurs in this situation can be very useful for defining forms among the shadows. Another feature of this kind of light is that it reveals transparency, translucency and any fine detail or texture along rim-lit edges.

Lighting from above:

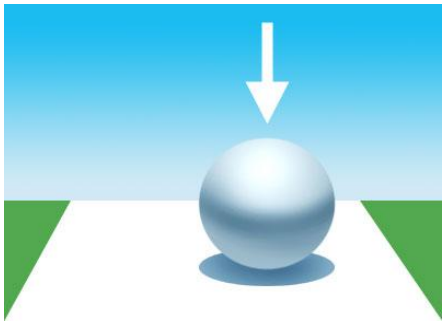


FIGURE 5: LIGHTING FROM ABOVE

Top lighting is a slightly more unusual situation, although it is common in overcast daylight. It can also be encountered in sunshine at midday. Under soft light it is an effective way of showing form. Under hard light it can lend an air of mystery by casting dramatic shadows which conceal most of the forms beneath them.

2.4.2 NATURAL LIGHT

The natural light comes in a wide range of different flavors, and the difference between them can be enormous. The source of all our natural light is the Sun, however it takes on different characteristics at different times of day and in different weather conditions, turning this source of light into essentially many different ones ranging from hard and warm to soft and cool.

The earth's atmosphere scatters the shorter wavelengths of the light; this has the effect of creating the blue sky and of reddening the light from the sun itself. The more air that the sunlight has to travel through, the more scattering occurs. This means that as the sun gets lower in the sky it has to travel through a thicker layer of atmosphere, thus causing more scattering at the beginning and the end of the day.

Obviously, this means that the sunlight has very a different character at different times of day. There are also the special conditions that occur when the sun is below the horizon, when skylight scattered from the sun is the only source of light.

The clouds also have a major impact on both the color and the character of sunlight. The clouds are translucent, which means that they let the light pass through them, but in a diffuse manner. When the light travels through a transparent surface such as glass, the rays remain parallel; however when the light travels through a translucent surface it is deflected by the substance and the rays bounce around inside it and emerge from it in several directions. This is a similar phenomenon to the scattering of blue light by the atmosphere, except that in the clouds it occurs across all the wavelengths of the light, not just the shorter ones.

The effect that this diffusion has on the sunlight is to soften it, turning a small hard light source (the sun) into a large and soft one (the whole sky). The color is also profoundly affected by the cloud cover, since the clouds conceal the blue sky and the light coming from it.

Midday sunshine:

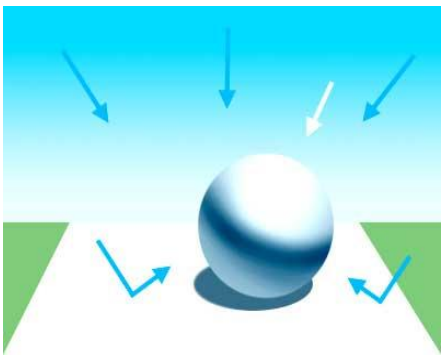


FIGURE 6: MIDDAY SUNSHINE

When the sun is at its highest point in the sky, the light is at its whitest and strongest. The contrast is very high, the shadows are very dark, so dark in fact that film emulsions generally render them black - although with the naked eye it can still be possible to see some detail in the shadows. For this kind of lighting to be believably recreated very strong and high contrast is needed.

The strong light has the effect of bleaching out the colors and these appear to be less saturated than at other times of day. The strong contrast can make it difficult to create appealing images in this sort of light; however, in situations where the contrast is naturally lower it can work very well. The water for example can really benefit from this strong light, and many images of tropical seas are taken at midday.

Late afternoon/early evening:

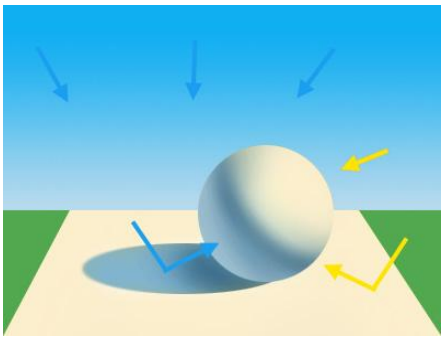


FIGURE 7: LATE AFTERNOON / EARLY EVENING

As the sun goes down, its light gets progressively warmer, so that by the evening the sunlight has very obvious yellow cast on it. The color of the sky also takes on a deeper shade of blue due to the decreasing light levels.

The evening light is generally considered to be very attractive, the warm colors and softer contrast are very easy on the eye. From about an hour before sunset this effect is at its most noticeable.

The color saturation at this time is very high and the color of the light itself has a huge effect on our perception of the surfaces it touches, lending them a warm and rich appearance. By an aesthetically pleasing coincidence the shadows are near to the complimentary color of the highlights (yellow against blue), and the main light is a warm yellow while the shadows are a cool blue.

Overcast:

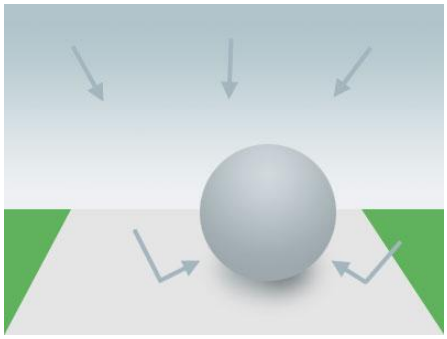


FIGURE 8: OVERCAST

Overcast light comes in a few varieties, depending on the thickness of the cloud and the time of day. Contrary to popular opinion, it can actually be quite beautiful and it does have quite a few attractive qualities. Since the whole sky is acting as one light source, the light is soft and diffused, with very soft shadows. The contrast is low and the color saturation is usually quite high.

The color is dependent mostly on the time of day. If the sun is high, the light appears to be white or grey, and the thicker the cloud the whiter the light. It is only when the sun gets lower in the sky that the overcast light becomes bluer, and the lower the sun goes the more obvious this becomes.

Bright overcast:

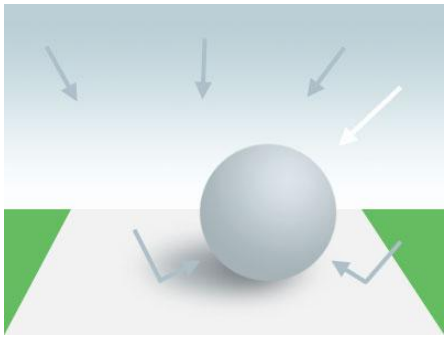


FIGURE 9: BRIGHT OVERCAST

On days with thinner cloud it is possible to get a little directional sunlight coming through that creates stronger shadows which can still be soft as long as there is cloud in front of the sun. Bright overcast is an almost ideal compromise between the strong contrast of the sunshine and the relative dullness of a heavy cloud.

On days with thinner cloud the sky can have a lot of texture, whereas on days with heavy cloud it tends to look a solid white or grey. Varying cloud thickness or small gaps between the clouds can also help to introduce color into the sky, with blue skylight and yellow sunlight reflecting onto the surface of the clouds. Colors in the sky can vary enormously when cloud is thinner, and the sky can often be very striking with thin or broken cloud. Another factor influencing the cloud color is that distant clouds can appear yellow or even orange because of light scattering, even in the middle of the day.

Broken cloud, stormy light and dappled light:

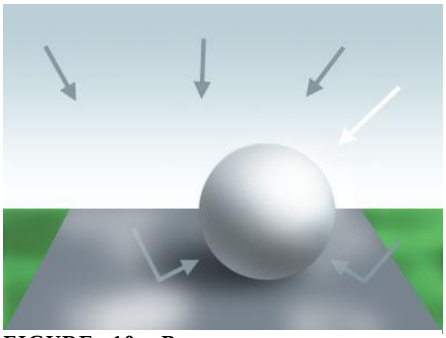


FIGURE 10: BROKEN CLOUD, STORMY LIGHT, DAPPLED LIGHT

It is also quite common to come across mixtures of light and shade in natural environments.

With broken cloud you get a different sort of light to pure sunshine or overcast because the blue fill light from the sky is absent; yet the sun can shine brightly if there is a gap in the cloud.

Clouds will cast visible shadows on the landscape and there will be patches of sunlight in between these shadows.

2.4.3 SKY AND WATER COLORS

Color in the sky

The sky is often very colorful, it can produce amazing and complex ranges of different colors. In addition to the time of day and the cloud cover, the thickness of the clouds is important as well as the space between them. If the cloud is of uneven thickness, or if there are small gaps in between closely spaced clouds you will get a variation in the amount of the color and the quality of the light in the sky. This creates texture and a great deal of unpredictable variation.

The natural light and the sky in particular, almost always have some color, even on the bleakest day. The sky is a constant diffuse light source during the day, no matter how bright or dim the sun is.

Water

The water also plays a big part on how the natural light interacts with the world around us, being a common feature of the landscape in the form of rain and dew, or lakes, rivers and the sea.

Water changes the surfaces that are wet because unlike most natural substances it is highly reflective and causes strong directional highlights. Dew in grass, for instance, can cause thousands of little highlights as it catches the morning sun, with each drop acting as a lens. Specular reflections are comparatively rare in nature, unless water is present, and so we can instantly recognize when surfaces are wet. Like volume in the air, water can be very atmospheric.

Another major effect of the water on light is that it reflects the light back up into the landscape; as a consequence, if we are by the sea we will have a lot more light reflecting on us because of this. [47]

3. RELATED TECHNOLOGIES

This chapter will briefly present the most significant technologies and standards used in, or useful to the research concerning the thesis.

3.1 MODERN DIGITAL CAMERAS AND THEIR CAPABILITIES

Considering the transition from film-based cameras to digital cameras, the domain of photography has evolved regarding the following features:

- Storage

  Most digital cameras utilize some form of removable storage to store image data. While the vast majority of the storage media types are some form of memory card using flash memory (CompactFlash, SD, etc.) there have been storage methods that used other technologies such as Microdrives (very small hard disk drives), CD single (185 MB), and 3.5" floppy disks. The flash memory cards have a capacity of up to 128GB with a theoretical maximum of 2TB for the current storage formats. This advancement, along with the fact that the users can freely transfer the photos to their computers and take new pictures, has led to the tremendous increase in the personal user collections, thus enhancing the need for research on image annotation, indexing and retrieval.

- Resolution

  The latest advancements in technology have enabled very high resolutions in the digital images and the resolution values follow Moore's Law. The common aspect ratios for the produced images are 4:3 and 3:2, and this reduces the number of useful image sensor sizes. The camera's maximum resolution is measured in megapixels, which are computed by calculating the pixels comprising the produced image (for example an image with resolution 1,600 x 1,200 = 1,920,000 is produced by a "2 megapixel" camera).

- Image File Format

  The three dominant image file formats for digital cameras are RAW, TIFF and JPEG. A raw image is the unprocessed set of pixel data directly from the camera's sensor, while TIFF and JPEG are compression formats (lossless and lossy respectively). The latter is the most commonly used by modern digital cameras. It contains various metadata in its header. The metadata standard used for describing the image capture and other parameters is Exif [2]. More information about Exif and its metadata tags is presented in section 2.2.3

- Wireless Connectivity

  Since the majority of portable devices have already begun integrating wireless connection technology, digital cameras are also following this trend. Bluetooth™ enables easy connectivity with other sensors such as GPS and it also allows sending the photos taken to a nearby device. Some digital cameras also feature a Wi-Fi radio for the same purposes. It the future it is expected that these

technologies will allow the cameras to communicate with other external resources and acquire essential information about the image taking context, before and after the image capture and without the help of other computers or PDAs.

- Sensor Integration

  Digital cameras can now interoperate with external sensors using wireless technology. Since the Exif metadata standard contains location tags, pairing a camera with a GPS receiver results in recording automatically the location information in the Exif header of the image. There are a few such receivers that also integrate a digital compass, allowing for direction information to be recorded additionally. Finally, laser rangefinders can be used to calculate and store the subject distance for the photos. For the experiments conducted for this thesis, a camera system utilizing an external compass-enabled GPS receiver with a compatible digital camera has been used. Since then, these sensors have already been integrated internally in some cameras and in some mobile phones.

3.2 GPS AND NMEA0183

NMEA 0183 (or NMEA for short) is a combined electrical and data specification for communication between marine electronic devices such as echo sounder, sonars, anemometer (wind speed and direction), gyrocompass, autopilot, GPS receivers and many other types of instruments. It has been defined and is controlled by the U.S.-based National Marine Electronics Association [48]. Its communication protocol is text-based.

A GPS receiver transmits the satellite data in forms of NMEA sentences, short text strings that contain information such as position, speed, etc. The most important NMEA sentences include the GGA (Global Positioning System Fix Data) which provides the current satellite fix data, the RMC (Recommend Minimum Specific GPS/TRANSIT Data) which provides the minimum GPS sentence information, and the GSA (GPS Dilution of Precision and Active Satellites) which provides the Satellite status data. GPS receivers with integrated digital compass also output the HCHDG sentence, containing heading/direction information.

For the image capturing context, NMEA sentences provide the following useful information:

- GGA sentence: latitude, longitude, satellite fix quality, number of satellites being tracked, altitude.
- GSA sentence: satellite information, satellite fix type (no fix, 2D, 3D), dilution of precision.
- RMC sentence: time of the fix, latitude, longitude, speed over the ground in knots, track angle in degrees, date, magnetic variation.
- HCHDG sentence: magnetic heading, magnetic variation.

While the RMC sentence provides the most relevant position parameters, altitude can only be obtained from the GGA sentence. When using a digital camera integrated with a GPS receiver, almost all the information contained in the transmitted sentences is automatically embedded within the image.

3.3 EXCHANGEABLE IMAGE FILE FORMAT (EXIF)

Exif is a specification for the image file format used by digital cameras. Even though the camera manufacturers universally use the Exif metadata standard, the specification is not currently maintained by any industry or standards organization. It defines a number of metadata tags that are embedded in the header of the image during image capture and cover a broad spectrum:

- Date and time information. Cameras keep time and date information and record it in the metadata.
- Camera and image capture settings and parameters. The camera model and makeup are parameters that do not change for the same camera, while others concern the process of image taking and include focal length, aperture, metering mode, orientation and other dynamic information.
- A thumbnail for previewing the picture on the LCD screen of the camera, in file managers, or in photo manipulation software.
- Descriptions and copyright information.
- Position and direction values.

In the system developed, the relevant Exif tags are extracted from the images and the parameters are used for automatic image annotation and association with map features and semantic objects.

3.4 ELEVATION DATA

Any map of an area can be augmented with elevation data, so that each point on the map is or can be associated with an elevation value. Elevation data can be obtained from Digital Elevation Models (DEMs) which are themselves produced using various techniques, such as remote sensing and land survey.

The maps used in this work have been augmented by a decent quality DEM from the Shuttle Radar Topography Mission (SRTM) [49] of the National Aeronautics and Space Administration [50] agency. This global DEM contains elevation values for approximately every 30m (in the United States) and 90m (in the rest of the world).

Even though the research for this thesis has been conducted in Europe, thus having a lower quality DEM available, the results were satisfying and only close range elevation values had a negative overall impact.

3.5 COLOR FEATURES AND COLOR MODELS

The color is a perceptual characteristic of light described by a color name. In particular, color is light, and light is composed of many colors. Objects absorb certain wavelengths and reflect others back to the viewer. We perceive these wavelengths as color. The wavelengths are not colored, but produce the sensation of color.

The wavelengths that the human eye can detect cover only a small portion of the electromagnetic energy spectrum. This portion represents the visible light spectrum. At one end of the visible spectrum are the short wavelengths of light we perceive as blue. At the other end of the visible spectrum are the longer wavelengths of light we

perceive as red. All the other colors we see in nature are found somewhere along the spectrum between blue and red. [51]

- RGB Color Model

The RGB color model is an additive color model in which the red, green and blue colors are combined in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green and blue.

To form a color with RGB, three colored light beams (one red, one green and one blue) must be superimposed (for example by emission from a black screen or by reflection from a white screen). Each of the three beams is called a *component* of that color and each of them can have an arbitrary intensity, from fully off to fully on, in the mixture. *Zero* intensity for every component gives the darkest color (no light, considered the *black*) and *full* intensity of every one gives a *white*, the quality of this white depends on the nature of the primary light sources, but if they are properly balanced, the result is a neutral white matching the system's white point. When the intensities for all the components are the same, the result is a shade of gray, darker or lighter depending on the intensity. When the intensities are different, the result is a colorized hue, more or less saturated depending on the difference of the strongest and weakest of the intensities of the primary colors employed.

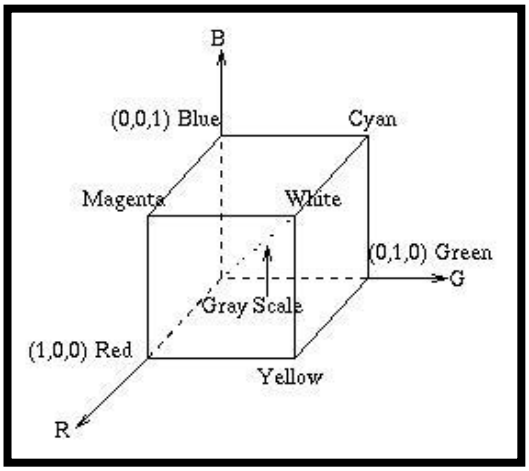


FIGURE 11: RGB CUBE

When one of the components has the strongest intensity, the color is a hue near this *primary* color (reddish, greenish, or bluish). When two components have the same strongest intensity, then the color is a hue of a *secondary* color (a shade of cyan, magenta or yellow).

The RGB color model is represented as a Cartesian cube, with usually Red being the x axis, Green being the y axis, and Blue being the z axis, as in the diagram of Figure 2 (on the left). The diagonal from black (0,0,0) to white (1,1,1) is the gray scale.

- HSL Color Model

HSL stands for hue, saturation, and lightness, and is often also called HLS. HSL is a cylindrical-coordinate representation of points in an RGB color model, which rearrange the geometry of RGB in an attempt to be more intuitive and perceptually relevant than the Cartesian (cube) representation.

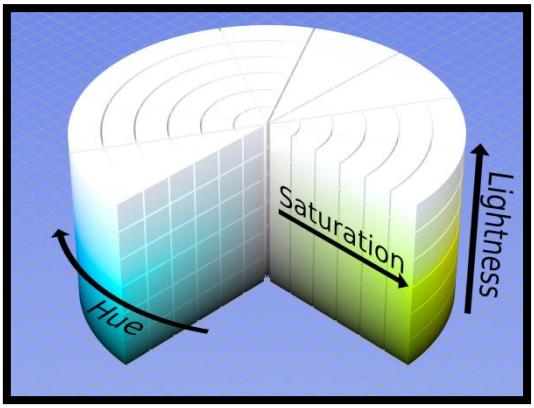


FIGURE 12: HSL CYLINDER

In the HSL cylinder, the angle around the central vertical axis corresponds to "hue", the distance from the axis corresponds to "saturation", and the distance along the axis corresponds to "lightness".

HSL, as mentioned before, is a cylindrical geometry, with hue, the angular dimension, starting at the red primary at 0°, passing through the green primary at 120° and the blue primary at 240°, and then wrapping back to red at 360°. The central vertical axis comprises the neutral, achromatic, or gray colors, ranging from black at lightness 0, the bottom, to white at lightness 1, the top. The additive primary and secondary colors – red, yellow, green, cyan, blue, and magenta – and linear mixtures between adjacent pairs of them, sometimes called pure colors, are arranged around the outside edge of the cylinder with saturation 1; in HSL these have lightness ½. In HSL, both tints and shades have full saturation, and only mixtures with both black and white, called tones, have saturation less than 1 [52].

4. THE SUNLIGHT PARAMETER IN SPIM+

This chapter deals with the *Sunlight* parameter and explains how its value is acquired and how it is used by the framework in order to help associating the image content with semantic objects of the real world.

The sunlight, in the broad sense, is the total frequency spectrum of the electromagnetic radiation given off by the Sun. On Earth, the sunlight is filtered by the Earth's atmosphere, and the solar radiation is called daylight when the Sun is above the horizon.

When sunlight reaches an object, it can travel through the object if the object is transparent, it can be reflected from a shiny object or it can be absorbed if the object is opaque. The color an object is perceived to have is determined by the wavelengths of the light that are absorbed or reflected by the object.

The set of the nature digital images that we had at our disposal in order to ameliorate the initial SPIM system led us to take into account which the sun position was when we captured the images and how the sun enlightened the captured landscape (see Section 5.4 for details).

4.1 CALCULATION OF THE SOLAR POSITION

The calculation of Solar Position is essential To determine how the Sunlight affects the captured natural scene.

The position of the Sun in the sky varies continually during the day and also changes seasonally throughout the year. The sun position is also very much location-dependent.

Thus, the position of the Sun in the sky as seen from the Earth is determined by the following four parameters:

1. The time.
2. The motion of the Earth in its orbit around the Sun, which does not happen at constant speed, because of the eccentricity of the orbit of the Earth.
3. The angle between the axis of rotation of the Earth and the plane of the orbit of the Earth, which is not equal to 90 degrees. This causes the seasons.
4. The location of the observer on the Earth, which determines how high the Sun can get in the sky.

The position of the sun is generally given as an azimuth and altitude angle:

- **Azimuth** represents the horizontal angle of the sun relative to true north.
- **Altitude** represents the vertical angle the sun makes with the horizontal ground plane.

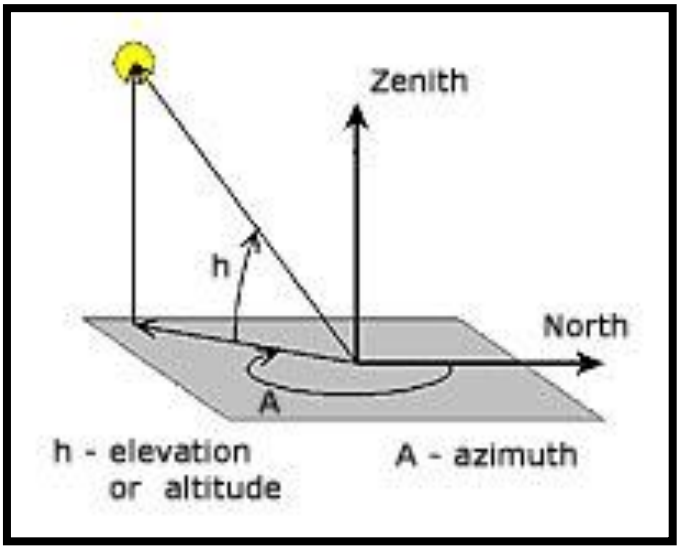


FIGURE 13: AZIMUTH AND ALTITUDE ANGLES

4.1.1 SOLAR POSITIONING ALGORITHM

This subsection describes the *Solar Position* algorithm, which is responsible for the calculation of the Solar Position. The azimuth and altitude of the Sun are calculated using the formulae proposed by Manuel Blanco Muriel, et al [53].

The *Solar Position* algorithm inputs are the time and the location. The time given includes the date (year, month, and day) and the Universal Time (hours, minutes and seconds). The location is given as the longitude and latitude of the observer in degrees. Latitude is considered positive to the North and longitude to the East.

The *Solar Position* algorithm uses the formulas listed below in order to calculate the Solar coordinates:

The Julian Day, JD, is computed from the input data using (1).

$$JD = \frac{\left(1461 * \left(y + 4800 + \frac{(m-14)}{12}\right)\right)}{4} + \frac{\left(367 * \left(m - 2 - 12 * \left(\frac{(m-14)}{12}\right)\right)\right)}{12} - \frac{\left(3 * \left(\frac{\left(y + 4900 + \frac{(m-14)}{12}\right)}{100}\right)\right)}{4} + d - 32075 - 0.5 + \frac{hour}{24.0} \quad (1)$$

where *m* is the month, *y* is the year, *d* is the day of the month and *hour* is the hour of the day in Universal Time in decimal format (i.e., it includes the minutes and seconds as a fraction of an hour, and all the divisions except the last one are integer divisions).

The ecliptic coordinates[1] of the Sun are computed from the Julian Day, by the following set of equations((2),(3),(4),(5),(6) and (7)):

- *Difference between the current Julian Day and Julian Day 2451545.0*

  (= noon 1 January 2000)

  $\mathrm{n} = \mathrm{JD} - 2451545.0 \quad (2)$

  $\Omega = 2.1429 - 0.0010394594 * n \quad (3)$

- *Mean Longitude[2] of the Sun*

  $L = 4.8950630 + 0.017202791698 * n \quad (4)$

- *Mean anomaly[3] of the Sun*

  $g = 6.2400600 + 0.0172019699 * n \quad (5)$

- *Ecliptic longitude of the Sun*

  $l = L + 0.03341607 * \sin g + 0.00034894 * \sin 2g - 0.0001134 - 0.0000203 * \sin \Omega \quad (6)$

- *Obliquity of the ecliptic (*in other words *the Earth's axial tilt)*

  $ep = 0.4090928 - 6.2140 * 10^{-9} * n + 0.0000396 * cos\Omega \quad (7)$

The *Ecliptic latitude* of the Sun, the perpendicular distance of the Sun from the ecliptic, is always so small that it can be omitted.

The conversion from ecliptic to celestial coordinates[4] is accomplished using the standard trigonometric expressions:

- *Right ascension: The longitudinal angle of the equatorial (celestial) system. It measures an angle that increases toward the east as measured from a zero point on an equator. The zero point is known as the first point of Aries, which is the place in the sky where the Sun crosses the celestial equator at the March equinox.*

  $ra = tan^{-1}\left[\frac{\cos ep * \sin l}{\cos l}\right]$

- *Declination: The latitudinal angle of the equatorial (celestial) system. It measures the angle of an object above or below the celestial equator.*

  $\delta = sin^{-1}[\sin ep * \sin l]$

The conversion from celestial to horizontal coordinates[5] is achieved by the following set of equations ((8),(9) and (10)):

[1] The ecliptic coordinate system is a celestial coordinate system that uses the ecliptic for its fundamental plane. The ecliptic is the path that the sun appears to follow across the celestial sphere over the course of a year. It is also the intersection of the Earth's orbital plane and the celestial sphere. The ecliptic latitude or celestial latitude is measured positive towards the north. The ecliptic longitude or celestial longitude is measured eastwards from 0° to 360°.

[2] Mean longitude is the longitude at which an orbiting body could be found if its orbit were circular, and free of perturbations, and if its inclination were zero.

[3] The position that the Earth would have relative to its perihelion if its orbit were a circle is called the mean anomaly

[4] The equatorial coordinate system functions by projecting the Earth's geographic poles and equator onto the celestial sphere. It allows all the earthbound observers to describe the apparent location in the sky of sufficiently distant objects using the same pair of numbers: the right ascension and declination.

– *Greenwich mean sidereal time*

$$gmst = 6.6974243242 + 0.0657098283 * n + hou \quad (8)$$

– *Local mean sidereal time*

$$lmst = (gmst * 15 + long) * \left(\frac{\pi}{180}\right) \quad (9)$$

, *long* represents Geographical Longitude

– *Hour angle*

$$\omega = lmst - ra \quad (10)$$

The *Solar Azimuth* is given by (11).

$$\gamma = tan^{-1}\left[\frac{-\sin\omega}{\tan\delta * \cos\Phi - \sin\Phi * \cos\omega}\right] \quad (11)$$

,Φ represents Geographical Latitude

The *Solar Zenith angle* is given by (12).

$$\theta_z = cos^{-1}[\cos\Phi * \cos\omega * \cos\delta + \sin\delta * \sin\Phi] \quad (12)$$

The zenith angle is the angle between the sun and the vertical and it holds that altitude = 90° - zenith.

The outputs of the *Solar Position* algorithm are the Solar Azimuth and the Solar Zenith Angle which constitute the Sun Coordinates.

[5] The *horizontal*, or altitude-azimuth, system is based on the position of the observer on the Earth, which revolves around its own axis once per a sidereal day (23.hours, 56 minutes and 4.091 seconds) in relation to the "fixed" star background. The positioning of a celestial object by the horizontal system varies with time, but is a useful coordinate system for locating and tracking objects for observers on the Earth. It is based on the position of stars relative to an observer's ideal horizon.

4.1.2 Relative Position of the Sun with respect to the User

At this point, the Sun Coordinates have been calculated but they are just numbers, they mean nothing to the Framework. Therefore, it is necessary to give them semantic meaning. What is needed exactly is the information whether the Sun is on the left or on the right of the user. The procedure described below is responsible for this task.

The procedure, presented in Algorithm 1, is simple. It takes as input the Solar Azimuth angle and the User's Azimuth angle. The User's Azimuth angle coincides with the compass direction value which is embedded in the photo metadata. Both angles are measured from the north increasing towards the east.

Then, a subtraction takes place. The User's Azimuth angle is subtracted from the Solar Azimuth angle. The difference that results shows whether the Sun is on the user's left or on the user's right. Firstly, if the difference is positive it means that the sun precedes the user else the sun follows the user. Afterwards we check if the difference is negative and less than -180 then the Sun is located to on the user's right, if the difference is negative and greater than -180 the Sun is located to on the user's left, otherwise if the difference is positive and greater than 180 then the Sun is located to on the user's left and finally if the difference is positive and less than 180 then the Sun is located to on the user's right.

Algorithm 1 - Relative Position of the Sun with respect to the User

```
Double SolarAzimuth, UserAzimuth, SunPosition;
SunPosition =      SolarAzimuth – UserAzimuth;
if (SunPosition < 0) {
        if (SunPosition < -180) → Sun isRight
        else Sun isLeft
        }
 if (SunPosition > 0) {
        if (SunPosition < 180) → Sun isRight
        else Sun isLeft
        }
```

Figure 14 depicts an example of the aforementioned procedure. Suppose that the Solar Azimuth angle equals to 60 degrees and the User's Azimuth equals to 120 degrees. The Sun Position equals to 60 – 120 = -60, which is less than zero and greater than -180. Then the procedure makes the decision that the Sun is on the user's left.

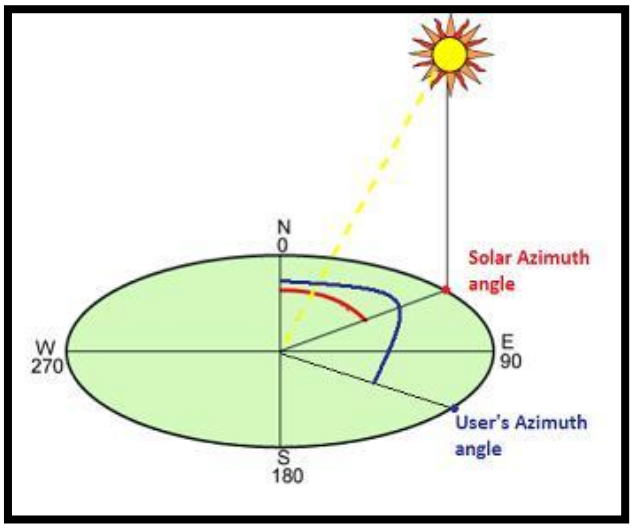


FIGURE 14: RELATIVE POSITION OF THE SUN AND USER

4.2 SUMMARY

To sum up, knowing the direction of the Sunlight as it is perceived from our direction, at the moment we take a photograph, allows us to manage the image content more effectively. This information, in combination with the weather conditions that prevailed at the moment of image capturing, help us to decide which objects/individuals are better visible in the image (see Section 5.4 for details). This way, the possibility of recognizing them correctly increases and better off, we provide the image registration algorithm with correct and consistent data inputs.

5. EDGE PROCESSING IN SPIM+

This chapter analyzes the procedures that have been developed for detecting and extracting the boundaries of any individual included in the 2D-model image of a photo. The boundaries of the individual may be detected not only from the 2D-model image but also from the segmentation image. The detection and extraction of the boundaries from the two images is an essential procedure, which is described in Sections 4.2-4.5, since it provides the inputs for the matching algorithm (LineSegmentMatcher).

The activity diagram of Figure 15 outlines the procedure followed by the framework in order to convert a photo to semantic map. As is shown in the diagram user photo is inserted in the system and then the procedures for designing the 2D-model image and the Segmentation output image, begin. The next step is to extract from both these images the boundaries of the individuals. Finally these boundaries are given as inputs to image registration algorithm.

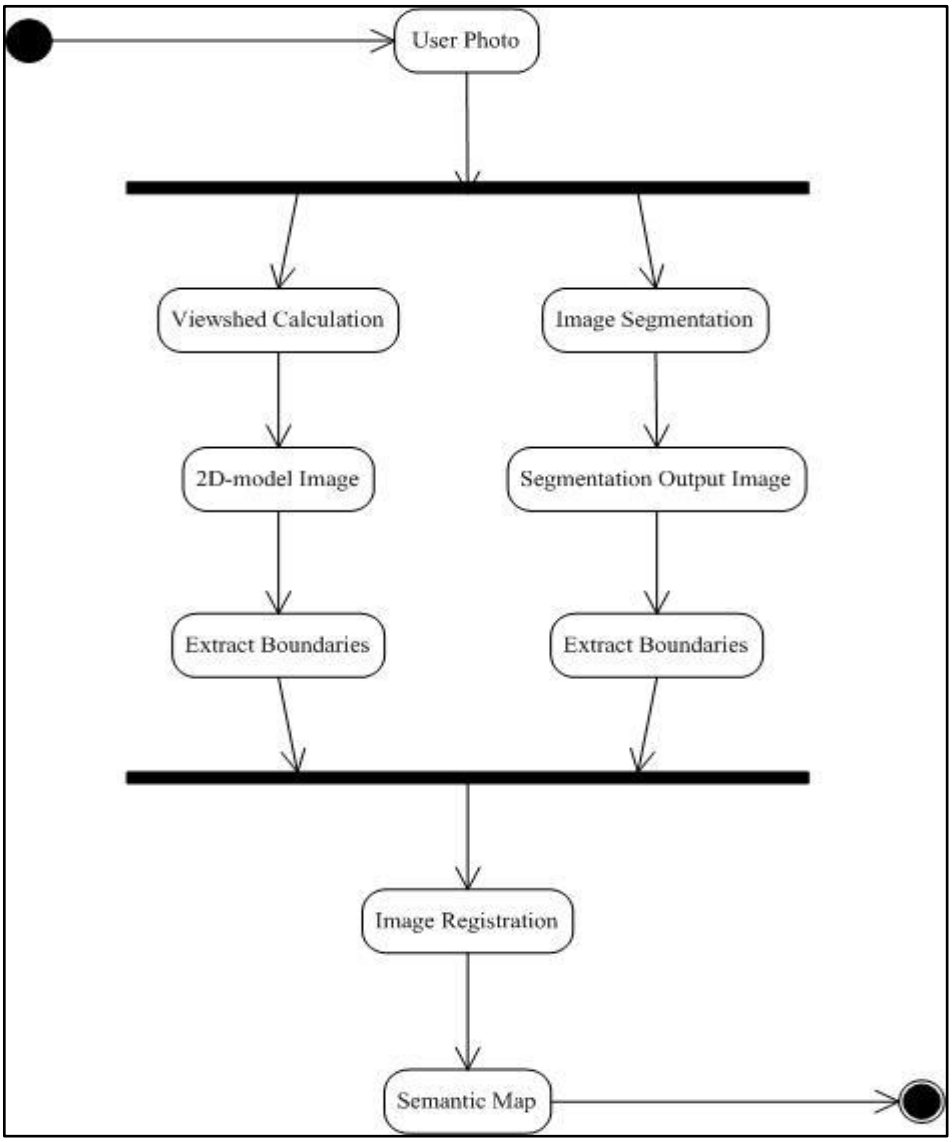


FIGURE 15: ACTIVITY DIAGRAM OF THE PHOTO TO SEMANTIC MAP CONVERSION

5.1 PRODUCTION OF THE 2D-MODEL AND SEGMENTATION IMAGES

This section describes the procedures which produce the 2D-model image and the segmentation image of a photo. The procedures have been implemented in the SPIM System and they are fundamental parts of its architecture. For this reason the SPIM+ system continues using them.

5.1.1 VIEWSHED CALCULATION

Firstly, the Viewshed Calculation procedure takes place. The term Viewshed is defined as an area of land, water or other environmental element that is visible to the human eye from a fixed point. Here, the human eye is the camera lens and the visible area is only computed for a specific direction and angle.

The algorithm is a modified version of the direct method for visible area computation presented in Franklin et al. [54]. The algorithm traces rays that begin from the user, follow the direction of the camera and are within the angle of view. At regular intervals, it determines if the current GPS position is visible from the point of view and checks if a semantic object is visible at this position. Whenever the visibility becomes broken following a visible area, a new visible point is found and recorded. The visibility information is stored in suitable data structures.

The Viewshed Calculation algorithm (Algorithm 2) takes as input the headings corresponding to the start and end of the angle of view, the elevation and GPS coordinates of the point where the user is standing, the number of rays to be casted and the maximum distance for each ray to search for visible areas. The output is a vector containing all the information about each ray cast.

The distance covered in each interval is variable. Near the user, more detail about the visible areas and individuals is required and the interval is small. When the ray moves farther, the intervals are getting bigger. For example, we start with an interval of 2 meters length. Then, gradually as the distance becomes larger than 20km, the interval gets increased to 20 meters.

A Great Circle Calculator provides the next GPS position according to the distance and direction from the point of view. The elevation, the individuals located and the height of this point are computed.

The tangent of the angle between the horizontal line parallel to the level of the User and the line that connects the User and the current position is calculated. This is the view angle and its tangent is used for the construction of the 2D model image.

To determine if a point is visible to the observer or not, the Line of Sight (LOS) function (13) is used [55].

$$LOS = max\left(PointHeight\,, UserHeight + \frac{Distance}{PrevDistance} * (PrevLOS - UserHeight)\right) \quad (13)$$

Right after computing the LOS, the algorithm compares its value to the current point's height. If LOS is greater or equal to the current point's height, then the point is considered visible. In this case, information about the visible individuals at this point and the view tangent calculated beforehand are stored in the data structures of the current ray.

After this step, the ray is added to the vector of rays, which is the output of the algorithm. When all the rays have been cast, the algorithm ends. Then, all the needed information to create the 2D model image of the photo has been collected.

Algorithm 2 - Viewshed Calculation

```
Input:
Integer StartAngle, EndAngle, UserHeight, NumOfRays, KMLimit
GPSPoint OriginPoint
Output:
Vector<Ray> Rays

Begin
//Local variable declarations
GPSPoint CurrentPoint = null;
Integer CurrentAngle = StartAngle, CurrentHeight = -1, CurrentHeading =
StartHeading, CurrentIteration = 0;
Ray CurrentRay = new Ray();
Boolean Descending = False;
Double CurrentDistance = 0.0, PreviousDistance = 0.0, CurrentTan = 0.0, LOS
= 0.0,
AngleStep = (StartAngle - EndAngle) / NumOfRays;
Vector Footprints = new Vector();
//for each ray
while(CurrentIteration < NumOfRays){
CurrentIteration++;
//for each GPS point located "CurrentDistance" kilometers away from
//the camera up to KMLimit kilometers away
while(CurrentDistance < KMLimit){
//calculates the distance to the next position in the ray
CurrentDistance += getKMStep(CurrentDistance);
//moves to the next GPS position following the ray, according to the current
//angle of the ray and the distance from the camera
CurrentPoint = getNextGPSPoint(OriginPoint, CurrentAngle,
CurrentDistance);
//finds the elevation at the current position and current distance, taking also
//into account the curvature of the Earth
CurrentHeight = getElevation(CurrentPoint, CurrentDistance);
//finds which individual footprints are located in the current position
Footprints = getFootprintsAtPoint(CurrentPoint);
//returns the height of the highest footprint in the current position to compute
//the height of the current position
CurrentHeight += getHighestFootprintHeight(Footprints);
//calculates the tangent of the angle the camera views the current position
//(CurrentDistance is converted to meters)
CurrentTan = (CurrentHeight - UserHeight) / (CurrentDistance*1000);
//calculates the line of sight function
LOS = getLineOfSight(LOS, UserHeight, CurrentDistance, PreviousDistance,
CurrentHeight);
```

```
if(LOS ≤ CurrentHeight){ //the current point is visible
//finds which footprints are visible and adds their information to the ray
CurrentRay.findAndAddVisibleFootprints(Footprints);
if(Descending == True){ //The ray was descending, but now it is ascending
Descending = False;
} //end if
else{ //(LOS > CurrentHeight) – the current point is not visible
//if the ray was ascending but now isn't, a visible peak has been found
if(Descending == False){
//stores the tangent of the current view (used for creating the 2D model)
CurrentRay.addVisiblePeak(CurrentTan);
Descending = True;
} //end if
} //end else
//stores the previous distance for use in the next step
PreviousDistance = CurrentDistance;
} //end of inner while
//proceeds to the next angle for the next ray
CurrentAngle += AngleStep;
CurrentDistance = PreviousDistance = 0.0;
Descending = False;
Rays.add(CurrentRay);
}
End
```

Figure 16 and 17 show a photo and its 2D-model image created with the aforementioned algorithm.

FIGURE 16: USER PHOTO

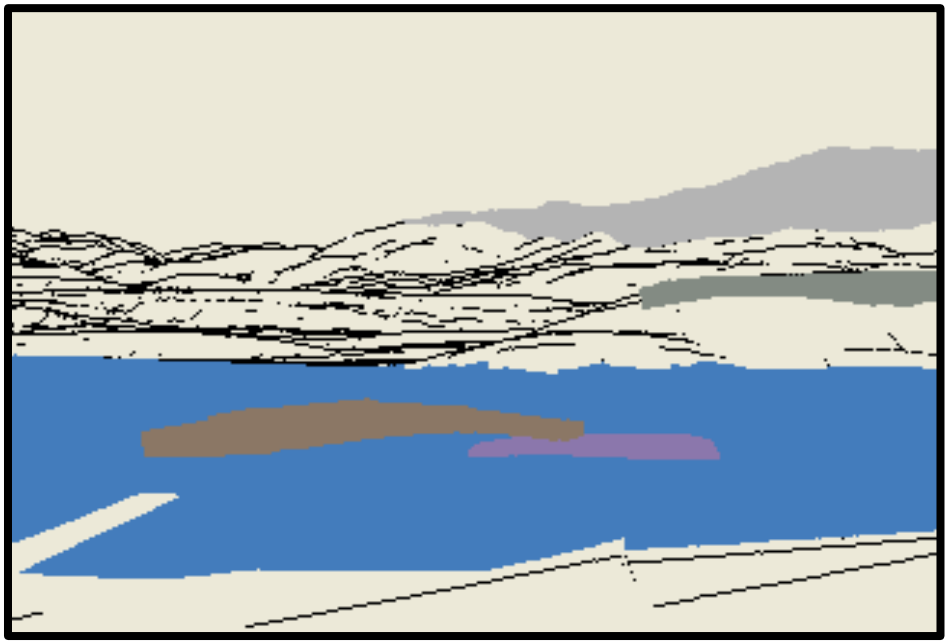


FIGURE 17:2D MODEL IMAGE

5.1.2 IMAGE SEGMENTATION

Image segmentation is the second procedure. Its aim is to facilitate the detection of geographic features on the picture. The image is segmented into regions, using the statistical region merging method by Nock and Nielsen [56]. The authors provide a simple but effective region growing method that can be parameterized to define the limits of how easily the regions are merged. The segmentation is performed on a scaled version of the original image to make the computation faster. Figure 18 shows the image segmentation of the photo of Figure 16 using this method.

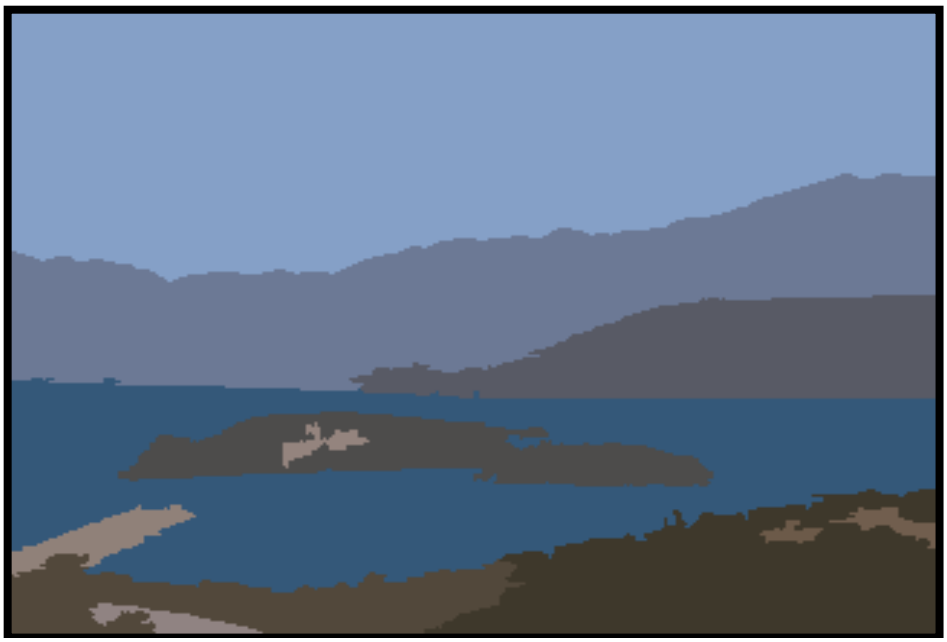


FIGURE 18: IMAGE SEGMENTATION

5.2 SKYLINE OR RIDGELINE

Skyline is the line on which the Sky and the Earth's surface appear to meet.

A ridge is a geological feature that features a chain of mountains or hills that are of a continuous elevated crest for some distance. Here, the Ridgeline refers to the line on which the Sky and the ridge meet each other.

The initial SPIM system uses this kind of lines in order to register the images. The implementation is briefly described below.

- 2D-model image

Having completed the Viewshed Calculation, the Viewshed Calculation algorithm has collected all the necessary information and creates the 2D model image of the picture.

The first step is detecting the Skyline. The skyline is defined as the set of visible points with the highest view tangent for all the rays. The Ridgeline is detected by following the Skyline points belonging to an individual that represents a mountain.

Figures 19 and 20 show a photo and its 2D-model image. In Figure 19 the Ridgeline is marked with red color in order to become clear which this line is.

FIGURE 19: USER PHOTO

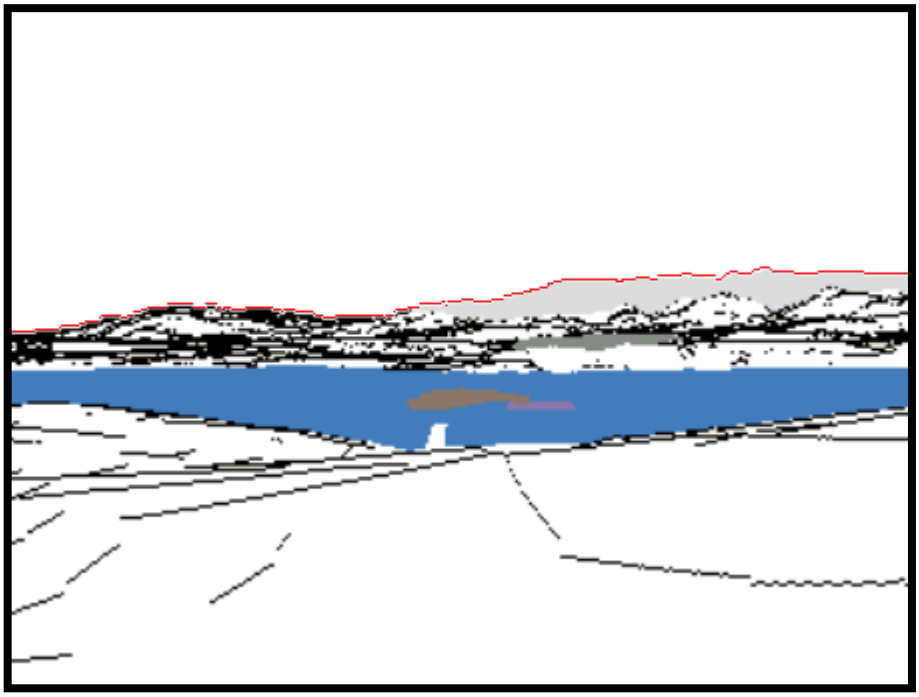


FIGURE 20: DETECTION OF SKYLINE- RIDGELINE

- Segmentation image

The next step is to detect the Skyline (and Ridgeline) from the Segmentation image. There is a general assumption that the Sky is the topmost region of the image. However, this assumption is not always true. Sometimes there are two or more regions that belong to the sky, which are either small obstacles (for example clouds) in the way or other major obstructions. The algorithm presented here (SkyDetector) utilizes heuristics and focuses on solving this problem.

Initially, the first sky region is discovered and established by checking the topmost row of pixels and detecting the best candidate sky region. SkyDetector then scans the Segmentation image from the top, one column of pixels at a time, and detects regions where the sky region is interrupted during the scan and then found again below them in the Segmentation image. These regions have a higher probability of being obstacles such as clouds and are merged with the sky region if they interrupt the scanning too much. Whenever a new region is merged with the sky region, the algorithm scans the image from the beginning. It ends when there are no candidate obstacle regions detected during a complete scan of the image.

Extracting the Skyline points from the resulting Sky region is as simple as taking the lowest pixel of each pixel column that belongs to the Sky region. As with the 2D model skyline case, points that break the smoothness of the skyline are discarded at a later step. Figures 21 and 22 show the effect of the SkyDetector algorithm. In the first figure, the variable colors of the sky are divided into several regions. SkyDetector determines that these regions belong to the Sky and merges them. It returns the segmentation result of the second figure.

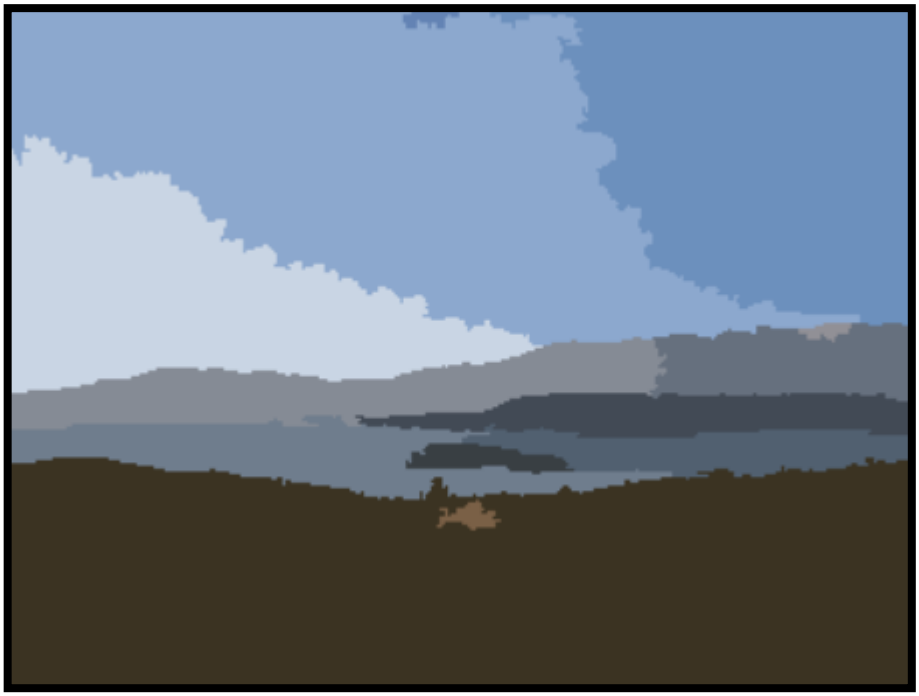


FIGURE 21: SEGMENTATION IMAGE

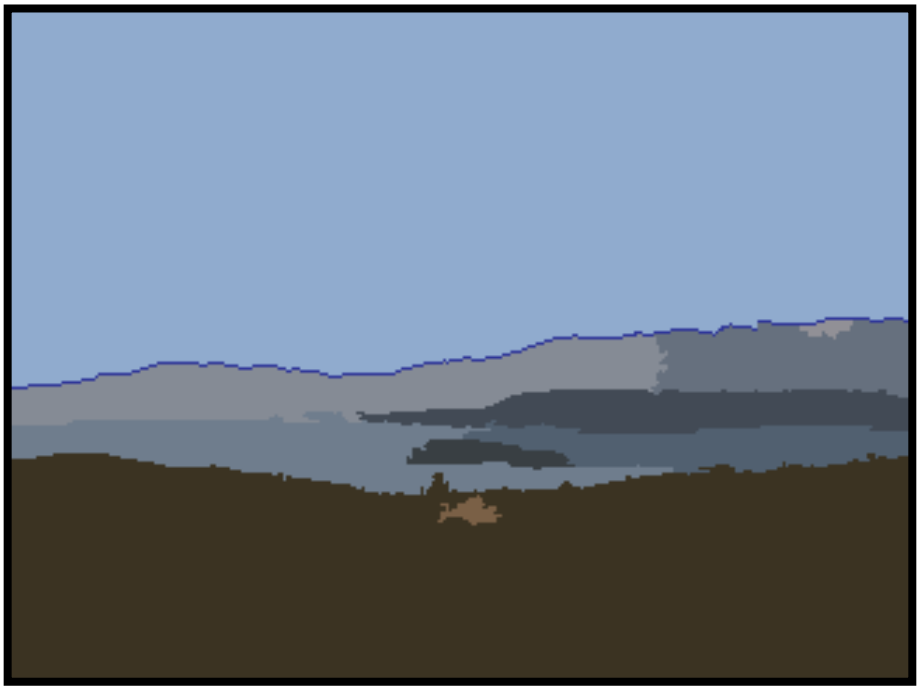


FIGURE 22: DETECTION OF A SINGLE SKY REGION AND EXTRACTION OF THE SKYLINE - RIDGELINE

5.3 COASTLINE

The Coastline is the line at which the land and the sea appear to meet. In this section we present the coastline detection algorithms for both the 2D-model and segmentation images.

- 2D-model image

Detecting the Coastline in the 2D-model image is quite easy. The procedure promises the existence of a semantic individual which represents the Sea. The algorithm scans the 2D-model image from the top, by one column of pixels at a time. For each column, it records the first pixel in which it detects the semantic individual "SEA" and continues until it finds the last pixel that belongs to the semantic individual "SEA", which also records. Finally, it collects all the last pixels from all the columns. This collection of pixels forms the Coastline. Figure 23 shows the procedure of Coastline detection on 2D-model image.

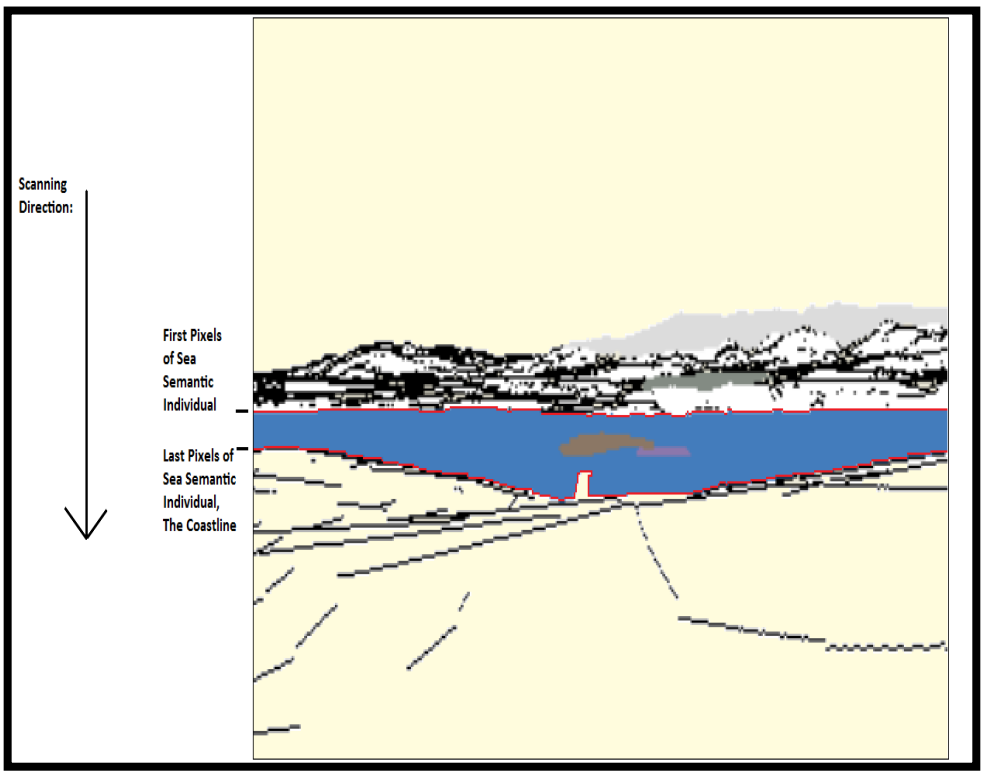


FIGURE 23: COASTLINE DETECTION

- Segmentation image

A heuristic algorithm has been implemented for the detection of the Coastline in the Segmentation image. The algorithm is called LandDetector and it detects the land regions in the image and merges them properly.

The algorithm scans all the regions of the image and filters them according to two criteria, the RedPrevalence and the GreenBlueDifference. RedPrevalence filters out all regions in which the red component of the color model prevails over both the green and blue components over a predefined threshold value. This procedure excludes the regions that do not have the brown land color. However, there is a possibility that there are some regions that have a strong red component and are not land regions. The job of the GreenBlueDifference criterion is to filter out these regions. It has been found experimentally that the color of the land has the following characteristic: the green component is always stronger than the blue one. LandDetector takes that information into account and removes the regions that do not have a strong green component over the blue one, leaving the regions that truly represent the land. They are then merged accordingly.

It has been noticed that the color of regions that represent Land is a set of different shades of brown. The shades of brown, according to the RGB color model, have the following characteristics: the Red Component is greater than the other two Components (Green and Blue) and also the Green Component is greater than Blue Component. Therefore, the LandDetector algorithm, whose implementation is based on this logic, detects successively the Land regions and merges them properly.

The LandDetector algorithm returns one region which represents Land area. Then we scan this region bottom-up by one column of pixels at a time. For each column, we record the last Land pixel. Finally, we collect all the last pixels from all the columns. This collection of pixels forms the Coastline. Figure 24 shows the procedure of Coastline detection on segmentation image.

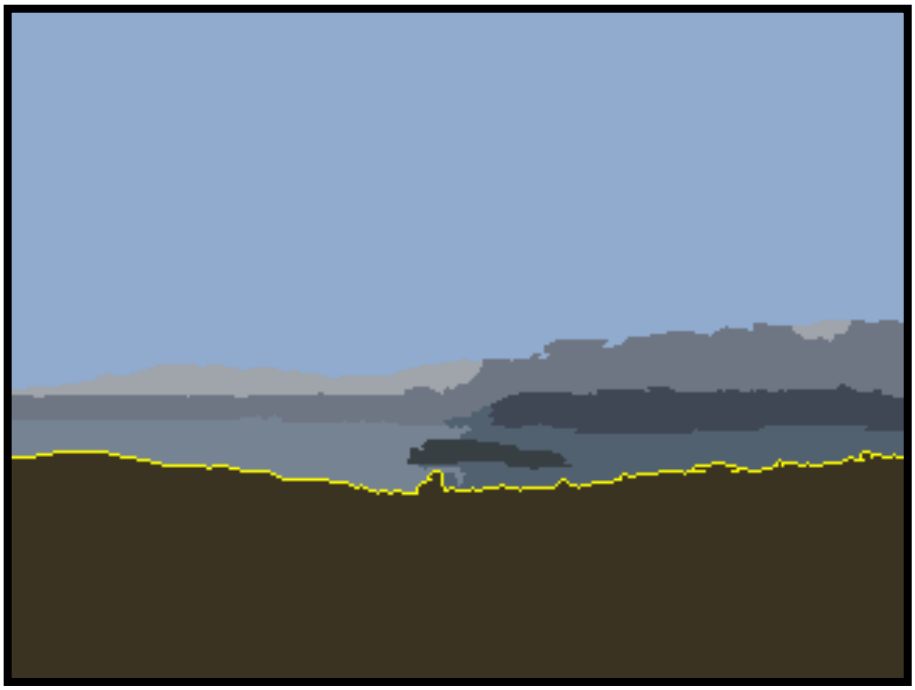


FIGURE 24: DETECTION OF A SINGLE LAND REGION AND EXTRACTION OF THE COASTLINE

Figure 25 depicts the activity diagram of the Land Detection Algorithm. It is worth mentioning that the algorithm checks if the image contains land regions right in front

of the User. More precisely it checks if the last pixel on the bottom right of the image is Blue. If the specific pixel is Blue then it means that right in front of the User there is Sea.

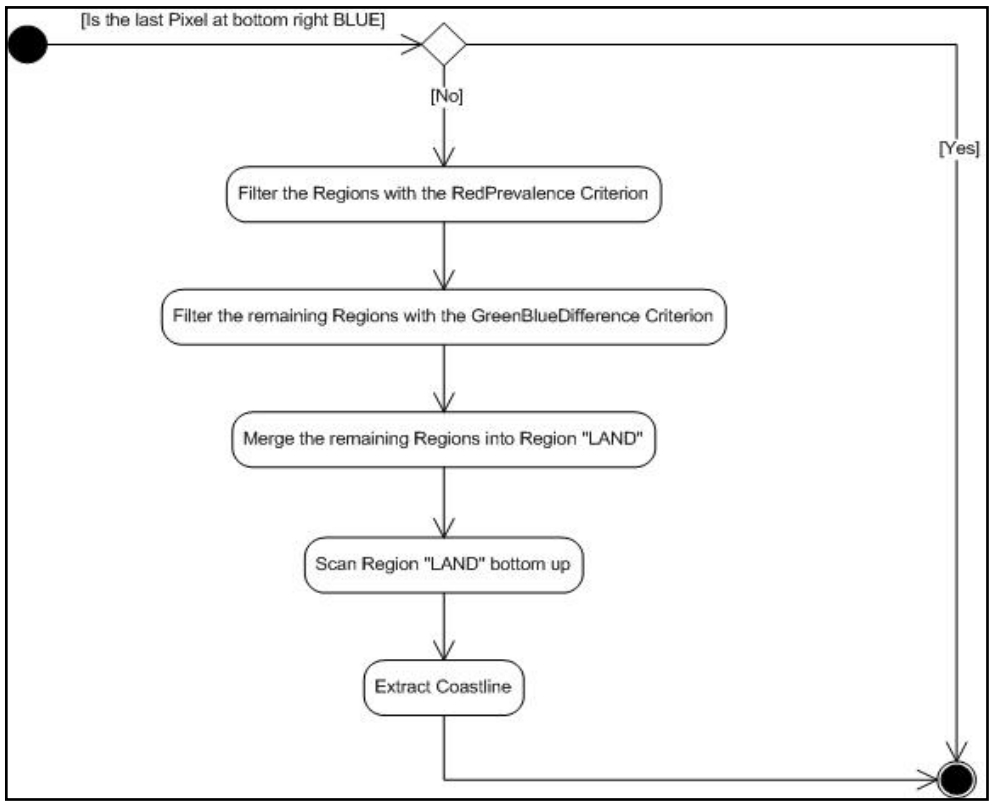


FIGURE 25: ACTIVITY DIAGRAM OF THE LAND DETECTION ALGORITHM

5.4 AFTER THE SEA COASTLINE

The After the Sea Coastline is the line at which the sea and the mountains appear to meet.

This line is not always distinguishable in the segmentation image. There are cases that it is not distinguishable at all, there are cases that it is distinguishable in its whole and there are cases that only a part of it, it is distinguishable.

- 2D-model image

Detecting the After the Sea Coastline in the 2D-model image is quite easy. The procedure premises the existence of a semantic individual which represents Sea. The algorithm scans the 2D-model image from the top, by one column of pixels at a time. For each column, it records the first pixel in which it detects the semantic individual "SEA". Finally, it collects all these pixels from all the columns. This collection of pixels forms the After the Sea Coastline. Figure 26 shows the procedure of Coastline detection on the 2D-image. The Opposite of the User Coastline is painted red.

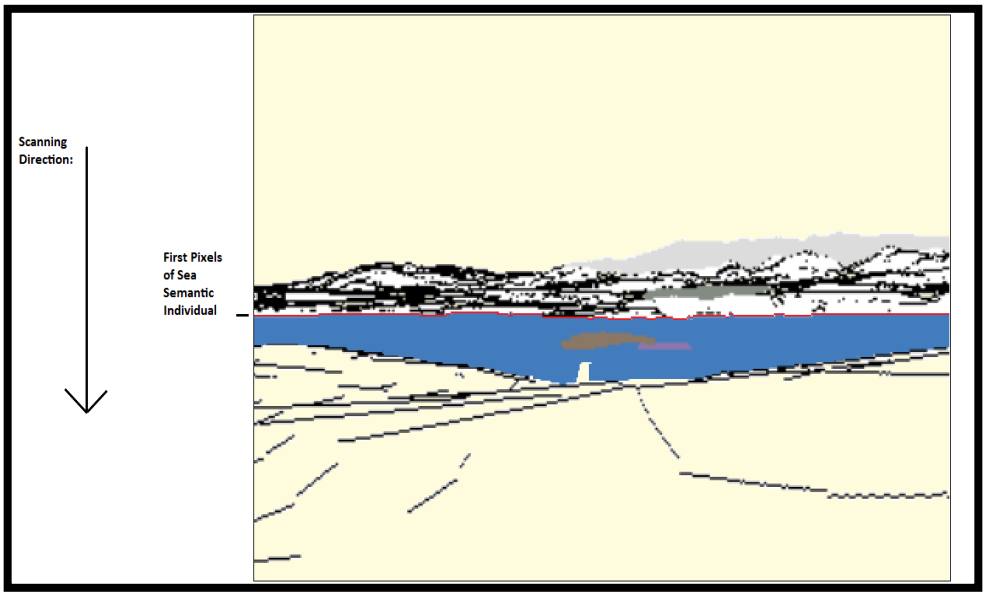


FIGURE 26: AFTER THE SEA COASTLINE DETECTION

- Segmentation image

In the Segmentation picture it is quite difficult to detect correctly the region that represents a Sea individual. One may think at first that a region with blue color is highly expected to be a sea region. However, there are many cases of photos where, besides the region or regions that represent sea, mountain regions are also blue, different shades of blue but still blue. So the logic: find the blue regions of the Segmentation image and merge them in order to form the Sea region will not always function properly. This way, the possibility of a bad Sea region detection is high enough since the final region which will represent the Sea will have embedded regions that represent mountains too.

By observing the segmentation images of a big set of nature digital photos, a better solution to the problem was found. We observed that the part of the sea region in which the sun enlightened more has two important characteristics and we took advantage of them. Firstly, in the segmentation image, the part of the Sea which is enlightened more by the Sun forms a separate region which represents this part of the Sea properly. Secondly, the color lightness value of this region is the highest of all the other blue regions and it is greater than 50. This second fact determines the criterion used in detecting the enlightened by the Sun Sea region.

A heuristic algorithm has been implemented for the detection of the aforementioned region. The algorithm is called SeaLightDetector. It uses the Sunlight parameter, to which we referred in the third chapter. The logic of the algorithm is the following:

→ Use the information about **the position of the Sun relative to the User** and decide which part of the photo is enlightened by the Sun.
→ Find the blue regions in this part (blue regions that represent Sky are already excluded).
→ Convert their RGB[6] color values to HSL[7] color values.
→ Calculate their lightness values.
→ Sort them in descending order, based on their lightness value. (The region with the highest lightness value is first in the order).
→ Check if the first region neighbors with the shore. If yes, then this is the region we are looking for. If not, check the next region.
→ The algorithm continues until it finds a region with lightness value greater than 50, which also neighbors with the shore. If there is not such a region it returns the null value.

The procedure takes into account the regions with a lightness value greater than 50. Lightness is represented as percentage. 0% lightness means black color, 100% lightness means white, and 50% lightness means "normal". Apparently lighter colorations, which we look for, have lightness values between 50 and 100.

Figure 27 depicts the activity diagram of the SeaLightDetector algorithm.

[6] Red, Green, Blue
[7] Hue, Saturation, Lightness

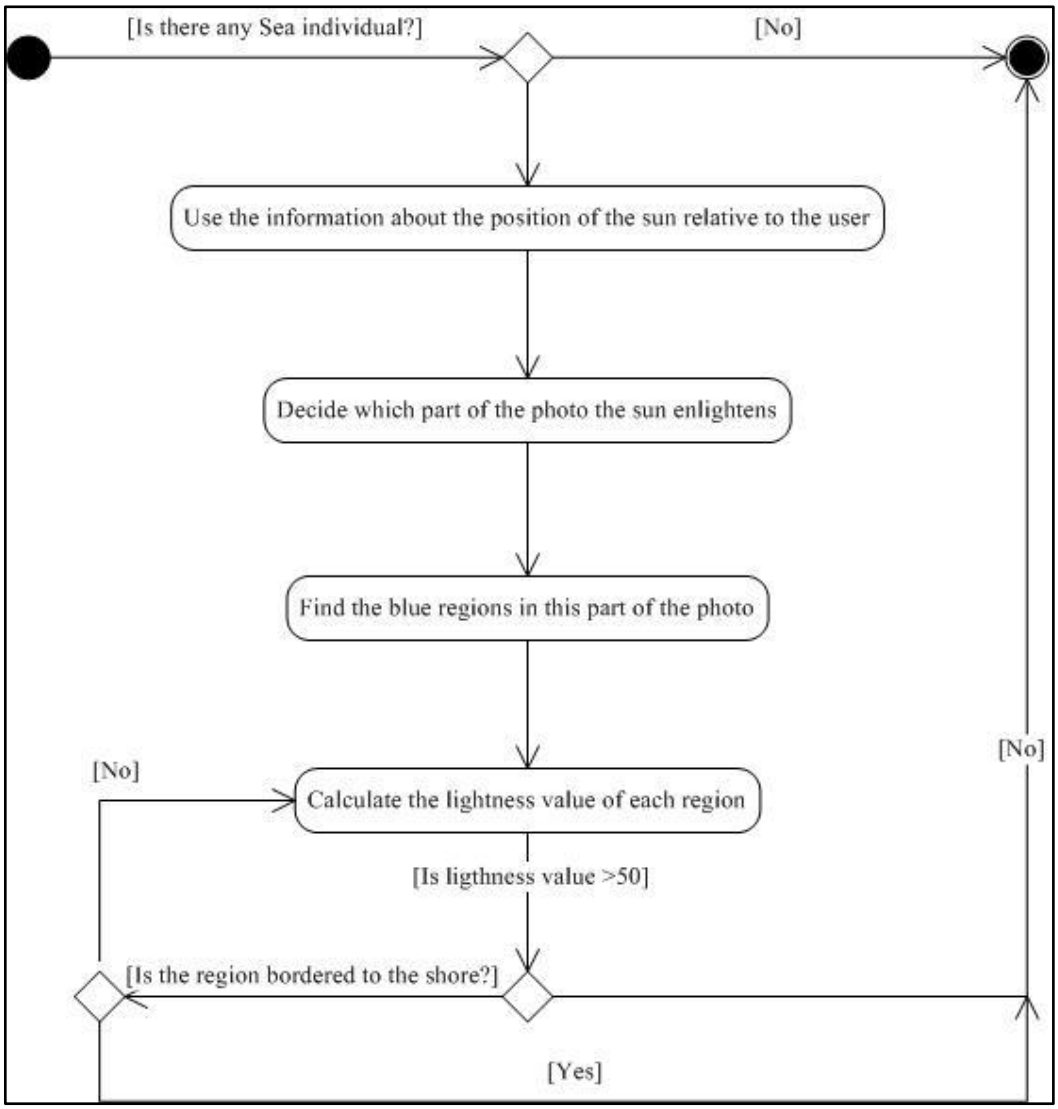


FIGURE 27: ACTIVITY DIAGRAM OF THE SEALIGHTDETECTOR ALGORITHM

When this algorithm returns, we have detected the enlightened by the Sun part of the Sea. Then we scan this region bottom-up in order to extract its upper boundary. This boundary represents the After the Sea Coastline.

Following figures "cast a light" on the whole procedure.

FIGURE 28: USER PHOTO

When the User captured the above picture the Sun was on his left. This means that the Sun enlightens more the left part of the photo. This fact is quite obvious when you look the above photo.

Let see what happens in the segmentation image of this photo, which is depicted in Figure 29.

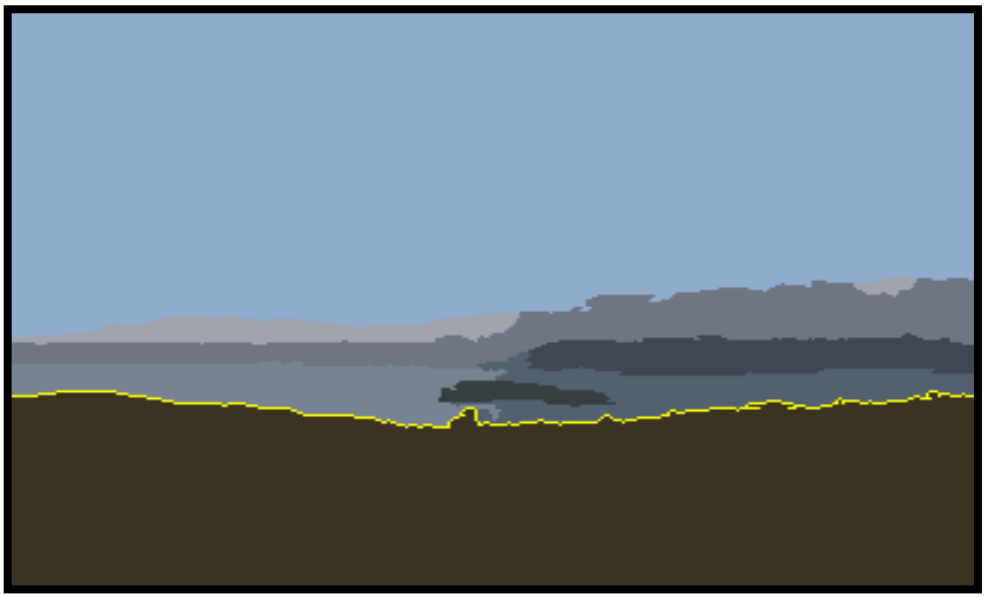


FIGURE 29: SEGMENTATION IMAGE OF THE PHOTO OF FIGURE 28

Initially, it is confirmed by a human eye that the left part of this image has lighter colors. At this point it has been decided by the algorithm that it will seek for a light sea region at the left part of the image.

The candidates to be sea regions are marked in the Figure 30.

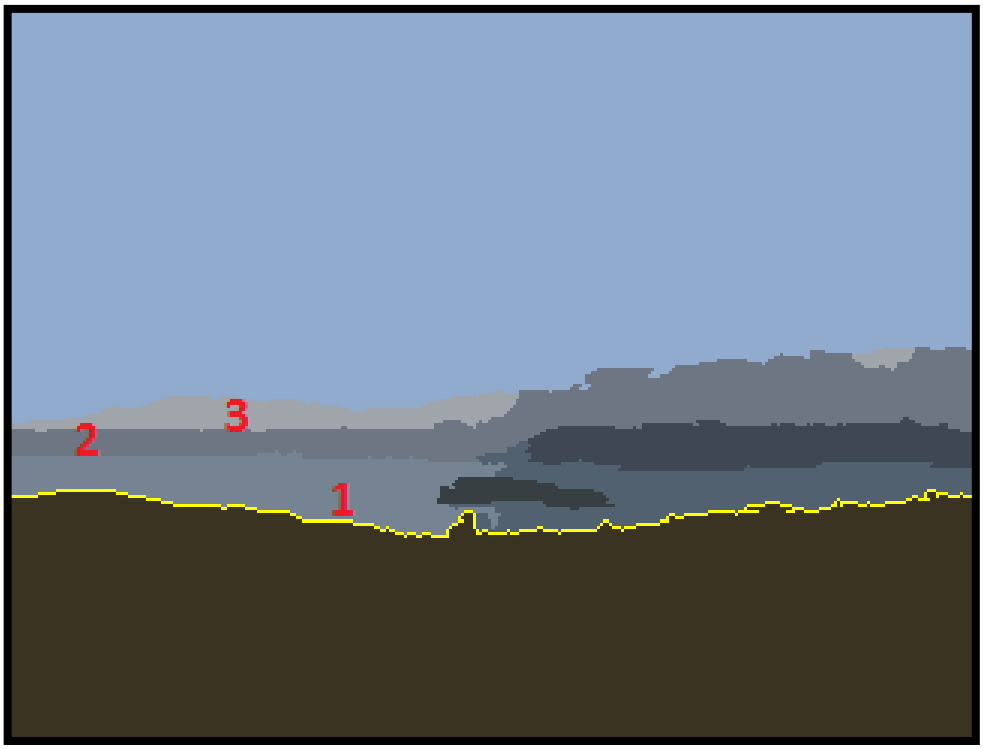


FIGURE 30: THE CANDIDATE SEA REGIONS ENLIGHTENED BY THE SUN

These three regions have the Color characteristics presented in Table 1.

Region	R	G	B	Lightness
1	118	131	160	52
2	110	118	164	47
3	160	164	171	65

TABLE 1:REGIONS COLOR CHARACTERISTICS

The SeaLightDetector algorithm sorts the regions as is shown inTable 2.

Region	R	G	B	Lightness
3	160	164	171	65
1	118	131	160	52
2	110	118	164	47

TABLE 2:SORTING OF THE REGIONS

Then it examines the first region (region 3). Its lightness value (65) is greater than 50, so the algorithm continues. It checks if the region neighbors with the coast. Region 3 does not fulfill this criterion so the algorithm goes on. The next region which is examined is region 1. Its lightness value (52) is greater than 50 and it is also bordered to the coast. Therefore region 1 is the region we are looking for and the algorithm returns. Figure 31 shows the result of the algorithm.

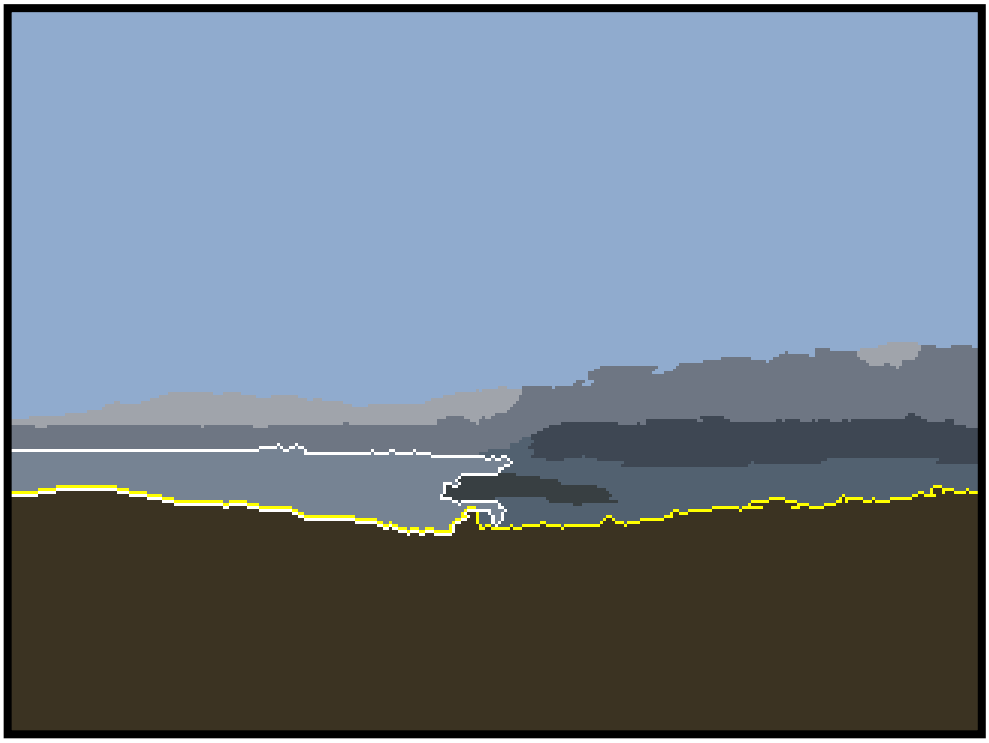


FIGURE 31: THE OUTCOME OF THE SEALIGHTDETECTOR ALGORITHM

At this point, the SeaLightDetector algorithm has detected the enlightened from the sun part of sea. Then we scan this region bottom-up by one column of pixels at a time. For each column, we record the last pixel of the region. Finally, we collect all the last pixels from all the columns. This collection of pixels forms the After the Sea Coastline. Figure 32 shows the extraction of the aforementioned line, which is colored in red.

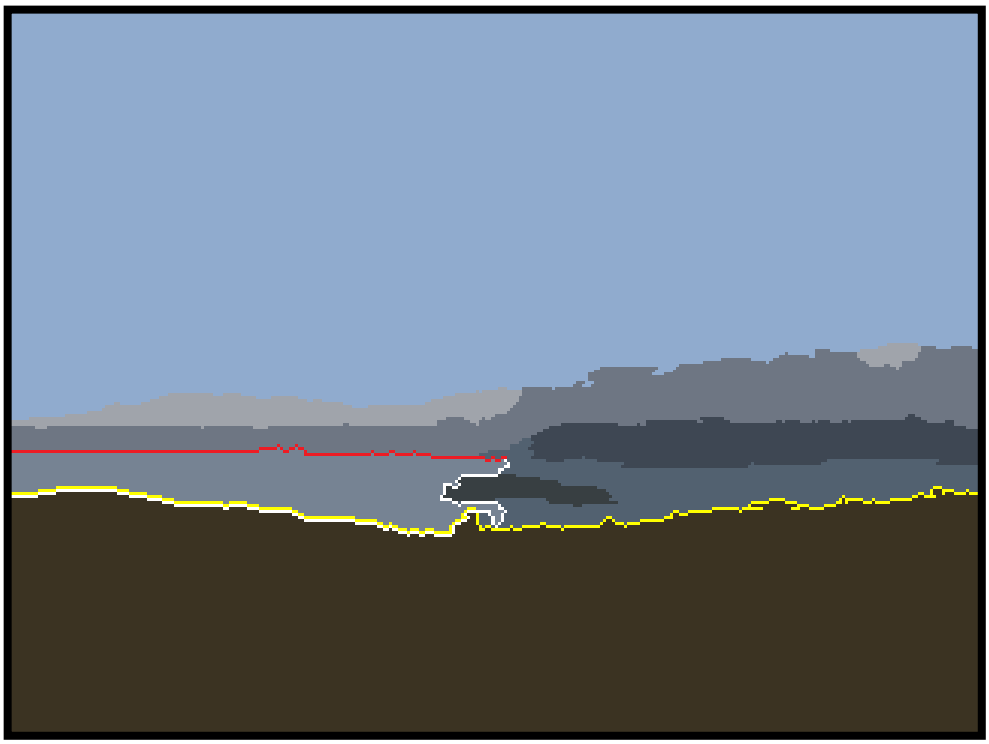


FIGURE 32: THE AFTER THE SEA COASTLINE EXTRACTION

A significant detail is that, in the Segmentation image, the After the Sea Coastline extracted, is a part of the whole line, while in the 2D-model image, the particular line is extracted wholly. Given that both the images (the Segmentation image and the 2D-model image) have the same dimensions, we find the coordinates of the points-pixels at which the short line begins and ends and then we cut the long line at these points. Thus, from both the images we take the correct line.

5.5 ISLAND UPPER LINE

An island is any piece of land that is surrounded from water. The Island Line is a closed curve (a circle) at which the island appears to meet with the water. The Island Upper Line is the arc of the upper semicircle of this circle.

- 2D-model image

Detecting the Island Upper Line in the 2D-model image is quite easy. The procedure promises the existence of a semantic individual which represents an Island. The algorithm scans the 2D-model image from the top, by one column of pixels at a time. For each column, it records the first pixel in which it detects the semantic individual "ISLAND". Finally, it collects all these pixels from all the columns. This collection of pixels forms the Island Upper Line. Figure 33 shows the procedure of Island Upper Line detection. The Island Upper Line is painted red.

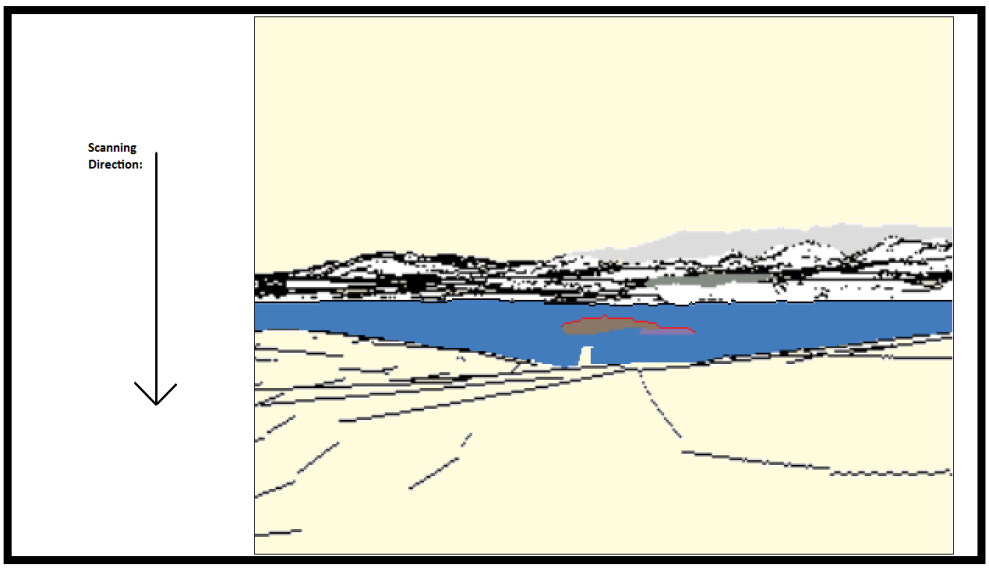


FIGURE 33: THE ISLAND UPPER LINE

- Segmentation image

In the Segmentation picture it is quite difficult to detect correctly the region that represents an Island individual. Two criteria are taken into account, the color of the region and the area of the region. By observing the segmentation images of a big set of nature digital photos, we saw that the color of the region which represents the Island belongs to the grayscale. Moreover, the area of the Island is already known from the 2D-model image.

A heuristic algorithm has been implemented for the detection of the Island region. The algorithm is called IslandDetector. It has been developed in order to detect regions with gray color and area equal to the area occupied by the "ISLAND" Individual in the 2D-model image.

The logic of the algorithm is the following:

→ Find the gray regions.
→ Calculate their area.
→ Find the gray region whose area is sufficiently equal to the area of the "ISLAND" Individual in the 2D-model image.

The IslandDetector algorithm returns one region which represents an Island area.

Figure 23 shows the Island Region detection. The contour of island is painted pink.

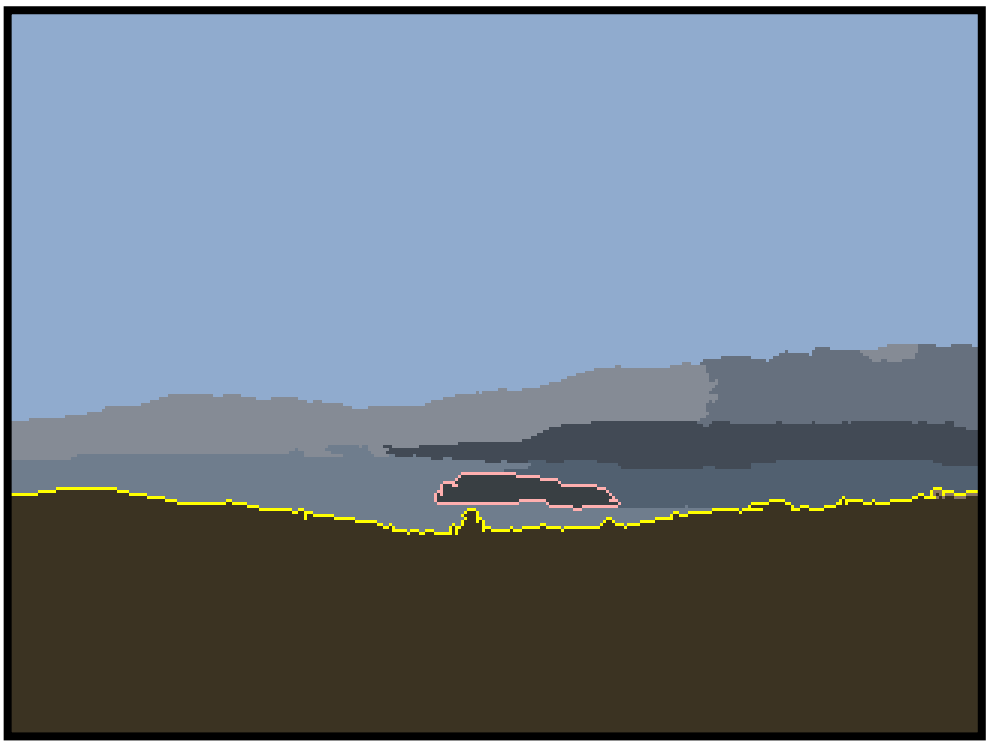


FIGURE 34: ISLAND REGION DETECTION

Then we scan this region bottom-up by one column of pixels at a time. For each column, we record the last Island pixel. Finally, we collect all the last pixels from all the columns. This collection of pixels forms the Island Upper line.

Figure 35 shows the procedure of Island Upper Line detection. The Island Upper Line is painted purple.

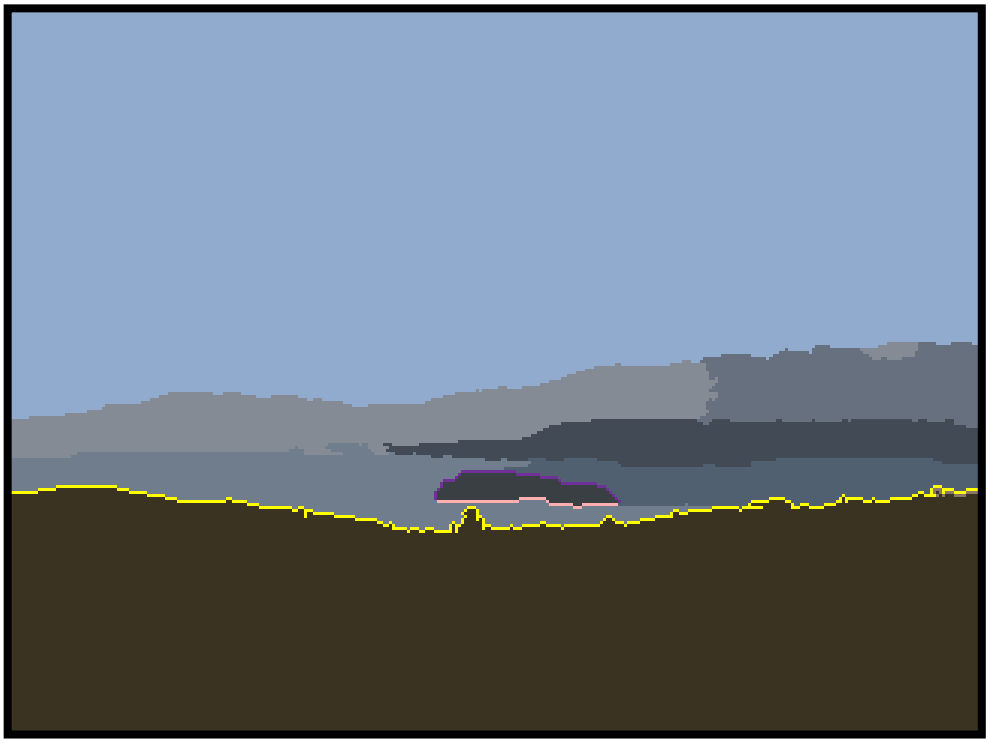


FIGURE 35: ISLAND UPPER LINE DETECTION

5.6 SUMMARY

This chapter presented the procedures which are followed in order to generate not only more inputs for the image registration algorithms but also more consistent inputs. The image registration is affected by algorithms which match the line segments. So, in this chapter it becomes clear which lines are used by the image registration algorithm and how they are detected.

6. IMAGE REGISTRATION ALGORITHM

Image Registration is the process of spatially aligning two or more images of a scene. The process in effect establishes a point-by-point correspondence between the images.

Typically, one image, called the **base** image or **reference** image, is considered the reference to which the other images, called **input** images, are compared. The image registration objective is to align the input image with the base image by applying a spatial transformation to the input image. The differences between the input image and the output image might have occurred as a result of terrain relief and other changes in perspective when imaging the same scene from different viewpoints. Lens and other internal sensor distortions, or differences between sensors and sensor types, can also cause distortion.

As we have already mentioned, the third part of the photo to semantic map conversion deals with Image Registration. It consists of the matching algorithm, which determines the transformation parameters and applies the transformation to the input image. The matching algorithm approximates the extracted boundaries of the individuals from both the photo and the 2D model with line segments. Then it tries to find the optimal match between the line segments. The optimal match provides the best fit transformation for transforming the 2D-model image to fit the photo properly. LineSegmentMatcher is called several times with slightly different parameters and the best result is kept. The matching method is the "Steepest Descent Local Search", as described in Beveridge and Riseman [57] and it is described in Section 5.1.

6.1 THE LINE SEGMENT MATCHING ALGORITHM

The 2D model image construction algorithm and the segmentation algorithm provide the boundaries of any individual as a set of pixel points. The LineSegmentMatcher functions with line segments, so the point sets must be approximated as line segments. The line segments corresponding to the point sets of the 2D model are called model line segments, while the line segments corresponding to the point sets from the segmentation algorithm are called data line segments.

The method used for approximating point sets as line segments is based on the method proposed by Mohamed Ali Said [58]. It works by computing the upper or lower convex hull of the point set as an initial approximation and computes an error function that compares the perpendicular distance of the points from the line segments that approximate them. If the distance is greater than the specified tolerance, the alternate convex hull is used for further approximation of the points.

Figure 36 shows how the algorithm works for the point set of Figure 36A. First, the upper convex hull is computed between the first and the last point, as shown in Figure 36B and provides a rough approximation. Since most of the points are distant from the approximated line segments, the alternate convex hull needs to be computed for the points between these line segments. The result is shown in Figure 36C. The algorithm continues calculating the alternate convex hulls until all the points are close enough to their corresponding approximated line segments, according to the error tolerance value. The upper convex hull for the first three points is computed in Figure 36D for a more detailed approximation. This is only an

example; in reality, the approximations should produce line segments that are long enough for the registration algorithm to work efficiently.

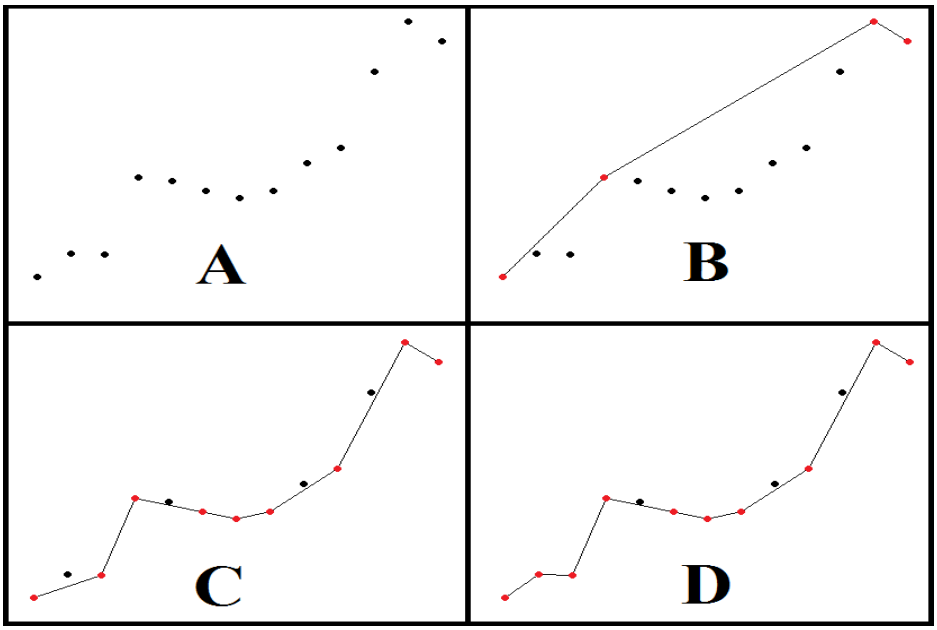


FIGURE 36: LINE SEGMENT APPROXIMATION EXAMPLE. A) INITIAL POINT SET. B) APPROXIMATION BASED ON COMPUTING THE UPPER CONVEX HULL. C) FURTHER APPROXIMATION AFTER COMPUTING THE LOWER CONVEX HULL. D) FINAL APPROXIMATION USING THE UPPER CONVEX HULL AGAIN FOR THE 3 FIRST POINTS.

After the approximation, an initial estimation of the matches between the model line segments and the data line segments along with the total error for this estimate is computed. This estimation is composed of a set of matched line segment pairs with each pair containing one model line segment and one data line segment. From now on the LineSegmentMatcher is ready to begin the iterations. In each iteration, a line segment pair is either added to (if it is not in the current estimation) or removed from (if it is already in the estimation) the matched set temporarily and the total error is again calculated for each case independently.

The addition and/or removal of the line segment pair that produces the least total error is permanently chosen for the best match for this iteration and is added or removed accordingly. The algorithm then moves to the next iteration and performs the same steps using the new set of line segment pair matches, until a constant number of iterations has passed or the algorithm converges, which means that the total error cannot decrease anymore.

The total error comprises of the following components:

Fit error: It measures how well the model line segments fit the data line segments. This error metric (E_P) measures and adds the perpendicular distance between each chosen line segment pair. The greater the distance, the greater the fit error becomes. The fit error value is computed according to (14):

$$E_F = \left(\frac{1}{L_D}\right) \sum_{i=1}^{h} \frac{l_i}{2} \left(u_{i1}^2 + u_{i2}^2\right) \quad (14)$$

L_D is the cumulative length over all the matched data line segments, l_i is the length of the i-th data line segment and h is the number of matched pairs. ui_1/u_{i2} are the perpendicular distances from the endpoints 1 and 2 of a data line segment i to the corresponding infinitely extended model line segment i. The derivation and details of this error metric are provided in Beveridge and Riseman [13]. The authors alternatively used the integrated squared perpendicular distance, which is not just endpoint based, but also takes into account the whole line segment. This distance was tried by LineSegmentMatcher with no success and the endpoint based distance was used instead. The authors also defined the omission error which penalizes matches where few model line segments participate in the final match. The omission error was not used for LineSegmentMatcher because the model line segments are not necessarily all correct and in some cases some of them need to be excluded from the final match without penalty.

Scale error: It discourages transformations that have a very large or very small scale factor s, since the 2D-model image has similar scale to the photo.

$$E_S = \begin{cases} \frac{1}{S} - r, & S < 1/r \\ 0.0, & 1/r \leq S \leq r \\ S - r, & S > r \end{cases} \quad (15)$$

The error function allows for a parameter r that allows a specific range of scale values not to produce any error at all. If the scale change grows or shrinks beyond r, then the error grows in proportion to the relative change in scale. The scale error is defined as the transformation error in Beveridge and Riseman [13].

Rotation error: It discourages unlikely rotations for the match. The higher the rotation, the larger the rotation error that is computed as shown in (16). This metric also allows a small range of rotation values to produce small error values:

$$E_R = \begin{cases} 0.1 * \vartheta, & |\vartheta| < r \\ 10 * \vartheta, & |\vartheta| \geq r \end{cases} \quad (16)$$

The parameter r and the angle ϑ are in degrees. The scale error is defined as the transformation error in Beveridge and Riseman [13].

The steepest descent local search iterates having as guides the above error metrics. One disadvantage of this method is that it can converge in local optima instead of the best fit. The fit error defines how the search finds the best fit, while scale and rotation errors help the algorithm avoid falling into local optima. When the iterations complete, a final fit transformation has been computed. It provides values for translating, rotating and scaling the 2D model image to fit properly in the user photo.

Finally, the fit transformation calculated from the aforementioned algorithm is applied on all the pixels of the 2D-model image. Then, the individuals are extracted from the image one by one by finding their bounding boxes, saving them as images and storing their coordinates. Afterwards, they are superimposed on the photo. Figure 37 presents the visualization of the newly created photo/semantic map using the familiar user photo of Figure 16. The individuals present in the 2D model are now placed on top of the photo in their correct location.

FIGURE 37: USER PHOTO AS A SEMANTIC MAP

6.2 SUMMARY

This chapter described how the image registration takes place. Specifically, it was explained how the correspondence between selected image features is determined and how the transformation parameters are derived from the corresponding feature points. It was also explained how the 2D-model image is transformed and overlaid on the base image.

7. EXPERIMENTS

This chapter describes the experiments we performed in order to improve the original algorithm of SPIM.

7.1 EXPERIMENTS WITH EDGE PROCESSING ALGORITHMS

The edge processing algorithms (see Chapter 4 for details) were developed in order to generate more data inputs (lines) for the image registration algorithm. This section shows the results of the image registration algorithm using these data inputs, on the images that comprise the training set.

Training Set Description

To begin with, the training set we used consists of 46 nature images. All of the 46 photos contained land areas right in front of the user as well as sea areas, but only 13 of them contained enlightened from the sun sea areas, 44 contained islands and 14 contained distinguishable ridgelines.

Table 3 shows the results of the application of the SkyDetector, LandDetector, SeaLightDetector and IslandDetector algorithms. In particular it shows in how many images of the training set, ridgeline, coastline, after the sea coastline and island line were detected correctly. In 30% of the images ridgeline is detected correctly whereas coastline is correctly detected in all (100%) images.

Individuals	Correctly Detected	Percentage
Ridgeline	14	30%
Coastline	46	100%
AftertheSea Coastline	13	100%
Island line	10	22%

TABLE 3: TRAINING SET RESULTS I

Experiments on the Training Set

Then we made the following image registration experiments on the training set:

1. Image registration supplying the matching algorithm only with the **ridgeline**.
2. Image registration supplying the matching algorithm only with the **coastline**.
3. Image registration supplying the matching algorithm with both the **ridgeline** and the **coastline**.
4. Image registration supplying the matching algorithm only with the **After the Sea Coastline**.
5. Image registration supplying the matching algorithm only with the **upper island line**.
6. Image registration supplying the matching algorithm with both the **coastline** and the a**fter the Sea Coastline**.
7. Image registration supplying the matching algorithm with both the **coastline** and the **upper island line**.

Our aim was to manipulate the matching algorithm to produce high quality semantic maps from the photos. The measure for determining the quality of the semantic maps is, besides the visual perception, the matching error value. The matching algorithm returns a fit transformation that is used to transform the points of the 2D model so that the 2D model fits properly in the user photo. Afterwards, the average vertical pixel distance between the corresponding lines that used by the matching algorithm is computed. The average vertical pixel distance error is used as a measure for determining the best call of the matching algorithm and consequently the best fit transformation. If its value is less than 2 then most of the times the semantic map is of good quality.

The table below summarizes the results for the five aforementioned experiments.

Experiment	#Photos with AverageVerticalPixelDistance error < 2
1	14
2	37
3	6
6	6
7	10

TABLE 4:TRAINING SET RESULTS II

Conclusions

The conclusions reached by these experiments are:

1. Image registration with the ridgeline is precarious in cases where the image context contains clouds, fog etc. In these cases, the ridgeline extracted from the segmentation image is not of a good quality and thus it affects the matching algorithm negatively. At this point it should be noted that the lines extracted from the 2D-model image are always of a good quality. Figures 44-47 are illustrative.

FIGURE 38: USER PHOTO

Figure 38 is a typical example proving the conclusion stated above.

It is obvious on Figure 39 that the ridgeline is not of a good quality whereas the coastline is.

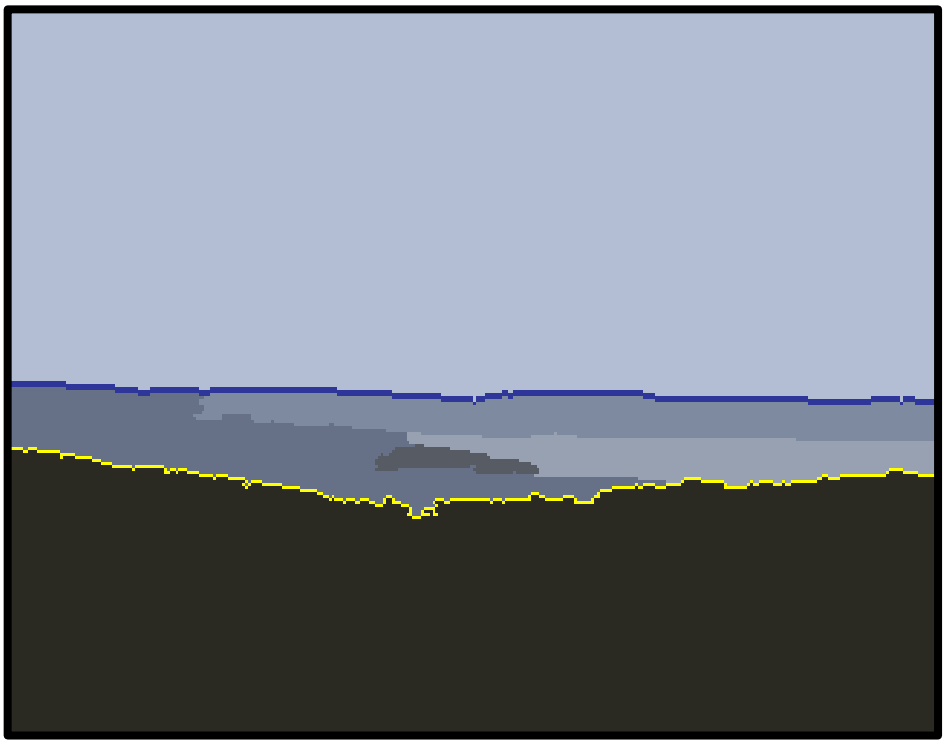


FIGURE 39: SEGMENTATION IMAGE

Figure 40 depicts the result of image registration using the ridgeline.

FIGURE 40: IMAGE REGISTRATION WITH RIDGELINE

Figure 41 shows the result of image registration using the coastline.

FIGURE 41: IMAGE REGISTRATION WITH COASTLINE

2. If image registration succeeds using either the ridgeline or the coastline, then the final result will be much better if both these lines are given as inputs to the matching algorithm. Figures 42-46 show this fact.

FIGURE 42: USER PHOTO

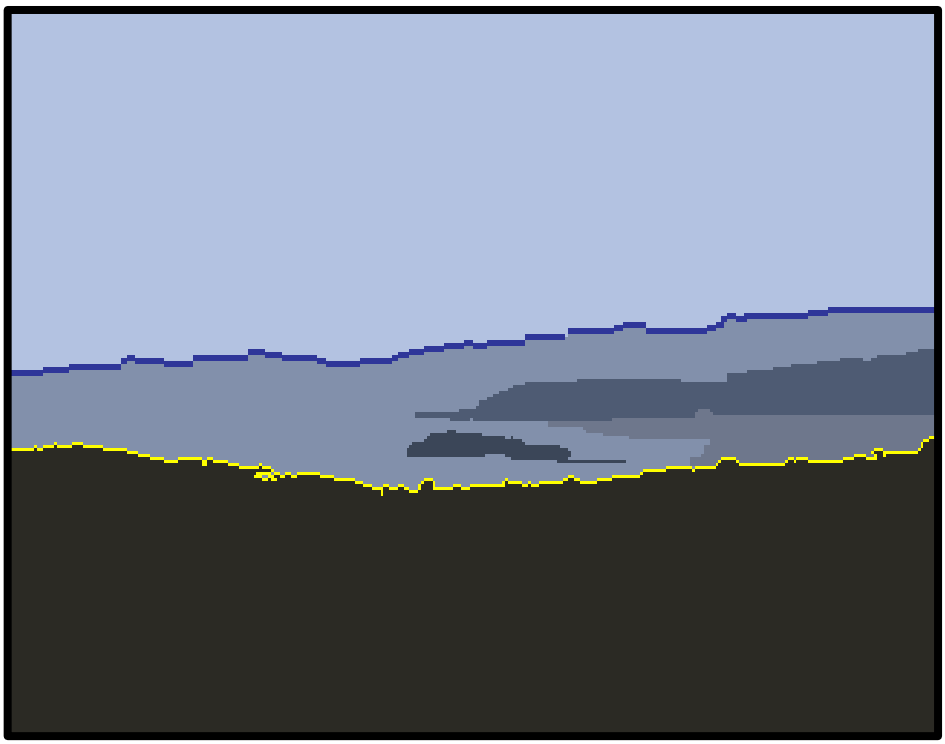


FIGURE 43: SEGMENTATION IMAGE

Figure 44 is the result of image registration using the ridgeline.

FIGURE 44: IMAGE REGISTRATION WITH RIDGELINE

Figure 45 is the result of image registration using the coastline.

FIGURE 45: IMAGE REGISTRATION WITH THE COASTLINE

Figure 46 is the result of image registration using both the ridgeline and the coastline. It is obvious that the semantic map created this way is of better quality.

FIGURE 46: IMAGE REGISTRATION WITH BOTH THE COASTLINE AND THE RIDGELINE

3. After the Sea Coastline and Island Upper line are lines with small length. Image registration supplying the matching algorithm either only with the After the Sea Coastline or with only the Island Upper line does not have acceptable results. A short line is divided into little line segments which are quite similar to each other and as a result the matching algorithm is confused and does not function properly. It cannot decide which segments must be matched as all the segments look alike. Figures 47-49 confirm this conclusion.

FIGURE 47: USER PHOTO

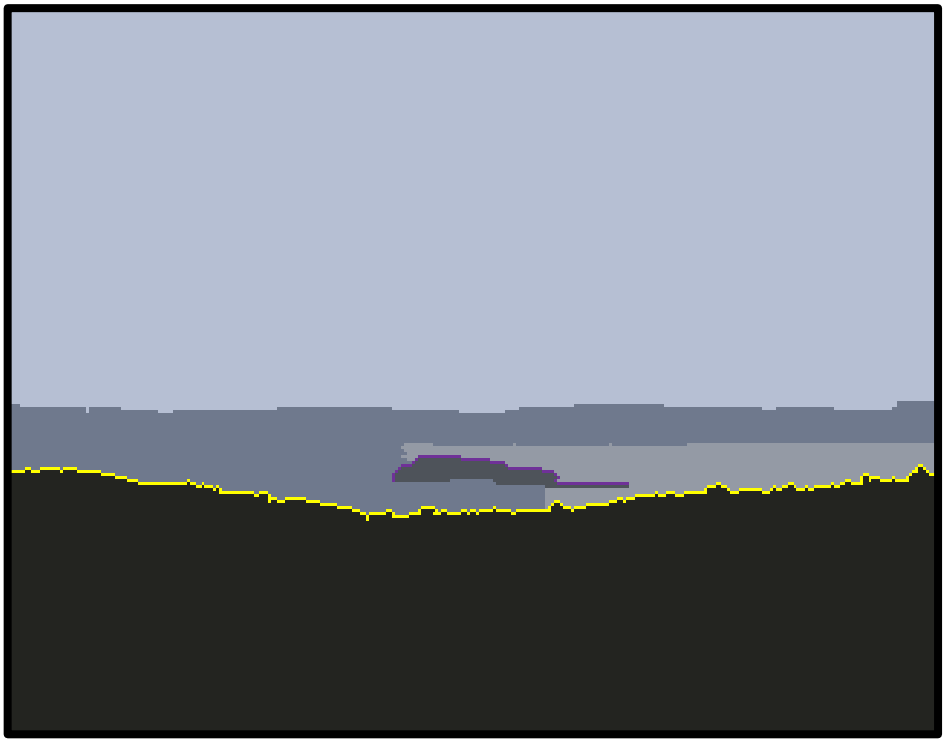


FIGURE 48: SEGMENTATION IMAGE

FIGURE 49: IMAGE REGISTRATION WITH THE ISLAND UPPER LINE

4. Image registration supplying the matching algorithm with both a long line (ridgeline or coastline) and a short line (Island Upper line or After the Sea Coastline) of good quality has better results. Thus, it is confirmed that supplying the matching algorithm with two lines than one, produces better results. Figures 50-57 are illustrative.

Image Registration with both the coastline and the island upper line:

FIGURE 50: USER PHOTO

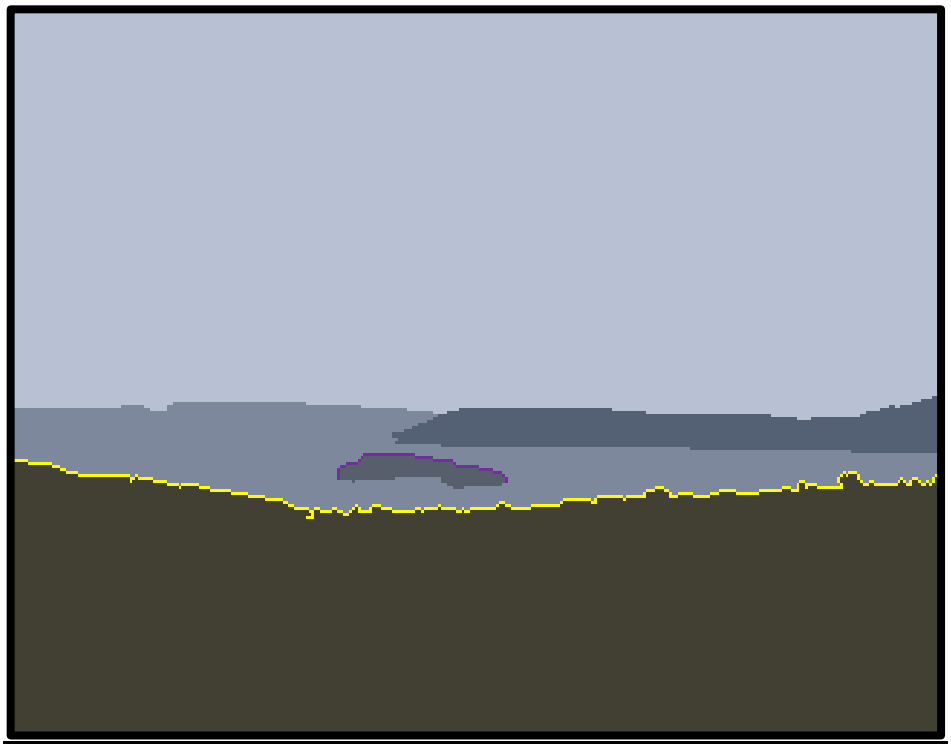


FIGURE 51: SEGMENTATION IMAGE

FIGURE 52: IMAGE REGISTRATION WITH ONLY THE COASTLINE

FIGURE 53: IMAGE REGISTRATION WITH BOTH THE COASTLINE AND THE ISLAND UPPER LINE

Image Registration with both the coastline and the After the Sea Coastline:

FIGURE 54: USER PHOTO

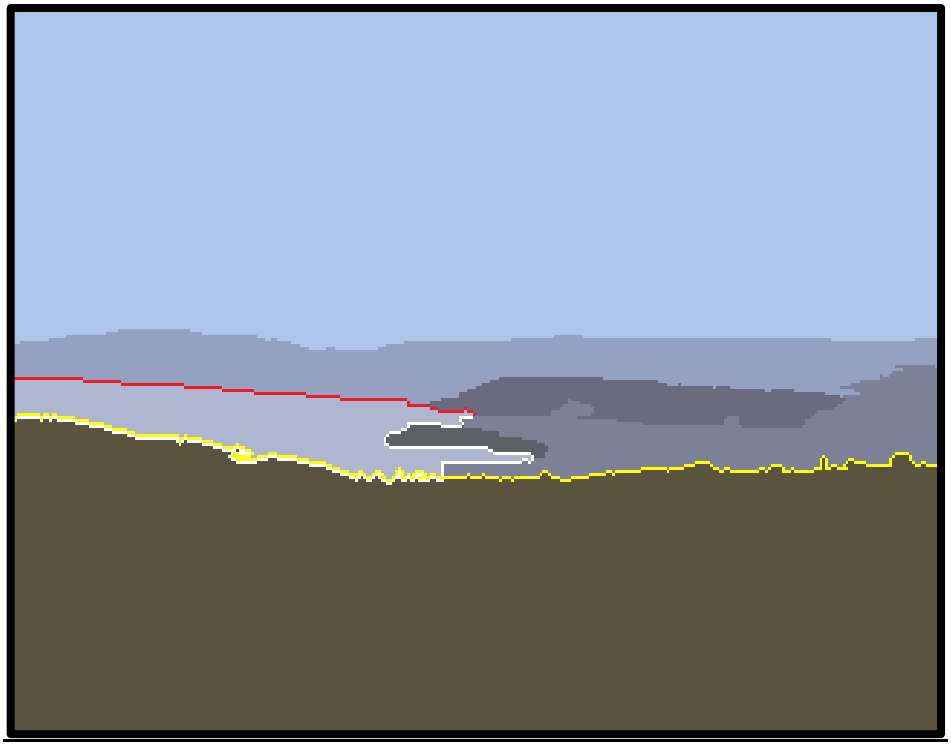


FIGURE 55: SEGMENTATION IMAGE

FIGURE 56: IMAGE REGISTRATION WITH ONLY THE COASTLINE

FIGURE 57: IMAGE REGISTRATION WITH BOTH THE COASTLINES

7.2 MULTIPLYING THE IMAGE HEIGHT

Multiplying the height of the image and keeping its width fixed leads to the creation of an oblong image. The boundaries of the semantic individuals extracted from such an image are lines with excessive ups and downs. Eventually, the line segments that shape such a line are quite different to each other. This experiment tests if these refitted line segments help the matching algorithm to function more properly.

Experimental Setup

At first, we tried to multiply the height dimension of 5 images. These images have distinguishable ridgelines. When the image has its initial dimensions, image registration with the ridgelines occurs with satisfactory success.

Then we resized the images by multiplying their height with a factor (2,3,4,5,6,7) . For each factor, we provided the matching algorithm with the new "wavier" ridgelines. The matching algorithm then returned a fit transformation that was used to transform the ridgeline of the 2D model so that the 2D model fits properly in the user photo. Afterwards, the average vertical pixel distance between the corresponding ridgelines was computed. This was used as a measure for determining the best call of the matching algorithm and consequently the best fit transformation.

Table 5 shows the average vertical distance match error for every multiplication.

Photo	x1	x2	x3	x4	x5	x6	x7
#367	0,95	1,26	0,85	0,92	**0,75**	1,29	1,48
#355	2,12	0,68	**0,59**	0,73	1,16	1,14	2,05
#380	1,01	0,70	1,03	**0,61**	1,43	1,13	1,84
#382	1,37	1,42	2,95	**0,65**	0,93	1,05	0,67
#384	1,86	0,82	1,63	0,51	**0,49**	0,89	3,65

TABLE 5: MULTIPLICATIONS AND AVERAGE VERTICAL DISTANCE MATCH ERROR

The column x1 refers to the errors that occur when the image has its initial dimensions. Column x2 includes the errors that occur when the height of the image is multiplied by 2, column x3 includes the errors that occur when the height of the image is multiplied by 3 etc.

As we can see, multiplying the height of the image by up to 5, gives us smaller match errors. Multiplication with 6 and 7 did not give any better results so that led us to stop the research at this point and not try multiplication with 8, 9, 10 etc.

Figures 58-60 demonstrate the above procedures.

FIGURE 58: USER PHOTO

FIGURE 59: IMAGE REGISTRATION WITH THE INITIAL RIDGELINES

FIGURE 60: IMAGE REGISTRATION WITH THE MULTIPLIED BY 3 RIDGELINES

Figure 60 shows that matching the multiplied by 3 ridgelines has produced a semantic map of better quality than this of Figure 67. Focus on the ridgeline and on how well the layer that represents the mountains has covered the "real" mountains of the photo.

This technique has an effect only on ridgelines, since the ridgelines are "wavy" lines from the beginning. What happens to them is that they become wavier. The coastlines consist of long horizontal segments and by multiplying them the only thing that changes is the y-coordinate value of their points and not their shape.

7.3 EXPERIMENTS ON THE Q PARAMETER

As we have already mentioned, image segmentation is attained using the statistical region merging method by Nock and Nielsen[56]. It concerns a fast and robust algorithm that can be parameterized in order to segment an image into regions of similar color.

The algorithm starts with one region per pixel and then applies a statistical test on neighboring regions. If the color expectations of neighboring regions are sufficiently similar enough then the regions are merged. This process continues until all the pixels are allocated to a region.

An interesting parameter of the algorithm is the Q parameter. The tuning of this parameter allows controlling the number of the regions in the segmented image. The smaller the value of Q is, the smaller the number of the regions in the resulting segmentation image. Another remarkable fact is that as Q increases, the regions found are getting smaller, but they represent more accurately the real regions of the initial image.

Apparently, the Q parameter plays a decisive role in the outcome of the segmentation algorithm. The more precise the result of the segmentation is, the more accurately the image registration will be done. In order to see how the tuning of parameter Q influences the performance of the image registration, we conducted a series of experiments.

Figure 61 shows some segmentation images which have different values of the Q parameter.

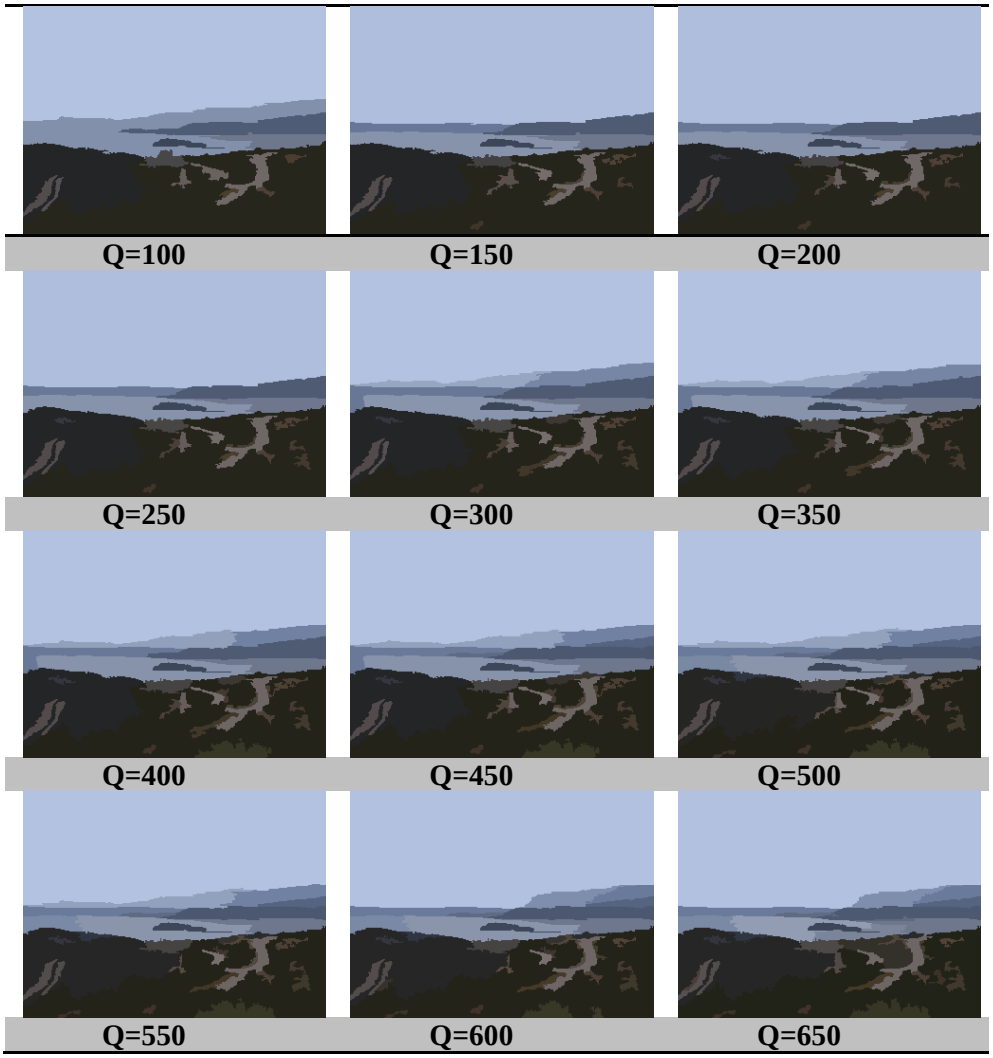


FIGURE 61: SEGMENTATION IMAGES WITH DIFFERENT Q

Experimental Setup

The experiment is based on a dataset of 40 nature images. The Q parameter takes values from zero to one thousand (0-1000). The Q value that the SPIM system uses is Q=100. Having as an incentive to see if increasing the Q value we will get better results, we decided to evaluate the system performance for the following Q values: 150,200,…,650,700. For each value of Q we recorded the performance of the image registration algorithm.

The evaluation results and the conclusions drawn are presented in the diagrams and the discussions that follow.

Conclusions

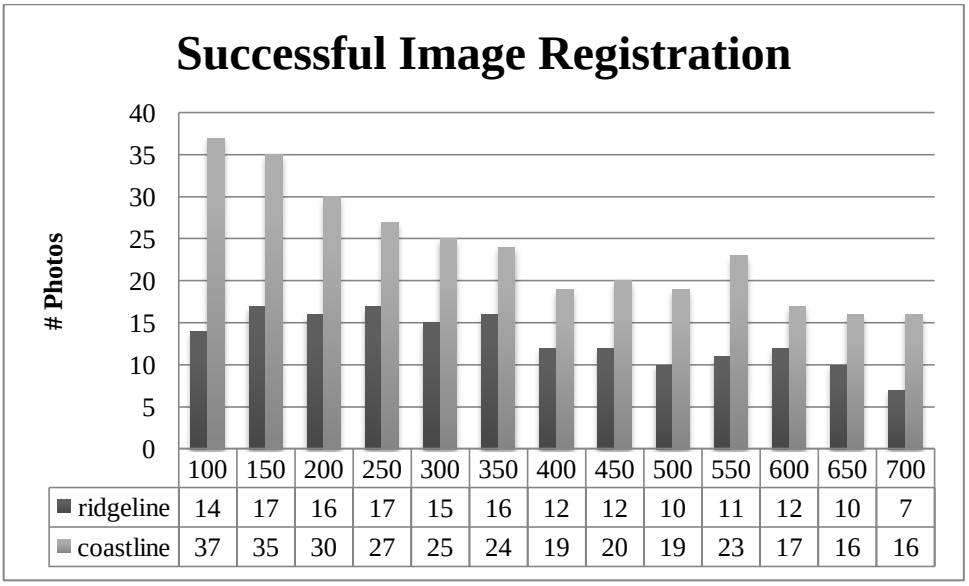


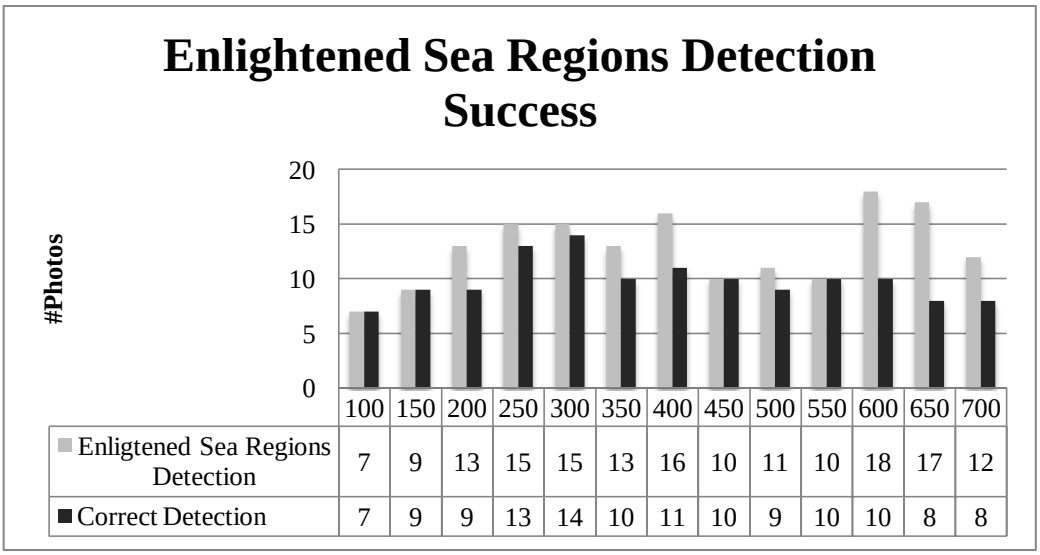
FIGURE 62: SUCCESSFUL IMAGE REGISTRATION

The diagram of Figure 62 shows the number of successively registered images for each value of the Q parameter, using for the line matching either the ridgeline or the coastline.

Notes:

- When the value of the Q parameter increases, the segmentation image of each photo obtains more regions with a slight color difference among them and smaller size. This causes many conflicts. The most important is that the detection algorithms (sky, sea, land) face greater difficulty to converge. This way the System is being less stable.
- For the case of Image Registration using the coastline, Q=100 is proved to be the better choice, since 37 photos out of 40 are registered successively.
- For the case of Image Registration using the ridgeline, Q=150 or Q=250 are proved to be better choices, since 17 photos out of 40 are registered successively while with Q=100 only 14 photos out of 40 are registered successively.
- For Q values between 350 and 700 the performance of the System falls off.

The following diagram shows the number of photos in which the enlightened sea regions were successively detected for each Q parameter value.



	100	150	200	250	300	350	400	450	500	550	600	650	700
Enligtened Sea Regions Detection	7	9	13	15	15	13	16	10	11	10	18	17	12
Correct Detection	7	9	9	13	14	10	11	10	9	10	10	8	8

FIGURE 63: ENLIGHTENED SEA REGIONS DETECTION SUCCESS

Notes:

- The Enlightened Sea Regions have the characteristic that theirs lightness value remains constantly the highest of all of the other regions. So the detection of them does not easily fail.
- As the Q parameter increases the size of the regions is getting smaller and that is the reason why the success falls off. A smaller region with the highest lightness value in the segmentation image not always represents a correct region of real Sea.
- For Q=300 we take better results. Correct detection takes place in 14 out of 40 photos.

The following diagram shows the number of photos in which island region was successively detected for each Q parameter value.

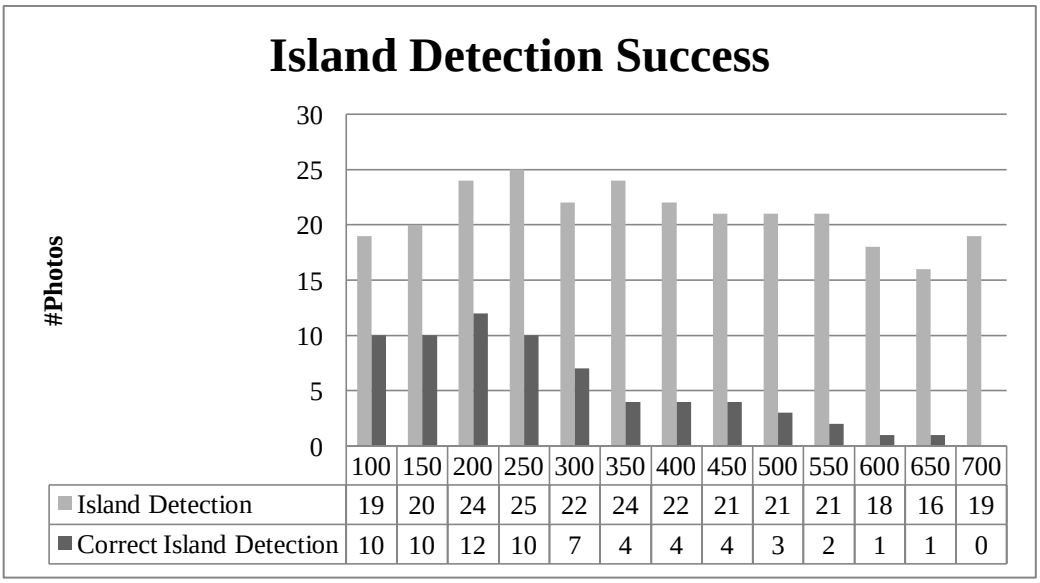


FIGURE 64: ISLAND DETECTION SUCCESS

Notes:

- The Island Region is detected using the criterion of its area, based on the percentage of pixels it covers over the sum of pixels of the photo. This is a rather precarious criterion because as the Q parameter increases, the size of the regions is getting smaller and their number is getting bigger. This fact brings confusion to the Island Detection Algorithm. It cannot decide which region is the region that represents the Island region.
- For Q=200 we take better results. Correct detection takes place in 12 out of 40 photos.

7.4 THE SPIM+ MAIN ALGORITHM

This section introduces as the new SPIM algorithm, the SPIM+ main algorithm. This algorithm arose from the conclusions reached by all the aforementioned experiments.

The SPIM+ Main Algorithm incorporates all the conclusions that we have gathered. It is called 5 times, one for each Q value. In the beginning it tries image registration with every line that has been detected. It estimates the final result by calculating the average vertical pixel distance error and if the value of this error is less than 2, then the corresponding line is of good quality. Afterwards it stores the lines which have acceptable results.

If it finds only one line of good quality then it does the image registration with this line and returns this way the best result.

If it finds more than one line of good quality, it tries image registration with every combination of lines. It calculates the new average vertical pixel distance error and returns the result with the minimum error.

If among these lines is the ridgeline, then it multiplies its y-coordinates by 1,2,3,4,5 and tries image registration with every multiplied ridgeline.

Description of the algorithm is following.

Algorithm 3 – SPIM+ Main Algorithm

```
Inputs: ridgeline, coastline, afterTheSeaCoastline, islandUpperLine

For (int Q=100; Q<350; Q+=50;){
        For each line → Do image registration
                        Calculate the error
                        If error < 2 → Store this line or these lines

        If #lines >= 2 {
                If (ridgeline Ǝ lines) →
                        For (int i=1; i<6; i++) {
                        Multiply y-coordinates of the points of the ridgeline by i
                        Use the new ridgeline
                        Do image registration with every combination of the
                        available lines
                        Calculate the error
                        }

                Store the combination that produces the minimum error
                Return the corresponding result
                }
        Else
                Return the result that corresponds to image registration with one
                line that has the minimum error.
        }
```

7.5 SUMMARY

This chapter presented the experiments we conducted in order to reach some conclusions, useful for improving the performance of the initial SPIM system. Finally, the SPIM+ Main Algorithm which derived from the conclusions we made, was presented.

8. SUPPORT FOR INEXPENSIVE CAMERAS

This chapter describes the functionality added in the initial SPIM system in order to support the processing of images that do not have in their metadata information for compass measurement.

There are two reasons that led us to add this functionality in the system:

1. Nowadays, the market is flooded with Smart phones. Smart phones are equipped with digital camera and GPS receiver. So, they are capable to capture images and add geographical identification metadata to them. These data consist of latitude and longitude coordinates, though they can also include altitude, bearing, distance, accuracy data, and place names. What Smart phones produce is geotagged photographs. Even if applications for a digital compass have already been developed, they cannot be exploited since they cannot integrate the appropriate direction measurement in the image metadata.

2. Buying a digital camera with integrated GPS receiver and digital compass is rather costly. Since buying a digital camera only with an integrated GPS receiver is a more economical choice it is more likely that the consumers will purchase this version of the product.

8.1 METHODOLOGY

This section describes the methodology followed by the SPIM+ framework in order to support inexpensive cameras. The procedure uses the main parts of the SPIM architecture and in particular the parts of Viewshed Calculation and 2D model image creation.

The basic idea, since we do not have compass data, is to design an image that depicts what we see from the given location (GPS data) by looking simultaneously towards all the possible directions. In other words, this image depicts the panoramic view from the given location and is the new 2D model image used for image registration. From now on, this new 2D model image is referred as Panorama.

8.1.1 PANORAMA CALCULATION

It should be mentioned at first that when the metadata (Exif data) of an image does not include a value for the tag GPSDirection (which shows the Compass direction), then the default value for this tag is 0^o.

The algorithm for the Panorama Calculation, traces rays that begin from the user, follow every possible discrete direction of the camera and are within the full angle of view (0^o – 360^o). It takes as inputs the headings to the start (0^o) and end (360^o) of the full angle of view, the elevation and GPS position of the point where the user is standing, the number of rays to be cast and the maximum distance for each ray to search for visible areas. All the information about each ray cast is stored in appropriate data structures and is used for drawing the Panorama.

8.1.2 PANORAMA DRAWING

Having completed the Panorama calculation, the algorithm has collected all the necessary information to create the Panorama image. The "canvas" is a white image with width equal to the number of rays cast for the Panorama calculation and height proportional to the vertical angle of view of the image.

Since the width of the "canvas" is equal to the number of rays cast, each column of pixels of a "canvas" corresponds to one ray. For each visible point found in a ray, the view tangent has been calculated. The first ray to be processed is the ray which falls upon the central column of pixels of the "canvas" (if the resolution of the Panorama image is (x,y) then the central column of pixels is at x/2). The first pixel to be drawn is the central pixel of the "canvas", with coordinates (x/2, y/2). Then all the objects with positive view tangent are located above that pixel and all the objects with negative view tangent are located below it. Figures 65-68 show a user photo, its Panorama image and its 2D model image (assuming that its GPSDirection tag has a value).

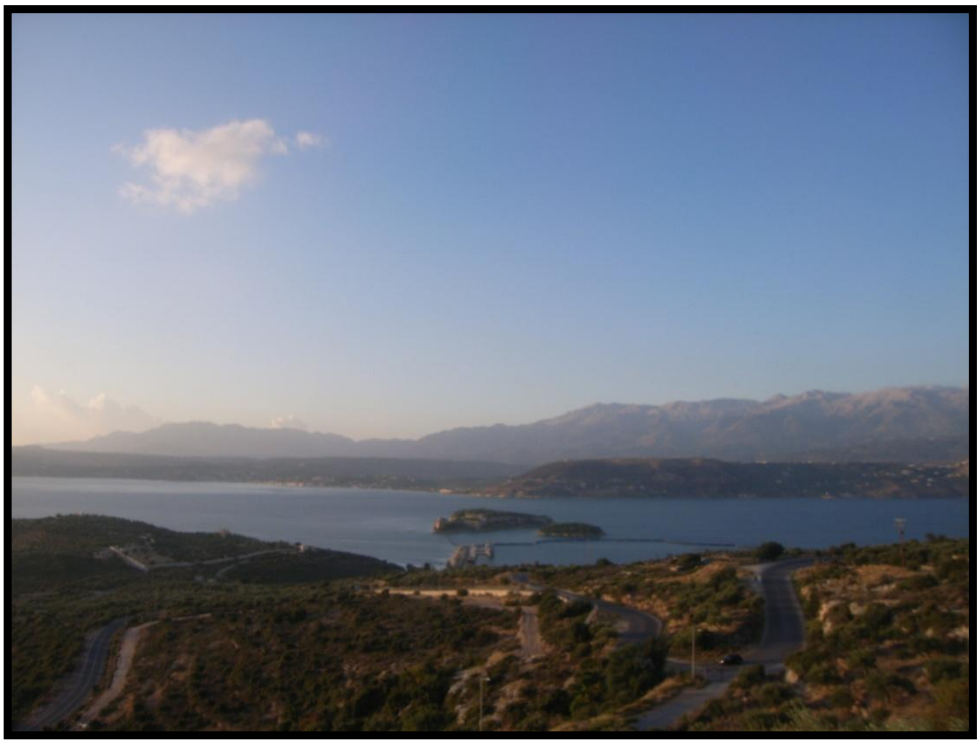


FIGURE 65:USER PHOTO

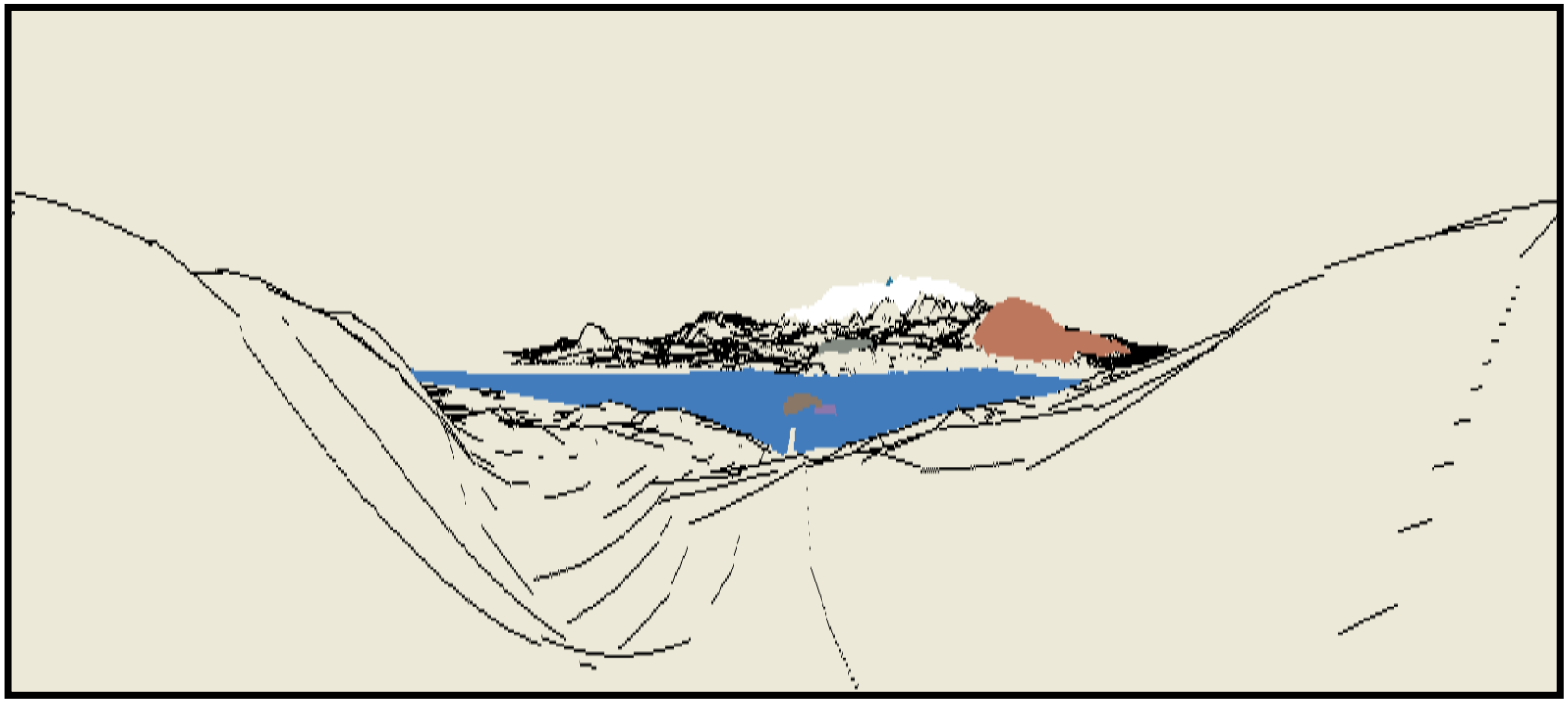


FIGURE 66: PANORAMA IMAGE

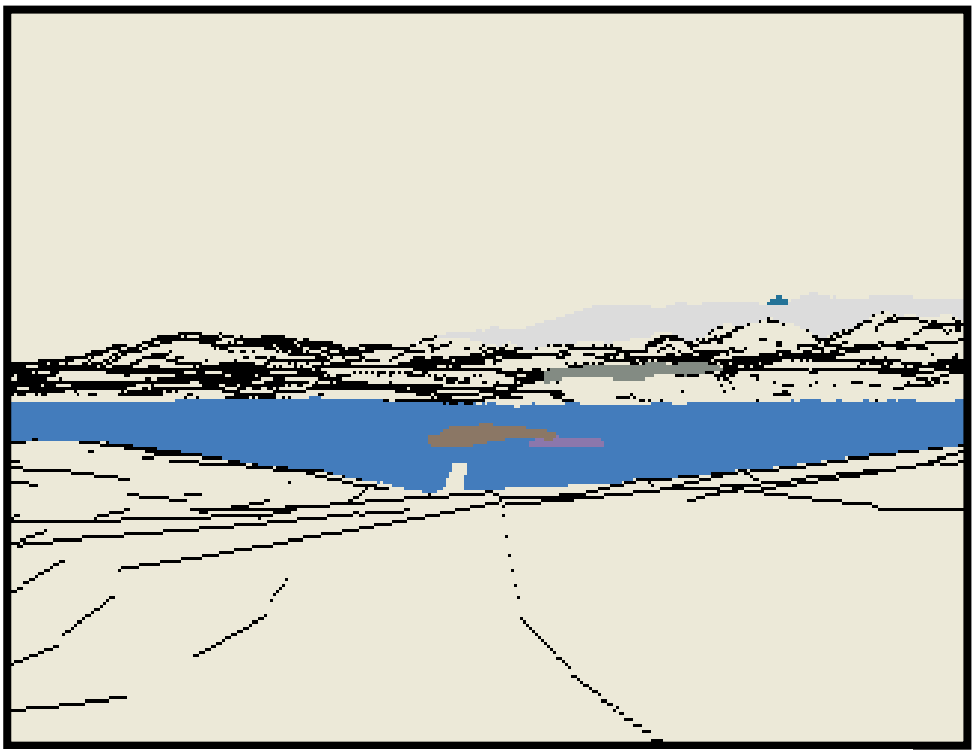


FIGURE 67:2DMODEL IMAGE GIVEN THE DIRECTION

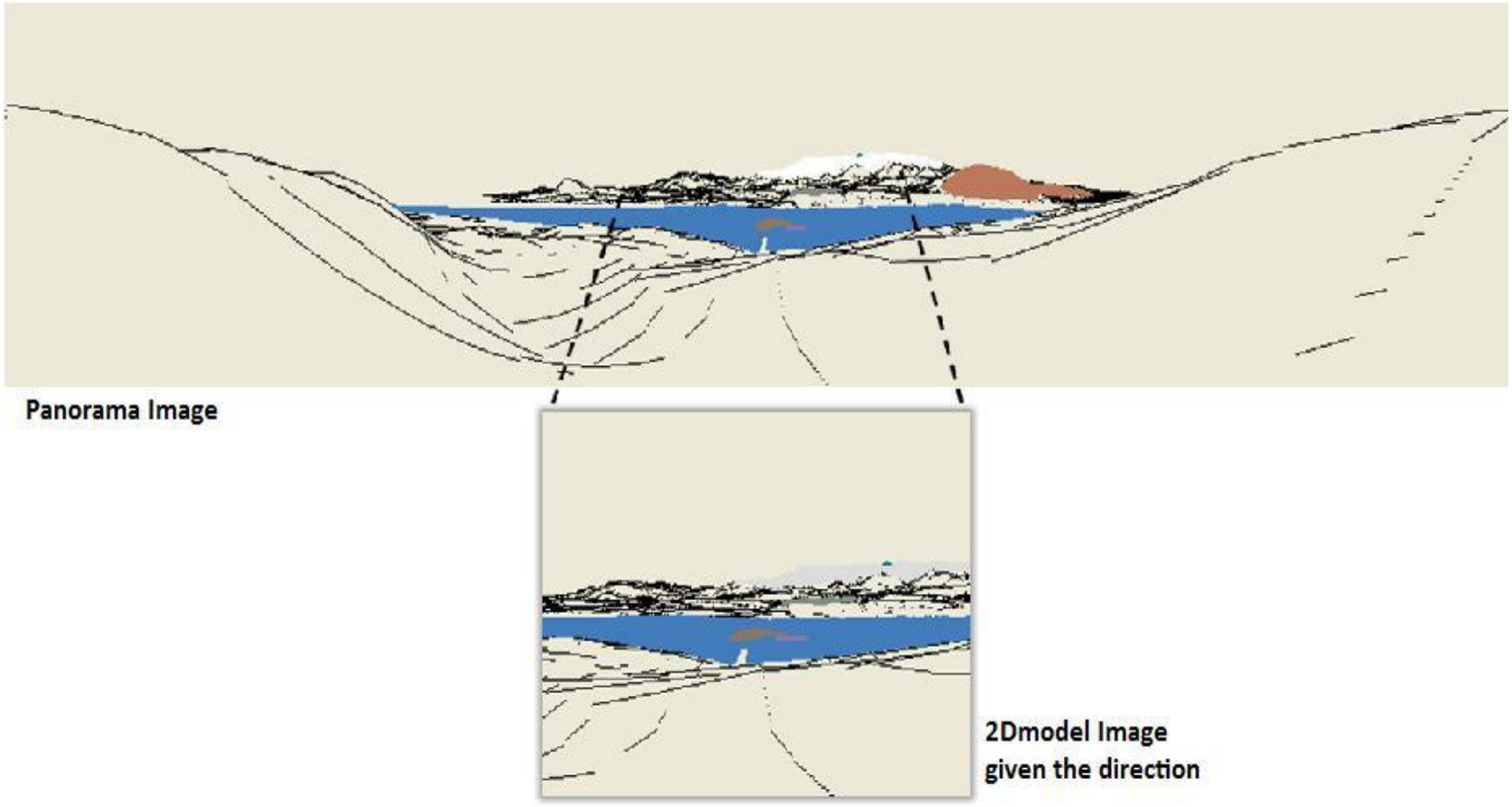


FIGURE 68:RELATION BETWEEN PANORAMA IMAGE AND 2DMODEL IMAGE

8.2 IMAGE REGISTRATION USING PANORAMA

In this case too, Image Registration is carried out by the Line Segment Matcher algorithm (the matching algorithm that the SPIM framework uses) which is modified (see Algorithm 3).

The Line Segment Matcher algorithm approximates the boundaries of the semantic individuals from both the photo and the 2D-model image with line segments. Then it tries to find the optimal match between them. The optimal match provides the best solution for transforming the 2D-model image to fit the photo as correctly as possible.

In particular, at this point we have the boundaries of the semantic individuals from the Panorama construction and Segmentation algorithms. The problem that occurs is that the Panorama image and the Segmentation image no longer have the same dimensions: Obviously, the Panorama image is larger. Consequently, the boundaries of the semantic individuals from the Panorama image are longer than the boundaries of the semantic individuals from the Segmentation image. More precisely, the boundaries of the semantic individuals from the Segmentation image are part of the boundaries of the semantic individuals from the Panorama image.

Figure 69 shows the relation between the boundaries of the semantic individuals from the Panorama image and the boundaries of the semantic individuals from the Segmentation image.

The matching algorithm takes as inputs these peculiar boundaries, normal from the Segmentation image and longer from the Panorama image. To overcome this peculiarity, the algorithm cuts the long line (Panorama image) in segments of equal length to the normal line (Segmentation image). Afterwards, it checks which of these segments matches more precisely with the appropriate line from the Segmentation image. When the algorithm returns, the right part of the Panorama has been found and using it the image registration takes place. The optimal match between the line segments of the Segmentation image and the correct part of the Panorama image gives the best fit transformation for transforming the 2D-model to fit the photo. Figure 70 shows the result of Image Registration.

Algorithm 4 – Modified LineSegmentMatcher

```
int P_Width; // width of Panorama image

int S_Width; // width of Segmentation image

int PminusS; // the difference between P_Width and S_Width

int i,j; // the points to be thrown away from the Panorama boundaries, i refers to points from the left and j to points from the right

double minMatchingError = Double.MAX_VALUE;

for(i = 0; j = PminusS; i <= PminusS; i+ = 1; j- = 1;)

{

  Throw Away i points from left and j points from right

  Panorama Boundaries and Segmentation Boundaries have same length

  TransformationResults = LineSegmentMatcher (PanoramaBoundaries, SegmentationBoundaries);

  MatchingError = calculateMatchingError();
```

```
if (MatchingError < minMatchingError){
        minMatchingError = MatchingError;
        bestFitTransformation = this.TransformationResults
        }
}
```

8.3 SUMMARY

It was shown in this chapter how a photo that does not have in its metadata information for compass measurement can be transformed into a semantic map. The procedure includes finding a 2D model of the panoramic view from the camera position and segmenting the photo into regions. Then the lines extracted from both these images are processed by the matching algorithm, after a modification, in order to provide a fit transformation to apply on the 2D model and overlay the right footprints of the individuals on top of the photo. After these steps the semantic map is ready.

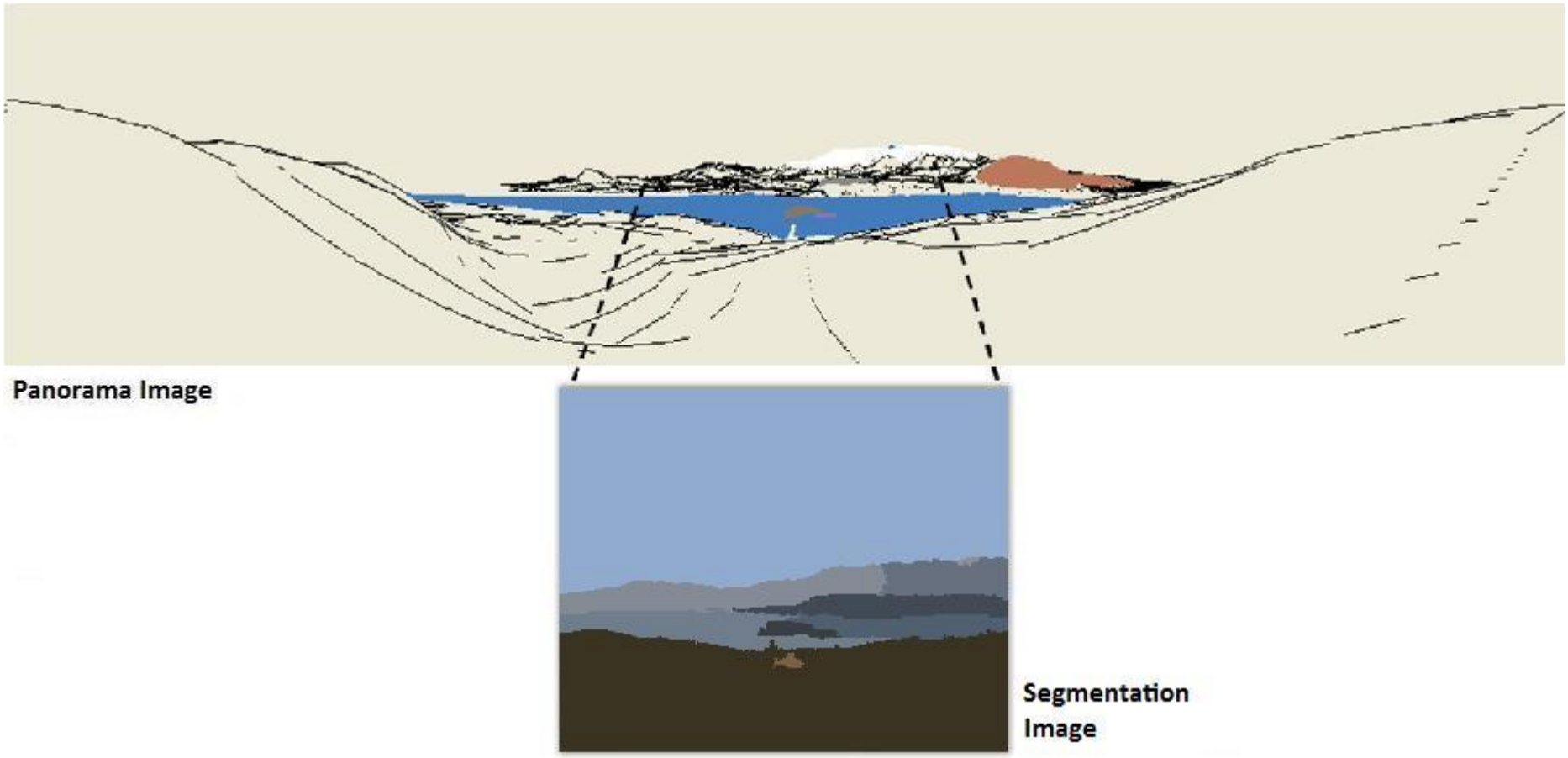


FIGURE 69:RELATION BETWEEN PANORAMA IMAGE AND SEGMENTATION IMAGE

FIGURE 70:IMAGE REGISTRATION USING PANORAMA MODEL

9. EVALUATION OF THE SPIM+ ALGORITHMS

This chapter describes the evaluation of the algorithms developed in the SPIM+ framework. For this purpose, photos were captured and given as input to the system. The evaluation procedure tested the image registration precision and the quality of the semantic maps produced from the photos.

9.1 EVALUATION OF SPIM+ MAIN ALGORITHM

The evaluation procedure used the edge detection algorithms and the SPIM+ Main algorithm on the dataset.

The dataset we used consists of 50 nature images. From the 50 photos, all contained mountains and land areas right in front of the user, 48 contained sea areas but only 15 contained enlightened from the sun sea areas and 21 contained islands.

A subset of the dataset of the nature images is shown in Figure 71.

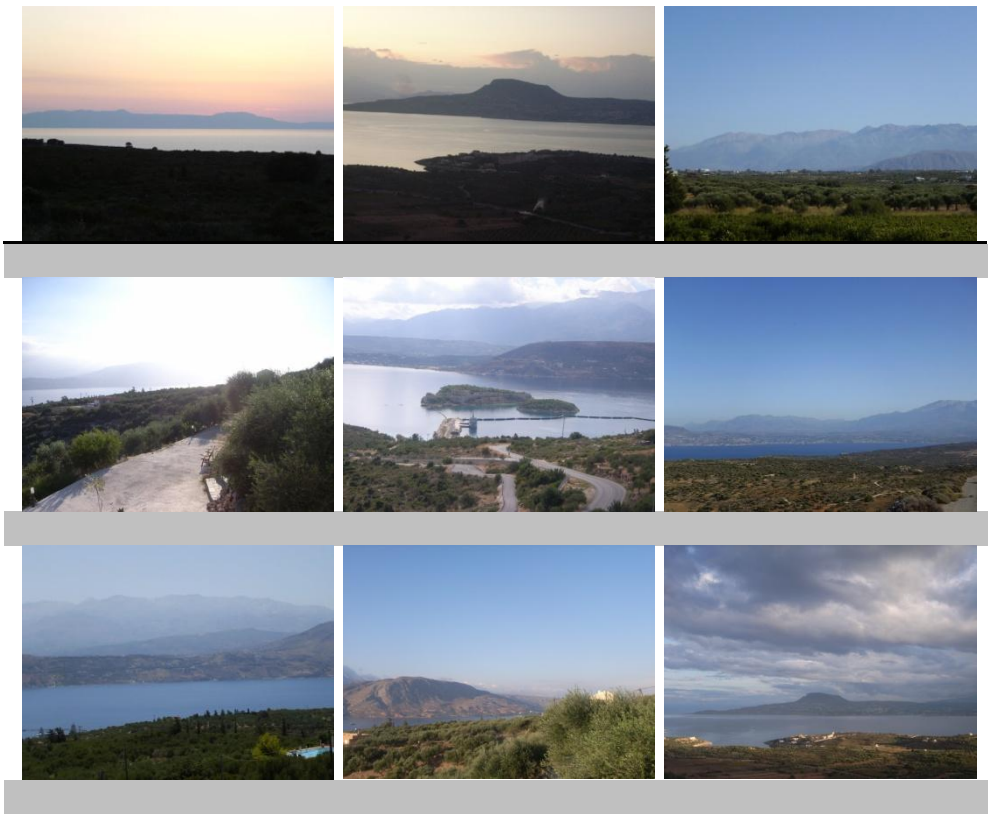


FIGURE 71: SUBSET OF THE NATURE IMAGES

The evaluation results are presented below, followed by comments and discussion on them.

Ridgeline Detection

Detecting the ridgeline (in other words skyline) from the 2D view models was a trivial task; the difficulty was in segmenting the image correctly to ensure that the skyline can be extracted without significant errors. The segmentation algorithm tried to detect and merge the multiple regions representing the sky and also remove obstacles such as clouds.
Applying the segmentation algorithm on the 31 images of the dataset yielded the following statistics:

- 23 complete failures, caused by the sky detection algorithm due to low visibility of the distant mountains.
- 27 satisfactory ridgeline (skyline) detections

Overall, the ridgeline detection technique worked for the 58% of the images.

Coastline Detection

Once again, coastline detection from the 2D view models was done without complications.
The results of the coastline detection from the segmentation images of the 31 images of the dataset are below:

- 5 complete failures, caused by the land detection algorithm due to the fact that the land area in front of the user contained gray stones.
- 45 images had satisfactory coastline detection.

Overall, the coastline detection technique worked for the 84% of the images.

After the Sea Coastline Detection

Once again, After the Sea Coastline detection from the 2D view models was done without complications.
The results of the coastline detection from the segmentation images of the 6 images of the dataset are below:

- 3 complete failures, caused by the land detection algorithm because the lightness value of the regions is less than 50.
- 12 images had satisfactory After the Sea Coastline detection.

Overall, the coastline detection technique worked for the 50% of the images.

Resulting Semantic Map Quality

The resulting semantic maps were divided into 7 categories, each corresponding to an adjective describing the quality of the semantic map. From the best to the worst they were: *Perfect*, *Good*, *Acceptable*, *Average*, *Bad*, *Awful* and *Failed*. The first four were considered "pass" grades, while the other three "fail" grades. We gave a grade to each adjective: 5 (Perfect), 4 (Good), 3 (Acceptable), 2 (Average), 1 (Bad), 0 (Awful and Failed). Then we calculated the average score.
Each resulting semantic map was inspected thoroughly in terms of how well the individual footprints were matched to the picture contents. The whole process was done in an unbiased way. Table 6 summarizes the results of the evaluation.

Distinction	#Photos
Perfect	6
Good	15
Acceptable	11
Average	10
Bad	6
Awful	1
Failed	1
Pass	42
Fail	8
Total	50
Pass Percentage	84%
Fail Percentage	16%

TABLE 6: RESULTS OF SPIM+ MAIN ALGORITHM

The average score of SPIM+ Main Algorithm is 3.

Figure 72 shows a sample subset of the produced semantic maps and their rank.

FIGURE 72: THE PRODUCED SEMANTIC MAPS AND THEIR RANK

9.2 COMPARISON OF SPIM+ MAIN ALGORITHM WITH SPIM ORIGINAL ALGORITHM

We compared the performance of SPIM+ Main algorithm over the performance of SPIM Original algorithm using the same dataset of nature images.

As we have already mentioned the SPIM Original algorithm transforms the natures images to interactive ones using only the ridgeline. Consequently its success is expected to be limited compared to the success of the SPIM+ Main algorithm.

The resulting semantic maps by SPIM Original Algorithm were inspected in the same way as described in section 8.1. Table 7 summarizes the results of the evaluation.

Distinction	#Photos
Perfect	6
Good	9
Acceptable	6
Average	6
Bad	4
Awful	4
Failed	15
Pass	27
Fail	23
Total	50
Pass Percentage	54%
Fail Percentage	46%

TABLE 7: RESULTS OF SPIM ORIGINAL ALGORITHM

The average score of SPIM Original Algorithm is 2.1.

The diagram following shows the success of the SPIM and SPIM+ algorithms in comparison to one each other. The horizontal axis represents the percentage of the images that were transformed to interactive by SPIM and SPIM+.

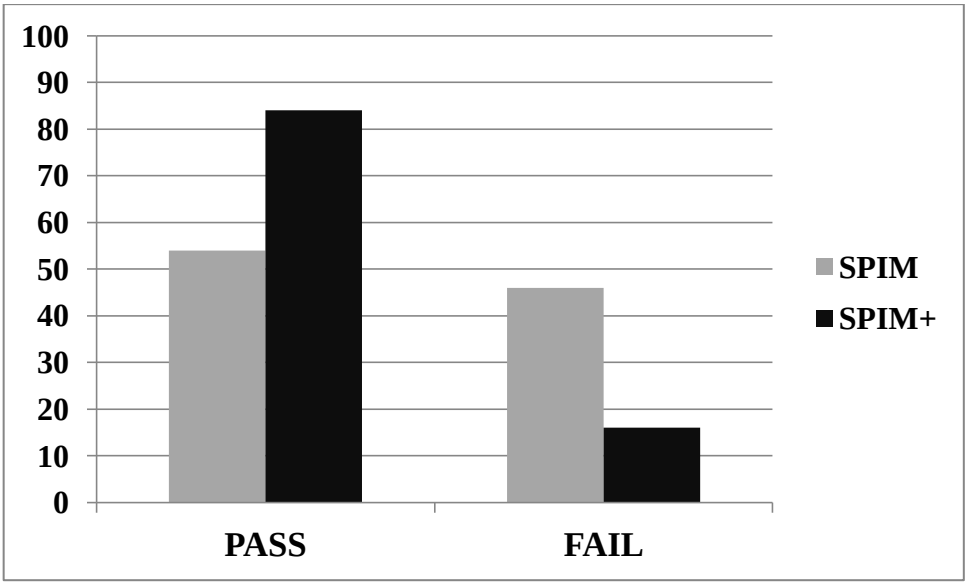


FIGURE 73:SPIM VS SPIM+ SUCCESS

The following diagram shows the average score of SPIM in comparison with the average score of SPIM+

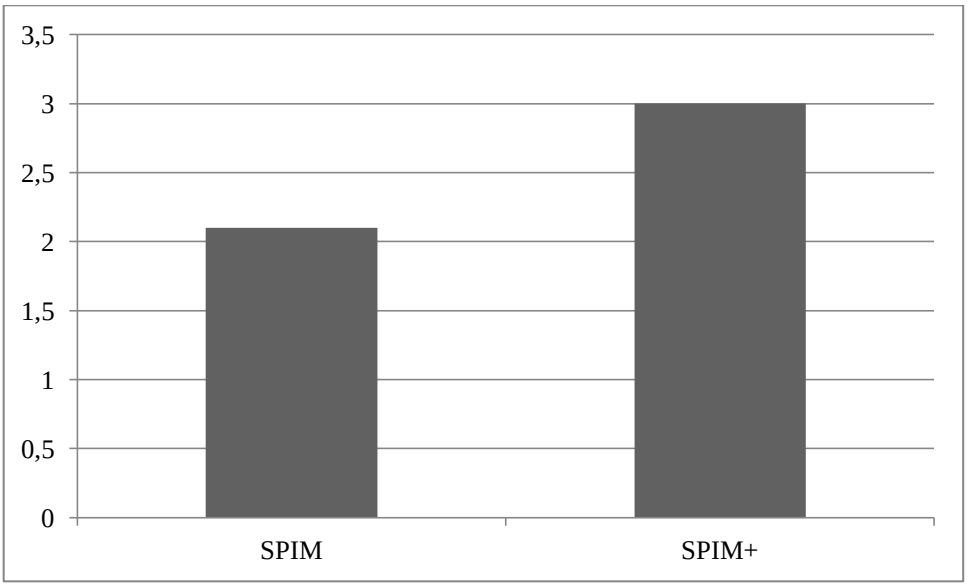


FIGURE 74:COMPARISON OF AVERAGE SCORES

Figures 75-77 show an example of an image converted to semantic map by SPIM and by SPIM+.

FIGURE 75: USER PHOTO

Result of SPIM Original Algorithm:

FIGURE 76: SEMANTIC MAP BY SPIM

Result of SPIM+ Main Algorithm:

FIGURE 77: SEMANTIC MAP BY SPIM+

9.3 Evaluation of the Inexpensive Camera Support Algorithm

This section shows the results of the algorithm implemented for the inexpensive camera support. A simple digital camera was used to capture 20 images of various places on the area of Chania. The photos depicted mountains, sea areas, islands, mountain peaks and archaeological sites. The metadata (Exif data) of the images did not include a value for the tag GPSDirection. A subset of the photos is shown in Figure 78.

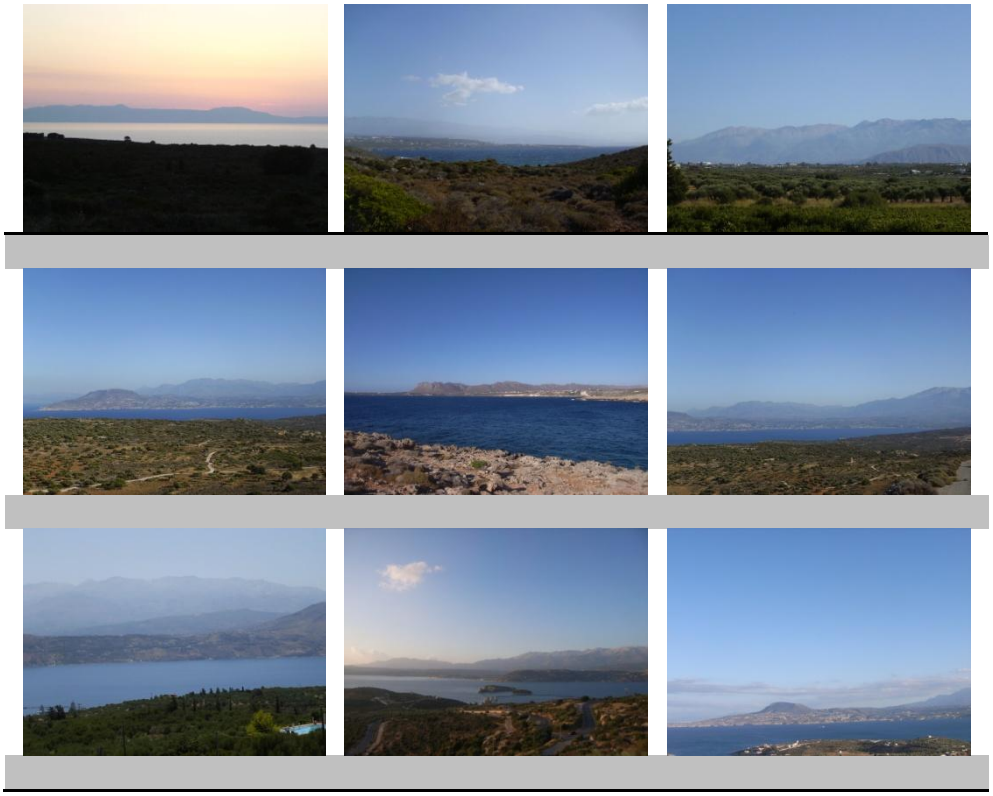


FIGURE 78: SUBSET OF NATURE IMAGES WITHOUT DIRECTION INFORMATION

Resulting Semantic Map Quality

The resulting semantic maps were classified into 7 categories according to their quality. From the best to the worst they were: *Perfect*, *Good*, *Acceptable*, *Average*, *Bad*, *Awful* and *Failed*. The first four were considered "pass" grades, while the other three "fail" grades.
Each resulting semantic map was inspected thoroughly in terms of how well the individual footprints were matched in the picture contents. The whole process was done in an unbiased way. Table 8 summarizes the results of the evaluation of the algorithm for inexpensive camera support.

Distinction	#Photos
Perfect	6
Good	4
Acceptable	3
Average	2
Bad	3
Awful	0
Failed	2
Pass	15
Fail	5
Total	20
Pass Percentage	75%
Fail Percentage	25%

TABLE 8: EVALUATION OF THE CREATED SEMANTIC MAPS

Figure 79 shows the produced semantic maps and their rank.

FIGURE 79: THE PRODUCED SEMANTIC MAPS AND THEIR RANK

The semantic maps that were produced by the inexpensive camera support algorithm are satisfactory. The algorithm functions well. The only drawback is its slow execution time. This is due to the size of its inputs. Its inputs are the boundaries of the semantic individuals from the Panorama image and the boundaries of the semantic individuals from the segmentation image. The first ones are longer because Panorama image has bigger dimensions. The algorithm tries to

find which part of the long boundary matches properly with the short boundary (segmentation image) and this makes the algorithm slow.

9.4 SUMMARY

This chapter presented the experimental evaluation of the SPIM+ framework. A set of images captured by a camera system including a state of the art digital camera and a small GPS receiver with digital compass was assembled and used for the SPIM+ framework evaluation. A simple digital camera was also used in order to capture images without direction metadata.

The images were then given as input to the system algorithms which used all the available data from a semantic map of Crete to detect the semantic individuals present in the photos. Then, after segmenting the images, the registration procedure took into account features of the segmented images (skylines, sea boundaries, mountain peaks) and their 2D view models and computed a transformation for overlaying the semantic individuals from the 2D view models on top of the photos. The application of that transformation resulted in transforming the photos into interactive semantic maps. The evaluation of the resulting semantic maps has shown that SPIM+ achieved its objectives to improve and extend the SPIM framework.

10. CONCLUSIONS AND FUTURE WORK

10.1 CONCLUSIONS

This thesis introduced the SPIM+ framework. SPIM+ is based on the SPIM framework and used its principal components. It was required to add more functionality in the initial system and ameliorate it, since there were some aspects which could be improved.

Regarding the SPIM performance enhancement, SPIM+ managed to generate more data inputs (lines) for the image registration algorithm, to provide the image registration algorithm with as more reliable and consistent data inputs as possible and to adjust the sensitivity of the image registration algorithm. Regarding the addition of more functionality in the system, SPIM+ extended SPIM in order to calculate the Sun Position at the time the photos were taken and use this information and to support inexpensive cameras (cameras with only a GPS receiver and no Compass).

For all the aforementioned enhancements, several algorithms have been developed and they are based on the key components of the SPIM framework architecture. Details about these algorithms have been presented and experiments have been conducted to prove their effectiveness, with satisfying results and visualizations.

In conclusion, SPIM+ enhances SPIM and as a result we have a system that provides a richer and more integrated functionality for managing a personal database of digital pictures and digital maps, in a semantic spatial information processing environment. The pictures are registered by more efficient means. Therefore, the pictures are converted in more consistent semantic maps. The user can select them and view on top of them the associated information.

10.2 FUTURE WORK

The following research directions are recommended as future work for this thesis:

- Once modern technology allows it in a wider scale, experiments with an increased number of sensors should be conducted, allowing for more powerful 2D view models and giving better contextual parameters for the purpose of image annotation.
- In order to extract the boundaries of the semantic individuals from a photo more accurately, an algorithm which will combine a segmentation and an edge detection algorithm could be implemented.
- The algorithm for the support for inexpensive cameras also has room for improvements. It could be modified in order to calculate the direction of the camera when the photo was taken and inform the user about it.
- The system could take into account the weather conditions under which the photos were taken. An idea is the system to retrieve data from weather forecasting websites and exploit them accordingly towards a deeper understanding of the image context and a more accurate image registration.
- The ability to represent persons (actors) within the images has not been exploited by the system. Portrait pictures and pictures with humans can be identified with current technologies and techniques such as face recognition. They could be used for the automatic or semiautomatic detection of specific persons and their silhouettes or faces.

BIBLIOGRAPHY

1. **Foukarakis, Michail.** *Informational System for Managing Photos and Spatial Information Using Sensors, Ontologies and Semantic Maps.* Chania : s.n., 2009. pp. 44-62, MSc Thesis.

2. *Exif Metadata Standard.* [Online] http://www.exif.org/specifications.html.

3. **Sarvas, Risto.** *Designing User-centric Metadata for Digital Snapshot Photography.* Helsinky : s.n., December 2006.

4. **Flickr.** [Online] http://flickr.com..

5. *Generating summaries and visualization for large collections of georeferenced.* **Jaffe, A., et al.** Santa Barbara : s.n., 2006. International Multimedia Conference. pp. 89-98.

6. **Rattenbury, T. and Naaman, M.** Methods for Extracting Place Semantics from Flickr. *ACM Transactions on the Web (TWEB).* January 2009, Vols. 3,1.

7. *Real-time computerized annotation of pictures.* **Li, J. and Wang, J.** Santa Barbara : s.n., 2006. 14th annual ACM international conference on Multimedia. pp. 911 - 920.

8. *Inferring Generic Activities and Events from Image Content and Bags of Geo-tags.* . **Joshi, D. and Luo, J.** Niagara Falls : s.n., 2008. Conference On Image And Video Retrieval. pp. 37-46.

9. *Towards Automatic Extraction of Event and Place Semantics from Flickr Tags.* **Rattenbury, T., Good, N. and Naaman, M.** Amsterdam : s.n., 2007. Annual ACM Conference on Research and Development in Information Retrieval. pp. 103-110.

10. *Leveraging Probabilistic Season and Location Context Models for Scene Understanding.* **Yu, J. and Luo, J.** Niagara Falls : s.n., 2008. Conference On Image And Video Retrieval. pp. 169-178.

11. **Smeulders, A.W.M., et al.** Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Anaysis and Machine Intelligence.* 2000, Vols. 22, 12, 1349-1380.

12. *Bridging the semantic gap in multimedia information retrieval: top-down and bottom-up approaches.* **Hare, J. S., et al.** Budva : s.n., June 2006. 3rd European Semantic Web Conference.

13. *Image-to-word transformation based on dividing and vector quantizing images with words.* **Mori, Y. and Takahashi, H.** Orlando : s.n., 1999. First International Workshop on Multimedia Intelligent Storage and Retrieval Management.

14. *Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary.* **Duygulu, P., et al.** London : s.n., 2002. 7th European Conference on Computer Vision. pp. 97-112.

15. *Automatic image annotation and retrieval using cross-media relevance models.* **Jeon, J., Lavrenko, V. and Manmatha, R.** New York : s.n., 2003. 26th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 119-126.

16. **Lavrenko, V., Manmatha, R. and Jeon, J.** A model for learning the semantics of pictures. 2004, Cambridge : MIT Press.

17. **Hare, J.S.** *Saliency for Image Description and Retrieval.* University of Southampton. 2005. Phd Thesis.

18. *An inference network approach to image retrieval.* **Metzler, D. and Manmatha, R.** Dublin : s.n., 2004. 3rd International Conference on Image and Video Retrieval. pp. 42-50.

19. *On image auto-annotation with latent space models.* **Monay, F. and Gatica-Perez, D.** Berkeley : s.n., 2003. 11th ACM International Conference on Multimedia. pp. 275-278.

20. *Multiple bernoulli relevance models for image and video annotation.* **Feng, S.L., Manmatha, R. and Lavrenko, V.** Washington, DC : s.n., 2004. IEEE Computer Society Conference On Computer Vision And Pattern Recognition. pp. 1002-1009.

21. **Jeon, J. and Manmatha, R.** Using maximum entropy for automatic image annotation. 2004, Vol. Vol. 3115 of Lecture Notes in Computer Science, pp. 2040-2041.

22. *Modeling annotated data.* **Blei, D.M. and Jordan, M.I.** New York : s.n., 2003. 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. pp. 127–134.

23. *Latent dirichlet allocation.* **Blei, D. M., Ng, A. Y. and Jordan, M. I.** 2003. Cambridge : The Journal of Machine Learning Research. pp. 993 - 1022.

24. *Matching words and pictures.* **Barnard, K., et al.** 2003. Cambridge : Special issue on Machine learning methods for text and images. pp. 1107 - 1135.

25. **Oliva, A. and Torralba, A. 2001.** Modeling the shape of the scene: A holistic representation of the spatial envelope. 2001, Vols. 42, 3, pp. 145-175.

26. *Scene-centered description from spatial envelope properties.* **Oliva, A. and Torralba, A. 2002.** London : s.n., 2002. Second International Workshop on Biologically Motivated Computer Vision. pp. 263–272.

27. **Oliva, A. and Torralba, A. 2003.** Statistics of natural image categories. . 2003, Vols. 14, 3, Network: Computation in Neural Systems., pp. 391-412.

28. **Yavlinsky, A., Schofield, E. and Ruger, S.** Automated Image Annotation Using Global Features and Robust Nonparametric Density Estimation. 2005, Vol. Vol. 3568 of LNCS, pp. 507–517.

29. *Improved blue sky detection using polynomial model fit.* **Gallagher, A. C., Luo, J. and Hao, W.** 2367- 2370 : s.n., 2004. International Conference on Image Processing. p. Singapore.

30. **Platt, J.** AutoAlbum: Clustering digital photographs using probabilistic model merging. 2000, Fort Collins : IEEE Workshop on Content-based Access of Image and Video Libraries.

31. **Vailaya, A., et al.** Image classification for content-based indexing. *IEEE Transactions on Image Processing.* January 2001, Vols. 10,1, pp. 117 - 130.

32. *Combination Of Content Analysis And Context Features For Digital Photograph Retrieval, Semantic and Digital Media Technologies.* **O'Hare, N., et al.** London : s.n., 2005. 2nd European Workshop on the Integration of Knowledge. pp. 323- 328.

33. **Boutell, M.** *Exploiting Context for Semantic Scene Classification.* University of Rochester. 2005. Doctoral Thesis.

34. **Boutell, M. and Luo, J.** Beyond Pixels: Exploiting Camera Metadata for Photo Classification. Pattern Recognition. July 2005, Vols. 38, 6, pp. 935-946.

35. **Naaman, M.** *Leveraging Geo-Referenced Digital Photographs.* Stanford University. 2005. Doctoral Thesis.

36. *Metadata creation system for mobile images.* **Sarvas, R., et al.** Boston : s.n., 2004. 2nd International Conference on Mobile Systems, Applications and Services. pp. 36-48.

37. *Context-Aware Metadata Creation in a Heterogeneous Mobile Environment.* **Volgin, O., Hung, W., Vakili, C., Flinn, J., Shin, K.G.** Stevenson, Washington : s.n., 2005. International Workshop on Network and Operating System Support for Digital Audio and Video. pp. 75-80.

38. *Mote external sensor.* [Online] http://en.wikipedia.org/wiki/Motes..

39. *Inquiry with Imagery: Historical Archive Retrieval with Digital Cameras.* **Smith, B. K., et al.** Orlando : s.n., 1999. International Multimedia Conference. pp. 405-408.

40. *Annotating Collections of Photos Using Hierarchical Event and Scene Models.* **Cao, L., et al.** Anchorage : s.n., 2008. IEEE Conference on Computer Vision and Pattern Recognition. pp. 1-8.

41. *Image annotation using personal calendars as context.* **Gallagher, A. C., Neustaedter, C. G., Cao, L. and Luo, J., Chen, T.** Vancouver : s.n., 2008. 16th ACM international conference on Multimedia. pp. 681-684.

42. *MINOTAURUS: A Distributed Multimedia Tourism Information System.* **Christodoulakis, S., et al.** Edinburgh : s.n., 1997. Information and Communication Technologies in Tourism. pp. 295-306.

43. *A modular approach to support GIS functionality in tourism applications.* **Christodoulakis, S, et al.** Istanbul : s.n., 1998. Enter 98.

44. **Spinellis, D.** Position-annotated Photographs: The Geotemporal Web. *IEEE Pervasive Computing.* 2003, Vols. 2, 2, pp. 72-79.

45. *Knowledge management and knowledge agents in Campiello.* **Koch, M.** 1998. Proceedings of the Workshop on Intelligent Agents in CSCW.

46. *Geographic Location Tags on Digital Images.* **Toyama, K., Logan, R. and Roseway, A.** Berkeley : s.n., 2003. 11th International Conference on Multimedia. pp. 156-166.

47. **Yot, Richard.** *Light for Visual Artists: Understanding & Using Light in Art & Design* . 2011.

48. *National Marine Electornics Association.* [Online] http://www.nmea.org.

49. *Shuttle Radar Topography Mission.* [Online] http://www2.jpl.nasa.gov/srtm/.

50. *National Aeronautics and Space Administration.* [Online] http://www.nasa.gov.

51. **Stone, Maureen C.** *A Survey of Color for Computer Graphics.* 2001.

52. **Michael W. Schwarz, William B. Cowan, John C. Beatty.** An experimental comparison of RGB, YIQ, LAB, HSV, and opponent color models. *Journal ACM Transactions on Graphics (TOG).* April 1987, Vol. 6, 2.

53. **al., M. Blanco-Muriel et.** *COMPUTING THE SOLAR VECTOR.* 15 November 2000.

54. **Franklin, W.R., Ray, C.K. and Mehta, S.** *Geometric Algorithms for Siting of Air Defense Missile Batteries.* 1994. Columbus: Technical Report DAAL03-86-D-0001.

55. **D., Izraelevitz.** A Fast Algorithm for Approximate Viewshed Computation. *Photogrammetric Engineering and Remote Sensing.* 2003, Vol. 69, 7, pp. 767-774.

56. **Nock R., Nielsen F.** Statistical Region Merging. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2004, Vol. 26, 11, pp. 1452-1458.

57. **E.M., Beveridge J.R. and Riseman.** How Easy is Matching 2D Line Models Using Local Search? *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 1997, Vol. 19, 6, pp. 564-579.

58. **M.A., Said.** Polyline Approximation of Single-Valued Digital Curves Using Alternating Convex Hulls. *Computer Graphics & Geometry Internet Journal.* 2002, Vol. 4, 2, pp. 75-99.