Titulo: Image Stitching for UAV remote sensing application

Volum: 1/1

Alumne: Vincenzo Cani

Director/Ponente: Enric Pastor Llorens/Cristina Barrado Muxi

Department: CAD-EPSC (UPC)Barcelona

Data: 25 de Enero 2011

DADES DEL PROJECTE

Títol del Projecte: Image Stitching for UAV remote sensing application

Nom de l'estudiant: Vincenzo Cani

Titulació: Enginyeria Tècnica en Informàtica de Gestiò

Crèdits: 22,5

Director/Ponent: Enric Pastor Llorens/Cristina Barrado Muxi

Departament: CAD-EPSC (UPC Barcelona)

MEMBRES DEL TRIBUNAL (nom i signatura)

President: Miguel Valero Garcia

Vocal: Antonio Chica Kalaf

Secretari: Cristina Barrado Muxi

QUALIFICACIÓ

Qualificació numèrica:

Qualificació descriptiva:

Data: 25 de Enero 2011

Image Stitching for UAV remote sensing application

Master Degree Thesis
Computer Engineering Turin Polytechnic
commissioned by Prof. Cabodi

Vincenzo Cani

Supervised by:
Director: Dr. Enric Pastor Llorens
Ponente: Dr. Cristina Barrado Muxi

School of Castelldefels of Universitat Politècnica de Catalunya
25 January 2011-Barcelona Spain

Contents

1 Introduction

The objective of the project is to write an algorithm that is able to join top view images to create a big map. The project is done in the School of Castelldefels of UPC, within the research laboratory Icarus[5] of EETAC Faculty.

The goal of the project is to detect an area of this map, thanks to the analysis of this images, that can be visuals or thermals.

The images are taken by the two camera aboard on an Unmanned Aerial Vehicle (UAV) built by the Icarus group leaded by Enric Pastor[1]. First I collected information about the argument to implement a sample code.

The code written it's about resolving the base case (join only two images) and it's written in C using Accord.Net framework (specific for image processing).

in order to do it first some interest point in the images have been detected, using an algorithm that can extract these feature points.

After founding these points we must correlate the points in the two images to detect what points can match in the images that we have. In this project were used already existing algorithms like SIFT, SURF and RANSAC.

The first two algorithm are used to extract feature point, RANSAC is utilized to discard erroneous correlation between points and to do that exploit statistical estimation of the points correlated.

The work was based on finishing to write the code and writing the final document.

- Firstly the code will be able to make a big image, derived by the stitching of many singles images.
- Integration of stitching algorithm as a service in MAREA (Middleware Architecture for Remote Embedded Applications) middleware 2.3.

- A parallel work is the study of GeoTiff images(5). Thanks to this images kind , it's can use gps coordinates to help the images stitching in the correct position.

The descriptive part of the work includes:

- Description of the activities of Icarus Group
- Basic methods for dealing with the stitching panoramic images[7]
- Discussion of the issue of fully automatic stitching in the case of a 2D and multi-row, using local invariant feature[9] to find matches between all of the images (Harris corner detector(4.1.1), SIFT(4.1.2), SURF(4.1.3), homography estimation(4.4), RANSAC (4.3) and blending(4.5)). Some of this algorithm are insensitive to the ordering, orientation, scale and illumination of the input images. It is also insensitive to noise of images that are not part of a panorama, and can recognize multiple panoramas in an unordered image data set.
- Analysis of work: automatics images stitching with Accord.NET [11] of César de Souza that makes a demonstration of the method with the use of Accord and Aforge.NET frameworks.
- Description of the implemented code to adapt the methodology to the case of photos taken by UAV and integrated in MAREA middleware[2].

I am a student of computer engineering master degree at Turin Polytechnic.

As part of the Erasmus project, for a period of six months, I have developed a thesis on Image Stitching used on a UAV at the University of Barcelona, where, at the same time I am also attending two subjects.

The kind of project should be "C mode", because the thesis is realized in another University. The task has been commissioned by Prof. Cabodi, teacher at Turin Polytechnic, and is supervised in Barcelona by Dr. Enric Pastor Llorens.

2 Motivation

Recent studies denote that the military UAV (Unmanned Aerial Vehicle) application is growing up in a technology sense. Nowadays organizations like the Air Force, Army, Marine Corps, and Navy operate with some type of UAV for intelligence, surveillance, reconnaissance, strike and combat support.

UAV is an aircraft that can use different types of method to provide vehicle lift, can fly autonomously or be piloted remotely, can be expendable or recoverable.

The UAV can be called by different names: uninhabited aerial vehicle (UAV), unmanned aerial vehicle (UAV), remotely operated aircraft (ROA) and Unmanned Aircraft System (UAS). Nowadays, this last term is substituting the acronym UAV. This due to the development that converted the word “Vehicle” in a “System” that includes other parts like sensors, GPS, cameras and CPUs.

These air-crafts equipped with sophisticated equipment can assist firefighters battling the flames. These UASs no needs crew, improves fire fighting strategies and reduce risks for firefighters. These remotely piloted aircrafts fly over the sky carrying on-board cameras. Throw these cameras UASs try to find the zone with flames or smoke. After they transmit these information, more or less in real time, to firefighters that can organize their strategies to reach an happy ending mission.

The principal progress in the UAS’s technologies come from the military research, with the exception of the multiple challenges and uses in the civil world. Military planners have conceived the use of UASs as a technology that could spy the enemy or even deliver munitions to a target without endangering a human pilot. The general perception is that UASs are too expensive to use for most missions. The total costs for UAS operations do not make economic sense currently for many of the missions. Other missions that are

considered unsafe and dangerous, where human life can be damaged, can exploit the use of UASs, justifying the expenses. We can divide UAS's costs in two categories: non-recurring and recurring cost. Non-recurring costs are those expenses that arise only once, typically, these costs come out from engineering, fabrication, test and integration. Instead, recurring costs are those that are directly proportional to the number of the flight hours of the UAS. Typically, these costs are measured by cost per hour.

It is expected, in this field of research, to continue tohedevelopment of those technologies that are considered needful.

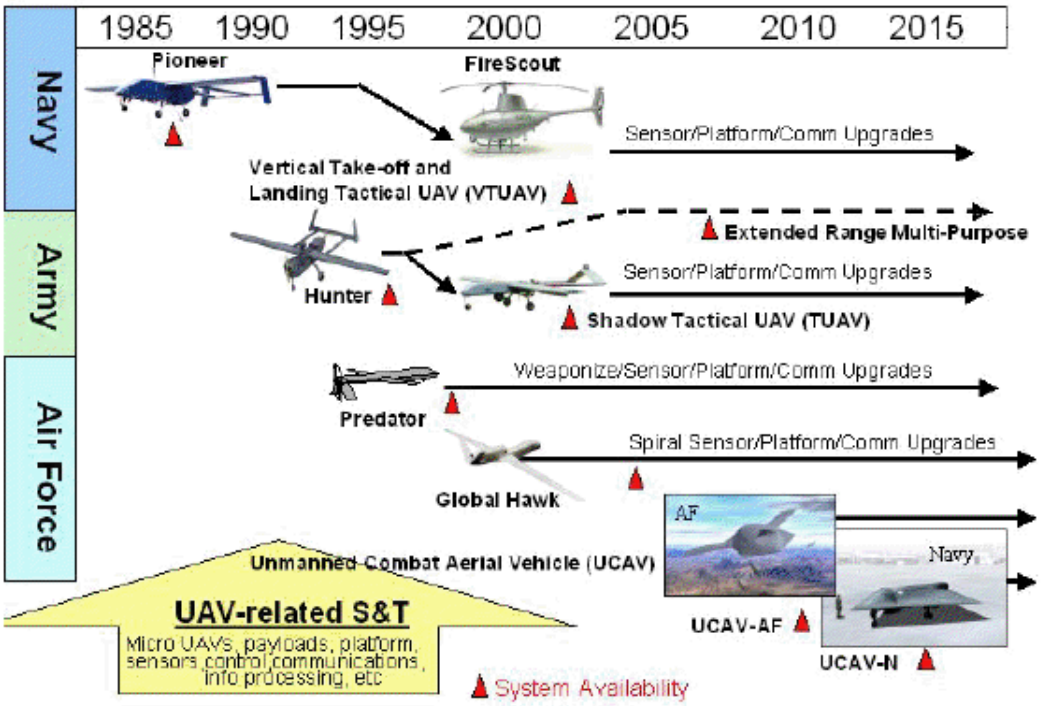

Figure 2.0.1: History of Military UAVs

As testify figure 2.0.1, the history tells us that there is a continue evolution and proliferation of these UASs, due to the existence of particular missions (these missions may be the long, boring and repetitive ones or ones required to operate in dirty areas such as volcanic plumes).

My contribution in this project is to collect the images that ICARUS's UAS snaps to build a global image of the area scanned. This map of the earth seen from the UAS can be useful to detect areas with fire. All the code is written in C# implementing software components,

called services, they rely on middleware, called MAREA (2.3), that manages and communicates the services. There are several way to analyze images to stitch them together. A first method (3) may be to analyze the colors of image's pixel, to make a comparison of these pixels between the two images. The zones with pixels that have similar value of color are used to find the translation between the two images. Another approach(4) is to use, in place of analyzing most of the image's pixel, feature points extraction (4.1).

A totally different modus operandi would be not to analyze image's pixel, but take advantage from other image's data that we could have. GeoTiff is a clear example of that. GeoTiff format has several cartographic information associated with TIFF images coming from satellites, aerial photography and scanned maps. Having a telemetry of all the photos, it's can place images in correct position among them, without make several iterations to control if images are similar.

2.1 The ICARUS Group

This work is developed within the ICARUS group.[4]

Many researchers of the Technical University of Catalonia - Barcelona Tech (Universitat Politècnica de Catalunya, UPC) work within ICARUS research group.

ICARUS is a member of the UPC's Research Center for the Aeronautics and Space (Centre de Recèrca per l'Aeronàutica i l'Espai, CRAE). The group is substantially formed by the faculty of the Computer Architecture Department and the Technical School of Castelldefels (Escola Politècnica Superior de Castelldefels, EPSC).

It keeps multidisciplinary activities as computer science, aeronautic and telecommunications engineering.

The activities (teaching, research etc.) are performed at the Parc Mediterrani de la Tecnologia (PMT) in Castelldefels (Barcelona), Catalonia - Spain and Researchers, PhD and Ms. students are also part-time or full-time members of the group.

The ICARUS group begun its initial activities in 2005, but was formally recognized as a research group by the Catalan government in 2009.

Within it are developed various publications as:

- Journal Papers and Book Chapters authored or co-authored by members of ICARUS
- Papers in Conference Proceedings
- PhD. Thesis
- Ms. Thesis and Final Degree Projects advised or co-advised by members of ICARUS

In figure 2.1.1 is reported the actual project in process by ICARUS:

Figure 2.1.1: MIME Project Overview

The main goals of of the ICARUS group are:

- Automation and development of on-board avionics and ground systems to support manned and Unmanned Aerial Systems (UAS) while providing high levels and low costs
- Improving the efficiency of air transport, reducing environmental impact and his general costs.

A special line of research is dedicated to Air Traffic Management (ATM) automation together with the integration of UAS in civil airspace.

The Group, as explain figure 2.1.2, uses extensively Information and Communication Technologies (ICT) along with methods and techniques coming from Computer Science (CS) and Operational Research (OR) disciplines and develops specific applications for both UAS and Air Transportation research lines.

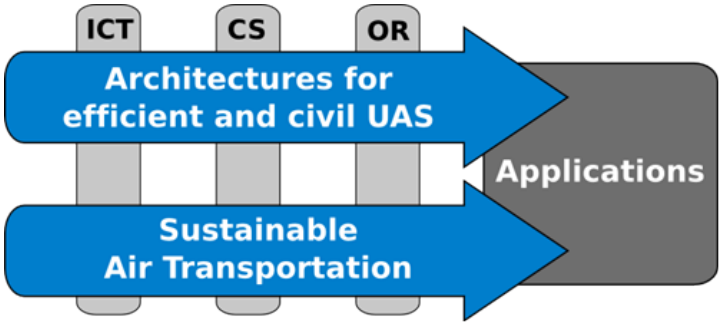

Figure 2.1.2: ICARUS's Research Lines

2.2 Architectures for efficient and civil UAS

The goal is to develop technologies to build low cost UAS (Unmanned Aerial Systems)[1] and to extend their applications to civil missions with high levels of automation and autonomy.

Unmanned Air Vehicles (UAV) (example in figure 2.2.1) are low cost vehicles that can fly without a human pilot on board, but piloted by embedded avionics and supervised by on operator on ground.

Figure 2.2.1: UAS of ICARUS Group

These characteristics are specially interesting for operating in dangerous situations.

The main advances in the UAS's technologies come from the military research, but their multiple challenges and uses in the civil world are generating much interest.

Some examples of situations were UASs can be of a great interest are environmental applications, emergency management, communications, surveillance and monitoring etc.

UAS are mostly being used, but with the evolution of avionics technology, today a huge market in civil applications is now emerging.

To compensate for the lack of hardware and software support to develop UAS potentialities for civil domains is projected a implement an innovative software architecture for UAS called UAS Service Abstraction Layer (USAL) in order to use the same platform for military applications to build economically viable UAS solution for civil applications and to implement a variety of missions with some little reconfiguration allowing the easy and fast mission design and the re-usability of the platform in a cost-effective way.

The existence of an open-framework avionics package specifically designed for UAS alleviates the development costs, allowing them to

be redesigned for different missions by a simple parametrization.

For this software abstraction layer, a distributed Service Oriented Architecture (SOA) is used. Functional units are implemented as independent services that interact with each other using a Local Area Network (LAN).

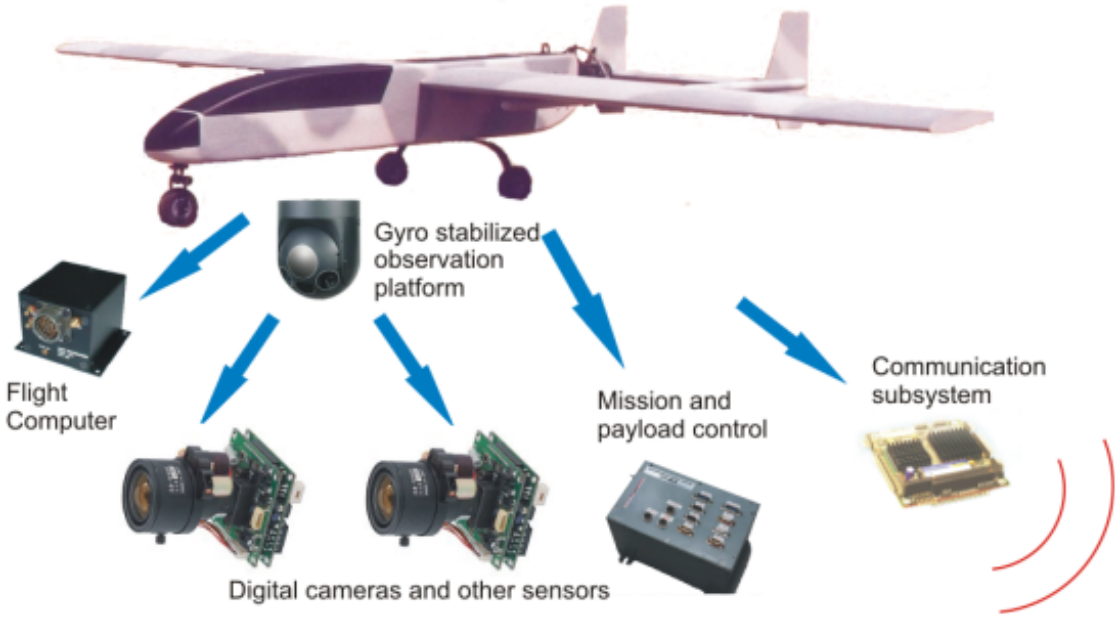

Figure 2.2.2:

Thus, the USAL provides a list of common services needed to develop the different civil missions identified. These services have been organized into four different categories, each containing services that cooperate in the same main objective such as Flight, Mission, Payload and Awareness.

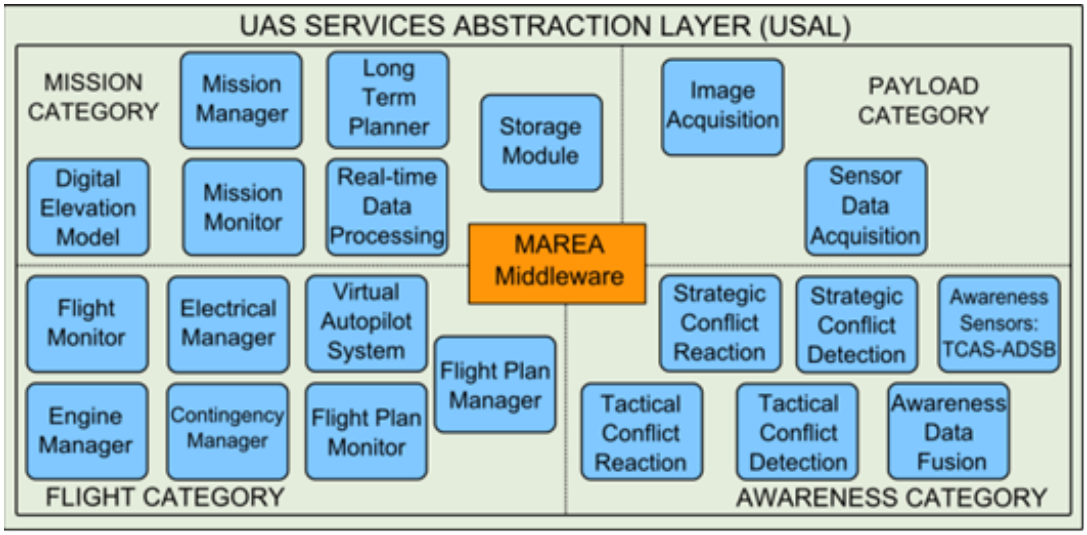

Figure 2.2.3: Graph of common services

2.3 MAREA Middleware

The need to use several data links both for redundancy and for turning with different communication requirements has increased the Unmanned Aircraft Systems (UAS) communication subsystems complexity. This made it necessary to implement modern digital avionics as distributed computing architectures. The middleware specifically designed and developed, at the Technical University of Catalonia by Juan Lopez, to resolve both the communications and their application problems is Marea [3](Middleware Architecture for Remote Embedded Applications)

As UAS missions support more intelligent and cooperative behavior, UAS communication systems will benefit from more complex communication patterns like request-response, asynchronous events or remote procedure calls.

His function is to provide a quality of service to various applications and is mainly aimed at facilitating the rapid development of services to create a distributed application. Its use is very easy to program event-driven.

Despite the area of application software is UAS, it can also be extended to many other areas of interest. The middleware is imple-

mented as the main program and services are seen as collections of libraries. The C# development language and the code has been tested on Windows NT/Vista, Linux (with Mono) and Windows CE. The future goal is to implement a lighter software to run in embedded systems. Marea-based application should be composed of Marea.exe file running on each node of the network that make up the application. This file builds the Container concept, i.e., the middleware that provides the following four communication primitives: variables, events, remote invocations and file transfers.

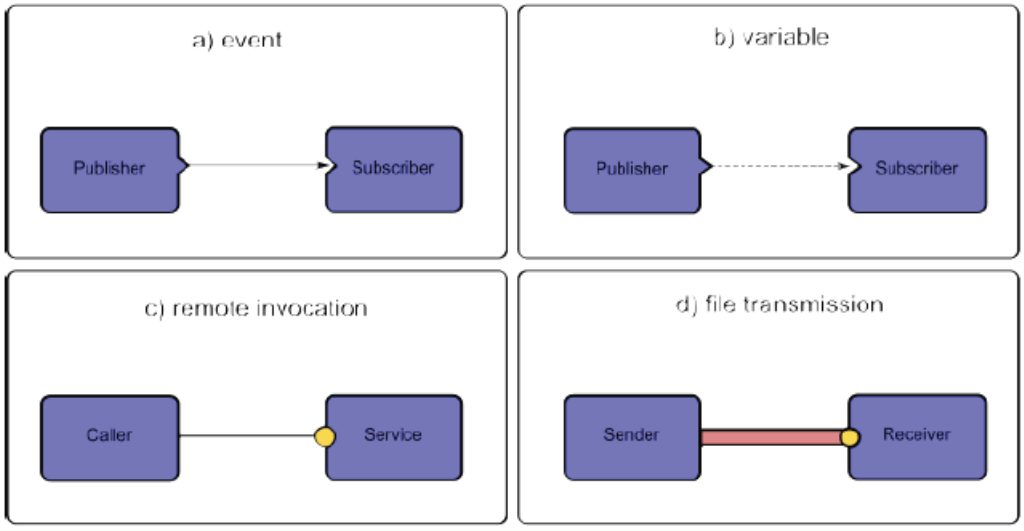

Figure 2.3.1: Communication Primitives

The middleware has to develop easily distributed applications, that are sets of services executing instances of Marea Containers Services must be subclasses of the IService Class defined in the form of source code or DLL to be linked with the Marea.exe file. The Marea distribution comes with the Marea.exe file plus a Test folder. The Test folder provides several running examples of the four communication primitives. Two services are given for each primitive: One shows the publisher code and the other the subscriber code. UAS communications system benefits for more complex patterns (request response, asynchronous events , remote procedure calls etc). Two different approaches are given: federated (every avionics functionality is integrated into a black-box and resources are not shared) and modular IMA (functionality are distributed into logical partitions

which may be allocated in the same physical computing module or into a different one). For effect of resource sharing, the resources can be used and allocated more efficiently.

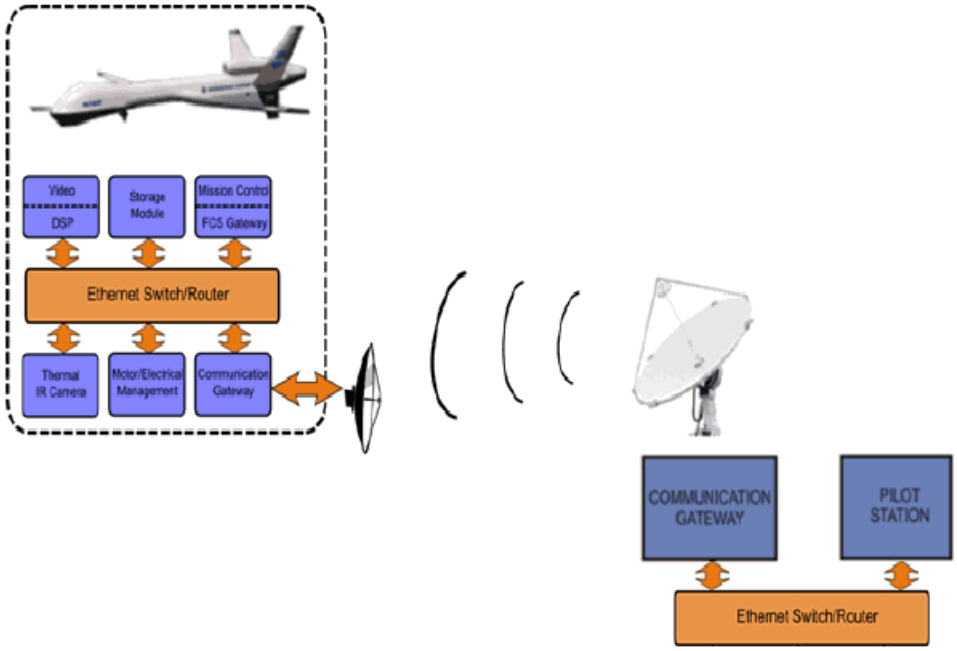

Figure 2.3.2: UAS Communication Architecture. The UAS seen as a network of distributed components.

UAS avionics are usually systems less complex than the airliners (less instrumentation, less engines, less pressure control, etc.) but the on board avionics should control the flight, navigation, mission and payload of the aircraft. This involves more complex software to implement the complex interactions between different UAS avionics components. The transparency and the flexibility are very important in these communication mechanisms, both inside the UAS airframe and between the UAS and the ground control station. The systems are composed of network of cooperating components implementing the logic of the system and a layer of integrating middleware that abstracts the execution environment and implements common functionality and communications channels. The different components are semantic units that acts as both producers and consumers of data coming from other components. The localization of other components is not important because the middleware manages their discovery. Furthermore it handles all the transfer work needed. Marea

offers a modular architecture based on services. In fact the avionics system consists of set of distributed elements, i.e. the services, which operates on top of the framework of communication middleware. The services are located in different computing nodes that are connected by a low-cost local network.

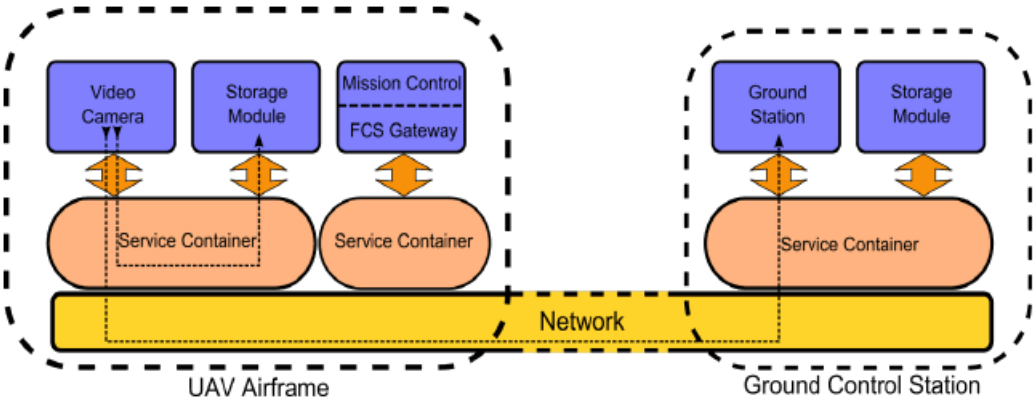

Figure 2.3.3: Marea middleware. The figure shows two Marea containers supporting several services both on board the UAS and on the Ground Control Station.

Most of the current middleware is based on client-server relationships between components. While this approach is applicable for most systems, it does not take advantage of the multicast capabilities of local networks. Publish-Subscribe communication are better suited for this type of systems. Marea combines both client-server and multi-point communications message, and is particularly suitable for communicating low-cost components connected by local networks.

In Marea the middleware has the form of a component called service container. The services are managed by a service container in each network node of the distributed system. The service container handles different services and provides common functionality as network access, local message delivery, name resolution and caching, etc. The communication primitives, that provides Marea services, are able to identify the service in a way transparent and to attached them to provider without having to know their physical final. The four communication primitives (Variable, events, Remote Invocation and File Transmission) give to the developer a wide range of opportunities to interconnect and to make interact services. A Variable is

a information offered by a service in a publish-subscribe. This information can be sent at regular intervals or when changes occur. An Event is similar to a variable, but the middleware ensures reliability of the transmission.

They are used to inform of facts important for other services. Remote Invocation is the way to model the interactions between distributed components. A File Transmission is a continuous transfer of data information, including photographic images, video, configuration files or program code. Marea primitive identifies exchanged data rather than their suppliers or consumers. The architecture of Marea has its components distributed in four layers: Protocol, Encoding, Presentation and Transport.

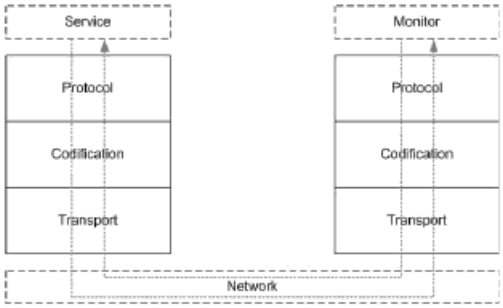

Figure 2.3.4: A high level view of Marea middleware layers. The figure shows two Marea containers with their internals layers: Protocol, Encoding and Transport.

The code implemented in Marea is divided in two principle services: FindPhoto and ImageStitching. This two services communicate between them through Variable publication and subscription. The types of the variables that they send and receive are strings. These strings contain the pathnames of the images, taken by the camera, that we want to join together. The task of the first service is to find the photo to join, using the class System.Windows.Forms, ask the user to select a folder where the images are stored. Every images taken by the camera, when are saved, have a default name of the type "NAMECAMERA.DATE.HOUR.MINUTE.SECOND.MILLISECOND". This permit to catalog images, moreover it's can't have two images with the same name. The FindPhoto service splits the pathname of the images, that are in the folder selected, and try to find photos that

are snapped in near time (less than 5 second). It finds two images with this peculiarity, almost always these photos have an overlapping region between them and in most cases it can join them. The last work of this service is to communicate by string variables what are this images that probably can be stitched. The one who receive these string variables is the ImageStitching service, this service is more elaborated than the previous and we are going to describe it in the next chapters.

3 The image sticthing basics

In this section are described basic methods for dealing with the stitching panoramic images.

The image stitching is a form of image mosaicing, than has become increasingly common, especially in the making of panoramic images.

The stitching is used in applications such as interactive panoramic viewing of images, architectural renderings, and other applications associated with the 3D modeling environment using images acquired from the real world.

A panorama is a picture that is made by combining a series of photos into one large picture. By combining a series of photos, it can provide a complete view of an area or location that cannot fit in a single shot.

The Panoramic Image Stitching is the process undertaken to create a panoramic image by overlapping a series of smaller images.

The first step for generating a panoramic image is to locate and capture images by deciding in advance what kind of image overview will be achieved. Depending on this can be taken one of many methods of acquisition.

After the images have been acquired, it may take some processing to be applied to images before stitching, because the distortions caused by the lens of a camera must also be corrected before the images are processed further.

Principally the process of stitching images is divided into two phases, the comparison and the blending of images .

During the first phase, the portions of adjacent images are compared to find the shifts necessary to align them.

Once the overlapping images have been compared, they must be joined together to form a single panoramic image. The process of image comparison is performed to make visually undetectable the transition between adjacent images. Finally panoramic image is gener-

ated after the images were stitched. For the generation of panoramic images, the images can be acquired using a relatively inexpensive camera and the image viewing angle is determined by the user. The stitching image is greater resolution panoramic image captured from a panoramic camera.

3.1 Image acquisition

Different methods can be used to acquire input images in order to produce different types of panoramic images, depending on the required type and the availability of equipment.

The configurations of imaging for the generation of panoramic images described and discussed below is the case in with the camera is placed on a slide plane parallel to the surface of the earth and the images are obtained by moving the camera on it (acquisition by camera translations) capturing top view image of the earth.

The case of study concerning this work can be included in this category. The photos are captured by incremental steps of 5 seconds between them, made with two cameras mounted on a Unmanned Air Vehicle moving with a speed almost constant and maintaining its trajectory on the almost horizontal plane.

The figure 3.1.1 shows the coordinate system of the camera, where the Z-direction towards the object of interest and the y-axis coincides with the optical axis of the camera and Y is parallel to the surface of the earth.

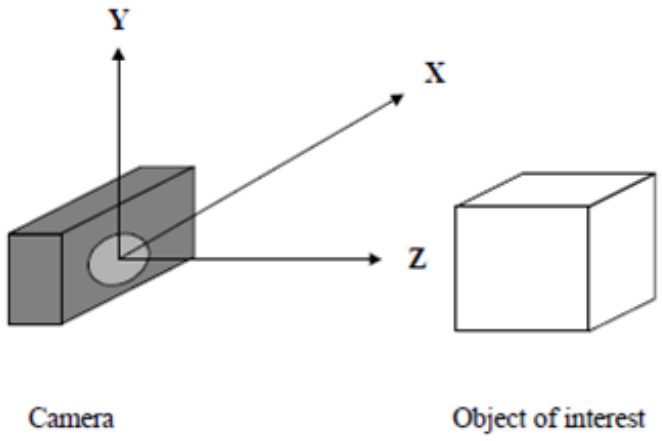

Figure 3.1.1: Camera co-ordinate system

The view angles of the camera in the horizontal and vertical directions determine the coverage of each image in the horizontal and vertical.

The view angles are defined in figure 3.1.2, where the angles α and β respectively represent viewing angles of the camera in the horizontal and vertical directions.

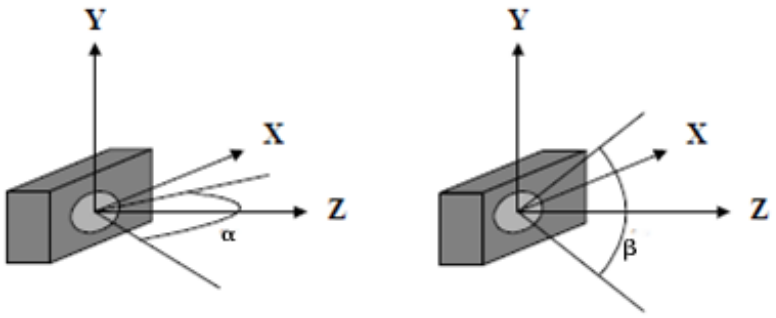

Figure 3.1.2: The horizontal and vertical angles of view

In this method of acquisition, the camera moves in a direction parallel to the image plane. It's supposed that the distance between the UAS and the earth, substantially, remain the same.

The camera is placed in front of the objects of interest (the surface of the earth) and an image is taken for each translation of the camera until the series of images covering the desired range.

Figure 3.1.3 shows the configuration of this method, the camera is in line with the plane where the UAS is moving so that this plane is parallel to image plane with an opportune orientation of the camera.

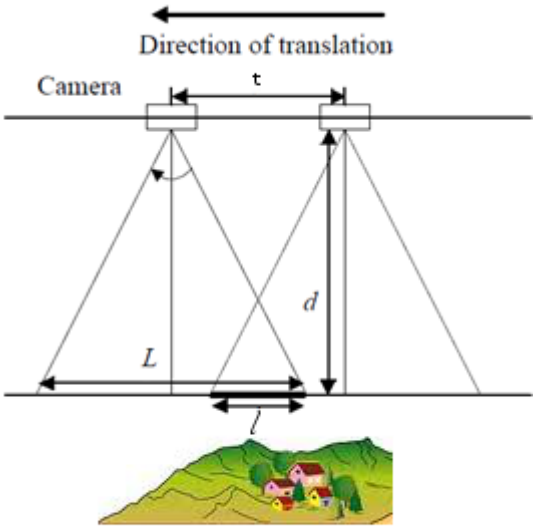

Figure 3.1.3: Geometry for image acquisition by camera translations.

Given the translation of camera, t, the distance between camera and object of interest, d, and horizontal viewing angle ϑ, L represents the width of the acquired image and l represents the width of overlapping region between adjacent images. $\frac{1}{L}$ is the relationship between the region of overlap for the entire image and can be estimated as in figure 3.1.4 .

$$\frac{1}{L} = 1 - \frac{t}{2d\tan(\frac{\vartheta}{2})}$$

Figure 3.1.4: Relationship between the region of overlap for the entire image

However, the actual size of the region of overlap between two successive images is determined by the accuracy in setting up of the camera.

With the acquisition of images through translation, it's important to ensure that the image planes are parallel to the direction of movement of the camera.

Otherwise, the size of objects in the images changes when the camera moves, causing problems in the image stitching. A disadvantage of this method is that the request translation, t, increases with the distance between the camera and the object of interest, d, if the images are acquired for the same size area of overlap.

Thus, the acquisition of images in which the object of interest is far from the camera is more difficult because of the magnitude of the necessary translations.

3.1.1 General problems in image acquisition

One of the most frequently problems in image acquisition is the change in light intensity between adjacent images.

Ideally, an area or object must have the same intensity if represented in adjacent images. However, due to the variation in the intensity of lighting or the angle between the camera and light source, the intensity values for the same region or object are different in adjacent images.

Other causes of variation in intensity between the contrast images are made during development or during the scanning of photographs, which can both be avoided by directly using a digital camera.

Another problem associated with the lighting is the reflected light on areas, like zones with glasses or shiny metal.

During the time needed to adapt the equipment to the next position after the acquisition of each image, the objects within the scene may have been moved from their previous position.

Therefore, you should consider this when moving objects are to be included in a series of images.

In fact it will be very difficult to properly record the images, once an object in the images has moved from its original position.

Images can also suffer from distortions of the lens depending on the lens used to capture them. These distortions can be corrected using the same objective to capture an image of a grid. Using the known parameters of the original grid, you can find the transformation that has distorted the original scanned image.

The transformation can be applied to each of the images taken with the same goal to correct the distortion.

In the particular case of photos taken on board the UASs are to be considered strong engine vibrations that cause distortion in the frames. To avoid this ICARUS group utilized elastomers to attenuate the resonance of the plane where the camera is placed.

3.2 Image Transformation

In this section are described basic methods for image comparisons explained in [7].

I decided to write about this case of study because, initially, I implemented the code about this method. To work faster I tried to work in image with lower information; specifically before applying the algorithm explained below, the images need some transformations.

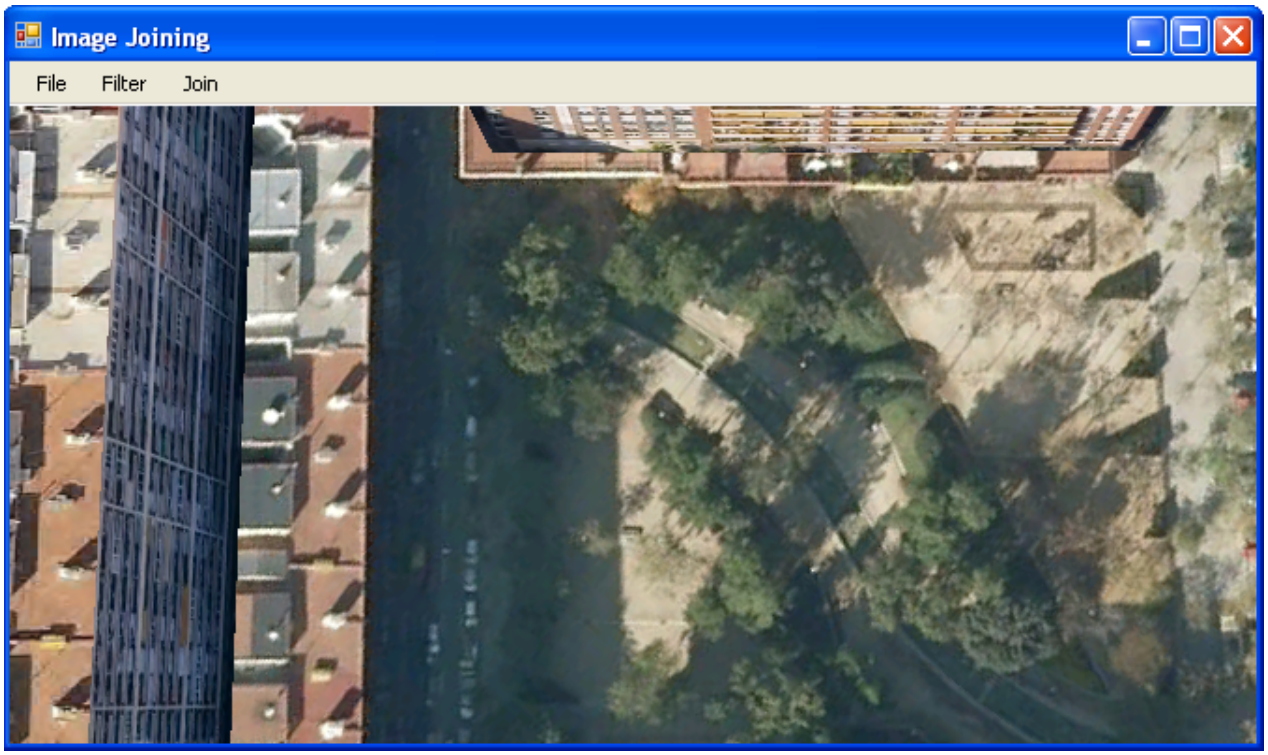

Figure 3.2.1: Image RGB

A solution implemented was to convert from rgb image to grayscale, this first task allows to work with less informations because a rgb image is a three-dimensional matrix and a grayscaled image has only one-dimensional matrix.

Figure 3.2.2: Image Grayscaled

The next step was to apply a Canny edge detection operator. This filter returns the borders of an image and uses calculus of variations for doing that (fig.3.2.3).[13]

The last phase is to turn the resulting image in a binary image, this means that the range of values that a pixel can have turns from 0-255 to 0-1.

Transforming to a binary image, figure 3.2.4, obviously leads to a loss of information, but is necessary to perform this preliminaries conversions because improve the speed of algorithm.

Figure 3.2.3: Image with Canny edge filter applied

Figure 3.2.4: Image Binarized

3.3 Image Comparison

To form a larger picture with a series of overlapping images, it's necessary to find the translations in order to align the images.

The process of stitching images is proposed to find the translations to align two or more overlapping images such that the projection of pixels is always aligned through any position. The steps required can be represented by the diagram in figure 3.2.3 .

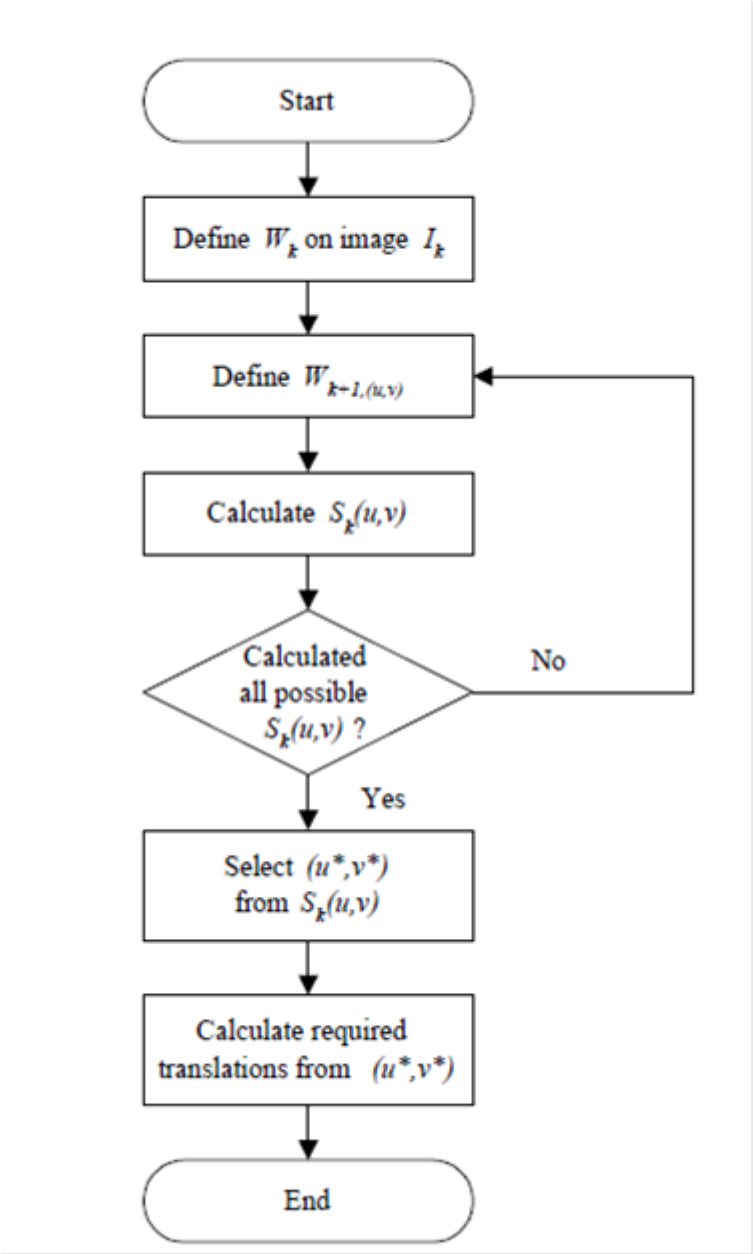

Figure 3.3.1: Flowchart of steps involved in image comparison

From an estimated ratio of the area of overlap is defined a window, that is described in figure 3.3.2. The right side of the left image must be 50% of the width and height of the input images in order to be sure that it's within the region of overlap of the input. Observing the first image on the left side means that the region of overlap between I_k and I_{k+1} is to the right side of the I_k and to left side of I_{k+1}.

W_k is defined on the right side and centered of I_k vertically in the middle of the image. Keeping the components and properties of the window in mind, now describes the method of recording images in more detail.

I_k is the image obtained from the average intensity of red, green and blue channels of kth image in the sequence of input images, where k can be from 1 to total number of images in the series of images.

W_k is defined the window mxn in I_k, with the top-left hand corner at position (a, b) of I_k, as shown in figure 3.3.2 . The image on the right hand size of k_{th} image, the image I_{k+1} , is transformed by a transformation chosen by the search set. A window, W_{k+1}, the same size and shape of W_k, is defined on I_k. The position of W_{k+1} on I_{k+1} is obtained by applying the inverse of the selected transformation to (a, b).

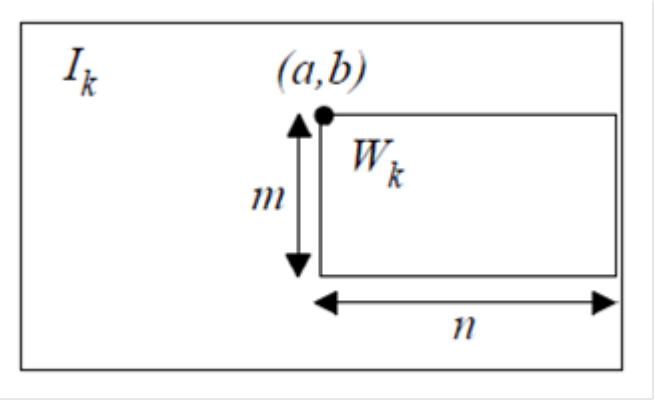

Figure 3.3.2: W_k is the m by n window at position (a,b) in I_k.

This method of stitching images usually consists of four main components. They are:

- **feature set**, that defines what you need to compare images,
- **similarity measure**, that is how to assess the similarity of the

images,

- **search sets**, that is the range of possible transformations between the images,
- **search strategy**, that explains how to decide the next transformation of assessment based on the measure of similarity.

By varying the content of these four components can be constructed different methods of registration with different behaviors.

The **feature set** is the set of features that are used in the comparison of images as the intensity values in the color, contours, textures and so on. A set feature must be selected for each method of image registration. The characteristics, that can be chosen according to the algorithm used, are extracted from the images and compared in the recording of images.

The **similarity measure** is a function that returns a scalar value which gives an indication of the similarities between two features. Similarity between the characteristics is the similarity with the orientation, size and color characteristics. The intensity values of the selected features from images are used to calculate the similarity measures. The values of the similarity measures are used to select the transformation to align the images.

The **search set** is a set of possible transformations to align the images. It contains the transformations such as horizontal or vertical translation, rotating, or other more complex transformations obtained with a combination of translations and rotations.

The changes contained in the search sets are assessed by the similarity measure to decide the best transformation (better value of similarity measure) necessary to align the images provided.

The **search strategy** is the algorithm that decides how to choose the next transformations from the search set.

Over the years, a number of methods for recording images have been proposed. These methods generally involve pattern to identify the transformations needed to align the images[16, 17, 18].

The figure 3.3.1 shows an example of a pair of adjacent images.

Another thing that needs to be said is that the center of the image tends to contain more information in general and the central regions are less likely to be of uniform intensity values. So by putting the

Figure 3.3.3: Example of input images

window toward the center of the image, the similarity measure calculated from the contents of the window provides a more reliable indication for the similarity measure of the windows.

The similarity measure between two windows is the sum of the differences of the absolute values of the two windows.

$$S_k(u,v) = \sum_{i=1}^{m} \sum_{j=1}^{n} \mid W_k(i,\, j) - W_{k+1,(u,v)}(i,\, j) \mid$$

Once the similarity measures for all possible positions were calculated, the optimal matching position, denoted by (u*, v*), is chosen by examining the magnitude of the values in S_k.

$$S_k(u^*, v^*) = \underset{1 \le i \le H-m,\, 1 \le j \le L-n}{Min} \{S_k(i,\, j)\}$$

It's intuitive to choose the position which has the minimum similarity measure as the best position to match. The reason is that the sum of the differences in absolute values of the two windows was used as a measure similarity and a smaller difference usually implies a high degree of similarity between the windows. Therefore, a position is chosen so that the value of similarity measure in that position is the lowest in all the similarity measures calculated.

Results

I discover that the execution time was huge, because it's use iteratively comparisons pixel by pixel. Using image with big resolution this time wasted was not acceptable.

However, this method may return incorrect translations due to various reasons. One of these is that the location, where the similarity measure for the windows is generally minimal, it was assumed to provide translations for optimal alignment of the images. In the event that the intensity between adjacent images differ significantly, the absolute value of differences in average intensity may not be a good indication of the similarity of images.

As a result of this, the methodology just described has not yielded significant results in the experience carried out taking pictures from a helicopter UAS to form views. However is an alternative algorithm that can be used and brings good result with accurate precision, but his delay affects the performance of the system.

4 Automatic Recognizing Panoramas

The panoramic image stitching has a vast research literature and several commercial applications.[11]

The methods for automatic alignment and mosaic of images typically fall into two categories: direct and based on features.

Direct methods attempt to iteratively estimate the camera parameters by minimizing an error function based on the intensity difference in the area of overlap.

They have the advantage of using all available image data, and thus they can provide very accurate records, but require, as mentioned, very careful initialization.

The basic geometry of the problem is well understood and involves estimating a 3×3 camera matrix (homography) for each image.(4.4)

This process requires an estimate of initialization, usually provided by user input to roughly align or reorder the images.

For example, some software requires a horizontal or vertical scanning, or a square matrix of images or provide user interfaces for the approximate location of the images with the mouse to be used prior to automatic comparation.

Part of the code is taken from the work developed from César de Souza[11].

In order to make image stitching is used Accord.NET Framework, extension framework of AForge.NET (an open source C# framework designed for developers and researchers in the fields of Computer Vision and Artificial Intelligence - image processing, neural networks, genetic algorithms, fuzzy logic, machine learning, robotics, etc.).

Accord.NET implements tools and features that are not available in AForge.NET, among other things, to develop programs for automatic image stitching and possibly for automatic panorama creation.

In the examples below described, will be demonstrated how to use the features, already available in Accord.NET, to stitch together two images and create a small and simple panorama.

4.1 Feature Extraction

Feature based methods begin by establishing correspondences between points, lines or other geometrical entities.

Local invariant features are used to find matches between all images.

This makes the job all insensitive: order, orientation, scale and lighting of the input images, as well as insensitivity to image noise, which are not part of a landscape.

It is also capable of recognizing multiple panoramas from data set of unordered images.

The extraction of feature points from an image is a job that can return different results, depending on the used methods.

In consideration of speed and accuracy of finding this points there can distinguish many different algorithms, each of them has strengths and weaknesses. Below we are going to describe three algorithms implemented to extract these feature points. Some of the implementations of these algorithms are invariant to scale, rotation and change of illumination; this imply a finer selection of feature points and so more possibilities to reach a satisfying blending.

Finding the corrects feature points is fundamental to perform a right stitching. It's the first step, so it's crucial to choose the correct detector of this points.

4.1.1 Harris Corners Detector

This operator was invented by Chris Harris and Mike Stephens in 1988[10]; it was only a processing phase to analyze a robot's environment represented by images.

Harris detector seeks for points that have large intensity variations among neighbourhood. The characteristics of this algorithm lead to finding in most of cases corners and edges. The aim is to extract corners feature and use a normalized cross-correlation of the local

intensity values in particular points to match them.

A first version of Harris detector was made by Moravec; after Harris and Stephens improved it by considering the differential of the corner score with respect to direction directly.

To create the panorama, having identifying the interest points between the two images, one of the images is projecting on top of the other in an effort to match those points.

For more details around the kind of those feature points, it may examine the source code for the Harris Corner Detector algorithm developed by César De Souza.

Below, in figures 4.1.1 and 4.1.2, we have a practical example of feature points extracted with Harris Corners Detector. There are used two top-view pictures of the earth, with different sizes among them.

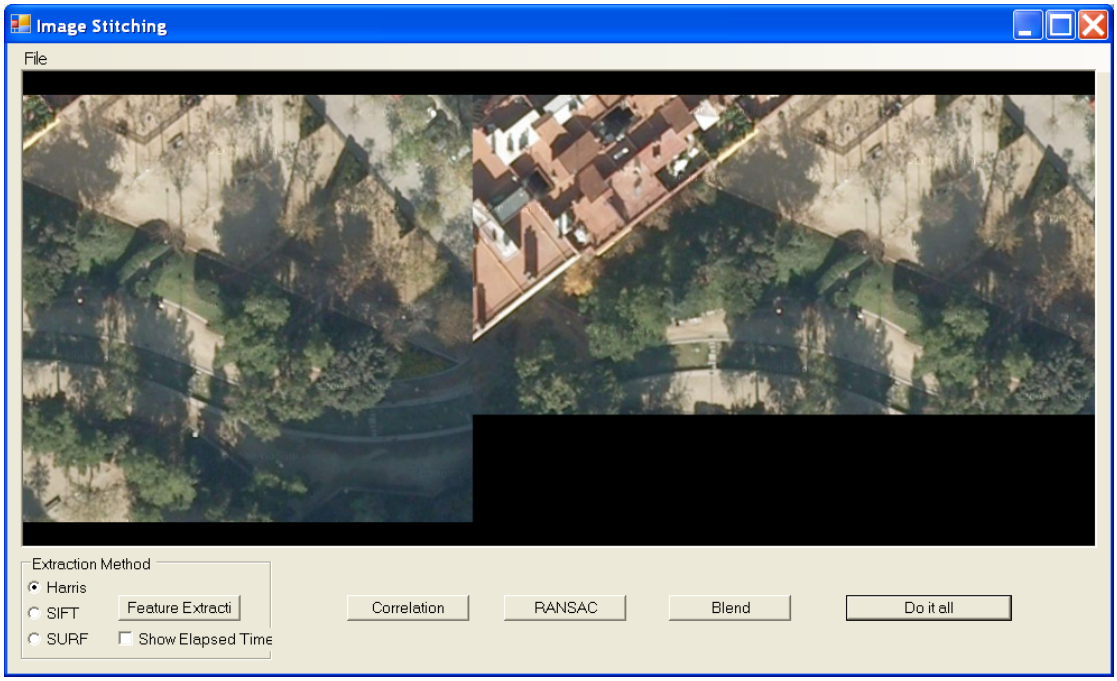

Figure 4.1.1: Pictures with detected interest points not marked

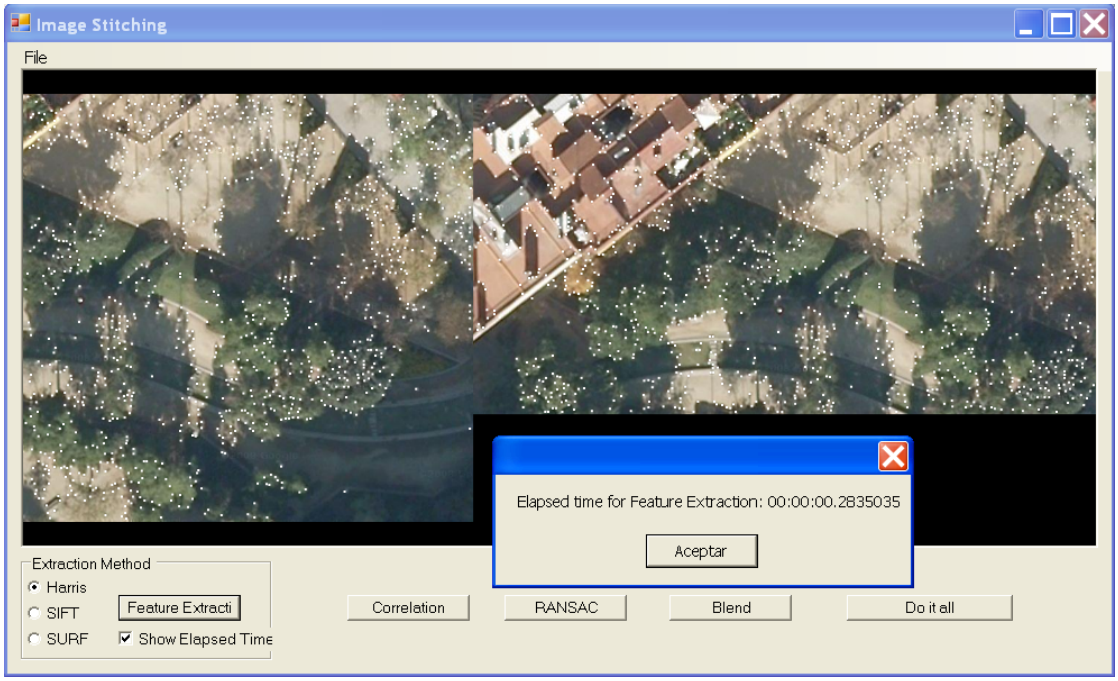

Figure 4.1.2: Pictures with detected interest points marked in white

It can see, from figure 4.1.2, that this algorithm is very fast (0.28 seconds for, approximately, 2500 feature points founded). This is the principle feature that characterize it. This is not an attribute to underestimate, especially for real-time system, because sometimes you must execute within strict constraints on response time. But his admirable reaction can't be compared to his reliability; in fact this detector could return different results depending on the image's orientation and size.

The code implementation is made using Accord.NET Framework that is a C# framework extending the excellent AForge.NET Framework.

4.1.2 Scale Invariant Feature Transform

The dealing with the problem of fully automatic blending multiple top-view images is referred to the report of Prof. David G. Lowe of University of British Columbia, Vancouver, Canada. Lowe described in his publication Scale Invariant Feature Transform algorithm (SIFT)[19], used in computer vision to detect local features in images.

The feature based methods begin by establishing correspondences

between points, lines or other geometrical entities.

They do not require initialization, but the traditional methods (such as the correlation of parts of the image around the corners of Harris) did not have the invariance property needed to make it can reliably match an arbitrary sequence of panoramic images. The SIFT approach is based on a feature invariant[20] to achieve the "seam" fully automatic panoramic images. This has several advantages over previous approach:

- The use of invariant features allows reliable matching of image sequences, panoramas, despite the rotation, zooming and change of illumination in the input images.
- Considering the stitching of the image as a multi-image matching problem, we can automatically discover the relationship of correspondence between the images and recognize landscapes in an unordered data set.
- It can generate high quality results using multi-band blending output to display seamless panoramas.

SIFT detects a larger number of features from the images than Harris detector, which reduces the probability of errors by these local variations in the average error of all feature matching errors. Lowe needed a method to identify object into images and, even if this object is partially overlapped with other object, his algorithm is so powerful to recognize the object we are looking for.

Anyhow SIFT has also its point of weakness received from its complexity. This searching of several feature points has a cost that affects on the time spent on execution. This is the big compromise between quality of feature points and speed performance of the system.

4.1.3 Speeded Up Robust Features

Presented by Herbert Bay in 2006[12], Speeded Up Robust Features (SURF) is a scale and rotation invariant interest point detector and descriptor, that is used in computer vision works like 3D reconstruction or object recognition. It is an improvement of the SIFT descriptor, that guarantees a better efficiency.

The implementation of SURF is various times faster than SIFT and, as enhancement of the last, demonstrates to be more robust against different image transformations than SIFT.

In order to detect features, the value considered is the sum of Haar Wavelet responses[21]that is applied around points iteratively.

Every interest point has a neighbourhood from which the descriptor is calculated. The region must have a squared shape and is subdivided in many subregions. The number of these descriptors has an impact on the number of calculus to do. In simplified form, the scale-normalized determinant of the Hessian[22] computed from Haar wavelets is used as the basic interest point operator in the SURF descriptor. The Hessian is an important factor; in practice a point became a feature point only if the value of the hessian is bigger than a threshold.

Here there is presented a screenshot using SURF algorithm where are extracted only features point with hessian larger than 500 and basic descriptors (64 elements each).

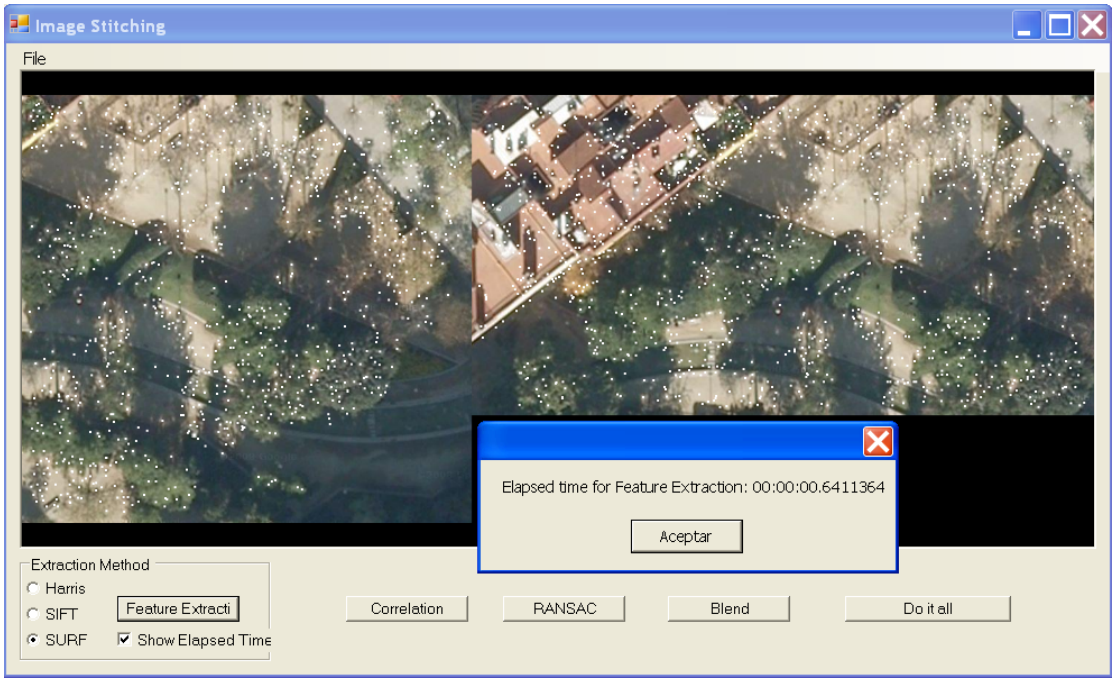

Figure 4.1.3: Pictures with detected interest points marked in white

To make a comparation with Harris detector, we use the same images for both and, in this case, the number of feature points obtained, more or less, is the same. Immediately we realize that the

time elapsed to extract interest points is twice longer than Harris's method . This is acceptable because these feature points founded are robust to noise and as we said before are invariant to scale and rotation.

4.2 Intensity Cross-Correlation

After we have extracted feature points, with the methods we have explained before, we must analyze them to find what points corresponds to the others in the two different images. The aim is to examine iteratively pixel around feature points in the first image to find and correlate them with pixels around every other points in the second image.

It's essential to match uniquely feature points among them, Intensity Cross-Correlation[14]compares local neighborhoods of interest points, it returns a similarity measure that is obtained as follows:

$$C = \sum_{i=-N}^{N} \sum_{j=-N}^{N} \left(I(x-i, y-j) - \overline{I}\right) \left(I'(x'-i\,,\, y'-j) - \overline{I'}\right))$$

This is the value of correlation between two point $(x,\, y)$ and $(x',\, y')$, I and $I'$are the intensity values in a point, $\overline{I}$ and $\overline{I'}$ the mean intensity value of the considered neighborhood and N represents the size of the neighborhood.

Through this formula Intensity Cross-Correlation is calculated for every possible combination. The highest bidirectional correlation value indicates the couple of points correlated. However, sometimes it's happens that many points have been wrongly correlated, so we need a mechanism to understand which pairs of points has been associated in a erroneous way. The number of feature points found in the precedent step affects the results of this method, it is not necessary to have several points of interest in order to obtain a good solution, but only with few points the number of erroneous pairs became a big percentage of the total and neither using RANSAC4.3 it can correct false correlations between two points.

4.3 RANdom Sample Consensus

RANSAC is the acronym of “RANdom SAmple Consensus”. This filtering algorithm was published by Fischler and Bolles in 1981.[8, 9] It is a non-deterministic algorithm, because it doesn’t ensure to return acceptable results, the probabilities of success increases if more iterations are made.

RANSAC distinguishes two types of data: “Inlier” and “Outlier”. If a data, compared to the set of data, follows a determinate rule, that can be explained with mathematical model, is called “Inlier”. “Outlier” are erroneous data, they don’t fit with the model and disobey at the statistical criteria of “Inlier”.

This kind of data come out from a wrong correlation, in our case.

The data analyzed by RANSAC, in this project, are homographies4.4 of feature points correlated; “Outlier” are mistakes made by Intensity Cross-Correlation and “Inlier” all the good association between interest points.

RANSAC is an algorithm that iteratively, from a set of observed data, estimates parameters of a mathematical model attempting to find “Outlier”and to disassociate them.

RANSAC tries to find the best model testing different correlation between feature points; the homography that returns the highest number of correct matches is chosen as the solution of the problem.

As it can see from the figure 4.3.1 “Outlier” are the diagonal lines that don’t follow the statistical direction of “Inlier”. These incorrect correlations may cause troubles in blending phase, so it need to remove them with RANSAC.

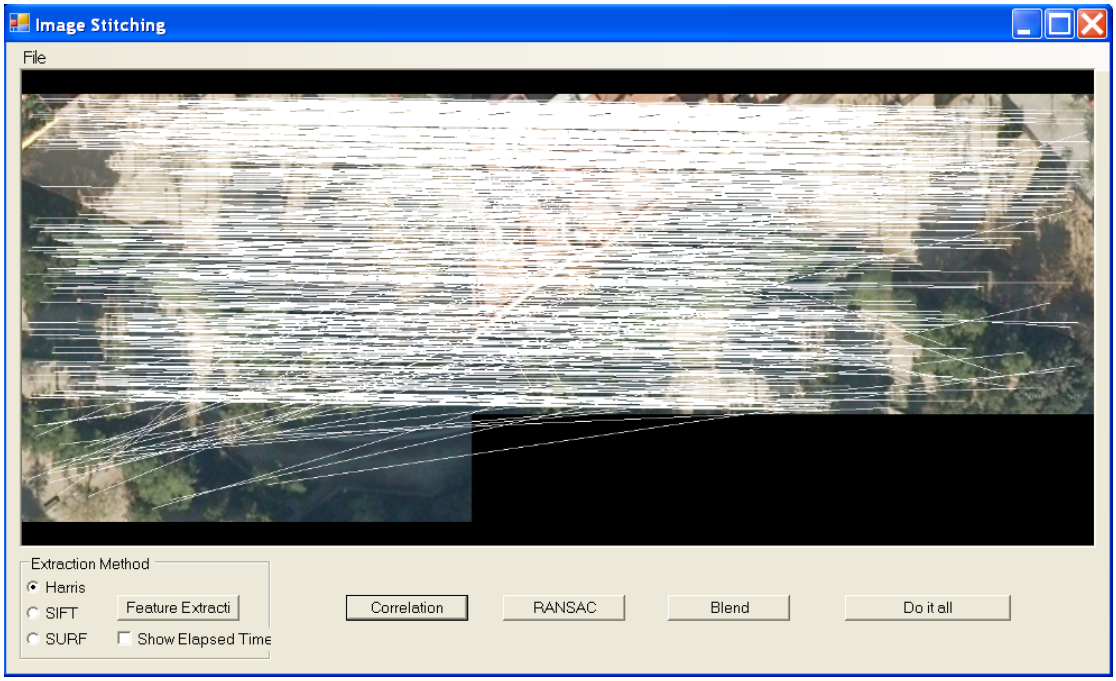

Figure 4.3.1: Pictures with feature points correlated marked in white("Inlier" and "Outlier")

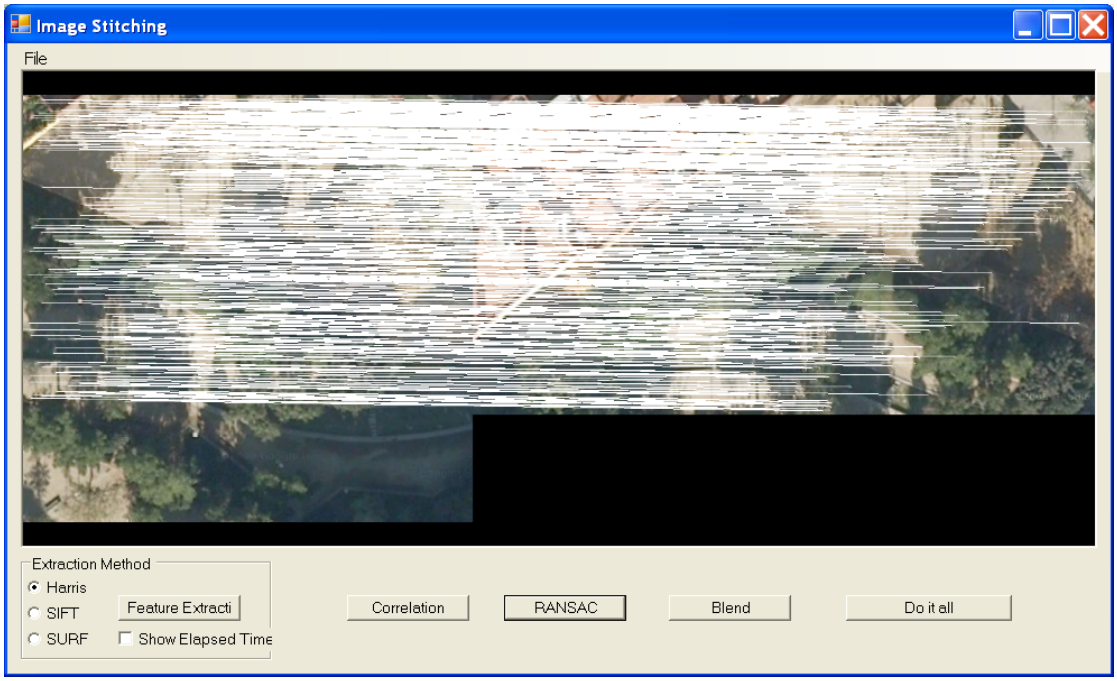

Figure 4.3.2: Pictures after execution of RANSAC (only "Inlier" in white)

The effects of RANSAC can be see in figure 4.3.2; only the correct matches don't disappear in our images.

This happens because RANSAC did find a homography matrix relating most of the points, and discards the incorrect matches as outliers. AForge.net and Accord.net Frameworks gives all the tools to implement RANSAC algorithm.

4.4 Homography Estimation

Having correlated feature points between the two images, it can determine a model that can convert points of one image to the other.

We are talking about an homography matrix, that includes many kinds of transformation. It gives the possibility to overlap two images according position of correlated feature points. Homography permits to use projective transformations, that maps lines to lines (but does not necessarily preserve parallelism). Applying homography matrix it can use affine transformation, that can change an image's shape, giving thesensation that the point of view of the viewer changes.

An affine transformation preserves straight lines and length ratios and so also parallelism is preserved. Basically to apply the transformation it needs a matrix multiplication. Examples of affine transformation are scaling, translation and rotation.

A problem is that translation cannot be represented with a 2X2 matrix product. To overcome this problem, we can add an extra coordinate, k, to every point, so a pixel of the image is identified with $P(x, y, k)$, instead of $P(x, y)$. k is called an homogeneous coordinate. It is a scale parameter. Two sets of homogeneous coordinates indicate the same point if they are a multiple of each other. For example $P(3, 2, 5)$ and $P(6, 4, 10)$ represent the same point. If k is different from 0, you can divide x and y by it to get Cartesian coordinates of the point $(x/k\,,\, y/k\,,\, 1)$. With $k = 0$, point is said to be at infinity.

The size of the homography matrix is a 3x3 matrix with 8 degrees of freedom, this is due to the homogeneous coordinates.

Homogeneous coordinates are essential, if you want to use transformation matrix to make affine transformation; this means adding an further coordinate expanding the two dimensional transformation matrix by one row and column. In this way any linear transformation can be reproduced, as we see in figure 4.4.1, by a general transformation matrix assigning zeros to the last column and row,

except for the lower-right corner, that, for simplification, is fixed at 1.

$$\left\{ \begin{array}{ccc} a & b & 0 \\ d & e & 0 \\ 0 & 0 & 1 \end{array} \right\}$$

Figure 4.4.1: Linear Transformation Matrix

$$\left\{ \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & 1 \end{array} \right\}$$

Figure 4.4.2: General Homography Matrix

The homography matrix, described in figure 4.4.2, has to be multiplied iteratively by our feature points of the first image to found the best correspondence between feature points of the other image, see figure 4.4.3. Obviously a feature point has Homogeneous coordinates identified by the tuple$< x, y, k >$.

$$\left\{ \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & 1 \end{array} \right\} \times \left\{ \begin{array}{c} x \\ y \\ k \end{array} \right\} = \left\{ \begin{array}{c} x' \\ y' \\ k' \end{array} \right\}$$

Figure 4.4.3: Product between Homography matrix and Homogeneous coordinates of a feature point

Figure 4.4.4 describes how to revert a point to his original Cartesian coordinates from Homogeneous coordinates. They need simply divide every coordinate by k, the scale parameter, then, after the division, ignore the last coordinate and take only the first two.

$$\begin{array}{c} \frac{<x',y',k'>}{k'} =< \frac{x'}{k'}, \frac{y'}{k'}, \frac{k'}{k'} >=< \frac{x'}{k'}, \frac{y'}{k'}, 1 > \\ < \frac{x'}{k'}, \frac{y'}{k'} >= P_{xy} \end{array}$$

Figure 4.4.4: Change from Homogeneous coordinate to Cartesian coordinate

4.5 Blending

Thanks to homography matrix and SURF points extraction, it's can join images that have an equal shared zone. As SURF is invariant to affine transformation and the homography matrix exploits this, the two images can be scaled, translated and rotated among them. At this time it must blend together the images according to homography matrix. To do that, it can use a linear gradient alpha to help blending the zones overlapped by the two images. The gradient manipulates final image according to the position of the pixel giving a sort weighted average in accordance with the distance from centers of the two images.

In the figure 4.5.1 is examined the behavior of all the image stitching method using an image that differs from the other for a scale of 70% and a reasonable translation.

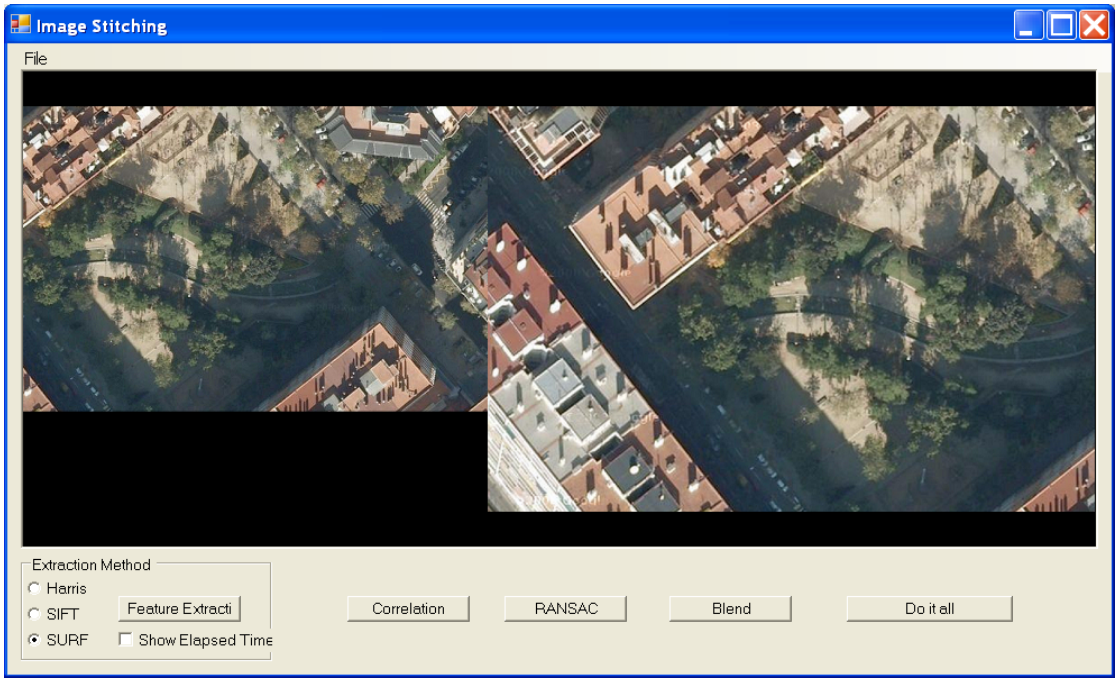

Figure 4.5.1: Pictures translated and scaled between them

As it can see, in figure 4.5.2, the stitching is successful and with 70% of scaling between two images the algorithm can join well. The transformation of scaling corresponds to a change of UAS's altitude, but between two photos, that are taken, approximately, every 5 seconds, is improbably that there is a greater scaling.

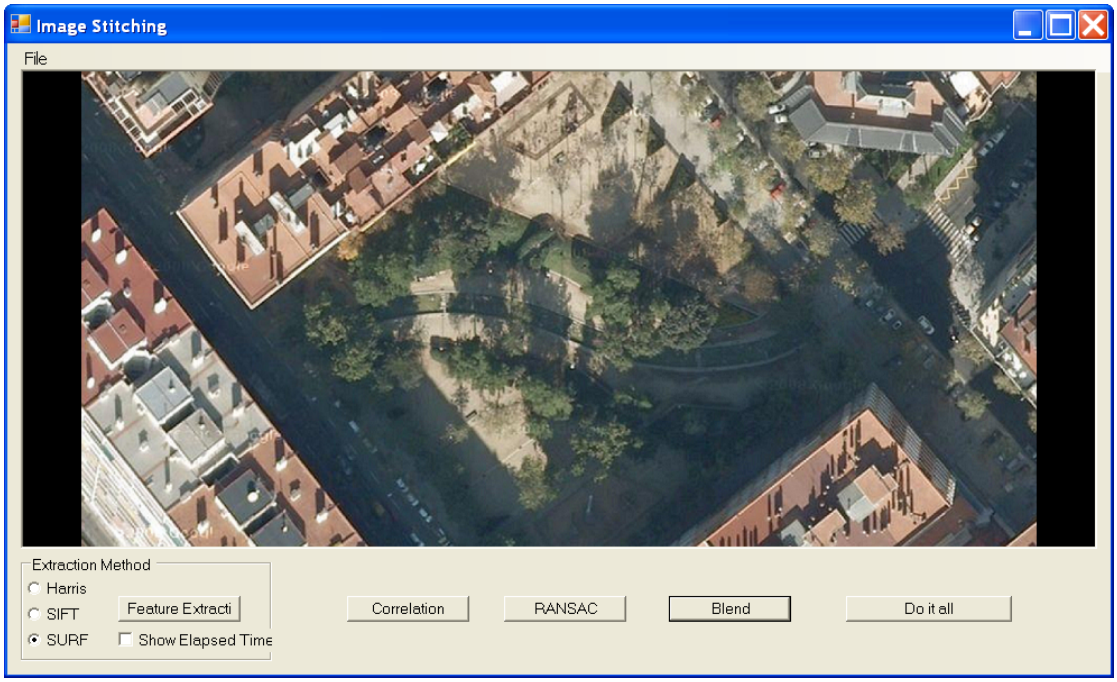

Figure 4.5.2: Final overlap between images translated and scaled

Now let's examine, in figures 4.5.3 and 4.5.4, the performance of the system with rotation and translation of the images, is applied a rotation of 10 degrees and a translation around 20%.

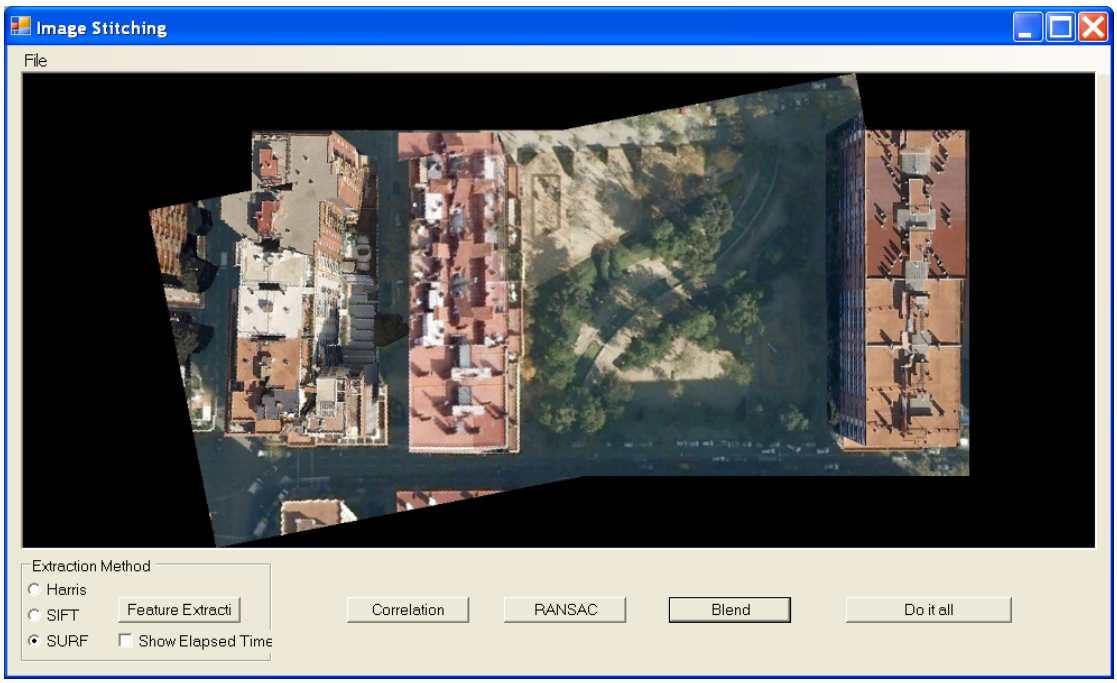

Figure 4.5.3: Final overlap between images translated and rotated

The resultant image, in the figure 4.5.4, seems to be acceptable.

Using this method with an angle of rotation, between the two images, larger than 10 degrees, the quality of final picture is not satisfactory. This is due to feature extraction method, because, in reality, SURF is partially invariant to affine changes. This means that feature points extracted by different images may not be the same if the angle of rotation and length of translation are considerable.

If feature points extracted are different, then the image stitching goes fallacious. This is not a huge problem because directives of project were to join images with, more or less, the same angle of rotation and a little translation. This due to automated UAS's path. Figure 4.5.5 represent directions where UAS move itself and consequently what zones it goes to scan and photograph. For the zone with change of direction, a solution can be to rotate by 90 degrees the images in the correct direction. To know if UAS is changing its direction it can take advantage from telemetry, taken by autopilot AP04[1].

Figure 4.5.4: UAS's Automated Path

As it can see from figure 4.5.5, direction of flying is straight, except for the changes of directions, so, also considering small variations of

directions provoked by external phenomenas, this method of image stitching remain reliable.

5 A possible future develop with GeoTIFF

A possible development of the methodology described in this thesis is to create panoramas using geo-referenced image sequence by using GeoTIFF[15].

The TIFF (Tagged Image File Format) was, in recent decades, widely accepted as a very suitable system for storing, transferring, viewing and printing of raster images, and has therefore been used for managing digital images acquired by satellite, aerial photography and scanned maps.

This was made possible because the format is based on tags. The main advantages of TIFF are:

- suitability for a wide range of applications and its independence by computer's architecture, operating system, and graphics hardware.

- reasonably compactness and managment of black-and-white, grayscale, and color images, allowing a user to adjust for the unique characteristics of a scanner, monitor, or printer.

The TIFF format has the structure shown in figure 5.0.6 . From highest to lowest, the levels are:

- A file Header. The TIFF file begins with an 8-byte header, which gives basic information about the file such as byte

- One or more directories called IFDs (Image File Directories), containing codes and their data, or pointer to the data.

- Data.

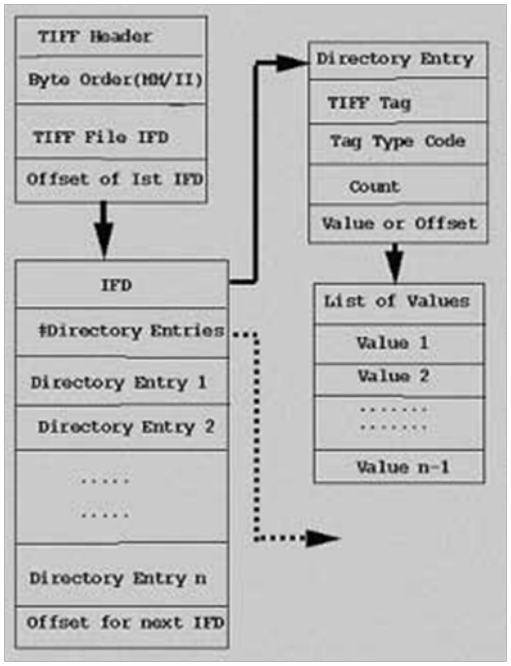

Figure 5.0.1: The File Structure of a Standard TIFF File

To store digital images via satellite, however, was necessary to integrate the geographic information (latitude, longitude, map projections, etc.) to those in the TIFF file format by obtaining GeoTIFF, which is a type of metadata, released into the public domain, which allows to embed geographic reference in an TIFF image. Potentially, it can include what is needed to establish the exact spatial reference for the file.

The GeoTIFF format is fully compatible with the TIFF 6.0 specifications, so the software that cannot interpret the metadata will ignore them, but remained able to view the image. In this way the data could be easily used by different packages GIS (Geographic Information System).

The geographical data of a GeoTIFF file can then be used to view the position the image in the correct location and the geometry on the screen of a display relative to some geographic information.

The GeoTIFF specification defines a set of TIFF tags, called GeoKey, to describe all the cartographic information associated with TIFF images coming from satellites, aerial photography and scanned maps, allowing to connect a raster image to a known image model space GeoTIFF fully complies with the TIFF specification and therefore can use a small set of reserved TIFF tags to store a wide range of infor-

mation for georeferencing.

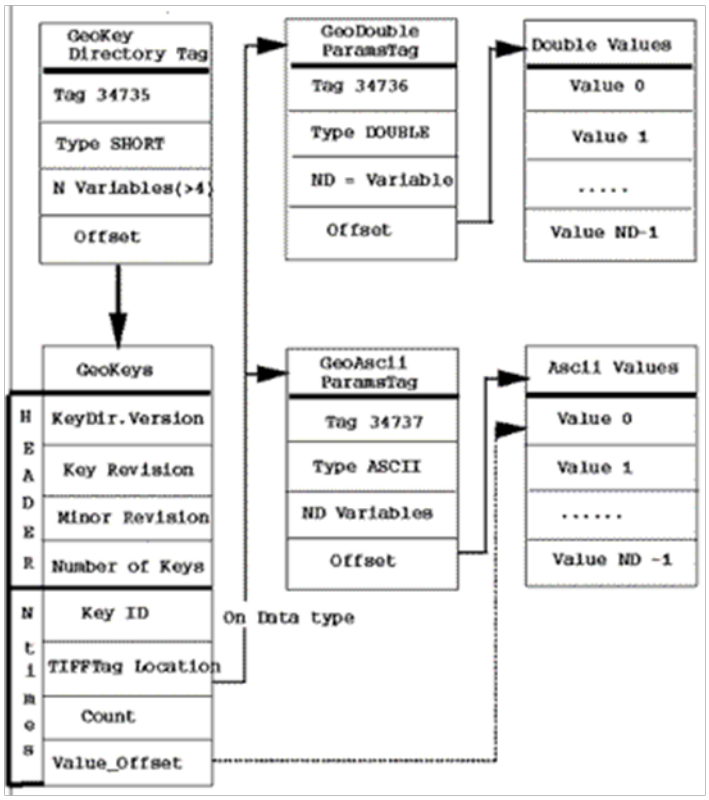

Figure 5.0.2: GeoTIFF file structure

A GeoTIFF file inherits the file structure a TIFF 6.0 and his specific information are encoded in a number of additional tags TIFF confidential and contains private Image File Directory (IFD's), binary structures or other private information invisible to standard TIFF readers.

In the GeoTIFF nomenclature, "georeferencing" refers to tying raster space to a model space M, while "geocoding" refers to defining how the model space M assigns coordinates to points on the earth.

As almost all standard GIS and Image Processing packages support GeoTIFF, it has emerged as a standard image file format for various GIS applications worldwide. The TIFF flexibility to add new Tags and portability has given a lot of scope for GeoTIFF expansion in future.

6 Conclusions

This work has been oriented to the problem of image stitching in order to get realistic panoramic images using photos taken by two camera posed in Unmanned Air Vehicles during flights in the territory of the Mediterranead Tecnologic Park in Catelldefels.

The first part therefore presents the activity undertaken by the ICARUS Research Group of the Technical University of Catalonia.

They concern precisely the development of low cost technologies to build "Architectures for efficient and Civil Unmanned Aerial Systems" and their applications to extend to civil missions with high levels of automation.

This complex system requires the need to implement modern digital avionics as distributed computing architectures using Remote Middleware Architecture for Embedded Applications (MAREA). His function is to provide a quality of service to various applications and is mainly aimed at facilitating the rapid development of services to create a distributed application.

A future goal of Icarus group is to implement a lighter software to run in embedded systems. Marea-based application should be composed of Marea.exe file running on each node of the network that make up the application.

The code implemented in Marea is divided in two principle services: FindPhoto and ImageStitching.

The task of the first service is to find the photo to join; almost always these photos have an overlapping region between them and then may communicate the images that probably can be stitched. The second receives these information and make the stitching. This service is more elaborated than the previous.

The remaining parts of thesis are devoted to examining the activities of image stitching to build a panoramic view by overlapping a series of smaller images. One of the most frequently problems in

image acquisition is the change in light intensity between adjacent images; another problem is associated with the lighting that reflectes light on areas, like zones with glasses or shiny metal.

After the acquisition the images stitching process is divided principally into two phases: the comparison and the blending.

A method of comparing images usually consists of combining the follows main components: the feature set, the similarity measure, search sets and search strategy. However, this method may return incorrect translations due to various reasons: the location, where the similarity measure for the windows is generally minimal can be assumed to provide translations for optimal alignment of the images; if the intensity between adjacent images differs significantly, the absolute value of differences in average intensity may not be a good indication of the similarity of images. As a result of this, this methodology just has not yielded significant results in the experience carried out taking pictures from a helicopter UAS to form panoramas.

For best results are used feature based methods beginning by establishing correspondences between points, lines or other geometrical entities.The local invariant features are used to find matches between all images.

Harris Corner Detector seeks the points that have large intensity variations among neighbourhood. The characteristics of this algorithm lead to finding in most of cases corners and edges. The aim is to extract corners feature and use a normalized cross-correlation of the local intensity values in particular points to match them.

The Scale Invariant Feature Transform (SIFT) approach is based on a feature invariant to achieve the fully automatic panoramic images "seam", allowing reliable matching of image sequences, despite the rotation, zooming and change of illumination in the input images and permitting also the automatic discovering the relationship of correspondence between the images and the recognizing landscapes in a date set unordered.

Speeded Up Robust Features (SURF) is a scale and rotation invariant interest point detector and descriptor, used in computer visions, 3D reconstructions or object recognitions.It is an improvement of the SIFT descriptor, but guarantees a better efficiency.

To compare this last two methods with Harris detector, we have used the same images for both and, in the our case, the number of feature points obtained, more or less, is the same, but the time occurred to extract interest points is two times bigger than Harris method.

Then is used a probabilistic algorithm RANSAC (RANdom SAmple Consensus) that returns acceptable results if more iterations are made distinguishing two types of data: “Inlier” and “Outlier”. “Outlier” are erroneous data, that don’t fit with the model and disobey at the statistical criteria of “Inlier”. Correlation and “Inlier” all the good association between interest points.

Having correlated feature points between the two images, It can determine a model that can convert points of one image to the other by an homography matrix, that includes many kinds of transformation. It gives the possibility to overlap two images according position of correlated feature points. Homography permits to use projective transformations and use affine transformation, that can change an image’s shape, giving the sensation that the point of view of the viewer changes.

At this time is necessary to blend together the images according to homography matrix, to do that it can use a linear gradient alpha to help blending the zones overlapped by the two images. The gradient manipulates final image according to the position of the pixel giving a sort weighted average in accordance with the distance from centers of the two images. The resultant image seems to be acceptable because the directives of project was to join images with, more or less, the same angle of rotation and a little translation. This due to the specific automated UAS’s path. A future implementation of these methods are to georeference the panoramas using GeoTIFF.

Bibliography

[1] Pablo Royo Chic, Enric Pastor Llorens, Cristina Barrado Muxi. An Open Architecture for the Integration of UAS Civil Applications, May 2010

[2] J. Lopez, Pablo Royo Chic, Enric Pastor Llorens, Barrado C., Santamaria E. 2007 (November). A Middleware Architecture for Unmanned Aircraft Avionics.

[3] C. Barrado, J.Lopez, 2009 - Marea User Manual

[4] http://icarus.upc.edu/

[5] http://icarus.upc.edu/publications

[6] Royo P., Lopez J., Pastor E., Barrado C., 2008 (Jan). Service Abstraction Layer for UAV Flexible Application Development.

[7] Chia-Yen Chen 1999 Image Stitching - Comparisons and New Techniques

[8] Matthew Brown and David G. Lowe. Recognising panoramas.International Conference on Computer Vision (ICCV 2003), Nice, France (October 2003)

[9] Matthew Brown and David G. Lowe 2007. Automatic Panoramic Image Stitching using Invariant Features.

[10] Chris Harris and Mike Stephens-Plessey Research Roke Manor, United Kingdom 1988. A combined corner and edge detector.

[11] C.De Souza Automatic Image Stitching with Accord.NET

[12] Herbert Bay, Andreas Ess , Tinne Tuytelaars, Luc Van Gool 2006 Speeded-Up Robust Features (SURF)

[13] http://www.pages.drexel.edu/~weg22/can_tut.htm

[14] http://cmp.felk.cvut.cz/cmp/courses/dzo/resources/tutorial-pollefeys-eccv/node53.html

[15] http://www.gisdevelopment.net/technology/ip/mi03117pf.htm

[16] L. G. Brown, “A survey of image registration techniques”, Computing Surveys, vol. 24, 1992, no. 4, pp.325-376.

[17] C. Chen and R. Klette, “An image stitcher and its application in panoramic movie making”, Proc. DICTA’97, Dec. 1997, pp.101-106.

[18] D. L. Milgram, “Computer methods for creating photomosaics”, IEEE Trans. Comput., vol. C-24, Nov. 1975, pp.1113-1119.

[19] David G. Lowe, "Object recognition from local scale-invariant features," International Conference on Computer Vision, Corfu, Greece (September 1999), pp. 1150-1157.

[20] http://homes.esat.kuleuven.be/~tuytelaa/tutorial-ECCV06.pdf

[21] http://amath.colorado.edu/courses/5720/2000Spr/Labs/Haar/haar.html

[22] http://en.wikipedia.org/wiki/Blob_detection#The_determinant_of_the_Hessian