PANORAMA IMAGING FOR MOBILE PHONES

MIGUEL BORDALLO

Bordallo M. (2010) Panorama Imaging for Mobile Phones. University of Oulu, Department of Electrical and Information Engineering. Master's thesis, 58p.

ABSTRACT

A panoramic image is a wide-angle mosaic image done by combining several still images. Panoramas usually refer to single viewpoint images, created by rotating the camera around its optical center. Digital panoramas, on the other hand, are assembled from a series of perspective images through a computationally costly and a memory hungry multistage process.

This thesis describes a real-time implementation of a video based image stitcher, designed to create high-quality image mosaics on a mobile phone, which can be integrated on the image pipeline. Handheld devices such as mobile phones have limited memory and processing capabilities. The slow performance or lack of floating point arithmetic units on mobile processors requires a careful selection of the panorama construction algorithms and their implementations.

In the context of this thesis, several solutions have been developed to adapt the computationally costly application to a small footprint platform. These include fixed point implementations of the most expensive algorithms and the use of a carefully tailored memory management.

The application runs in real time on multimedia phones, producing 360 degrees panorama pictures on the fly. In most practical cases the quality of the pictures for a human viewer is indistinguishable from optical panoramas.

Keywords: panorama, video, mobile phone, stitching, mosaic

Bordallo M. (2010) Panoraamakuvantaminen matkapuhelimilla. Oulun yliopisto, Sähkö- ja tietotekniikan osasto. Diplomityö, 58 s.

TIIVISTELMÄ

Panoraamalla tarkoitetaan yleensä kuvaa, joka on luotu kiertämällä kameraa sen optisen keskipisteen ympäri. Panoraamakuva voi olla myös erillisistä kuvista muodostettu laajakulmainen mosaiikkikuva. Digitaalisten panoraamojen koostaminen sarjasta perspektiivikuvia on monivaiheinen prosessi, joka vaatii suorittimelta runsaasti resursseja.

Tämä tutkimus esittää kuvamosaiikkien luomiseen matkapuhelimella tarkoitetun videopohjaisen järjestelmän reaaliaikaisen toteutuksen, joka voidaan integroida kuvankäsittelytoimintoihin. Kannettavien laitteiden muistirajoitteet, pieni laskentateho ja liukulukuyksikön puute edellyttää panoraamalgoritmien huolellista valintaa ja toteutusta.

Panoraamakuvauksen vaatimista algoritmeista on kehitetty useita ratkaisuja, joissa suorittimelta runsaasti resursseja vievä sovellus on mukautettu pienen laskentatehon alustalle. Lisäksi eniten resursseja vievät algoritmit on toteutettu kiinteän pisteen tekniikoilla ja niiden muistinhallinta on räätälöity.

Sovellus toimii reaaliajassa multimediapuhelimissa ja tuottaa 360 asteen panoraamakuvia kuvanottotahdissa. Useimmissa käytännön tapauksissa ihminen ei pysty näkemään eroa tuotettujen kuvien laadussa verrattuna optisiin panoraamoihin.

Avainsanat: panoraama, sulautettu järjestelmä, mosaiikki

Bordallo M. (2010) Imágenes Panorámicas para Teléfonos Móviles. Universidad de Oulu, Departamento de Ingeniería Eléctrica y Procesado de Información. Proyecto Fin de Carrera, 58 p.

RESUMEN

Llamamos imagen panorámica a un mosaico obtenido por la combinación de varias imágenes sencillas. El término "panorama" normalmente se refiere a una imagen con un solo punto de vista, creada al rotar la cámara alrededor de su centro óptico. Las imágenes panorámicas digitales, sin embargo, están compuestas a partir de una serie de perspectivas mediante un proceso multietapa, computacionalmente muy costoso y que conlleva un gran consumo de memoria.

Esta Proyecto Fin de Carrera describe la implementación de una aplicación diseñada para componer imágenes panorámicas de gran calidad a partir de fotogramas de una secuencia de vídeo obtenidos con un teléfono móvil y que puede ser integrada como parte del sistema de adquisición de imágenes del mismo. Los pequeños dispositivos electrónicos como los teléfonos móviles, tienen una cantidad de memoria limitada y unas capacidades de proceso muy reducidas. La falta de unidades de punto flotante en los procesadores móviles o el bajo rendimiento de las mismas, requieren una cuidadosa selección de los algoritmos y las implementaciones a utilizar.

En el contexto de este Proyecto Fin de Carrera, han sido desarrolladas numerosas soluciones técnicas para adaptar este tipo de aplicaciones a una plataforma con capacidades reducidas. Las soluciones incluyen implementaciones con aritmética de punto fijo de los algoritmos más costosos y el uso de un cuidadoso manejo de la memoria utilizada.

La aplicación desarrollada, es capaz de producir imágenes panorámicas de 360 grados en tiempo real. En la mayoría de casos prácticos, la calidad de las imágenes es indistinguible para el ojo humano de las obtenidas con sistemas ópticos.

Palabras clave: Panorama, teléfono móvil, composición de imágenes, mosaico

CONTENTS

FOREWORD

This Master's Thesis was done in the Machine Vision Group of the Department of Electrical and Information Engineering at University of Oulu.
I would like to express my gratitude to Professor Olli Silvén for giving me the chance to work in the MVG and for supervising this thesis. Without his invaluable ideas and guidance, this thesis would never have been completed.
I would like to thank Dr. Jari Hannuksela for reviewing the thesis manuscript. His valuable comments greatly improved the quality of the thesis. Thanks to Dr. Jani Boutellier for advising and helping me at an early stage of my research.
Thanks to Dr. Markku Vehviläinen and Dr. Marius Tico from Nokia Research for their invaluable ideas and useful feedback during the project.
Many thanks to my working room mates and my colleagues in the Machine Vision group for creating such a nice working environment.
I would like to thank my family for their support from the distance and for their care and love over the years.
Finally, I also want to thank Hillevi for her love, support and patience even when she had to read my thesis over and over.

Oulu, April 2010

Miguel Bordallo

GLOSSARY

ARM	32-bit RISC processor architecture widely used in embedded designs
CCD	Charged Coupled Device. Used often as a synonym for a type of image sensor
CPU	Central Processing Unit
DLL	Dynamic Link Library
DSP	Digital Signal Processor
FFT	Fast Fourier Transform
FPS	Frames per Second
GPU	Graphics processing Unit
LCD	Liquid Crystal Display
MMF	Multi Media Framework
MVG	Machine Vision Group
OS	Operating System
PDA	Personal Digital Assistant
RAM	Random Access Memory
RGB	A color space that is based on the three additive primary colors, Red, Green, and Blue
UI	User Interface
Y	Luminance of a picture, usually a function of Red, Green and Blue values of a pixel

1. INTRODUCTION

Smartphones are normal mobile phones with some extra features that make them the perfect all-in-one handheld device. They offer access to the internet through GPRS, 3G or Wireless LAN adapter, they can record and reproduce sounds, they can be used as a personal digital assistant (PDA), and it is possible to install in them some applications that allow mobile banking or online purchase, maps, positioning systems, music players and so on.

The memory capacity and the speed of the processors of handheld devices are improving all the time, showing better capabilities that create opportunities for new applications. Most smartphones have built-in digital cameras with still image and video capturing. Smartphones have a fast processor and some free RAM memory and some even offer a floating-point unit and hardware graphics accelerators. With the available imaging and computational capabilities, we may conclude that computer vision applications are among the developing paths on mobile platforms. Figure 1 depicts two mobile communications devices both with two cameras: Nokia N93i and Nokia N900.

Although the imaging capabilities of mobile phones have improved close to the performance of digital cameras, the lack of space makes it almost impossible to provide for enough image detail or very wide angle capabilities. As a replacement for optics, image sequences can be used as a source for wide angle images. Taking high resolution still images is a slow process that requires a large amount of offline processing. Video sequences, on the other hand, can be used in an interactive manner in real time to capture and compose a high resolution panorama, trading off the optical performance for computational power.

Figure 1. Two mobile communications devices with two cameras.

Panorama construction from a video sequence is a complex multi-stage process where several subtasks have to be performed. Each one of the tasks presents their own problems that have to be solved in order to achieve a good final image. The main tasks that have to be performed are an accurate image registration system, the selection of the frames that are going to be used and the blending of these frames in order to be combined into a single panorama.

The main problems that have to be faced when thinking about an interactive handheld panorama construction depend on the camera and the environment. These include eventualities such as moving objects across the scene, color imbalance problems, dynamic change of gain or the occasional blur of some frames due to motion or autofocus. This cannot be ignored when trying to obtain quality images. Figure 2 represents an example panorama image with some of these typical problems.

Figure 2. Example panorama with moving objects and luminance changes.

1.1. Scope of the thesis

In this thesis, a series of problems on the way to a complete interactive handheld panorama application have been considered. The main contribution of this thesis is the development of a 360 degree real-time video based panorama application for a mobile phone. This application is also the first one whose implementation is described.

Panorama imaging needs some sub-tasks like frame capturing, image registration, and blending. Some of these tasks have been applied in several other purposes and can alone represent a substantial field of study.

Image blending is a computationally costly low level image processing operation that needs to be performed in real time and that will determine the quality of the final picture. The source images for the blending stage have to be carefully selected with image quality assessment methods such as moving object detection or blur computation, forming together a complete frame selection process. The images that are going to be selected need an accurate image registration system that is capable of identifying the matching parts of the individual frames. These frames can be captured as a video sequence in an interactive manner, as opposed to the slow high-resolution still image capturing. This approach allows us to take advantage of the small variations between frames and defines the requirements of the panorama algorithms.

The Machine Vision Group (MVG) at the University of Oulu has made significant progress in panorama imaging and has developed different algorithms for each one of the sub-tasks. These algorithms cannot be used directly in handheld devices. In this thesis, the goal was to study, adapt and implement some of these algorithms in a handheld embedded system environment as well as providing a working solution capable of executing on a mobile platform with a good performance.

1.2. Structure of the thesis

This thesis is organized as follows: First, general definitions on panorama imaging and different approaches to it are given in Chapter 2. Several currently used stitching algorithms are also described. Chapter 3 deals with the multimedia phone platform that has been used for the development. In Chapter 4, the tasks of a stitching algorithm are described and possible solutions are discussed. In Chapter 5, the actual implementation of the main tasks of a panorama algorithm, such as image registration and image stitching on the target device, is described. The chapter also shows how to improve the panorama construction system with a video frame selection stage. Chapter 6 describes the implementation of the application and its architecture and memory management. Some performance experiments and results are also presented. Chapter 7 gives a summary of the whole process.

2. PANORAMA IMAGING

Traditionally, a panorama is a wide angle image that shows at least a horizontal field of view as wide as the human head-eye combination is capable of. Often, left-to-right widths up to 360-degree are used.

To be able to obtain a panorama result multiple images are needed, as well as a stitching method, to compose a wide image from the input frames. The amount of needed pictures and the aspect of a 360-degree panorama depend on the lens used as well as on the overlap between images. For example, a 360-degree panorama with a standard 35mm focal length lens can have an aspect ratio comparable to 57:9, which corresponds to 19 horizontal images. If a wide-angle lens (18mm) is used, the panorama can become about 57:14 or 13 horizontal images. Mosaic pictures are generalized panoramas where the camera movement is not horizontally restricted. Figure 3 represents an example of panorama and mosaic constructing.

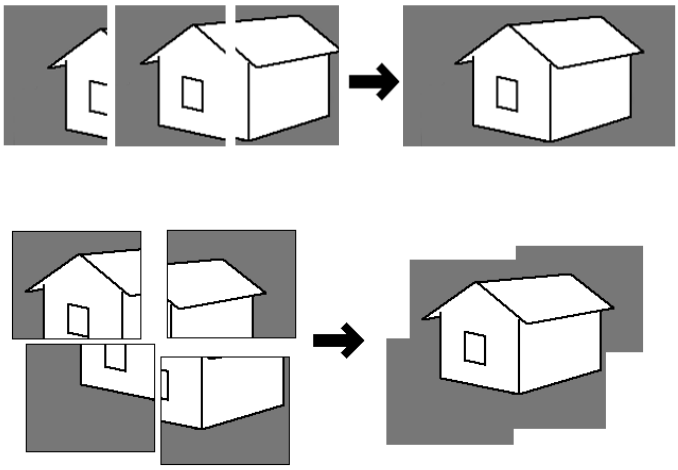


Figure 3. Example of panorama (up) and mosaic (down) constructing.

2.1. Traditional panorama imaging

Originally, panoramic imaging had to be done with conventional film cameras and without the aid of computers. Several methods have been used:

- **Conventional method –** This method consisted only of cropping or masking the upper and lower part of a conventional image to give the false idea of a wider field of view. A wide angle lens gives better results.
- **Segmented panorama** – This panorama is just a group of pictures, usually taken at the same time, that overlap about one third with the next and the previous one.
- **Swing lens panorama** – This kind of panorama has a big advantage over using a fish-eye lens on a regular camera. The film is held by a curve support, while the lens moves from one side to another in the camera. It is possible to obtain an up to 150-degree horizontal field of view.
- **Rotational panorama** – These panoramas are made by cameras that can rotate around a single point that acts as the point of view for images. The film

moves in the opposite direction at the same speed as the camera does. Some of these cameras have existed since the early 1900s.

- **Strip panorama** – This kind of images are usually taken with stationary cameras. The film is moving at the same speed as the moving image. It is also possible to use them by moving the camera in a linear way.

2.2. Types of digital panorama images

In the case of digital panorama pictures, we can distinguish among several ways of computing the result. Depending on the use that we are going to give to the image, we can make a choice. Basic types of digital panorama images are:

- **Flat or planar panorama** – A conventional picture taken with only one shot is just a planar picture. These flat images are intended to be viewed without any perspective correction. A panorama made this way is just a wider angle conventional picture made from a group of stitched images. In planar panoramas we can recognize a starting and ending point. Flat panoramas are the best choice if we need to print or publish them. Figure 4 depicts a flat panorama image.

Figure 4. Flat Panorama taken with the developed application.

- **Circular-planar panorama** – If the coordinates of a flat panorama are converted into polar coordinates, a circular panorama is obtained. For a 360-degree panorama, the result is a circle where the whole field of view is covered. Circular panoramas present deformation in both axes. Figure 5 shows a circular panorama.

Figure 5. Circular Panorama taken with the developed application.

- **Cylindrical panorama** – If the picture is intended to be viewed as if it were curved around one point like the inner face of a cylinder, a cylindrical panorama is needed. These pictures can be viewed surfing them horizontally with a left to right eternal loop. Images of this kind have to be 360-degree panoramas, and viewing software is needed. Displaying these images as flat will show curves in the horizontal axis. Figure 6 represents a cylindrical panorama.

Figure 6. Example of a cylindrical panorama.

- **Spherical panorama** – Spherical panoramas are stored as equirectangular image files. They represent 180-degree vertical axis and 360-degree horizontal axis. If not all of this field of view can be covered, blank pictures can fill the empty zone. These images are intended to be viewed from the center of a hypothetic sphere, and when they are shown in a flat format, the horizontal and vertical axes are curved. In a spherical projection of a panorama, the horizon is represented by a circumference around the central point of a flat image. Horizontal distance is represented then as angular distance and vertical distance is represented by radial distance to the geometrical centre. The main difference with this format when compared to the others is that the very top and the very bottom of the image can also be displayed. Special viewing software is needed in order to display this kind of images. Figure 7 represents a spherical panorama.

Figure 7. Example of a spherical panorama.

- **Cubic panorama** – The projection of this 360-degree horizontal axis and 180-degree vertical axis is done in a flat way towards the six faces of a cube. Six different images are stored. A special viewer is needed in order to display them, but each face of the cube can be printed or published the same way as a planar image. Figure 8 shows an example of a cubic panorama.

Figure 8. Example of a cubic panorama.

2.3. Different approaches to panorama stitching

As digital imaging devices are becoming cheaper and better, more and more possibilities to compose panorama images are appearing. Several approaches to image capturing can be discussed. Traditionally, stitching panoramas from several still-frames has been the most developed approach. Nowadays video capabilities are available in most of the digital imaging devices. Video sequences as a source present some pros and cons that deserve to be studied and discussed.

2.3.1. Panorama from several pictures

The most common way to approach to panorama imaging consists of stitching several pictures that overlap at least the 30% to enable precise registration. The pictures are carefully selected, taken exactly from the desired point and on the same horizontal or vertical plane. A tripod is normally used to avoid undesired rotation and tilting.

Nowadays, digital compact cameras and other handheld devices are so popular that trying to compose a panorama image has become usual. Several pictures are then taken in order to match the requirements, and after that, they are stitched automatically with the help of a computer.

The main advantage of taking a panorama from a set of pictures is the possibility of carefully taking each one of the panorama parts. Every picture can have a good resolution and quality even after image compression. This is a good approach for

users that do not mind storing a set of pictures and stitching them afterwards with a quality algorithm in a personal computer.

2.3.2. Panorama from a video sequence

Another approach to panorama imaging can be the use of a video sequence as the source for image capturing. Video applications have usually been designed for content creation, access and replay. Nowadays, it is also possible to employ video to enable new applications. In this case, the motion information can be extracted from the image sequence captured by the camera to construct a real-time panorama builder.

Video sequences usually have less resolution than still images. However, they have a high frame-rate and a big overlapping area between consecutive frames. The high redundancy of the images allows us to construct the panorama and provides compensation for a lower resolution.

Shooting a video sequence to make a panorama is a very intuitive process. The starting point of the video is the beginning of the mosaic and then it can be moved to desired directions with soft transitions. To get a panorama image from a video, usually a sequence is recorded in order to post-process it. Each frame from the video is separated and treated as a single image that overlaps for example, around the 90% with the previous one.

In addition to interactivity benefits, the use of low resolution video imaging can be defended from purely technical aspects. In low resolution mode the sensitivity of the camera can be better as the effective size of the pixels is larger, therefore reducing the illumination requirements and improving the tolerance against motion blur.

This thesis also explains the method developed by Boutellier and Silvén [1] that specifies how to select only the best group of frames in a video sequence to capture a panorama and discarding the others. The panorama quality improves with sharper input images.

2.4. Applications of panorama imaging

Graphical panoramas are a valuable tool to display a 3D world environment in an easily understandable manner. In cartography, panorama techniques have been used to map three dimensional objects into a two dimensional image. With the development of the automatic stitching systems, a wide field of applications to these techniques have been developed.

Nowadays, visual surveillance systems employ panorama stitching from multiple camera data using it for several industrial and security processes [2]. Another possible application is the development of a handheld scanner either for documents or large posters using the blending of some frames. A multirow scan of a document can lead to a high resolution mosaic picture suitable to be digitally stored for later uses.

2.5. Panorama algorithm types

Generally, the panorama stitching algorithms can be separated into two categories. Batch-type algorithms are applied offline after the images that will compose the mosaic are already taken. These algorithms process every frame one by one and then stitch them together into a single image.

The other group of algorithms, in which the proposed solution leans on, is related to the real-time algorithms. In these kinds of solutions, the frames are processed as they are being captured, and the resulting image is obtained just when the capturing ends. The processing requirements for this second group are demanding as several approximations may be needed due to the target device constrains.

2.5.1. Batch-type algorithms

Batch-type panorama algorithms have been used in Personal Computers for several years and they are therefore well known. With these algorithms, the starting point is a series of already taken images that are supposed to be blended into a single scenario panorama picture.

In fully automatic algorithms, each single picture is registered against the others to obtain the relative position in the whole landscape or scenario. Once this position has been determined, the images are blended into the final mosaic, correcting the conflicting points.

2.5.2. Real-time algorithms

Real-time panorama algorithms are relatively new because substantial amounts of resources are needed. In these algorithms, the starting point is a single frame that is used as the beginning of the panorama image that grows as the frames are coming. Most of the processing sub-tasks for each frame must be done before capturing the next one. These algorithms can be fully automatic or they may require some interaction from the user to achieve better performance.

The interactivity of real-time algorithms presents a considerable advantage over batch-type realizations, but there are also disadvantages: The quality may suffer due to streamlined stitching techniques. However, the possibility of a high overlap between frames and the chance of providing user feedback during the capture and stitching, leads to satisfactory results.

2.5.3. Prior Implementations

There are some commercial solutions already in the market in both algorithm categories. Canon PhotoStitch [3] implements a batch type algorithm as a part of the EOS Cameras software package. PhotoStitch allows a computer user to select, display and merge a series of pictures taken with a digital camera specifying certain parameters such as the approximate overlap between pictures and the focal length. Some other solutions following the same overall scheme are PanoStitch, 3DVistaStitcher, the plug-ins integrated into Photoshop Elements or the Microsoft Stitching software present in Digital Image Pro, Digital Image Suite, and Windows Live Photo Gallery [4].

Real-time algorithms have been developed for embedded systems such as digital cameras or mobile phones. Bitside Panoman, Scalado Autorama [5], the latest QuickPanorama solution from Morpho Inc [6], and the panoramic shooting mode of SonyEricsson and Nokia phones are some of these commercial solutions that can be found.

SonyEricsson's application relies mostly on the user feedback to get good captures for a total amount of 3 or 5 frames. The user is instructed on how to shoot the next frame from a good point and the blending process is performed [7].

First version of Panoman and Nokia panoramic mode work quite similarly. The user is asked to move the camera in a smooth manner, and some frames are shot on the process, converted into panorama stitching blocks and blended together on a fixed size resulting image that can be composed of several frames [8]. The latest version of Panoman works as a mix of the first version and SonyEricsson´s application but it allows high resolution frames with the cost of several minutes of post-processing time.

3. MULTIMEDIA PHONE PLATFORM

Implementing mosaic stitching for a mobile environment is more challenging than implementing such a system, e.g., for a PC environment. On the other hand, when compared to a 'generic' implementation, the specificity of the mobile environment requires knowing the internals of the chosen platform and environment.

Modern mobile communication devices are becoming attractive platforms for multimedia applications as their display and imaging capabilities are improving together with the computational resources. Many of the devices have two built-in cameras, one for high resolution still and video imaging and the other for obtaining lower, e.g., VGA resolution (640x480 pixels) frames. Handheld devices offer versatility of imaging equipment in comparison to laptop computers.

To illustrate the current typical designs, we consider two modern cellular phone designs that both contain two cameras (see Figure 1). The flip phone has a high resolution camera on the cover, on the same side with a display. However, it is intended to be operated with the lid open, exposing a higher resolution display and an additional camera to the user. The monoblock design is similar to digital still cameras with the high resolution display and cameras on opposite sides. It is obvious that with the display side cameras the designers have aimed at handheld video telephony, while at the same time satisfying the needs of occasional photography, video capture, and playback. Figure 9 depicts a simplified scheme of the organization of a typical portable multimedia device.

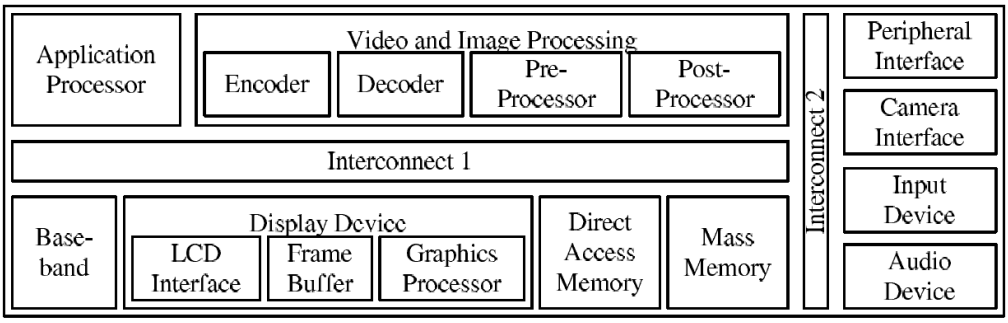


Figure 9. Organization of a typical portable multimedia device.

Nokia N93 Smartphone was used as the main target hardware for the development of the panorama application. The N93 phone is running Symbian OS v9.0, and Series 60 3rd Edition platform. A previous version of the application running on a Nokia N90 phone with Symbian OS v8.1 and Series 60 2nd Edition has been also developed for testing purposes. In the development of the panorama stitching application, the quality of the main camera was the most important feature in selecting the target device.

3.1. Symbian S60 multi-media framework and camera access

Symbian OS, produced by Symbian Ltd., is a proprietary operating system, designed for mobile devices, with associated libraries, user interface frameworks and reference implementations of common tools. It is a descendant of Psion's EPOC and runs exclusively on ARM processors.

For hardware, Symbian OS is optimised for low-power battery-based devices and for ROM-based systems. There is a strong emphasis on conserving resources, using Symbian-specific programming idioms such as descriptors and a cleanup stack. There are similar techniques for conserving disk space. Furthermore, all Symbian OS programming is event-based, and the CPU is switched off when applications are not directly dealing with an event [9].

The S60 Platform (formerly Series 60 User Interface) is a software platform for mobile phones that uses Symbian OS. S60 consists of a suite of libraries and standard applications, such as telephony, PIM tools, and Helix-based multimedia players. It is intended to power fully-featured modern phones with large color screens, which are commonly known as smartphones. Within the main characteristics of S60 platform we can find the allowance of multitasking, the standardization of several interfaces such as buttons or devices access and the possibility of executing Java MIDP and Symbian C++ applications [10].

A multimedia framework is a software framework that handles media on a computer and through a network. A good multimedia framework offers an intuitive API and a modular architecture to easily add support for new codecs or container formats. It is meant to be used by applications such as media players and audio or video editors [11].

Symbian Multi-Media Framework is known as MMF. It is composed by a MMF Client API that handles all the interaction with the sound, audio and video devices through the MMF Controller Framework. This framework can access the audio and video controller plug-ins where several codecs and formats can be used. Symbian MMF codecs and formats can be easily extended through the API. Figure 10 presents the organization of the Symbian multimedia framework [12].

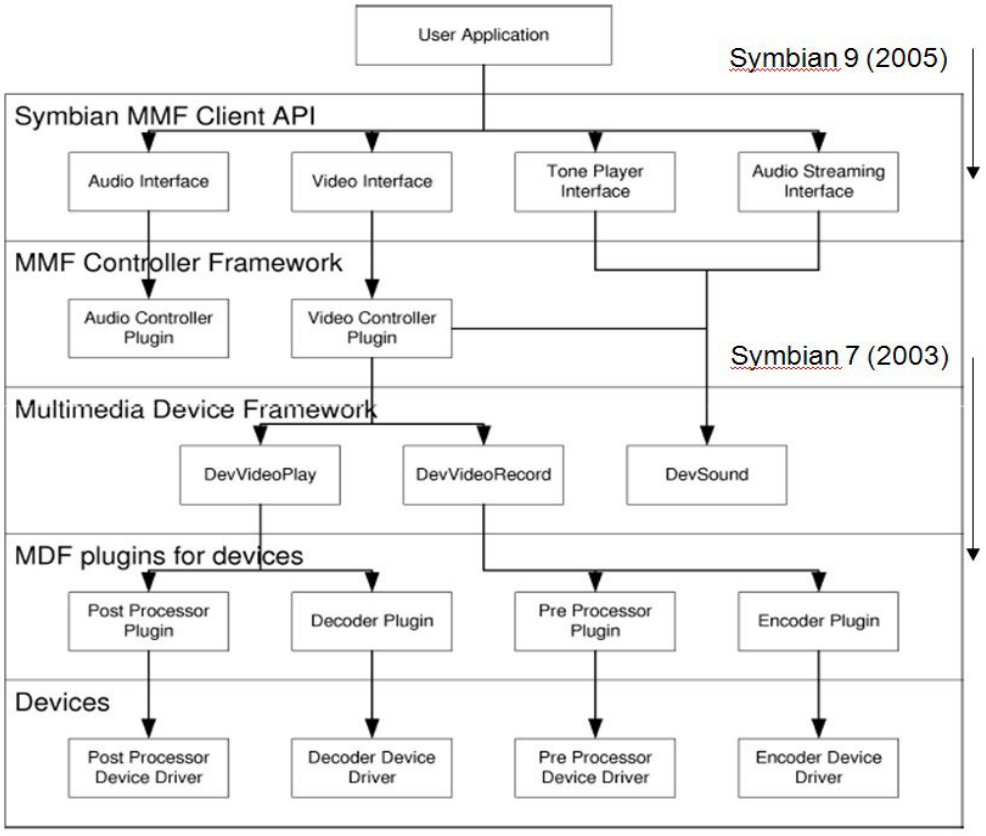


Figure 10. Nokia Symbian S60 Multi-Media Framework architecture

The camera device can be accessed through the camera API. With similar structure as the MMF, a Client API handles all the interaction between the developer and the device through a Camera Controller Framework.

3.2. Nokia N93 mobile phone

Nokia N93 phone is a clamshell-shaped, twistable-screen mobile phone that includes several features that can be used for the stitching application purposes. It includes a three megapixel camera with both optical and digital zoom. A smaller VGA camera is also present pointing towards the user and can be mainly used for video conference. Figure 11 depicts the internal organization of a Multimedia Cellular device [13].

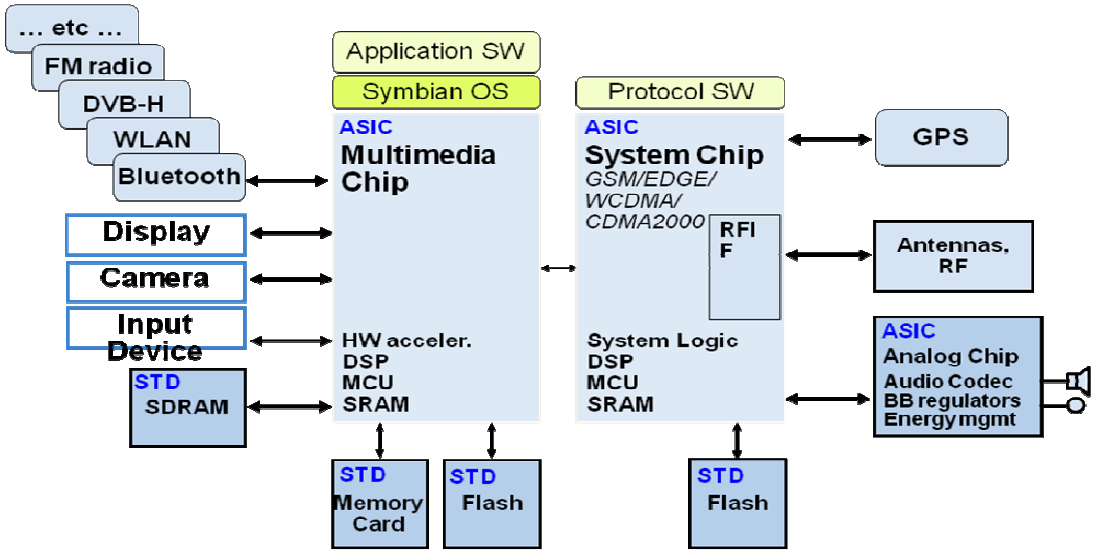


Figure 11. Internal organization of a Multimedia Cellular device.

The application processor of Nokia N93, the Texas Instrument OMAP2420 controller, includes an integrated ARM1136 processor (330 MHz), a TI TMS320C55x™DSP (220 MHz), 2D/3D graphics accelerator, imaging and video accelerators, high-performance system interconnects and industry-standard peripherals.

The multimedia enhancements of OMAP2420 include an imaging and video accelerator for higher-resolution still image capture application and video encoding and decoding of VGA resolution sequences at 30 frames per second. A floating-point unit and an OpenGL ES 1.1 compatible GPU 2D/3D graphics accelerator are available. The graphics processing unit can be used even for image processing by feeding camera images to the textures interface.

The available RAM memory, when the operating system is started and several basic applications are running, can be up to 20MB, offering enough space to manipulate uncompressed resulting images. Figure 12 shows a Nokia N93 device with the screen flipped to capture camera images.

Figure 12. A Nokia N93 phone.

N93 imaging capabilities

The two cameras present in the device can be accessed through the Symbian API. The API allows capturing up to three megapixel images and up to 640x480 raw video frames with a frame rate up to 15 frames per second. Still frames can be directly obtained from the camera at several resolutions either as raw bitmaps or JPEG objects.

The N93 screen presents a fixed resolution of 320x240 pixels but several fast scaling functions are available. The video frames are received as an ARGB viewfinder image that can be drawn directly in the screen or processed before, but also YUV frames can be obtained directly from the camera without any conversion. Also, a compressed video interface can be used, resulting in an encoded file that is stored on the memory card, but with no access to individual frames.

N93 phone offers the possibility of switching between two cameras. Both cameras, main and videoconference, can be used in the same way, but it is not possible to use them at the same time, and fast changes between them are also not implemented on the API.

4. PANORAMA FROM MOBILE DEVICE VIDEO

4.1. Multistage approach

A panorama algorithm needs to go through several stages, each of which can be treated as a separate problem. These stages have been case of study of their own, and several solutions can be chosen to deal with the needs of a complete algorithm. Figure 13 shows a multi-stage scheme of a panorama algorithm.

The first stage refers to the capturing of every frame of a video sequence while second stage deals with the preparation of each frame through cropping or pre-processing. The second stage describes how to register every frame against the previous ones. The registration task has several sub-tasks that depend on the selected method.

The third stage in the system is the computation of the quality of every frame to enable the selection of the most suitable ones based on several criteria. Along with the evaluation of the quality, also corrections can be performed to every frame to normalize certain values. The final phase of the algorithm blends the selected frames together into one image to obtain the resulting image that can be saved in a mass storage device.

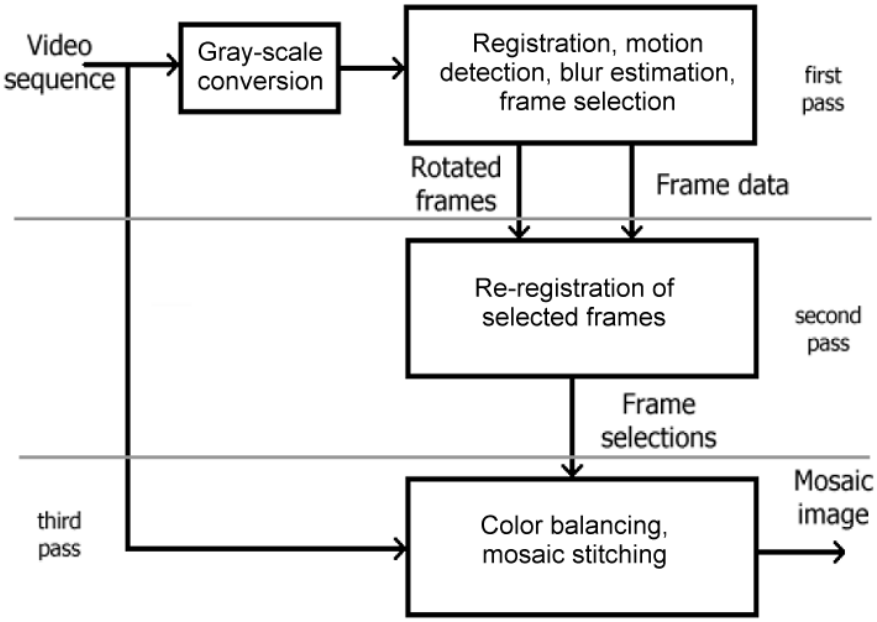


Figure 13. Scheme of an example multi-stage panorama algorithm.

4.2. Projection and camera motion model

The first step of designing an algorithm capable of stitching several frames into a single image is defining a camera model and the projection desired. The projection model will express how a set of real 3D objects are going to be translated into 2D images. Several projections and models can be used to achieve this, but for the definition of a video based panorama algorithm developed on a constrained environment, a simple projection and camera motion model have to be chosen.

The manifold projection, originally introduced by Peleg [14], has been used as a camera motion model. In manifold projection, a thin stripe is taken from the centre of each frame to be used in the construction of the mosaic image. If there is no significant change of scale or motion parallax, the frames can be registered accurately by a rigid camera motion model. Manifold projection is quick to process since frames do not have to be projected onto a different surface and it offers excellent image quality in the resulting mosaic.

4.3. Frame registration stage

Frame registration or image registration refers to the transformation of two different sets of data containing some coincident information into a single reference coordinate system. In the particular case of a panorama stitching, frame registration has to be able to identify the overlapping parts of two frames and the transformations that should be applied to them to match the same view with the best accuracy possible. Figures 14 and 15 show two examples of image registration procedures.

Figure 14. Feature-based registration scheme.

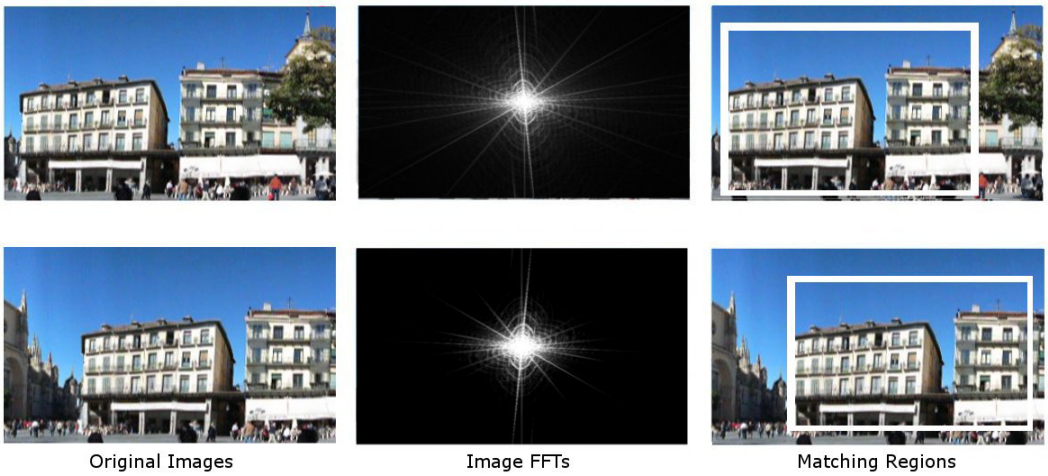


Figure 15. Phase correlation-based registration scheme.

Image registration methods can be divided into two separated categories. Feature-based approaches search for common feature points in a pair of images and then calculate parameters that make these points to match in the resulting image. They pay attention to small regions of the image and the shapes such as lines, curves, corners or boundaries that can be found on them. The definitions and properties of these two fundamentally different methods are explained well in the survey of Zitová and Flusser [15].

Depending on the chosen method, a transformation model has also to be defined based on the parameters that are obtained with the registration of each frame. The most limiting factor on a small footprint platform is defined by the computational complexity of the registration algorithms.

4.3.1. Frame registration sub-tasks

Although frame registration can be considered just a single task, in many cases, it can also be decomposed into several single smaller sub-tasks that can be performed simultaneously or with a cascade approach. Given two frames with a different point of view of an overlapping area, a transformation can be defined with up to six parameters. The point of view can be shifted in three axes (vertical, horizontal and scale) and also rotated around them. Some registration methods assume some of these parameters to be zero, thus simplifying the computation of image registration.

The main sub-tasks that can be identified on the image registration are shift estimation, planar rotation estimation, change of scale estimation and warping estimation. The shift estimation task should be able to measure the camera movement while kept perpendicular to the scene. Rotation estimation can measure the rotation around the axis that is perpendicular to the scene. Change of scale and warping estimation solutions must also be capable of measuring any other movement or tilt of the camera. Some other smaller sub-tasks can be considered an image pre-processing stage. A pre-processing stage composed by window filtering and image transformations like grayscale conversion might be needed to improve the performance.

4.3.2. Approaches to frame registration

The reference algorithm selected for this work relies on the recent image registration method of Vandewalle [16], a direct method which has many advantages. It is very robust against blur and image intensity changes, and it can handle rotated images. The method is also computationally fast. On the other side, it does not offer the possibility of detecting or correcting any change of scale or perspective. Thus it can provide for only three parameters.

This method presents the following characteristics:

- **Fast to compute** – This is probably the most desirable feature for a method that has to be used on a mobile device. Developing a real-time panorama builder requires a method capable of offering good accuracy while allowing the possibility of computing several frames per second on a constrained environment. Vandewalle's method relies on the frequency computation via Fast Fourier Transforms (one and two dimensional).

- **Robust against blur** – The quality of the individual frames captured from a video sequence with a handheld device is relatively low compared with still frames from a regular digital camera. Blur, and especially motion blur, is present in some of the frames. Although it is possible to minimize this effect with the selection of the best frames, the method used for the image registration has to be robust against blurry frames. Methods based on frequency and phase analysis, as Vandewalle's method, present excellent results against blur.
- **Can handle rotated images** – Taking handheld sequences always result in some tilting and undesired small rotations between frames. This feature offers us the possibility of achieving a better user experience.
- **Robust against image intensity changes** – Handheld devices such mobile phones present a very limited access to the camera settings. Using a sequence as the entry point for a panorama can result in different luminance and exposure time between frames. This can be corrected with frame selection and color balance. However, the registration method has to be robust against image intensity changes.
- **Highly optimizable** – The used method relies on well known operations such as FFT. This method offers the possibility of optimizations and fast routines that result in a fast performance. It has also been experimentally found that the method presents an excellent numerical precision performance, enabling the use of very short numerical word lengths.

The shortcoming of the method is that it estimates only the shift and rotation of each frame against its neighbours. This is often acceptable with panoramas, but the construction of mosaics becomes almost impossible without a model that can compute and correct the changes in the perspective and parallax.

A gradient feature machine has been implemented [17] to obtain a more flexible registration method. The implementation runs in two stages. The feature extraction stage identifies some points of interest on both of the images to be registered. The feature matching stage tries to match these points on both of the images using best linear unbiased estimation. Although the feature machine algorithm is not as robust against blur and image intensity changes, it provides more flexibility and is also able to compute changes of scale or perspective in addition to shifts and rotation.

The gradient feature machine method presents the following characteristics:

- **Fast to compute** – Like Vandewalle's method [16], a gradient feature machine can perform in a very fast manner when the regions to be matched are small and the number of features extracted is reduced.
- **Can handle changes of scale images** – Taking handheld sequences always results in some small changes of scale between frames. Although it is not always the most important factor to achieve quality panoramas, the computation of this factor allows the frame selection subsystem to discard images with excessive scaling.
- **Can handle changes of perspective**– Depending on the selected motion model, the computation of the change of perspective between two images, offers the frame correction subsystem the possibility of achieving a better final frame before being blended.

- **Highly optimizable** – The feature-based method relies on the extraction of several points of interest. When capturing images to compose a panorama, the small distance between video frames offers the possibility to carefully adapt the number and size of the points to the desired image size and characteristics. It has been experimentally found that the method presents excellent accuracy with a small number of low-complexity features extracted.

Some other registration methods have also been studied. Zitová [15] surveys and analyses several area based registration methods such as the one published by Lucchesse [18], which is able to compute full affine transformations from operations on the frequency domain. In feature based methodology, several other feature types have been surveyed. Lowe [19] proposed the extraction of Scalar Invariant Feature Transforms or SIFT descriptors that can be used for image registration. Speeded-Up Robust Features or SURF descriptors, published by Bay [20] are inspired by SIFT descriptors. They several times faster and provide for a good alternative to gradient features. Although these methods provide for an apparently attractive basis, the computational cost of them results in several major problems while facing the actual real-time implementation on mobile platforms.

4.4. Frame stitching stage

The frame stitching stage refers to the process that merges a group of frames together, producing a seamless single image. The frames must be processed in order to create the mosaic. Then the final image can be used for print, as a motion picture or screen display.

Blending a set of frames usually requires several sub-tasks that allow global and local correction of each individual frame and methods to eliminate the presence of perceivable differences in the seam area. For a given pixel location and based on one or more pixel values and their neighbours, the stitching function must give the final value of the pixel in the resulting image. Smooth results with no steps, stripes or steep curves between frames are always preferred when composing high quality panoramas. Figure 16 shows an example of Gaussian mask blending.

Figure 16. Frame stitching example.

4.4.1. Frame stitching tasks

Panorama stitching usually assumes that the sequence of images has a close illumination condition, but when the luminance changes from frame to frame, one problem becomes apparent. If the neighbouring images are stitched with notable illumination difference, the final panorama will change abrupt in illumination when we browse it. This effect gives us an unnatural impression. To avoid these artefacts on the final image, a color balance method should be performed.

The merging of two images is most critical in the area where the images meet. This seam area can be processed in many ways, but some local luminance correction or blending must be implemented. The complex seam handling is most critical when the registration step produces erroneous results.

4.4.2. Approaches to frame stitching

In the proposed solution, color balancing is made in a manner used before by Xiao [21]. This blending method adopted a simple statistical analysis to implement the alignment by imposing one image's illumination characteristic to the next one. The mean and the standard deviation of the luminance are computed. The next frame is corrected using the minimization of square errors.

The paper of Zomet [22] contains a good comparison of image blending methods and also proposed a new approach for optimizing the stitching result by a gradient based cost function. Szeliski proposed a simple local blending method to eliminate seam artifacts [23]. However, only straightforward methods like a Gaussian weighting mask [2] or a linear weighting mask are computationally affordable for a real-time approach on current mobile platforms.

4.5. Frame selection subsystem

To improve the quality of a panorama stitching system based on video sequences, another subsystem can be incorporated to the solution. A frame selection subsystem should be able to measure the quality of each individual frame, offering the possibility of discarding the worst ones in the panorama construction process.

The frame selection process is a matter of weighting the importance of different frame features. An important criterion is the presence of moving objects, since the most severe problems are created in the blending process by objects that get clipped. Another high importance factor is the amount of blur on a frame. The frame selection subsystem also gives the chance of automatically discarding several frames that might have a possible error in the image registration phase. Figure 17 shows an example of the selection process.

Figure 17. Frame selection scheme.

4.5.1. Frame selection tasks

To be able to select the best frames from a video sequence, several sub-tasks have to be performed. An image stitching algorithm needs to employ of motion detection to avoid the distortion of moving objects [24]. A motion detection algorithm analyzes the frames by looking for regions that contain moving objects and provide for motion values that can be used as a measure for quality. Figure 18 shows the impact of motion detection in panorama construction. Figure 18 a) is a part of a panorama created by our algorithm with motion detection enabled. Figure 18 b) is also from our algorithm, but motion detection is disabled. Figure 18 c) is created by Autostitch [25].

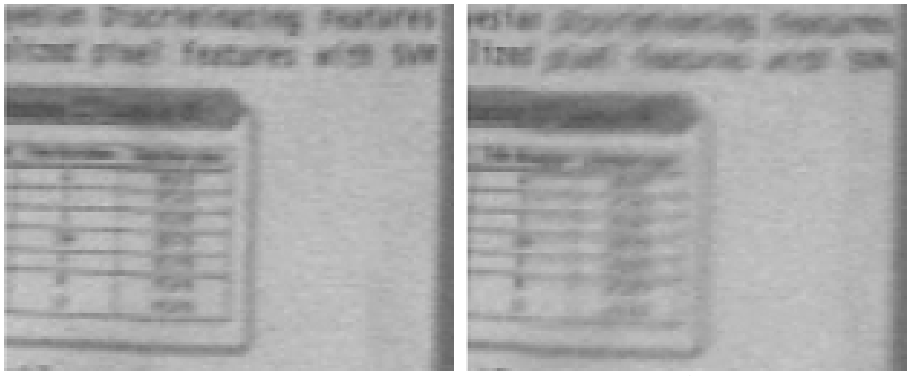
(a) (b) (c)

Figure 18. Effect of motion detection in panorama construction.

A blur detection subsystem also needs to be developed to be able to judge the global quality of a frame. Blurry frames, even if registered correctly, lead to the loss of some sharpness in the construction of the final image and an overall worse panorama. Blur has to be computed and each frame must score a value that contributes to the final quality measurements.

Other factors such as errors in other stages or sudden luminance changes between two consecutive frames may affect the evaluation of the quality of a frame. If a frame with a moving object is not discarded, the region selection must avoid cropping the

object in the final panorama. Finally, the frame discarding stage must put together each already evaluated factor that affects the quality of individual frames and discards the ones that are not suitable to construct the final image. Figure 19 shows the effects of blur detection. Figure 19 a) is a detail from a panorama constructed with blur detection enabled. Figure 19 b) is constructed without blur detection.



(a) (b)

Figure 19. Effect of blur detection: a) with blur detection, b) without blur detection.

4.5.2. Approaches to frame selection

The frame selection concept applied to panorama stitching was first introduced by Boutellier and Silvén [1] who explain how to select only the best group of frames in a video sequence to capture a panorama. With this approach, the image quality improves notably. Previous to that work, Hsu [26] has considered the criteria of a suitable overlap to select the best subset of images for panorama construction. Li [27] has also considered this selection based on the amount of distortion caused by rotation and perspective.

Each sub-task of the frame selection subsystem has been widely analyzed across the time. Motion blur can be detected in several ways. One simple way relies on obtaining a value as a side-product [16] of Vandewalle's method used in registration. It requires the calculation of the amplitude spectrum for each image to be registered. The spectrum is now also used for blur estimation by calculating the amount of high-frequency components in it.

Vandewalle states that the frequencies above a certain limit ρ_{max} need to be discarded in the registration process, since those frequencies contain alias if the image is blurred. The darkened area in Figure 2 depicts the frequency area from ρ_{max} to the maximum frequency, which we use to estimate the amount of blur. If the sum of the frequencies in the area is small, it means that sharp image details are absent due to blurring. A large sum tells us that the frame is free of blur. Figure 20 represents an amplitude spectrum of a fictional image. The darkened area represents the high-frequency area that is used to estimate the image blur.

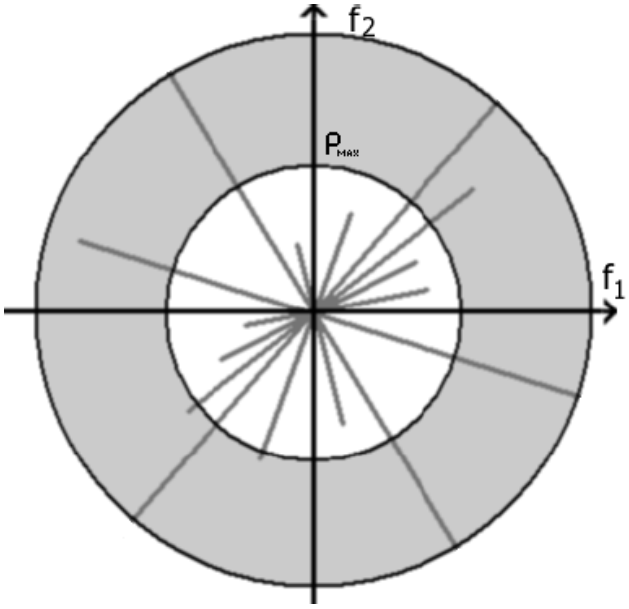


Figure 20. An amplitude spectrum of a fictional image.

This method produces one single number that expresses the amount of high-frequency detail in the image. This value can then be used comparatively between the overlapping parts of two consecutive frames. If an image I_a scores a higher result than I_b, it usually means that I_a is sharper than image I_b, but in some occasions the difference in the image content can distort the result.

Other methods used for blur detection have also been considered. The lack of resources existing on the target device suggests the use of straightforward methods. According to Liang [28], the amount of blur in a frame can be obtained by a summed derivatives computation. This method consists of summing together the derivatives of each row and each column to obtain a single number that expresses the amount of blur contained in a picture.

To detect moving objects across the scene, a very simple procedure can be used in order to make the process fast. In this method, the difference between the current frame and the previous one is calculated. The result is a two-dimensional motion image that covers the overlapping area of the two frames. After that, this motion image is low-pass filtered to remove noise and thresholded against a fixed value to produce a binary motion map.

If the binary image contains a sufficient amount of pixels that can be classified as motion, the dimensions of the moving object are determined statistically. The centre point of the object is approximated by computing the average coordinates of all moving pixels, and the standard deviation of coordinates is used to approximate the dimensions of the object. However, this method can produce a wrong result if there are multiple moving objects in the same frame.

Motion detection can be performed using sophisticated methods that are computationally costly. A good survey on this problem and its solutions is covered by Radke [29]. In the method created by Zhu [30], the moving objects were extracted by differencing three successive frames and defined further by calculating the active contour of each object. Davis [24] solved the problem by drawing the seams around the moving objects. However, the computational restrictions of the target platform suggest that only straightforward methods are convenient for a real-time application.

4.6. Frame correction subsystem

A frame correction subsystem can also be added to the panorama construction solution in order to improve the quality of the resulting images. A frame correction subsystem should be able to transform the input frames of the panorama system according to certain registration parameters and the problems shown on the frame selection and quality assessment stage. Figure 21 shows an example of luminance, warp and perspective correction.

Figure 21. Frame correction examples.

4.6.1. Frame correction tasks

To be able to correct the input frames on a video sequence, several sub-tasks have to be performed. A luminance correction algorithm analyzes the illumination difference between two matching regions and corrects one of them to match the other. Too dark or too bright frames used as input, even if registered correctly lead to worse overall resulting panoramas.

A warping and perspective correcting algorithm should be able to perform a full affine transformation between two matching regions, including rotations and changes of scale. The algorithm must interpolate the values of the corresponding pixels in the final image to avoid the loss of any detail. The correction of the perspective on the input images can be a crucial stage when the difference between subsequent frames is substantial.

4.6.2. Approaches to frame correction

In the proposed solution several interpolation methods have been tested. Although the nearest neighbour interpolation is computationally the most efficient method, it has been discarded due to loss of image quality.

Among the most typical interpolation methods, bilinear interpolation is preferred to bicubic interpolation due to its computational simplicity. For the luminance correction, the method followed just adds the intensity average difference to the frame that wants to be corrected.

5. PANORAMA BUILDER ALGORITHMS AND IMPLEMENTATIONS

5.1. Overview of the algorithm

In Chapter 4, we defined the panorama construction process using a multistage approach. A panorama construction implementation needs to compute several operations in each one of these stages. A global overview of the panorama construction algorithm is given in Algorithm 1.

(1) Transform the input ARGB frame into grayscale images.
(2) Multiply the current frame by a Tukey window to make it circularly symmetric.
(3) Compute the Fourier transform of the current frame.
(4) Compute the rotation estimation.
    a) Compute the Polar coordinates of the current frame.
    b) For every angle a, compute the average value of the Fourier coefficients.
    c) Find the maximum of the correlation between the current frame and the previous one between 15 and -15 dg. This is the estimated rotation angle.
(5) Compute the vertical and horizontal shift estimation.
    a) Compute the phase difference between the current frame and the previous one.
    b) The planar shift corresponds to the maximum of the shift matrix.
(6) Error correction:
    a) Check if the shift or the rotation estimation surpasses a threshold comparing it with the previous one.
    b) Use previous motion vector if an error is found.
(7) Detect moving object with simple difference.
    a) Compute simple difference of pixels.
    b) Mark regions with movement.
(8) Estimate the blur of an image by the computation of summed derivatives.
(9) Compute the change of luminance between matching regions calculating the difference of intensity averages.
(10) Calculate quality score.
    a) Use moving objects, luminance changes, blur, and rotation values to compose the quality value.
(11) Select best frame on the stitching range based on quality score.
(12) Correct selected frame.
    a) Correct luminance.
    b) Rotate frame using bilinear interpolation.
    c) Select blending region based on moving objects.
(13) Calculate image global coordinates.
(14) Stitch image using local blending techniques.
    a) Calculate the seam location avoiding crossing moving regions.
    b) Blend using a Gaussian or linear mask.

Algorithm 1. An overview of the panorama construction algorithm

5.2. Projection and camera motion model

Traditionally, the camera movement on a panorama capturing process is assumed to be a pure rotation around the optical axis. However, for a handheld panorama, this assumption is not accurate. The final version of the panorama application uses the flexible camera motion model provided by the manifold projection [14]. In a

manifold projection, a thin strip is taken from the centre of each frame to be used in the construction of the panorama image. The camera motion speed is considered to be close to a linear one. Figure 22 shows an example of a typical camera motion during the panorama capture and the manifold projection used by the application.

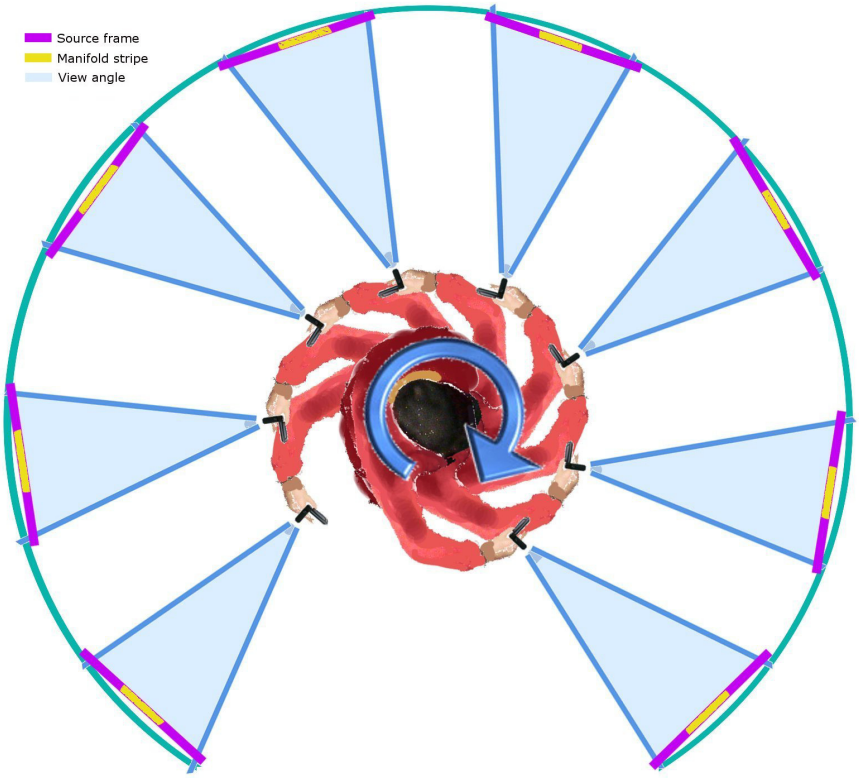


Figure 22. Example of the camera motion in a typical handheld panorama.

5.3. Image pre-processing

To conform to the manifold projection model and to be able to register each frame against the previous one, several operations must be done. First, the frames should be cropped to save just the central part of them, where the image quality is better. Figure 10 shows how the original frames are cropped to ensure a better quality in stitching and to save memory use. Our algorithm crops frames immediately after capturing them. There is a lot of data redundancy between consecutive frames and thus cropping saves both memory space and computational resources in image registration. The cropped frame has roughly one third of the original frame's width and 90% of its height.

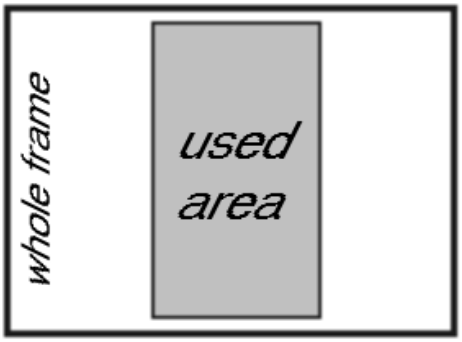


Figure 23. Image cropping example.

Most of the existing registration algorithms use grayscale images to compute the transformation parameters between two pictures. Therefore, the central part of the source images must be converted into grayscale to be used in the actual registration process. A downscale to the processing square resolution can be performed to achieve a smaller size that results in fewer operations.

Several versions of the RGB to grayscale converting function have been implemented. The size of the cropped and the coefficients for the grayscale conversion can be arbitrarily chosen, as well as the desired downscaling factor. In the more straightforward case, the luminance Y is defined as follows:

$$Y = \frac{R + G + B}{3}, \tag{1}$$

To be able to use a smaller mask without changing our source images size, a downscaling function with a factor of N has been coded. The original frames are kept the same size as the camera gives, but the grayscale conversion is done using the average value of NxN pixels. This average filter acts as a low-pass filter and cuts off parts of the high frequency values, decreasing the effects of possible aliasing. The resulting mask that is going to be used for image registration is now NxN times smaller. The number of operations needed to perform a registration with a smaller mask is much less, resulting in a significant decrease of the processing time per frame.

Window filtering

To provide for robustness during the registration process, a Tukey window function has been implemented to filter the gray-scale images. A Tukey window is a tapered cosine window defined as follows:

$$w(x) = \begin{cases} \frac{1}{2}\left(1+\cos\left(\frac{2\pi}{\alpha}\left[x-1+\alpha/2\right]\right)\right) & 0 \le x < \alpha/2 \\ 1 & \alpha/2 \le x < 1-\alpha/2 \\ \frac{1}{2}\left(1+\cos\left(\frac{2\pi}{\alpha}\left[x-1+\alpha/2\right]\right)\right) & 1-\alpha/2 < x \le 1 \end{cases} \tag{2}$$

where α denotes the ratio of tapered section to constant section with $0 \le \alpha \le 1$.

A windowing function applied to each frame guarantees its circular symmetry, providing for a better performance of the phase correlation base registration method. Figure 24 presents a Tukey window filtering scheme.

Figure 24. Tukey window filtering scheme.

The actual implementation of the Tukey window filter consists of a lookup-table constructed with fixed value coefficients for the desired size. This table can be pre-calculated on a desktop computer or just on the starting routine of the library, whenever the mask size or resolution is changed. To filter each image, a table look-up is performed and the value needed to multiply each pixel is selected.

5.4. Image registration

Vandewalle's method [16] uses a frequency domain algorithm to estimate the motion parameters between the reference image and each of the other images. In this method, only planar motion parallel to the image plane is computed. The motion can be described as a function of three parameters: horizontal and vertical shifts, $\Delta x1$ and $\Delta x2$, and a planar rotation angle ϕ. A frequency domain approach allows us to estimate the horizontal and vertical shift and the (planar) rotation separately. Assume we have a reference signal $f_1(\boldsymbol{x})$ and its shifted and rotated version $f_2(\boldsymbol{x})$:

$$f_2(x) = f_1(R(x+\Delta x)),$$
$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad \Delta x = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix}, \qquad R = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \tag{3}$$

This can be expressed in the Fourier domain as

$$
\begin{aligned}
F_2(u) &= \int\int_x f_2(x) e^{-2\pi u^{\mathrm{T}} x} dx \\
&= \int\int_x f_1(R(x+\Delta x)) e^{-2\pi u^T x} dx \\
&= e^{-2\pi u^T \Delta x} \int\int_{x'} f_1(Rx') e^{-2\pi u^T x'} dx'
\end{aligned}
\tag{4}
$$

with $F_2(\boldsymbol{u})$ the Fourier transform of $f_2(\boldsymbol{x})$ and the coordinate transformation $\boldsymbol{x}' = \boldsymbol{x} + \Delta\boldsymbol{x}$. After another transformation $\boldsymbol{x}' = \boldsymbol{R}\boldsymbol{x}'$, the relation between the amplitudes of the Fourier transforms can be computed as

$$
\begin{aligned}
\left|F_2(u)\right| &= \left| e^{-2\pi u^T \Delta x} \int\int_{x'} f_1(Rx') e^{-2\pi u^T x'} dx' \right| \\
&= \left| \int\int_{x'} f_1(Rx') e^{-2\pi u^T x'} dx' \right| \\
&= \left| \int\int_{x'} f_1(x'') e^{-2\pi u^T (R^{\mathrm{T}} x'')} dx'' \right| \\
&= \left| \int\int_{x'} f_1(x'') e^{-2\pi u^T (Ru)^{\mathrm{T}} x''} dx'' \right| \\
&= \left| F_1(Ru) \right|
\end{aligned}
\tag{5}
$$

where $|F_2(\boldsymbol{u})|$ is a rotated version of $|F_1(\boldsymbol{u})|$ over the same angle φ as the spatial domain rotation, $|F_1(\boldsymbol{u})|$ and $|F_2(\boldsymbol{u})|$ do not depend on the shift values $\Delta\boldsymbol{x}$, because the spatial domain shifts only affect the phase values of the Fourier transforms. Therefore we can first estimate the rotation angle φ from the amplitudes of the Fourier transforms $|F_1(\boldsymbol{u})|$ and $|F_2(\boldsymbol{u})|$. After compensation for the rotation, the shift $\Delta\boldsymbol{x}$ can be computed from the phase difference between $F_1(\boldsymbol{u})$ and $F_2(\boldsymbol{u})$.

Fast Fourier transform

An FFT computes the Discrete Fourier Transform and produces exactly the same result as evaluating the DFT definition directly with the only difference rooting on the faster processing time of an FFT. Let x_0, ..., x_{N-1} be complex numbers.

Evaluating the DFT definition directly requires $O(N^2)$ operations: there are N outputs X_k, and each output requires a sum of N terms. An FFT is any method to compute the same results in O(N log N) operations.

To perform the image registration in fixed point, the general purpose FFT project developed Mark Borgerding has been used [31]. The code consists on a Fast Fourier Transform implementation based on the “Keep it simple” principle. The FFT project is reasonably efficient mixed radix FFT library that can use either fixed or floating point data types and that has a very small footprint.

Several modifications have been made to the routine in order to port it to efficient Symbian C++ code. The final performance of this operation in fixed point can be up to 25 times faster than the same algorithm in floating-point for a mobile phone without a dedicated chip. Table 1 shows the computation times of the FFT function with several profiles.

Table 1. FFT computation times on a Nokia N93 device

Size	64 bit floating-point	32 bit floating point	32 bit fixed point
64x 64	3200 ms	950 ms	20 ms
128x128	3800 ms	1200 ms	50 ms
256x256	13500 ms	4500 ms	200 ms

5.4.1. Shift estimation

The shift of the image parallel to the image plane can be expressed in Fourier domain as a linear phase shift. The shift parameters $\Delta \boldsymbol{x}$ can thus be computed as the slope of the phase difference $\angle(F_2(\boldsymbol{u})/F_1(\boldsymbol{u}))$.

$$\begin{aligned} F_2(u) &= \int\int_x f_2(x) e^{-j2\pi u^T x} dx = \int\int_x f_1(R(x+\Delta x)) e^{-j2\pi u^T x} dx \\ &= e^{j2\pi u^T \Delta x} \int\int_{x'} f_1(x') e^{-j2\pi u^T x'} dx' = e^{j2\pi u^T \Delta x} F_1(u) \end{aligned} \tag{6}$$

To estimate the shift between two frames with the Vandewalle's method [16], the results of the frequency components of the two frames are divided element by element and this result is normalized, dividing again each element by its amplitude. After applying the inverse FFT function, the amplitude of the result is computed to obtain a registration matrix. The greatest element of that matrix will show the number of pixels that should be taken as the shift.

If a downscale factor is applied during the construction of the processing square, the resolution of the estimated shift will decrease on the same factor. To deal with this subject, the single-pixel resolution can be interpolated linearly with the values of the neighbours in the shift estimation matrix. The result of this interpolation is single real number which has to be rounded up and represents the actual estimated shift.

5.4.2. Rotation estimation

The rotation angle between the spectrum of two images can be computed as the angle for which the Fourier transform of each one is rotated. The computation of a spectrum rotation for every evaluation of the correlation is computationally heavy. When the images' spectrum is expressed in polar coordinates, the rotation over the angle is reduced to a circular shift over it.

The chosen reference algorithm uses a slightly different approach that is computationally much more efficient. Once the 2-dimensional FFT of the grayscale images is obtained and the frequency components are identified, a function that gets the radial sum of them is performed. The radial sum function adds the frequency

values corresponding to a specific angle ranged with the desired resolution. An array of summed values corresponding with the angles is obtained. The computation of the frequency content *h* as a function of the angle can be expressed as follows:

$$h(\alpha) = \int_{\alpha+\Delta\alpha/2}^{\alpha+\Delta\alpha/2} \int_{0}^{\infty} \left| F(r,\theta) \right| dr d\theta. \tag{7}$$

In order to get the value of the relative rotation with two frames, the two arrays containing their radial sums are compared. To look for a correlation, a 1-dimensional FFT is performed on each radial sum, and one of the results is multiplied element wise with the conjugate of the other. After performing an inverse FFT of the multiplied array, the maximum value is obtained, representing the rotation between both frames. Figure 25 shows on the left the spectrum of an image and its rotated version. The right side shows the correlation between the frequencies of the radial sums.

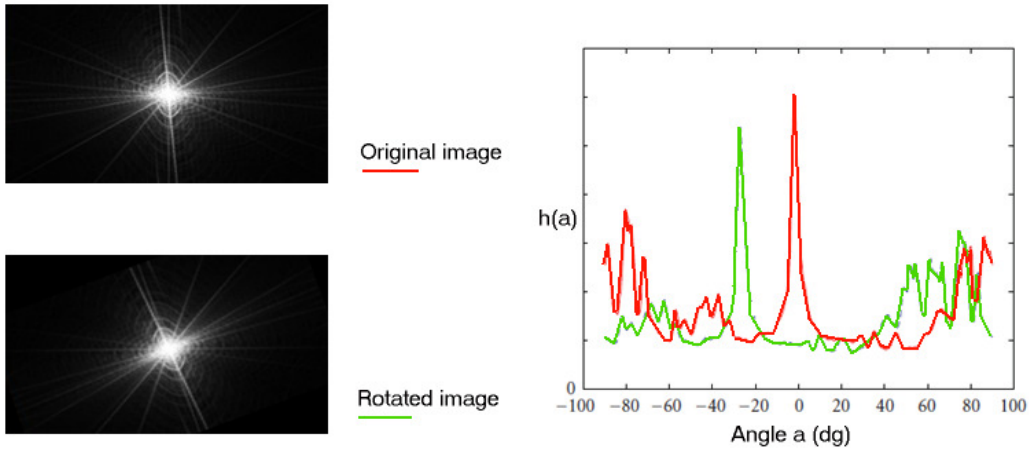


Figure 25. Rotation estimation example.

In the final application, two different profiles are used and rotation estimation is disabled by default. Although rotation estimation has been theoretically proven to enhance the final results of panorama stitching, the actual implementation of this presents some inconveniences. Estimating the rotation can be a quite costly task that decreases the global speed of the system. This can result in fewer frames obtained each second with less overlapping areas between frames and subsequently more registration errors.

5.4.3. Image registration comparison

A feature-based registration algorithm can provide more flexibility and is able to compute changes of scale or perspective in addition to shifts and rotation. However, feature-based methods are less robust against blur and intensity changes. Hannuksela et al. [17] implemented a gradient feature machine that is able to register frames in real time and runs on mobile devices. The code provided by this implementation has been adapted to the panorama building application. Table 2 shows the error rate for a similar processing time of the phase correlation method and the feature machine

method. The error rate is defined by the number of registrations detected as a failure case divided by the number of total registrations.

Table 2. Time and error rate comparison of registration methods.

Method	(64 x 64 pixels)	Time consumed	Error rate
Phase Correlation (FFT)	Shift	100 ms.	0.2%
	Shift + Rotation	150 ms.	0.5%
Feature Machine (Gradient Features)	Shift	100 ms.	0.3%
	Shift + Rotation + Scale	120 ms.	0.6%

When only the shift and rotation parameters are needed, the phase correlation method offers a lower error rate for a fixed computation time. However, if more parameters are needed in the model, a feature machine still offers a low error rate with a good speed performance.

5.5. Moving objects detection and blur estimation

Once the registration process is completed, the quality of each frame has to be determined. To compute the motion detection, a very simple procedure can be used in order to make the process fast. The difference between the current frame and the previous one is calculated. The result is a two-dimensional matrix that covers the overlapping area of the two frames. After that, this matrix is low-pass filtered to remove noise and thresholded against a fixed value to produce a binary motion map. The motion map is used as a quality measure for the frame selection stage and the blending region determination.

The computation of the blur as a side product of the registration method has been discussed in the previous chapter. However, the lack of resources existing on the target device suggests using a less ambitious method that can be performed faster.

The final version of the application acquires the amount of blur in the frame by a summed derivatives computation [28]. This method of blur calculation produces a single number that expresses the amount of high-frequency detail in the image. When taken from the same area of the image, this usually means that the frame with more high-frequency detail is sharper than the others. In some occasions, the difference between consecutive frames due to changes in the landscape, e.g., a moving object, can distort the result.

The main contributing factor for the selection of the best frame in our final implementation is the result of the blur detection algorithm, since the moving objects estimation is only activated in some of the profiles.

5.6. Frame selection and evaluation

A video frame selection sub-system can increase the quality of a panorama stitcher application. The frame selection process is a matter of weighting the importance of different frame quality features. The quality values are computed for a certain scope of frames at a time. The scope encompasses the frames that have a suitable translation with respect of the previous selected frame, as depicted in Figure 26.

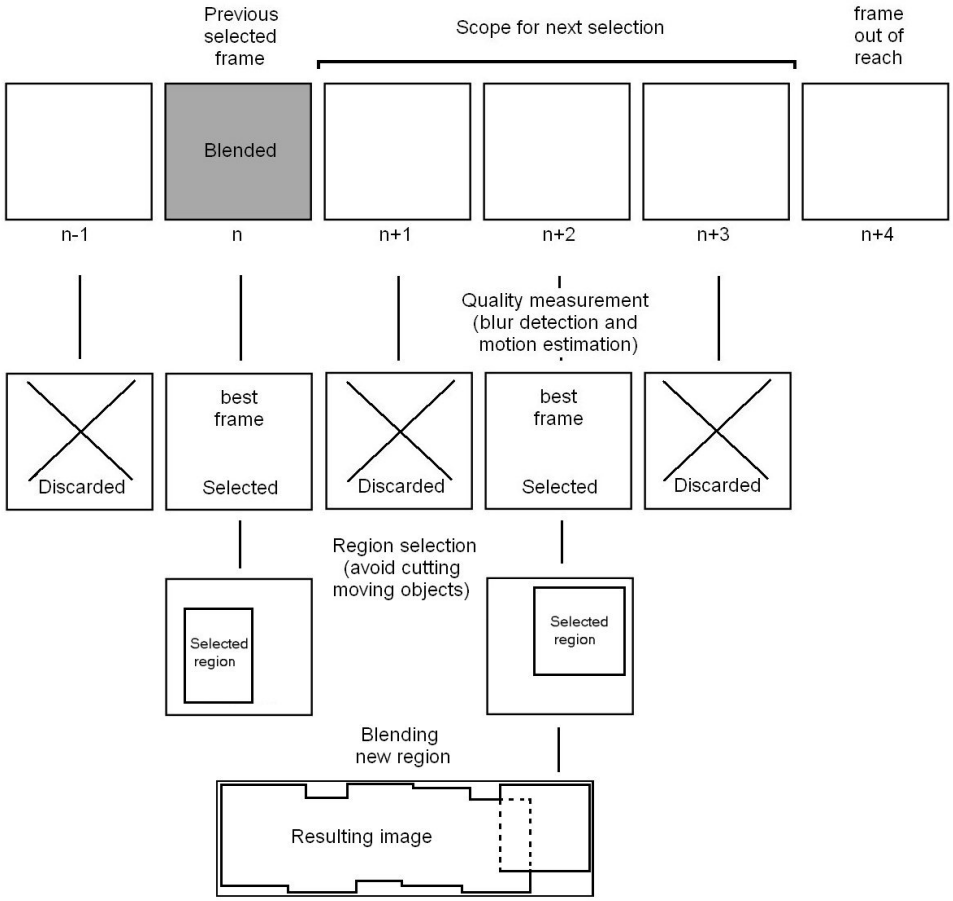


Figure 26. Frame selection scheme.

Each one of the quality evaluation tasks can be switched on and off depending on the desired speed performance of the final application. In practice, each frame gets a quality value that is calculated from the aforementioned factors and this can be implemented successfully in many different ways. According to experiments, the presence of moving objects is the most important criteria, since the most severe artifacts are created in the stitching process by moving objects that get clipped. However, the motion detection stage is not activated in every profile. The factor of second highest importance was chosen to be the amount of blur. Finally, frames that are highly rotated, that present high luminance differences with the previous one or that are likely to have registration errors are less preferred.

Blending region selection

The seam location used on the blending stage is determined by the size of the coincident regions of the two images that are to be blended. By default, the seam is located exactly in the middle point of the coincident region. However, in the presence of moving objects, the seam is drawn outside the boundaries of the objects. If no suitable seam can be drawn on the frame, it is discarded.

5.7. Image blending

The previously published image blending methods keep all the frames of a sequence on the memory and then proceed to stitch them. The memory constrictions of a mobile phone suggest that the best way of facing this task consists of blending each frame before capturing the next one. This way, only the resulting image is kept in memory and some post-capturing processing time can be saved. Figure 27 depicts a scheme of the blending algorithm data flow. The first row shows how only the selected regions of the best quality frames are blended. As new frames are selected, they are merged into the final image.

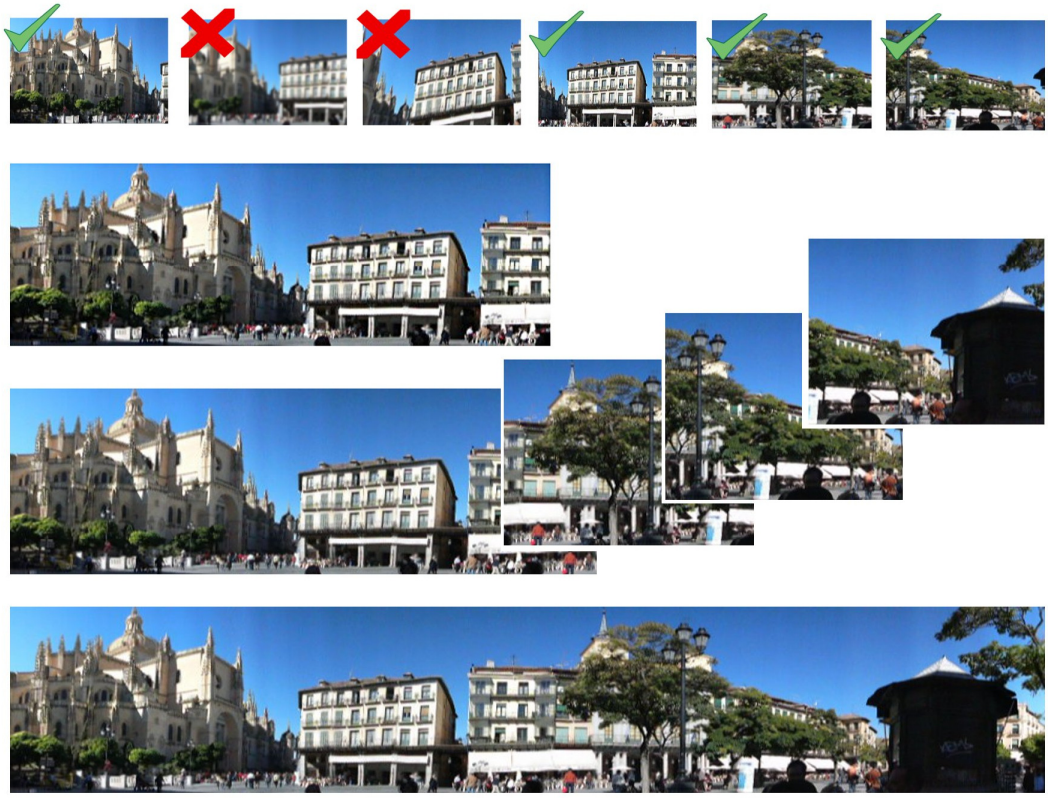


Figure 27. Sequence scheme of the blending algorithm after registration.

5.7.1. Global Luminance correction

Global luminance and exposure correction is needed because most mobile phones' video devices adjust automatically the camera's aperture and exposure time to prevent over- or underexposure of the frames. Although automatic adjustment makes it pleasant to watch a video on playback, it makes the creation of a panorama much more complicated.

The best results are obtained when the camera API allows keeping the exposure fixed. In this case, only local correction that works solely on individual pixels is needed. This correction is usually referred to pixel luminance correction and it is described in the next subsection. Figure 28 shows an example of a panorama construction after luminance correction.

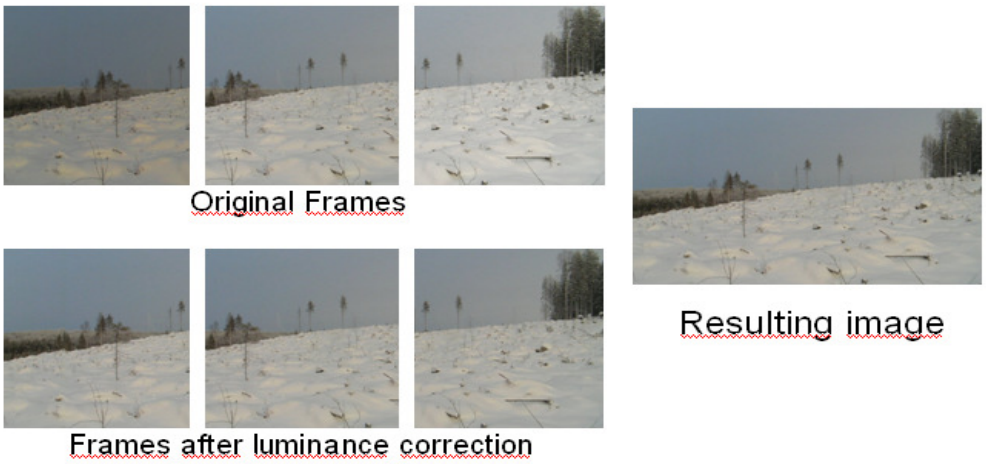


Figure 28. Luminance correction example.

Symbian S60 camera's API does not allow keeping an exposure value fixed. The frame selection and the low frame rate make the shifts between stitched frames large enough to consider that the exposure has changed. Global correction of luminance must be implemented. To perform this correction, the average luminance value is computed over the pixels belonging to the overlapping area. Every pixel of the current frame is corrected according to the value of the average on the previous one, adding or subtracting this difference to green, blue and red values of the frame to stitch. Figure 29 shows how the global luminance correction routine only uses the overlapping area.

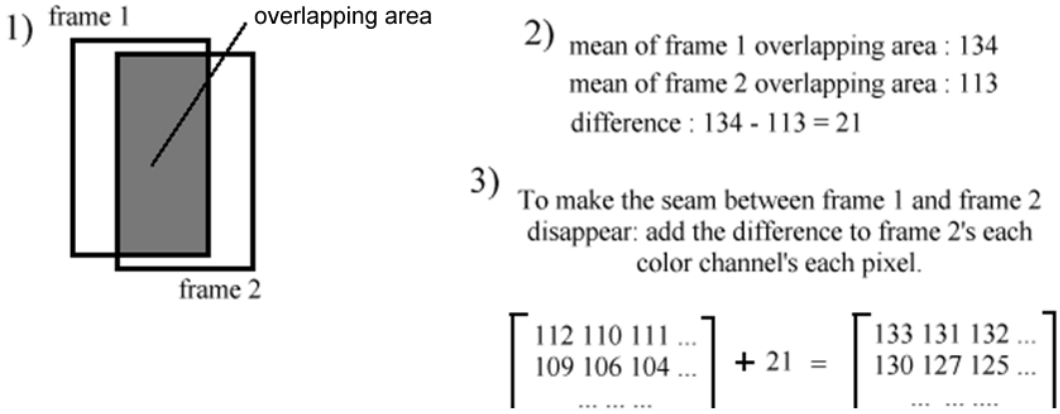


Figure 29. Global luminance correction.

5.7.2. Pixel Luminance correction

Instead of using the Gaussian weighting mask present on the reference algorithm, a less computationally costly method has been selected. A linear function that gradually merges one frame to the next, changing the weight assigned to each one, is used. This method prevents some vertical stripes from being visible between the individual frames that compose the panorama. A seamless image with smooth transitions between frames is then created.

The RGB image pixels of a N93 Symbian bitmap can be interpreted as 32 bit unsigned integers. The weighting function is implemented as a floating-point coefficient that multiplies each channel on each pixel, and computes the contribution of each one. Figure 30 depicts the linear contribution of pixels during the blending process. The weight of each pixel depends only on the distance to the seam point.

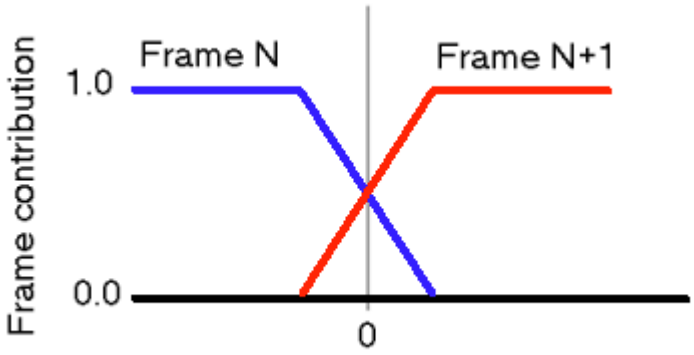


Figure 30. Linear contribution of pixels.

6. USER INTERFACE AND EXPERIMENTS

The panorama building solution analyses the frames in the video for motion and moving objects, quantifies the quality of each frame, and stitches up to 360 degree panoramas from the best available images [32, 33]. The high interactivity of video sequences and mobile devices makes the design of a user interface particularly challenging.

The proposed solution follows an intuitive process. In order to get a final panorama image, the user focuses the camera to the desired starting point of the mosaic. The camera starts turning around up to 360 degrees and a sequence of images starts to be captured. Each image is then individually processed to estimate the shift and rotation. The blurriness of each picture is measured and moving objects are identified. Based on the quality of each individual frame, a selection process takes place. Each frame is either accepted or discarded. For every selected frame, if a moving object is present and it fits the image, the image is blended drawing a seam that is outside the boundaries of the object. If only a partial object is present, the part of the frame without the object is the one blended. Figure 31 represents the image capturing process of the panorama application.

Figure 31 Panorama capturing process.

6.1. PanoramaX Application

The basis for the mobile face detection implementations created for this work is the *PanoramaX* application. The application was made for using the actual panorama stitching modules, which were implemented as Symbian dynamic link libraries (DLLs). The application also provides a user interface that is capable of saving the images, starting and stopping the capturing or saving the resulting images.

PanoramaX calls the stitching algorithms always through a fixed interface created for this purpose. The same application can be utilized in running all the algorithms that were created in this project, as well as any other application that processes video frames from the camera. The application works by receiving camera frames captured by the phone, and before drawing them on the display, the shift against the previous frame is computed. Figure 32 shows the application flow diagram of the panorama builder.

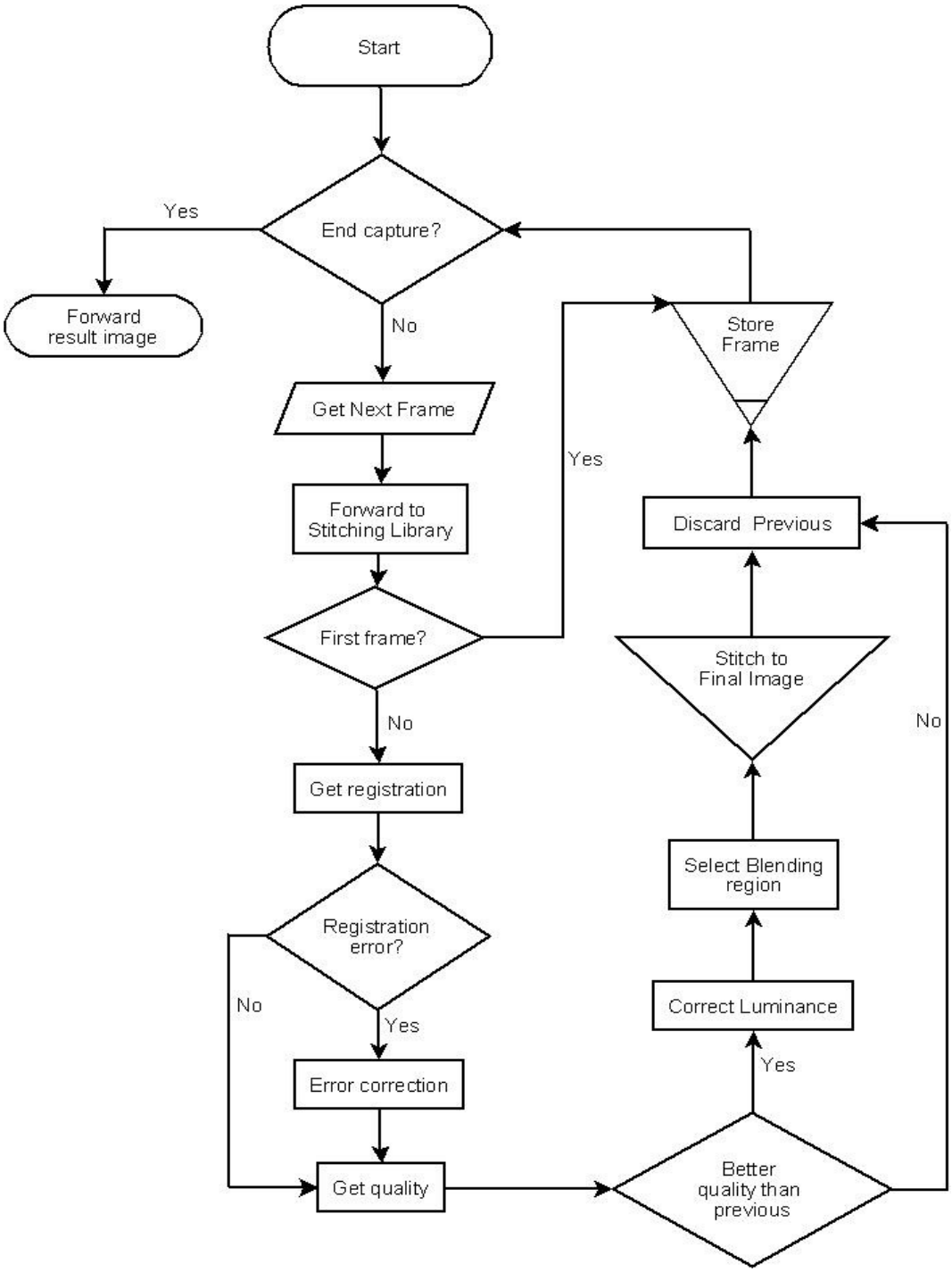


Figure 32. Application flow.

6.1.1. User interface

The user interface has been kept as simple as possible and modified based on user feedback. When the application is started, the screen shows a viewfinder with the images straight forwarded from the camera. Figure 33 shows the user interface during the capture. In order to take a picture, the user should follow the next sequence:

- Once the desired starting point for the panorama image has been chosen and can be seen on the screen, the user has to press the capture button.
- The device has to be slowly and softly moved from left to right in order to capture and process the frames
- When the desired ending point for the panorama has been reached, the user has to press the capture button again.
- The whole image and a detail of it are shown on the screen. Browsing through the image is possible using the device's joystick.
- If the capture button is pressed again, the image is saved on the desired format, with a sequenced name. The cancel button will discard the picture and exit the application.
- An information note is displayed once the image is saved. The viewfinder starts again, and another capture can be made.

Figure 33. User interface during image capture.

Some additional functions can be accessed with key presses or by menus, such as selecting the image saving format (.PNG or .JPG), displaying the version information, or changing the size of the viewfinder. Figure 31 shows the user interface on the image capturing phase.

Figure 34. Using interface during image saving phase.

6.1.2. Frame capturing interface

The construction of panoramic images requires a set of single frames that have to be captured. To reduce memory needs and improve the user experience, most of the computations can be done in the time between the capture of the frames.

To construct an application that can work in real time, the frame capturing interface has to offer frames at a sufficient frame rate, providing also the possibility of processing each one of them before the capture of the next. To achieve this, a camera application with viewfinder has been developed. The interface sends a call-back once the camera is ready to capture a new video frame. After the capture and when the main thread is idle, this frame is sent to the processing routines. Once the process is done, the handle is given back to the frame capturing interface that captures and sends another frame. Figure 35 represents the processing diagram and the application data flow.

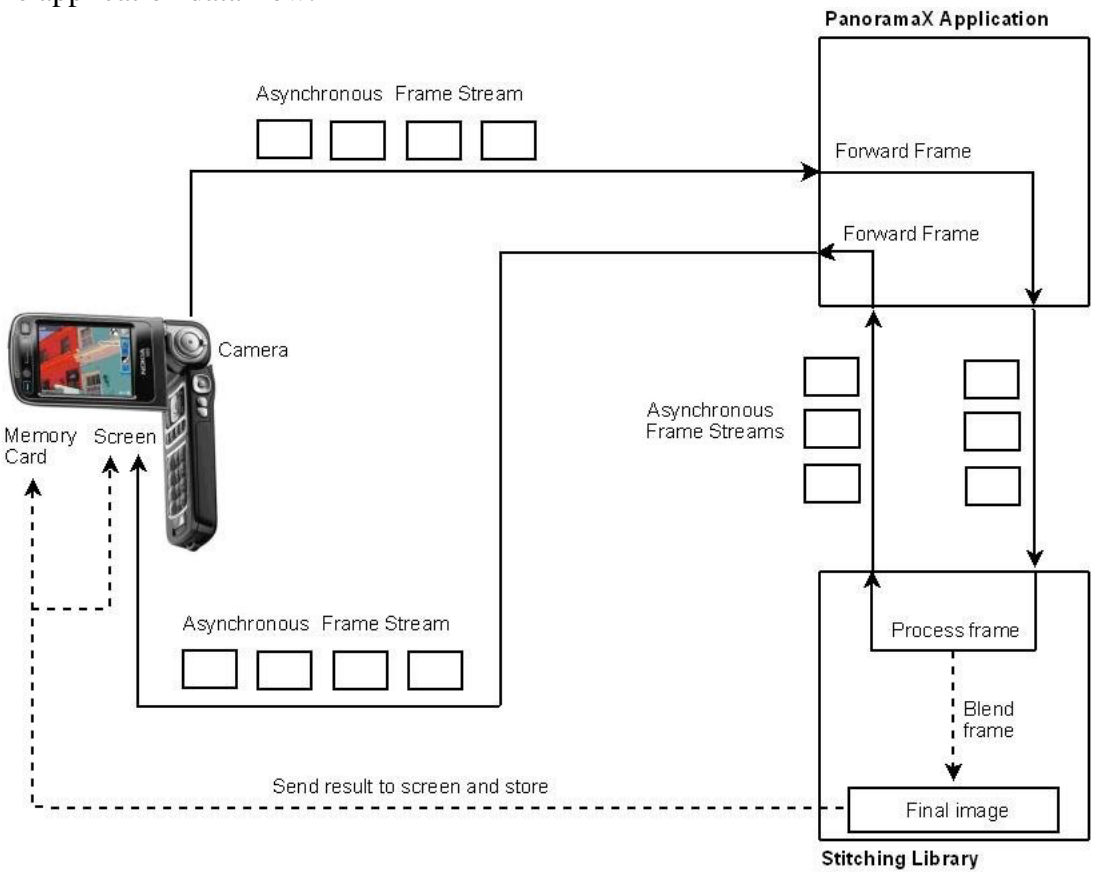


Figure 35. Processing diagram and data flow.

This capturing interface is a good solution for a real-time application because it is able to offer frames asynchronously, depending on the needed processing time. The interface can adapt to a variable frame rate without any difference among frames and without any intervention from the user. Several different applications and processing algorithms based on video analysis can be developed, using exactly the same interface such as a *Point & Find* application [34] or a handheld document scanner [35].

6.2. Stitching DLL

The stitching algorithm module has been implemented as a Symbian static Dynamic Linked Library. The implementation of a module as a DLL is usually the best solution especially when several applications can make use of it or when several versions of one algorithm are going to be developed.

Only one copy of a static DLL is loaded into memory and this same copy is shared between every application using it. However, it also addresses one restriction in the version of Symbian OS: the static DLLs should not have any writable static data. This includes both the static variables inside functions and the global variables that are defined outside any class, structure or function. This constraint is an obvious consequence of the fact that in the case of static DLLs the same binary code of the modules is shared between multiple users. Thus, all the binary code of Symbian static DLLs is read only.

The EKA2 kernel version of Symbian OS allows writable static data on a DLL, although in most situations, the use of this kind of data in Symbian C++ code is not justified. This is because as with old kernels, the DLLs might be used by several users. Each user of the DLL would need their own memory chunk for their global or static writable data. The use of static writable data is not recommended even when developing software for new Symbian OS versions. In the latest Symbian OS versions, this kind of data is supported mainly in order to facilitate software porting. For instance, in ANSI C code it is very typical to have global non-const variables as the “glue” between C function calls. Therefore, software originally developed for other environments may contain large amounts of static data, which can make porting this software for Symbian OS a challenging process.

6.3. Memory management

Symbian S60 applications offer by default a small chunk of memory heap available for applications. A minor tweak of the compiling and linking options has to be done in order to be able to store all the intermediate information needed for real-time performance. Even so, the reduced RAM capacity on mobile phones makes on-the-fly stitching procedures more complicated and calls for a carefully implemented memory management scheme. All the image processing should be done following a cascade structure with the help of some buffers to save previous data.

In the final version of the program, the first frame is captured and cropped and its frequency components and blur are calculated and stored to the memory. The next frames are acquired asynchronously. Just after each frame is acquired, the image blur is computed, frame registration is performed and a decision about the quality of the frame is then made. If the quality is better than that of the previous frame, based on blur detection, exposure correction and blending functions are called.

The result image is growing frame by frame each time the blending function is called. There is no need to store several frames in memory at all; only the last computed one and its features: its frequency components, its shift estimation and the total amount of shift in this moment, as well as the evolving final image.

When the user stops capturing the frame sequence that will become the mosaic, current resulting image is stored on a mass storage device, memory objects are deleted and application is reset. The application can capture several images in a short period of time with no need of a restart.

6.4. Error correction

Repeating textures, images excessively blurry or bad illumination can lead to catastrophic registration errors, as a single small error on a single frame can accumulate through the panorama stitching. To cope with these errors, a simple error-correcting method has been implemented: each time a frame is registered, its planar motion vector is compared to the planar motion vector of the previous frame. If the difference between the adjacent motion vectors surpasses a certain threshold, the motion vector is replaced by the previous motion vector. This works very well in panorama imaging from a video, where the usual motion trajectory of the camera is close to a linear one.

6.5. Adjusting parameters

When developing an application like a panorama stitcher under limited resources, several constraints have to be taken into account. After coding every algorithm, a fine tuning of the parameters, based on the experimentation and the observation of the specific characteristics of the device, is needed.

Description of parameters

The most important parameters that can be changed on the registration and evaluation of the frames are:

- **Video frame size** – This is the most important characteristic in the application. It affects the final resolution and size of the resulting images. It is strongly limited by the interface that the mobile phone provides. A frame size of 320x240 pixels (QVGA) was selected as the default configuration, but some other sizes can be selected as well.
- **Coefficients for grayscale conversion** – All the image processing is done with grayscale images. To perform an RGB to Y conversion, several weights can be assigned of each one of the RGB channels. Several options have been tested, deciding finally to use a uniform approach with equal contributions. If the specifics of the camera sensor are not known, this option is the safest one.
- **Mask size for the processing square** – This is the parameter that has the most important effect on the global performance. It affects the number of mathematical operations that we should perform on each frame to obtain a registration. It has been experimentally found that with the quality of the raw frames obtained by the camera. One sixth of the original frame size is a lower bound for a correct registration process. With a fixed frame size of 320x240 frames, a square mask of 128x128 pixels is the default choice. However, it is possible to lower it by downscaling and interpolating the result.
- **Downscaling and interpolation factor** – If we downscale and filter the processing square, we can reduce significantly the size of it. An interpolation of the registration results will give back the lost precision. A factor of two on each dimension has been found to offer excellent result. Other values tried were no downscaling and a factor of three downscaling where the interpolation of results did not offer exact values.

- **Maximum number of frames** – Although there is not any theoretical limit for the size of the images, the RAM memory of a handheld device is a scarce commodity: with every tested device, 300 frames offer the possibility for a 720 degrees panorama.
- **Frame selection buffer length** – When selecting frames, we should keep in the memory a set of them to be able to discard the worst ones. Several sizes have been tried, but due to computational reasons, a small buffer of only 2 frames is used in the final version of the panorama stitcher.
- **Out of reach indicator threshold** – While selecting frames, it is necessary to establish a maximum shift in pixels to search for the best one. The size is related to the processing square size. The experiments showed that a good indicator threshold is half of the size of the processing square.
- **Error correction threshold** – This threshold is used to detect when a motion vector is unusual when compared to the previous one, and should be corrected. Its value depends on the frame rate. A distance of 30 pixels on the motion vector has been chosen to score as an error.
- **Coefficients for frame quality score** – To measure the quality of a frame, several criteria can be used. As no motion estimation is included on the Symbian implementation, blur detection has become the most important one, along with the distance to the previous stitched frames.

After the registration and evaluation of the frames, some other parameters can be taken into account in order to create the best possible blending algorithm depending on the specific characteristics of the images that are obtained from the previous phase. Some of these parameters are:

- **Blending seam size** – This parameter references the number of columns used in the blending operation. The best performance depends strongly on the sharpness of the raw images, the size of the overlapping portion of consecutive frames and the error rate. When global luminance correction is performed, the error rate is low and the source images are sharp not too many levels for local pixel blending have to be chosen. Shorter values of this parameter offer sharper images while high values help correcting errors and imprecision at other stages.
- **Coefficients for pixel weight** – In the final implementation, a linear weighting mask for blending has been selected. The slope of the linear function is another parameter that has to be taken into account. Steep slopes give sharper results while mild slopes offer a softer transition between frames, ensuring that no vertical stripes are caused by differences in luminance or registration errors.
- **Luminance correction thresholds** – When global luminance correction is performed on each frame, soft transitions between frames give better result than an exact match of luminance values. A threshold for the maximum value of the luminance correction value and a threshold to define the tolerance between frames are needed. It has been experimentally found that tolerances of about five luminance steps and a maximum change of ten levels offer soft transitions without degrading too much each frame due to the loss of dynamic range.

6.6. Quantitative performance on a N93 phone

The panorama builder can construct high quality images in real time under various different conditions with a good quality performance. Figure 36 shows a typical case for panorama capture. A 180 degrees image of a uniformly illuminated landscape is constructed seamlessly. Figure 37 depicts a tough case. A 180 degrees image is composed from source frames with different illuminations and moving objects across the scene.

Figure 36. A typical case for a 180 degree panorama.

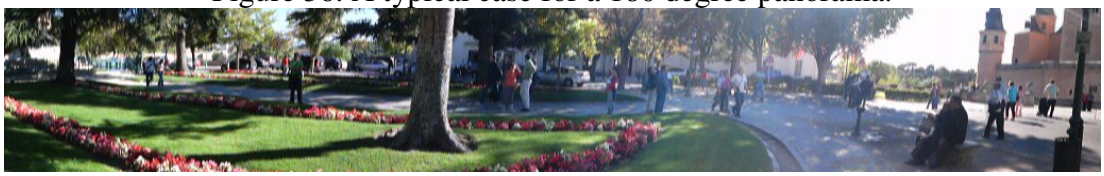


Figure 37. A tough case for a 180 degree panorama.

The memory requirements of the application are related to the video frame size and to the final image desired. Table 3 shows the free RAM memory needed by the application when capturing a 360 degrees panorama for different video frame sizes.

Table 3. Application memory usage.

Image Size	Memory use (kB) (M)	Final image 360 dg (kB) (F)	Memory needed (kB) (1,125 * M + 2 * F)
160x120	77 kB	1 536 kB	3 093 kB
320x240	**307 kB**	**6 144 kB**	**12 374 kB**
480x360	691 kB	6 912 kB	27 842 kB
640x480	1 229 kB	12 288 kB	49 498 kB

The speed of the panorama builder depends on the video frame size and the result image to be stitched. The final image size also determines the time needed to save it on the memory card. Table 4 shows the computation times of the application for a single frame when the capture ends. The overall speed decreases proportionally to the final size due to the time consumption of the blending stage. The video capturing interface has a frame rate limit of 15 frames per second on the N93 phone.

Table 4. Application speed performance.

Image Size	Time per frame (ms)	Frames per second	Saving time (360 dg) (ms)
160x120	30 - 62 ms	15.0 fps	125 ms
320x240	**62 - 125 ms**	**8.0 - 15.0 fps**	**500 ms**
480x360	140 - 250 ms	4.0 - 7.5 fps	1000 ms
640x480	300 - 500 ms	2.0 - 3.5 fps	2000 ms

The frame processing time consumed by the panorama builder can be decomposed on the speeds by each one of the stages of the algorithm. Table 5 shows the

approximated time consumptions of the key functions in the panorama construction process with different image sizes. The registration time for a pair of frames is constant during the whole process while the times for the rest of the stages depend on the number of frames already captured and the overall quality and size of the source images.

Table 5. Time consumption of panorama stages.

Image Size	Registration	Correction	Selection	Blending
160x120	15 ms	0- 3 ms	2-4 ms	13-45 ms
320x240	**40 ms**	**2-10 ms**	**5-10 ms**	**15-65 ms**
480x360	75 ms	5-20 ms	12-25 ms	48-130 ms
640x480	140 ms	10-50 ms	25-50 ms	125-260 ms

The time of the image registration stage is defined by the registration method, the size of the desired computation region and the downscaling factor used. The presence of errors in the registration process depends also on the selected registration parameters. Table 6 depicts the time differences and the error rates for several parameters on a QVGA image registration process. For the final application, compromise between the error rate and speed was chosen.

Table 6. Registration speed performance.

Region size	Interpolation factor and region size		Frames per second	Average error rate
70% QVGA	No	256x256	4 fps	0,1%
	2	128x128	9 fps	0,2%
35% QVGA	No	128x128	8 fps	0,5%
	2	**64x64**	**20 fps**	**0,7%**
20% QVGA	No	64x64	22 fps	2,5%
	2	32x32	40 fps	>5,0%

6.7. Other Symbian phones

Although the main target of the developed application has been the N93 phone, several other phones such as the older N90 or the more modern Nokia N95 and Nokia 6120 have also been extensively tested with excellent results. The application has experimentally shown to be portable.

Nokia N95 phone offers a new platform based on Symbian 9.1 operating system as well as some other improved capabilities compared with N93 phone. The main camera of N95 can capture still frames of up to 5 megapixel, and raw video frames interface is speeded up to 30 frames per second. The processing capabilities of the phone are quite similar to N93's although the larger amount of available RAM allows more space for image manipulations.

Nokia 6120 is a mid-budget phone that offers a fast general purpose processor. The included ARM processor has a clock frequency of 369 MHz. Most of the extra chipsets present in more expensive phones have not been included on the platform. The imaging capabilities of the phone consist of a two megapixel camera that allow the capture of still frames through the imaging API and a video interface that can obtain VGA frames 15 frames per second rates. The application shows excellent performance on this device due to the very little use of floating-point operations.

6.8. Further directions

The model presented in this thesis is a first approach to the mobile panorama stitching problem. However, in order to get best results, the image registration method has to be able to detect not only global motion and rotation on two axes, but any free movement that can result on single frame warping. To achieve such an image registration system, some other methods different than phase correlation have to be explored. A carefully coded feature machine registration method could lead to the estimation of any shift, rotation and tilt between frames even in real time with a good computational performance.

Blending, warping, rotating and changing the scale of a frame are quite costly operations if only a general purpose processor is used. To increase the performance of these operations, the general purpose graphics chips included in newer phones can be used. Taking advantage of a graphic processing unit, if it is present, will save processor time that can be used to improve the quality of the computations. Interpolations and transformations to the frames can be done at high speed with GPU [36]. Figure 38 shows an example of a warping correction procedure using OpenGL ES 1.1.

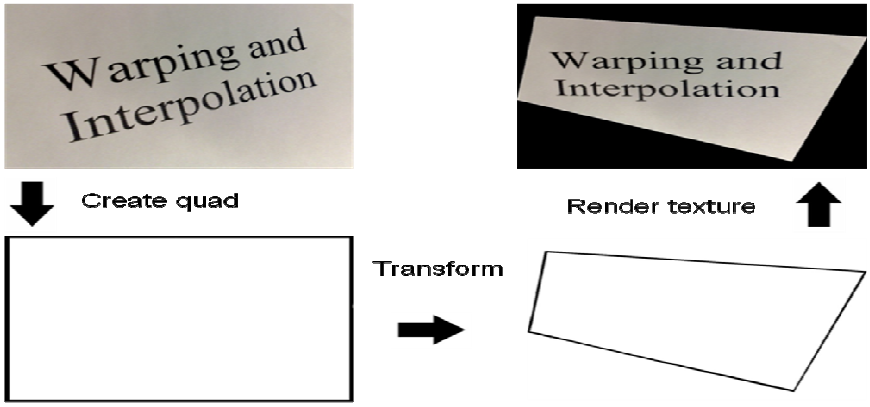


Figure 38. Image warping with bilinear interpolation on OpenGL ES 1.1.

If the graphics processing unit includes a programmable pipeline, it can also be used for general purpose calculations in the registration phase such as feature extraction or robust estimation. Figure 39 shows the differences between a fixed and a programmable graphics pipeline.

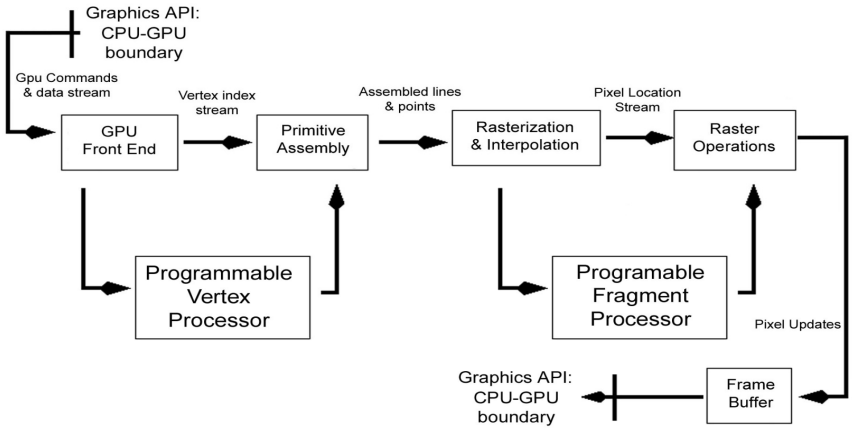


Figure 39. Fixed graphics pipeline vs. programmable graphics pipeline.

7. SUMMARY

Cameras were primarily intended for capturing still and video frames, while the computing and display resources have been optimized for video playback. As a result, the efficiency of other uses is inferior. However, mobile devices' cameras create the need of new applications, capable of using them in different ways other than the usual ones. They usually lack good optics, but the advanced processing capabilities of a mobile phone compared to a normal camera can make the image processing a good solution for solving this issue.

The goal of this thesis was to investigate how a real-time image mosaicking and panorama imaging application could be included in a mobile platform. The implementations were mainly coded for a Nokia N93 although they have been proved to work on several other phone models. A floating-point implementation offered a first approach to the topic, giving the information needed to identify the most challenging parts to improve and offering a first profile of the final application. A fixed-point implementation reached the desired performance in real time with the use of a carefully tailored algorithm and memory management. A library that can take advantage of this algorithm has also been implemented for Symbian OS during this project.

As a result we have a standalone application that can run in several mid-budget phones and that is able to stitch frames at a speed near 15 frames per second, resulting in more than 360 degree panoramas. The algorithms were tested using several sets of parameters. Depending on them, the size and quality of the final image might vary as well as the frame rate that can be high enough to lead the user to a better user experience. Although in this kind of application the user is always collaborative, minor camera shaking due to the handheld capturing process can be compensated.

The selection of the proper algorithms turned out to be a key to achieve real-time performance and good quality images. The results of the development were very promising and panorama stitching will probably become a common feature in mobile phones in the near future.

8. REFERENCES

[1] Boutellier J. & Silvén O. (2006) Panoramas from partially blurred video. IWICPAS06, pp. 300-307.

[2] Heikkilä M. & Pietikäinen M. (2005) M. An image mosaicing module for wide-area surveillance. VSSN ’05: Proceedings of the third ACM international.

[3] Canon Photostitch (accessed on December 2009) URL: http://www.canon.com/camera-museum/tech/report/200206/report.html

[4] Image Composite Editor web page (accessed on 20 January 2010) URL: http://research.microsoft.com/groups/ivm/ICE/

[5] Scalado autorama. (accessed on November 2009) Press Release. URL: http://www.scalado.com

[6] Morpho QuickPanorama web page (accessed on 20 January 2010) URL: http://www.morphoinc.com/QuickPanorama.htm

[7] SonyEricsson (2004) K700i user manual. PDF file.

[8] Bitside Panoman home page (accessed on 15 August 2009) URL: http://panoman.net

[9] SymbianOS (accessed on 20 January 2010) URL: http://developer.symbian.org/main/documentation/index.php

[10] S60 platform web page (accessed on 20 January 2010) URL: http://www.forum.nokia.com/main/platforms/s60/index.html

[11] Hiipakka J. Symbian OS 7.0S (2004), Multimedia Framework. Multimedia Seminar - Mobile Multimedia Application Platforms.

[12] Symbian MMF web page (accessed on 20 January 2010) URL: http://forum.nokia.com/index.php/MMF

[13] Neuvo Y. (2007) DSP, from filter design to system design, past and future. Opening lecture, Finnish Signal Processing Symposium.

[14] Peleg S. & Herman J. (1997) Panoramic mosaics by manifold projection Computer Vision and Pattern Recognition, pp. 338-343, IEEE Computer Society, Washington, DC, USA.

[15] Zitová B. & Flusser J. (2003) J. Image registration methods: a survey. Image and Vision Computing 21(11), pp. 977–1000.

[16] VandeWalle P., Süsstrunk S. & Vetterli M. (2006) A Frequency Domain Approach to Registration of Aliased Images with Application to Super-Resolution. EURASIP Journal on Applied Signal Processing, special issue on Super-resolution. pp. 1-14.

[17] Hannuksela J., Sangi P & Heikkilä J. (2007) Vision-based motion estimation for interaction with mobile devices. Computer Vision and Image Understanding: Special Issue on Vision for Human-Computer Interaction 108(1–2), pp. 188–195.

[18] Lucchese L., Doretto G. & Cortelazzo G.M. (2002) A frequency domain technique for range data registration. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, pp. 1468–1484.

[19] Lowe D. (2004) Distinctive image features from scale-invariant keypoints. Int. Journal of Computer Vision, 60(2):91–110.

[20] Bay H., Ess A., Tuytelaars T. & Van Gool L. (2008) SURF: Speeded Up Robust Features. Computer Vision and Image Understanding. Computer Vision and Image Understanding, Vol. 110, No. 3, pp. 346—359.

[21] Xiao F., Wu H.Z., Xiao L., Tang Y. & Ma W.J. (2004) Auto method for ambient light independent panorama mosaics. Proceedings of International Conference on Machine Learning and Cybernetics, 6, pp. 3851–3854, IEEE, Shanghai, China.

[22] Zomet A., Levin A., Peleg S. & Weiss Y. (2006) Seamless image stitching by minimizing false edges. IEEE Transactions on Image Processing 15(4), pp. 969–977.

[23] Szeliski R. (1996) Video mosaics for virtual environments. IEEE Computer Graphics & Applications 16, pp. 22-30, March.

[24] Davis J. (1998) Mosaics of scenes with moving objects. Computer Vision and Pattern Recognition: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 354–360, IEEE Computer Society, Washington, DC, USA.

[25] Brown M. & Lowe D.G. (2003) Recognising panoramas. International Conference of Computer Vision: Proceedings of the Ninth IEEE International Conference on Computer Vision, p. 1218, IEEE Computer Society, Washington, DC, USA.

[26] Hsu C.T., Cheng T.H., Beuker R.A. & Horng J.K. (2000) Feature-based video mosaic. Proceedings of the International Conference on Image Processing, 2, pp. 887–890, Vancouver, Canada.

[27] Li J.S. & Randhawa S. (2004) Improved video mosaic construction by selecting a suitable subset of video images. Conferences in Research and Practice in Information Technology: Proceedings of the 27th conference on Australasian computer science, pp. 143–149, Australian Computer Society, Inc., Darlinghurst, Australia.

[28] Liang J., DeMenthon D. & Doermann D. (2006) Camera-based document image mosaicking. International Conference in Pattern Recognition,

Proceedings of the 18th International Conference on Pattern Recognition , pp. 476–479, IEEE Computer Society, Washington, DC, USA.

[29] Radke R.J., Andra S., Al-Kofahi O. & Roysam B. (2005) Image change detection algorithms: a systematic survey. IEEE Transactions on Image Processing 14(3), pp. 294–307.

[30] Zhu Z., Xu G., Riseman E. & Hanson A. (1999) Fast generation of dynamic and multi-resolution 360 degrees panorama from video sequences. Proceedings of the IEEE International Conference on Multimedia Computing and Systems, 1, pp. 400–406, Florence, Italy.

[31] KISS FFT sourceforge project. (accesed on January 2010) URL: http://sourceforge.net/projects/kissfft/

[32] Boutellier J., López M.B., Silvén O., Tico M. & Vehviläinen M. (2007) Creating Panoramas in mobile phones Proceeding of SPIE Electronic Image, vol. 6498.

[33] López M.B., Boutellier J. & Silvén O. (2007) Implementing Image Stitching in Mobile Phones. Finnish Signal Processing Symposium.

[34] Silvén O., Hannuksela J., Bordallo López M., Turtinen M, Niskanen M, Boutellier J., Vehviläinen M. & Tico M. (2008) New video applications on mobile communication device. Multimedia on Mobile Devices. IS&T/SPIE Electronic Imaging. International Society for Optics and Photonics.

[35] Takacs G, Chandrasekhar V, Gelfand N, Xiong Y, Chen W-C, Bismpigiannis T, Grzeszczuk R., Pulli K. & Girod B. (2008) Outdoors augmented reality on mobile phone using loxel-based visual feature organization. IEEE Transactions on pattern analysis and machine intelligence.

[36] López M.B., Hannuksela J. & Silven O., Vehviläinen M. (2009) Graphics hardware accelerated panorama builder for mobile phones. IS&T/SPIE Electronic Imaging (pp72560D). International Society for Optics and Photonics.