Mobile augmented reality system for maritime navigation

Jean-Christophe Morgère Morgère

THÈSE / UNIVERSITÉ DE BRETAGNE SUD
UFR Sciences et Sciences de l'Ingénieur

Sous le sceau de l'Université Européenne de Bretagne

Pour obtenir le grade de :
DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE SUD

Mention : STIC

présentée par

Jean-Christophe Morgère

Lab-STICC

Mobile Augmented Reality System for Marine Navigation

Thèse soutenue le 04 Avril 2015,
devant la commission d'examen composée de :

M. Olivier Romain
Professeur, Université de Cergy Pontoise - ENSEA / Président

M. Emmanuel Casseau
Professeur, Université de Rennes 1 - IRISA / Rapporteur

M. Pascal Guitton
Professeur, Université de Bordeaux - Inria / Rapporteur

M. Goulven Guillou
MCF, Université de Bretagne Occidentale - Lab-STICC / Examinateur

M. Jean-Philippe Diguet
Directeur de recherche CNRS, Université de Bretagne Sud - Lab-STICC / Directeur de thèse

M. Johann Laurent
MCF, Université de Bretagne Sud - Lab-STICC / Co-encadrant de thèse

Système de réalitée augmentée mobile pour la navigation maritime

~

1 Mots clefs

Réalité augmentée, aide à la navigation.

2 Résumé

A l'instar d'autres domaines d'activités, l'environnement du marin s'est enrichi d'appareils électroniques à des fins d'aide à la navigation et de sécurité. Dans le cas de la navigation maritime, ces dispositifs ont pour principal but de donner des informations sur l'environnement dans lequel évolue le bateau: profondeur d'eau, bouées de signalisation par exemple et sur son état: cap, vitesse, etc. Le complexité et le nombre d'appareils à bord des bateaux dépendent de la taille du bateau, d'un GPS portatif pour un jet-ski à un ensemble complexe d'ECDIS, de radar et d'AIS sur les cargos.
Aujourd'hui, malgré l'évolution des appareils d'aide à la navigation, les accidents perdurent (échouage et collisions) et sont en partie dus aux erreurs humaines (33%). Il éxiste 3 grandes causes:

- La charge cognitive trop importante. Cette dernière est liée à la complexité des appareils (ECDIS par exemple) [Jie and Xian-Zhong, 2008], leurs nombres et leurs orientations différentes de la vue pont [Prison and Porathe, 2007].
- La mobilité de l'information. La majorité des appareils électroniques à bord des bateaux est positionné a un endroit et ne permet pas l'accès aux données en dehors de cet emplacement.
- Le manque de pratique de plaisanciers. Les règles de navigation sont indispensable à la sécuritée du marin et peuvent être oubliées par manque de pratique et causent des incidents (ou accidents).

Dans ces travaux, nous proposons une solution mobile pour rendre accessible les données en tout point du bateau. De plus, le prototype est basé sur la technologie de réalité augmentée pour afficher les données dans le champ de vision de l'utilisateur pour une reduction de la charge cognitive. Enfin l'application executée sur le prototype delivre uniquement les informations essentielles pour une navigation sécurisée.

La première partie de cette thèse détaille une étude utilisateur menée pour comprendre les habitudes et les besoins des plaisanciers afin de sélectionner les données à afficher et leur représentation graphique. Suite à cette

étude, une analyse du besoin logiciel et matériel a été menée. Le but de cette analyse est de lister les differents éléments pouvant supporter les contraintes maritimes (soleil, brouillard, nuit, houle, mobilité) afin de proposer un appareil et une application mobile sur technologie de réalité augmentée.
La deuxième partie de cette thèse aborde le design du prototype (architecture, display) et du logiciel. Cet ensemble a pour but d'aider le plaisancier lors de conditions météos difficiles. La technologie mise en oeuvre dans ce nouvel outil est la réalité augmentée, plus précisément Optical See-through. Le principe de l'application exécutée sur le prototype est un serveur proposant des services à l'utilisateur en fonction des appareils disponibles à bord du bateau tout en limitant la charge cognitive du marin. En effet, l'application conserve uniquement les informations utiles et les affiche dans le champ de vision de l'utilisateur, de plus elles sont superposées à sa vue directe sur le monde.
La dernière partie, elle, décrit le générateur de carte qui nous permet d'extraire les données maritimes issues de différents formats de carte pour les adapter au format 3D utilisé dans l'application. Ce dernier est capable de placer automatiquement des objets 3D pour représenter le système de balisage, les amers, etc. Ce générateur a été développé pour reconstituer des zones interdites comme les zones de profondeurs, d'interdiction de chasse sous marine, etc. Ceci a été mis en oeuvre afin de répondre au mieux aux besoins de tous types de plaisanciers, de l'utilisateur de jet-ski au yacht et du pêcheur au plongeur amateur par example.

Mobile Augmented Reality System for Marine Navigation

~

3 Key words

Augmented reality, navigation aid.

4 Abstract

Compared to the other activities, the sailor's environment has been enhanced with electronic devices in order to help the sailor and improve the security. In case of maritime navigation, these devices are mainly used to give information on the environment which the boat moves such as the water depth, seamarks for instance and its state: bearing, speed, etc. The complexity and the number of devices onboard depend on the boat size, from a wearable GPS on a jet-ski to a complex set of ECDIS, radar and AIS on a merchant ship. Today, despite the evolution of the navigational aid devices, the accidents still happen and they are due in part to human errors (33%).
There are 3 main causes:

- The cognitive load issue. The latter is linked to the complexity of devices (ECDIS for instance [Jie and Xian-Zhong, 2008]), their quantity and their orientations which is different from the bridge view or the user's field of view [Prison and Porathe, 2007].
- The information's mobility. Most of the devices onboard are put in a specific place and the data are not accessible outside this place.
- The lack of practice of the recreational boats owner. The maritime navigation rules are vital to sail safely but some of sailors can forgotten some significations and an accident may result from a wrong choice.

In this thesis, we provide a mobile solution to make accessible data everywhere on the boat. Furthermore, the prototype is based on the augmented reality technology to display data in the user's field of view to reduce cognitive load. Finally, the application run on the prototype delivers only the relevant information for a safe navigation.

The first part in this thesis details a user study conducted to understand the sailor's habits and their needs in order to select data to display and their graphical representation. Following this study, a needs analysis on the software and hardware has been realized. The purpose of this analysis is to list the different elements that are usable under maritime constraints

(sun, fog, night, swell, mobility) in order to provide a device and a mobile application based on the augmented reality technology.
The second part in this thesis reach the prototype (architecture, display) and the software design. Both of them aim to help the sailors when the weather is bad. The technology used in this new tool is the augmented reality, more precisely an Optical see-Through system. The principle of the application run on the prototype is like a server, which provides services to the user depending on devices available onboard, while limiting the sailor's cognitive load. Indeed, the application keeps only the useful data and display them in the user's field of view, furthermore they are superimposed on his/her direct view of the world.
The last part in this thesis details the chart generator, which allows us to extract maritime data from different chart formats to adopt the 3D one used in the application. The latter is able to automate the placement of the 3D objects to build the buoyage system, landmarks, etc. This generator has been developed to recreate the danger areas such as depth areas, prohibited diving areas and so on. This has been done to satisfy user's expectations from the jet-ski to the yacht owner and the fisher man to the diver for instance.

Contents

Chapter First

Main introduction

The context of the thesis is the study of a new solution for mobile assistance to navigation assistance. We first need to remind the main objectives of Marine Navigation, present the current use of electronics in this domain and detail the current challenges. Finally we introduce the contribution of the thesis.

1.1 Marine navigation

The navigation is the science or combination of methods that answers three major issues: where am I, where do I go, and what is the environment in which I move? (3W). Indeed, the navigator should know, first, its position relative to a reference system (geographic coordinate for instance) or a fixed point. Secondly, the navigator has to be able to compute or measure the safest way to reach its destination. Other complementary information relative to the navigation can be retrieved such as distances, duration, speed, time of arrival, etc. Those data may also be taken into account during navigation to answer more precisely the 3W.

Marine navigation is all human maritime traffic on the sea. There are two kinds of marine navigation that can be practiced, the coastal navigation and the ocean navigation. Historically, the first coastal navigation appeared 100.000 years ago. Men used pirogues to move on the water (on very short distance) for hunting and fishing. Only view with landmarks allowed identifying and understanding position and the environment wherein the boat moves, no other means were available for navigators except visual marks. This is still the case today for boats such as kayaks, dinghies, etc. Gradually, science improved the marine navigation and led to discoveries and inventions. Boats, navigation techniques and navigational instruments improved. However, few innovations allowed the practice of the second kind of marine navigation: ocean navigation. This term is employed when there are no lands visible from the boat. The first ocean navigation technique used only stars to answer the question where do I go and where am I. This is only usable at night but requires the sky to be clear (Antiquity).
Next, we will only present the most important innovations on navigational instruments to understand their importance. The first one is the compass;

it has been invented by the Chinese and brought to Europe in the 13th century. This instrument gives the direction of the North Magnetic Pole (to get the answer for "where do I go?"). The second one is the Astrolabe. This navigational instrument was invented in the Antiquity, it can compute the latitude by measuring an angle between a star (Polaris for instance) and the horizon ("where am I?"). It also gives the possibility to compute time with the sun during the day and the stars at night. Both innovations were paramount to enable the discoveries of the oceans and the world in the 15th century. Next is the Mercator projection, which allows the representation of the earth on a plane surface in the 16th century, this projection is the biggest improvement for nautical charts and currently it is still the same. The next important discoveries appeared in the 18th century, the first is the chronometer, it was invented by John Harrison and it was used to compute longitude with the help of stars. The sextant is the second one, it was the best instrument to get precise latitude, longitude, and also an altitude even if the boat is moving. It is used to measure the angular distance between two visible objects (celestial or not). This instrument is still used in ocean navigation when there are no electronic instruments onboard.

During the numerous trips and visits, sailors established new nautical charts that were much more improved with Mercator projection and the modern hydrographic, which appeared in the 18th century. The latter allows to get more and more precision for the contour of lands and water depths in coastal waters and rivers. Currently, in France, this is the SHOM (Service Hydrographique et Océanographique de la Marine), which is in charge of measuring depths, contours and updating nautical charts. An example of a recent nautical charts is visible in Fig.1.1. This kind of chart is always orientated in North-Up and there are three kinds of information written on it: data under the sea (depth water, rocks, etc.), above the sea (buoys, beacons, etc.) and on land (contours, landmarks). There are much more useful information but we will explore them in more details and see how a chart is used during navigation.

In addition to the nautical charts, the sailor can use a new orientation system. It has been introduces to deal with the ever growing maritime traffic and improve the overall security, this system is made with seamarks. They are visible from the user's view (bridge view) and are used to indicate some danger or the way to follow safely. So seamarks bring some rules to the sea, the user must know their meaning to sail safely in the maritime traffic as the signs are to the road. Today, the seamarks are compliant, in most of countries, with a standard: the International Association of Lighthouse Authorities (IALA) Maritime Buoyage System [Pielou, 1978]. It defines one system of marks of maritime buoys/beacons for maritime navigation. Seamarks are specified with four parameters: shape, colour, a top mark and depending on the buoy/beacon an emitted light that dissociates the type of danger and shows the right way. But for historical reasons, two

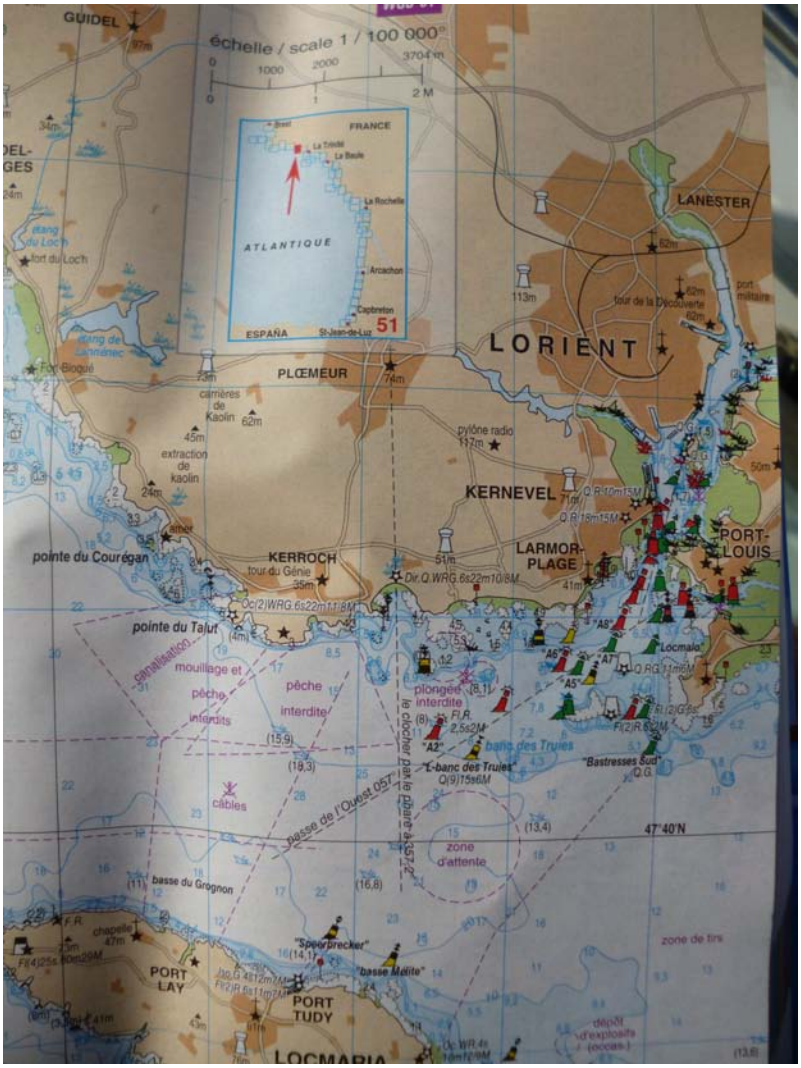


Figure 1.1: Picture from coastal bloc, Lorient harbour chart

different systems are used worldwide: region A and B. Only lateral buoys colour differ from another (all Figure in the thesis will use region A). Lateral buoy/beacon for instance (red and green buoys in Fig.1.2) define the way to go or to leave the harbour, cardinal buoys/beacons (black and yellow buoys in Fig.1.2) define the way to take relative to the north to avoid danger (reef) and isolate danger buoys/beacons (black and red beacon in Figure) report a danger area around it.

In addition to the seamarks sailors use landmarks. The most known are the lighthouses: this landmark, emits a flashing light to show the ones to follow and it can be visible from far away due to its size and the power of lights. But it is not the only one, indeed some elements on the land are visible from

Figure 1.2: Representation of an harbour approach with IALA compliance from Sealite.com

far away: church/water towers, etc. Most of the time, it is the higher point in a town or a village on the seaside that is used to get a fixed point to follow a heading during navigation. This is often used to take a specific way to retrieve the entry of an harbour for instance. The use of landmarks is very important in navigation and that is why we can retrieve them in the nautical charts, an example is visible in Fig.1.1, church and water towers are visible on it, on the top. Thanks to the seamarks and landmarks, the user is able to understand the environment wherein he/she is sailing (in coastal navigation) and can refer to the nautical charts to get more information.

On the sea, there are two kinds of users: recreational boats and professionals. Both of them use seamarks/landmarks and a nautical chart to get a safe maritime navigation. The latter is not exactly the same for all because first, the draught of the boat can vary from a few centimeters to a few meters. Second, some electronic devices are onboard today to help the sailor to take the right way and avoid dangers. But they vary from one boat to the other depending on the size, the level of equipment is usually related to the cost. Electronic devices are made to help the user, to offer more safety and more and more services. Electronic devices are more and more present on boats in the same way they spread in automotive, home and everyday human environment. In the next section we detail the current devices the sailor can take advantage of.

1.2 The electronics in the service of navigation and safety

The most electronic device used in navigation is the GPS. It is the easiest and fastest solution to get a position (latitude, longitude). It is possible to extract a lot of information from a GPS to enrich travel data. Some useful data such as course over ground (COG), speed over ground (SOG), time are readable every seconds.

Then we can mention the sounder, which is a useful on a boat, it is able to measure depth water with acoustic technology (for the most of them). The sounder is used to check the water depth under the hull or the keel to avoid groundings. It is also used to identify the kind of seabed (sand or rocks) to anchor or to fish in the context of pleasure boating. Data from this sensor will not be taken into account in the thesis even if it is one of the most important data to avoid groundings. But it is possible to read the water depth from electronic nautical chart and the tide table and then, it is possible to estimate the current depth water and display this information to the user. Moreover in practice it is complicated to communicate with this device on recreational boats, it can be only plugged to chartplotter in recreational boats.

The navigation software or chartplotter is a device that reads electronic charts, generally a GPS is plugged to them. This is more secure for sailor to use this kind of device than a classical GPS and a paper chart because the position, the speed and the course of the boat is directly visible on the display. The user has no computation to do (retrieve its position on the chart). The sailor can understand where is he/she on the chart thanks to a symbol that shows its position and orientation in real time (North-up or Course-up sometimes).

PC are more and more used onboard (recreational boats), some navigation software are able to decode AIS (Automatic Identification System) data. The AIS is an electronic device, which can be a receiver or both emitter and receiver. It was made to send some information by wireless at a few kilometers around the emitter. There are two kinds of data, data boat with size, name, type, draught for instance and navigation data such as position, speed, course, status, etc. With some computations between AIS and GPS data it is possible to detect if boats are on a collision way and prevent the user from a close collision. This device is useful for ocean navigation and into maritime traffic channel. But it costs few hundred Euros for just a receiver and its antenna and has to be plugged to a computer or chartplotter to decode and display data. So only professional, regatta and a minority of recreational boats have got one onboard.

The next safety device is similar to the AIS: this is the marine radar. It is also used to detect danger and collision but this is not the same technology

as the AIS. More than AIS, the radar is able to detect some Unidentified Floating Objects (UFO). With some computations the device offers information on the object detected such as heading (if it is moving) and distance to help the user avoiding collision. It is not a common device for recreational boats, like AIS it requires the installation of an antenna and a receiver, it requires space but is primarily a cost reason.

ECDIS (Electronic Chart Display and Information System) is the only electronic device that can be used instead of paper nautical chart for professional. In addition to the GPS, ECDIS is able to decode RADAR data and plot on the ECDIS display some information about mobile objects on the sea. This is the International Maritime Organization (IMO) that decided to accept ECDIS as a replacement of paper nautical chart. IMO also force boats starting from 500 gross to use AIS onboard for instance (since 31 December 2004).

All electronic devices were made to help the user in navigation, and also to get a safe navigation but devices cannot be the same on a recreational boat or a cargo for instance. So, we focus mainly on recreational boats because there are lot of kind of boats and we want to offer the same service even if the size and complexity vary and impact electronic onboard. However the improvement of equipment based on electronic devices is not enough, the way to use and to display information is also a major concern.

1.3 Observation

Even if there are more and more electronic devices onboard, accidents still happened on both vessels and recreational boats. In this thesis, we focus on the groundings and collisions because these kind of accidents can be decreased for many reasons, which are detailed later on. Real observations (in Table 1.1) show that groundings and collisions represent one third of accidents for recreational boats in 2012 in the USA [Guard, 2012]. These accidents are not caused by damages on the boat or its engine, or even by the lack of information onboard. Most of the time, they are caused by human mistakes. On ferries and cargo ships for instance, there are lot of navigation

Table 1.1: Top five primary accident types in 2012 in USA [Guard, 2012]

Accident Rank	Accident Type	Number of Accidents	Number of deaths	Number of Injuries
1	Collision with recreational vessel	1010	47	711
2	Flooding/swamping	509	68	193
3	Collision with fixed object	475	50	340
4	Grounding	422	10	244
5	Skier mishap	387	20	388

assistance instruments available such as AIS, radar, digital nautical chart plugged to ECDIS (often two) and nautical charts in addition to the bridge view. So there are enough devices to prevent from collision, groundings and keep a safe navigation but there is still accidents. For example Consta Concordia (2012), Express Samina (2000), Gibraltar strait accident (1999) were only due to awkwardness or distraction. Despite many tools currently available to navigators, it appears that the groundings and collisions are not decreasing.

It's because of the human factor. Some reasons of the human errors are explained in the following, we focus on the recreational boats, which are our targets in the thesis. The first cause of this kind of errors is the orientation computation, indeed charts are always printed with the North on the top (North-up), whereas ENC, radar and AIS have often the course of the boat on the top of the display (course-up). This difference between those orientations leads to cognitive issues, the sailor must realize mental arithmetic to put information in the bridge view (user's view) and to make a decision (to act on the course and boats speed).

The second cause is the number of information to decode that grows with the number of electronic devices and the complexity of instruments [Jie and Xian-Zhong, 2008]. So, the cognitive load increases and the sailor has multiple orientation computations to do. Moreover in case of emergency, stress increases the probability of making wrong decisions, in the case of arriving to an harbour with traffic and/or bad visibility for instance.

The third cause of accidents is the mobility of information, indeed; most of sailing boats don't have the chartplotter close to the bridge view. So the sailor must go inside the boat to check the paper chart or electronic devices before taking a decision on the way to follow.

And the last one is the ignorance or the oversight of marine navigational rules. This is especially true with occasional use of rental boats. Fig.1.3 gives an example of a rule to follow on the sea to avoid groundings: the cardinal marks. In navigation, the sailor must understand the signification of the top mark, colours or lights from cardinal buoys: is it north, south, east or west? Then, sailor has to answer this question: is it the way to follow or the danger way ? It is not easy to answer these questions, so let's consider the following example to understand the underlying problem. The left part in Fig.1.4 is a view of a nautical chart with a North-up orientation. A red symbol represents the boat's position and its course. The user wants to go to the waypoint represented by the yellow mark surrounded with a red circle. The sailor looks at the cardinal buoy (right side in Fig.1.4) and has got a compass that indicates the course of the boat related to the magnetic north. Then the sailor has to take the right decision: go on the left (port) or the right (starboard) of the buoy. First the sailor must know the meaning of this buoy (explained in Fig.1.3): the buoy shows the right way, not the danger position. Secondly the sailor must compute the orientation between his/her

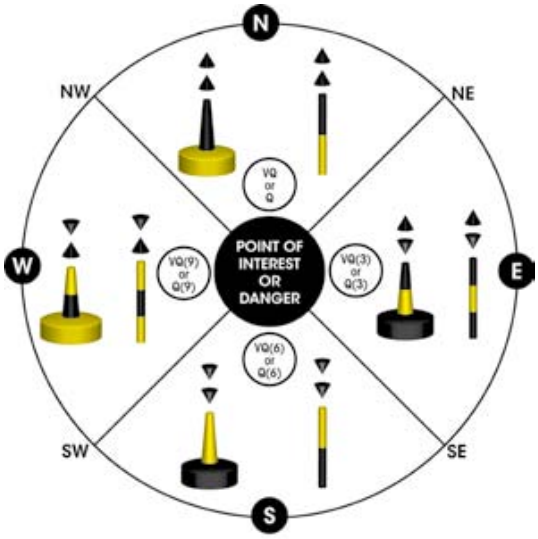


Figure 1.3: Cardinal marks meaning with IALA compliance

own relative to the north (with a compass for instance) and the way relative to the north indicated by the buoy to get the right way (in Fig.1.4). In this

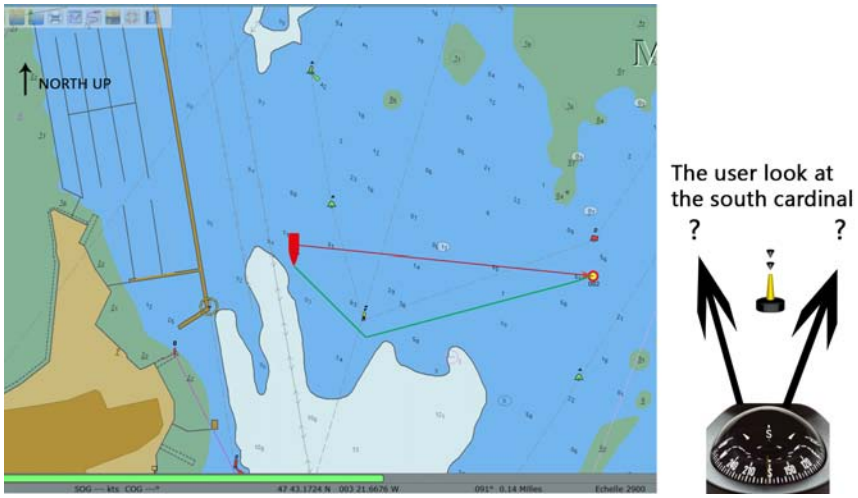


Figure 1.4: Example of cognitive load with a cardinal mark

example, the sailor has to let the buoy on the port (left) of the boat not on the starboard (right) to avoid the danger, as it is represented by the green and red lines. It represents a cognitive load which grows up with the number of electronic devices onboard, boats and seamarks on the sea and moreover it adds stress. The cognitive load is also not compatible with tiredness or absent-mindedness, which increase time to compute, and error in decisions. In conclusion the device which can reduce the human mistakes should be the

next maritime revolution.

1.4 Issues & Contributions

As we shown in the previous section, the scientific challenge are the data mobility, the quantity of information to decode, the orientation computation and the ignorance or careless mistake. In order to respond to the data mobility issue, the system has to be portable or wearable. Indeed, on a boat the user has to access data easily and those data should be available at all time in any situation. One of the solutions that answers all the constraints inherent to the size of the boat (from jet-ski to sailing boats) would something small, ergonomic, with a low footprint that is able to get GPS and nautical chart data. Our solution is a prototype named Marine Mobile Augmented Reality System, which is a fully embedded system that matches the size and power consumption constraints from recreational boats to professional vessels.

The next scientific challenge is the huge quantity of information. Indeed, filter all data from a paper chart, a chartplotter and the bridge view is very painful for sailors depending on the traffic and the area (lot of rocks and seamarks). What is why the system has to help the user to find only the right information to take a quick decision on the way to follow without error. We offer a 3D chart generator to sort data and construct a new 3D chart. This generator is necessary to handle the tedious conversion of objects. For instance, buoys and beacons only represent 128 objects in the entrance of Lorient harbour. For every objects we must obtain GPS position, translate it, load the right 3D model in a 3D software and export it. We also offer flexibility to generate charts depending on the user profile. A recreational, a fishing or a sailing boat don't have the same kind of requirements, for instance regatta data could be introduced with the generator to get race buoys, fishing areas or others. This generator also reduces onboard computing and offers enough information to get a safe navigation with seamarks, landmarks and danger areas related to the water depth.

The third contribution is the orientation computation, sailors have their own field of view which is different from the North-Up or course up. So, the system must display or show sea data in the user's field of view. The system must also work in all sea conditions on the sea without any troubles and during several hours. The sea is an outdoor and complex environment : humidity, solar, night, rain, fog and movements. Our solution is a Marine Augmented Reality Navigation Assistance Application (MARNAA) that is developed on embedded systems with graphical interface under ergonomics, mobile and context (marine navigation) constraints. We take into account the new embedded technology: augmented reality glasses, and more precisely the see-through display to let the user see the real world. The system has to

be easy to use and design for everybody, from a landlubber to an old salt, to limit the careless mistake on recreational boats.

1.5 Adopted plan

We made the choice to adopt a distributed presentation of related work that fits with the disparity of the contributions, namely we have different state of the art in each chapter. The first section in this thesis, titled Application design, describes the methodology in order to get a navigational assistance application: from users to application. To design an acceptable application from users, a study on their habits and needs has to be put in place, this step is described in the first part of this section. Then, we describe how we use the ergonomic and context criteria to design the application interface. In the last part we detail the software needs and hardware chips to satisfy user expectations in term of colours and forms.

The second chapter, entitled 3d chart generator, details the proposed method to filter ENC data and generate application charts. This section is broken down in two parts; the first one is focused on the sorting, which is based on data queries and the second describes the 3d generator that uses 3D object library, deduced from IHM queries. The objective of this step is to generate 3D chart that will be exploited in the navigation assistance application. Some results are presented as example.

The third section, entitled Embedded design, details a mobile augmented reality prototype. The first part of this chapter is a description of the hardware architecture, then the software architecture to support the application that takes into account the user habits and expectations is detailed.

Finally, the last chapter concludes on the application interest and embedded system proposed. The future works and perspectives are then taken up.

Chapter Second

Application Design

Contents

2.1 Introduction

One of the contributions of the thesis is the navigational assistance application embedded in a mobile augmented reality device. Very recently applications have emerged, most of them after we start this study. Today there are few solutions that could help the user as we can see in section state of the art. We don't know if some user studies have been realized to design them because these solutions are very different from one to the other in terms of graphical interface. So, we have decided to conduct a study about the interface with the aim to favour acceptance by users. We first propose to study the devices and data that are used onboard in order to understand the boat environment, this part is detailed in Section 2.3.2. Once the required data are listed, the graphical elements have to be designed in accordance with the maritime context and under ergonomic rules. This is the next step of our study and it is presented in Section 2.3.3. The end users are the real decision makers, so they must be integrated in the design loop, that's why we chose to integrate them in the three phases. Finally, the Section 2.4 is an analysis of needs related to the software tasks and the hardware architecture. All required elements are deduced from the graphical interface study and the maritime context (positioning of data). The aim is also to consider the needs with regards to the context of a system embedded in a wearable device with enough resource to execute the application.

2.2 State of the art

The most advanced work to reduce human mistakes in navigation are based on a 3D application from Thomas Porathe. He offers an application that draws 3D charts [Porathe, 2006] with the land, NOGO areas, indicates boats from AIS data, other objects from radar and so on in order to help users to make the right decision in a very short time. All data are displayed in the bridge view, with the help of the course from a GPS, on LCD screen. A representation of the prototype application is visible in Fig.2.1.

The first interest of his application is the use of the bridge view to limit users computation and reduce time to make a decision [Prison and Porathe, 2007].

The second one is the use of augmented data such as cone, coloured surface (NOGO areas), in order to interpret some data from charts in ergonomic and intuitive form visible in Fig.2.2. In the latter, the sailor can see from far away the red and green channel buoys (starboard and porthand) thanks to the cones drawn by the application. Some other information are displayed as the road sign to make it more comprehensible for the user.

The last key point of his application is the management of the most important danger for a boat: the water depth.

Figure 2.1: Prototype and photo of the Gothenburg approach.

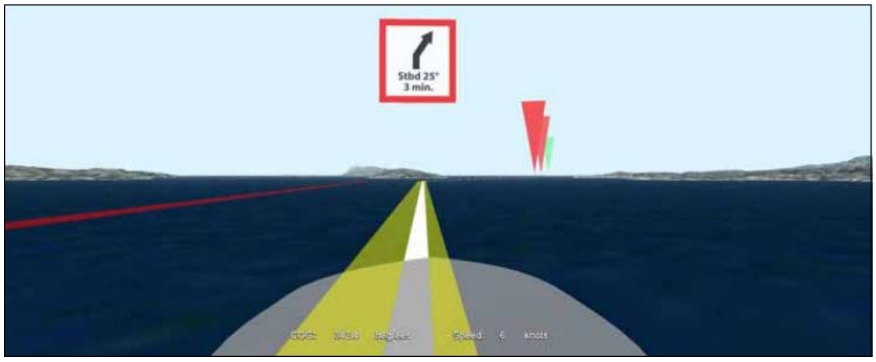


Figure 2.2: Prototype of the Porathe application

The main feature in a navigation assistance application is avoiding groundings, but it depends on the tide and the boat draught. His application is very precise on NOGO areas. Indeed, it is able to take into account the tide of the sea, the boat draught even the swell size to compute the right depth between seabed and the hull or the keel boat. So the NOGO areas are presented as green areas in Fig.2.1 and this is the best representation to understand easily if there are enough water under the hull or not. The sailor just has to sail outside those area to avoid groundings.
Even if this application is very interesting thanks to the ergonomic aspect, the NOGO areas and 3D charts, this system is designed as a software for a fixed PC on a vessel bridge. The 3D charts take a lot of storage and are very time consuming to be implemented on embedded device. So, it is not suitable for recreational boats (including sailing boats) that may not have a PC onboard in order to run the application. Contrary to Porathe's PC application, we don't aim to introduce an important amount of information including complex chart and 3D representations of islands and seafront because this is not fit our needs.

The application is close to a virtual reality one, so this is not useful to superimposed the real view with this virtual world on a mobile device. Moreover, there are too much information on the display for a mobile device such as a smartphone or an augmented reality glasses that not have a large field of view.
The next drawback is that only charts from a region in Sweden has been built, the other areas are not available and need to be designed, there is no information about how this step can be automatized. In conclusion this application is not suitable for recreational boats since it requires a large amount of computations and it is limited for Sweden areas.

Another solution to improve the security on maritime navigation has been provided [Hugues et al., 2010]. The specific device used in this solution is a gyroscopic thermal camera to detect boats. This camera can be rotated to look around the boat. With the help of augmented reality, the system is able to follow some boats and display the direction and speed [Hugues et al., 2010]. The system seems to be efficiency because it provides useful information in fog and night conditions, but there will be more constraints with a rough sea or swell that could hide the boats). The main drawback of this solution is the architecture. Indeed, it uses a gyroscopic camera thermal, a PC and an IMU, so the cost may be expensive and the place and energy required is too big for a recreational boat. This solution aim only professional and yachts.

An application appeared during this thesis, this is a mobile application designed for Apple smartphones and tablets: SeaNav. The main function of this navigational application assistance is the capacity to load electronic

nautical charts. But this application also offers an AR mode that uses the camera to catch the real world and superimposed some data on it. A screenshot of the AR mode is visible in Fig.2.3.

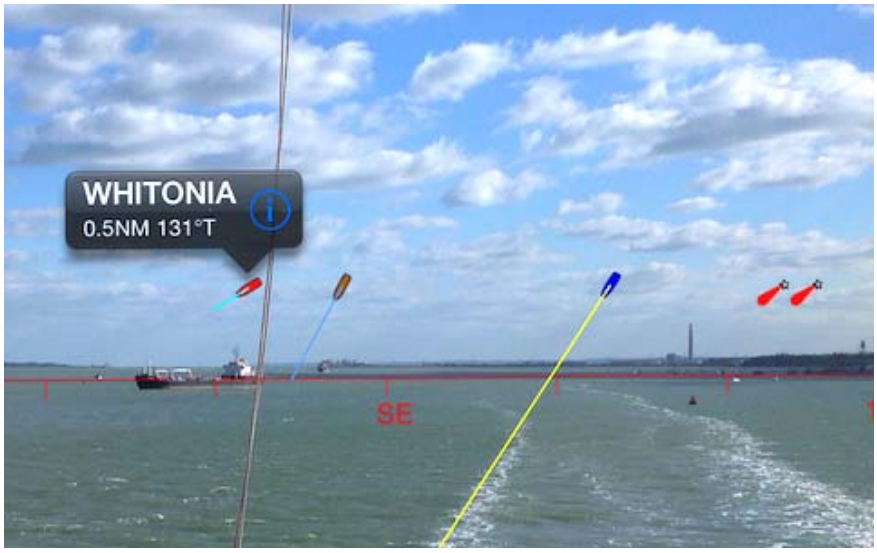


Figure 2.3: iPad version of SeaNav in augmented reality mode.

This mode offers the visibility of maritime data symbols of electronic nautical charts for the USA or the UK. The symbols displayed in the application are the same as we can see on a classical nautical chart, so they are understandable for users that are accustomed to charts. But it is not possible to test the application with the AR mode in France, so we don't know if the application is useful or not, ergonomic or not.

This application seems to be less ergonomic and intuitive than Porathe's one. Indeed, some objects visible in Fig.2.3 are not placed on the sea but on the sky. Some boats are placed on the display with a line which may represent their course but there are no perspectives, so it is difficult to understand the boat position and course. The other disadvantage is the mobile device used, the application is limited to Apple devices. Moreover a smartphone is not practical to use on a boat, it requires the use of the hands to hold it. All devices which need to be hold in hands are not ergonomic in maritime navigation. If a gust arrives during that the sailor uses the application on the smartphone, he has to free his hands and manoeuvre the sailing boat in emergency for instance. So, the smartphone or the tablet could be thrown out or dropped by the sailors, it is not the right support for this kind of application.

In conclusion, this application is not useful because it requires one or two hands, an Apple device, the interface is not ergonomic and it is usable only in UK or USA.

The last application is the closest to ours, it also appeared during the thesis and it is named SmartChartAIS. It is an Android/iPhone application that loads electronic charts from USA as a classical navigation software. With a web connection the application is able to get AIS data from a web server and draw boats on the display. There is an augmented reality mode that runs camera and displays maritime data from charts and community data such as people, harbour around the user for instance. A screenshot of the AR mode is visible in Fig.2.4. The use of the camera's field of view

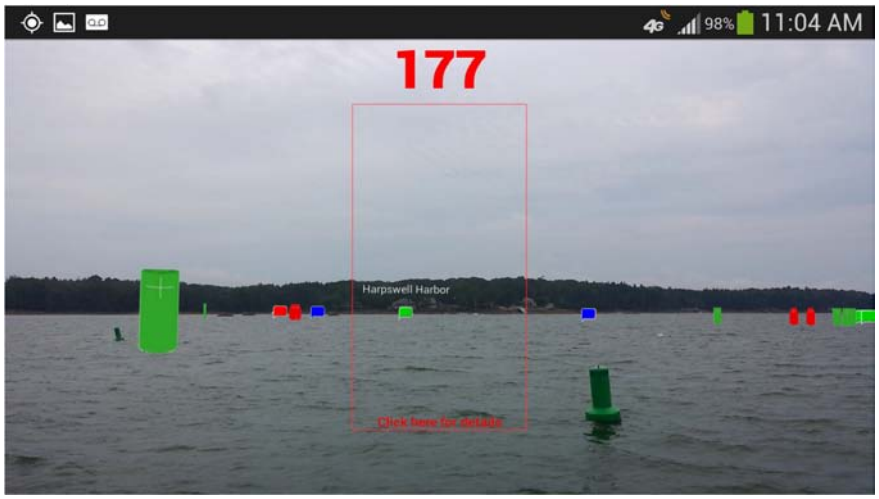


Figure 2.4: SmartChart AIS view from a tablet

which is close to the user's field of view to display data is an advantage. However, this application is focused on community features even if it is also able to load electronic navigational charts as SeaNav. A specific AR mode is available to help the user to detect some buoys, but it is only available in the US. Like the SeaNav application, it is only accessible on smartphone and tablets, which are not the best solution for ergonomic reason in navigation.

The application doesn't seem to be ergonomic because the only objects related to the navigation are 3D objects (see Fig.2.4), which are superimposed on the camera view. This should be not the best graphical solution for users, this is the only information visible in the screenshot.

In conclusion, this application doesn't meet our objective for 4 reasons: it requires to hold the device, the interface has been design with human-interface concerns and methodology, it is limited to US sailors and it is based on the use of a power-hungry camera.

Today, if a sailor wants to use a navigational assistance application with an augmented reality mode on his recreational boat, he has only two solutions: SeaNav and SmartChartAIS. But with these applications the sailor

must own a mobile device (smartphone, tablet) and navigate in the US or UK. In the two cases, the application displays few maritime data in the user's field of view if the sailor holds the device in front of his eyes.

In SmartChartAIS and SeaNav, we don't know if the developers have taking into account the user habits and opinions because the design is very different from one to the other. In SmartChartAIS, elements are not much and very simplistic, whereas in SeaNav the position of the objects is not good. In both applications, there are nothing about the NOGO area to prevent from grounding risks. In the two applications, we think that the graphical elements can be improved with the help of the end users.
The position between the real buoys and the graphical objects is not precise as an indoor application with markers for instance. But, due to the application context, a precision less then 5 meters is enough to help the user in his navigation (in Section 3.2.2).

In order to avoid these problems, we put the end users in the center of our considerations. Our solution is designed under graphical and ergonomic constraints and based on a see-through augmented reality system to avoid the use of hands in navigation. We detail this approach in the next sections of this chapter.

2.3 From user to application

2.3.1 Introduction

It is very important to work with the end users to design and offer an useful navigation assistance application especially in a domain like maritime navigation assistance where the user can be in hazardous situations. To avoid a reject from users we managed a study to get end users in the design loop, which is detailed in Section 2.3.2 of this chapter. Our method is based on Fig.2.5, we study first the end user habits with the help of queries, second we realize some graphical objects, which are also evaluated with the end users to find the best graphical interface for the application.

As the first step of the study, we met a restricted panel, step 1 in Fig.2.5, which is composed of five recreational boats owners from a local yacht club. The boat list is the following: one inflatable boat (5,5m), one daycruiser boat (5,75m), and three sailing boats (7m, 8m and 12.5m). The first boat has only one electronic device onboard, this is a sounder but it also has a classical chart because it is compulsory. The daycruiser boat has a combined device: a GPS plus a chartplotter. The three sailing boats have a GPS, the 8m has furthermore a chartplotter and an navigation processor and finally the 12.5 boats has an AIS receiver in addition to the other devices. This panel represents the typical boat owner in a yacht club, so they help us at every step of the study. We discussed with them to understand globally user navigation habits and better known instruments onboard, not only electronic

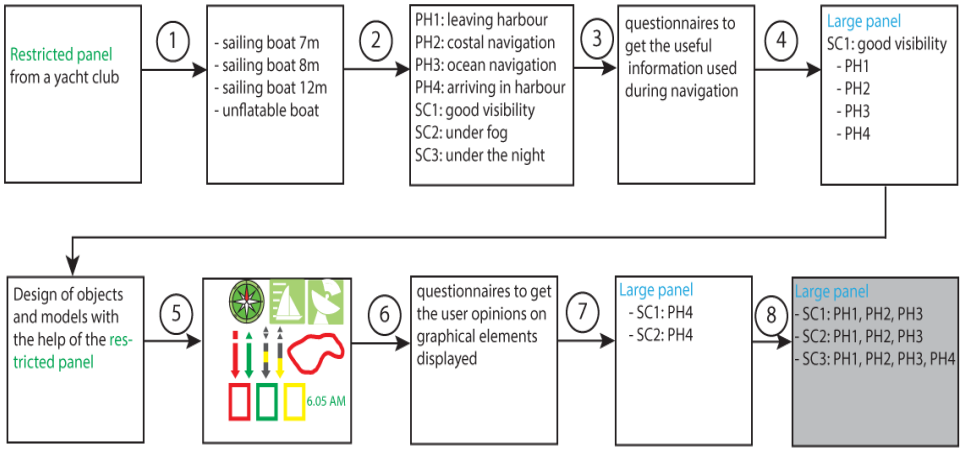


Figure 2.5: Ergonomic step interface

devices but all of them (step 2 in Fig.2.5). The feedback allowed us to build few details queries on the data used onboard, step 3 in Fig.2.5. We sent the queries to a larger panel to get a bigger data set, this is the step 4 in Fig.2.5. Then we designed graphical elements under ergonomic constraints and the help of the restricted panel, this is the step 5 in Fig.2.5. The step 6 (in Fig.2.5) is the building of the queries with the results from step 4 and graphical elements designed in step 5. This is done to test if the graphical interface is correct or not. Samples of the queries were sent to the larger panel to improve the graphical interface, this is the step 7 in Fig. 2.5. Finally we get some feedback on the samples and the rest of queries will be sent latter to complete the set of answers (step 8 in Fig.2.5).

All the steps described in this introduction are more detailed in the following sections.

2.3.2 User study

2.3.2.1 Navigation

Before heading out sailing, the sailors must check the meteorology: the wind, the sea tide and the weather in order to evaluate if there is a risk to sail or not depending on experience and skills. The visibility is very important according to the users, generally (for the panel) they don't sail if the visibility is very bad, such as night conditions and fog. So, with the restricted panel (step 2 in Fig.2.5) we defined three scenarios, which take into account the visibility.

The first one named SC1, represents all sea conditions with a good visibility; an example is shown in Fig.2.6. The second one, named SC2, represents all sea conditions under fog; an example is shown in Fig.2.7. The third one, named SC3, represents all sea conditions under the night. Indeed, the sailor

Figure 2.6: A view of Lorient harbour under a good visibility (SC1)

Figure 2.7: A view of Lorient harbour under a light simulate fog (SC3)

must know the specific maritime rules for night conditions. Buoys and boats emit lights, so the sailor must know their meanings to understand which are the buoys and boats visible on the sea to avoid dangers. With the front, port and starboard lights on a boat, it is also possible to retrieve approximately the boat's bearing under the night with experience and a good knowledge of maritime rules. But sometimes, in the harbour areas, all buoys don't

emit a light. This is the case in Lorient harbour for instance, only the main channel emits lights, so the user as to often check the current heading and the chartplotter.

So, according to the panel, SC2 and SC3 are the most critical conditions apart from storm that's why we chose them. The application must be specific in those conditions to help users, moreover it has to be more useful than the other classical devices onboard.

The next observations are more specifics to the phases of a maritime navigation. The second key point, which was highlighted from the restricted panel (step 2 in Fig.2.5), is the definition of four navigation phases. PH1 is the leaving from an harbour, PH2 is the coastal navigation, PH3 is the ocean navigation and the last one is the arriving to an harbour (PH4). Indeed, the restricted panel needs data, which are not exactly the same in those phases. In conclusion, the first discussions with the restricted panel have highlighted scenarios related to the visibility and navigation phases. We use these observations to focus data to display in the application, moreover a thorough study with queries for a large panel is necessary in order to get a bigger data set to create profiles and the corresponding interfaces.

2.3.2.2 User habits queries

The user habits queries are built to sort the data used in each phase. This group of the queries (step 3 in Fig.2.5) are divided in two parts. The first part is related to the user profile, we want to know who are the users to sort some profiles. The second part is a list of questions on a device and the data available on it. We want to know if the device is used and the data are important or not depending on the navigation phases. The questions presented just below, in box 1, are extracted from queries sent to the large panel (step 4 in Fig.2.5). They are used to sort user experience, boat type and the marine devices onboard.

In the next step of these queries we try to get statistics on the use of the data that depends on the scenarios and the phases. We have discussed with the restricted panel to suggest devices and their data that can be used onboard (step 3 in Fig.2.5). The device list is presented just below: compass, sounder, tide table, GPS, AIS, radar, navigation software or chartplotter, seamarks, landmarks. Every device or semarks/landmarks provides more than one information, so to filter the data really used, we have implemented two steps in the queries. In the first step we ask the user if the device is used or not during the phase (PH1, PH2, PH3, PH4), as it is presented in the box 2. The large panel has just to tick yes or no.

The second step depends on the answer from the previous question (box 2), if it is a negative answer the user is switched to the next device and if it is a positive one, then we ask more details about the use of the device. As an

Which is the CE norm of your boat?
A ◯ B ◯ C ◯ D ◯

Which kind of boat have you got?
jet-ski ◯ inflatable dinghy ◯ motor boat ◯
sailing boat ◯ cruising motor boat ◯ cruising sailing boat ◯

Which kind of navigation do you practice?
coastal navigation ◯ ocean navigation ◯

How many time do you sail?
$<$ 10 per year ◯ 10 $<$ X $<$ 50 per year ◯ $>$ 1 per week ◯

How long have you been practicing pleasure-boating?
$<$ 2 years ◯ 2 $<$ X $<$ 5 years ◯ $>$ 5 years ◯

Which are the device onboard?
Compass ◯ Sounder ◯ tide table ◯
GPS ◯ AIS ◯ RADAR ◯
chartplotter ◯ Navigation software ◯ PC ◯

OTHER

Box 1: Questions about user profile from user habits queries

example, the GPS provides more than one data, a sailor can read the course, speed, position and time from it. But the user may only use one or two data from it, so we have to sort them. We just ask to the sailor in queries, as it is represented in the box 3, if the data is used or not. A little space for a commentary is accessible at the end of the questions in the case of a sailor would add some information.

To get a complete data set, we should provide four queries (four phases) for every scenarios, but during the thesis only the first scenario (SC1) with PH1, PH2, PH3, PH4 was sent to a large panel (step 4 in Fig.2.5) to validate our approach. If the results are concluding, the other queries will be sent to the large panel and we consider that it is the last step of our approach, step 8 in Fig.2.5.

Is the GPS used in this phase?
Yes ◯ No ◯

Box 2: Questions about the used of a device

2.3.2.3 Results

In Table 2.1, we compare the queries results from the four phases: leaving an harbour (PH1), coastal navigation (PH2), ocean navigation (PH3) arriving in an harbour (PH4) under a good visibility (SC1).

Table 2.1: data used under SC1 during PH1, PH2, PH3, PH4

Device	data	PH1	PH2	PH3	PH4
compass	heading	36%	57%	83%	43%
GPS	course over ground	21%	71%	83%	57%
	speed over ground	29%	71%	100%	57%
	position	36%	86%	83%	57%
	time	7%	7%	50%	14%
sounder	water depth	100%	57%	17%	100%
tide table	time	57%	43%	33%	71%
	coefficient	57%	29%	33%	57%
AIS	boat position	7%	0%	83%	0%
	collision risk	7%	0%	83%	0%
radar	object detection	0%	0%	0%	0%
chartplotter	buoys	93%	100%	67%	100%
	danger area	93%	100%	83%	100%
	course over ground	29%	71%	100%	29%
	speed over ground	71%	57%	100%	57%
	position	50%	100%	100%	86%
seamarks / landmarks	buoys	100%	100%	100%	100%
	name	93%	86%	83%	100%
	landmarks	86%	100%	83%	86%

SC1, PH1 To leave an harbour, the sailors need to know the tide state, there are 57% of the panel that check time and coefficient, but 100% of the panel check the sounder. In this phase people want to know if there is enough depth to leave the harbour safely, without any grounding.
The other important data comes from an electronic device, this is the boat speed. 71% of the people look at the boat speed on the chartplotter or the GPS. A lot of sailors need this data because there are some speed limits in harbour channels and they have to stay under it to respect the law.
The last important data is related to the position of the boat, this is not the coordinates but the position compared to the navigation marks (seamarks, landmarks) that is essential. All sailors compare the bridge view with the chartplotter or the navigation software to check if the boat position is correct

Do you look at the Course Over Ground from the GPS?
Yes ◯ No ◯
Is it important for you ?

	1	2	3	4	
Useless	◯	◯	◯	◯	essential

Do you look at the Speed Over Ground from the GPS?
Yes ◯ No ◯
Is it important for you ?

	1	2	3	4	
Useless	◯	◯	◯	◯	essential

Do you look at the POSITION from the GPS?
Yes ◯ No ◯
Is it important for you ?

	1	2	3	4	
Useless	◯	◯	◯	◯	essential

Do you look at the TIME from the GPS or your watch?
Yes ◯ No ◯
Is it important for you ?

	1	2	3	4	
Useless	◯	◯	◯	◯	essential

COMMENTARY

Box 3: Questions about the used data from a device (GPS example)

or not. So the user needs to see the buoys (100%), buoys name (93%) and landmarks (86%) to locate the position of the boat. In the case of a good visibility, all sailors navigate visually, the charplotter is often used to check the name or the position of the next buoys to follow the harbour exit.

SC1, PH2 In coastal navigation some reefs can appear if the sailor is close to the coast, so the sailor must check regularly the water depth with sounder for instance (57%). As in the leaving phase, the sailors use the boat speed (71%) and moreover some of people in the panel use this information to compute the time to arrival. The next important data is the heading from the compass (47%) or course over ground from the GPS (57%). This data is often used to follow a course from one waypoint to the other. The last used data from the GPS is the position, more precisely on a chartplotter or

navigation software. These devices offer an overall view of the boat, buoys, seafront positions and this is the main utility in this phase. Once again the bridge view let the user see all seamarks and landmarks visible on the sea and the user can check if it takes the right way or not.

SC1, PH3 In ocean navigation there are no marks on the sea or just a few, so the sailors navigate with waypoint on charts or navigation software. So they must follow a course or heading to respect the way to follow, the only solution is the use of the compass (83%), or the GPS (100%) to keep a bearing. The speed is also an element to compute time to arrival and get information about the navigation, once again the GPS let the user know about the speed over ground (100%). In this phase, the position from GPS is very important, 83% of the panel tick this information in the query. But it is difficult to decode this result because, it could be the reading of the latitude and longitude or once again just a check on a chartplotter or navigation software to understand where is the boat. Indeed, this kind of devices is able to place the boat position on a chart and it offers the overall view from departure to arrival. New data have appeared in this phase, the data from the AIS (87%). The sailors are attentive to the boats around them in ocean navigation; this device is able to detect boat position and moreover collision risks. Few people from the panel have added the binoculars as a useful device onboard during ocean navigation because it offers a better view to recognize far objects from the boat.

SC1, PH4 This phase is very close to the first one, leaving an harbour. But few people from the large panel have written as an observation that they would like to know the current speed and direction when they arrive in an harbour. The last observation from the panel is about the time. Indeed, sometimes there are some locks or the water depth is limited depending on the tide in harbours, so the user has to check if he his at time or not.

Assessment As the first observation, the application has to be configurable in order to satisfy the user expectations. Data used onboard are not the same depending on the navigation phases even if the scenario is still the same. In the three phases PH1, PH2 and PH4 the computation of the water depth is the most important data, so the application must compute and display this information for the user. The detection of the buoys and their name is very important, afresh, the application has to display information to detect buoys and display their names. In ocean navigation, the AIS is important through the used of alerts to prevent the sailor from collision risks. Finally the speed, the course and maybe the time could be displayed or removed according to the user needs.

We are able to sort the information from queries results on user habits in

three categories aside from the different phases and scenarios thanks to the large panel (step 4 in Fig.2.5). The first one is the state data related to the GPS. The second one is the alert data to prevent from grounding, collision risks and speed limit in restricted area. The last one is all the navigational aids to sail safely such as seamarks, daymarks and some danger areas.

State data The GPS data are similar to a system status (the boat) with the course, speed, position and the time, it can be related to a supervision system. The most important data to display are the course and the speed, in real time. The time is not essential, but should be accessible and the position is not necessary. The latter is often used to correct the course from the next waypoint, so it is useless to display it in a mobile application.

Alert data The user wants some information from the AIS only in case of an ocean navigation because in some channels there are a lot of boats and it could be painful and not practical to see all boats on the application. A speed alert can be also added in a restricted navigation area such as channel or harbour because the user has to follow the speed limitation. Finally, an alert could be displayed in order to prevent from a grounding risk if the user approaches an area, which is close to the draught of his boat. An other solution could be similar to Porathe's application, the danger area are coloured on the display and usually named the NOGO areas.

Navigation data According to the user habits queries results, the sailors want to check their position compared to the seamarks and landmarks even if the visibility is good (scenario under a good visibility). They are used to looking at the sea, detecting seamarks and daymarks and comparing them to an electronic nautical chart. So the application has to display seamarks and daymarks with their name and some graphical objects. We may conclude that the navigation data will be even more important with the other scenarios: sail with fog and night.

Now that we know the user needs and expectations, the next step is the building of the graphical elements. To build an accepted interface from end users, the graphical elements have to be design under few constraints: the maritime context and the user habits. So the next section details the different steps to design the application interface.

2.3.3 Graphical interface

A mobile augmented reality application must be designed in a different way compared to a PC one. First, a mobile device is limited by computation capacities that impacts power consumption and battery life. The second point is the bridge view, we can't display the same information as a marine navigational software because too much information on the display is very tedious

to decode and not efficient enough to understand the boat situation, in a very short time. But we are forced to follow the maritime context rules such as object colors, shapes to make easier the understanding of the application. Third depending on the display technology (see-through, smartphone/tablet) the colours and field of view will be different as a standard LCD display. This involves the use of ergonomic constraints, indeed graphical elements have to be designed and placed in the right place on the screen.

The graphical elements are designing with the contribution of the restricted panel, this is the step 5 in Fig. 2.5, the user habits queries results (step 4 in Fig.2.5), the maritime context and under ergonomic constraints. So we start first with the maritime context in Section 2.3.3.1.

2.3.3.1 Maritime context

The maritime context is linked to the rules on the sea and the meteorological conditions. First, the maritime objects displayed on the mobile device have to be in accordance with the International Association of Lighthouse Authorities (IALA). This involves the use of specific colours and shapes. As a reminder, a 3D representation of what are lateral buoys (harbour), cardinal marks (reefs) and isolate danger marks is visible in Fig.2.8. Secondly, there

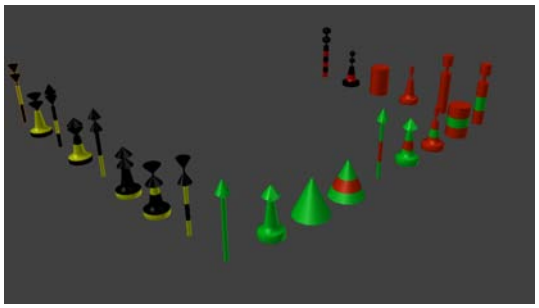


Figure 2.8: A set of 3D buoys in accordance with IALA

are some inconveniences due to the weather. On the ship's bridge, there are no disturbance related to the weather on the applications that run on fixed devices with classical displays onboard. But on a wearable systems and more specifically on a see through system, the weather impacts the visibility of the displays because of the luminosity variations from a shining day to a dark night and even more on boats without any cover bridge. Colours can be very different from a night to a sunny weather, there are colour nuances of blue, grey, white that we have to keep in mind for the interface colours.

It is not enough to take into account the maritime context, the application has to be ergonomic: place of elements, colours, size, shapes. This is important to reduce time to decode on display information and draw only useful data. An explanation of those constraints are presented in the follow-

ing section.

2.3.3.2 Ergonomic constraints

We have considered the ergonomics rules of the human-computer interaction for a supervision system. In order to design a suitable interface, a supervision system has to get only few specific colours, text size, alert shape for instance. The following subsection details these constraints.
Firstly, the colours used in an supervision interface have some rules: the number, the background and the choice of colours placed next to each others. A maximum of 8 different colours is preconized in a supervision system with a weak density displaying [Nielsen, 1994], [Luria et al., 1986]. For example, the best colour combination [Helander et al., 1997], [Sears and Jacko, 2002] is a set of five colours: green, red, white, blue and yellow. But Bruce and Foster [Bruce and Foster, 1982] have proposed a combination of colours that should be avoided, the red and the green colour should not be used with the blue one for instance and moreover, the red, green, yellow should not be next to the blue one. It's also better to avoid or limit the coloured surface on the display because this can overload the display and catch user's attention.
Secondly, the size of characters have also rules, the width might be between 50% to 70% of the height; with a distance of one meter the character's height has to be 6mm for instance and the font should not be sophisticated [Gilmore, 2012].

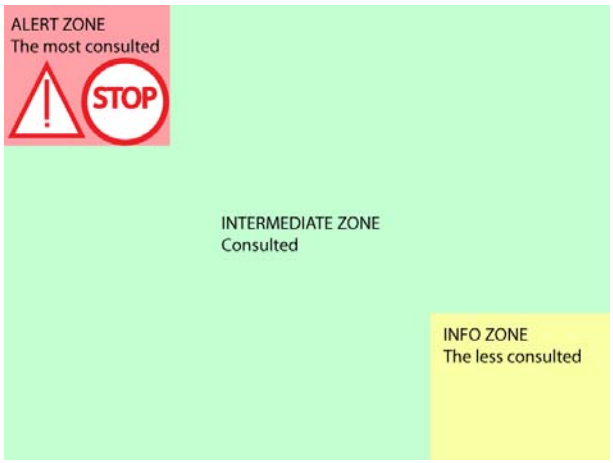


Figure 2.9: The fourth different interest zones on a display

Thirdly, a screen is divided in four interests zones (in Fig.2.9), the first one is the up/left zone, this is the most consulted one. In general alerts are located there with text or symbols. The alerts can be a flashing information between 3 and 5Hz with a message smaller than 12 characters [Ravden, 1989] otherwise there could be some inconveniences during the reading and a risk

to get tired of it. The up/right and bottom/left zones are considered as the intermediate area and the last one is the less read (bottom/right).
Lastly, the density information should not be too important because it could create errors or confusion risks and the user could have some difficulties to decode all the information on the display in a very short time. The interface must have limited information, no overload display, only simple symbols and lowest object animations as possible.

To build the graphical elements, we use the following method. First we take into account the results from the user habits queries (in Section 2.3.2.3), which were filled in by the large panel (step 4 in Fig.2.5) and we list all needed elements. Then, these elements: COG, SOG, danger area, etc. are presented to the restricted panel as a brain storming to find a good graphical representation of the elements (step 5 Fig.2.5). We have to take the restricted panel into consideration to respect the user expectations but all the elements have to be designed carefully with the ergonomic rules listed in Section 2.3.3.2. So the schema of an interface for a specific navigation phase is built with three group of elements, first one is the state data from GPS, second one are the alerts and the last one are navigation elements.

State data In a supervision system, the green colour is preconized because of its high level of legibility, it is also related to positive connotations [Helander et al., 1997] (normal state, everything is alright), so we choose this one. Even if it is not advised with blue [Bruce and Foster, 1982], we keep the green because some of the AR glasses have photocromic, solar glasses or a treatment on it to limit inconvenience in outside use. Some settings on the real AR device under sun, fog and rain for instance will allow to finalize this choice.

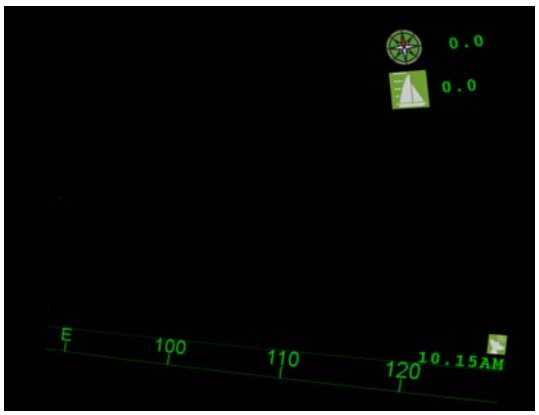


Figure 2.10: State data: COG, SOG, GPS status, Time and User orientation. This Figure seems to be a little deformed because it comes from a screenshot from the application that is running in a Ski mask prototype which has a deform screen.

As shown in Fig.2.10, we decided to place the State data on the top/right corner [Helander et al., 1997], which is a regular seen zone. The boat speed and the course from the GPS are placed as a list, one below the other, this is a choice from the restricted panel. One sailor in the restricted panel, the less experienced, suggest to provide a more pleasant interface so, we added logos next to the text. The final graphical version of these group of information will be selected with the IHM queries answers (step 7 in Fig.2.5).
Then, the next data from the GPS that can be used is the time, this data is not the most important, it is place in the less seen zone, in the bottom/right as it is represented in Fig.2.9.
Finally there is a scale with numbers that represents an orientation in the bottom of the Fig.2.10. In case of AR glasses for instance it represents the head orientation, this data was not listed before because people don't know that they can get it. During the brain storming, most of the restricted panel have asked to get this data on the screen, it can be used to change the course of the boat if the sailor takes an alignment with his view.

Alert data Collision detection alert is displayed with a flashing logo in red and white in case of a collision detection with AIS data. If the boat goes into a restricted area with a limited speed, the application is able to display a flashing logo with a Speed alert. The collision detection and speed limit alerts are visible in Fig.2.11. In the ergonomic interfaces, related to the supervision systems, alerts should not be more than 10% of the display, so we use only logos with limited text to reduce alerts size.

Figure 2.11: Alert logos: Collision risk and speed limit

Navigation data Navigation data are all the navigation aids visible on the sea and the land: buoys, beacons, lighthouse, water tower, church. In accordance with ergonomic rules we created 3D seamarks, arrows and symbols. A 2D or 3D object can be displayed on the buoy's GPS point in accordance

with the IALA representation to show under bad visibility the seamarks position to the user.
Due to the maritime context and technology that will be used to run application (OST displays), there are two specific cases because of the colours. The cardinal and isolate danger marks are painted with the black colour.
An Optical See-Through device can't display the black colour, this is transparency for these devices, therefore it is not visible on those displays. So, the solution adopted is the replacement of the black colour by the grey one. The latter is derived from the black, and should be visible. Even if the colour is not exactly respected, some augmented objects as arrow and topmark let the information understandable.

In order to provide more visibility to the seamarks, it is possible to place an arrow above them. Even if the user has the 3D representation of the seamarks, he can understand its position at few kilometers from him. We decided to add its top mark at the top of the arrow to allow the user to decode easily the information when the object is far from the user's position. Indeed, more than 8% of people have some troubles with colour perception, that is why in IALA buoyage system there are colours and topmarks to describe a buoy.

An example of a set of 3D objects is visible in Fig.2.12. The main colours

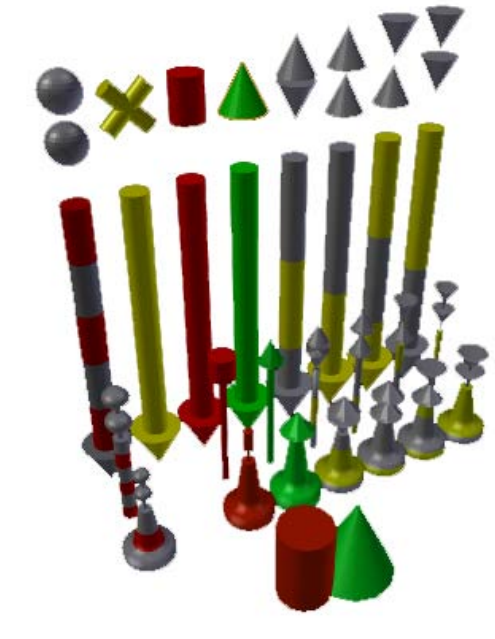


Figure 2.12: Our main 3D buoys, arrows and their top marks for see-through display.

used in the graphical interface to represent navigation aids are red, yellow

and green colour on a blue, white or grey background (sky or sea). Some tests and settings will be realized on the final device to get the best object visibility.

Sometimes, people who don't navigate regularly (this is the large majority of boat owners) can forget the cardinal rules, as it was explained in section 1.3. We want to display the NOGO areas (as danger one), the area next to the cardinal mark will be visible and the sailors will be helped to understand where is exactly the danger.

Now, with the help of the user habits queries (in Section 2.3.2.3) results, we know that most of sailors from the panel use the buoys name in their navigation. This is not the best choice to display it because of ergonomic advices. Indeed, some buoy names can be more than fifteen characters and it takes a lot of place on the display and it could be not efficient enough or negative. However this solution has to be tested with the large panel in the IHM queries, this is the step 7 in Fig.2.5.

The next type of data, which is used in ocean navigation with a good visibility (in Section 2.3.2.3) are the data related to the AIS device. If the system has access to those data, the application can retrieve boats position, speed and course and prevent from a collision risk. As the buoys, an arrow is drawn (orange colour) up to the boat position to make it more visible and the one with a collision risk has a red and flashing symbol.

The last important information which emanate from our first results is the positioning of the danger areas. Actually all the sailors would appreciate to know if their position is safe from groundings, two solutions were emitted: an alert or displaying the edge of the danger. Moreover, the water depth depends on the tide and the draught of the boat. As an example, in Saint Malo harbour, in Saturday 22 of October 2014, the tide coefficient is 80 and some water depths depending on the time are listed next:

- 11:25AM: 4m
- 11:35AM: 3.75m
- 11:45AM: 3.49m

In 10 minutes, the water depth decreases $25cm$ and in 20 minutes about $50cm$, this is enough for a boat to run aground at 11:45AM and not 20 minutes before. So, this information is significantly every 10 minutes, according to the SHOM, depending on the tide coefficient and the geographic zone.

Currently Porathe is able to retrieve all danger area in his application, he chose to colour all the area. It may be the best solution to understand easily where are the danger area and their size. But in ergonomics, a weak density displaying is recommended. Moreover, it may be not adapted in augmented reality due to the small field of view (between 14 degrees to 50 degrees in diagonal). One of the compromise are the danger area edges, the area are

visible and coloured surface limited but the user doesn't know if is inside or outside the danger area. After the brain storming with the restricted panel, we decided to keep only the edge of danger area and this solution is test in the IHM queries with the large panel. This is the step 7 in Fig.2.5.

In the next section we have submitted ideas to the restricted panel to get the first feedback on the graphical elements.

2.3.3.3 IHM queries

In order to evaluate the graphical elements designed with the restricted panel's collaboration, we use some IHM queries (step 6). The building of the queries and the collection of answers, first, takes a lot of time and second, we want to validate our method: models with questions. So we decided to take into account two cases, not the 3 scenarios and the 4 phases. The first case is the phase 4, an arriving to the Lorient harbour with a good visibility (SC1), visible in Fig.2.6. The second one is the same phase with fog, SC2, visible in Fig.2.7.

The queries have two parts, the first part is used to get the user profile and

Figure 2.13: Queries on SC2, PH4 Speed over Ground

the second one to get opinions about the proposed graphical elements. So, the first part is the same than in Section 2.3.2.2, it provides user profiles. The second part is based on the pictures in Fig.2.6, 2.7. On Optical See-Trough glasses frames are seen as transparency so, the graphical elements are also drawn with transparency in the templates. For every objects, we let the user decide if the objects are suitable or not (colour and shape proposed), also if it is legible and understandable. An example of a template for the Speed Over Ground data from GPS is shown just below, in Fig.2.13. This example

is an arriving in Lorient harbour (PH4), under the fog (SC1).

The questions on the graphical elements, about the SOG, are presented in box 4. At the end of the three questions, we ask the user if there are two

Is the Speed over Ground useful with this representation ?
YES ◯ NO ◯

Is the Speed over Ground legible in this case ?

	1	2	3	4	
Illegible	◯	◯	◯	◯	very legible

Is the Speed over Ground understandable in this case ?

	1	2	3	4	
incomprehensible	◯	◯	◯	◯	very understandable

Is there too much information on the display?
YES ◯ NO ◯

COMMENTARY

Box 4: Questions on the graphical representation of the speed over ground

much data or not on the display. Indeed, after one object on the template, we add the next object and so on, until the last one. This is done to evaluate the limit of objects to draw, in order to estimate the cognitive load. A template of an overload picture is visible in Fig. 2.14.
Finally we place a text box to let the user expressing their observations and opinions about the graphical element and the object quantity. This allows us to determine if the graphical elements are useful.

2.3.3.4 Results

In Table 2.2, we show the results of IHM queries for one phase: arriving in an harbour (PH4) under two conditions: a good visibility (SC1) and under fog (SC2). For one information, there are 3 answers (Box 4 in Section 2.3.3.3). The first line is all positive answers, in percentage, if the information is useful. The second line is about the legibility of the information, this is the average between 1 and 4 which is written. The last line is about understandable, this is the same representation than the legibility answers. Sometimes, the average is between 3 and 4 for instance, so it is written 3/4 in Table 2.2.

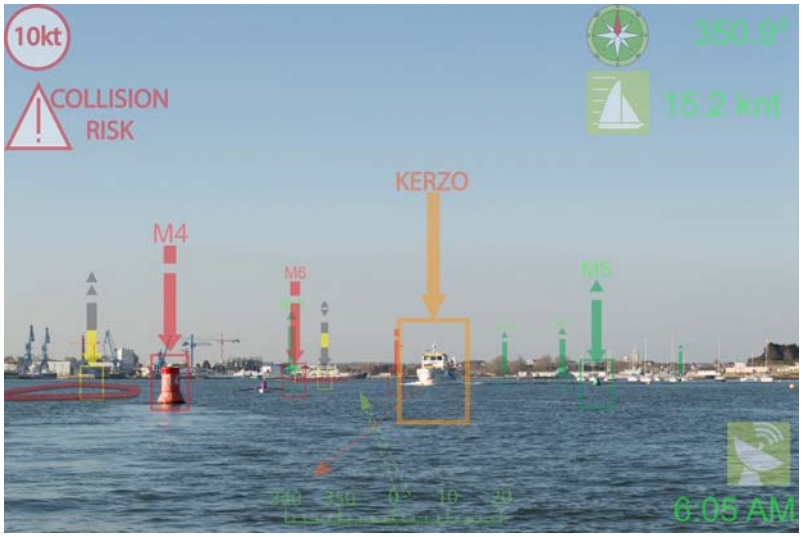


Figure 2.14: A view of Lorient harbour with augmented reality objects

SC1, PH4 The first observation is about the data, which are not enough legible. The speed over ground, the buoy names and the device orientation have only the number 2 on a scale from 1 to 4. So, these graphical elements proposed by the restricted panel in the step 5 in Fig.2.5 are not enough legible and must have more visible colours.
The second observation is about the data which are not enough understandable. The danger area and the buoys (see Fig.2.14) have only the number 2. The displaying of the danger area edge is not the good one, the form has to be changed to be more understandable.
Finally if we want the best graphical elements, few objects could be improved. Every objects could be improved to get the number 4 in the legible and understandable scales, but currently, we keep all objects between 3 and 4.

The second kind of results are the observations written by the large panel. Firstly, the most experienced sailors don't need logos, just the values with unities, the speed for instance can just be displayed as follow: KNT 15.2, SOG 15.2 or 15.2 knt.
Secondly, there are too much graphical elements to detect a seamarks position. Two objects are enough such as the buoy names and the arrows which are the best one according to the users.
Thirdly, most of the panel think that the contrast of the green colour is not enough strong and could be improved.
Fourthly, sailors have the habit to read on their displays the water depth from a sounder and few from the panel would like to read on the AR device the value in real time without any logo.
Fifthly, some sailors don't need the GPS status because all the displayed

Table 2.2: IHM queries answers during PH4 under SC1 and SC2

Data	property	SC1	SC2
COG	useful	36%	82%
	legible	3	4
	understandable	3	4
SOG	useful	86%	73%
	legible	2/3	3
	understandable	3/4	4
time	useful	36%	55%
	legible	3	2
	understandable	3	3
GPS status	useful	43%	43%
	legible	3/4	3
	understandable	3/4	3/4
rectangle	useful	71%	82%
	legible	3	3
	understandable	3	3
arrow	useful	86%	100%
	legible	4	4
	understandable	4	4
name	useful	50%	73%
	legible	2	2
	understandable	3	3
danger area	useful	50%	45%
	legible	3/4	4
	understandable	2	3
orientation	useful	29%	36%
	legible	2	2
	understandable	2	2
boat position (AIS)	useful	29%	82%
	legible	4	3
	understandable	4	4
collision risk (AIS)	useful	57%	82%
	legible	4	4
	understandable	4	4

data from the GPS device will be set at zero if there are no reception. Finally, course over ground logo was criticized, this is not enough understandable, and it has to be improved.

SC2, PH4 The results on the colours with the fog (SC2) are very closed compared to the template with the good visibility (SC1). Depending on the nuance of white and green, the graphical elements are more or less legible. This is the case for the SOG, danger area, GPS status and boat position with the AIS for instance. According to the panel, in this scenario (SC2), people need more information than the first scenario (SC1). It seems that rectangle are less important than the arrow and this one can be used in addition to the buoy name which is more used in this scenario than the other one.

As the previous part, panel wrote some observations.
First, there is an observation on NOGO area, some of the panel would like

to see all the danger area, as a coloured surface, not only the edge.
Second, the orientation information is a good one but it will be interesting to display it on the top of the display and change the shape or the colour.
Third, the blinking alert logo for collision risk is a good idea but it could be a nuisance if the rectangle around the boat is blinking to.
Finally, one person written that it will be useful to select the information to display depending on the dangers: collision risk with a boat, grounding risk, etc.

Assessment There are some interesting and expected results on the display objects in the templates.
First, there are a little difference with the green colour on the white and blue backgrounds. The SOG and COG information have only the value 2 with a blue background whereas 3 with the white backgrounds (Table 2.2). So, the contrast and colour have to be test on the real conditions, i.e with a see-through device to confirm the user opinions.
Second, even though a fully coloured surface is not advise in ergonomic, the users want this representation, the same as Porathe, to prevent from NOGO area.
Some data are not enough explicit; the orientation, buoys name, GPS status and time are not understandable (Table 2.2). We need to change their shapes and delete GPS status.

The most part of the second panel think that the display has too much information if more than seven data are displayed. So, the application should offer the choice to add and remove in real time some data depending on the phase, scenario or danger around the boat to satisfy the user expectations and limit cognitive load issue.

The panel wants to see the buoys but depending on the visibility, the close buoys don't need to be displayed with a good visibility whereas with fog it is necessary. Conversely, with a good visibility, the panel wants to detect buoys a little far from them. So, it may be useful to let the user select this information in the application. This functionality should be implemented as a scale with the minimum and maximum distance to detect buoys on the mobile AR display. It seems that most of the people would like to configure elements to display on the application and the distance to display navigation aids.

2.3.4 Conclusion

To understand user habits and build a graphical interface, we use the approach presented in Fig.2.15. There were two objectives, get all the most important data used onboard and design graphical elements to display in the application depending on phases and scenarios.
The results of the first objective come from the step 4 in Fig.2.15. At this

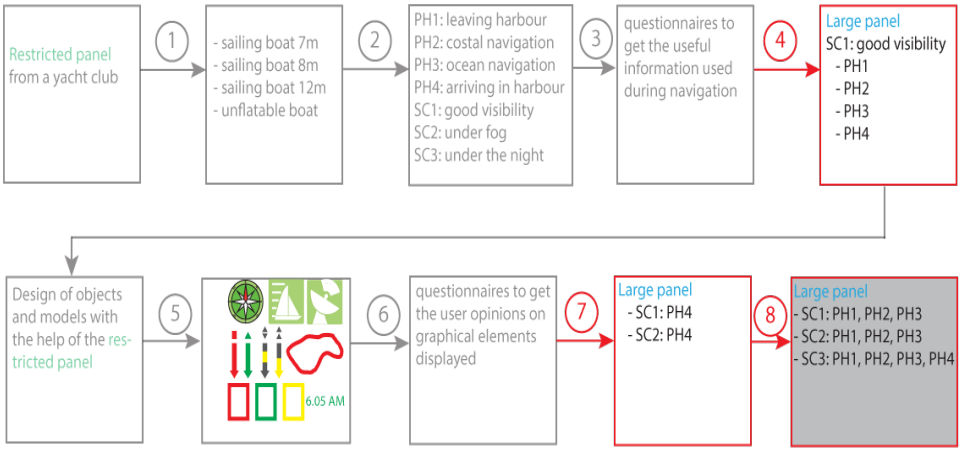


Figure 2.15: Ergonomic step interface

step, we know all the main data used onboard when a sailor navigates with a good visibility, thanks to the large panel.
The results of the second objective come from the step 7 in Fig.2.15. We are able to provide a graphical interface to the sailors in order to improve safety in a good visibility or fog navigation.

As a first analysis, the time was selected as an interesting data but it is only related to the tide in most cases. So if the depth water is shown in real time, the time data could be dropped off. The other key point related to the depth water is the NOGO areas, they have to be fully coloured as it is expected by users even if it diverges from ergonomic rules.
Then, some objects and colours have to be a little different from the templates as orientation. The green colour is pointed out under good visibility but some tests with the real AR display, under real conditions should be realized to validate or not this choice. Indeed, it depends on the technology used, there are some solar or photocromic glasses on AR device and the results will not be the same. Normally the green colour is the best one, this was the one used on the old monochrome CRT display because the eye is more sensible to the green colour wave length.
Thirdly, some objects have to be changed such as buoys name, rectangle around the buoys and the state data logos. Redundancy data is not the right solution, rectangles have to be removed, buoy names have to be improved and logos should be displayed as an option in the final graphical interface. Moreover the data to display should be presented in a configuration list to let the user make some profiles depending on the experience, boat and weather. So, a list of data to select should be implemented in the application to satisfy all kind of sailors from landlubber to the old salt.

In conclusion, we are satisfied with the results from the approach presented in Fig.2.15. It is based on queries about user habits and graphical

elements made with a restricted panel and sent to a larger panel to get more opinions. We think this is an effective approach because it involves the end users.

We met recreational boats owner from the ANPLF (Association Nautique de Port La Foret) [1], they were very enthusiastic and active to contribute to the study. They represent the large panel in our user study. We have interesting results, they allow us to improve our IHM to satisfy the sailors needs. In order to finalize our study, we have to submit the others queries to the large panel, this is the last step in Fig.2.5. The last are the IHM queries under fog and night conditions in the four phases: leaving an harbour, coastal navigation, ocean navigation and arriving in an harbour.
Now that we know the shapes and coloured of the objects to display in the application, we have to analyze which are the software tasks and hardware parts to provide the navigational assistance application in a mobile augmented reality device. This part is detailed in the following section.

2.4 Needs analysis

2.4.1 Introduction

The data and graphical elements chose by the users have highlighted some software and hardware needs. Indeed, the sailors want to read information from maritime devices such as the GPS, AIS and radar. So, the mobile device has to communicate with them and decode data, to display the useful information as graphical one. Then, the graphical elements displayed in the navigational application assistance are 2D, 3D and text objects. The system has geolocated the user, get the user's field of view and manage the graphical data to display them in the right place at the right time.

2.4.2 Application needs

The application requires multiple software tasks to position and display the graphical objects in order to help the sailor in navigation. Four kinds of tasks can be used to meet application needs: a set of decoding, positioning, coordinate transformation and graphical tasks.
Firstly, the implementation of the decoding tasks is necessary to retrieve and filter the right information from electronic devices such as the GPS, AIS and radar. Once the data to display are stored, another task has to transform the coordinate from the GPS and navigation aids to another system. Then, all information have to be displayed in the user's field of view, so a specific task is in charge to compute it from sensors. Finally the last set of tasks is the graphical one, the tasks have to load/remove, place and orientate objects

[1] http://www.anplf-asso.fr/

in the display in order to show the state, alert and navigational data to the user.

2.4.2.1 Decoding

AIS, radar, GPS and maritime devices are encoded under the NMEA0183 or NMEA2000 norms [2]. So it is essential that the application can decode sentences from those devices to get the right data and display the corresponding graphical objects. A specific decoding task will be implemented to detect potential collisions with the help of AIS and GPS data.

As an example, the NMEA sentence in box 5 comes from an GPS:

```
$GPGGA,092750.000,5321.6802,N,00630.3372,W,1,8,1.03,61.7,M,55.2,M,,*76
```

A decoder is require to extract information from the NMEA sentences and check if there are some errors or not by computing the checksum. As an example, the NMEA sentence in box 6 comes from an AIS:

```
!AIVDM,1,1,,B,181:Kjh01ewHFRPDK1s3IRcn06sd,0*08
```

The data are not directly workable, a decoder is needed to get the boat ID, position, COG, SOG and so on data on it.

2.4.2.2 Coordinate transformation

In geolocation application that generally uses data from GPS, it's easier to manipulate another format than the classical Degree Minute Seconds (DMS) or degrees. The boat position is extract from a GPS with latitude and longitude, like all the navigation aids data, which come from nautical charts; there are also stored with a latitude and a longitude. The use of metric unity is advised and there are two ways to process the geographic coordinates. The first one is a projection in a local 3D coordinate system such as the NED (North East Down) which is used for drone for instance and ENU (East North Up) that can be used for boats. The second one is a 2D projection on a flat surface such as UTM (Universal Transfer Mercator) which is used to write some nautical charts. This is a conform projection that keeps angles and distances error are very small in a surface of few kilometers. Both methods use the WGS84 [Sandwell, 2002] geodetic system that specifies references and get angular position on the earth.

2.4.2.3 Positioning

The positioning task has to retrieve the user orientation because we have to display in the user's field of view the navigational aids data in the mobile

[2] http://www.nmea.org/

optical see-through display. This leads the software tasks to compute pose estimation at 30Hz, which is the minimum frequency to catch user head movements [Najafi et al., 2003]. This is the constraint to be able to display the 3D objects for instance on the right place in the user's field of view as the real one. So a pose estimation is an orientation in 3 axis: roll, pitch and yaw and if the pose estimation is too long some lags will appear and this is not pleasant for the user especially in an AR application that superimposes some data on the real world view.

2.4.2.4 Graphical

According to the user needs, the application has to display state data (COG, SOG, time) with text. But a category of users would like to see some 2D coloured logos to get a more pleasant interface. The alert data are also 2D logos. This is different for the navigation data, which are the buoys and the NOGO areas. The buoys could be displayed as 2D objects, like alert data. The size of these objects can be changed easily to display a far object but this is not the same for the NOGO areas. This is like a 3D object, they have to be placed in a perspective view. The user progresses in a real 3D environment, the real objects are visible in a perspective view. So the embedded system must have enough graphical capacities to compute text, 2D and render 3D objects to draw them on the display.

The next graphical key point is the objects handling. The user position

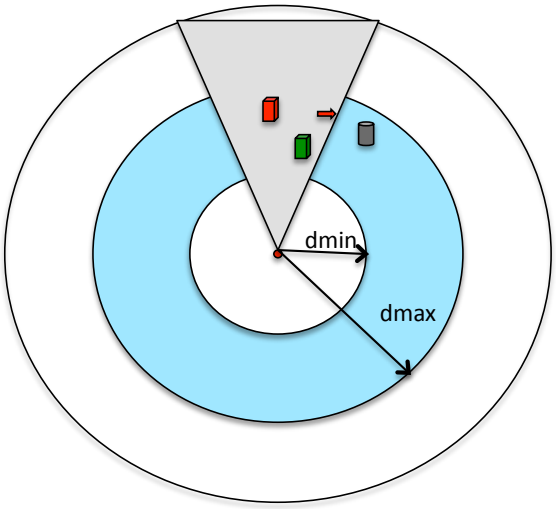


Figure 2.16: Objects and Field of View.

and orientation change, also the navigation data, so a graphical task has to load and unload objects when they go inside or outside the user's field of view (Fig.2.16).

The frame rate is constrained first by the user head movements (30Hz 2.4.2.3) and the persistence of vision which is closed to 25Hz. So, to get a more pleasant graphical application, we want that all tasks work between 25 and 30 frames per second.
To improve the perspective view in a 3D application, the shadows or reflection on objects can be added but it is not easy to realized these effects. Some tests should be done to evaluate necessity and efficacy of those 3D methods.

2.4.3 Hardware needs

The hardware system that runs the applications is manyfold. It must provides sensors to feed the processing of orientation and position, it must offer a system on chip to decode data and run compute intensive tasks such as graphical tasks and finally it must come with an augmented reality see-through display.

The maritime context decrees the use of sensors aside from the weather conditions (luminosity). It is also constrained by a wide open space, user and boat movements. Indeed a swell period vary between 0.05Hz and 0.1Hz [Massel, 1996] and the frequency required to capture users head movement is 30Hz. The mobile system needs another sensor or hardware solution to get the user coordinates.

Then one or more chips are required to load/remove graphical objects and run the graphical pipeline: 3D/2D projection, Z-buffering, lighting and frame buffer updates. All the specific hardware parts are detailed as follow: Orientation (Secction 2.4.3.1), Geolocation (Section 2.4.3.2), Computing (Section 2.4.3.3) and the AR display (Section 2.4.3.4).

2.4.3.1 Orientation

To find solution to acquire user orientation we have to take into account of the boat environment: the weather conditions, the user, the boat movements and the depth field of the scene. We did not choose a camera-based approach, which is not adapted to our context for several reasons.
First a camera is disturbed by brightness, which can be very intense on the sea. In case of direct lighting on the lens, the picture cannot be directly exploitable because of the glare. This problem is due to the direct or reflected sunlight on the lens. In addition to the glare, lens flare is caused by a reflection on the lens due to a little angle between the light source and the lens. Some rings or circles may appear on the picture and induce image processing to correct it.
Second, in night conditions we could use an infrared camera[Hugues et al., 2010]but unfortunately we would still require a normal camera (like a Smartphone camera) as well that increases the price of the system and also image processing steps to decode frames extracted from the infrared camera.

Third, in case of fog, some additional image processing has to be performed to eliminate noise [LYYN, ncom], but it requires large computations capacities and far objects might not be perceived anyway.
Finally, in normal condition (with no disturbance), some computing must be done on frames such as scale because of the camera FoV (Filed of View), which is different from a real size scene. The use of the camera induces an image processing rate at 30Hz and the data flow vary with the camera resolution. In maritime navigation, this is a wide open space, so the camera must acquire small elements in far distance, the resolution has to be about 720p to keep those information but it requires a lot of computing. Indeed, a SVGA resolution has 800*600 pixels coded with 24 bits (in case of RGB) let 1440kB to process every 33ms whereas a 720p resolution are 2765kB, about the double. Finally the considered objects are far by definition and the accuracy of the positioning is not critical. So, the incremental cost of image processing compared to the results and our application context don't justify the use of camera in our system.

This solution is also not adapted to our small footprint and low power objectives. We conclude that the Microelectromechanical systems (MEMs), in addition with a filter, is one of the best solution for pose estimation in marine navigation because of all the previous inconveniences.

2.4.3.2 Geolocation

To get the user position as a geographic coordinate, there are two solutions: an embedded or a distant solution. The first one is the implementation of a GPS chip in the mobile augmented reality system. This solution lets the system to be independent of one another. The second solution is the use of wireless connections to access data from a remote GPS. It is possible to embed a WiFi or Bluetooth chip in the system and connect to a smartphone, a PC and so on that have a GPS chip, on the boat.

2.4.3.3 Computing

There are few specific tasks to implement the application: orientation, geolocation and graphical tasks.
The first one is related to the orientation computing, to retrieve the user's field of view. The MEMs are the solution to get the head orientation, so it involves orientation computation with specific filters. The system would be more efficient with a Digital Signal Processor onboard or a Floating Point Unit to compute the filters with a sampling frequency up to 30Hz.
The second is involved by the geolocation task. Indeed, the mobile device has to get enough computing capacities to run both orientation computation and geolocation/decoding tasks.
Finally, the most complex tasks to run are the graphical ones. Indeed to

efficiently compute graphical elements (text, 2D/3D objects) a GPU coprocessor is required. It also frees the processor that can handle tasks such as orientation computation, wireless communications and display control. Some graphical APIs are used such as OpenGL ES to compute all 3D specific tasks such as 3D/2D projection, Z-buffering, lighting, rasterization and frame buffer update.

It is very important to keep in mind the portability of the system, indeed the system should be easy to be implemented on few kind of devices. So we have to find a solution which is fully adopted, like a standard.

2.4.3.4 Displays

The mobile system must provide a brightness that is efficient enough to handle luminosity variations from a shining day (maybe solar-glass) to a dark night. Two kinds of AR devices are available, the smartphones/tablets and AR glasses. The two main characteristics of the displays are the resolution and FoV.
First, higher is the resolution and better are the details on the display. So a smartphone/tablet is currently better than AR glasses. Indeed a tablet display have a full HD display or more whereas mobile AR glasses are limited to a Wide VGA (800*600) or 1280x720 resolution with a pocket computer [3]. However our main solution is based on AR glasses since they provide a hand free solution, which is required by sailors, moreover we can expect better resolution in the future.
Second, the FoV, for navigation in a wide open space, should be as large as possible to maximize the space where meaningful data can be displayed and superimposed on the user vision. We have considered two available AR glasses with a FoV of 24 degrees [Optinvent, 2014] and 50 degrees [Technologies, 2013] in diagonal. It means a very important difference illustrated in Fig.2.17 with a real picture. Google glass are the most popular AR glasses but they don't fit with our application requirements, indeed we can easily understand that a FoV of 14 degrees, a low luminosity is not enough in our maritime context application and the worst point is the display position. The Google Glass display is not place in the user's field of view, he has to look at the display to see something. Some arrows can be added to indicate to the user some points of interests are visible out of the FoV in the case of a collision risk for instance. But a too small FoV means a lot of head motions to capture data in a 180 open space and so will be uncomfortable in practice.

[3] https://www.spaceglasses.com/

Figure 2.17: Difference between two AR glasses Fields of View.

2.4.4 Conclusion

Our ambition is to design a useful and easy to use application for a simplified and safe navigation. We have to display graphical objects in text, 2D en 3D in the user's field of view. This involves the implementation of tasks to decode data from NMEA sentences, compute collision detection, pose estimation, transform the coordinate system and compute graphical tasks. All these software tasks are implemented in a mobile system which must embed a CPU, GPU, a GPS or wireless connectivities to communicate with NMEA devices. In order to retrieve user's field of view, the use of MEMs is preconized in our maritime context. Finally, the AR device is an Optical See-Through system with enough luminosity to see graphical elements on a display around 50° in diagonal.

2.5 Conclusion

In this chapter there are two parts, a study to get a useful interface for our application and a need analysis to find the right technology to embed the application. We used some queries to understand user habits and design a graphical interface for the navigational assistance application with the help of end users. An example of our results is visible in Fig.2.18a and 2.18b, we know which data, their shapes and colours to display when a sailor is arriving

in an harbour under the fog. At this step, we know that the application is correctly designed, this interface can help recreational boats owner in their navigation.

(a) IHM before queries for an arriving in the harbour under fog

(b) IHM after queries for an arriving in the harbour under fog

Few profiles have been highlighted thanks to the profile part in the queries. The next list represents the most important profiles that can be defined as default profile in the application:

- 1: Coastal navigation, bad visibility.
- 2: Ocean navigation.
- 3: Coastal navigation, good visibility.
- 3: Regatta.
- 4: Professional.

Some settings must be provided to the user to let them refined their data to be displayed because there are some different needs between a landlubber and an old salt.

Then we made a need analysis to list all the software tasks deduced from the user needs, the main tasks are represented in the Fig.2.19. In addition to the software tasks, we study which kind of chips have to be used to get a complete mobile augmented reality system usable in maritime context. The main parts are visible in Fig.2.19. The prototype will embed an Operating System, so a RAM, SDcard and buttons will also be implemented.

We distinguish from SeaNav or SmartChartAIS, our IHM is very different and the augmented reality device chosen is ergonomic and conform to the users expectations: mobility, displaying navigational aids in the user's field of view.

Now, all the hardware and software tasks are known thanks to the analysis. The next step is the design and the achievement of the prototype:

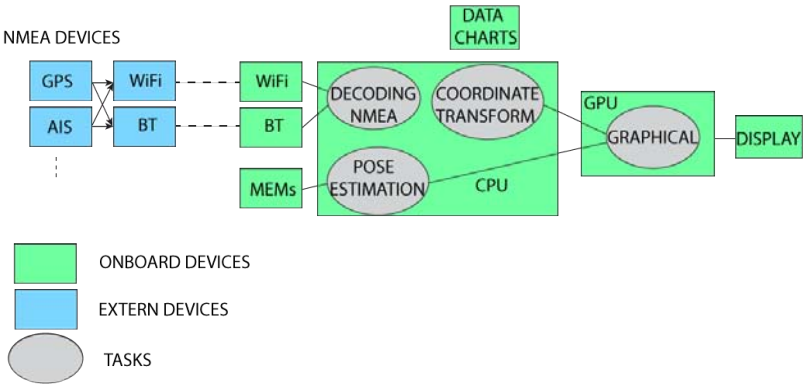


Figure 2.19: Overview of tasks and chips for the prototype.

hardware platform and the application with these objectives: performances, low power, portability and wearable.

Chapter Thirst

Prototype

Contents

3.1 Introduction

At the end of 2011, the available AR glasses were only displays connected to PC by VGA cable for instance, there weren't any fully embedded devices on the market. So, to provide a complete solution, we realized a prototype with an hardware architecture that is able to run the application tasks listed in the previous chapter.
In this chapter, we first present the state of the art in Section 3.2, which sums up available hardware parts at the beginning of the thesis. Then, we describe all the hardware parts and devices selected to design the prototype that runs the 3D application, in Section 3.3. The Section 3.4 details the role of the software tasks and how they work together. Today more and more mobile AR devices (glasses, Smartphones, tablets) are available on the market, so we decided to port the application on main stream market devices, which run Android OS. This part is explained in Section 3.5. Finally, we conclude on our prototype named Marine Mobile Augmented Reality System (MMARS) that runs our 3D application named Marine Augmented Reality Navigational Assistance Application (MARNAA). Both, prototype and application, are the second contribution of this thesis.

3.2 State of the Art

3.2.1 AR Display

In our application context, we want to let the user looking at the real world and display information without masking it. So, Virtual Reality devices are not suitable, we focus on see-through devices only. Three kinds of AR displays exist, the first one is a head mounted display with optical see through, it's usually an AMOLED or special LCD screen combined with an optical device to project virtual objects in the the user's field of view as it is presented in Fig.3.1. The second one is also a head mounted display but with

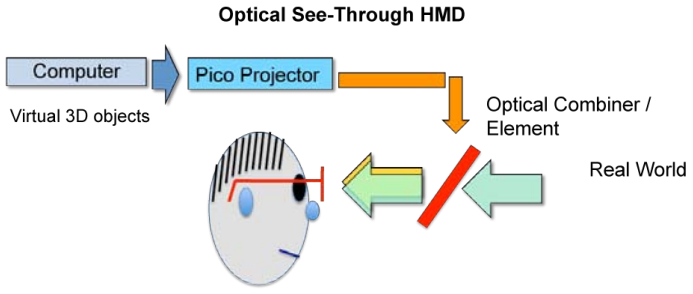


Figure 3.1: See-through glasses technology

a video see-through (VST), a camera is used to capture the real world and

overlay objects on it. The last one is a handled display with a flat panel LCD or LED, the device displays frames from a camera like the previous one (it could be a smart-phone, a tablet and so on). Handled display with a flat panel LCD (or AMOLED) display in addition of a camera is not comfortable because there is a parallax error due to camera being mounted away from true eye location.
We want our display to meet following constraints:

- The user has to see the real world.
- The OST display has to be in the user's field of view.
- The device must has enough luminosity for outdoor applications.
- We want the biggest field of view available.
- We want a higher display's resolution available.

One of the major problem on AR display is the weak luminosity. A majority of devices couldn't be used in the outdoor and marine context because the luminosity is not sufficient. Another drawback is the resolution of the screen, high resolution is needed to see a lot of details in a little screen just in front of eyes, a resolution under VGA is not enough for little objects (far buoys) or texts (buoy names). Also, field of human eyes is 180 degrees but displays have a field of view about 30 or 40 degrees for the best monocular systems. Binocular systems would be perfect to enlarge the usable field of view but we observed that current solutions are not mature and robust enough to be used in real conditions.

So, few manufacturers propose real OST devices with more or less the constraints listed before: Laster Techologies [Technologies, 2013], Optinvent [Optinvent, 2014], Lumus [LUMUS, 2011] and Vuzix [VUZIX, 2011]. A representation of the devices or prototypes is visible in Fig.3.2d, 3.2a, 3.2c, 3.2b. At the beginning of the thesis HMD systems were usually based on a wired or wireless connection with Smartphones or laptop that compute and provide data to be displayed.

Table 3.1 gives a list of all main specifications of the glasses or prototypes accessible at the end of 2011. There are no devices with all necessary embedded chips (CPU, GPU, IMU, GPS) to run AR application, only the Vuzix Star 1200 has got MEMS and Laster technologies a GPS embedded. The other main drawback of these solutions in our context is the lack of luminosity, the brightness was obviously under 3000 candela/m^2, which is currently the average value. Most of the glasses were prototypes, this is why they were very expensive, price was about few thousands Euros against few hundred Euros today. This evolution confirms our initial intuition.

Vuzix, Optinvent, Lumus and others propose now intelligent hands-free displays to be connected or not to Smartphones like Google Glasses, but

Figure 3.2: Pictures of OST AR glasses (end of 2011)

(a) Prototype from Optinvent

(b) Vuzix Star 1200

(c) Prototype from Lumus

(d) Laster MG1

Table 3.1: Overview of the available See-Trough displays (end 2011)

Manufacturer	Optinvent	Vuzix	Lumus	Laster Technologies
display	Monocular	Binocular	Binocular	Monocular
FoV	24°	31°	27.5°	50°
Resolution	640*480	852*480	640*480	800*600
GPS	-	-	yes	-
MEMs	-	9DOF	-	-
connection	HDMI	VGA	VGA	VGA
price	10000€	4200€	-	8000€

it wasn't the case at this period (end of 2011). Some of the most known manufacturers of OST AR glasses, which sell SDK and glasses, are visible in Fig.3.3a, 3.3b, 3.3a, 3.3c, 3.3d, 3.3e, 3.3f, 3.3g.

In Table 3.2 we only list the FOV and resolution of the available glasses, as a SDK, at the end of 2014. A comparison with the prototypes and models in 2011 (Table 3.1) is possible. The characteristics glasses have not been very improved, only display resolution. The main improvements for three years are linked to the size of OST system that now lets place to the electronics and sensors to run AR applications without the use of a distant device.

There are not much displays available in 2011 and electronic in glasses was not sufficient to compute user orientation. The first main electronic part is the one required to compute orientation. We concluded in the Chapter 2 that the orientation must be compute an IMU with MEMs because

Figure 3.3: Pictures of OST AR glasses (end of 2014)

(a) Optinvent ORA

(b) Vuzix Star 1200XLD

(c) Lumus DK-40

(d) Laster See thru

(e) Google Glass

(f) Meta

(g) Epson Moverio B-200

of their low footprint and there are not disturb by the weather (fog, night for instance). Even if there is one in the Vuzix Star 12000, data are only accessible by USB cable and time to log, compute orientation and used it in the application leads to lag. The second part required to run application is a SOC, as it was highlighted in the previous chapter, afresh we have to include it on the glasses.
Considering weather conditions (light reflection, fog, humidity) it turned out

Table 3.2: Overview main stream OST displays

Manufacturer	Model	FOV	resolution
Optinvent	ORA	24°	640x480
Laster Technologies	See thru	25°	800x600
Google	Google Glass	14°	640x480
Vuzix	STAR 1200XLD	35°	852x480
Epson	Moverio BT-200	23°	960x540
Meta	META 1	35°	960x540
Lumus	DK-40	25°	800x600

that Laster Technologies was providing the better solution for our outdoor conditions in the first year of the thesis. Indeed, it has 50° as a FOV in diagonal with a VGA resolution and this model can be equipped with an AMOLED or LCOS display. The LCOS technology is better for luminosity, up to 5000 candela/m^2 whereas few hundreds with AMOLED and the branch has got enough place to put an electronic board with our hardware architecture.

Once the display are listed, the next step is the overview of head tracking components. In the previous chapter, we chose the IMU as the solution, on the market, to capture head movements for the application. The next section sum up the most interesting ones.

3.2.2 Head tracking

In our outdoor context, efficient markers-based solutions [Herout et al., 2012] can't obviously be used. Relevant markerless solutions based on image processing [Comport et al., 2006, Karlekar et al., 2010] could be proposed instead, the solution described in [Yang and Cheng, 2012] is for instance promising on mobile platform. However it is not applicable in our outdoor conditions and other issues such as the important distance of targets and boat motions disqualify this approach.
For outdoor positioning and orientation, some solutions based on GPS data are proposed on Android and Apple devices but are based on handled devices and are not accurate enough to place an object at the right place.

A solution, based on a differential GPS was proposed in [Feiner et al., 1997] with inclinometer and magnetometer to display text information on campus buildings. The authors noticed the acceptable inaccuracy with regards to the application requirements, but they also indicated issues with visibility in sunny conditions and difficulties with the inertial sensors. Sensors and related signal processing have been improved since, but visibility and integration still need to be solved.

As it was provided in the previous chapter, a possible solution must be based on an embedded system able to track head movements with MicroElec-

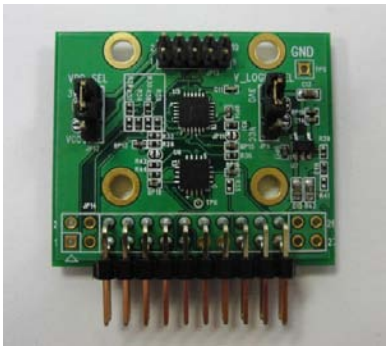
troMechanical systems technology, which provides integrated and low cost Inertial Movement Unit (IMU) combined or not with a GPS. MEMs can track until 9DOF according to the three axis of the accelerometer (gravity and linear acceleration), magnetometer (earth magnetic field) and gyroscope (angular velocities). This kind of chips are currently available for robotic and Smartphones/tablets applications.
Our constraints are listed as follow:

- 9 Degree Of Freedom (9DOF), 3 axis for each components (accelerometer, magnetometer and gyroscope).
- The lowest footprint.
- An interface easy to connect with the main processor.

The target characteristics are listed as follow:

- Accelerometer: scale $8g$, resolution 16 bits, consumption 500μA.
- Magnetometer: scale 1000μT, resolution 13 bits, consumption 500μA.
- Gyroscope: scale $2000°/s$, resolution 16 bits, consumption 5mA.

There are few manufacturers of MEMs in the world such as Invensense, ST Microelectronics and Analog devices for instance. Some IMU were available with two packages, as the Invensense MPU-6050 or ST LSM303DLH. This kind of solution is very interesting in order to reduce the footprint of the IMU part and the data acquisition is realized with an easy to use interface: i2c [Semiconductors, 2000].
These chips are available on development boards and distributed as the Invensense MPU-6050EVB (see Fig.3.4a) and ST Microlectronics iNEMO (see Fig.3.4b). Their characteristics are very closed such as the scale, consumption or size for instance. So, there are two interesting IMU that can be used to compute user orientation but another one appeared few months latter: the MPU-9150. This component is an update of the MPU-6050 with the addition of the magnetometer in a same package: 4mmx4mmx1mm, and few functions were added in the Motion Processor Unit. The MPU-9150 is our final choice. The next step to get a complete system is the research of a computing architecture, which answers our needs. This part is at the heart of the mobile device, it has to embed all complex tasks and provide the frames to the AR display. The main solutions that can be used to validate all the main software tasks of the application are detailed in following section.



(a) MPU 6050 development board

(b) iNEMO development board

Figure 3.4: Pictures of IMU development board

3.2.3 Computing architecture

Hardware architecture has to be embedded with the AR Display for mobility reasons, so power consumption and miniaturization are of the utmost importance. The embedded system must support the following functions: head tracking, object orientation and drawing, data acquisition must be implemented through a wireless network for sea-mark data, boat GPS, AIS data, stream and wind Gribs, finally the video has to be sent to the AR Display.
Our constraints are listed as follow:

- Compute pose estimation from IMU with i2c bus.
- Acquire data from a wireless connection (BlueTooth and/or WiFi).
- Load Electronic Navigational Charts and gribs data from a memory.
- Compute 2D and 3D objects position and orientation.
- Send frames to a display.
- Compute application functions such as collision risks and so on.

Head tracking based on inertial sensors is a complicated task but considering the sensor acquisition rate (e.g. 30Hz), it can be performed [Diguet et al., 2013] with 32 bit low-power processors (e.g. ARM Cortex or Intel Atoms) available on mobile SoC. The main computing task are graphic ones when complex objects are considered, the availability of GPU combined with OpenGL framework simplify the programming of this part of the application. It is also important to observe the use of see-though device offer interesting opportunities to reduce the computing complexity, first ergonomics means few and simple object and secondly there is no background. It means that the

computation requirements are far from the complexity of video games, this is an interesting opportunity to save power. Some solutions have been proposed to simplify the programming of AR applications for instance for Android OS [Kurze and Roselius, 2010], but beyond programming productivity power consumption is not really addressed as it should be. Based on the previous observations, some dedicated architectures have been proposed [J-Ph. Diguet and Morgére, 2015] as an aggressive solution to save power and area but they require the costly design of ASICs or FPGA-based embedded systems and can't barely take benefit of existing OS and programming environment.

Mobile platform such as tablets are the best trade-off between productivity and power efficiency considering the active developer community and the growing set of available APIs. Indeed this device provides accesses to sensors such as inertial sensors, light or pressure for instance. They implement SoC, based on GPGPU (general-purpose processing on graphics processing units) architectures, which combines multiprocessor architectures and graphic accelerators (e.g. Two Cortex A9 and PowerVR in TI OMAP4 for instance). This device is also able to communicate with wireless connectivities, store/load data and display 2D and 3D objects on a display. But they usually implement other devices such as Image and Video processor, audio chips and so on, which are not necessary in our case.

The TI OMAP3 / OMAP4, Nvidia Tegra 2, ST-Ericsson A9500, Qualcomm

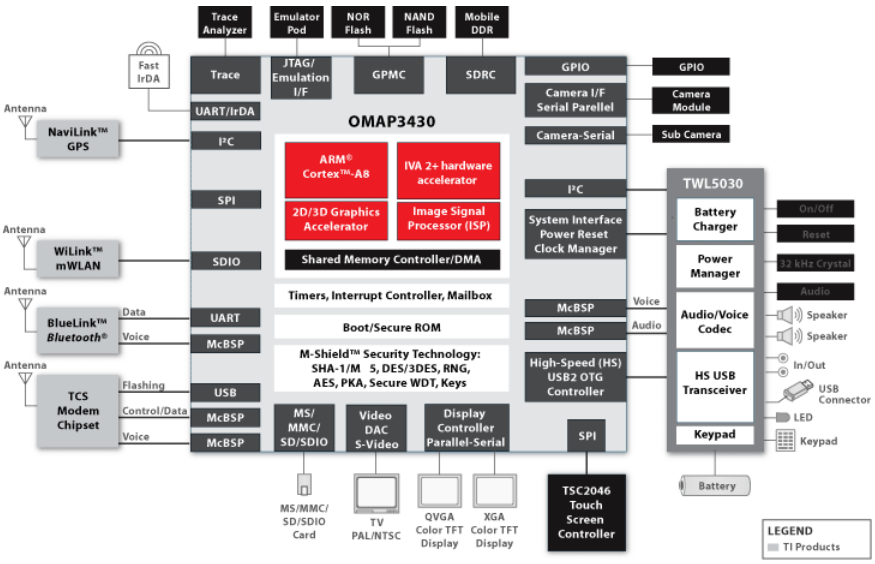


Figure 3.5: Texas Instrument OMAP3430 architecture

Snapdragon, Samsung Exynos 4 and Apple A5 are the different SOCs available at the end of 2011 but only few of them were accessible as development board, the latter are visible in Fig.3.5, 3.6, 3.7 and 3.8. We present only SOC architecture from development board because we would test the most

important software tasks on this kind of architecture to confirm the use of a GPGPU architecture in our context application.

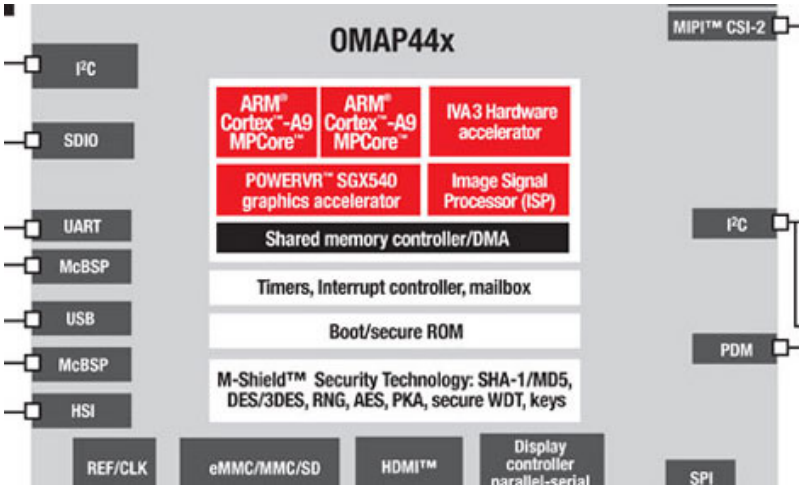


Figure 3.6: Texas Instrument OMAP4430 architecture

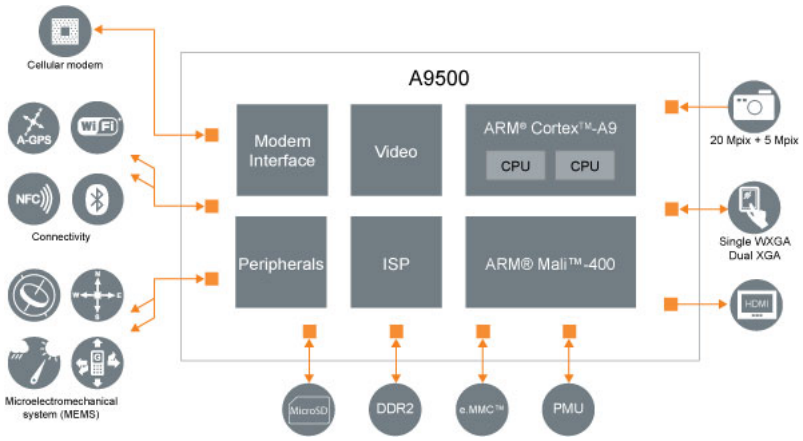


Figure 3.7: ST-Ericsson A9500 architecture

Table 3.3 is an overview of all chips and elements used in 3 main development boards: the Overo, the Pandaboard and the Snowball. In our mobile architecture, some specific chips are required, they are cited in Section 2.4.3. In Table 3.3 the most interesting development board to run an augmented reality application is the Snowball board. First, the System on Chip is a double cortex A9 (CPU) with a Mali 400-MP (GPU) that runs a Linux or Android OS. The SoC is able to run complex tasks such as pose estimation, decoding, etc. The head tracking and positioning are also available thanks to the GPS and MEMs on the board. Finally wireless connections can be established with a distant PC to acquire GPS or AIS data for instance. A

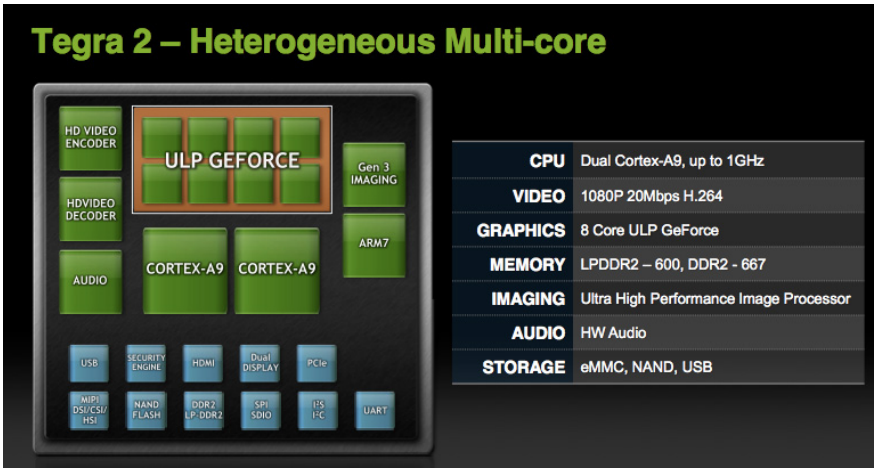


Figure 3.8: Nvidia Tegra2 architecture

Table 3.3: Overview of few available development boards with SoC in October 2011

Model	Overo	Pandaboard	Snowball
SoC	OMAP3530	OMAP4430	A9500
video output	HDMI	HDMI/DVI	HDMI
i2c	yes	yes	yes
UART	yes	yes	yes
microSD	yes	yes	yes
WiFi	802.11b/g	802.11b/g/n	802.11b/g/n
BT	v2.0+EDR	v2.1+EDR	v2.1+EDR
GPS	extension board	no	yes
MEMs	extension board	no	yes
OS	Linux	Linux	Linux / Androïd
Size	58mm x 17mm	115mm x 102mm	85mm x 85mm

VGA connection is missing but a HDMI/VGA converter can be plugged to get the video stream in the glasses.

All elements to get a complete mobile augmented reality can be tested on this development board before the design and the achievement of a prototype. So, the next section is a description of our prototype named MMARS: Marine Mobile Augmented Reality System.

3.3 Hardware prototype

3.3.1 Introduction

As it is written in previous section, the first tests have been realized on a development board. This was done to test the useful components and feasibility

of the architecture chosen to run a mobile augmented reality application in marine context. All the following software tasks: compute pose estimation with filters, display 3D objects with the help of a 3D engine, decoding the AIS/GPS data have been implemented on the Snowball board. More details on every software tasks are providing in the section 3.4 in this chapter. But before running the application, a full embedded system is required and this section details the design and the achievement of our prototype.

3.3.2 See-through display

The product presented in Fig.3.9 is the LCOS version of the classical MG1 ski mask from Laster Technologies. The display technology is based on a

Figure 3.9: Laster See Through mask, snowball GPGPU mobile platform

LCOS panel with LED backlight to reach expected high luminosity that cannot yet be delivered with AMOLED technologies. The resolution and the field of view are suitable for a see-through device: 800x600 and 40*30 degrees respectively, which is better for light intensity, resolution and FoV than other solutions, available at this moment, like Vuzix or Google-glasses. Display specifications give to the user an impression of a screen size about 97,5 inches at 2,7 meters, which is large enough to display our information (text and 3D objects). The power consumption is 1.3W under 7.2V with a display frequency of 85Hz in the AR device.
There are two main parameters on the Laster Technologies ski mask, the first one is the display frequency and the second one is the light intensity. The first parameter can be set between 10 to 85Hz, so the best frequency in

order to safe power and get enough speed for human eye is 25Hz. The second parameter has five levels, from 20% to 100% of the maximum luminosity, which can be automatically set depending on the outdoor light thanks to the light sensor on the mask.

The next step is the choice of components to acquire head orientation to display according to the display FoV and user orientation the graphical elements.

3.3.3 Head tracking

Considering the constant boat motion and the inapplicability of camera-based solution it is necessary to pay particular attention to the head tracking problem based on inertial sensors. Integrated 9DOF MEMS are now available (three axis accelerometer, magnetometer and gyroscope), and we adopted this solution to compute angular positions of the user head. Considering shocks and object distances the accuracy can be relaxed, one or two degrees of errors are acceptable. If we consider an error about 2 degrees, the position error from MEMS will be as follow:

- 0.35m for 10 meters.
- 3.5m for 100 meters.
- 35m for 1000 meters.

In case of an harbour arriving or leaving the speed is limited, so this error compared to the GPS one is not dangerous for the sailor, moreover when the visibility is not null, the users can adapt the position themselves and establish the relation between reality and map indications. To get a precision about 2 degrees, calibrations and filters must be implemented.

It turned out that the main difficulties are the MEMs calibration.
Firstly, the accelerometer can be calibrated once, this component can have an offset, an error alignment and a scale factor error. To correct these errors, some easy proceedings can be done but they have to be very accurate. An accelerometer is not disturb by environment but it measures linear acceleration and gravity, the software task has to separate these data for pose estimation. Moreover, the gravity is changing according to the position on earth and we could improve the precision by corrected it with maps.
Secondly, magnetometer has also scalar factor and misalignment errors but the offset can be more important than accelerometer or gyroscope ones. Actually, this offset is generated by hard iron distortions, it arises from permanent magnets and magnetized iron or steel it adds a constant magnitude field component along all axis. In order to detect an error due to hard iron distortions, the system has to be aware of the magnetometer norm. Magnetometer is also disturbed by soft iron due to soft metals that perturb the

earth's magnetic field, and this error depends on the magnetometer orientation.
Thirdly, the gyroscope presents an offset on the three axes like the others MEMs but this sensor is used to get a position with angular velocities as output data from it. This induces an error caused by the integral and it has to be corrected at runtime after the calibration step. This correction is performed by Kalman-like filter for instance that estimates the gyroscope data drift.

As an example, MTi is IMU commercialized by Xsens with MEMs that has a static accuracy less than one degree and a dynamic accuracy about two degrees RMS. So, our target is to get an accuracy close to a commercial IMU, the MTi one, for our head tracking system. With the help of calibrations and filters, head tracking by MEMs is very interesting for its size, power consumption, low processing needs and accuracy.
During the research of the MEMs components for our prototype, the company InvenSense realized a new full embedded IMU: a 3 axis accelerometer with a 3 axis magnetometer and a 3 axis gyroscope MEMs. So, as it was written in previous section, we chose the MPU-9150 because of its small chip, only 4mm by 4mm, it requires a classical i2c link to acquire data and performance are enough to get the same performance than the Xsens MTi.

Once the IMU and the display have been chosen, the hardware prototype can be designed with the computing architecture part, the following section details our electronic board to run AR application.

3.3.4 Computing architecture

The AR glasses available at the end of 2011 might not be used alone in geolocation applications, only orientation can be used according to the MEMs or the camera present in some of the glasses listed in Section 3.2.1. This is why, according to our objective, which is a low power, low footprint autonomous wearable solution, we have designed our own embedded system. When we started the design of the prototype, a new version of the small development board appeared: the Duovero. The Duovero board [GUMSTIX, s355] has the same size as the Overo, only 58mm * 17mm, it's just an hardware update from it (Fig.3.10).
In addition to this board, some extension boards were also available with all signals accessible, we decided to use them to design a new one with our constraints. So, our prototype is based on a Gumstix Duovero Zephyr plugged (as a motherboard) to an ad hoc extension board (daughter-board), that we have specifically designed and it is visible in Fig.3.11. The daughter-board size is 78mm * 21mm. This motherboard implements a System on Chip (SoC) based on a TI OMAP4430. It includes a dual cortex A9, a PowerVR SGX 540 GPU, a video controller and a power management. This SoC allows to compute 3D rendering on the GPU using the vertex and fragment

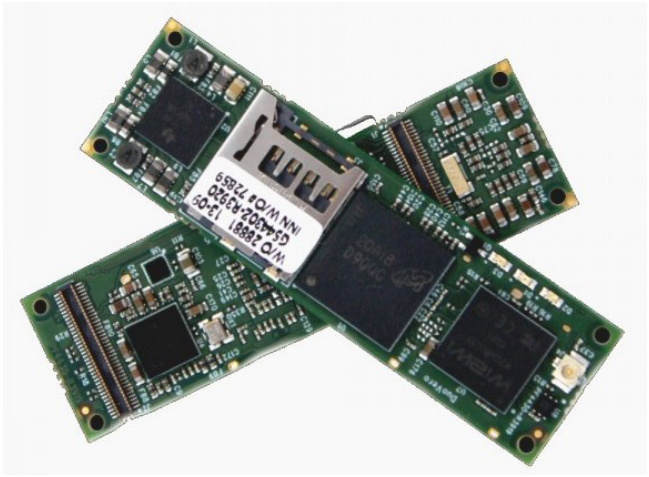


Figure 3.10: Bottom and top view of the Gumstix Duovero board

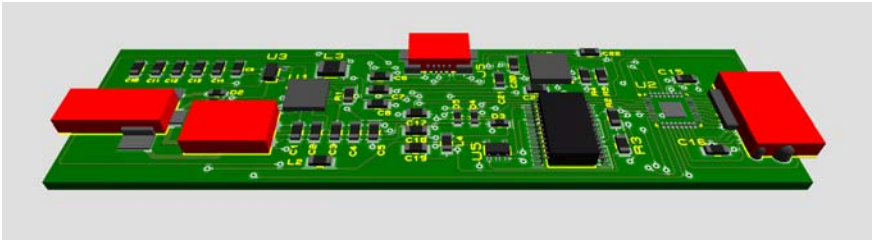


Figure 3.11: Top 3D PCB view of our daughter board

shaders and to execute concurrently separate tasks, such as acquisition and data fusion, on the dual core processor. Moreover the CPU frequency can be set between 300MHz and 1GHz according to the power management policy. The video controller is used to send frames directly to the see-through display. The output resolution and the frame rate can be in accordance with the display thanks to the video component. It should provide a frame rate about 30Hz to limit the power consumption and a resolution equals to 800x600. Near the SoC, a Secure Digital card slot allows us to store applications, data and an Operating System. An important memory storage is required to store OS, drivers, 3D objects, charts and application files. Finally the board implements wireless connectivity: Bluetooth 3.0 and WiFi 802.11(b/g/n) to communicate with the boat environment.

Our system doesn't need power greedy wireless connection as it was used before. Indeed, the first AR glasses sent sensor values by wireless connectivity, frames were computed on a distant device such as a smart-phone or a PC and the latter sent frames to the glasses. We only need few data by wireless connectivity to get GPS, AIS data for instance because the display updating, the geolocation and orientation estimation are all computed on

the embedded system, not on a distant device.

In case of boats equipped with GPS and AIS devices a low data-rate Bluetooth connection can be established with the embedded system. Some improvements are still possible, currently our Bluetooth chip is under BT 3.0 but it will be upgraded to a BT 4.0 chip with a better energy efficiency. Another improvement will be the addition of an embedded GPS. If we consider the case of recreational boats without any devices on board, MMARS could be fully autonomous if the on-board GPS were activated and if nautical charts were pre-loaded. Finally, the mobile device needs to interact with the users by mechanical systems in order to set parameters. So, push buttons or/and touch-pad should be added in future to let the user choice or select profiles, change the displayed data and so on. This is necessary, in order to provide a Human-computer interaction.

3.3.5 Conclusion

Due to the maritime context, the mobile device, which embeds a navigational assistance application, has to be glasses for ergonomic an mobility reasons. Moreover, the sailors have the habit to look at the sea and seafront to check navigation aids/landmarks, so the device also has to let the real world visible. Finally the weather conditions can vary a lot, so the information on the display have to be visible in order to help the sailors in any conditions.

This is why, we realize a full embedded prototype, composed of a ski mask from Laster Technology, which is an OST glasses in addition with an electronic board to offer an autonomous system. We design a small board (see bottom in Fig.3.12) with all the necessary hardware (IMU, SoC, etc.) to run an Augmented Reality application named MMARS. Few improvements will be implemented in a pre-industrialization phase, the addition of a GPS chip, improving the BlueTooth chip to a 4.0, reduce the size for instance.

Our hardware board (see Fig.3.13) is now able to run all software tasks and APIs of our Marine Augmented Reality Navigation Assistance Application, which are described in the following section.

3.4 Software architecture

3.4.1 Introduction

In order to design a flexible application, we decided to follow the schema in Fig.3.14 with the 5 steps. The mobile system has to acquire data from maritime devices such as GPS, AIS, radar, navigation processor and use 3D ENC data. So, we have identified five services, which depends on the devices connected to the system and these services will be provided to the user as follow:

- Service S1 is related to the GPS data.

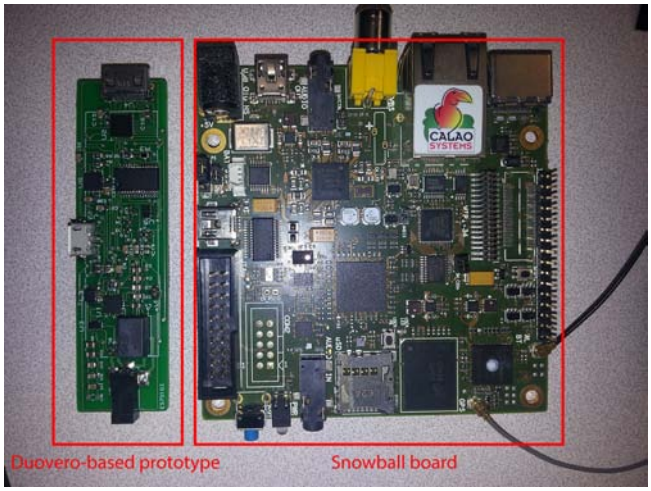


Figure 3.12: Comparison between Snowball board and our prototype

Figure 3.13: Our Marine Mobile Augmented Reality System (MMARS)

- Service S2 is related to the AIS data.
- Service S3 is related to the radar data.
- Service S4 is related to the Electronic Navigational Charts (ENC) data.
- Service S5 is related to the navigation processor data.

Each of them will be presented in detail latter, in Section 3.4.2. Firstly, the application have to connect the system to the boat network and detect the available services, it also has to detect onboard services, this is the step 1 in Fig.3.14. Then, once the services are known, the corresponding data structures are allocated and IHM generated, this is the second step in Fig.3.14.

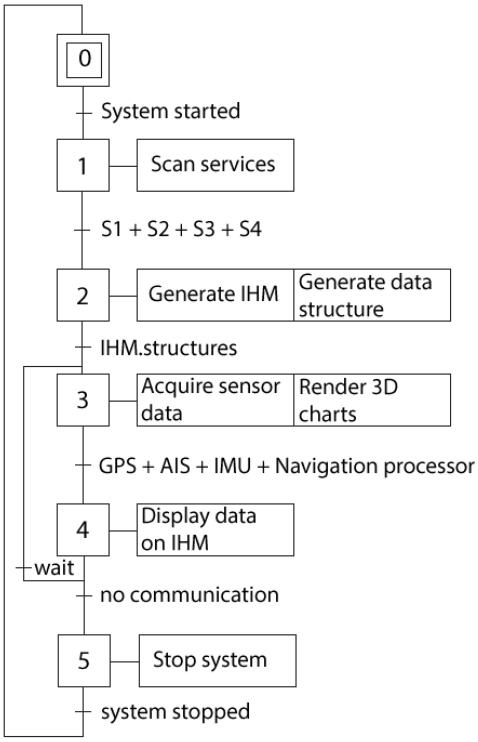


Figure 3.14: Grafcet, which represents the global working of the application

The next step is the acquisition of the data from the boat network like the GPS, AIS for instance and onboard sensors such as the GPS and the IMU (step 3 in Fig.3.14). Once the data are stored, the application has to sort and decode them to update the right information on the display, this is step 4 in Fig.3.14. The step 3 and 4 are repeated until the communication is stopped between the system and sensors or when the user decides to stop application, this is the last step in Fig.3.14.
So this Grafcet represents how the application will work to provide graphical information to the user automatically, in order to realize it, the tasks identified in Section 2.4.2 have to be implemented. An overview of the interaction between software tasks and services is visible in Fig.3.15, in the following section we explain in details first the services, then the software tasks.

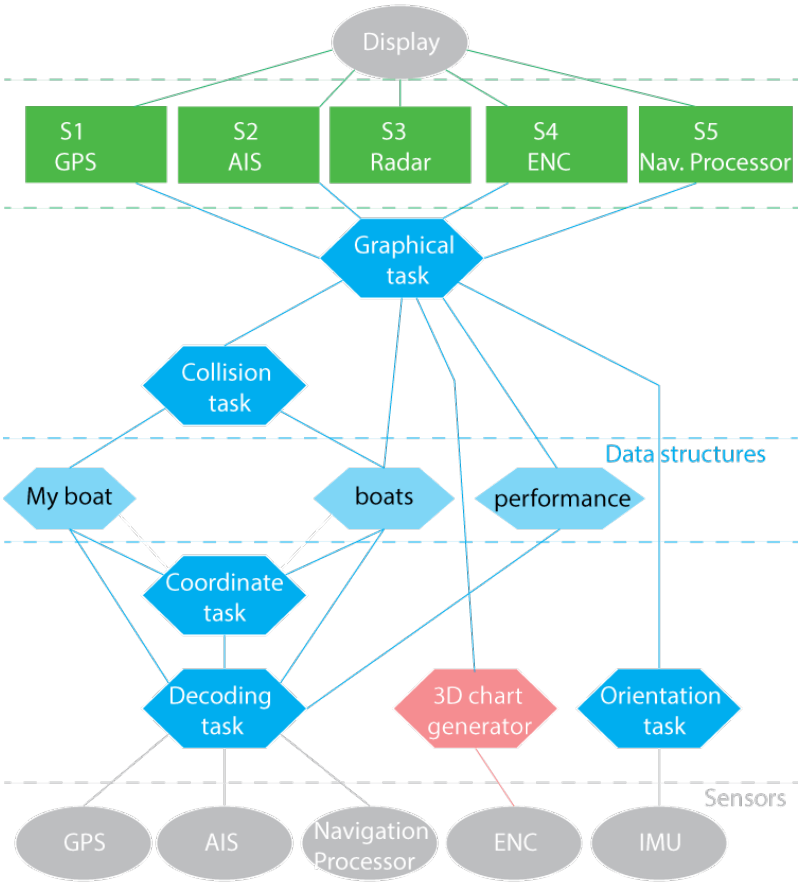


Figure 3.15: Software tasks used to provide services

3.4.2 Application

All services identified in Section 3.4.1 depend on the boat environment, indeed the services for recreational boats without any devices are not the same than fully-equipped ferries for instance. So the system must be adaptive, at start time, it has to scan the boat environment to detect which services are available on the boat. Then, the user should select which one he wants, as depicted in Fig.3.16. If no services are detected on the boat and if the ENC service is not available, the system uses its GPS device and runs the simplest orientation application.
In a typical example for a recreational boats, we consider GPS, AIS devices and ENC data, so our system detects the S1, S2 and S4 services, collects data from ENC, load a 3D chart and starts a navigational application with seamarks and information about boats in the vicinity.

In order to offer an helpful application to the user, we have designed it as a services server. This server currently offers five kinds. S1, S2 and S3

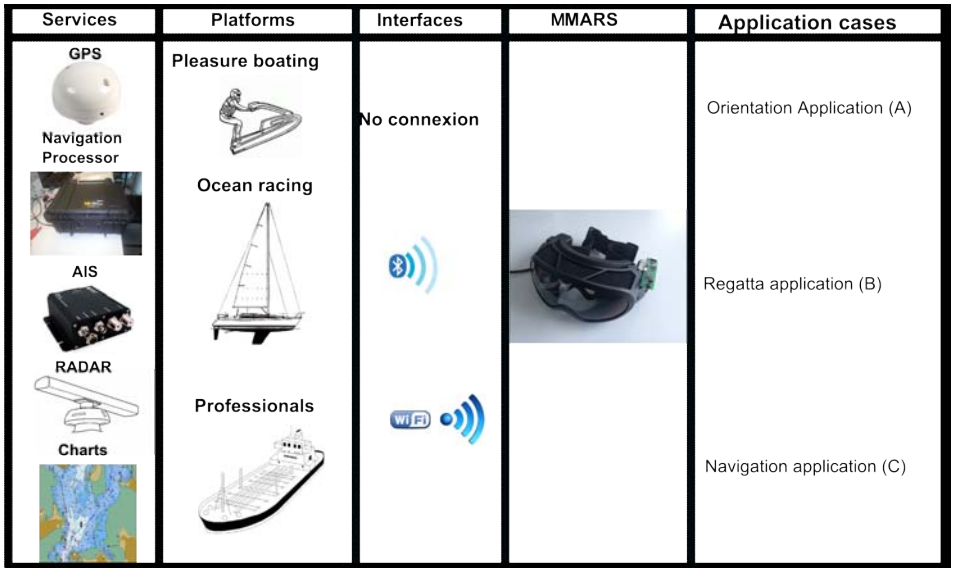


Figure 3.16: Pervasive environment: depending on services available and according to the interfaces on platforms MMARS offers multiple application cases

services are related to positioning devices, S4 gives access to nautical charts and the last one S5 is related to on-board sensors to evaluate the boat performance.
Four information are extracting from S1: position, COG (Course Over Ground), SOG (Speed Over Ground) and time, this is the main service, no applications could be run without boat position. S1 is able to offer sufficient information to run the application that displays in alphanumeric format the data related to speed, course and time.
S2 provides information on boats in busy areas, it is used to prevent collisions with other boats that emit AIS data according to the availability of S1. With combination of services S1 and S2, the application is able to display information about surrounding boats as presented in Fig.3.17 and can compute collision routes, can raise alerts and could display time to impact.

S3 could offer specific ARPA (Automatic Radar Plotting Aid) services based on radar data, it includes the closest target (boats or other objects like emerged reefs) position (CPA) and the time to target (TCPA) as well as the course and speed of the detected objects.
S4 is a service able to load and extract data from 3D charts (see Chapter 4) and gives all graphical elements related to the navigation aids and depth water to the task in charge of displaying 2D/3D information in the navigation assistance.
If the boat is equipped with a navigation processor, then the system is able to provide the service S5. A navigation processor on a well equipped sailing

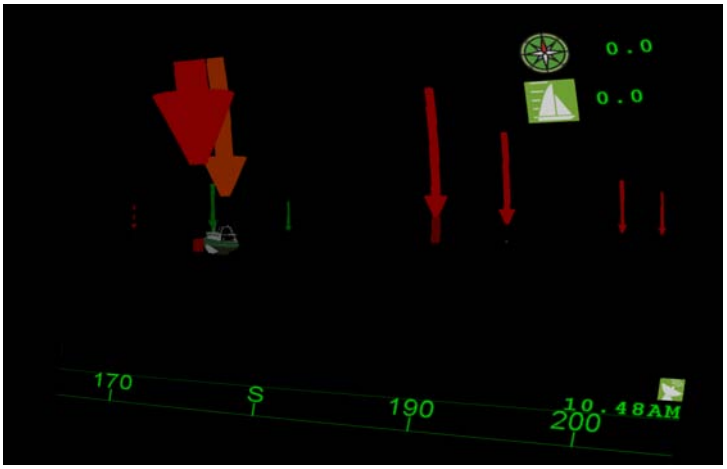


Figure 3.17: Navigation application (C) - Services: S1 + S2 + S4 *(black background for transparency on LASTER see-through glasses)*

boat typically provides data such as apparent and true winds, the boat speed over water, heading, roll and pitch. These data are extracted from sensors plugged to the navigation processor (anemometer, IMU, speedometer, compass, etc.) So, S5 is a service, which is able to use a list of data that are activated or not according to the sensors plugged to the navigation processor. Those information can be displayed like in Fig.3.18 as it was presented in [Douguet et al., 2013].

Figure 3.18: Regatta application (B) - Service: S5 (smartphone version)

Now, these services are identified but they need some software tasks to provide the right information to the user. The next sections details the tasks, which offer an easy to use and a flexible application by services provided to the user.

3.4.3 Tasks

3.4.3.1 OS

In our case, the mobile Operating System is an embedded Linux that offers a large amount of APIs and drivers. In augmented reality applications, the main functionality is the graphic task, since the system must draw 2D/3D objects with colours or textures. In embedded systems with mobile OS, such as Android or Linux, the OpenGL ES2.0 API allows the programmers to access a lot of 3D functions such as rotate, scale, translate, texturing, lighting and material rendering, according to the vertex and fragment shaders. We need a mobile Linux with X server because the 3D engine used to run graphical tasks needs it. This is a graphic environment implemented on Linux, it represents the graphical user interface. The graphical part is more detailed in the following section.

3.4.3.2 Graphical

If the services details in Section 3.4.2 are compared to the graphical interface deduced from the IHM study in Section 2.3.3, we have 5 sub-graphical tasks. The analogy between services and graphical tasks is the next one:

- S1 (GPS): this is the state graphical task.
- S2 (AIS): this is the AIS graphical task.
- S3 (Radar): this is the radar graphical task.
- S4 (Navigation processor): this is the boat performance task.
- S5 (ENC): this is the navigation aid task.

But one information is missing, this is the orientation object, we consider the orientation graphical task as the sixth. All the software tasks are brought together in Fig.3.19, only the graphical task related to the radar is not written because it is not developed for now.

All graphical tasks are managed by the free open source 3D engine Irrlicht. To understand why we use a 3D engine, we explain first, what is a 2D and a 3D object, then the utility of a 3D engine and we describe the role of the OpenGL ES API.

2D and 3D Further to the user study on the IHM 2.3.3, the application has to be able to display text, 2D logos and 3D objects (depth areas) to get a useful scene.

In a 3D application, texts are always based on fonts, as textures which can be png, bmp or jpeg files. To display text with a graphical API, OpenGL ES for instance, every characters are extracted from the font and placed on

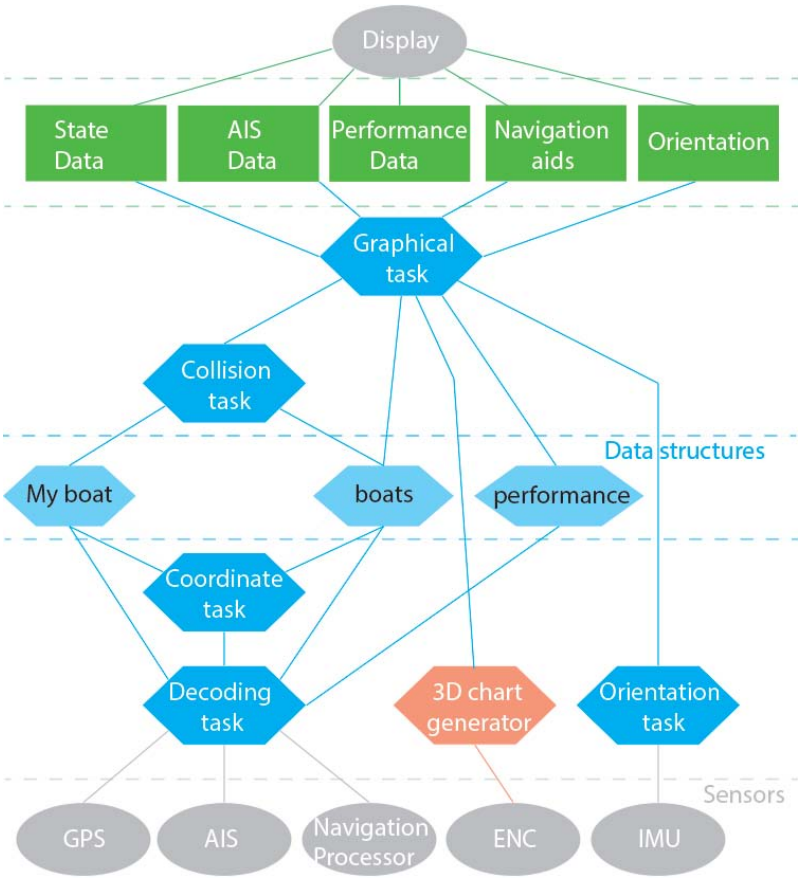


Figure 3.19: Software tasks used to display information

the display with coordinates in pixels. The conversion between a string to a text on the display is a tedious task for the developer, many functions have to be implemented to realize this feature.
To improve the feeling of perspective, we will use the arrows, buoys, top marks as 3D objects. But we have to define first what is a 3D object in 3D applications.

A 3D object is often named a mesh, which is composed by vertices, edges and faces. The smallest detail on a 3D object is the vertex. A vertex is a point with 3D coordinates: x, y and z as it is shown on the left in Fig.3.20. Then, the connection by a straight line between two vertices is an edge and they are used to construct the base object for a mesh: a face. An example of an edge is shown at the middle in Fig.3.20. So, vertices are not visible on the final object, there are only the skeleton of a mesh, the main part is the face. Finally, the last step to construct a mesh, or a 3D object, is the assembly of faces, which is the surface between three vertices, an example of a mesh built by faces is shown at the right in Fig.3.20.

All navigational aid objects used in the application will be designed as the west cardinal buoy in Fig.3.20.

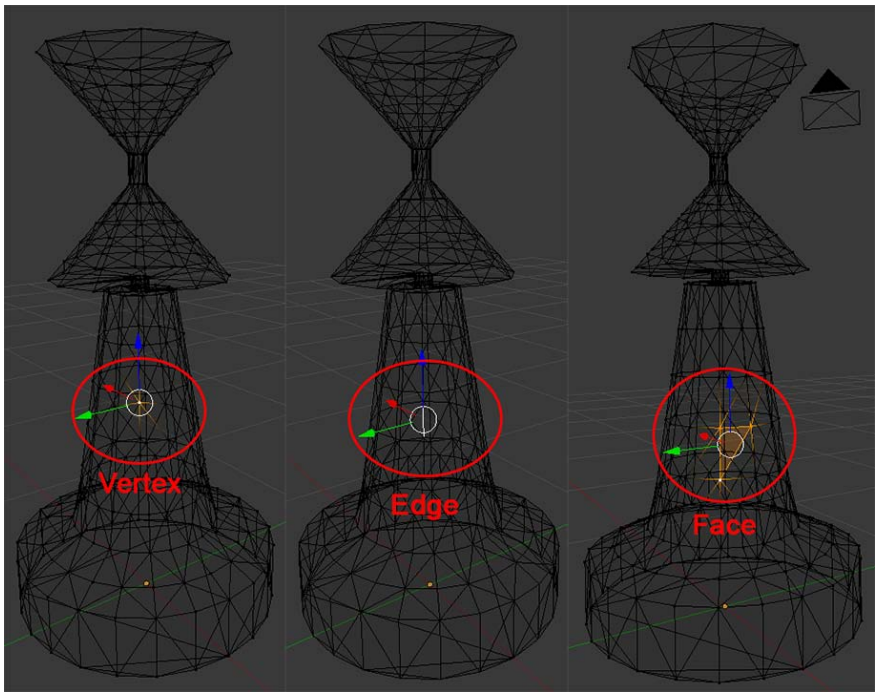


Figure 3.20: 3D west cardinal buoy designed with vertices, edges and faces

Now, the object is built but colours have to be added, otherwise it cannot be visible in the rendering. In 3D, a material represents a set of surface parameters such as the colour, shininess, reflectance, etc. In addition to these properties, some more complex effects can be also added on the surface: transparency, diffraction and sub-surface scattering. One or several materials is the first solution to colour a mesh and the other one is the used of textures. Indeed, it is also possible to use a picture as a texture and applied it on the shape, to assign object colour but even more, the colour properties such as intensity and reflection light angle.
However in our context, the objective is to get shapes easy to understand regardless of the conditions, so we will use a subset of the features. All the meshes used in the application are coloured only with materials: an example applied on a buoy is shown at the left in Fig.3.21.

Once the mesh is built and materials are applied, a 3D engine or a 3D animation software is able to render a scene with the mesh, a light (with parameters) and a virtual camera with the help of a graphical library. A scene is visible at the middle of Fig.3.21 and the result (the rendering) on the left part. The light and shadow combined with a perspective view create a feeling of 3D environment even if the screen is flat. This kind of representation

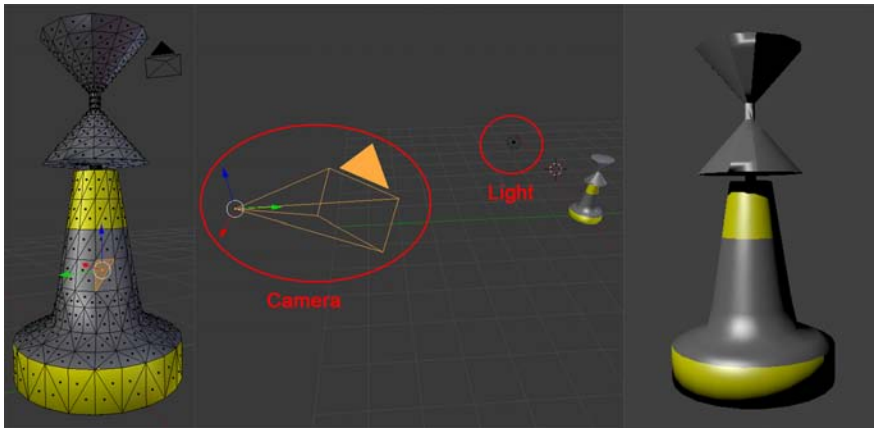


Figure 3.21: From a mesh with material to the rendering

is better in order to retrieve a perspective and the depth of field in a 3D environment, but the rendering with specific materials and lights requires more computing on the GPU. The 3D objects can be rendered directly by an API such as Direct 3D, DirectX, OpenGL or OpenGL ES, without a 3D engine but, for many reasons, which are explained in the following paragraph, a 3D engine makes easier the use of 3D.

3D engine A 3D engine is a user interface, which deals with devices and also simplifies the management of the meshes, lights and the virtual camera. It takes into account many file formats for the font and textures such as png, bmp, jpeg and it can load many 3D Object files such as Wavefront OBJ, 3ds, Collada, etc.

For instance, in a video game, some parameters can be set with the help of buttons, text lists and so on, in order to let the user interact with the application by the graphical interface and the user also sends commands by devices such as a mouse, keyboard and touchscreen. In a First-Person Shooter (FPS) on a PC, the gamer moves a player by using the arrows keys on a keyboard and orientate the view with a mouse in a virtual world, he can change details on the virtual player, settings on the screen, etc. With the devices and the 3D engine, the gamer changes the virtual camera coordinates and orientation at every frames and the rendering is computed on the GPU with the new parameters. This virtual camera is orientated and positioned on three axes: x, y and z.
The 3D engine makes easier the management of the Frustum in the graphical pipeline with the help of functions written in an higher abstraction level than a graphical library (OpenGL ES, Direct3D, DirectX, etc.). An explanation of the what is a Frustum is given in paragraph 3.4.3.2.

In conclusion, A 3D engine is able to take into account the scene settings such as the light settings, camera, etc. to simplify the use of graphical API,

it helps the programmer to manage the meshes with specific functions for loading/unloading many 3D object files, it implements a UI and handles event from devices. The next paragraph details what is the Graphical API OpenGL ES.

OpenGL ES There are two methods to render a scene, the ray-tracing and the rasterisation. The first one is the most realistic, the rendering is very close to real pictures and it implies strong computing.

Ray-Tracing is a rendering method, which computes the path travelled by the ray of light corresponding to each pixel of the image. The 3D-engine computes the path by travelling it backwards following the law of physics such as reflection and refraction.

In the rasterisation method, the rendering is a process where objects are represented as triangles (like the faces), on which the graphical processor applies some geometric transformations and computes the 2D coordinates of the triangles to display them on the flat surface (the screen).

In OpenGL ES, the programmer uses the vertex shaders to modify this step. A vertex shader is a program computed on the GPU to modify the geometry of the triangles. Once the 2D coordinates of the triangles are computed, the GPU changes them into a group of pixels, then it computes the colour of each pixels individually (fragment or pixel shader). In OpenGL ES API, the shaders are programs written in a language close to the C to specify actions in the rendering pipeline.
In both methods, the number of visible objects is limited to a given frustum by a near and far plane as depicted in Fig.3.22. This is a representation of

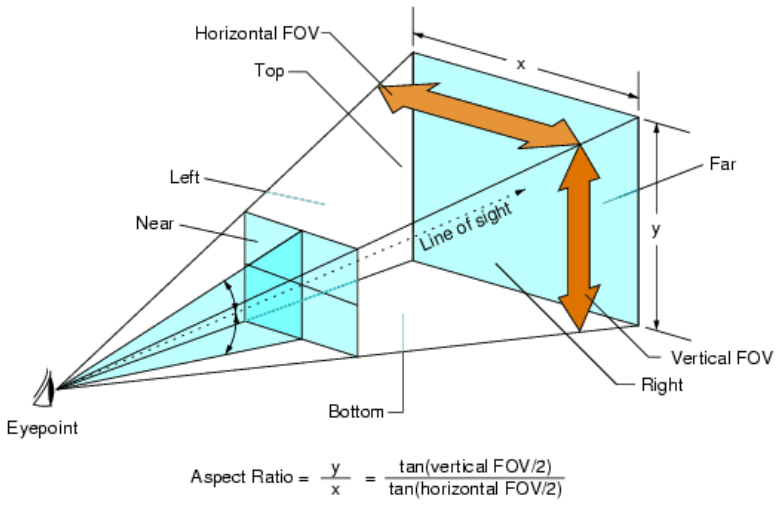


Figure 3.22: Frustum in OpenGL ES

a field of view that can be tuned with the Graphics API (OpenGL ES 2.0

for example) according to user choices. Indeed, the Field of View can be set in degrees or radian and the Znear / Zfar limit the start and the end of the visible area. Moreover, it is possible to activate the Z buffer, this is a process to manage automatically, by the graphical API, the depth (Z axis) of the 3D objects. For instance, if a buoy is in front of a boat, when the Z buffer is active, the API will not display the boat part normally hidden by the buoy. In case of the Z buffer is not active, if the buoy is the first object displaying and the boat the second, the buoy will not be visible in the rendering.

In conclusion, graphical computing run on the GPU to unload the CPU in order to realize other tasks such as orientation, decoding, etc. There are two kinds of rendering methods but only one can be used in our context, this is the rasterisation. Finally we have chosen to use the 3D engine Irrlicht to use OpenGL ES graphical API for many reasons. The first reason is the simplification of the management of 3D objects, second, it allows the use of simplest functions to run rendering with a graphical API. Moreover, this 3D engine is compiled in C++ for embedded Linux and Android and OpenGL ES is available for the rendering.

The device data (GPS, AIS, IMU) and their position or orientation are not workable for the 3D engine. The application has to decode the NMEA sentences from devices, change coordinate system and compute orientation. These tasks are detailed in the following subsections.

3.4.3.3 Decoding

In our application context, four services are linked to maritime devices: GPS, AIS, radar and navigation processor. The output data used on these devices is a common format, the most commonly used data format in marine electronics: the National Marine and Electronics Association (NMEA[1]).
The decoding task is in charge of decoding NMEA sentences but it also has to sort only the useful data and stores them. A NMEA sentence is composed as follow: The fifth characters identify the kind of device and the kind of

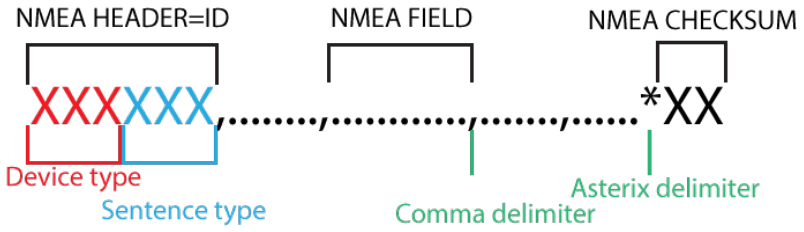


Figure 3.23: NMEA-0183 sentence.

[1] http://www.nmea.org/

sentence, then the features are separated by commas, a Asterisk delimit the end of the sentence with the checksum represented by two characters.
Now if we consider that the GPS and the AIS are plugged to the system, the decoding task will receive headers as follow:

- $GPXXX
- !AIXXX

The first header means that it is a sentence from a GPS and the second from an AIS device. The last 3 characters of the GPS header is used to get the kind of sentence, there are more than 20 kinds of GPS sentences available. The application only needs the position to display navigation aids, the COG, SOG and time, which are extracted from one GPS sentence: the GPRMC one. An example is visible in box 7 and the feature meanings are in the Table 3.4.

```
$GPRMC,053740.000,A,2503.6319,N,12136.0099,E,2.69,79.65,100106,,A*53
```

Box 5: Trame from GPS

Table 3.4: NMEA sentence from a GPS

Field Number	data	meaning	result
1	053740.000	UTC time	05:37:40
2	A	status	Valid data
3	2503.6319	Latitude ddmm.mmmm	2503.6319
4	N	N/S indicator	N
5	12136.0099	Longitude dddmm.mmmm	
6	E	E/W indicator	E
7	2.69	Speed Over Ground	2.69 Knots
8	79.65	Course Over Ground	79.65 degrees
9	100106	Date ddmmyy	10 january 2006
10	-	-	
11	-	-	
12	A	positioning mode	Autonomous
13	*53	Checksum	53

It is possible to acquire more data to get information about the signal quality, the altitude and so on. Indeed, the number of satellites being tracked is one of the data to know the accuracy on the receiver's position. Another example is the height of geoid (mean sea level), this data can be read to improve the position in our virtual environment. Both data are read from the GPGGA sentence on a GPS.
Contrary to the GPS, AIS data are not directly readable, there is a decoding

to apply on the sentences to sort data and extract them. Currently, there are 27 kinds of AIS sentences [2] but, in the application we focus on one set of sentences: 1, 2 and 3, which are the position report. In normal operations, an AIS transceiver will broadcast a position report every 2 to 10 seconds depending on the vessel's speed and every 3 minutes if the vessel is not moving (anchorage for instance). In the box 8, a type 1 AIS sentence is visible and its associated decoding in Table 3.5.

```
!AIVDM,1,1,,B,177KQJ5000G?tO`K>RA1wUbN0TKH,0*5C
```

Box 6: Trame from AIS

Table 3.5: AIS sentence type 1

Field Number	data	meaning
1	!AIVDM	Input AIS message
2	1	count of fragments in the currently accumulating message
3	1	fragment number of this sentence
4	-	sequential message ID for multi-sentence messages
5	B	radio channel code
6	177KQJ5000G?tO`K>RA1wUbN0TKH	data payload
7	0	
8	*5C	checksum

Only the type 1, 2 and 3 share a common reporting structure for navigational information, named the Common Navigation Block (CNB). This is the main information for the decoding task. The useful information are placed in the field 6 as in previous example (see Table 3.5) in an AIVDM sentence. The details of the fields in the CNB block are visible in Table 3.6, this is an example of a boat in a moored area. Once the decoding task has extracted data from AIS and GPS sentences, the data are ready to update boats and Myboat structure in Fig.3.19. Then, the collision task is able to compute if there is a collision risk [Perera and Soares, 2012], [Wang et al., 2013], [Kwik, 1989]. Finally, the graphical task prevent the user from a risk by displaying a warning message and highlighting the boat. Other AIS sentences could be decoded to get more information about the boats such as the size, name, draught or destination for instance but currently, we only use the CNB one. To simplify the computing for collision detection, we need to transform the coordinate data off the GPS and AIS, this computing are operate by the coordinate task and the latter is detailed in the following section.

[2] http://www.imo.org/OurWork/Safety/Navigation/Pages/AIS.aspx

Table 3.6: Decoding data payload AIS sentence type 1

Field	Length	Description	Results
0-5	6	Message Type	1
6-7	2	Repeat Indicator	1
8-37	30	MMSI	477553000
38-41	4	Navigation Status	Moored
42-49	8	Rate of Turn (ROT)	0°/min
50-59	10	Speed Over Ground (SOG)	0.0 knots
60-60	1	Position Accuracy	-
61-88	28	Longitude	-122.345832°
89-115	27	Latitude	47.582833°
116-127	12	Course Over Ground (COG)	51°
128-136	9	True Heading (HDG)	181°
137-142	6	Time Stamp	03:54:15
143-144	2	Maneuver Indicator	-
145-147	3	Spare	Not used
148-148	1	RAIM flag	-
149-167	19	Radio status	-

3.4.3.4 Coordinate transformation

Now, to detect a collision way and manipulate objects in the application, we need to change the coordinate system because a position in degrees or DMS format is a circular system and cannot be directly workable. It is easier to treat data in meters on a plane or a 3D environment (plane with altitude). So there are two solutions, the first one is the used of a 2D projection, similar to a paper chart and the second one is a local Cartesian coordinate system.

Firstly a geodetic system is a coordinate system to locate an object on the Earth surface with a specific reference for an ellipsoid. To locate someone with a GPS or the AIS, the position is compute with the world geodetic system WGS84 [Sandwell, 2002] in degree minute second (DMS). For example, the WGS84 specifies a semi-major axis (a) 6378137.0m a semi-minor axis (b) 6356752.314245m and a inverse flattening (1/f) 298.257223563 for the ellipsoid, where

$$f = (a - b)/a$$

A representation of the shape is visible in Fig.3.24.

A GPS is able to know the sea surface, which is different from the sea level, named the geoid. An example is visible in Fig.3.25. The position from a GPS (latitude/longitude) is a point on the ellipsoid, and the altitude is the height between the latter and the object. Some of the GPS receiver are able to take into account the distance between the geoid and the ellipsoid to get a better precision for the altitude. In our context, the GPS is on a boat, so we don't need to take consideration of the altitude.

Our first solution is the UTM (Universal Transverse Mercator) projection [Agency, 1989] that is used for maritime charts. The UTM projection divides

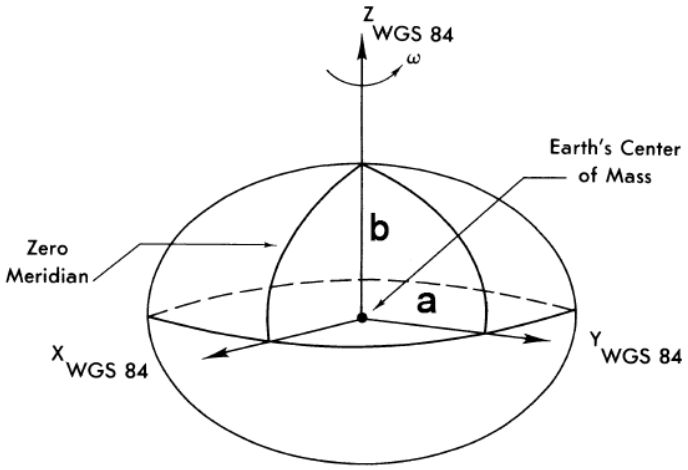


Figure 3.24: World geodetic system WGS84

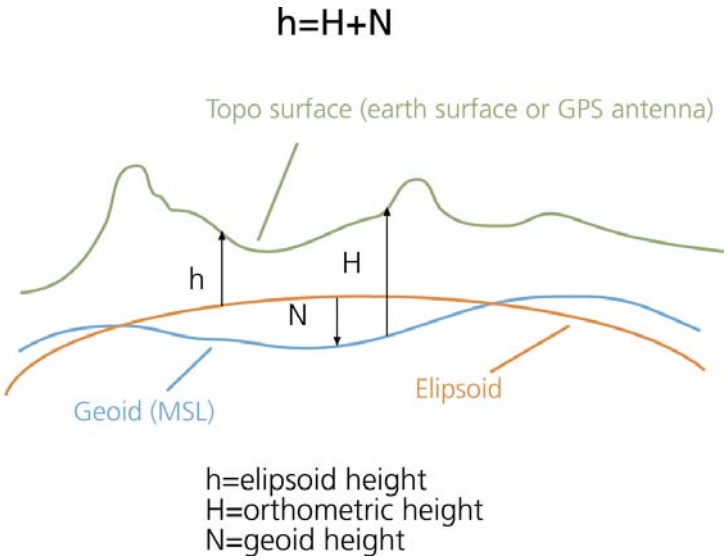


Figure 3.25: Difference between altitude, geoid and ellipsoid

the earth in sixty zones with a specific correction for each of them. The zone and the corresponding correction can be retrieved with a computation from GPS data. The position accuracy does not depend on the UTM projection which can be about one nanometre but on the electronic navigational chart scale. Due to the GPS accuracy, a precision of 1 meter is sufficient. Finally, the UTM coordinates system corresponds, in a limited area, to the real world view because it is a conformal projection (keep angle) and the distance errors are limited (less than a GPS error).

The second solution is the transformation of the GPS coordinate into a local Cartesian coordinate system named East North Up (ENU), which is also named an Earth-based coordinate system. There are two steps to get the position in ENU, we must compute the LLA (Latitude Longitude Altitude) to ECEF (earth-centered earth-fixed) coordinates first, then compute ECEF to ENU coordinates. Fig.3.26 presents an overview of these three coordinates systems.

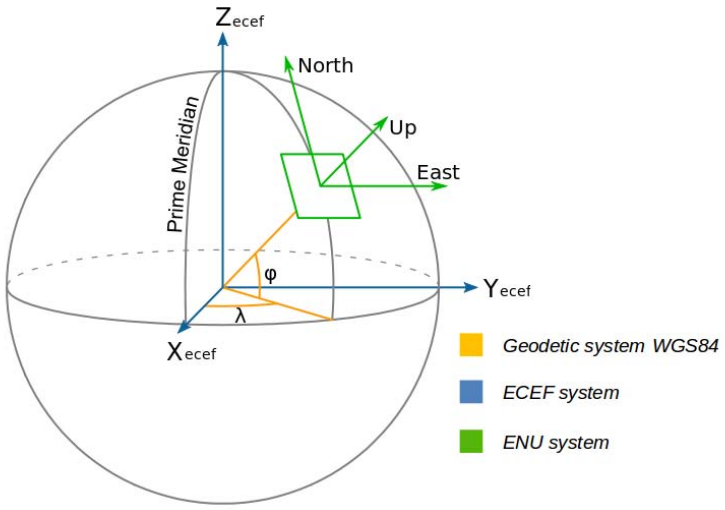


Figure 3.26: Few geodetic coordinate systems

Currently, we use the UTM solution in the application, the precision of both solutions is very close, but even if ENU solution requires less computing, UTM solution allows easier computing (2 axes) for collision detection. An other aspect is the generation of the depth area, which will be explain in the next chapter, the UTM projection has only 2 axes whereas the ENU has 3. In the second solution, the user is place on a tangent plane on the Earth surface, so it induces an altitude due to the Earth shape (ellipsoid). So if we display data at few kilometres far from the user, the sailor will have a similar view than the real one, with small objects that are not visible on the horizon as it is presented in Fig.3.27. This solution is more realistic for the user, so some tests are being conducted to use ENU coordinate system in the application instead of the UTM, which is currently used. With the help of the coordinate task, the user is positioned on the globe but we don't know where the sailor is looking at. Thus, the orientation task is explained in the following paragraph.

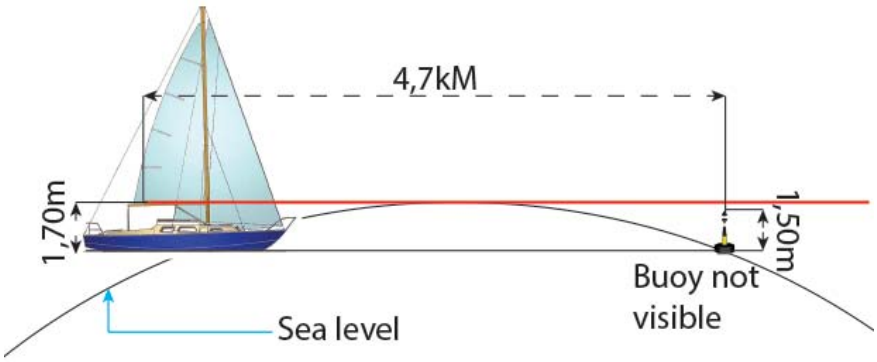


Figure 3.27: Impact of the Earth shape on the user's view

3.4.3.5 Orientation

As it was previously seen is Section 2.4.3.1, the pose estimation is computed with information, which comes from the MEMs. The orientation of a rigid body is determined when the axis orientation of a coordinate frame attached to the body (the body frame) is specified with respect to an absolute coordinate system, usually named the local frame or Earth frame.

First, the accelerometer is used to compute the roll and pitch, these are the rotation angles around the X and Y axes from the sensors frame, visible as blue axes at the top in Fig.3.28. Then, the magnetometer is sensible to the ambient Earth's magnetic field, and the azimuth angle is extracted from this sensor, this is the orientation around the red Z axis in sensors frame (top in Fig.3.28). Both sensors on the PCB are not in the same coordinate system, so a conversion in a common body frame is necessary to get the three orientation angles. A representation of the body frame is visible at top and middle in Fig.3.28.
When the board has the 3 orientations at zero, its Y axis is oriented toward the magnetic North, the X axis is pointing to the East and the Z axis is oriented to the Up, in a tangent plane of the Earth's surface (see middle in Fig.3.28). Once the orientation is computed in this Cartesian axes (body Frame), it has to be applied in the 3D engine, which has an imposed Cartesian axes as it is presented at the right in Fig.3.28.

But the raw data are not directly workable. Indeed, the magnetometer and accelerometer can't be used on a body in movements. A filter has to be implemented to smooth data and correct orientation between the MEMs with the help of a gyroscope. There are few kinds of filters:

- Complementary filter [Lawitzki, 2012].
- Gradient Descent filter [Sebastian O.H. Madgwick, 2011].
- Extended Kalman Filter [Brigante et al., 2011].

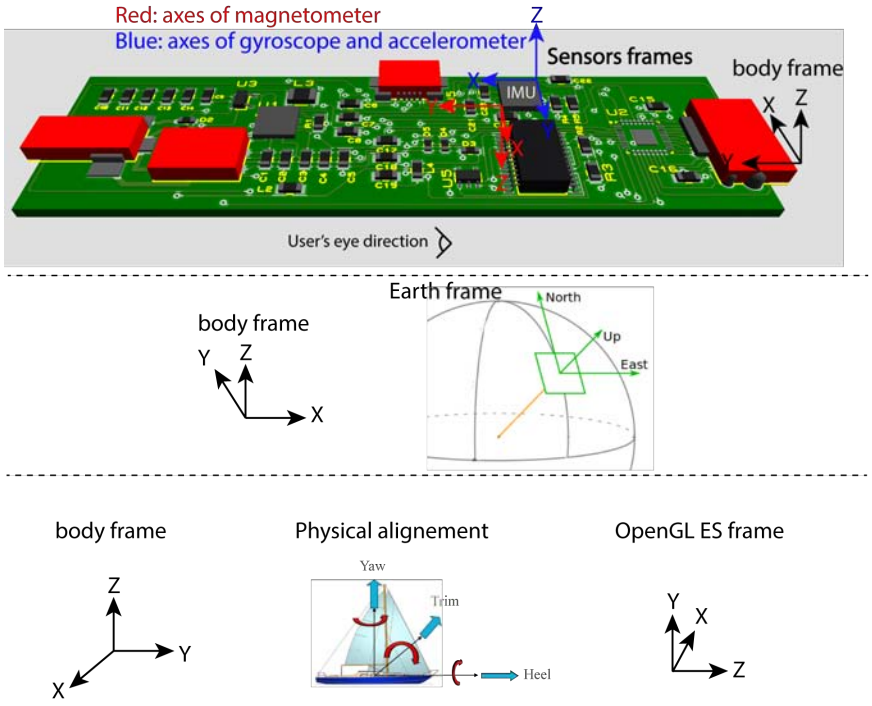


Figure 3.28: Sensor frame, body frame and OpenGL ES frame

These filters have specific roles depending on the needs and there is a big scientist community around these filters, the literature offers a wealth of information. So, this is a set of filters adapted to the practical used case. Indeed, we selected the filters usable in the maritime context, on a boat there is a low dynamic, this is not either video games constraint nor flying one. More details on the filters and their use are explained next.

Most of the filters compute the pose estimation with quaternions. A quaternion is a vector in four dimensions, which represents an orientation around a direction. This is very important to use this representation, firstly to avoid the Gimbal Lock [Hoag, 1963] even if the risk is weak on a boat. Secondly, computing with quaternions avoid the numerical instability in matrix computing. In 3D, the quaternion and the rotation matrix 4x4 are used to apply some orientations on the virtual camera and 3D objects, a representation is visible in Fig.3.29[3]. The matrix contains three kinds of information, the translation, the rotation and the scale. The translation can be applied independently from the others, the rotation and scale are affected each other (left part in Fig.3.29). On the other hand (right part in Fig.3.29), a quaternion has three numbers, which are like a vertex coordinates and represent a direction. The fourth number is the angle value applied around the direction

[3] http://blog.db-in.com/cameras-on-opengl-es-2-x/

represented by the vector.

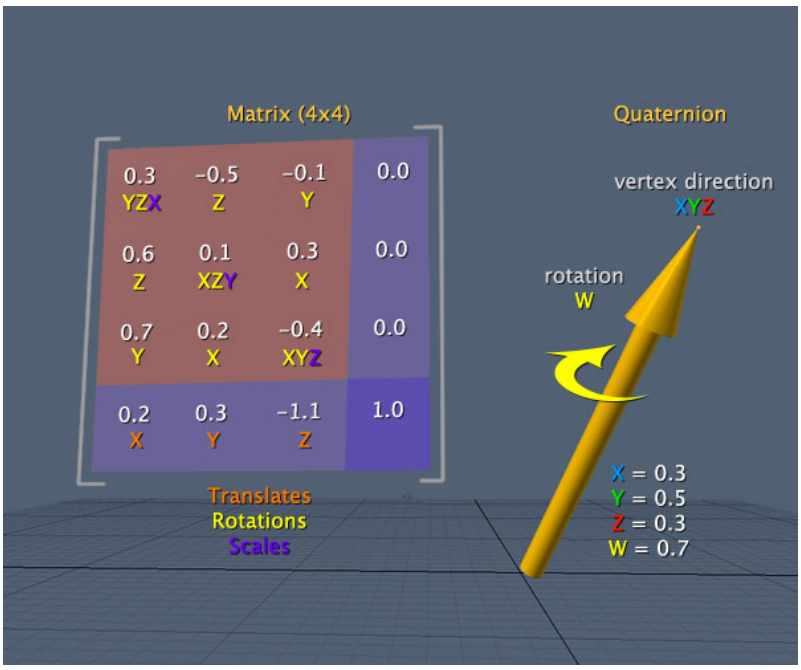


Figure 3.29: The rotation matrix and quaternion representation

Complementary filter This filter algorithm fuses accelerometer, magnetometer and gyroscope sensor data. The first step is the orientation computation with the accelerometer and magnetometer with the true north as reference in static condition (low frequency). In order to improve the accuracy in dynamic orientation, the use of a gyroscope is required. But this sensor leads to filter data because of the drift caused by the integral to get orientation from the angular velocity. The solution is the use of this sensor at a high frequency, filter with a high-pass filter and correct its orientation with the one computed by the accelerometer/magnetometer. This is the simplest filter to implement and an overview is visible in Fig.3.30.

This kind of filter should be improved with the addition of the sensor's offset. Higher data rate is, better is the accuracy. In this filter there are no calibration phases, it could be used on every smartphones or tablets easily but it cannot get the same accuracy as the Gradient Descent or Kalman like filters.

Gradient Descent filter In [Sebastian O.H. Madgwick, 2011] the gradient descent algorithm is used to compute an orientation from the magnetometer, accelerometer and the gyroscope, this orientation is represented by a quaternion ($\hat{q}$). The Gradient algorithm minimizes an error function by

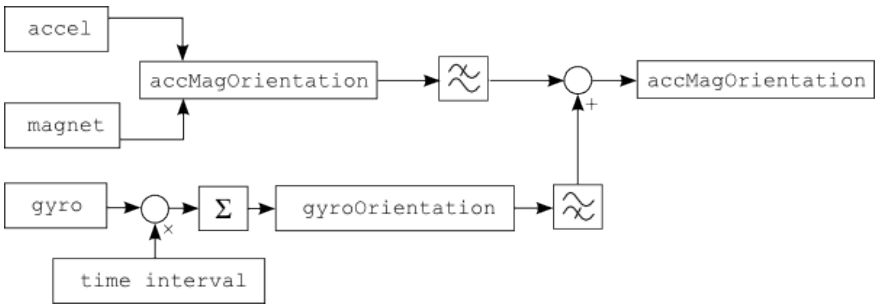


Figure 3.30: Complementary filter schema [Lawitzki, 2012].

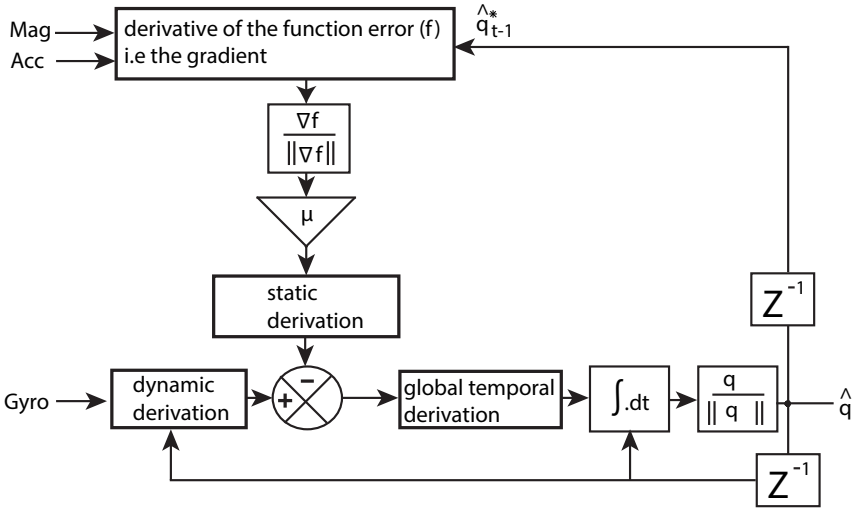


Figure 3.31: Gradient Descent filter schema [Sebastian O.H. Madgwick, 2011].

computing a derivative function (f in Fig.3.31) by a set of inputs. The function error represents the error between the measurements from accelerometer/magnetometer and an expected measurement ($\hat{q}_{t-1}$).
There are two specific parts in the algorithm: static and dynamic conditions. The first one is the static condition. Data from sensor are computed from body frame to the Earth frame. The expected measurement comes from the theoretical model of gravity and Earth magnetic field vectors, which are rotated by the estimated quaternion ($\hat{q}_{t-1}*$). The estimated quaternion is the orientation computed in the previous iteration of the algorithm.
The derivative of the function error is the gradient (f). This gradient holds the minimum's direction of the error's function. Then, the gradient is normalized and multiply by a specific coefficient (μ). The results is considered as a static derivation because the orientation or quaternion from the accelerometer and magnetometer is computed in the static condition. Then,

the gyroscope is used to compute a derivative quaternion, this is the dynamic derivation because we consider the gyroscope as the dynamic condition. Both quaternions are subtracted to get a global temporal derivation of the quaternion, which is integrated with the estimated quaternion. μ acts on the weight accorded to the static orientation and the dynamic orientation. The results is normalized to get the new quaternion, which will become the future estimated quaternion.
The results are pretty good [Sebastian O.H. Madgwick, 2011] because the magnetic distortion and gyroscope bias drift compensation have been considered in the Gradient Descent filter. The accuracy could be around 2 degrees on the azimuth in dynamic with a data rate close to 50Hz. The performance of this filter are enough in stable phase, but it converges slower than an Extended Kalman Filter. So, we also have to consider the EKF, which is explained in the following paragraph.

Extended Kalman Filter A Kalman filter is defined as a recursive solution to the discrete signal linear problem where the error between prediction and observation is minimized in the sense of the mean square error. It provides an estimate of the state of process in a way. This filter is used to predict parameters and correct errors on the prediction. The latter is based on sensor measurements. Indeed, to apply a Kalman filter, the system has to be modelled by parameters (F), which have to be estimated by a linear system. This filter provides a state vector (x_k the parameters to estimate) and also the error covariance matrix (P_k). The covariance matrix Q represents the noise of the dynamic model (F).
The Kalman and Extended Kalman Filter has two steps. The first one is the prediction.

- $\hat{x_k} = F.x_{k-1}$
- $\hat{P_k} = F.P_{k-1}.F^T + Q$

This step depends on the model (F), which describes the system's dynamic over time. To compute this step, the equations of the model are used to predict the new state vector ($\hat{x_k}$) and the new error covariance matrix ($\hat{P_k}$). The second one is the update.

- $x_k = \hat{x_k} + K_k.(y_k - H.\hat{x_k})$
- $P_k = (I - K_{k+1}.H_{k+1}).\hat{P_k}$
- $K_k = \hat{P_k}.H_k^T.(H_k.\hat{P_k}.H_{k+1}^T + R_k)^{-1}$

This step is based on the data from sensors. With the parameters from the prediction step, the estimation of the data sensor (H) is computed. The difference between the estimated and real data sensors is used to update

the state vector (x_k) and the error covariance matrix (P_k). The covariance matrix R represents the measurement noise and the matrix I is an identity one. This step is considered as the static observation, as opposed to the prediction step (dynamic evolution).

When the orientation is computed with MEMs, the function F is not linear, so there are few solutions in order to overcome this problem. One of the solutions is the Extended Kalman Filter, which is based on a linear approach by sections (i.e. linear between only two samples). In [Bijker and Steyn, 2008], [Brigante et al., 2011] the state vector (x_k) contains the orientation (a quaternion) from the gyroscope and the gyro bias vector (dynamic evolution). The sensors data (y_k) come from the magnetometer and accelerometer (static observation).

The complexity and the accuracy vary between the three filters, In our context the HW is accessible so can be calibrated, this step is crucial to reach accurate estimates. In addition to an accurate calibration on the magnetometer [Guo et al., 2008] and the accelerometer [Frosio et al., 2009], the use of an Extended Kalman Filter can provide a dynamic accuracy of 2 degrees on the azimuth angle with a data rate close to 30Hz. Even if the computation is more complex than the other filters, this is the most accurate and the solution to keep on the prototype. An EKF has been implemented on the prototype but the calibration algorithm are still under development. With the three filter, it's easy to get a better accuracy by improving the data rate but the power consumption increase and this is very bad for battery life, this is necessary to keep a data rate around 30Hz to limit the latter.

3.4.4 Conclusion

Services are provided to the user depending on the devices available: GPS, AIS, Navigation processor and 3D charts. We have designed the application to find first the services and get a list of the available ones in order to let the user select the services corresponding to the navigation context (regatta, navigation). To provide the geographical data (navigation aids and depth areas) in the user's filed of view, we have implemented few tasks. The geographical task is able to convert GPS data to an UTM projection and an orientation task provides the head movement to the application in order to display the graphical elements on the right place. In this way, the user can see buoys position and prohibited areas independently from the weather conditions, such as fog or night.

3.5 Implementation on main stream AR system

3.5.1 Introduction

Even if our first prototype has been designed on the OST AR ski mask from Laster Technologies and our own hardware platform, it is necessary to be open to other devices in order to access the main stream market.
In main stream AR devices, there are two kinds of displays, the OST like AR glasses and VST (video see through), like smartphones/tablets. The mobile devices run mobile OS such as Windows, iOS or Android but the most used in these devices is Android. This is first reason why we decided to port the application on this OS. Second, the implementation of MARNAA on Android is made easier by the fact that both OS share a Linux Kernel. However, Android has higher level of abstraction, with managers, which allow an easier use of the components. The functions are less optimized, this increases computation, so the main drawback is the increase of the power consumption.

3.5.2 Tasks

3.5.2.1 OS

On Android, we cannot use exactly the same software tasks and architecture like in a Linux OS. Indeed, a classical C/C++ application on Linux, as MARNAA, is programming with a main thread and other sub-threads implemented with the Posix API [4] for instance. But we cannot use exactly the same way because Android imposes the programming of specific applications with rules. An application can be written with one or more activities with the lifecycle [5] visible in Fig.3.32. There is a main thread named the GUI thread which cannot be blocked, this is the one used to communicate between the user and the application. So it induces some modifications on MARNAA to be runnable on this OS even if it is a Linux kernel. On Android, there is a SDK with API and Java class but there is also a Native Development Kit that allows to use native language (C/C++) to code functions. This is very useful because it lets the programer reuse some functions already coded in this kind of languages. And the porting is possible thanks to the NDK because the 3D engine and all tasks are coded in C/C++. The communication between JAVA and C++ class for instance is possible with the Java Native Interface integrated in the Java Development Kit that can be used to program Android application.

In the following sections, we don't detail all the modifications applied on MARNAA but only the most important.

[4] http://pubs.opengroup.org/onlinepubs/9699919799/
[5] http://developer.android.com/training/basics/activity-lifecycle/starting.html

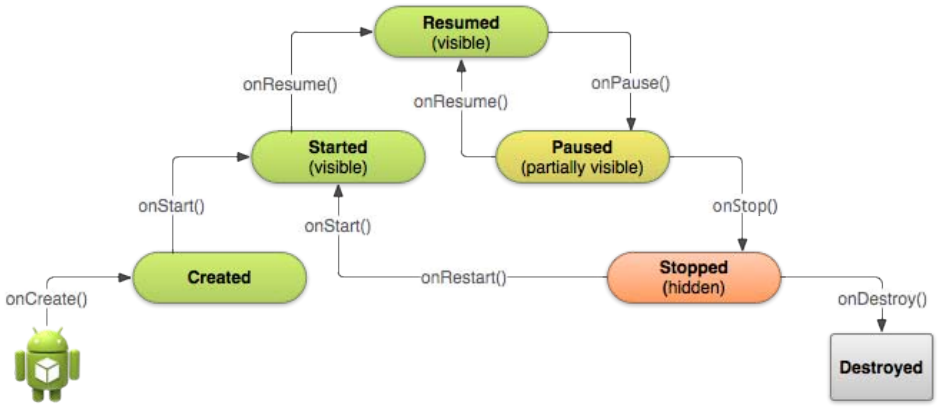


Figure 3.32: Android activity lifecycle

3.5.2.2 Graphical

There are important differences between an OST or a VST display. The first one is able to display information on the real world view, without any camera and in the user's field of view whereas the second one needs to catch pictures from a camera and adjust them on a display. It is necessary to detail which are the specific modifications to implement on the application to get the same as the one running on the ski mask from Laster Technologies.

OST display All main stream glasses have different display characteristics as shown in Table 3.2. So, if we want that MARNAA can display correctly the information on these displays, we must take into account the FOV and the resolution for each of them. But sometimes the manufacturer data are not exactly the same on the device, consequently the rendering from the application is not the good one. A calibration should be done on the glasses to adjust the FOV because it impacts the perspective, so the size and position of the virtual objects will be wrong. There are few methods to calibrate HMD display [Tuceryan and Navab, 2000], [Genc et al., 2002], [Grubert, 2014], [Wientapper et al., 2014] for the monocular and binocular glasses. There are mainly used to correct the misalignment between the camera the display and the eye. In our case, the camera is not used so, these methods are not adapted, a specific calibration has to be put in place in an empiric method. The solution is a modification in real time of the field of view and the rotation matrix 4x4 (see in Fig.3.29) to get the best perspective and object size compared to the real ones.

VST display In the case of VST display use (required for tablet applications), the real world is not visible directly through the screen but through a video stream. This technology induces the use of a back camera to capture picture and match camera parameters with the display, the physical and

virtual camera. The settings on a physical camera are the resolution, frame rate, focus, aperture and so on but the most important in our context is the resolution and the Field of View.

The screen resolution is often different than the camera one. For instance, the picture resolution is up to 1080p and screen resolution is only 720p on a Samsung Galaxy S III. The picture doesn't match the screen, so the picture have to be cropped to avoid a deformation on the picture. This operation selects a centered area and looses some information to display the picture on the screen resolution.

We need the FOV from the physical camera to match with our virtual camera in the 3D engine to get the same perspective and to superimpose correctly the real and graphical objects. This field of view in a smart-phone camera depends on the size of the camera sensor and the focal length of the lens. The FOV can be computed with the following formula:

$$\alpha = 2 \arctan d/2f$$

where d is the sensor size of the camera and $2f$ is the focal length. Some calibration process are normally needed to get the focal length and sensor size to retrieve the FOV, but since the version 8 of Android API, both values are directly readable. For instance, the Samsung Galaxy SIII has a screen's resolution of 720p, its vertical FOV is 49.3°. These data are necessary to get characteristics for the rendering in the 3D engine.

Once the physical camera parameter are known, there are some adjustments on the 3D engine to get a graphical interface proportionally to the display resolution. If theses characteristics are correctly used in the 3D engine, then the augmented objects are exactly the same as the one used in the prototype, the video is just mixed with the augmented objects as in an OST device.

3D engine The 3D engine Irrlicht has been ported on Android with the OpenGL ES API. Only few options are necessary to specify the OS, graphical API and tools before the compilation to get the Irrlicht 3D engine working on Android or embedded Linux. All functions are exactly the same, GUI, loading meshes, fonts, etc. But depending on the device, the resolution is different so we have to take into account their features in order to place all the graphical elements (text, logo) and re-size them.

3.5.2.3 Decoding

In the case where the GPS is used on the mobile device, Android simplifies the use of this chip with the help of a LocationManager. This manager is able to directly provide position coordinates, speed, and orientation but we can also reach the raw data (NMEA sentences) from the GPS.

In the first case, most of the decoding functions (for the GPS) can be replaced by the use of the LocationManager class and JAVA code. Once the data are acquired, the decoding task sends data with JNI to both, the data structures and the 3D engine. In the second case, only the management of the GPS (run, stop, settings) and the acquisition of raw data used the LocationManager. The decoding phase reuse the same codes than the Linux version thanks to the NDK and JNI API.

When the NMEA devices (AIS, GPS, etc.) are accessible by the boat network with a wireless connection, we have to use Android APIs, which provide Java class to manage the WiFi and the Bluetooth: WiFi Manager and Bluetooth Adapter. If we compare the Linux and Android version of these C++ and Java class, some modifications on the code are necessary to keep an access to the NMEA data. The Android class simplifies the use of the wireless connection but it induces the biggest modifications in the porting of MARNAA on Android. There are two solutions to implement a WiFi or BT connection, the programming in JAVA or the use of JAVA class with JNI. But in two cases it induces programming.

Once the connection is established between the device and the boat network, the decoding tasks (see Fig.3.19) has to sort, decode and store data in the different data structure (see Fig.3.19). So, the main advantage to get the application running on Android OS is the use of JNI because the tasks such as the decoding, collision detection and coordinate transformation tasks can be exactly the same as the one in the Linux version.
The last task that needs some modifications is the orientation task, this one cannot be completely port on the Android OS, the main modifications are explained in the following paragraph.

3.5.2.4 Orientation

In Android, a specific Java object has been implemented to manage the sensors: the SensorManager. This is a JAVA class to let an easy control on sensors such as the power on/off start/stop, sleep and get data. This SensorManager is able to provide to the programmer the orientation from MEMs with JAVA functions, but the parameters on sensors are limited. Indeed, there are only four modes for the data rate on the sensors: UI, normal, game, fastest and the speed is not provided, it can be different from one smart-phone to the other if sensors are not the same. To get more precisely the real value of the data rate, the developer has to use timing functions to compute time between two data. This is not exactly the same abstraction level as a sensor driver: this solution is easier to use but less precise, in the Linux version we are able to set exactly the data rate of the sensors written in the datasheet.

There are many ways to acquire orientation data on Android, the raw data from sensors, the Euler angles, the rotation matrix to apply directly in

OpenGL ES but all theses data are not filtered, the still require a Kalman or complementary filter for instance. Once again, the use of JNI offer the possibility to keep our filters written in C++ and we chose as a default value fastest data rate to be sure to get enough data to follow user's head movement. The filter's output sends event to the graphical task to update camera orientation in the same way as the Linux version.

3.5.3 Conclusion

Our ambition is the porting of the application on the Android main stream market devices, which are able to run AR applications based on IMU and GPS. There are two kinds of devices, the VST and OST, first one needs a camera whereas the other one doesn't need one, so two profiles of application on Android have been developed. The implementation of tasks without any relation with sensors are the same as the Linux version, the 3D engine is also identical. The main modifications are the management of sensors, wireless connectivity and few changes to meet Android OS rules (activities) and currently we are working on the WiFi and Bluetooth part to finish the management of services on the Android version.

3.6 Conclusion

In this chapter there are three parts, a presentation of our own hardware prototype, the list of the software tasks and the application design and the main modification to port the application on Android OS.

The prototype is composed by an Optical See-Through ski mask from Laster Technologies and an home made board. We use a SoC as the main part of the hardware architecture to get enough computing capacities on a CPU and GPU to run all the software tasks required and to limit time to develop. The orientation is computed with data from an IMU in order to access it in any weather conditions (fog, night, sun). With a Extended Kalman filter and calibration phases, we are able to get an angle error close to 2 degrees. In addition to the orientation, the position is accessible from a wireless connection or latter (pre-industrialisation phase) with an onboard chip. The hardware is able to display in the user's field of view text, 2D and 3D data with the software tasks detailed in the second part of this chapter.

The application has been designed a service server to get the most flexibility to the user and limit the complexity of the application. The services is a method to preselect the sensors and data to use, latter some adjustments on the kind of data will be possible. The use of a 3D engine simplifies the management of the data and the use of the graphical library to render the frames.

So if we sail, the user's view is the one represented in Fig.3.33.

Figure 3.33: Real view in Lorient harbour from a rigid-hulled inflatable boat.

Then the 3D engine updates the position and orientation of the virtual camera in a 3D world with the GPS data and IMU data, as it is represented in Fig.3.34.

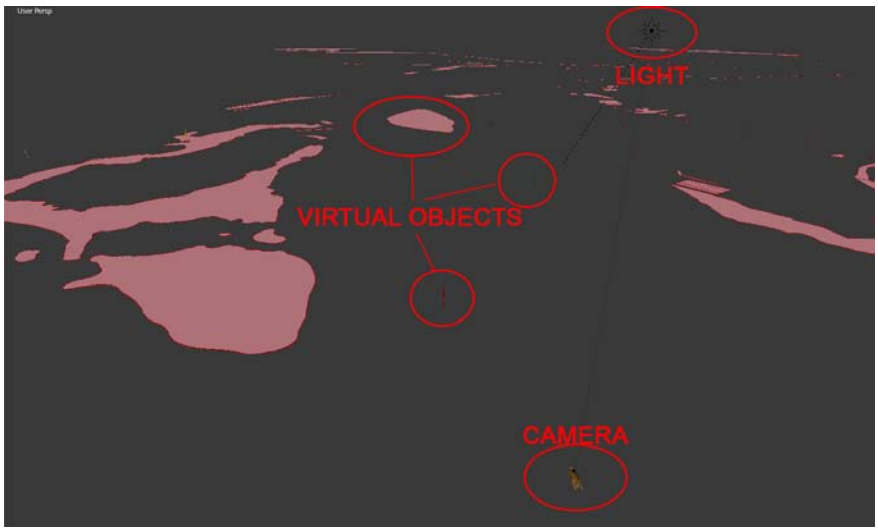


Figure 3.34: 3D scene of Lorient Harbour

Finally, the 3D engine runs commands with the Graphical library on the GPU and render frame as it is represented in Fig.3.35. In the latter, the front of a boat is visible to understand the analogy between both Fig.3.33 and Fig.3.35 but the real application render frames without the boat. More-

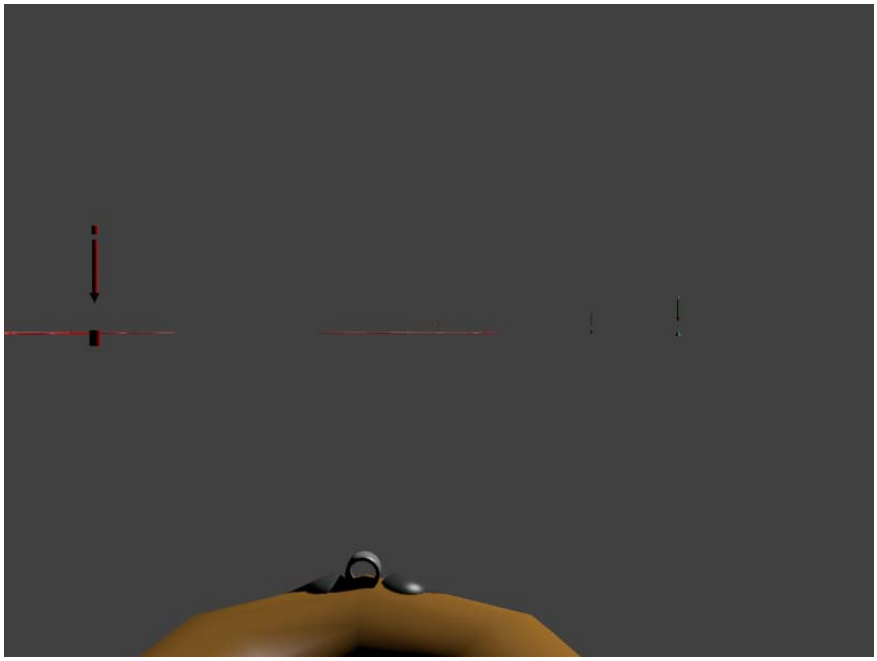


Figure 3.35: OpenGL renderer with 40° as a FOV and a resolution of 800*600 (like Laster glasses).

over, the gray background is normally black and the black colour means transparency in an OST display.

Then we decided to port the application on main stream devices (AR glasses, smartphone and tablet) to be more visible. In case of Video See-Through display, the addition of the camera is necessary and the camera parameters must be taking into account to get a cohesion between the 3D objects with real view. For both, VST and OST, the OS imposes rules, so the main modifications have been implemented to get a runnable application.

The next step is the management of the Electronic Navigational Charts data to feed the application: sort, decode and generate 3D data in order to satisfy user's expectations (arrow, depth area, etc.). This part is detailed in the following chapter.

Chapter Fourth

A 3D chart generator

Contents

4.1 Introduction

To facilitate the management of the 3D objects in the application we have decided to use pre-built charts. It is very time consuming to read GPS positions, load corresponding objects and place them manually in the 3D chart because there are few hundreds objects by charts. The last task would take many hours even for a small area, such as a harbour approach for example. An other key point is the updating of charts, they are regularly updated so it is crucial to automate it. The easier and faster solution to generate charts is the use of a 3D software with scripts, but this kind of software is only available on PC. So, we decided to create the 3D charts outside the mobile device: our solution is an offline generator for MARNAA application to build 3D nautical charts.
Our chart generator has five steps shown in Fig.4.1 and listed below:

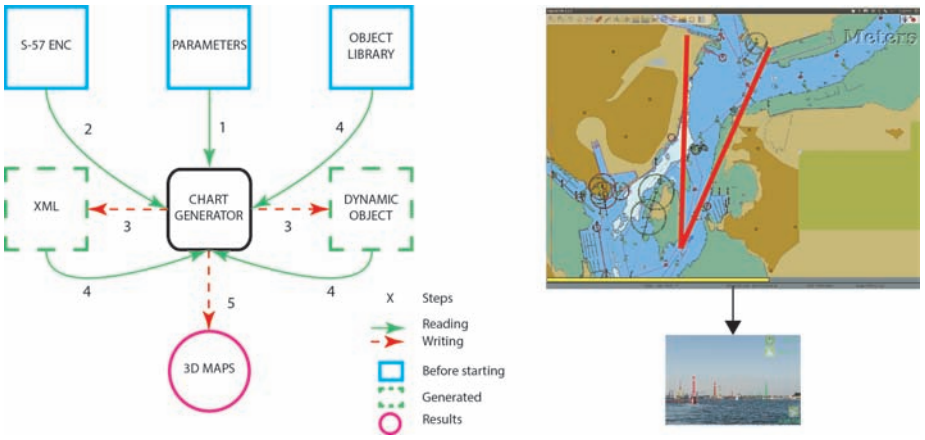


Figure 4.1: 3D chart generator schema

- Step 1: Read the list of parameters to keep.
- Step 2: Decode the input files.
- Step 3: Record the objects as text in a XML file (list of objects to draw with their positions and attributes...).
- Step 4: read all the objects stored in the XML file, and loads static objects or generates the danger areas (dynamic objects).
- Step 5: export the 3D chart in a 3D format file.

This generator has two main tasks: filtering and generation. In this chapter a section state of the art sums up some approaches to generate 3D charts. Then we will detail the filtering and generation phases in Section 4.3 and

Section 4.4. In the latter, some results will be presented through 3 charts provided by the SHOM from the Lorient harbour approach to the bay of Biscay. Finally we will conclude in Section 4.5.

4.2 State of the art

Chart generation Our problematic is the building of a nautical chart with the graphical elements chosen by the users. This is the same challenge as the Simultaneous Localization and Mapping (SLAM). SLAM is a method using a robot that maps its environment, simultaneously approximating its own position on that map. Both the position and the map are unknown at the beginning of the navigation. This approach is mainly used with drone or robot navigation in buildings, outdoor or under the sea for instance.

The AR display could be considered as a SLAM problem [Skoglund, 2011] where the IMU and the camera are the sensors to get information on the environment around the user. The boat with the GIS data could also be considered as a SLAM problem [Agarwal et al., 2014], where the seamarks and landmarks could be used to generate in real time the chart. In order to solve SLAM problem in real time, some solutions are provided [Grisetti et al., 2005], [Grisetti et al., 2007] and one of the solutions is accessible as an Open Source API [1].

But there are few drawbacks. Firstly, the SLAM resolution generates detailed charts and this is not our objective. Secondly, the boat moves in a complex environment (weather, swell movements), so the camera is not always usable to acquire environment details. Fourth, this method can't be applied in our context, we have to provide useful information to the sailor from the first navigation in an unknown place, so the user must have a pre-build chart.

A solution that could be related to the SLAM problem is the use of charts generated by a community, where the users are considered as robots: Openseamap[2] is an example of nautical charts automatically built with users. This solution is not the good one, it is in the same way as the use of a classical chart. We need another steps to make the solution usable in MARNAA application.

Our solution is based on video games, where the 3D charts are built before the application. Charts in video games can be very complex but in our context, only seamarks, landmarks and depth areas are necessary. So, we can use data from charts to get the identification and coordinates of the objects to draw, then with a 3D library like in video games, generate the 3D chart. The new data which appeared when the application run are only the boats data from AIS. These data are considered like enemies in video games,

[1] https://www.openslam.org/gmapping.html
[2] http://www.openseamap.org/

where the 3D model is stored and load when both the virtual camera and enemy positions are close. There are 3 steps in this solution:

- Find and extract data from a chart.
- Build 3D objects.
- Generate the 3D charts.

The first step is the extraction of data from a chart and the following paragraph details the two main formats: raster and vector charts.

Charts There are two kinds of charts, the raster charts and the vector charts. The first one is an historic format, generally the raster charts are pictures from original paper charts, scans or photos that can be assembled to obtain a complete digital chart. A view of a digital raster chart from the SHOM is visible in Fig.4.2, this is the Lorient harbour.

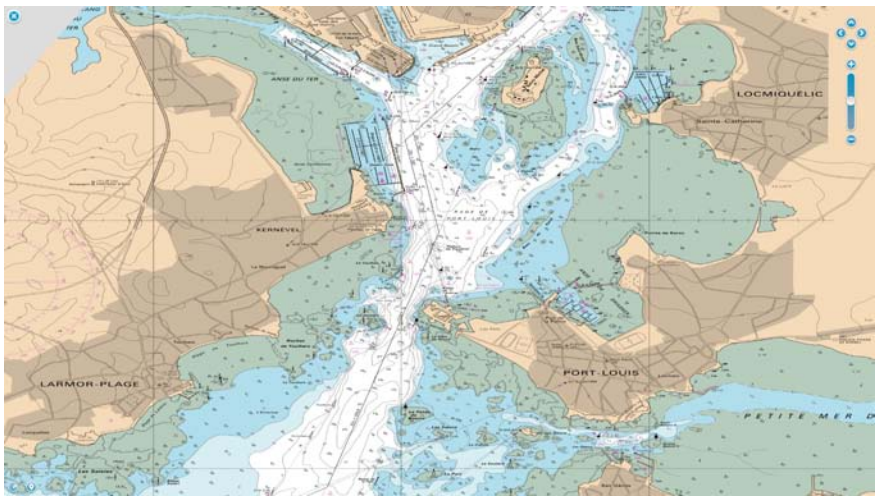


Figure 4.2: Raster chart of Lorient harbour (SHOM)

Unlike raster charts, the vector charts are full digital ones, generally there are in text format with a coding format or there are encrypted. S-57 is a coding format common to the hydrographic centre, whereas CMAP v2 for instance is an encrypted one. A view of a digital vector chart from the SHOM is shown in Fig.4.3, this is a S-57 file loaded in a navigation software and represents the Lorient harbour, like the previous one. There are about 200 kinds of different objects in a S-57 chart for instance, so we need to filter information to keep only the useful data.

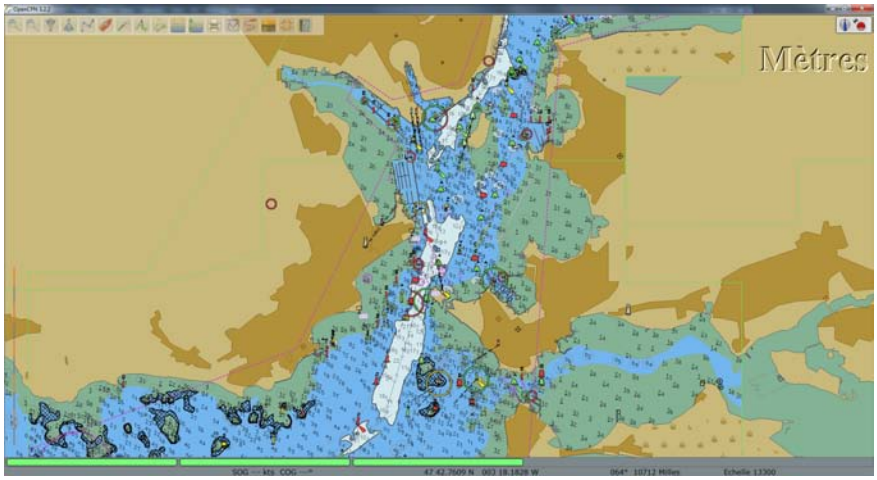


Figure 4.3: Vector chart of Lorient harbour (SHOM)

4.3 Data Filter

4.3.1 Introduction

The objects, to display in the application, depend on the weather conditions, user experience and the boat type. Indeed the needs between a jet-ski and a sailing boat (recreational boats) for instance are very different. So, it is important to get a flexible generator because we need to import several kinds of input files (different vector chart formats), to satisfy all kind of boats and conditions. To validate the whole system we have used the S-57 format because they are certified by the IHO.
The 3D chart generator has been divided in two main phases, the filtering and the generation phases, as it is shown in Fig.4.4. In this section, We describe the filtering phase through 3 steps:

- Step 1: filter the data.
- Step 2: extract information from S-57 file.
- Step 3: use an intermediate format between the filtering and generation phases.

So, the following section starts with the filters used in the generator.

4.3.2 The input filters

4.3.2.1 Input files

S-57 The Electronic Nautical Chart S-57 is a vector-based cartographic database, which contains a detailed description of every object (such as sea-marks, shipwrecks, leadlines, sector lights, moorings, etc.) visible in Fig.4.5;

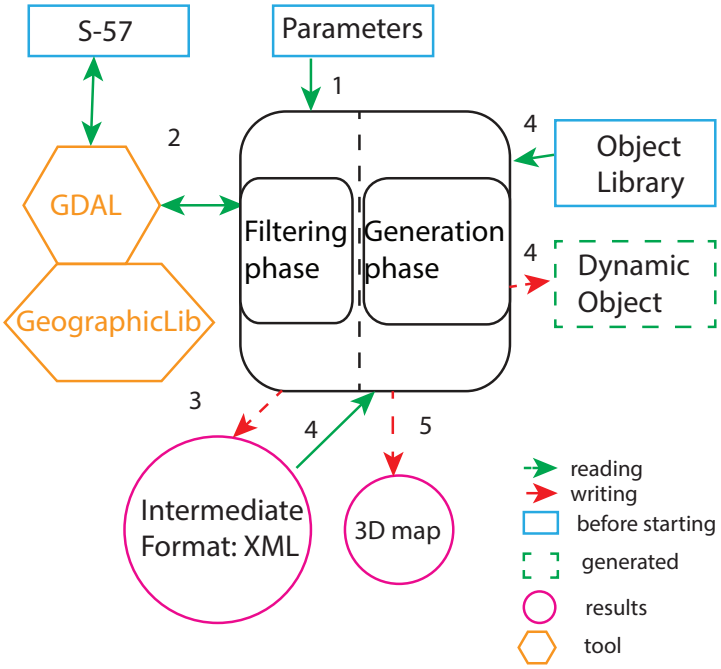


Figure 4.4: 3D chart generator schema divided in two phases

it is the digital equivalent of a nautical charts. It's possible to organize and

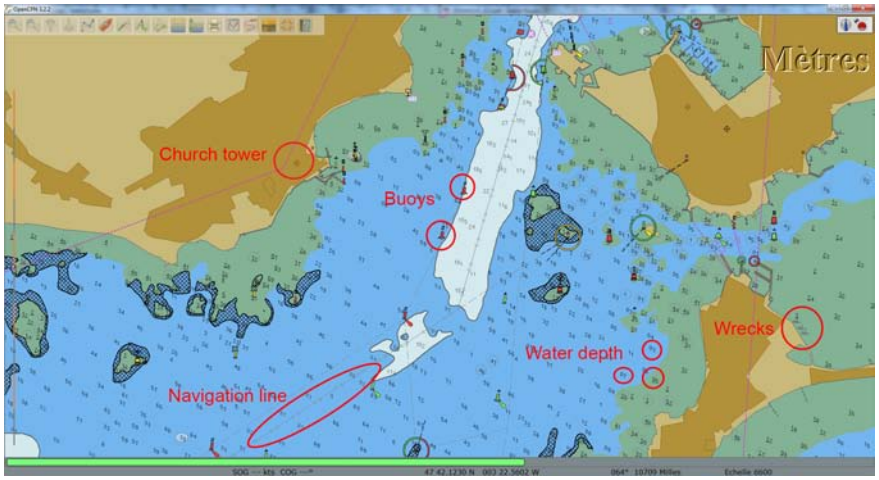


Figure 4.5: S-57 Electronic Navigational Chart from Lorient harbour

display information according to a certain area or a navigation mode and also with more accuracy and personalization. It also has a low memory footprint, around 2 MB for a chart. Furthermore S-57 ENC are classified in 6 categories according to their scale:

- Cat 1 : Global view < 1 : 1 500 000
- Cat 2 : General 1 : 350 000 - 1 : 1 500 000
- Cat 3 : Coast 1 : 90 000 - 1 : 350 000
- Cat 4 : Approaches 1 : 22 000 - 1 : 90 000
- Cat 5 : Port 1 : 4 000 - 1 : 22 000
- Cat 6 : Mooring > 1 : 4 000

FR602190 for instance is a S-57 chart from France in category 6 (Lorient harbour chart). The object's list is defined in the standard and visible on the web site[3]. An object is an association of two elements: a feature and a spatial object. Both are presented in two next paragraphs.

Feature Object A feature object contains the descriptive information, there are about 170 features objects classes defined in the standard S-57. These feature objects are defined by a set of attributes, that correspond to the description of an object and there are about 190 attributes classes that are defined in the standard S-57.
The S-57 data model defines four types of feature objects: the Geo, Meta, Collection and Cartographic types. The 159 Geo objects contain the descriptive characteristics of real-world entities (depth areas, land areas, beacons, buoys and so on). The 13 Meta objects contain information about other objects, the 3 Collection objects describe the relationship between other objects and the 5 Cartographic objects contain information about cartographic representation of real-world entities. All the feature objects have a unique 6-character codes: DEPARE (depth area), LNDARE (land area), BOYLAT (buoy lateral), LNDMRK (landmark) for instance and they are linked to the spatial objects.

Spatial Object A spatial object contains the position information with the latitude, longitude and depth for instance. It can be represented by a point, lines or areas and it is used to indicate the location of a feature object.

As an example, the navigational aid "Le pot" is a BOYLAT (lateral buoy) describes as follow:

- 47 42.7030 N 003 22.0380 W.
- BOYSHP: 2, can (cylindrical). This is the buoy shape.
- CATLAM: 1, port-hand lateral mark. This is the category of lateral Mark.

[3] http://www.s-57.com/

- COLOUR: 3, red. This is the colour of the buoy.
- OBJNAM: Le pot. This is the name of the buoy.
- MARSYS: 1, IALA A. This buoy is in the region A of IALA buoyage system.

A BOYLAT feature can be described with more attributes (about 22) but the BOYSHP, CATLAM, COLOUR, OBJNAM, MARSYS and the position are the main useful attributes for representing the object in the new 3D chart.

The first input filter parameter corresponds to the list of objects to keep in the 3D charts such as buoy, beacon, depth area and so on. To conserve a consistency in all filter steps, the S-57 acronyms are listed and used as inputs in the generator. Table 4.1 is a data filtering example from a nautical charts (S-57). This is a restricted list related to seamarks, landmarks and areas

Table 4.1: Selected objects in S-57 ENC

Maritime mark	Meaning	Acronym in S-57 chart
Lateral mark	way to go or leave harbour	BCNLAT / BOYLAT
Cardinal mark	way to avoid danger	BCNCAR / BOYCAR
Isolate danger mark	danger under beacon/buoy	BCNISD / BOYISD
Special mark	anchorage areas, pipelines, cables	BCNSPP / BOYSPP
Landmark	water tower, church tower,...	LANDMRK
Under water rocks	underwater rock, awash rock	UWTROC
Depth area	depth area between 0m to 5m	DEPARE

but the list can be adjusted depending on the needs. We consider that this example represents the minimum navigation aids to keep in the application to help the users in their navigation, as it was presented in Chapter 2. The other input parameter or filter is the profile, depending on the uses or the user, the information to keep can vary. In the following section, we detail how the profiles can act as an input filter on the 3D chart generator.

4.3.2.2 Profiles

The user study, detailed in Chapter 2, has highlighted different profiles depending on the weather (the scenarios) and the situations (the phases). The users don't need the same information everywhere and all the time, so this is not a good solution to generate one chart for a specific scenario as the fog, night or when the visibility is good. It will be painful for the users to download charts for all the different situations (scenarios and phases), thus we propose to generate one 3D chart for one digital chart. With this approach the user will use our application as their classical navigation software. The

weather conditions and phases will be considered as parameters in the application and not as a filter in the generator to limit the number of generated charts.

The use of a 3D engine (see Chapter 3) provides some opportunities to easily select only the useful 3D objects in a set of 3D files. The user's studies have conducted to a list of data and profiles. But, to provide flexibility in the application and limit the number of charts to generate and download, the generator has to take into all their cases in one pass. So, this solution induces the generation of multiple 3D files and storing them in a folder tree considering the arrows, topmarks, names for the navigation aids and the shapes of the depth areas.

In the application, the depth areas have to be adjusted depending on the draught of the boat. Indeed a recreational boat can be a rigid-inflatable boat with a draught about 30 cm or a sailing boat with a 3 meters draught. This parameter will be asked to the user in the application's user interface to adjust the displaying of water depth areas. Like the navigation aids, to get a generic chart for many recreational boats, the filter phase must takes into account depth area between the submerged area to the emerged with a height of 5 meters for instance. But it is very important to separate the depth areas in few categories depending on the DRVAL1 and DRVAL2. These attributes are the depth range of an area and in our case, they refer to the feature DEPARE, which is depth area (see in Table 4.1) in the S-57. Indeed we have to separate DEPARE every meters or half meters to adjust, for the different boat's draught, the displaying of the danger area and provide the best accuracy for each kind of boats. This approach gives more flexibility to the application and requires only one 3D generator run.

The navigational aids and depth areas objects are essential for most of the users but the generator is also able to address specific cases depending on the users needs. It means that we have to be able to add new features and attributes, the user may ask for, so the generator must have the ability to consider them. As an example, some sailors are practicing diving and this activity is prohibited in few areas as near the Lorient harbour entry and it is generally practiced around the wrecks. The location of the prohibited area (as danger area) and the wrecks position (as navigation aid object) can be added easily in the generator, but currently, there are not generated as default.

In the first step of the filtering phase (see Fig.4.4) is the reading of the object list, we decided to organize the latter in the same way as the feature object and attributes in the S-57 file: we use the unique 6-character in S-57 specification. The next step is the decoding and the extraction of the data from the S-57 charts (steps 2 and 3 in Fig.4.4 and they are detailed in the following subsection.

4.3.2.3 Extraction

The extraction step (step 2 in Fig.4.4) is made possible thanks to specific libraries:

- GDAL (Geospatioal Data Abstraction Library)[McInerney and Kempeneers, 2015].
- GeographicLib[4].

GDAL is vital in the filtering phase to decode, extract data and GeographicLib applies the UTM projection on the object coordinates. The extraction phase couldn't be applied without these libraries.
First, GDAL is able to read most of the Geographic Information System file formats such as S-57, GPX, shapefile and so on. This function is very important for our generator in order to take into account most of data related to the maritime domain. This library answers our needs and provides flexibility to the generator. Indeed, it includes all steps for model transform, read, filter, extract data from many Geographic Information System (GIS) files and allows the programmer to save them in another format file.

Second, the GeographicLib is necessary to save object's position in a new coordinate system. The library is able to proceed few types of coordinate systems such as ECEF, ENU, UTM in a WGS84 geodetic system with GPS data. We use it because the generating phase can't work with data in a degree format, this step was detailed in Section 3.4.3.4. This library meets our needs for the specific step of coordinates transformation.

So, the extraction is based on three steps. The first one is the reading of the features to keep, only the interesting features, listed as an input of the generator. The second one is the attributes sorting, to retrieve the feature specification and finally, the last step is the writing of these information in a file. In the latter, the generator writes a XML file that stores the features and the attributes selected (step 1 in Fig.4.4). This is an intermediate file format between the filtering and the generation phases (see Fig.4.4). It conserves only attributes related to the shape, position, name and some practical features such as the minimum and maximum depth. The goal is to limit the number of attributes in order to maintain a generic XML file skeleton, which is common to different input formats.

In box 9, we present an example of a lateral buoy (type is BOYLAT) that is compliant with our XML file format. The first tag indicates a new object in the XML file, then there is a list of subtags. The first one is used to get the object's acronym, this is the "type" tag. After the tag related to the object's acronym, the UTM position is stored in the tag "point" and its UTM zone in the tag "zone". The other tags specify feature object indications: "Le Pot" (name = Le Pot) is a port-hand buoy (category = 1) with a cylindrical shape

[4] http://geographiclib.sourceforge.net/

```
<object>
    <type>BOYLAT</type>
    <point>472449,5284325</point>
    <zone>30N</zone>
    <category>1</category>
    <shape>2</shape>
    <colour>3</colour>
    <name>Le_Pot</name>
    <region>1</region>
</object>
```

Box 7: BOYLAT

(shape = 2) in a red colour in the Region A of IALA standard (colour=3 and region=1). The category and the shape are enough to retrieve the right 3D object corresponding to the IALA representation. If the input format file was different, the position, name and buoy shape will be the same, these information are essential to understand and display later the right object.

In box 10, we present the XML representation of a depth water object (type = DEPARE), which is in the same UTM zone as the lateral buoy but the spatial object is a set of GPS point converted with UTM projection that describes the contour of the area. A depth water object is defined by a minimum and a maximum as a depth range in the case of a low tide. Those values represent the level at the lowest possible astronomical tide, a minimum equals to -7 meters and maximum equals to 0 meter for instance.

```
<object>
    <type>DEPARE</type>
    <point>...</point>
    <zone>30N</zone>
    <min>-7.0</min>
    <max>0.0</max>
</object>
```

Box 8: DEPARE

The generator writes in the XML file all the objects with the same acronym and moves to the next one. This method facilitates the execution of the generation phase when the input files are different such as the GPX format file for instance. The following paragraph details how we can

use a different format file in the generator through the GPX file example.

Open formats The GPX format[5] is one of the most used open format, it is often used to exchange GPS data between applications. A lot of mobile applications on smartphone for instance used this format to save waypoints or tracks. The GDAL API allows us to load and extract data from a GPX file. We have decided to use an intermediate format file to limit the changes into the generator. We just have to add new filters entry (step 1 in Fig.4.4) and store them into the XML file. Another specific task is required, the design of the new 3D object to display on the AR display. We propose to explain our approach with an example.
The GPX file can be used in the regatta context to get specific data from a committee for instance in order to help the sailor in its sailing. The Fig.4.6 is a picture from a video stream displayed during an America's cup regatta. In addition to the video stream, augmented data are added to show more details on the race and boats. In this picture, there are two kinds of data, carto-

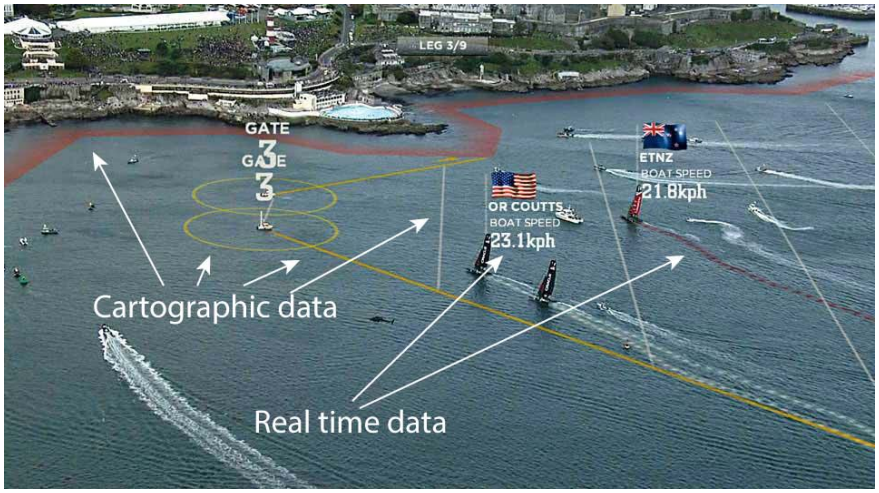


Figure 4.6: The Augmented Reality America's Cup television view

graphic data and real time data. Information related to the cartographic data are the buoys and the danger areas (red lines), which represent the prohibited areas due to the weak water depth. The white and yellow lines (see Fig.4.6) are used to represent the navigation limits and the distances between buoys. For this kind of regatta, the generator is able to generate the 3D chart with the cartographic data if they are stored as a GPX file. The buoys are represented by waypoint or specific buoys, which can be easily added in the generator and the lines could be TRACKS or another new FEATURE. Once the 3D chart is generated, the application is able to dis-

[5] http://www.topografix.com/gpx.asp

play the same augmented data as the one visible in Fig.4.6 in the mobile device to help the sailor during regatta.

The second kind of data, the real time data, are extracted from the GPS or AIS. These data cannot be used in the generator because the 3D chart is generated offline. The track, speed and course of the boat can be displayed in the same way we currently add information of the user boat. The difference is the location of the information on the display that is computed at runtime according to the distant boat GPS data. There are plenty of cases like the regatta one, the sailors have many activities such as regatta, fishing, diving, etc.

The flexibility of the GDAL library and the generator allow us to treat most of the maritime data used in nautical activities in addition to the maritime data from the ENC. Moreover, this method provides opportunity in other domain such as ski, cycling, walking. Indeed these activities are based on GPS data and information could be added for user's performance and security if they are stored in a open or GPX format for instance.

4.3.3 Conclusion

The filtering phase of the generator is represented in Fig.4.7. In the Electronic Navigational Charts there are too much information, which doesn't correspond to the useful data for a recreational boat, this is why we first filter the data with specific acronym for objects and their attributes. Then the GDAL library is used to decode, find and extract the data. Then they are transformed into an intermediate format (XML). The coordinates are projected into UTM format with the GeographicLib and all the data are stored in a XML file. The filter format and the use of GDAL provide flexibility to add or remove objects, attributes. Then the intermediate format approach is used to limit the modification in the generator when the input file is different than the S-57. Indeed, only few parts of the generator have to be changed in the filtering phase. Only the 3D representation of the objects have to be designed in the generation phase because both are independent. The second phase of the generator (generation phase) is detailed in the following section.

4.4 Chart generator

4.4.1 Introduction

The generator is composed of two phases shown in Fig.4.8. In the generation phase, the intermediate format file is an input and provides enough information to generate the 3D chart. The generator only need a library of 3D object corresponding to the real object (buoys) or augmented objects (arrow, topmark, etc.). To let flexibility in the application, each object type has its own 3D file for a given chart from the SHOM for instance. These

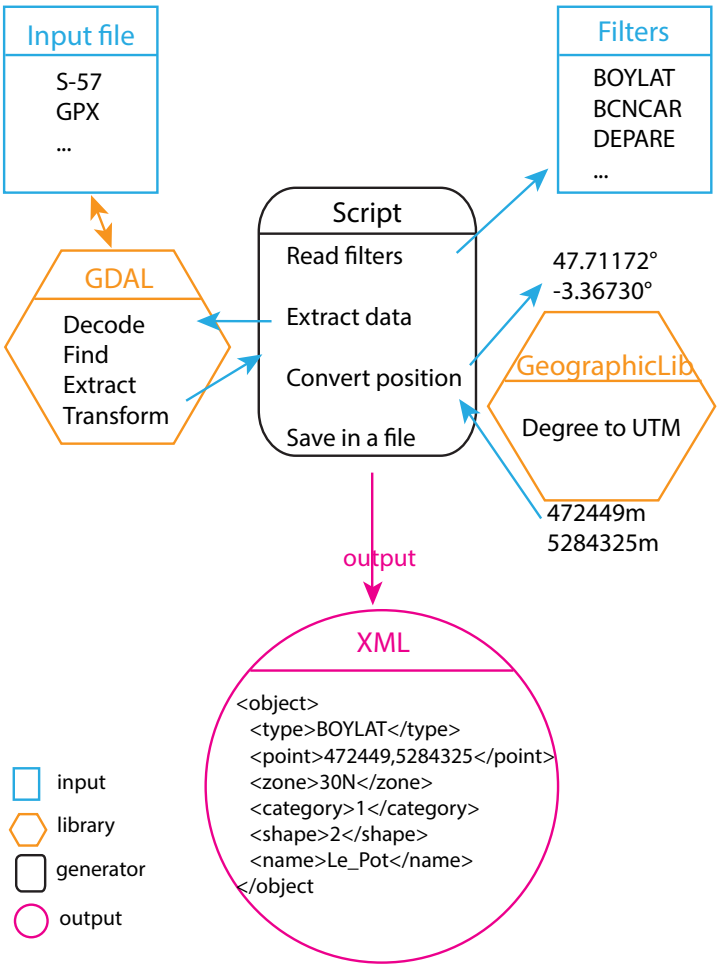


Figure 4.7: Schema which represents the first script of the generator: filtering phase

files must be generated using scripts to limit the time to create the 3D files, indeed loaded and placed one by one the objects with UTM coordinates is very long and painful. So, like in the filtering phase, the generation phase is based on a specific tool: a 3D software. We use Blender [6] to write and run some scripts with the python API. This tool provides functions to place, scale, export, import, etc. 3D objects, this is a free and open source 3D graphics and animation software.

In this section, we first detail a pre-study of the 3D chart memory size. This step was done to check if the approach is compliant with the mobile device's memory. Then we detail the management of the navigational aids and depth areas in the generating phase and some results are shown with

[6] www.blender.org

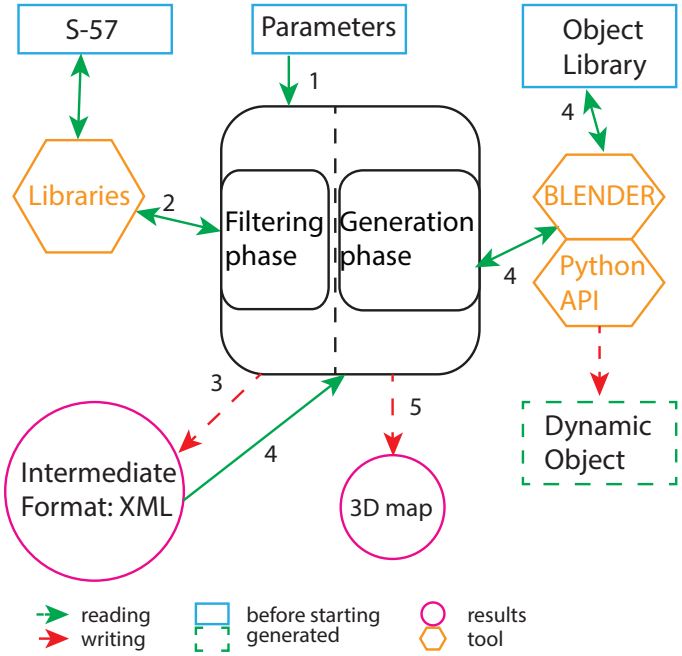


Figure 4.8: 3D chart generator schema divided in two phases

charts from the SHOM.

4.4.2 Pre-study memory size

Navigation aids Few tests have been realized to evaluate the memory size of a 3D file with the navigation aids. The latter are represented by arrows and topmarks (see Chapter 2). But in addition to these 3D objects, we decided to add the true representation of the 3D objects to identify the impact on the 3D chart memory size, in case wherein other complex objects will be added latter for users. This is also a solution to get more files and test in the application the management of the 3D objects and file paths, but in the final version of the application these objects will be removed.

The Fig.4.9 is a graphical representation of the objects used to control memory size, which is visible in Table 4.2. We chose the isolate danger arrow, topmark, pillar and spar buoys to make the tests because they take more memory than the other objects. So, it represents the worst case and we have used them to get a maximum memory size evaluation. Indeed, the isolate danger topmark is composed of two spheres (see Fig.4.9) and this is the most complex 3D object. To draw a sphere we need more vertices and faces than another one. The isolate danger topmark takes twenty times more memory than the lateral one for instance.

The isolate danger pillar buoy is the most voluminous 3D object compare

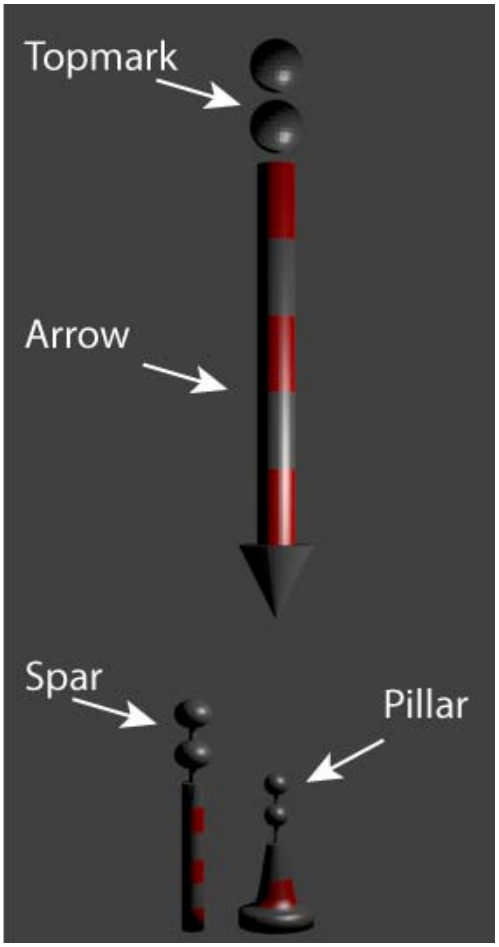


Figure 4.9: Isolate danger arrow, topmark, spar and pillar buoys

Table 4.2: Memory size of 3D objects

Object type	Memory size
Arrow ISD	14kBytes
Topmark ISD	46kBytes
BOYISD spar	80kBytes
BOYISD pillar	92kBytes
BOYISD pillar + topmark + arrow	159kBytes

to the other, its 3D file size is 92kBytes, this is also due to the two spheres placed on the top (see Fig.4.9). The set of 3D objects with the buoy, the arrow and the topmark takes about 160kBytes in a Wavefront obj file. If we consider the Lorient harbour S-57 chart, there are 133 buoys, so we can expect a maximum memory size less than 21.3MB.
The real memory size of the 3D navigational aids objects will be different

from the estimation. Firstly, in the tests, all the 3D objects were placed on the origin (coordinates 0,0) and secondly, we have evaluated the memory size only with isolate danger objects. In real, there are less isolate danger objects than a lateral one for instance so the memory size will be less than 21.3MB.

In conclusion, this is possible to generate a 3D charts with the 3D representation chosen, the mobile device is able to run an application with 21.3MB in the Random Access Memory. The process to manage the navigational aids objects is detailed in Section 4.4.3.1 but we present the depth areas memory size evaluation before.

Depth areas Few tests have been realized to evaluate the memory size of a 3D file with the depth areas. The next table is the results of tests, two shapes have been chosen, a square and a circle to identify which is the most voluminous with the two different sizes. Once the tests have been realized,

Table 4.3: Memory size of surfaces

Object type	Size	Memory size
plane	100m x 100m	0.251 kBytes
plane	1km x 1km	0.258 kBytes
circle	radius 100m	2.684 kBytes
circle	radius 1km	2.743 kBytes

the maximum size of a depth area 3D file has been evaluated with this table. Indeed, a circle with a radius of 1km takes 2.7kBytes as memory but this surface size is not the typical one, the most appropriate is the circle with a radius equals to 100m.

Well, there are 959 depth areas in the S-57 chart of Lorient harbour from the SHOM, so it could take 2.5MB for the whole. This is our estimation of the memory size for all depth areas between -15m and 5m in the Lorient harbour and this number seems to be light. The final memory size will be a little different because the estimation of the memory is based on objects placed with a position equals to (0,0), and the real depth area are not circle with 100m as radius. So, it could increase or decrease of few kilo bytes.

In conclusion, this study on the depth areas memory size is positive. It allows the use of coloured areas in addition to the navigational aids. The total memory size for both, is about 24MB and this is not to big for a mobile device with 1GB has Random Access Memory. The following section details how the 3D charts are generated.

4.4.3 3D chart Generation

The generating phase (steps 3, 4 and 5 in Fig.4.4) is possible thanks to a main software: Blender. The 3D software Blender is vital for this step, in the same way as the GDAL library in the filtering phase. Indeed, it is not possible to create or generate 3D objects/charts without any 3D software. We chose this software because:

- This is an open-source software.
- It allows the use of script.
- It is able to export 3D scene in many formats.

First, this is an open-source and free software with an active community, which is very important for designing 3D objects without low knowledge in 3D conception. Second, Blender is able to run scripts in Python, in order to automate some repetitive functions (load, place, colour, export). Finally, this 3D software can export a 3D scene in the same 3D file formats that are readable by the 3D engine Irrlicht used in the application. So, this software meets our needs, it is possible to generate quickly the 3D charts required in the 3D application MARNNA.

There are two main parts in the generating phase, the management of the spatial objects represented by a point (the navigational aids) and the others (the areas). These steps are represented in a schema close to a Grafcet, in Fig.4.10. The management of the two kinds of objects is very different, so we first explain how to build a chart with navigational aid objects (Section 4.4.3.1), then we explain in Section 4.4.3.2 how to build the 3D chart for the depth areas or equivalent, e.g. regatta tracks.

4.4.3.1 Navigational aid object

When the object read from the XML file (step 2 in Fig.4.10) is a spatial object represented by a point (buoy, beacon, landmark, etc.), the generator runs 3 main functions:

- 31: Read and decode attributes.
- 32: Find and load the corresponding 3D objects.
- 33: Place the 3D objects with the UTM coordinates.

In the first function (31 in Fig.4.10) the script decodes the attributes written in the XML file and build a path file. Then, the function 32 in Fig.4.10 is able to retrieve the corresponding 3D objects (topmark, arrow, etc.) and load it in a 3D scene. We have decided to organize the 3D library objects like in Fig.4.11. Every 3D objects in this folder tree have been designed on the 3D software Blender, where the generation script is run. Finally, the

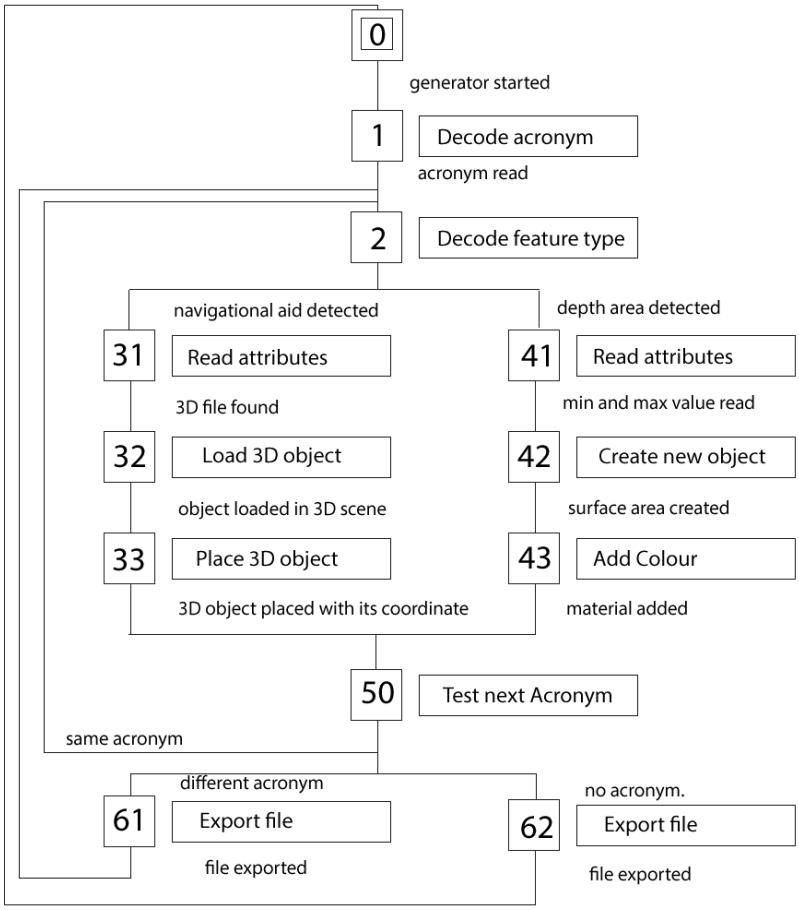


Figure 4.10: 3D chart generator schema divided in two phases

function 33 in Fig.4.10 read the object UTM coordinates in the XML file and moves it to the right place.
When the set of identical features are placed, the generator exports the 3D scene into a Wavefront OBJ format in an output tree, which is very close to the previous one and shown in Fig 4.12.

The next Table is an overview of the memory size for each kind of buoy/beacon from the chart tested in the generator (Lorient harbour). The navigational aids are represented by the three elements seen in the memory study: a topmark, an arrow and a 3D buoy. This result is less than the evaluations as it was expected, the total memory size is 5.42MB whereas the evaluation was 21.3MB. The result is four times less than the evaluation because the chart has only 2 isolate danger buoys. More tests will be presented in Section 4.4.5 accompanied with few commentaries.

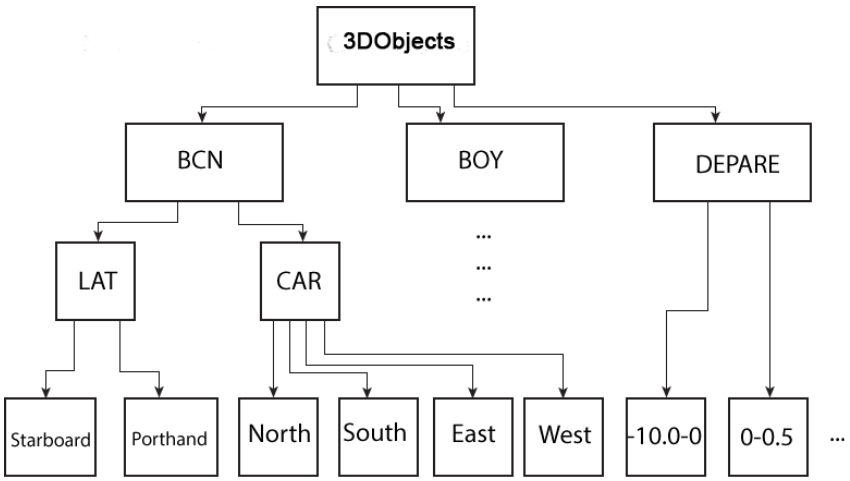


Figure 4.11: 3D objects library tree

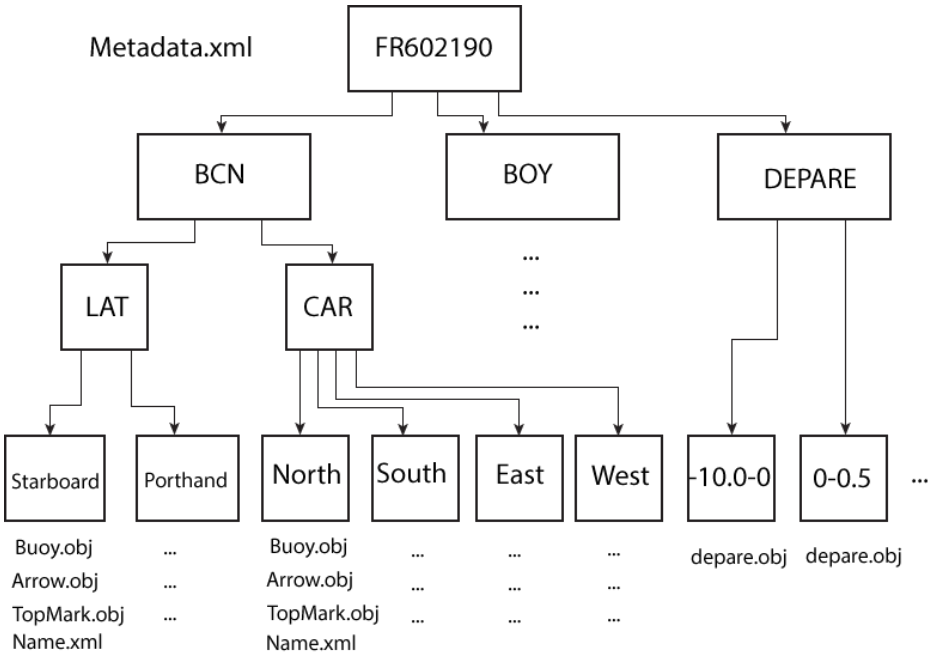


Figure 4.12: 3D generator charts output tree

4.4.3.2 Area Object

When the object read from the XML file (step 2 in Fig.4.10) is a spatial object represented by an area (depth areas, etc.), the generator runs 3 main functions:

Table 4.4: Memory size of navigational aids

Object type	Number	Memory size
boy/bcn lat	98	3.39MBytes
boy/bcn car	13	0.86MBytes
boy/bcn isd	2	0.38MBytes
boy/bcn spp	20	0.79MBytes
TOTAL	133	5.42MBytes

- 41: Read and decode attributes.
- 42: Create the new object.
- 43: Add the colour to the object.

In the first function (41 in Fig.4.10) the script reads and decodes the attributes written in the XML file. Then, in the function 42 in Fig.4.10, a vertex is placed on every point identified with (X,Y) coordinates of the area and they are connected with straight lines. Once all the points are linked, a new mesh is created and a material is added to get a colour on the object (function 43 in Fig.4.10). If all the same depth category are generated, the scene is exported as a 3D Wavefront obj file in the output folder tree (see Fig.4.12).

The left part in Fig.4.13 is a representation of an emerged area and the right side is a view of the generated one by the script in Blender. We colour

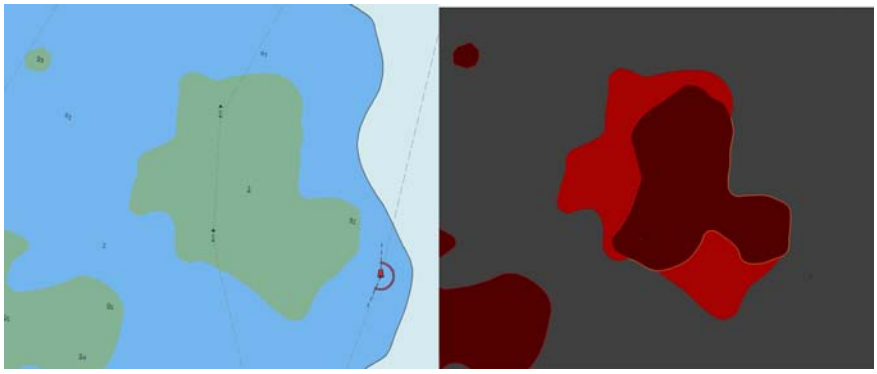


Figure 4.13: Navigation software and Blender views of a depth water area.

the area because it is more understandable for the user than a simple cylinder. Indeed, the user doesn't know if the boat is inside or outside of the dangerous area. Then, all the dynamic objects are generated and they are stored in the output tree depending on the depth category, every half meter. The next Table is an overview of the memory size for the depth areas from the chart tested in the generator (Lorient harbour). This result is less than

Table 4.5: Memory size of surfaces

Object type	Objects Number	Memory size
Depth Area	959	1.64MBytes

the evaluation, as it was expected. The memory size for 959 depth areas takes 1.64MB whereas the evaluation was 2.5MB. More tests will be presented in Section 4.4.5 accompanied with few commentaries.

4.4.4 Outputs

The generator output gives a tree composed of folders containing 3D files with the Wavefront OBJ filename extension and XML files. This tree structure is displayed in Fig.4.12. The first file generated is metadata.xml (Fig.4.12), which is a XML file storing the name and the contour (bounding box of the generated charts. It also gives to the application the possibility to load or not the chart corresponding to the navigation area of the boat. The latter can be obtained with the help of the GPS if the user owns the chart. The data stored in the application are updated and deals with the loading and unloading of the charts when the boat enters or leaves a zone. The folders are named according to the acronyms of the feature (from the S-57 standard) in order to be able to sort them out, to write the results and to read them in the MARNAA application with a common filepath. At the bottom of the tree, we can find OBJ files and a XML file (buoy.obj and name.xml for instance), like shown in Fig.4.12.

The drawback of the Wavefront OBJ file generated with the 3D chart generator is that it loses the position of the 3D objects if there are several objects in the same file. That's why we generate the "name.xml" file that stores object names and their positions. In addition to the two files, the generator adds Wavefront OBJ files, which contained the augmented objects for the navigation application: arrows and topmarks. It is implemented in other files to let flexibility to the user to add or remove those objects in MARNAA.

The whole of the system can be resumed with the schema in Fig.4.14 This schema represents the blocks needed to generate the 3D nautical charts for MARNAA application. More results are visible in Section 4.4.5, where 3 charts from the SHOM are tested and the memory size of the generated 3D charts detailed.

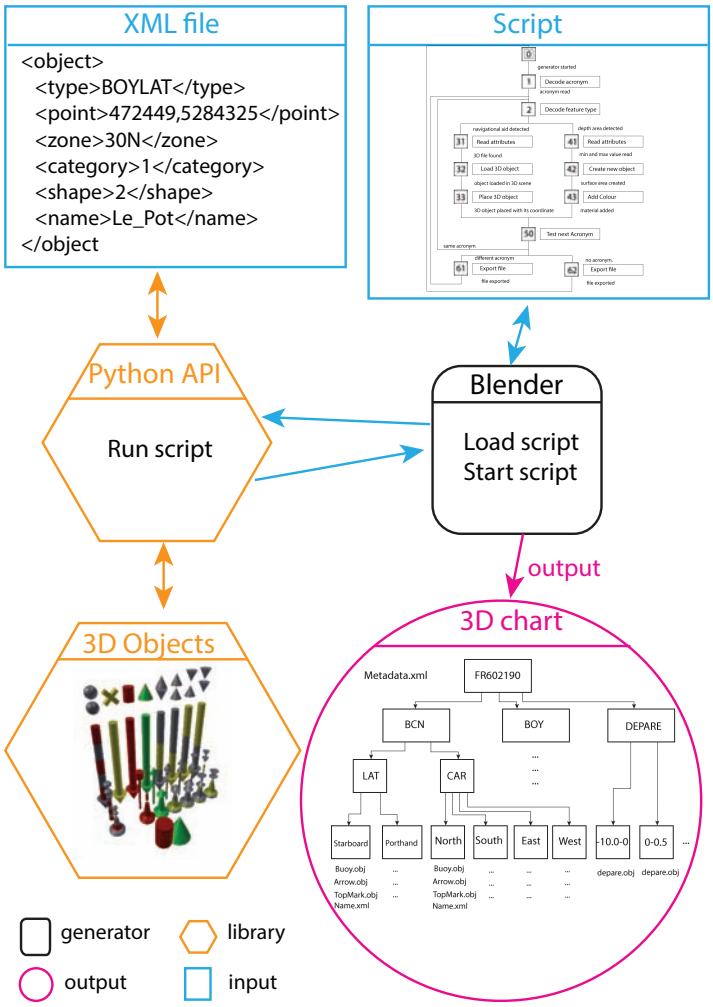


Figure 4.14: Schema which represents the second script of the generator: generating phase

4.4.5 Results

The generator in few figures In Section 4.3.2.3, we presented the Fig.4.7, which represents the filtering phase. This phase needs filters, input files, scripts and the generator can't work without the libraries. These libraries, GDAL and the geographic converter library take few tens MB as memory size and the scripts only few kB (for our own code). The time to filter and generate the XML file is close to 1 minute for the three cases presented next.

In Section 4.4.3, we presented the Fig.4.14, which represents the generating phase. This phase is based on a 3D software (Blender) and the python API to run few Python files. The latter, apart from the software and APIs only take few tens kB as memory size. But the time to generate the 3D files

for depth areas and the navigational aids, is longer than the filtering phase; the script needs few minutes to get the final 3D files tree. This time was evaluate for the 3 case studies presented just below.

Case studies The SHOM charts are divided into 6 categories, which depend on the scale, as it was seen in Section 4.3.2. We decided to test the generator on 3 different charts in order to determine the memory footprint of the generated 3D charts. The first chart is a category 2, (FR200010), which covers the bay of Biscay and its bounding box is visible in Fig.4.15. The next

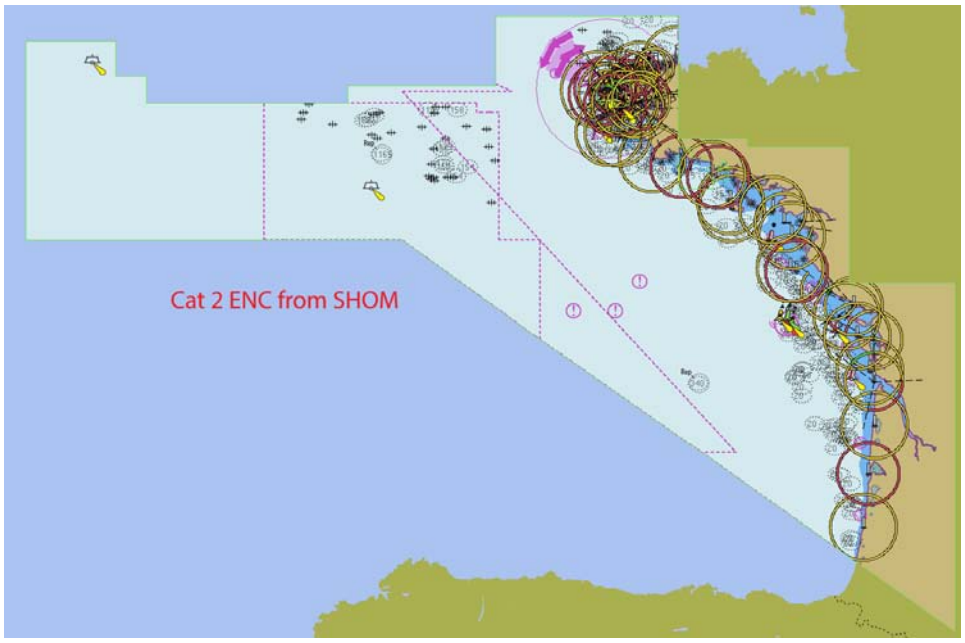


Figure 4.15: Navigation software view of the S-57 category 2 ENC from the SHOM

chart is a category 4, (FR402320), which covers an area between Glénan and Groix in south Brittany and its bounding box is visible in Fig.4.16. The last chart is the Lorient harbour area, this is a category 6 chart (FR602190) and its bounding box is visible in Fig.4.17. In this study, we have taken into account data extracted from the user's study in Chapter 2. The set of navigational aids are the following:

- Beacons / buoys lateral.
- Beacons / buoys cardinal.
- Beacons / buoys isolate danger.
- Beacons / buoys special mark.

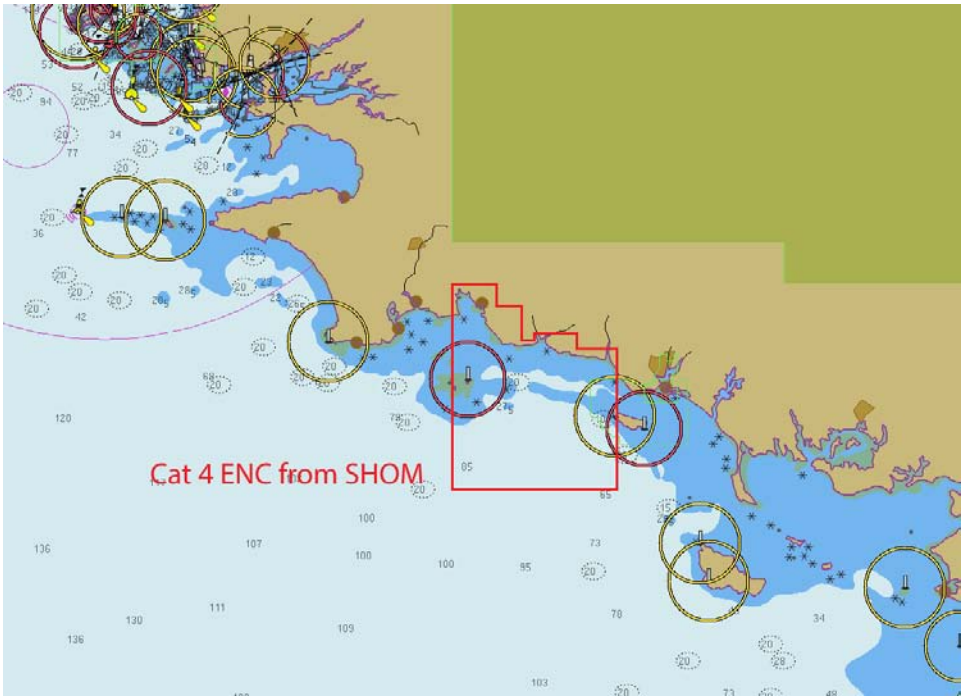


Figure 4.16: Navigation software view of the S-57 category 4 ENC from the SHOM

The second kind of information are the set of the depth areas between negative values (emerged area) until 5 meters below the see side (submerged area). So, we present both, the generated 3D chart and S-57 memory size in Table 4.6. Firstly, the chart's category has an influence on the memory size.

Table 4.6: Memory size of 3D charts generated

Chart area (category)	Object type	Number	Memory size
Lorient harbour (6)	depth Area	959	1.64MBytes
	boy/bcn	133	5.42MBytes
	S-57 file	full chart	1.92MBytes
Glénan to Groix (4)	depth Area	465	0.27MBytes
	boy/bcn	69	3.57MBytes
	S-57 file	full chart	1.01MBytes
Bay of Biscay (2)	depth area	325	0.25MBytes
	boy/bcn	43	2.46MBytes
	S-57 file	full chart	0.84MBytes

There are more elements in the category 6 chart (133 navigation aids) than the category 2 (43 navigation aids), so the memory size is higher even if the geographic area is smaller.
Secondly, there is a notable difference between the size of the navigation

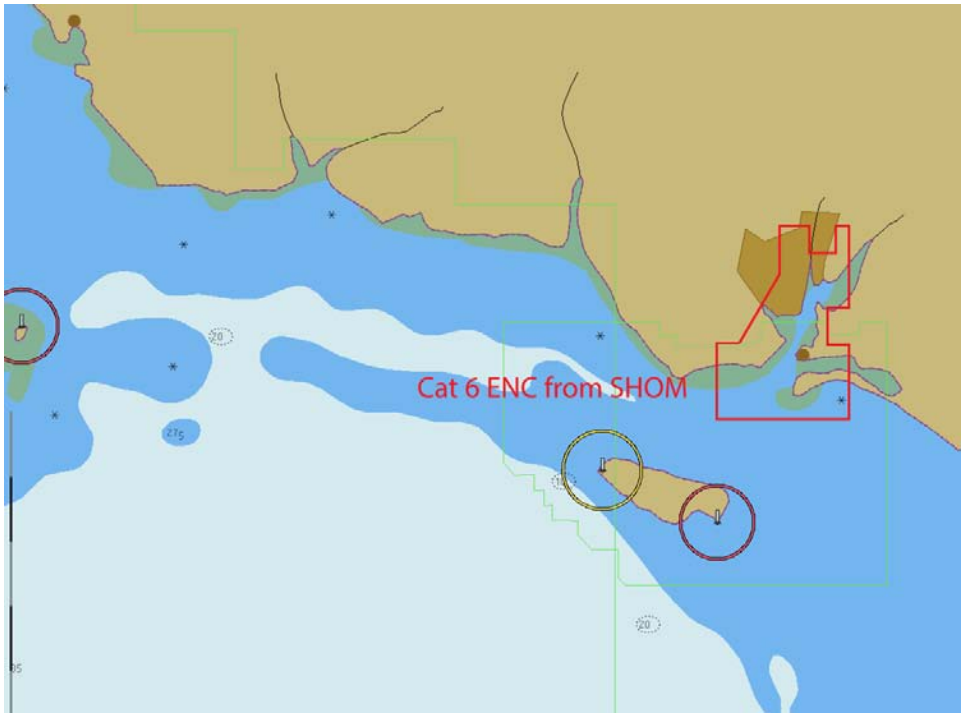


Figure 4.17: Navigation software view of the S-57 category 6 ENC from the SHOM

aids and the depth area in the generated chart, 959 depth areas take only 1.64MB whereas 133 navigational aids take 1.92MB. Even if a depth area is a complex shape it's still a 3D shape whereas the navigational aids objects are represented by 3D one. Indeed, the number of vertices and faces required to display an 3D arrow is bigger than a 2D area, this is why the memory size increase more with the number of the navigational aids than the depth areas. It means that we can be confident with the possibility to add other kinds of area information corresponding to specific user needs (diving, regatta, etc.). Lastly, the 3D chart has a memory size bigger than a classical S-57 file even if the number of objects is less. Indeed, the description of a 3D object in a Wavefront OBJ file is more complex and needs more text than the description of a feature with attributes in the S-57 standard.

In conclusion, the memory size of the 3D charts generated is acceptable for an application running on a mobile device (Smartphone or AR glasses) with a Random Access Memory near 1 GB in the three cases tested. The 3D charts generated are less than 10MBytes, furthermore, these results provide flexibility to add more information in the 3D charts depending on the user needs such as the prohibited diving area, wrecks for instance and so on.

4.5 Conclusion

The 3D chart generator has been designed under two main constraints:

- Satisfy both, the user and the mobile application needs.
- Provide flexibility.

First, this generator is able to retrieve useful data listed in the user's study in Chapter2 and transform the navigational aid and danger area in a 3D format usable in MARNAA application. Moreover, the memory size of the 3D charts is limited, less than 10MB for a chart with all depth water area until 5 meters and the most useful buoys and beacons to get a safe navigation. The automation of the filtering and generating phases provides an essential element, indeed only few minutes are required to obtain the depth area and navigational aids. A 3D view of the generated objects in a 3D software is visible in Fig.4.18, where the navigational aids are represented by augmented objects and depth area by coloured area. Secondly, the generator provides a

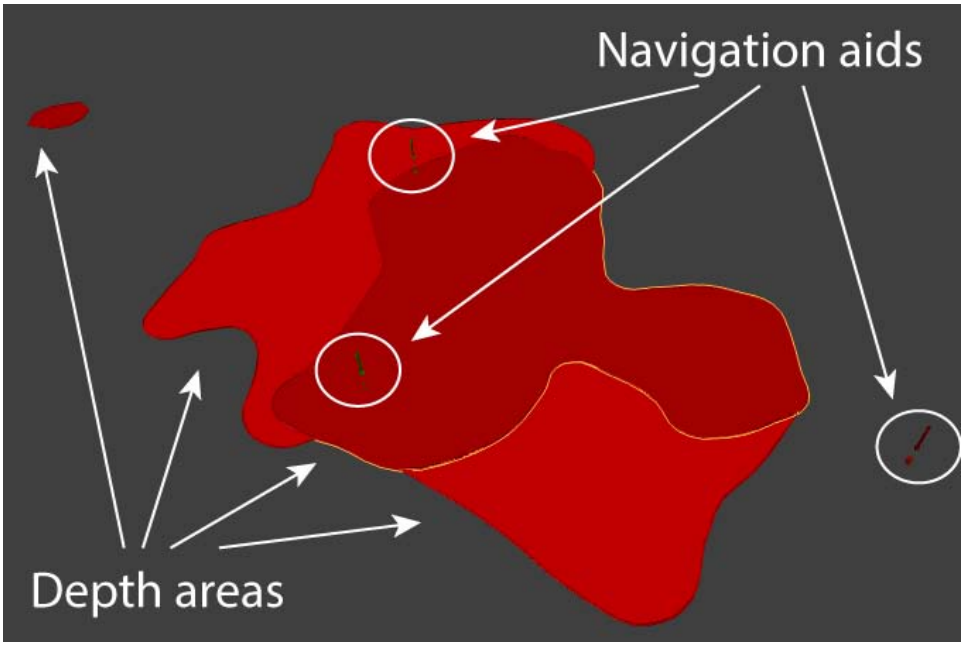


Figure 4.18: 3D view from Blender with depth area and 3D navigation aids

great flexibility due to XML files in input and output. Indeed, many input file formats can be read, even if in this work we use only the S-57 format since it is the standard format of the hydrographic center. In addition to this format, open format such as the GPX or vectorial formats are also usable thanks to the GDAL library. This flexibility allows us to take into account the user information needs in their nautical activities (regatta, fishing or diving for

instance). All these data are sorted and organized in the generator's output. As, 3D files are stored in a specific tree the application can load and unload easily the data in this way.

Chapter Fifth

General conclusion

5.1 General context of the thesis

The maritime navigation exists for few thousand years and it is constantly developing, both the boats and the navigational aid devices. Today, there is a wide variety of boats on the seas and oceans, from small recreational boats to big merchant navy boats. So, the maritime devices are primarily tools to secure navigation of boats. There are plenty kinds of devices from the compass to the ECDIS, but even if there are more and more electronic devices onboard there are still accidents [Law, 2012],[Guard, 2012].

First, the improvement of the technology onboard led to the complexity of devices [Jie and Xian-Zhong, 2008]. Second, they are often regrouped in a specific place on a boat and the displays are fixed; it makes the information not mobile. Then, the studies from Porathe [Porathe, 2006] have highlighted that human errors are due to the cognitive load, induced by the multiple orientation of the electronic devices onboard, which are different from the user's field of view. Finally, all the navigational aids are not available on recreational boats and the useful data are not accessible, only paper or electronic charts readable on chartplotter.

The electronic devices are not the only information available on the sea, today a lot of countries have unified their buoyage system to the IALA one. The latter has been put in place to help the users during the coastal navigation. This norm specifies the colours and buoy's shapes, moreover in night condition these buoys emit lights. So, the sailor must know these rules to sail safely on the sea, but it isn't always the case for recreational boats.

Thus in summary, there are today three main issues on the maritime navigation aid:

- Too much information are available and sometimes not suitable to the navigation context.
- The information is not mobile.
- Displayed symbols are not placed in the user's field of view.

Furthermore the weather complicates the navigation on the sea. Indeed, the visibility can vary (night, fog, sunbeam), as the sea conditions with the

boat movements due to the swell. This environment is very constraint for both, the mobile devices and applications. So, the solution has to take into account the maritime context, the weather conditions and answer the three issues for all kinds of boats.

5.2 Summary of Contributions

The first contribution of this thesis is the approach used to reduce the cognitive load issue. We decided to work with end users to sort the relevant information by phases and scenarios through questionnaires. This step is described in Chapter 2, like the graphical interface study. This study has been realized to find the best object's shapes and colours that will be display to the user. The first result from these two studies, information to keep and their graphical representation, is a list of information to display: state data, alert data, and navigational aid data and their graphical representation (text, 2D and 3D). Then, the second result is a list of profiles depending on two scenarios: good and bad visibility and two phases: leaving/arriving in an harbour and coastal navigation. These results are essential to help the sailor with a list of restricted data easy to understand, which provides a low cognitive load. The reduction of the cognitive load is also possible if the data are displayed in the user's field of view.

The maritime context imposes ergonomic constraints, the sailor must have the two hands free to steer the boat, trim the sails, and so on, furthermore data have to be visible both, when it's dark or on a very bright day. So, we have deducted the list of the hardware and software requirements in Chapter 2 and bring the solution: a specific Optical See-Through device to solve the mobility and orientation problematic. These two points are detailed just below.

The second contribution in this thesis is a set of a hardware and a software prototypes to provide two main functions:

- Display the maritime data extracted from the user's study.
- Make the data mobile.

In chapter 3, we detail first the design of our prototype to run an augmented reality application in the maritime domain. The architecture has been selected to run outdoor applications apart from the weather conditions (luminosity). The result is a prototype with a GPS and an IMU in order to geolocate and compute the user's head orientation. A GPGPU architecture and wireless connections are also embedded to run all the software tasks required to help the user.

The second result is the application MARNAA, which is able to acquire data from the boat network and provides services to display the most useful

information depending on the devices available and navigation context. The state, alert and navigational aid data are displayed in the user's field of view to help him/her in its navigation to avoid groundings.

The complete solution is detailed in Chapter 3 and have lead to publications. First, the regatta service has been tested on the Augmented Reality display and the Snowball board. The communication between the system and a navigation processor was established to get and display performance data and the results have led to a publication in an international conference: International Conference On Innovation In High Performance Sailing Yachts [Douguet et al., 2013]. Second, the application and the hardware prototype designed as a pervasive system that provide services to the user has led to a publication in the IEEE international conference on Embedded and Ubiquitous Computing [Morgere et al., 2014a].

The last contribution in this thesis is the 3D chart generator, detailed in chapter 4. The generator has been implemented to transform data from the textual chart format (S-57, GPX, etc.) to the 3D graphical format of the application. A 3D engine is at the heart of the application to render frames of a virtual world displayed on the augmented reality device. The method used to convert text to a 3D virtual world is similar to the Model-driven engineering. The generator takes the relevant data from the user's study to extract and export into the 3D format workable to the application. This tool has lead to a publication, it has been presented in the MTS/IEEE international conference OCEANS'14 St. John's [Morgere et al., 2014b] with a travel Grant obtained as a result of the selective student poster competition.

5.3 Perspectives

There are three main themes in the thesis, the user interface, the application and the hardware. So, the first part of the perspective is the list of tasks to reach on the user interface to satisfy the three application uses: recreational boats navigation, regatta and professional navigation. Then, the most important improvements that have to be implemented on both, the application and the generator are suggested. Finally, we think that the mobile augmented reality should take two main paths to take off: the professional domain with maintenance uses for instance and dedicated uses such as hobbies and sports. Our vision is detailed in the last part of the perspectives.

5.3.1 Ergonomic

There are three kinds of uses for the MARNAA application, the recreational boats, the regatta and professionals.

Recreational boats People are not systematically aware of weather conditions whereas the wind or the current can cause accidents. For the reasons we would add these two main data in the application in order to improve the security during navigation. The first action is a further user's study to find how to display these data in addition to the depth areas and navigational aids. These data can be extracted from gribs. Second, these data must be dynamic on the display because they can vary a lot during a navigation. Indeed, the direction and the wind speed can be very different between the leaving an the arriving in the harbour. We have to define the best graphical representation with the help of user feedbacks. As an example, a 3D arrow could represent the direction with an animation and the colour could changed depending on the speed. This approach can also be used to represent the current because it varies with the tide and the speed. In the Golf du Morbihan for instance the current is extremely variable in space, speed and direction, it can vary from -9 to 9knots at the entrance.

Regatta Both, user and ergonomic studies have to be put in place to get the most relevant data and the best representation in the regatta use. There is an example of a Augmented Reality scene during the America's cup picture in chapter 4 but this is a top view from the race. This scene seems to be not adapted to the user's field of view, small coloured area are easier to see from the top view. The mixing of user's feedback and ergonomic specification will conduct to a 3D scene in accordance with user expectations and environment constraints. The objective is the use of objects easy to understand in order to limit the cognitive load and help the sailor to improve performance.

Professional A specific study with professionals such as merchant navy, fishermen has to be realized to provide the best mobile augmented reality application. The expectations seem to be different from the recreational boat owners and there are more specific data and functions to implement depending on the boat and the kind of sailor.

5.3.2 Applications

MARNAA First, there are some improvements in the short-term. The pose estimation with the Extended Kalman Filter may be control in a depth study in order to see if we can improve the EKF model according to the specific boat moves. In the same way, the position from the GPS could be also improved. Sometimes the GPS has not a good reception and data can be lost, so the solution is to implement a loosely coupled GPS/INS system as an extension of our EKF filter, that can estimate the localization during short GPS outages. Another advantage of this solution is the rendering of the scene with both, the orientation and position with a 30Hz sampling rate. Indeed, currently the position is updated between 1 and 5Hz, so there

is a difference between the user and virtual camera position. Another phenomenon is present on the display, the objects can make a jump because of the GPS data rate, which is too slow. So a solution with the Kalman filter could improve the position rate and avoid object jump.

In the medium-term, as it was previously seen in the ergonomic section, we would add the wind and the current data on the display. It induces the use and the management of animated objects and colour's changing in real time. In 3D, the animations are realized on 3D software and modifications on the colours when the application run is applied with shader programs that may have to be create in this case.
The other modification on the application is related to the regatta interface. We would like to display, in real time, performance data and boat positions or tracks. Performance data are easy to display in a text form but we would have to use OpenGL ES functions to draw points or lines, this is more complicated than the loading of a 3D object with the 3D engine. Again we must include user in the loop since too complicated and dynamic objects may be rejected.

In the long-term, the management of 3D charts between the application and the generator has to be realized. We want that users can connect the mobile device to internet with the WiFi in order to download charts when they are at home or on the harbour. One of the solutions is a web server accessible by the application to select the chart and some data to keep. Then the generator could produce or send the 3D chart on the web server. The last step is the downloading of the chart in the mobile device to let the application load it. The application will also have to manage the charts stored on the device and load the most precise chart (best scale) if two charts overlap.

3D chart generator In the long term, we would like an interface between the generator and a web server to exchange charts. Like S-57 files, waypoints and tracks on GPX files could be used on the web server to get the regatta track for instance.

On motor boats (recreational boats), sailors often used a sounder, more or less equipped (depth, seabed shape, etc.). One of the most interesting information from this device is the shape of the seabed to anchor, fish, etc. So we want to provide the sailor with a better information than the prohibited area (depth areas) already displayed. We would like to add the relief on the 3D chart to display the drop off or rocks below the water surface. The S-57 file for instance, owns enough information to realize this improvement. Few colours and a light in the 3D engine are usable in order to provide a better perspective (with shadows). Some tests on the memory size will be necessary because it could be increased and become too numerous. Moreover the number of vertices are limited with the Open 3D engine Irrlicht, if the set of objects exceed 65534 vertices, the 3D engine doesn't work. Another key point on the relief is the ergonomic context, this is not recommended to

colour large area on a display. So if the solution is accepted by users, only the rocks and the drop off in shallow waters will be display to reduce the number of objects and the coloured areas.

5.3.3 Embedded system

The main stream AR glasses systems are taking too long to really get off the ground. Currently the uses of this technology are not obvious but, for us, this technology could be first accepted in specific domains with dedicated designs. We have in mind two promising domains: outdoor activities and maintenance. We think that the hardware should take two main paths: hobbies or sports for dedicated applications and the system maintenance.

Hobbies or sports The architecture currently used in main stream devices is a GPGPU with IMU, GPS and wireless connections. It is enough to provide useful applications but the packaging is not ready to be used in sport for instance. The actual Optical See-Through glasses [1] [2] [3] should be designed in the same way of the Recon Jet[4], Oakley ski mask[5] for instance.

There are two hypothesis that prevent AR glasses manufacturer to develop this kind of devices. First, the size of the optical system and the battery to power both, the display and the hardware lead to a bigger footprint than heads-up display without AR technology. Second, the OST glasses manufacturer want their AR glasses to become a wearable device like the smart watch to access the main stream market; specific embedded system with low footprint and low power techniques at communications and computations levels.

Maintenance The maintenance doesn't have the same needs as the applications for hobbies or sports. There are two ways: the local or distant maintenance. We consider the local maintenance when the technician is autonomous with an AR device.

The latter is able to show on a display the procedures to perform. In this case, the pose estimation is often compute with the help of two cameras. Image processing requires strong computing capacities. Currently the architecture used in the most recent tablets are able to process image processing and display 3D objects with the GPU. So in the actual AR glasses it is not possible to run this kind of applications and furthermore, the battery can stand few tens minutes.

[1] http://optinvent.com/see-through-glasses-ORA
[2] http://www.laster.fr/products/seethru/
[3] http://www.epson.com/cgi-bin/Store/jsp/Landing/moverio-bt-200-smart-glasses.do
[4] http://www.reconinstruments.com/products/jet/
[5] http://www.oakley.com/en/airwave/product/W0OO7049

We consider the maintenance as distant when a technician communicates with another person by audio and video stream in real time. The mobile device must send video and audio streams through wireless connection with enough resolution to be visible and audible on an another distant device. So the first constraint is the encoding of the streams. Then the mobile device has to send data to other device, in real time, and without lag (less than a second). Finally, the other person can send data or talk to the technician to help him. The first lock is that current architectures used in AR glasses are not enough powerful to execute this task. The second problem is the battery life, send video stream by the WiFi hugely increases the system consumption. Trade-off would have to be find between mobility and power supply to provide a wearable device.

In both cases, the power consumption analysis is required because it is currently not possible to realize local and distant maintenance with a fully embedded system (hardware and display). Studies on uses may be realized in order to know if people will accept drawback if mobile AR devices give added value. Indeed, the technician could have both, the architecture and the battery placed in his/her pocket for instance to get both, the mobility and enough power (battery and computing). Some manufacturers such as Mira or Epson deport the architecture and battery with a wire in a very small box but we think that it is not the only solution.

5.3.4 Mobile Augmented Reality challenges

There are still challenges for augmented reality glasses to be accepted by main stream market:

- Field of view must be about 85°.
- In specific cases the eye discomfort due to the glasses.
- Matching of augmented object and real view.
- Miniaturization of the glasses.
- Robustness for outdoor uses.
- Battery life.

We cannot really contribute to the 5th first point that mainly rely on AR devices technology but power management techniques can be apply to improve the last but important point. Recent progresses in AR devices such as the large NVIDIA solution to offer a 100° FOV indicate that we can be optimistic for AR devices. Be believe that this is also the case for power optimisation.

We can investigate the design of a dedicated ultra-low power architecture that approach the limits of the best power efficiency in terms of computations (e.g. 5pJ per MAC32) and communications (e.g. 100pJ/bit).

The first way is the ASIC architecture. In the latter, few very low power technologies such as the FD-SOI, non-volatile memories with power gating for instance may be integrated to reduce power consumption.
The second way is the used of dynamic power management that can be combined with accurate knowledge of the application. So, memory accesses, computations and communications can be activated only when necessary. For instance, we would like to have access to the video controller and the LED memories so that only pixels with information and update generate memory transfers. Other parameters like the sampling and the video rates can be dynamically adapted according to user moves and information to display.

In conclusion, the battery life challenge is a work for a thesis aim to study the hardware architecture and management of power consumption through management tasks.

Bibliography

[Agarwal et al., 2014] Agarwal, P., Burgard, W., and Stachniss, C. (2014). Survey of geodetic mapping methods: Geodetic approaches to mapping and the relationship to graph-based slam. *Robotics Automation Magazine, IEEE*, 21(3):63–80.

[Agency, 1989] Agency, D. (1989). The universal grids: Universal transverse mercator (UTM) and universal polar stereographic (UPS). Technical Report TM8358.2, Defense Mapping Agency, Hydrographic/Topographic Center, Fairfax, VA, USA.

[Bijker and Steyn, 2008] Bijker, J. and Steyn, W. (2008). Kalman filter configurations for a low-cost loosely integrated inertial navigation system on an airship. *Control Engineering Practice*, 16(12):1509 – 1518.

[Brigante et al., 2011] Brigante, C., Abbate, N., Basile, A., Faulisi, A., and Sessa, S. (2011). Towards miniaturization of a mems-based wearable motion capture system. *Industrial Electronics, IEEE Transactions on*, 58(8):3234–3241.

[Bruce and Foster, 1982] Bruce, M. and Foster, J. J. (1982). The visibility of colored characters on colored backgrounds in viewdata displays. *Visible Language*, 16(4):382–390.

[Comport et al., 2006] Comport, A. I., Marchand, E., Pressigout, M., and Chaumette, F. (2006). Real-time markerless tracking for augmented reality: The virtual visual servoing framework. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):615–628.

[Diguet et al., 2013] Diguet, J.-P., Bergmann, N., and Morgère, J.-C. (2013). Embedded system architecture for mobile augmented reality (sailor case study). In *PECCS13*.

[Douguet et al., 2013] Douguet, R., Morgère, J.-C., Diguet, J.-P., and Laurent, J. (2013). Coupled open navigation and augmented reality systems for skippers. In *International conference on innovation in high performance sailing yachts (Innov'Sail)*, Lorient, France.

[Feiner et al., 1997] Feiner, S., MacIntyre, B., Hollerer, T., and Webster, A. (1997). A touring machine: prototyping 3d mobile augmented reality systems for exploring the urban environment. In *Wearable Computers, 1997. Digest of Papers., First International Symposium on*, pages 74–81.

[Frosio et al., 2009] Frosio, I., Pedersini, F., and Alberto Borghese, N. (2009). Autocalibration of MEMS accelerometers. *Instrumentation and Measurement, IEEE Transactions on*, 58(6):2034–2041.

[Genc et al., 2002] Genc, Y., Tuceryan, M., and Navab, N. (2002). Practical solutions for calibration of optical see-through devices. In *Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on*, pages 169–175.

[Gilmore, 2012] Gilmore, W. (2012). *The user- computer interface in process control: A human factors engineering handbook*. Elsevier.

[Grisetti et al., 2005] Grisetti, G., Stachniss, C., and Burgard, W. (2005). Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2432–2437.

[Grisetti et al., 2007] Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1):34–46.

[Grubert, 2014] Grubert, J. (2014). Google glass, the meta and co. how to calibrate optical see-through head mounted displays. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 1–2.

[Guard, 2012] Guard, U. C. (2012). Department of homeland security (US). 2012 recreational boating statistics.

[GUMSTIX, s355] GUMSTIX (https://store.gumstix.com/index.php/products/355/).

[Guo et al., 2008] Guo, P.-F., Qiu, H., Yang, Y., and Ren, Z. (2008). The soft iron and hard iron calibration method using extended kalman filter for attitude and heading reference system. In *Position Location and Navigation Symposium (PLANS)*.

[Helander et al., 1997] Helander, M. G., Landauer, T. K., and Prabhu, P. V. (1997). *Handbook of Human-Computer Interaction*. Elsevier.

[Herout et al., 2012] Herout, A., Zacharias, M., Dubska, M., and Havel, J. (2012). Fractal marker fields: No more scale limitations for fiduciary markers. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 285–286.

[Hoag, 1963] Hoag, D. (1963). Apollo guidance and navigation: Considerations of apollo imu gimbal lock. *Canbridge: MIT Instrumentation Laboratory*, pages 1–64.

[Hugues et al., 2010] Hugues, O., Cieutat, J.-M., and Guitton, P. (2010). An experimental augmented reality platform for assisted maritime navigation. In *Proceedings of the 1st Augmented Human International Conference*, AH '10, pages 12:1–12:6, New York, NY, USA. ACM.

[J-Ph. Diguet and Morgére, 2015] J-Ph. Diguet, N. B. and Morgére, J.-C. (to appear, 2015). Dedicated object-processor for mobile augmented reality, sailor assistance case study. *EURASIP Journal on EMbedded Systems*.

[Jie and Xian-Zhong, 2008] Jie, W. and Xian-Zhong, H. (2008). The error chain in using electronic chart display and information systems. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, pages 1895–1899.

[Karlekar et al., 2010] Karlekar, J., Zhou, S., Lu, W., Loh, Z. C., Nakayama, Y., and Hii, D. (2010). Positioning, tracking and mapping for outdoor augmentation. In *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, pages 175–184.

[Kurze and Roselius, 2010] Kurze, M. and Roselius, A. (2010). Smart glasses: An open environment for ar apps. In *Mixed and Augmented Reality (ISMAR), 9th IEEE International Symposium on*, page 313.

[Kwik, 1989] Kwik, K. (1989). Calculation of ship collision avoidance manoeuvres: A simplified approach. *Ocean Engineering*, 16(5/6):475 – 491.

[Law, 2012] Law, J. (2012). Marine casualty analysis: We keep history from repeating itself. *Coast Guard Journal of Safety & Security at Sea, Proceedings of the Marine Safety & Security Council*, 69(3).

[Lawitzki, 2012] Lawitzki, P. (2012). Application of dynamic binaural signals in acoustic games. Master's thesis, Stuttgart Media University.

[LUMUS, 2011] LUMUS (2011).

[Luria et al., 1986] Luria, S. M., Neri, D. F., and Jacobsen, A. R. (1986). The effects of set size on color matching using crt displays. *Hum. Factors*, 28(1):49–61.

[LYYN, ncom] LYYN (www.lyyn.com).

[Massel, 1996] Massel, S. (1996). *Ocean Surface Waves: their Physics and Prediction*. Advanced Series on Ocean Engineering, World Scientific, Singapore - New Jersey - London - Hong Kong.

[McInerney and Kempeneers, 2015] McInerney, D. and Kempeneers, P. (2015). Introduction to gdal utilities. In *Open Source Geospatial Tools*, Earth Systems Data and Models, pages 61–62. Springer International Publishing.

[Morgere et al., 2014a] Morgere, J., Diguet, J., and Laurent, J. (2014a). Mobile augmented reality system for marine navigation assistance. In *Embedded and Ubiquitous Computing (EUC), 2014 12th IEEE International Conference on*, pages 287–292.

[Morgere et al., 2014b] Morgere, J.-C., Diguet, J.-P., and Laurent, J. (2014b). Electronic navigational chart generator for a marine mobile augmented reality system. In *Oceans - St. John's, 2014*, pages 1–9.

[Najafi et al., 2003] Najafi, B., Aminian, K., Paraschiv-Ionescu, A., Loew, F., Bula, C., and Robert, P. (2003). Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly. *Biomedical Engineering, IEEE Transactions on*, 50(6):711–723.

[Nielsen, 1994] Nielsen, J. (1994). *Usability Engineering*. Elsevier.

[Optinvent, 2014] Optinvent (2014). Ora.

[Perera and Soares, 2012] Perera, L. P. and Soares, C. G. (2012). *Detections of potential collision situations by relative motions of vessels under parameter uncertainties*. Taylor & Francis Group, London, UK.

[Pielou, 1978] Pielou, F. A. (1978). The introduction of iala maritime buoyage system a. *Journal of Navigation*, 31:422–425.

[Porathe, 2006] Porathe, T. (2006). *3-D Nautical Charts and Safe Navigation*. Mälardalen University Press dissertations. Mälardalen University. http://books.google.fr/books?id=NqCcNQAACAAJ.

[Prison and Porathe, 2007] Prison, J. and Porathe, T. (2007). Navigation with 2-d and 3-d maps: A comparative study with maritime personnel. *Proceedings of the 39th Nordic Ergonomics Society Conference*, pages 1–3.

[Ravden, 1989] Ravden, S. J. (1989). *Evaluating Usability of Human-computer Interfaces: A Practical Method*. Ellis Horwood Limited.

[Sandwell, 2002] Sandwell, D. T. (2002). *Reference Earth Model - WGS84*. Scripps Institution of Oceanography.

[Sears and Jacko, 2002] Sears, A. and Jacko, J. A. (2002). *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, Third Editiion*. CRC Press.

[Sebastian O.H. Madgwick, 2011] Sebastian O.H. Madgwick, Andrew J.L. Harrison, R. V. (2011). Estimation of IMU and MARG orientation using a gradient descent algorithm. *IEEE International Conference on Rehabilitation Robotics*.

[Semiconductors, 2000] Semiconductors, P. (2000). The i2c-bus specification. *Philips Semiconductors*, 9397(750):00954.

[Skoglund, 2011] Skoglund, M. A. (2011). Visual inertial navigation and calibration.

[Technologies, 2013] Technologies, L. (2013). Mg1.

[Tuceryan and Navab, 2000] Tuceryan, M. and Navab, N. (2000). Single point active alignment method (spaam) for optical see-through hmd calibration for ar. In *Augmented Reality, 2000. (ISAR 2000). Proceedings. IEEE and ACM International Symposium on*, pages 149–158.

[VUZIX, 2011] VUZIX (2011). Star 1200.

[Wang et al., 2013] Wang, Y., Zhang, J., Chen, X., Chu, X., and Yan, X. (2013). A spatial-temporal forensic analysis for inland-water ship collisions using {AIS} data. *Safety Science*, 57(0):187 – 202.

[Wientapper et al., 2014] Wientapper, F., Engelke, T., Keil, J., Wuest, H., and Mensik, J. (2014). [demo] user friedly calibration and tracking for optical stereo see-through augmented reality. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 385–386.

[Yang and Cheng, 2012] Yang, X. and Cheng, K.-T. T. (2012). Ldb: An ultra-fast feature for scalable augmened reality on mobile devices. In *International Symposium on Mixed and Augmented Reality (ISMAR)*.

Figures List

Tables List