SEVENTH FRAMEWORK PROGRAMME

FP7-ICT-2011-1.5 Networked Media and Search Systems
b) End-to-end Immersive and Interactive Media Technologies

Specific Targeted Research Project

VENTURI

(FP7-288238)

immersiVe ENhancemenT of User-woRld Interactions

[D4.2 Report on expected platform requirements of WP4 algorithms]

Due date of deliverable: [30-06-2012]
Actual submission date: [11-07-2012]

Start date of project: 01-10-2011
Duration: 36 months

Summary of the document

Document Code:	D4.2 Report on expected platform requirements of WP algorithms
Last modification: State:	11/07/2012 Ready to submit

Participant Partner(s):
 	
Editor & Authors (alphabetically):
 	
Fragment:
	FBK, Fraunhofer, ST-Italy, metaio, STE, Sony, INRIA
	
	Editor: Michele Zanin (FBK)
	
	Authors: Michele Zanin (FBK), Paul Chippendale (FBK), Benjamin Prestele (Fraunhofer), Valeria Tomaselli (ST-Italy), Paolo Pasteris (ST-Italy), Jacques Lemordant (INRIA)
	
	No

Audience:	☐ public ☐ restricted ☒ internal
Abstract:	This document presents an updated (with respect to D2.1.1 and D2.2.1) list of VeDi platform requirements, taking into account the current and expected evolution of WP4 algorithmic modules.
Keywords:	Requirement Analysis, Algorithms, VENTURI Platform, mobile
References:	Refer to the corresponding section at the end of the deliverable

Document Control Page

Version number	V1.3
Date	11/07/2012
Modified by	Paul Chippendale
Comments	Final check
Status	☐ draft ☒ WP leader accepted ☒ Technical coordinator accepted ☒ Project coordinator accepted
Action requested	☐ to be revised by partners involved in the preparation of the deliverable ☐ for approval of the WP leader ☐ for approval of the technical coordinator ☐ for approval of the project coordinator Deadline for action: 11/07/2012

Change history

Version number	***Date***	***Changed by***	***Changes made***
0.8	18/06/2012	Michele Zanin	First Version
0.9	22/06/2012	Michele Zanin	With initial contributions from partners
1.0	27/06/2012	Paul Chippendale	First check of quasi completed deliverable
1.1	04/07/2012	Michele Zanin	Integration of new contributions from partners; section 4.
1.2	07/07/2012	Michele Zanin	Section 4; review of Section 3; first version of conclusions
1.3	11/07/2012	Paul Chippendale	Final check before submission

Table of Contents

Executive Summary

Scope

This document presents an updated list of VeDi platform requirements, taking into account the current and expected evolution of WP4 algorithmic modules.

Audience

This deliverable is confidential, restricted to members of the consortium only.

Summary

This document analyses the algorithmic modules that are going to be developed in the context of WP4 and that are intended to be ported onto the VENTURI mobile platform VeDi. The goal of this analysis is to identify and define expected platform requirements imposed by WP4 algorithms, starting from what has already been identified in D2.1.1 [1] and successively refined in D2.2.1 [2]. The final use of this document is in the context of WP2, where the provided requirement list will influence the design phase of the next generation VeDi platform. At the time of writing, many of the considered WP4 modules are still in a very preliminary stage, so a large part of the improved requirement list presented here is based on reasonable, experience-based assumptions, and not on quantitative facts.

Structure

This deliverable is structured as follows: Section 1 overviews the topics of the document. Section 2 briefly presents the current platform requirements, as discussed in deliverable D2.2.1 [1]. Section 3 hypothesizes the algorithmic architecture of WP4, discussing each module's input/output interface, data flow, and interaction with other modules, with the goal of deriving reasonable platform requirements. Section 4 presents the same structure as Section 3, but from the converse point of view: each requirement is clearly described, underlining differences with respect to D2.2.1 [2], and making explicit the causal links from algorithms to the considered requirement(s). Section 5 summarizes this report and draws some conclusions.

1. Introduction

This document analyses the algorithmic modules that are going to be developed in the context of WP4 and will be ported onto the VENTURI mobile platform VeDi. The goal of this analysis is to identify and define expected platform requirements imposed by WP4 algorithms, starting from what has already been identified in D2.1.1 [1] and successively refined in D2.2.1 [2]. The final use of this document is in the context of WP2, where the provided requirement list will influence the design phase of the next generation VeDi platform.

2. Platform requirements from D2.2.1

An initial list of VENTURI platform requirements was presented in deliverable D2.1.1 (*Use cases, application definition and system requirements for STE U8500-based platform*) [1], which was initially derived from an analysis of the project's Use Cases. The requirements list was then analysed, discussed and refined in deliverable D2.2.1 (*Early Detailed Design Specifications for STE U8500-based platform*) [2].

Below, we repeat the requirements list from D2.2.1. It serves as a starting point for sections 3 and 4 where, when possible, the same notation (i.e. HW functional requirements, SW functional requirements, and SW non-functional requirements) and numbering system will be used.

HW functional requirements

HF1 (Camera): Hardware Platform must support a rear colour mono-camera. Camera latency should be feasible for a live view.

The following camera parameters must at least be supported: Resolution needed for the vision part: 640x480, grey-scale; Frequency needed for the vision part: 15fps; Resolution needed for the rendering part: colour images are ideally similar to the display resolution; Frequency needed for the rendering part: 30fps; Latency: max 180ms.

Moreover, image timestamp generation is required, ideally using the same clock as the one used for the inertial sensors.

HF2 (Connectivity): Platform must support at least one cellular network connectivity and one wireless LAN network connectivity. Supported bandwidth must be at least 4Mbps, with a round-trip time of 1000ms as a maximum.

HF3 (Sensors): Platform must support the following sensors: 3 axis accelerometer, 3 axis magnetometer, 3 axis gyroscope.

Sensor sampling frequency should be higher than the camera frame rate. Sensors must provide at least 10-bit resolution samples and support range selection capabilities.

HF4 (Input method): Platform must support touch-panel based human interface input device.

HF5 (Display): Platform must support a display with at least the following characteristics: Size: 3.7"; Resolution: WVGA (480 x 854); Frame rate: 30fps; DPI: 270.

In addition, the platform should be able to support different screen form factors with no, or little, board modifications.

HF6 (Graphics Hardware): Platform must include a graphics chip capable of performing complex 3D-graphics rendering with at least 15000 polygons/s, as a minimum, for all objects in the scene.

HF7 (Audio): Platform must support stereo audio playback via external headsets. The PCM latency must not exceed 300ms and the DAC frequency must not be less than 44100Hz.

HF8 (Power): The platform must be able to sustain at least 1 hour continuous battery operated operation. This is directly linked to SF7.

HF9 (Autonomous mode): Besides development and debug mode, the platform should also be able to be run in autonomous mode, that is battery operated, and no debug console (UART, JTAG, other).

HF10 (RAM memory): The platform must be equipped with at least 4 Gbits Random Access Memory.

HF11 (Storage memory): The platform must be equipped with at least 16 Gbits non-volatile memory.

HF12 (Frequency): Host processor peak value frequency must be at least 1GHz.

SW functional requirements

SF1 (User interface Adaptability): The System User Interface must be able to adapt to different screen sizes and form-factors with limited or no modifications to the application.

SF2 (Offline mode): The terminal must be able to operate without network connectivity, using cached data.

SF3 (Computing resources access): All non-critical platform computational resources must be accessible to the application. For example by means of standard APIs such as: OpenGL|ES, OpenCL, Renderscript, OpenMAX.

SF4-1 (Sensors Access, access to positioning sensors): The application must be able to access the platform's positioning sensor resources provided by the hardware platform by means of Operating System or other standard APIs.

SF4-2 (Sensors Access, video frames time-stamping): Video frames have to be time-stamped in an accurate manner. This requirement overlaps SF8.

SF4-3 (Sensors Access, positioning sensors time-stamping): (Positioning) Sensor samples have to be time-stamped in an accurate manner.

SF4-4 (Sensors Access, same time-base): To permit (positioning) sensor samples to be synchronised at the application level with the video frames, the sensor samples need to be time-stamped with the same time base as the video frames, to avoid offset and drift.

SF5 (Start-up/Exit time): The application must meet average user expectations for start-up time: applications must start in less than 15 seconds, with a notification that progress is on-going (e.g. UI with a clock or progress bar). The application must close gracefully and release all platform resources used during its operation. A shut-down progress bar must be displayed at application closure.

SF6 (Application Size): The application installer package should not be larger than 20 MB. This includes all code and application resources (icons, background images, etc.) but does not include any multimedia content.

SF7 (Power Management): The application should be able to run without interruptions for at least one hour with no external power supply. This requirement is strictly related to HF8 and should be treated as a transversal hardware/software requirement.

SF8 (Augmented Reality Video Pipe): The camera resolution needed for the rendering part: colour images are ideally similar to the display resolution. Camera frequency needed for the rendering part: 30fps. Time-stamps for the camera images (ideally using the same clock as the one used for the inertial sensors) must be supported.

SF9 (Synchronization of AR Video Pipe and Rendering Pipe): Video pipeline should provide two synchronized image qualities: one grey-scale low-resolution for the vision part, and one colour high-resolution for the rendering part.

SF10 (Replay Mode) [Optional]: To guarantee platform benchmarking repeatability, a modality to record and playback events that occur during platform operation, in a specific use-case mode, is needed. This modality will be added only if time and resources will be available.

SF11 (Exposition of Camera and ISP statistics): The application must be able to access camera and ISP statistics by means of Operating System or other APIs. The operating system should be capable of delivering camera statistics to the application, in viewfinder mode. The required statistics, which are to be exploited for context sensing and interpretation within the WP4, are: Exposure time; Aperture; Gain (ISO); White balance and possibly the mean values of the R, G, B colour planes; Focus; Global histogram (if available); Face position and size (if available); DCT coefficients (if accessible and available in viewfinder mode).

SW non-functional requirements

SNF1 (Portability): The application will run on Android OS. Nevertheless, applications should be architected in a way to enable the straightforward porting across different mobile operating systems (C/C++ code that can be recompiled on different OS).

SNF2 (Stability): The application shall not exhibit force closes or hangs. The performance of other applications running on the platform should not be impaired by the VENTURI stack.

SNF3 (Extensibility): The application should be extendable with new functionalities without requiring a user's manual intervention.

SNF4 (Restricted Access): Application shall enforce basic access control for privacy-sensitive content: personalized access or secure login capabilities.

SNF5 (Observability): The VENTURI system should be observable by means of software or hardware assisted profiling and tracing tools for performance and power consumption characterization.

SNF6 (Scalability): The VENTURI system should be scalable with respect to the number of users and the number of augmented reality content sources.

SNF7 (Debug): The Software platform should expose software ports for application and operating system debugging purposes, possibly using de-facto standard tools.

3. Algorithmic modules

Work Package 4 includes a wide range of algorithms for sensing the surroundings of the user. The individual algorithms are intended to interact closely with other parts of the systems, with the goal of satisfying the list of expected outcomes detailed in the WP4 section of the DoW [3].

This section is dedicated exclusively to WP4 algorithms that will be ported onto the Android-based VeDi platform.

The task-by-task structure of the DoW will be implemented following the preliminary identification of relevant algorithmic modules. Each module should be seen as a separate entity (comprised of one or more individual algorithms) that interface to the system through a well defined API. It should have an expected behaviour, and it should interact with other modules to produce expected outcomes compatible with the considered task.

In the following list, each module will be analysed to estimate its impact on VeDi platform requirements.

T4.1 Hardware sensor interpretation

The main goal of this task is to provide a precise continuous localisation estimate along a travelled itinerary/path, computed using embedded sensors, camera and map database. The architecture is composed of several on-board modules:

- An on-board router module to compute routes.
- An on-board Pedestrian Dead Reckoning module which comprises:
    - a step detection module;
    - a step calibration module;
    - a step length estimation module;
    - a motion and activity classifier;
    - a map-matching module with a provision for geo-tags (T4.3.1);
    - a heading module with compass, gyroscope and GPS data fusion;
    - an integrity manager to re-compute the route and give warnings when leaving it.

The input to this system is:

- An OpenStreetMap navigation network (indoor and/or outdoor) accessed from a server as an XML document.
- Real-time data from sensors: the system uses the nine usual IMU sensors plus the barometric sensor and GPS.
- Data from the camera for contextual information (scene classification, see M3.5).

The output of the system is:

- A tuple composed of position, heading and altitude; at a frequency corresponding to that of the foot steps.
- Contextual information like recognized user activities, stairs, lift, door, tactile paving, etc. in the form of events.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF3 (Sensors)***: frequency of 60 Hz minimum.
    - Access to barometric sensor with a relative 3 meters altitude precision (i.e. one floor inside a building)

T4.2 Auditory scene analysis

The goal of this task is not only to do semantic audio classification, but also to contribute to indoor localization without the need of additional infrastructure. This task will therefore complement T4.1 in all these aspects and could allow, for example, global positioning through audio-fingerprinting. The architecture is composed of:

- A module for standard spectral analysis.
- A module for audio-fingerprint extraction.

The input to this system is:

- Audio sample.
- Audio-fingerprints.
- OpenStreetMap semantic tags.

The output of the system is:

- OpenStreetMap semantic tags with a confidence interval.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF13 (Audio Recording)***: microphone circuitry on phones is typically designed for recording speech: 16-bit at 44.1kHz. Better performance could be required for the next VEDI platform depending on the result of the experiments.

T4.3.1 Matching of camera content with nearby geo-tagged imagery

The main goal of this subtask is to compare acquired live-video content to a database of geo-referenced imagery. At the current stage of development, it seems necessary to create and update the database off the phone and to store it on a remote server. On-board modules would then communicate with the remote server and accomplish the considered task (see requirement HF2). There are currently two architectural hypotheses, with different levels of complexity, communication models, and platform requirements. Their complementarity means that they are not mutually exclusive, and for different application contexts, one or the other could be preferable. We present here both hypotheses:

1. The first architecture is composed of two on-board modules: M3.1 (Information Extraction) and M3.2 (Position Refinement). Both of them communicate with a remote server and are detailed below.
2. The second architecture is composed of two on-board modules: M3.3 (Location Sender) and M3.4 (Object Matching). Both of them communicate with a remote server and are detailed below.

In addition, this task includes a scene classification module (M3.5, full description below) that will provide contextual information of general interest (i.e. not only to WP4 modules), and will also help other modules in T4.3.1 to search for objects/images that could indicate scene context. At the moment of writing, it is not clear if contextual information from M3.5 will reach other T4.3.1 modules directly, or through the integration modules of T4.5.

For obvious reason, both architectures impose connectivity requirements. Their needs are however already covered in the current version of HF2 (Connectivity). The same happens also for many other WP4 algorithmic modules.

M3.1 Information Extraction

This module needs to have access to camera data;

- Minimum image resolution is 640x480, in RGB format.

- Location and orientation information, either raw from sensors, or already improved by other WP4 modules.
- Optionally, it could also benefit from contextual information from M3.5.

M3.1 analyses the considered image and produces one or more of the following outputs (depending on context and available computational power):

- Image signature: a compact representation of the image, based on appropriate statistics.
- SIFT-like feature points and descriptors.
- Significant objects and a compact description of their visual appearance.
- Motion-based segmentation of foreground and background regions.

M3.1 then sends its outputs (together with location / orientation / contextual data) to a remote server. The remote servers matches this data to a geo-imagery database and returns a list of recognized regions, their identification, and precise geo-positioning.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: minimum resolution 640x480, RGB.
    - Access to the Spectral Sensitivity Functions of the Cameras.

Discussion about RGB requirement (valid also for M3.4 and computation on server): the algorithm we intend to use for discarding colour variations between images due to a change of illuminant, works in the RGB colour space. More precisely, it takes as an input the histograms of the colour channels red, green, blue. The colour quantization that guarantees the best performances in terms of colour correction and illuminant invariant image retrieval is 256. This method can be used for correcting the colours of two images A and B as they would be taken under the same illuminants. No doubt, this will also be exploited in the rendering aspects of WP5. The method is described in [4].

Discussion about Spectral Sensitivity Function (valid also for M3.4 and computation on server): The algorithm, we intend to use for removing shadows and managing changes of Planck's illuminants (for instance daylight, sun light, and fluorescent lamps, which are common in outdoor and indoor environments), requires as an input an image (whose colours are codified into the RGB space) and the spectral sensitivity functions of the camera. More precisely, for each RGB channel, we need to know the wavelength at which the sensor is maximally sensitive. The algorithm assumes that the sensors are narrow band. The method is detailed in [5].

M3.2 Position Refinement

This module works together with M3.1, taking as an input results from the remote server, i.e. a list of recognized regions, their identification, and precise geo-positioning. Its goal is to refine location and orientation of the device. It could also propagate geo-recognized objects to other VENTURI modules.

This module does not impose any modification on requirement list presented in Section 2.

M3.3 Location Sender

A very simple module that collects the current location and orientation hypothesis and sends this information to the remote server. The server, starting only from device location and orientation, analyses previously taken nearby images, looking for common groups of features, thus identifying candidate objects that could be visible also in the current picture. Subsequently, the server returns the visual appearance of expected nearby objects, maybe in the form of key-points with descriptors, texture signatures, etc.

This module does not impose any modification on requirement list presented in Section 2.

M3.4 Object Matching

This module works together with M3.3, taking as an input the current image (colour space RGB) and the results returned from the remote server, i.e. a list of expected nearby objects, each of them characterized by a representation of its visual appearance. The goal of M3.4 is to recognize instances of the expected objects and, if successful, refine the current estimate of location and orientation of the device. It could also propagate geo-recognized objects to other VENTURI modules.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: minimum resolution 640x480, RGB
  - Access to the Spectral Sensitivity Functions of the Cameras

M3.5 Scene Classification

The goal of this module is to classify, in a robust and computationally economical way, the visual input of the device into a set of predefined relevant classes, e.g. day vs. night, indoor vs. outdoor, manmade vs. natural.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: the minimum image resolution is 640x480, in YUV 4:2:2 or 4:2:0 format.
- ***SF11 (Exposition of Camera and ISP statistics)***: no changes with respect to SF11 in Section2. Note: *exposure, aperture, gain and white balance statistics, together with the global histogram, could help in discriminating day/night and indoor/outdoor classes. The focus measure, together with aperture, can help to distinguish between landscape/close-up conditions. Face position and size could be exploited to detect a portrait scene, where other semantic categories are not relevant. DCT coefficients could be used to extract features from the image, in order to detect more specific classes (e.g. home rooms, corridor vs. shelves in a supermarket, etc.). In addition to these features, the orientation of the device (portrait/landscape) could be useful as input to the scene classification to simplify the detection of a certain semantic class. Finally, the scene classification algorithm could benefit from the output of other modules, such as audio scene analysis, or Pedestrian Dead Reckoning (PDR) module. For example, the PDR module can find a region where the probability of a pedestrian crossing is high, and the scene classifier module could be activated to confirm or not this condition, according to the visual content of the scene.*

T4.3.2 Matching of camera content with geo-referenced synthetic 3D models

This tasks aims to directly match photo content to synthetic 3D models. In this context, two approaches are considered: matching content to terrain elevation data (2½D models, DEMs, seen from far away, e.g. mountains) and matching content to architectural models (3D models).

In the first case – terrain elevation data – we envision the collaboration of two algorithmic modules, M3.6 (DEM rendering on mobile) and M3.7 (Alignment Refinement). An optional third module M3.8 (Feature tracking) could improve final results.

In the second case – architectural 3D models – a dedicated module M3.9 (Full 3d) is described below.

M3.6 DEM rendering on mobile

This module generates a 360 degree synthetic representation of the terrain as viewed from the current VeDi position. Its inputs includes:

- Current location hypothesis.
- DEM raster covering a wide area (e.g. an entire country, a mountain range). Specialized compression algorithms will be considered [6].
- A database of geo-referenced Points of Interest POIs (optional, only if relevant for the considered application scenario).

- Orientation data from on-board sensors (optional, used only to prioritize the rendering of current direction).

M3.6 generates the following outputs:

- High resolution 360 degree vector representation of terrain profiles (i.e. depth discontinuities from the observer’s point of view) in spherical coordinates.
- POIs visible from the current location and their angular position (optional).
- Expected shadows, considering terrain shape and Sun’s position in the sky (optional).

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF6 (Graphics Hardware)***: a graphics chip capable of performing 3D-graphics rendering is needed by this task but, at the moment of writing, it is not known if the quantitative measure in Section 2 (i.e. 15000 polygons/s) is adequate. Graphics Hardware requirements for M3.6 will become clearer in the next few months.
- ***SF6 (Application size)***: while the code part of M3.6 will have a small impact on SF6, the DEM raster and the optional POIs database could easily be much larger than 20MB (even if compressed), so must be treated in the same way of other external multimedia content. If necessary for demonstration purposes, DEM raster could cover a very small region and therefore be compatible with the 20MB limit.

M3.6 also requires access to standard APIs OpenGL|ES, but that is already covered in the current version of SF3 (Computing resources access). The same also happens for other WP4 modules.

M3.7 Alignment Refinement

Given the synthetic profiles generated by module M3.6 and the current image from the VeDi camera, the goal is to align them, deriving a pixel-precise orientation measure and, if possible, correct small geographical positioning errors. From our current knowledge, the general problem of finding the alignment using only profiles and current image, has memory and computational requirements that are not compatible with a porting to a VeDi device. The more approachable problem of starting from an orientation hypothesis (provided by on-board sensors) and refining it until a match is found, is the core functionality of this module. Its input data includes:

- Current image to be aligned
- All available outputs from M3.6 for current location
- Orientation data from on-board sensors

M3.7 generates the following output data:

- Refined orientation data and confidence score.
- Indirect estimation of sensor error and, if relevant, offset to apply in order to improve hardware orientation measures.
- Refined geographical location (optional, only if the terrain-device configuration is favourable).

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: minimum resolution 640x480, grey-scale.
- ***SF-* (Sensor Access)***: access to sensor data is needed. Possibility to access low-level data (to apply previously computed correction offsets) is desirable.

M3.8 Feature Tracking

It is not certain if task T4.3.2 will require a feature tracking module, but one promising architecture contemplates its use. It is imagined that M3.7 will be too slow to perform image-based orientation refinement on every frame. It may therefore be necessary to use M3.7 once every N frames, and compute relative re-orientations for other frames thanks to a combination of sensor data (corrected with M3.7 offsets) and visual tracking. M3.8 is to be considered optional and, if other modules with similar functionalities are already running on the system (e.g. metaio SDK's tracker), their results could be directly exploited avoiding unnecessary duplication.

This module does not impose any modification on requirement list presented in Section 2.

M3.9 Full 3d

It is assumed that the object depicted in the image has been identified in a prior stage, based for example on the approximate location of the user and feature-based classification approaches (see task T4.3.1). The goal of this task is to estimate camera/object pose, thus accurately registering a geo-tagged 3D model to the image.

In this task, the algorithms to be developed are expected to be resource demanding in terms of CPU and RAM. To ease the investigation/evaluation of different registration strategies and enable early integration into the mobile platform, a two-phased implementation strategy will be followed. In the first phase, the mobile device will act merely as an interface for capturing and transmitting high-resolution photos to a server in the cloud. The server will perform the 2D-3D registration process and return a 3D-model along with registration data back to the mobile device, e.g. for rendering in the AR framework. Based on the results from this first development phase, the implementation of an optimized version of the registration algorithms running directly on the mobile device will be investigated in the second phase.

A list of expected requirements has been compiled below, with the more detailed requirements for an optimized mobile implementation still to be finalized when the first development phase has been completed.

Language / runtime requirements:

- High-level integration tasks (user interaction, image capturing, network communication, etc.) will target the Java language layer and SDK, or alternatively rely on functionality provided by the junaio AR browser.
- Image processing tasks to be implemented on the mobile device will rely on C++ for performance and resource efficiency.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***:
  - Access to the back-side camera for capturing colour images at high resolution (5+ MP) is required. Note: Accurate 2D-image/3D-model registration requires colour images at higher resolution than real-time visual tracking.
  - If real-time visual tracking shall be performed in parallel to the 2D/3D registration, it will therefore be necessary to have a second back-side camera. Having a stereo back-side camera available would be advantageous in this case, as one camera could be used for high-resolution still-image acquisition, while the other camera could be used for visual tracking in lower resolution video mode.
- ***HF10 (RAM)***: Memory requirements of the algorithms are still unknown, but 1GB of RAM is assumed to be available for processing.

T4.3.3 Unconstrained text detection

Depending on the considered scenario, detecting and recognizing text embedded in the environment could present different levels of difficulties. In some cases, the user could point the VeDi device towards the desired text and acquire an high resolution image to elaborate; in other cases, the video stream could be used directly without user cooperation. It seems reasonable to divide this task into two algorithmic modules that show different goals and complexities: text detection and text recognition.

M3.10 Text detection

There are at least two possible goals for this module. The first is to identify candidate regions of the image that could contain text. The second is more complex and is not only to determine text regions, but also to provide a binary mask that represents segmented candidate text. In this phase of the project, we consider only still images acquired with auto-focus.

Input data:

- Current image.
- Context information (optional, could help selecting which kind of text data to search for).

Output data:

- Image coordinates of detected text regions.
- Bitmaps representing segmented candidate text, with at least 3 aligned characters; image coordinates.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: resolution from 640x480 to 1280x960 (depending on text size), grey-scale. Still image acquired with auto-focus.

M3.11 Text Recognition OCR

The current hypothesis about the text recognition module considers two possible architectures. In the first, the binary mask from M3.10 is sent to an external server that performs OCR and returns the recognized text. In the second, OCR is performed on-board, thanks to off-the-shelf libraries. Preliminary experiments with state-of-the-art OCRs tell us that robust and unconstrained general-purpose text recognition is probably not possible. A dictionary and/or context information appears to be necessary to obtain satisfactory results.

In both cases, input data is:

- Bitmap or grey level zone with text.
- Dictionary and/or contextual information.
- Language.

Output data:

- Recognized text.
- Confidence score.

Impact on platform requirements (with respect to what presented in Section 2):

In the case of remote OCR through a server, there are no special requirements with respect to Section 2.

In the case of on-board OCR, requirements are determined by the chosen OCR library. As an example, let us consider:

- ABBY Mobile OCR SDK 2.0, for Latin and Cyrillic characters: 8 MB ROM, 10 MB RAM, i.e. no impact on requirements with respect to Section 2. http://www.abbyy.com/mobileocr/
- EXPERVISION: 5M ROM, 1MB RAM, i.e. no impact on requirements with respect to Section 2. http://www.expervision.com/ocr-sdk-toolkit/ocr-sdk-for-embedded-mobile-system-iphone-ipad-android-wince

- Free OCR Tesseract (Android port, source available, still unstable): requirements are not clear. https://github.com/rmtheis/android-ocr

T4.4 Physical surroundings modelling

The goal of this task is to capture and model the immediate surroundings of the mobile device to both create a snap-shot of the environment for 'World documentation' and to provide a means of immersive communications with others.

The various algorithmic modules for T4.4 are detailed below and specific requirements stated.

M4.1 Sparse SfM

The structure from motion algorithm needs to gain access to the following information:

1) Camera data where the maximum image resolution is VGA, YUV 420 format.

2) To register the environment to be augmented offline, the system requires:

a) A sparse cloud of 3D points, with associated X,Y,Z coordinates and their coordinate reference system

b) The source 2D points (X,Y) coordinates per each image in which the 3D points (see point a) were triangulated.

c)The set of images from which 2D points (see point b) were detected.

Note: pre-computed information is also crucial to extract 2D image descriptors associated to 3D points for online camera estimation.

M4.2 Pixel-dense 3d reconstruction

The pixel-dense 3D reconstruction approaches to be developed in this task are expected to be more demanding in terms of RAM and CPU than sparse feature-based 3D reconstruction methods. An implementation of dense reconstruction algorithms running exclusively on the mobile device may therefore not be feasible using the current platform. A two-phased implementation strategy will be followed in this case: In the first phase, the mobile device will act as an interface for capturing and transmitting high-resolution photos to a server in the cloud. The server will perform the dense reconstruction and return the 3D-model back to the mobile device, e.g. for AR rendering, or for storing it in a "user-library of world objects". In a second phase, algorithms will be optimized for porting to the mobile platforms.

A list of expected requirements has been compiled below, with the more detailed requirements for an optimized mobile implementation still to be finalized when the first development phase has been completed.

Language / runtime requirements:

- High-level integration tasks (user interaction, image capturing, network communication, etc.) will target the Java language layer and SDK, or alternatively rely on functionality provided by the junaio AR browser.
- Image processing tasks to be implemented on the mobile device will rely on C++ for performance and resource efficiency.

Impact on platform requirements:

- ***HF1 (Camera)***:
  - Access to the back-side camera for capturing colour images at high resolution (5+ MP) is required.

- Dense 3D reconstruction requires colour images at much higher resolution than real-time visual tracking. If real-time visual tracking needs to be performed in parallel with image acquisition for 3D reconstruction, it will therefore be necessary to have a second back-side camera. Having a stereo back-side camera available would be advantageous in this case, as one camera could be used for high-resolution still-image acquisition, while the other camera could be used for visual tracking in lower resolution video mode.

M4.3 Mobile user gesture tool

In this subtask, depending on the carrying modality of the device, e.g. dangling from a neck mounted chain for blind people, or handheld, the visual streams from the front- and/or rear-facing cameras will be used to track a user's facial and hand positions with respect to the device. This information will be exploited in WP5 to provide enhanced modalities of content presentation (e.g. AR information that changes with a user's point of view). In cases where a user's hands can be detected, e.g. when in a chest mounted configuration and hand are in from of the device, then basic gesture interpretation will be performed and results passed to WP5 to improve interaction flexibility.

A list of expected requirements has been compiled below, with the more detailed requirements for an optimized mobile implementation still to be finalized when the first development phase has been completed.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: Access to the front-side and back-side cameras simultaneously for capturing real-time colour images at medium resolution (1 MP) is required. Front or back stereo cameras would provide better precision if available, offering the possibility of eye tracking.

M4.4 Ad-hoc interactive surface locator tool

In this subtask, planar or other suitable surfaces potentially useful as ad-hoc interactive surfaces through projection will be automatically identified. This information will be exploited in WP5 to provide users equipped with devices containing pico-projector, the ability to enhance /augment the real-world through actual content overlay improving AR interaction flexibility.

A list of expected requirements has been compiled below, with the more detailed requirements for an optimized mobile implementation still to be finalized when the first development phase has been completed.

Impact on platform requirements (with respect to what presented in Section 2):

- ***HF1 (Camera)***: Access to the front-side and back-side cameras simultaneously for capturing real-time colour images at medium resolution (1 MP) is required. Front or back stereo cameras would provide better precision if available, offering the possibility of eye tracking.
- ***HF2 (Connectivity)***: Ad-hoc connectivity (without a server infrastructure) to other devices (either Bluetooth or WiFi) would be required for collaborative/social projection scenarios.
- ***HF5 (Display)***: Platform must support dual display output channels, one for the device and the other to feed content to the pico-projector (most probably via HDMI).
- ***SF12 (External Display)***: It must be possible to visualize different contents on the on-board display and on the external pico-projector.

T4.5 What is, or could be, going on around me?

For the moment, we do not foresee any particular impact of this task on platform requirements.

4. Updated list of requirements

Section 3 already contains all of the information about the expected platform requirements of WP4 algorithms. However, it is not straightforward to appreciate the differences and improvements with respect to what was already detailed in Section 2.

This section presents the same information in a concise manner and from the point of view of individual requirements, concentrating only on new requirements (i.e. not present in Section 2) and modified ones (i.e. unchanged requirements are ignored here).

Unchanged HF* requirements

WP4 algorithms have no impact on the following HW functional requirements (i.e. they are not relevant or the current version in Section 2 already covers what is needed):

- HF4 (Input method)
- HF7 (Audio)
- HF8 (Power)
- HF9 (Autonomous mode)
- HF11 (Storage memory)
- HF12 (Frequency)

HF1 (Camera)

Since the majority of WP4 algorithms are vision-based, HF1 becomes one of the most relevant requirements for the present document. The previous version of HF1 described in Section 2 can be modified and integrated as follows.

HF1 (Camera):Hardware platform must support at least one front (M4.3) and at least one rear colour camera. Ideally, there should be a front stereo pair (M4.3) and two rear cameras that can work together as a stereo pair (M4.3) or even independently (M4.2). Front and rear camera/s should be accessible simultaneously (M4.3, M4.4). Camera latencies should be feasible for a live view.

Spectral Sensitivity Functions of all cameras must be known and shared with VENTURI partners (M3.1, M3.4).

Most algorithms require a minimum resolution of 640x480, but it must be possible to acquire still pictures (with or without auto-focus) at a resolution of at least 5 MPixels (M3.9, M3.10, M4.2). The frequency needed for the vision part is 15fps.

Resolution needed for the rendering part: colour images are ideally similar to the display resolution; Frequency needed for the rendering part: 30fps. Latency: max 180ms.

Moreover, image time-stamp generation is required, ideally using the same clock as the one used for the inertial sensors.

HF2 (Connectivity)

As noted in M4.4 description, Bluetooth capabilities and support for ad-hoc server-less connectivity are required. HF2 becomes:

HF2 (Connectivity): Platform must support at least one cellular network connectivity and one wireless LAN network connectivity. Supported bandwidth must be at least 4Mbps, with a round-trip time of 1000ms as a maximum.

Ad-hoc connectivity (without a server infrastructure) to other devices (either Bluetooth or WiFi) would be required for collaborative/social projection scenarios.

HF3 (Sensors)

In addition to HF3 in Section 2, the algorithms from T4.1 requires a barometric pressure sensor and a minimum sensor frequency of 60Hz. The updated formulation of HF3 is:

HF3 (Sensors): Platform must support the following sensors: 3 axis accelerometers, 3 axis magnetometer, 3 axis gyroscope, a barometric pressure sensor with a 3 meters relative precision (one level inside a building) (see T4.1 in Section 3).

Sensor sampling frequency should be higher than the camera frame rate, and not less than 60Hz. Sensors must provide at least 10-bit resolution samples and support range selection capabilities.

HF5 (Display)

Module M4.4 adds a dual display requirement:

HF5 (Display): Platform must support a display with at least the following characteristics: Size: 3.7”; Resolution: WVGA (480 x 854); Frame rate: 30fps; DPI: 270.

In addition, the platform should be able to support different screen form factors with no, or little, board modifications.

Platform must support dual display output channels, one for the device and the other to feed content to the pico-projector (most probably via HDMI).

HF6 (Graphics Hardware)

The current formulation of HF6 is confirmed, but at the moment of writing it is not clear if the provided figure of 15000 polygons/s is sufficient or not. A better estimation of the quantitative part of HF6 will probably be available during the next few months.

HF10 (RAM memory)

Module M3.9 explicitly requires at least 1GB of RAM, so the requirement becomes as follows:

HF10 (RAM memory): The platform must be equipped with at least 1GB Random Access Memory.

HF13 (Audio acquisition) [new]

HF7 (Audio) requirement in Section 2 deals with audio playback, while T4.2 imposes requirements about audio acquisition. A new requirement HF13 is then introduced:

HF13 (Audio acquisition): microphone audio circuitry on the platform is designed for recording speech: 16-bit at 44.1kHz. Better performance could be required for the next VeDi platform, depending on the result of the experiments (see T4.2).

Unchanged SF* requirements

WP4 algorithms have no impact on the following SW functional requirements (i.e. they are not relevant or the current version in Section 2 already covers what is needed):

- SF1 (User interface Adaptability)
- SF2 (Offline mode)

- SF3 (Computing resources access)
- SF4-2 (Sensors Access, video frames time-stamping)
- SF4-3 (Sensors Access, positioning sensors time-stamping)
- SF4-4 (Sensors Access, same time-base)
- SF5 (Start-up/Exit time)
- SF7 (Power Management)
- SF8 (Augmented Reality Video Pipe)
- SF10 (Replay Mode) [optional]
- SF11 (Exposition of Camera and ISP statistics)

SF4-1 (Sensors Access, access to positioning sensors)

Just an additional note from M3.7: Possibility to access low-level data (to apply previously computed correction offsets) is desirable.

SF6 (Application Size)

M3.6 and M3.9 extend SF6 in the following way.

SF6 (Application Size): The application installer package should not be larger than 20 MB. This includes all code and application resources (icons, background images, etc.) but does not include any multimedia content, terrain model, POIs database, 3D model.

SF9 (Synchronization of AR Video Pipe and Rendering Pipe)

As noted in Section 3, some algorithms require RGB colour space, while others prefer YUV 4:2:0 (M3.5, M4.1) or just the Y component. SF9 needs then to be updated, since in its previous version the only colour space was grey-scale.

The different colour space requirements needs accurate software design in order to avoid the duplication of unnecessary conversions. Probably, colour space conversions should be centralized and not demanded to the individual module.

SF9 (Synchronization of AR Video Pipe and Rendering Pipe): Video pipeline should provide two synchronized image qualities: one colour low-resolution for the vision part, and one colour high-resolution for the rendering part.

SF12 (External Display) [new]

This new software requirement is directly linked to HF5 (Display) and is about the possibility to determine what to display on-board and what externally (i.e. with a connected pico-projector).

SF12 (External Display): In the context of pico-projector use, it must be possible to visualize different contents on the on-board display and on the external pico-projector.

SW non-functional requirements

WP4 algorithms have no impact on SW non-functional requirements.

5. Conclusions

In order to derive reasonable requirements, we re-organized the WP4 algorithms into a modular architecture, describing expected inputs, outputs and connections of each component. While the proposed architecture is very preliminary (it will be refined and finalized in the context of WP6), its analysis helps to understand how to update the requirement list of D2.2.1 [2].

The final version of the entire updated requirement list was presented in Section 4. The most significant impact of WP4 algorithms happens to be on HF1 (Camera), but many other HW Functional and SW Functional requirements need to be changed in order to meet WP4 needs. Moreover, two new requirements are now integrating the list: HF13 (Audio Acquisition), and SF12 (External Display).

It is important to remark that, at the time of writing, many of the considered WP4 modules are still in a very preliminary stage, so a large part of the improved requirement list presented here is based on reasonable, experience-based assumptions, and not on quantitative facts.

References

[1] VENTURI consortium, "D2.1.1: Use cases, application definition and system requirements for STE U8500-based platform," 2011.

[2] VENTURI consortium, "D2.2.1: Early Detailed Design Specifications for STE L9500-based Platform," 2012.

[3] VENTURI consortium, "Annex I - Description of Work," 2011.

[4] M. Lecca and S. Messelodi, "llluminant Change Estimation via Minimization of Color Histogran Divergence," in *Computational Color Imaging Workshop - CCIW 2009*, Etienne, France, 2009.

[5] C. Finlayson and S. Hordley, "Color constancy at a pixel," *J.Optical Soc. Am. A,* vol. 18, no. 2, pp. 253-264, 2001.

[6] R. Chen and X. Li, "DEM Compression Based on Integer Wavelet Transform," *Geo-spatial Information Science,* vol. 10, no. 2, pp. 133-136, 2007.