ECSEL2017-1-737451

FitOptiVis

From the cloud to the edge - smart IntegraTion and OPtimisation Technologies for highly efficient Image and VIdeo processing Systems

Deliverable:
D5.1 - Components Analysis and Specification

Due date of deliverable: (31-5-2019)
Actual submission date: (11-06-2019)

Start date of Project: 01 June 2018 Duration: 36 months

Responsible: Zaid Al-Ars (Delft University of Technology)

Revision: draft

Dissemination level		
PU	Public	✔
PP	Restricted to other programme participants (including the Commission Service	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (excluding the Commission Services)	

DOCUMENT INFO

Author

Author	Company	E-mail
Francesca Palumbo	UNISS	fpalumbo@uniss.it
Carlo Sau, Tiziana Fanni, Luigi Raffo	UNICA	carlo.sau@diee.unica.it; tiziana.fanni@diee.unica.it; raffo@unica.it
Zaid Al-Ars	TUDelft	z.al-ars@tudelft.nl
Pablo Chaves David Pampliega	SCHN	pablo.chaves@se.com david.pampliega@se.com
Pablo Sánchez	UC	sanchez@teisa.unican.es
Roman Čečil	UWB	rcecil@kky.zcu.cz

Document history

Document version #	Date	Change
V0.1	25/01/2019	Added table of content and initial inputs from survey
V0.2	07/05/2019	Integrated partner contributions from Feb to Apr
V0.3	14/05/2019	Integrated partner contributions till May 14
V0.4	22/05/2019	Integrated partner contributions till Brno meeting
V0.5	29/05/2019	Integrated partner contributions till May 29
V0.6	02/06/2019	Update based on feedback of internal review
Sign off	03/06/2019	Update figure captions
V1.0	08/06/2019	Finalize and upload for submission

Document data

Keywords	Hardware, software, networking
Editor Address data	Name: Zaid Al-Ars Partner: Delft University of Technology Address: Mekelweg 4, 2628 RJ Delft, NL Phone: +31152789097

Distribution list

Date	Issue	E-mailer
12-06-2019	Final	Patrick.vandenberghe@ecsel.europe.eu

Table of Contents

1 Executive summary

This deliverable is meant to report the results of WP5 during the first project year.

D5.1 Components Analysis and Specification – Report - M12 – Task Involved: 5.1 and 5.3 – PU - TUD
This deliverable is meant to report the results of the analysis carried out in T5.1 and T5.3 on state of the art and commercial SW and HW components with respect to the use case needs and requirements.

The content of this deliverable contributes to achieve project milestone MS3: Preliminary components and methods release with standalone assessment. Therefore, the initial list of identified components is reported, with a short analysis of their respect state of the art and commercial positioning (if available). In addition, the deliverable discusses plans for the using the components within the FitOptiVis project use cases, as well as the ongoing activities to extend the different components. The partners identified the TRL level of their components at the beginning of the project, and the expected TRL level at the end of the project.

This deliverable, since also activities in T5.2 are started, reports on the activity of all the three tasks of WP5 and focuses on both HW, SW IPs, as well as on communication components. All 22 partners of WP5 contributed a total of 28 of their respective component developments. All use cases of the FitOptiVis project have been covered by the contributed component developments.

2 Introduction

This deliverable is meant to report the results of WP5 during the first project year. The content of this deliverable contributes to achieve project milestone MS3: Preliminary components and methods release with standalone assessment. Therefore, the initial list of identified components is reported, with a short analysis of their respect state of the art and commercial positioning (if available). In addition, the deliverable discusses plans for the using the components within the FitOptiVis project use cases, as well as the ongoing activities to extend the different components. The partners identified the TRL level of their components at the beginning of the project, and the expected TRL level at the end of the project.

We refer to a component as a part of the computing platform or as a part of the application. The granularity of components that are going to be presented in this document is highly variable; some of them are atomic while others represent compositions of sub-elements. In the deliverable, we will refer to the following:

- HW IPs (Section 3) are intended as physical realization of processing units and application specific accelerators, described using a HW specification language (e.g. Verilog or VHDL). The SW code to be executed may not be necessarily defined and some of them could be highly use-case dependent.
- SW IPs (Section 4) are intended as an algorithm, described using a SW specification language (e.g. C, C++, MatLab, etc.), handling a specific processing issue. The final physical platform, where they will be executed, may not be necessarily defined and some of them could be highly use-case dependent.
- Communication-oriented IPs (Section 5) are intended as gateways and on chip traffic managers.

Progressive refinements of the components, and their composing elements or sub-components, are envisioned during the project life cycle.

In the rest of this section, we present the general structure that is going to be adopted in this document.

2.1 General template to be followed [involved partner(s)]

2.1.1 Short description

This section discusses the overall component description, following the topics below:

- Purpose and mission of the components (including the reference platform or software if already defined).
- List of its principal key features and properties.
- Processed inputs and produced outputs.
- Connectivity and communication.

2.1.1.1 Qualities, tunable set-points and available design-time and run-time parameters

This section describes the way the components link to the expected reference architecture discussed in WP2. Specifically the following aspects are discussed.

<u>Qualities:</u>

Intended as the tunable features of the component, upon which a Pareto analysis should be doable in order to take a design-time configuration decisions on the best possible set up. Some qualities may also be tunable at run-time, when this happen, proper explanations are provided in the run-time support section below.

Set-Point(s) and Parameters:

Tunable HW/SW features to allow the possibility of surfing among qualities, which means offering different processing knobs.

2.1.1.2 Design-time support

This section describes the way the components link to the expected design-time support discussed in WP3. This refers to commercial or academic toolsets to be used to handle components design, analysis, verification and deployments.

2.1.1.3 Run-time support

This section describes the way the components link to the expected run-time support discussed in WP4. This refers to the following aspects

Run-time adaptivity support:

Description of what can be (re-)configured or tuned at runtime. Triggers for adaptation are also mentioned.

Monitoring:

Description of what is observed during system/components execution. Monitored data in some cases represent triggers for adaptation.

Programmability and Programming API:

Programmability support of the system/components.

2.1.2 Related state of the art

Brief discussion of the state of the art and explanation of what the innovation is brought by FitOptiVis.

Success stories (usage of this component in other contexts) may also be reported here for those components that are extended in the project.

2.1.3 Extension within FitOptiVis

This section discusses the starting TRL of each component at the beginning of the project and the expected TRL at the end of the project.

TRL@M0:

Comments on current TRL, explaining the positioning.

FitOptiVis Extensions:

Bullet list of what you expect to add/improve.

TRL@M36:

Comments on foreseen TRL, explaining the positioning.

2.1.4 Prospective adoption

- Explain other possible applications.
- Explain in which use case and how this component is going to be used in FitOptiVis.

2.1.5 Related documents

This section lists the references.

3 Processing and acceleration components

This section describes the HW IPs developed by WP5 partners. They represent physical realization of processing units and application specific accelerators, described using a HW specification language (e.g. Verilog or VHDL).

3.1 Customized parallel soft-cores [TUT]

3.1.1 Short description

Customized parallel soft-cores from TUT are based on the Transport-Triggered Architecture (TTA). TTA-Based Co-design environment is a toolset lead developed by TUT which is used to design, test and program TTA-based customized parallel processors that can be implemented as soft-cores in FPGA fabrics.

3.1.1.1 Qualities, tunable set-points and available design-time and run-time parameters

The cores are typically customized to **latency** critical **real-time** tasks. The design-time customization points include register files, function units, operation sets, the interconnection network and number of cores. In SoC customization level the topology between co-processors is left to external tools with the TCE (TTA-based Co-design Environment) tools generating integration points to the system level via e.g. AXI bus interfacing load-store units.

Run-time parameters are design-specific. The co-processors are compiler programmable and thus the functionality can be changed at run-time via loading a new program and resetting the processor.

3.1.1.2 Design-time support

Design-time support is performed with TCE tools. Open source releases of TCE are made available in http://openasip.org.

3.1.1.3 Run-time support

Re-programmability at run-time is supported. Additional monitoring support is developed during FitOptiVis expanding the work in ALMARVI, in which a common IP interface (AlmaIF) was developed that included break point interface and a cycle counter that was memory mapped.

Programmability:

- Software compiler programmable
- Programming APIs: C, OpenCL C 1.2 + few features from OpenCL 2.0+

Monitoring:

- Cycle counters.

3.1.2 Related state of the art

TCE is a state-of-the-art open source application-specific instruction-set processor (ASIP) toolset. There are several commercial ones available since 1980s, but no open source tool with such maturity and that is supported with regular releases that provide compiler upgrades etc.

3.1.3 Extension within FitOptiVis

Within FitOptiVis, there are plans to improve TCE co-processors at least in the following aspects:

- Optimizing TCE cores to FPGA realization; improved clock frequency and reduced resource usage
- Streamlined use in FPGA soft core use
- Improved compiler quality

3.1.4 Prospective adoption FitOptiVis UC

Will be considering for the virtual reality use case for accelerating real time photorealistic graphics rendering by means soft cores in an FPGA.

3.1.5 Related documents

http://openasip.org/documentation.html

3.2 Optimized HW/SW cores [UTIA]

3.2.1 Short description

IP core with AXI stream interfaces for connection to wider system, AXI-lite interfaces for register level control and drivers enabling to control/configure the IP. The IP needs also C/C++ model in form of function executable on Arm A9 or A53 processor. The IP can have direct connect to I/O pins of the device and support various communication protocols. This needs to be also modeled in C/C++ as I/O from/to data files in the file subsystem of the Arm processor.

Input:

- C/C++
- Simulink diagrams

Output:

- Block description for Vivado HLS compatible with the concrete target HW platform(s)/system(s)

Connectivity:

- I/O at Component level AXI stream data with FIFO support and back pressure support (slower IP can slow down the IP on the input)
- DDR: Many components (not all) will have to communicate with DDR3 or DDR4 as HW Masters and have available different communication options (HW DMA, HW SG DMA, HW data mover). Optimally, the communication to the DDR needs to be 64bit wide for Zynq and 128 bit for the Zynq Ultrascale MPSoC devices.

Target:

- Zynq devices (Zynq 7000 28nm and Zynq MPSoC - UltraScale 16nm) with PL part and processor on single chip.

3.2.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Multiple set-points controlled from the AXI-lite register based interface and then valid without change for the processed data stream.

3.2.1.2 Design-time support

- Models availability: C/C++ functions representing functional (but slow) models of the HW IP component. Re-configuration capabilities are not expressed at the model level
- Xilinx SDSoC, Xilinx Vivado HLS
- High level C/C++ Code Debugger.

3.2.1.3 Run-time support

Run-time adaptivity is present, example: at the start of each video frame, the component parameters can be changed (like the number of lines of the frame to be processed by a selected IP) and some connections of components can be reconnected. This can result in changed quality of service like 30 FPS with a single core or 60 FPs with two cores, each handling 1/2 of the frame etc.

Programmability:

- N bitstreams - complete PL reconfiguration with Linux and app running on arm
- IP with firmware reprogrammable FSM and the compiler running on ARM
- Available APIs: SDSoC API, Vivado HLS API, Petalinux UIO API, There are OpenCV libraries compatible with SDSoC

Monitoring:

- Percentage of load of the AXI-Stream
- Cycle counters

3.2.2 Related state of the art

There are no comparable state of the art components developed for the targeted kernels.

3.2.3 Extension within FitOptiVis

TRL @M0:
TRL4 (SDSoC support and results from Almarvi for the Zynq 28nm)

TRL@M36:
TRL6 (SDSoC support for Zynq UltraScale with PetaLinux and Debian file OS)

3.2.4 Prospective adoption FitOptiVis UC

- Calibration of robot arm based on video processing
- Thales Ultrascale video processing system

3.2.5 Related Documents

http://sp.utia.cz/index.php?ids=projects/almarvi
http://sp.utia.cz/index.php?ids=projects/fitoptivis
https://www.trenz-electronic.de/

3.3 NEURAghe [UniCA]

3.3.1 Short description

NEURAghe is an accelerator for Convolutional Neural Network (CNN) based on the Xilinx Zynq SoC. It fully exploit heterogeneity of Zynq platforms: in the programmable logic a Convolution-Specific Processor (CSP) with a convolution engine and a programmable soft-core take care of the bulk of CNN workload; while the ARM General Purpose Processor (GPP) is in charge of executing hard-to-accelerate parts of the computational graph without the need of supervising acceleration thanks to the CSP soft-core. An overview of the system architecture is depicted in the figure below, where separation between GPP and CSP is highlighted.

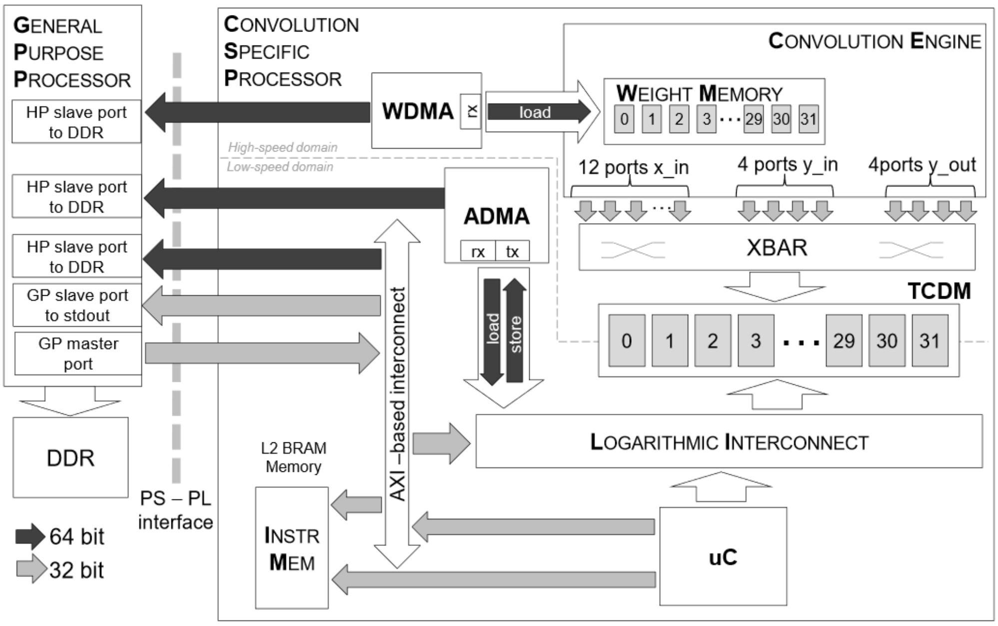

Figure 1: NEURAghe architectural template

The key features of NEURAghe are related to some critical aspects of CNN execution:

- support the deployment of arbitrary CNN topologies;
- acceleration of critical compute-bound operations (i.e. convolutional layers);
- hiding of memory-bound operations (i.e. fully connected layers) by overlapping them with the compute-bound ones.

NEURAghe offers to the developer several degrees of freedom.
Target:
Indeed, besides the CNN to be accelerated (it is generic in this sense), it is possible to target a different Xilinx Zynq SoC device according to the context. In particular, NEURAghe is already compliant with Z7010, Z7010, Z7045 and Ultrascale+, while support for new platforms can be envisioned during FitOptiVis, if needed.
<u>Input:</u>
Being accelerating CNN, the input of the system is data (images, sound, sensor data) under recognition/ classification/ analysis.
<u>Output:</u>
The output is a classification/detection response.
<u>Connectivity:</u>
The connectivity depends on the specific board selected for development. It may be ethernet-based or wireless-based for smaller nodes. The board can be equipped with sensors and cameras by means of the native I/O interfaces.
<u>Composability:</u>
Multiple NEURAghe instances can be composed in a single system using the previously mentioned connectivity. Multiple CSPs can be implemented on the same chip if this may fit the use-case.

3.3.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:

- Gops/s: number of Giga operations per second, depending on the data size and working frequency;
- Gops/w: number of Giga operations per second, depending on the data size and working frequency;
- Accuracy: precision versus performance trade-off can be exploited changing CNN algorithm at runtime.

Set-Point(s) and Parameters:

- network configuration: CNN to be executed and accelerated with NEURAghe;
- data size (8-16 bit): number of bits for fixed point numbers used to process the CNN;
- working frequency: clock frequency of the accelerator (arbitrary).

3.3.1.2 Design-time support

Design time support tools and methods:

- Xilinx Vivado (later than 18.1): commercial tool necessary for the deployment of NEURAghe, properly configured according to the application context, on the target Xilinx Zynq SoC;
- Python scripts: academic set of scripts used for CNN (ONNX) to C conversion to map the desired CNN on a NEURAghe deployed system.

3.3.1.3 Run-time support

Run-time adaptivity support:

- network configuration: CNN to be executed and accelerated which can be changed at run-time among the CNNs that have been generated (Python scripts conversion) at design-time;
- data size (8-16 bit): number of bits for fixed point numbers can be changed at run-time according to the desired Gops/s, Gops/w and accuracy;
- working frequency: clock frequency of the system can be tuned at run-time according to the desired Gops/s and Gops/w;

Monitoring:

- clock cycles: it is possible to measure clock cycles needed for execution of the CNN layers and use them to trigger adaptation, e.g. working frequency.

Programmability and Programming API:

- network configuration (CNN to be accelerated) can be changed by software to any already generated corresponding C description;
- NEURAghe is provided with custom low level C API that are used by the GPP to delegate computation to be accelerated on the CSP.

3.3.2 Related state of the art

Due to the intrinsic parallelism in convolutions, FPGAs are a very promising target technology for the implementation of hardware accelerators for different kind of neural networks. Exploiting dedicated hardware, FPGAs allow realizing very flexible architectures. They can integrate a high number of parallel Digital Signal Processing (DSP) units, enabling efficient implementation of MAC operations, delivering high operational throughput at limited clock frequencies. Moreover, FPGAs offer a significant amount of memory resources, which can be used to map efficient temporary storage buffers to store partial results of the convolutions, and other hardware primitives, such

as registers and look-up tables (LUTs), suitable to be used to implement the glue logic around the accelerators.
Thanks to the extensibility added within FitOptiVis NEURAghe is currently outperforming all existing architectures [1] on Z-7045 using 8-bit and 16-bit data (delivering up to 335 GOps/s or 173 GOps/s, respectively), achieves state-of-the-art performance on Z-7020 using 8-bit data (up to _85 GOps/s), and can be used even on tiny devices such as Z-7007s.
In FitOptiVis, we extend the NEURAghe template-based architecture to guarantee design-time scalability to multi-processor SoCs with vastly different cost, size and power envelope such as Xilinx's Z-7007s, Z-7020 and Z-7045.

3.3.3 Extension within FitOptiVis

TRL@M0:
TRL4: NEURAghe proof of concept has been already demonstrated in lab.
FitOptiVis Extensions:

- Improving efficiency, in terms of Gops/s and Gops/w, and implementability in different SoCs
- Exploration of new operating points for data size (e.g. 4 bit accuracy)
- Extension to less mainstream CNN patterns

TRL@M36:
TRL4: NEURAghe will be validated in laboratory.

3.3.4 Prospective adoption FitOptiVis UC

NEURAghe will be exploited in water supply use case and, in particular, it will be adopted to improve AITEK video processing software modules that are now based on traditional foreground and background separation. Here CNN algorithms can bring enhanced performance especially for tasks related to target classification, such as single shot object detectors where it is required to detect and classify multiple objectives within a video frame.

3.3.5 Related documents

[1] S. Mittal, "A survey of FPGA-based accelerators for Convolutional Neural Networks," Neural Computing and Applications, vol. 30, pp.1–31, 2018.

3.4 AIPHS – UNIVAQ [UnivAQ]

3.4.1 Short description

AIPHS is a library of elements to compose hardware monitoring systems for reconfigurable architectures. It is written in VHDL and it allows developing hardware monitors for AMBA based architectures. The goal is to provide the basic elements to perform a low intrusive monitoring action.
HW Type:
MicroBlaze and ARM Cortex A9 processors have been targeted with AIPHS.
Target:
FPGA, VHDL
Input:
There are two interfaces: initialization/collection results interface and monitoring interface. The former receives commands and provides results, while the latter receives low-level transactions.

Output:
Measures related to selected metrics. The output is parsable in Common Trace Format using a Python script.

3.4.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Quality: It depends on the configuration of the monitoring system.
Set-points: Different configurations available, grouped by metrics type: memory exploitation metrics, HW communication metrics, overall execution time metrics and code coverage metrics.

3.4.1.2 Design-time support

AIPHS requires tool to implement HW architectures on FPGA and Python 3 interpreter to parse results in Common Trace Format. Starting from them, it allows obtaining a custom monitoring system that can be connected to different on-chip interfaces. Supported interfaces depend on the content of some libraries that can be updated with new ones.

3.4.1.3 Run-time support

Programmability:
Programming APIs available

- Hardware description:
  - Technology: VHSIC Hardware Description Language (VHDL)
  - Version: IEEE 1076-2008
- Interaction with monitoring system with embedded Linux:
  - Technology: Native POSIX Threads Library (NPTL)
  - Version: 2.19
- Parse of trace data:
  - Technology: Common Trace Format (CTF)
  - Version: v1.8.2

Monitoring:

- Number of read and/or write operations in a given period of time for a specified memory area for a specified memory.
- Elapsed time between two memory accesses on a specified memory location for a specified memory, expressed in number of clock cycles.
- Total number of cache hits and misses in a given period for a specified cache memory
- Number of transactions occurring during a specified monitoring period over a monitored bus. A transaction is defined depending on the monitored bus.
- Size of data transferred over a monitored bus during a specified monitoring period.
- Amount of time required for burst data transfer, expressed in number of clock cycles. The definition of burst data transfer is implied by the selection of the targeted bus.
- Amount of time required for a single transaction execution. A transaction is defined depending on the monitored bus.
- Time required from address generation to the first service of data in a single bus transaction, expressed in number of clock cycles. A transaction is defined depending on the monitored bus.

- Overall run-time of specific parts of applications (indicated by means of segments of code) executed on a single core. Support for multi-core is work-in-progress.
- Percentage of instruction covered by a set of runs.

3.4.2 Related state of the art

A monitoring system can be useful in different applications, such as support for software developers to debug systems obtained through HLS tools [Goeders17]. Debug of extra-functional properties is required and there exist tools that allow it for the performance in reconfigurable logic scenarios [Shannon15]. In ASIC processors, there are various example of smart monitoring solutions: AMD Lightweight profiling feature [AMD10] and Intel Processor Tracing [Intel] are composed of hardware facilities (such as hardware performance counters) and software able to use information acquired at low-level. They are supported by necessary libraries to interpret collected data. In Leon3 based scenarios, Nam et al. [Ho14] proposed a performance monitoring unit integrated with perf_event API. Xilinx supports system level profiling using SDSoC environment [Xilinx], using performance counters in ARM Cortex A9 and performance monitoring units in programmable logic side. Moreover, it offers profiling solutions also in MicroBlaze soft-processor [Xilinx15]. In order to define a custom profiling system for embedded applications, a solution based on specific metric definition and implementation of necessary parts has been considered [Valente15]. This technique conducted to a definition of a library of elements, to be used to compose a hardware profiling system tailored for specific applications [Valente16].

3.4.3 Extension within FitOptiVis

The future way to perform monitoring solutions is based on hardware/software collaborative approaches where hardware implements the profiling action (profiling hooks and mechanisms) and software implements the functionalities that use collected data (algorithms, complex heuristics, etc.). The proposed extension within FitOptiVis focuses on different points.

The first is the possibility to tailor and customize the monitoring system for the system under exam, considering also accelerators: the customization will depend on when to use the monitoring action (i.e. during the lifecycle to characterize the system or during development phases to support the designer). It will depend on platform selected for the system (ASIC, reconfigurable logic). And it will depend on non-functional properties of the system itself (how much overhead can be introduced with the monitoring system, if a real-time profiling action is requested, etc.).

The second is the development of a framework able to support the designer on the selection of profiling solution.

3.4.4 Prospective adoption FitOptiVis UC

The water supply (Use Case 1) will be the FitOptiVis UC adopting this component.

3.4.5 Related documents

[Goeders17] J. Goeders and S. J. E. Wilton, "Signal-Tracing Techniques for In-System FPGA Debugging of High-Level Synthesis Circuits," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 1, pp. 83-96, Jan. 2017

[Shannon15] L. Shannon, E. Matthews, N. Doyle, A. Fedorova, “Performance monitoring for multicore embedded computing systems on FPGAs,” 2nd Int. Workshop FPGAs for Software Programmers, London, 2015

[AMD10] AMD (2010, August), “Lightweight profiling specification”. [Online]

[Intel] Intel, “Intel 64 and IA-32 Architectures Developer's Manual”. [Online]

[Ho14] N. Ho, P. Kaufmann, M. Platzner "A hardware/software infrastructure for performance monitoring on LEON3 multicore platforms," in 24th Int. Conf. Field Programmable Logic and Applicat., Munich, 2014, pp.1-4
[Xilinx] Xilinx, SDSoC Development Environment, [Website]. [Online]
[Xilinx15] Xilinx (2015, November 18). MicroBlaze processor reference guide (v 2015.4). [Online].
[Valente15] G. Valente et al., "Hardware Performance Sniffers for Embedded Systems Profiling," in 12th Workshop Intelligent Solutions Embedded Syst., 2015, pp. 29-34.
[Valente16] G. Valente et al., "A flexible profiling sub-system for reconfigurable logic architectures", in Proceedings of PDP 2016 (in press), February 2016

3.5 Image collection interface [TASE]

3.5.1 Short description

The Image Collection Interface will be a hardware IP developed in FitOptiVis in order to obtain images from the CMOS sensor CMV12000. The sensor CMV12000, which is a high sensitivity pipelined global shutter CMOS image sensor with a resolution of 4096x3072 pixels, has different possible configurations, which will be controlled by the Image Collection Interface.

Input:

- 64bit parallel LVDS at 300Mbps. 8-bit/10-bit/12-bit data words from sensor

Output:

- Exposure and Readout interface.
- Output Data Format configuration interface (bits per pixel and data rate)
- Image data

Target Technology:

- Main target: Kintex UltraScale KU060.
- Side target: UltraScale and UltraScale+ FPGA's families

3.5.1.1 Qualities, tunable set-points and available design-time and run-time parameters

- Image output of 4096x3072 pixels with
  - 12 bits per pixel,
  - 10 bits per pixel,
  - Or 8 bits per pixel.
- Selection of data rate with the clock delivered to the sensor:
  - 300MHz ➔300fps
  - 200MHZ➔200fps
  - 100MHz➔100 fps
  - 50MHz➔50fps
  - 10MHz➔10fps
  - 1MHz➔1fps
- The IP will also be capable of tuning the exposure of the obtained images.

3.5.1.2 Design-time support

Design time support tools and methods: Xilinx's Vivado (2018.1 or later) commercial tool used in order to develop the IP and to instantiate it.

3.5.1.3 Run-time support

Run Time Adaptivity:
The IP can change the configuration of the image collection on run-time being able to change the quality of the obtained images.

Programmability:
- Programming APIs:
    - Hardware description:
        - Technology: VHSIC Hardware Description Language (VHDL)
        - Version: IEEE 1076-2008
- Reconfiguration:
    - The IP could be reconfigured in case of malfunction due to failures induced by radiation conditions present in space and affecting SRAM Configuration Memory of the FPGA.

Monitoring:
- Selected Configuration.
- Watchdog for the camera.
- Power Measurement: The energy consumption of this block can't be determined but the changes on the power that the different configurations produce on the aggregate will be tracked.

3.5.2 Related state of the art

CCD sensors, which are much more expensive to manufacture than CMOS sensor dominate the space industry in earth observation, astronomy and exploration. This is because is a mature technology and with excellent electro-optical performance.
On the other hand, CMOS sensors are considered the future by the European Space Agency that is investing in order to develop an European CMOS foundry industry for development of high performance sensors. This is the reason behind the development of this IP block for controlling a state-of-the-art sensor developed by AMS (Austria Mikro Systeme).

3.5.3 Extension within FitOptiVis

TRL@M0:
TRL1
FitOptiVis Extensions:
Full development of the module on several Xilinx architectures
TRL@M36:
TRL6

3.5.4 Prospective adoption FitOptiVis UC

The IP will be used in the Use Case 10 of this project.

3.5.5 Related documents

- CMV12000 12Mp High Speed Machine Vision Global Shutter CMOS Image Sensor.
- CMOS Image Sensor developments supported by the European Space Agency – 2018 EIROForum Topical Workshop: CMOS Sensors. Kyriaki Minoglou

3.6 Space image processing [TASE]

3.6.1 Short description

Image processing block capable of extracting feature points through different algorithms and match them to a target image for object recognition. Each algorithm will have different performances in quality and power consumption and the selection of the one used in each moment will be determined by environmental circumstances.

Input:

Image data of 4096x3072 pixels with

- 12 bits per pixel,
- 10 bits per pixel,
- Or 8 bits per pixel.

Output:

- Feature points data.

Target Technology:

- Main target: Kintex UltraScale KU060.
- Side target: UltraScale and UltraScale+ FPGA's families

3.6.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Different feature extraction algorithms will be selected based on the environment, battery of the spacecraft and desired objective of each step of the mission, the possible algorithms will be:

- SIFT
- ORB
- Harris Corner Detector

3.6.1.2 Design-time support

Design time support tools and methods:

- Xilinx's Vivado (2018.1 or later) commercial tool used in order to develop the IP and to instantiate it.
- Matlab for the development of the architectural model.

3.6.1.3 Run-time support

Run Time Adaptivity:

Change of performance based on environmental conditions.

The IP could be reconfigured in case of malfunction due to failures induced by radiation conditions present in space and affecting SRAM Configuration Memory of the FPGA.

Programmability:

Programming APIs:

- Hardware description:
  - Technology: VHSIC Hardware Description Language (VHDL)
  - Version: IEEE 1076-2008

Monitoring:

- Power Measurement: The energy consumption of this block cannot be determined, but the changes on the power that the different configurations produce on the aggregate will be tracked.

3.6.2 Related state of the art

The algorithm that has been more used in the space industry for feature extraction is the SIFT algorithm. This algorithm is very robust but also has a very high computational cost. In 2011 ORB was presented with a similar performance to SIFT but much less computational cost.

3.6.3 Extension within FitOptiVis

TRL@M0:
TRL1
Extension within FitOptiVis:
Full development of the component on the project
TRL@M36
TRL4

3.6.4 Prospective adoption FitOptiVis UC

The IP will be used in the Use Case 10 of this project.

3.6.5 Related documents

N/A

3.7 Image transmission interface [TASE]

3.7.1 Short description

An *ad hoc* IP will be developed in order to transmit the data produced by the camera CMV12000 and the feature points produced by the Space Image Processing component described on 3.6.
Input:

- Compressed 4096x3072 pixels image.
- Feature points and descriptors extracted by the Space Image Processing component.

Output

- Images and feature points and descriptors packed to be sent by UDP.

3.7.1.1 Qualities, tunable set-points and available design-time and run-time parameters

This block will not have any tunable set-point nor run-time parameters.

3.7.1.2 Design-time support

Design time support tools and methods:

- Xilinx's Vivado (2018.1 or later) commercial tool used in order to develop the IP and to instantiate it.

3.7.1.3 Run-time support

Run Time Adaptivity:
No runtime adaptivity

Programmability:
- Programming APIs:
    - Hardware description:
        - Technology: VHSIC Hardware Description Language (VHDL)
        - Version: IEEE 1076-2008
- Reconfiguration:
    - The IP could be reconfigured in case of malfunction due to failures induced by radiation conditions present in space and affecting SRAM Configuration Memory of the FPGA.

Monitoring:
- Power Measurement: The energy consumption of this block cannot be determined, but the changes on the power that the different configurations produce on the aggregate will be tracked.

3.7.2 Related state of the art

Ethernet is widespread in earth applications but is not as common in space where the first PHY chips are being actually developed. The adoption of ethernet communications in space will allow the transmission of data rates which will be much higher than the ones present nowadays (Up to 2.5 Gbps with SpaceFiber).

3.7.3 Extension within FitOptiVis

TRL@M0:
TRL1
Extension within FitOptiVis:
Full development of the component on the project
TRL@M36:
TRL4

3.7.4 Prospective adoption FitOptiVis UC

The IP will be used in the Use Case 10 of this project.

3.7.5 Related documents

N/A

3.8 License plate detector [CAMEA]

3.8.1 Short description

License plate detector is designed based on state-of-the-art object detection algorithm trained particularly for license plate. It achieves a real-time performance on multiple scales, without the requirement of external DRAM memory.
Target:

It is fully accelerated and synthesized in form of FPGA firmware primarily for hybrid platform Xilinx Zynq. This component can be then used in smart cameras equipped by compatible hardware, e.g. the smart camera developed by CAMEA.

It receives images from camera sensor (possibly composed HDR), processes every frame and tries to find LP in it. The results are then passed to post processing engine (usually outside the FPGA but inside Zynq platform – running in ARM).
Composability:
Parts of detector processing chain can be eventually separated or re-implemented for various platforms.

3.8.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:
License Plate detector component has following qualities:

- Detection success rate – the main quality of detector is detection success rate of individual license plates.
- Low light performance – license plate detection has to work in most of weather and lightning condition (optimally without IR illumination)
- Power consumption – has to be kept low allowing battery operation and operation in sealed box (low heat dissipation)

Set-Points:

- LP size limit (in pixels) or reduced frame rate for lower system load (weaker platform) or lower power consumption.

Available model:
N/A

3.8.1.2 Design-time support

The FPGA firmware is designed using current commercially available tools (mostly VHDL oriented) such as Vivado.

3.8.1.3 Run-time support

Monitoring:
Profiling of units in FPGA and software, displaying in special tool and looking for hiccups and conflicts

- FPS counter
- load/power consumption estimator

RT adaptivity:
none
Programmability:
FPGA firmware with programmable units via registers (LP size, searching window, …)
Programming API:
IP core without proprietary API (only AXI bus access)

3.8.2 Related state of the art

Object detection in embedded systems is an important task that many applications of computer vision and scene analysis benefit from. Industrial quality control systems address various markers, traffic monitoring uses detection of cars and license plates, biometric systems detect faces and facial features, driver assistance systems detect cars and pedestrians. The detection is especially important in applications that directly rely on it, such as recognition or tracking, and in these applications, the speed, accuracy, power consumption, and/or robustness of detection matters most. Our component focuses on boosted detectors, which analyse sub-windows of an input image by a classifier composed from weak classifiers based on simple image features such as Haar [1] or Local Binary Patterns (LBP) [2]. Multi-scale detection is solved by scaling and processing of the input image in multiple resolutions – image pyramid. Embedded object detectors are often implemented directly in software using libraries such as OpenCV [3]. While this approach is easy and straightforward, it often is quite slow, as detection is computationally demanding task and embedded processors tend to be simpler and slower than desktop CPUs. Another approach is to implement a custom detection algorithm exploiting various acceleration resources of the target platform – CPU [4], GPU [5] or Field Programmable Gate Array (FPGA) [2], [6]–[11] units. This is advantageous in many areas where the deployment of standard PC-based or embedded software solution is not possible, e.g. because of resource consumption, physical dimensions, industrial or military conditions, etc. The object detection in embedded devices typically belongs to one of the three detection method categories. 1/ AdaBoostbased detectors – cascades of boosted classifiers [1] or soft cascades [12]. They typically use Haar image features [6], [9], [10], [13], or LBP [2]. 2/ Support Vector Machines (SVM) with Histograms of Oriented Gradient features (HOG) [7], [8], [14]–[16]; and 3/ Other methods implementing detection with background subtraction [17], keypoints [18], neural networks [19], or custom detection algorithms [20].

3.8.3 Extension within FitOptiVis

TRL@M0:
Within FitOptiVis project, license plate detector component started in form of proof of concept (TRL3/4).
TRL@M36:
At the end of the project, thanks to the evolution of detector - cutting edge algorithms, HDR extension, power aware redesign, the TRL6 is expected. Test and validation in simulated and optionally real environment will be done.

3.8.4 Prospective adoption FitOptiVis UC

The license plate detector component will be adopted in Road Traffic Surveillance use case within the FitOptiVis project.

3.8.5 Related documents

[1] P. Viola and M. J. Jones, “Robust real-time face detection,” IJCV, 2004.
[2] P. Zemcik, R. Juranek, P. Musil, M. Musil, and M. Hradis, “High performance architecture for object detection in streamed videos,” in FPL, 2013.
[3] OpenCV library. [Online]. Available: http://opencv.org
[4] R. Juranek, A. Herout, and P. Zemcik, “Impelementing Local Binary Patterns with SIMD instructions of CPU,” in WSCG, 2010.
[5] A. Herout, R. Joˇsth, R. Jur´anek, J. Havel, M. Hradis, and P. Zemcik, “Real-time object detection on cuda,” JRTIP, 2011.
[6] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, “Fpga-based face detection system using haar classifiers,” in FPGA, 2009.
[7] C. Kyrkou, C. Ttofis, and T. Theocharides, “Fpga-accelerated object detection using edge information,” FPL, 2011.
[8] S. Martelli, D. Tosato, and M. C. V. Murino, “Fast fpga-based architecture

for pedestrian detection based on covariance matrices," in ICIP, 2011.
[9] C. Huang and F. Vahid, "Scalable object detection accelerators on fpgas using custom design space exploration," SASP, 2011.
[10] C. Kyrkou and T. Theocharides, "A flexible parallel hardware architecture for adaboost-based real-time object detection," in VLSI, 2011.
[11] F. Kadlcek and O. Fucik, "Automatic synthesis of small adaboost classifier in fpga," in DDECS, 2013.
[12] P. Doll´ar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," PAMI, 2014.
[13] J. Granat, A. Herout, M. Hradis, and P. Zemcik, "Hardware acceleration of adaboost classifier," in Workshop on MLMI, 2007, pp. 1–12.
[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in CVPR, 2005.
[15] C. Kyrkou, C.-S. Bouganis, T. Theocharides, and M. M. Polycarpou, "Embedded hardware-efficient real-time classification with cascade support vector maschines," VLSI, vol. 27, 2016.
[16] T. Kryjak, M. Komorkiewicz, and M. Gorgon, "Fpga implementation of real-time head-shoulder detection using local binary patterns, svm and foreground object detection," in DASIP, 2012.
[17] C. H. Yeh, C. Y. Lin, K. Muchtar, H. E. Lai, and M. T. Sun, "Three pronged compensation and hysteresis thresholding for moving object detection in real-time video surveillance," IEEE TIE, 2017.
[18] D. Bouris, A. Nikitakis, and I. Papaefstathiou, "Fast and efficient fpga based feature detection employing the surf algorithm," in FCCM, 2010.
[19] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello, "Hardware accelerated convolutional neural networks for synthetic vision systems," in ISCAS, 2010.
[20] Y. D. Sato and Y. Kuriya, "Multi-scale elastic graph matching for face detection," EURASIP Journal on Advances in Signal Processing, 2013.

3.9 MDC-generated coarse grained reconfigurable HW accelerators [UNISS, UNICA]

In heterogeneous platforms, the programmable logic is tightly coupled with on board microprocessors running embedded operating system (OS) [Ahmad16]. This makes FPGA devices suitable to run highly demanding applications at reasonable power cost, while guaranteeing high performance. In such architectures, the embedded OS manages HW tasks in a classical SW like manner [Bergmann13], easing the access to the configurable logic and making the HW-SW approach more appealing.

3.9.1 Short description

Purpose and Mission

MDC tool, as described in D3.1, provides coarse-grain reconfiguration (CGR) support, generating application-specific reconfigurable substrates and packing them within accelerators ready to be used in Xilinx FPGA platforms.

Principal key features and properties

CGR is applied at the word/data level, rather than at bit level. The accelerator can switch among a set of pre-defined functionalities or surf among different available profiles of a given application. MDC-compliant systems follow an application to architecture approach: the CGR substrate is shaped according to a set of desired applications, resulting in an application specific platform capable of achieving strong execution efficiency, but limited to a fixed set of applications.

Processed inputs and produced outputs

Application-Specific (user defined)

Connectivity and communication

MDC tool offers different processor to co-processor communication infrastructures. The user can select the type of coupling among memory-mapped and stream-based, and the possibility of adopting DMA to manage data transfers. Proper APIs are automatically built upon the setting preventing the user from the burden of low-level infrastructure details management.

Composability and granularity of the component

MDC tool exploits horizontal composability properties to build accelerators. It adopts a high-level synchronous dataflow representation of the kernels to be accelerated; starting from them, data-path merging techniques are used to create the reconfigurable substrate. Crossbar switches (SBoxes) are inserted at the crossroads of different paths, and data are driven according to the specified configuration patterns.

3.9.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:

MDC-compliant CGR accelerators are particularly suitable to offer run-time reconfiguration among different system execution profiles. Implementations are application-specific.

Approximate HEVC interpolation filters - Energy versus Quality Trade-Off

High Efficiency Video Coding (HEVC) exploits the temporal redundancy in a video; in a sequence of frames, it codes only the reference frame, while in the following only the movement of the objects is coded. If the movement of the objects involves fractional values of pixels, a block is predicted by interpolating its reference block. Exploiting functional approximate computing it is possible to save up to 28% of energy, with a small degradation of the video quality, obtained by reducing the number of taps in the filter [Nogues16]. In HW, when the quality can be downgraded, part of the computational stages in a CGR infrastructure can be excluded from the computation by properly driving the switching elements, as shown in the figure below.

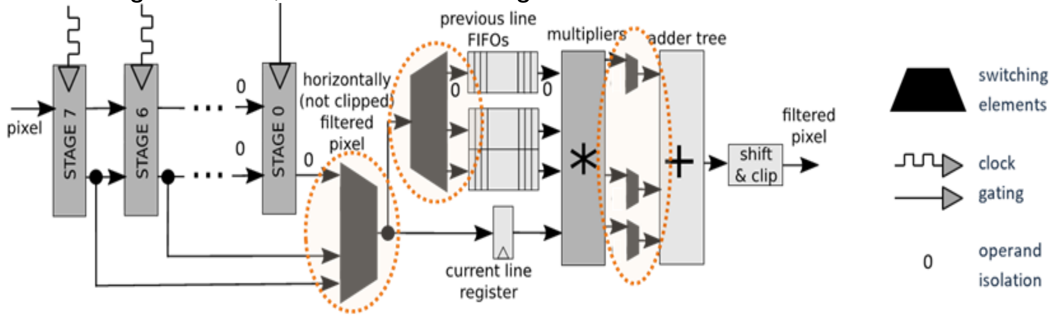

Figure 2: Approximate HEVC interpolator.

MDC-compliant CGR filter guarantees different trade-off in terms of energy versus quality, by changing at runtime the number of taps involved in the computation [Palumbo17, Sau17]. The figure below shows a clear trade-off between energy consumption and image quality. This variability can be exploited on a smart device equipped with a proximity sensor: when the user is close to the device, quality should be high, but when he/she is far, and cannot distinguish details, the quality can be lowered.

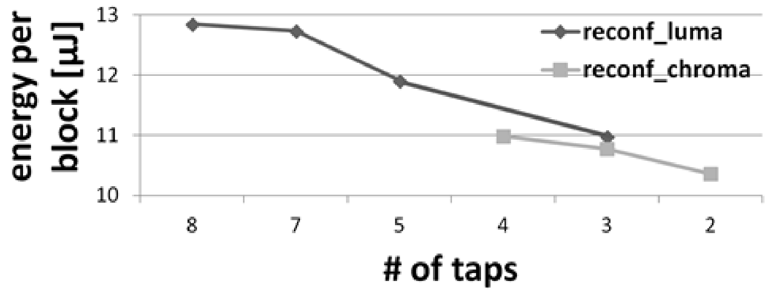

Figure 3: Quality versus Energy Trade-Off in the HW Approximate HEVC interpolator.

Set-Point(s) and Parameters:
Surfing among configurations in MDC-compliant CGR accelerators is made possible by configuring the accelerator through proper APIs, selecting the network identifier *ID*. The table below summarizes the available trade-off for the HEVC interpolator.

Design	Working points	Trade-off	Target Technology
HEVC	Number of taps involved in the computation	Energy vs Quality **High quality** - 8 taps luma and 4 taps chroma. **Low Energy** - 3 taps luma and 2 taps chroma.	Artix-7 FPGA

3.9.1.2 Design-time support

1. Design and customization of MDC-compliant CGR accelerators is done using the MDC tool (details are provided in D3.1) [Palumbo17b, Sau15].
2. The outcome of the MDC tool, intended as CGR stand-alone datapath, is a target independent HDL description. However, synthesis and deployment of MDC-compliant CGR systems is target dependent, since vendor specific flows have to be used for synthesis
3. As explained in D3.1, MDC allows the automated deployment of ready-to-be-used accelerators over Xilinx FPGA platforms. The Vivado design environment is used in this specific case.

3.9.1.3 Run-time support

In MDC-compliant accelerators, the cost of reconfiguration is minimized, in terms of both time (word-based reconfiguration is performed in one clock cycle) and power (no need of downloading a new big bitstream through dedicated channels).
MDC tool handles programmability, keeping trace of the SBoxes configuration patterns while merging the input specifications. Those patterns are saved into dedicated Look-Up Tables. A configuration module in the HDL properly sets the SBoxes selectors according to the network ID.

Run-time adaptivity support:
In CGR systems, different paths of data can be activated at run-time, excluding the un-necessary resources from the computation. In MDC-compliant accelerators, different kernels/profiles have different IDs, which activates different paths of data.

Monitoring:
The monitoring approach we are investigating is compatible with the Performance Application Programming Interface (PAPI) [PAPI19]. To access the monitors a PAPI-MDC Component, compatible with the already existing PAPI components, has been defined in [Fanni19]. This component is automatically configured, according to the number of events to be monitored, using an XML file.

Programmability and Programming API
The MDC design suite offers advanced features related to system integration. Each CGR HW accelerator can be easily programmed by means of simple APIs that expose to the users C functions to run accelerator with a certain set point, hiding the details of the accelerator, like coupling with the processor or DMA usage.

Dealing with monitoring features, HW monitors are accessed through the configuration registers of the accelerator. To facilitate the reading and make it transparent for the user, the PAPI calls [PAPI19] are used.

3.9.2 Related state of the art

Modern complex applications are required to be flexible and power-aware, especially in the Cyber Physical Systems (CPS) and Internet of Things (IoT) contexts. Despite Application Specific Integrated Circuits (ASICs) can better fit with the computing and power efficiency demand, they are not flexible enough to meet reactiveness and dynamism requested in the CPS context. The FPGA-based platforms are meant to provide efficiency close to the ASICs, while offering also programmability [Trimberger15]. CGR approaches as those used by MDC-compliant accelerators already demonstrated to be suitable to support adaptivity needs in CPS [Palumbo17, Sau17] contexts, providing requested flexibility and dynamism. At the same time, their deployment and usage are made friendly by a complete flow for automatic customization and programmability support. With respect to this context, FitOptiVis studies related to CGR accelerators is focusing on different aspects.

- Approximate Computing: It introduces quality as a new design metric to be traded-off with power consumption, performance or resource occupancy, expanding the design space. Combining approximate computing with CGR seem to be a viable and suitable solution to address runtime trade-off management, but there is a *lack of specialized supporting tools for approximate computing*.
- Monitoring: One of the most adopted SW approaches for performance monitoring is based on the Performance Monitoring Counters (PMCs) existing on modern CPUs. PAPI provides a unified method to access these PMCs [PAPI19]. Madroñal et al. [Madroñal18] presented PAPIFY, a tool that provides a lightweight monitoring infrastructure by means of an event library aimed at generalizing PAPI for embedded heterogeneous architectures. A first tentative to exploit PAPI to monitor HW reconfigurable systems has been done by Suriano et al. [Suriano18], which present a custom approach to read monitors of a HW architecture based on Dynamic and Partial Reconfiguration. The monitoring approach to be adopted by MDC-compliant accelerators goes in the same direction. PAPI-based accesses are meant to be generic enough to be easily extended to different types of application specific HW substrates, not necessarily CGR based ones.
- Runtime modelling: Since CGR acceleration has to guarantee fast reconfiguration, the correct execution associated to the different configurations of the system, a runtime strategy to monitor and estimate of the performance metrics is needed. Methods based on abstractions reduce the explosion of complexity, such as problems of scalability. Among these methods, modelling of architectures (MoA)

can be used in order to achieve energy and latency estimation, as similarly proposed in [Pelcat17]. *MoAs for different estimation of qualities have to be derived, to be used at run-time to drive the choices upon trade-off to be taken.*

3.9.3 Extension within FitOptiVis

Defining TRL level for MDC-compliant accelerators it is not feasible. These accelerators are the outcome of an academic tool, with TRL3-4. What we propose is more a methodology to define flexible and re-programmable accelerators rather than one specific single component.
<u>TRL@M0</u>:
N/A
<u>FitOptiVis Extensions:</u>
N/A
<u>TRL@M36:</u>
N/A

In future releases of this document we expect to describe MDC-generated accelerators specifically conceived for FitOptiVis use-cases. In those cases, it will be possible to provide quantitative evaluation of each specific component.

3.9.4 Prospective adoption

MDC-compliant accelerators are meant to be used within the water supply use case. Starting from a high-level dataflow description of the functionalities to be executed, MDC tool generates ad-hoc accelerators with multiple, configurations (if more than one functionality shall be executed on the same accelerator) or working points (if given a certain functionality more profiles, with different trade-offs, shall be enables).
We expect to deliver at M24 the fist MDC-compliant accelerators for the water supply use case, however the exploitation of this type of acceleration/reconfiguration is generic enough to be used in other application scenarios.

3.9.5 Related documents

[Sau15] C. Sau et al., “Automated Design Flow for Multi-Functional Dataflow-Based Platforms”, Springer Journal of Signal Processing Systems, vol. 85, issue 1, pp. 143-165, 2015, 10.1007/s11265-015-1026-0
[Palumbo17] F. Palumbo et al., “Challenging CPS Trade-off Adaptivity with Coarse-Grained Reconfiguration”, 2017 Conference on Applications in Electronics Pervading Industry, Environment and Society, 10.1007/978-3-319-93082-4_8
[Banik16] S. Banik et al., “Adaptable AES implementation with power-gating support”, 2016Conference on Computing Frontiers, 10.1145/2903150.2903488
[Sau17] C. Sau et al., "Challenging the Best HEVC Fractional Pixel FPGA Interpolators With Reconfigurable and Multi-frequency Approximate Computing" IEEE Embedded Systems Letters, vol. 9, no. 3, pp. 65--68, 2017, 10.1109/LES.2017.2703585
[Palumbo17b] F. Palumbo et al., “Power-Awarness in Coarse-Grained Reconfigurable Multi-Functional Architectures: a Dataflow Based Strategy”, Springer Journal of Signal Processing Systems, vol. 87, issue 1, pp. 81--106, 2017, 10.1007/s11265-016-1106-9
[Sau15] C. Sau et al., "Reconfigurable coprocessors synthesis in the MPEG-RVC domain," 2015 Conference on ReConFigurable Computing and FPGAs, 10.1109/ReConFig.2015.7393351
[Fanni19] D. Madroñal and T. Fanni, “Run-time performance monitoring of hardware accelerators”, 2019 Conference on Computing Frontiers, 10.1145/3310273.3323423

[Ahmad16] S. Ahmad, V. Boppana, I. Ganusov, V. Kathail, V. Rajagopalan, and R. Wittig, “A 16-nm Multiprocessing System-on-Chip Field-Programmable Gate Array Platform”, IEEE Micro, vol. 36, n. 2, pp. 48--62, 2016, 10.1109/MM.2016.18
[Bergmann13] N.W. Bergmann, S. Shukla, and J. Becker, “QUKU: a dual-layer reconfigurable architecture”, ACM Transactions on Embedded Computing Systems, vol. 12, n. 1s, pp. 63:1--63:26, 2013, 10.1145/2435227.2435259
[Nogues16] E. Nogues, D. Menard, and M. Pelcat, "Algorithmic-Level Approximate Computing Applied to Energy Efficient HEVC Decoding," IEEE Transactions on Emerging Topics in Computing, vol. 7, n. 1, pp. 5--17, 2019, 10.1109/TETC.2016.2593644
[PAPI19] Performance API. Last access: March 2019. http://icl.utk.edu/papi/
[Trimberger15] S. Trimberger, “Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology”, Proceedings of the IEEE, vol. 102, n. 3, pp. 318--331, 2015, 10.1109/JPROC.2015.2392104
[Madroñal18] D. Madroñal, A. Morvan, R. Lazcano, R. Salvador, K. Desnos, E. Juarez, and C. Sanz, “Automatic Instrumentation of Dataflow Applications Using PAPI”, 2018 ACM International Conference on Computing Frontiers, pp. 232--235, 10.1145/3203217.3209886
[Suriano18] L. Suriano, D. Madroñal, A. Rodriguez, E. Juarez, C. Sanz, and E. de la Torre, “A Unified Hardware/Software Monitoring Method for Reconfigurable Computing Architectures Using PAPI” In 13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip, 2018, 10.1109/ReCoSoC.2018.8449389
[Pelcat17] M. Pelcat, A. Mercat, K. Desnos, L. Maggiani, Y. Liu, J. Heulot, J. F. Nezan, W. Hamidouche, D. Ménard, and S. S. Bhattacharyya, “Reproducible evaluation of system efficiency with a model of architecture: From theory to practice”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 37, n. 10, pp. 2050--2063, 2017, 10.1109/TCAD.2017.2774822

See also:

1. http://sites.unica.it/rpct/
2. https://www.youtube.com/watch?v=_cyYFJCDR3U&list=PLql1YxTzHalZztJPu7wn0uzAYbr81QTpH

3.10 Low energy dynamic voltage and frequency scaling enabled CPU with accelerators [UTU]

3.10.1 Short description

This component will be a full custom IC. Actually three (3) versions are planned over the whole project duration. The first version is currently in design phase, and will be submitted for processing in the fall, in the next suitable batch. The CPUs will be based on RISC-V architecture, guaranteeing good SW support and compilers and so on. The idea is to test DVFS components in the first version, make them autonomous in the second version, and optimize the system in the final version. Additionally application specific accelerators will be implemented to versions 2 and 3.

Type:
HW
Target:
Full Custom IC
Input:
Any data, but in FitOptiVis context we expect video stream (2D and/or 3D)
Output:
Any data, but in FitOptiVis context we expect 3D data and/or content analysis data

3.10.1.1 Qualities, tunable set-points and available design-time and run-time parameters

The DVFS system allows tuning energy consumption based on the load experienced by the CPU. This tuning will be done during run-time, and later versions will have autonomous tuning capabilities, while the first one will be software controlled.
One of the accelerators planned is a limited precision convolutional filtering engine. It will provide design-time parameters for desired precision and also fixed kernels for filter weights.
Other accelerators will be added, once our partners reach a state of maturity, so that we can implement functions that provide benefits to their use case(s). Such co-operation is planned with Nokia and TUT. Others are possible, but not agreed upon yet.

3.10.1.2 Design-time support

We will provide guidelines and manuals on how to tune the limited precision accelerator core. Also at least a technical note will be provided regarding the DVFS capabilities in the CPU.

3.10.1.3 Run-time support

Software support for using the limited precision core and DVFS capabilities will be provided.

3.10.2 Related state of the art

Some DVFS capable implementations of the RISC-V core already exist. We plan to extend the capabilities, increase automation and include accelerator cores to the DVFS scheme. Also, we plan to investigate including memories to the DVFS domain. Typically memories operate at such a small margin, that they do not function correctly under other conditions than the nominal. If tests show promising results, full custom memory banks that can use varying voltage and frequency will be included in later versions.

3.10.3 Extension within FitOptiVis

<u>Expected Extensions:</u>
Within FitOptiVis, there are plans to improve DVFS capable CPU at least in the following aspects:

- Optimize DCDC based voltage scaling
- Optimize clock frequency scaling
- Automate proper DVFS operating point selection (In HW)
- Add support for accelerators
- If feasible, add DVFS capable memory structures

3.10.4 Prospective adoption FitOptiVis UC

Usage in FitOptiVis use cases: UC2
Fields of application – Improved execution engine, with very low energy footprint.

3.10.5 Related documents

RISC-V documentation is freely available, as it is an open source core.

3.11 Multistream video image scaler and compositor [PHL]

The multistream video image scaler and compositor component is the hardware basis for use case 6: Multi source streaming composition.

3.11.1 Short description

In the Multi source streaming composition use case several Ethernet based video streams are combined into one or more large screen images in a medical Xray application. Although the final goal of this use case is to avoid image scaling as much as possible, it must be supported for legacy image sources (those that cannot adapt their output to the requested format). But in the transitional period when a viewport-layout is changed, the video sources cannot react instantaneously so they still output the 'old' video format for some period of time so scaling is also needed during layout changes. Since the use case mandates very short latency for selectable input streams, the component needs to be optimized for this purpose and will support synchronization between the output device (display) and an input source. As such, this component consists of the following sub-components:

- Ethernet inputs and router
- A set of noise compensated LancZos-2 downscalers shared by multiple input streams
- Image memory
- Output scaler/compositor(s)
- Ethernet output

Input**:**

- Multiple 10G optical Ethernet interfaces.
- Overlay image stream(s) generated by the controlling processor subsystem

Output:

- Multiple 10G optical Ethernet interfaces
- May be shared with the input interfaces

Target Technology:

- Main target: Xilinx Ultrascale+ MPSOC ZCU7EV
- Side target: other members of the Utrascale+ MPSOC family and the upcoming Xilinx Versal FPGA's

3.11.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Image input comes in 2 flavours:

- Fixed format (legacy devices)
    - 640x480
    - 1024x768
    - 1280x720 (720p)
    - 1920x1080 (1080p, FHD)
- Variable formats with dynamic adjustment
    - Any format between 640x480 till 1920x1080
- Image output:
    - 3480x2160 (4K, 60 Hz)
    - 2560x1440 (1440p, 60 Hz)

3.11.1.2 Design-time support

We will provide guidelines and manuals on how to use this core and the accompanying driver that runs on the ARM-R5 cores of the FGA device.

3.11.1.3 Run-time support

There will be a software driver that does all the low level runtime calculations that runs on the ARM-R5 cores of the FPGA including a communication interface library.

3.11.2 Related state of the art

Although Lanczos image scalers already exist for a long time, they are usually implemented on general purpose CPUs or GPUs using standard buffering techniques. This means that power consumption is relatively high and due to the generic buffering the latency is long (1 - 2 complete frame times). Especially for downscaling, the Lanczos algorithm leads to large filter kernels and thus a high computational load so for FPGA implementations it is usually replaced by a small bicubic interpolation kernel leading to less image quality.

3.11.3 Extension within FitOptiVis

TRL@M1:
TRL1

Expected Extensions:
Within FitOptiVis for this component, we will develop a set of FPGA-resource optimized scalers that can do true Lanczos interpolation also for downscaling including the required buffering and synchronization for low latency (< 3 msec) processing.
The up-scalers will support image composition onto 1 or more large screens.

TRL@M1:
TRL6-TRL7

3.11.4 Prospective adoption FitOptiVis UC

Usage in FitOptiVis use cases: UC6
Fields of application – Image scaling and compositing

3.11.5 Related documents

- Description of the Lanczos filtering and resampling: https://en.wikipedia.org/wiki/Lanczos_resampling
- Xilinx Ultrascale+ MPSoc documentation: https://www.xilinx.com/products/silicon-devices/soc/zynq-ultrascale-mpsoc.html
- 10G optical ethernet standards

3.12 Efficient magnetic field generators [PHL, FUT]

3.12.1 Short description

The purpose of these components is to convert an electric current into magnetic field of a specific target shape. The efficiency of these components can be expressed in terms of stored energy and dissipation for a defined reference characteristic of the generated field.

The input is therefore electric current (in Amperes) and the output is magnetic field (in Tesla).
For FitOptiVis, two types of magnetic field generators are considered:

- The gradient coil. The target spatial field distribution is a spatially linear increasing magnetic field in a low (< 6kHz) frequency range. The power consumption is in the order of tens of kW. The rapid switching of the magnetic field of a gradient coil is capable of causing stimulation of the peripheral nerves or even cardiac stimulation, as well as the source of high levels of acoustic noise (up to 130 dBA).
- The body coil. The target spatial field distribution is a uniform field at a virtually fixed frequency of tens of MHz. This field is capable of causing significant tissue heating.

3.12.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities: N/A, the current to magnetic field ratio is fixed.
Set-Point(s) and Parameters:
The set-points are related to the physiological stress that can be caused by driving the component. These physiological stresses (tissue heating, peripheral nerve stimulation and acoustic noise) are described by a model:

- The input is the current.
- The output is a measure of the physiological stress. E.g., for acoustic noise this is the Sound Pressure Level.
- The models also have parameters.

Each model is evaluated design time, that is, at the time a scan protocol is defined by the user. At that moment, the input currents that will be driving the components during the execution of the scan are determined. Once these currents are delivered to the components (the scan is 'executed'), a very limited number of run-time changes are possible because of the extreme sensitivity of the spin state to the magnetic fields.

3.12.1.2 Design-time support

In-house developed design tools, simulation packages like ANSYS, development tools like Matlab.

3.12.1.3 Run-time support

Not applicable.

3.12.2 Related state of the art

In the state-of-the-art no MRI product offers the user the ability to control the energy consumption explicitly.

3.12.3 Extension within FitOptiVis

Expected Extensions:
The energy consumption and acoustic noise shall be modeled with an accuracy of at least 10% and used in a multi-objective optimization scheme.

3.12.4 Prospective adoption

The exploitation of this component will be in a Philips MRI product. The developed HW and SW components have specific proprietary interfaces. The MRI market is characterized by a few players: Canon (Japan), GE Healthcare (US), Hitachi (Japan), Philips (the Netherlands), Samsung (Korea), Siemens Healthineers (Germany) and United Imaging (China).

Energy consumption of an MRI system is not yet an important part of the product data. COCIR, the European Trade Association representing the medical imaging, radiotherapy, health ICT and electromedical industries, has defined a standard measurement method to characterize the energy consumption of an MRI system. The data is intended for potential customers to compare products. At the time being, publication of this information is still voluntary. Philips references COCIR when publishing the average power consumption during scanning in the Philips MRI product data.
In view of the successful sustainability strategy of Philips, an MRI product that allows the user to control the energy consumption explicitly will fulfill market need.

3.12.5 Related documents

Only internal documents were used for the description of this component.

4 Software components

This section presents FitOptiVis SW IPs intended as an algorithm, described using a SW specification language (e.g. C, C++, MatLab, etc.), handling a specific processing issue.

4.1 Person tracking software [UGR]

4.1.1 Short description

The person tracking component for indoor environments entails two subcomponents: 1) a person detector and 2) a tracker for the detected target. The person detector extracts the position of a person moving around the scene. Then, it encapsulates the person in the smallest rectangular region that is called bounding box. The bounding box and the location determined as the person position are seeded to the tracker. The tracker computes the trajectory of the person over all the input sequence while moving around the scene.

Target:

In order to perform in real-time, the component is developed for a GPU (Graphic Processing Unit), using NVIDIA CUDA C++ for high-performance computation.

Input:

The person detector input is the set of consecutive frames.

The tracker takes as input the video sequence extended by the bounding box and the location of the person.

Output:

The person detector outputs are: position is determined by *(x,y)* coordinates in pixels in the first image, and the bounding box is centred at that location and characterized by *(width, height)*.

The tracker outputs the trajectory of the person *(x,y)* coordinates in pixels in each frame.

While the person detector is run only once to provide the input for the tracker, the tracker is run continuously for each frame, in order to reduce computation. However, the person detector is run every 1 or 2 seconds and the position is used to reduce the uncertainty of the person tracker, that can be affected by ambiguities such as occlusions, changes in illumination, or multiple targets.

4.1.1.1 Qualities, tunable set-points and available design-time and run-time parameters

With respect to the qualities, we are taking into account accuracy and performance. Accuracy is measured by the Deviation of the location determined for the target, and the PASCAL criterion measured by the overlap of the bounding box covering the ground-truth defined by a person. Bear in mind that the component is defined by the tracker, and the person detector is only meant to provide the position of the target to initialize the location. Then, the qualities to be measured concern only the visual tracking operation, not the human detection (although it also affects the final accuracy). Performance is given by the number of frames tracking locations provided per second. With respect to the setpoints, some of the potential features to be tuned are given by the following trade-offs: a) number of targets to be processed vs number of frames processed per second; b) accuracy vs number of frames processed per second.

4.1.1.2 Design-time support

In order to develop this component, general design and development tools are used to implement the CPU and GPU code.

4.1.1.3 Run-time support

- Monitoring: Three different profile configurations providing different number of tracked targets at the expense of lower processing performance
- Programming APIs: NVIDIA GPU boards, and CUDA package C++ programming language

4.1.2 Related state of the art

The process is performed usually as follows: 1) extract candidate regions for potential human targets, 2) extract features to build a model of the target, 3) determine if the target is a human or not, 4) compute the centre of masses and the dimensions of the smallest bounding box around the target. Some of the potential human descriptors listed in the literature are shape; Local Binary Patterns, Histogram of Gradients; motion; fusion of different features.

Second, tracking is the estimation of the trajectory of a target moving around the scene, over consecutive frames. A taxonomy is provided next:

A) *Matching*: These methods compute a representation of the target model using previous frames, e.g. by sampling potential candidates in a window around the position of the previous target location. Then, all possible candidates are compared by matching using normalized cross correlation, intensity gradient measures in the Lucas-Kanade fashion, or mean-shift. After that, the new location is determined selecting the candidate with the highest score.

B) *Matching with appearance model*: previous approach extended with an appearance model strengthened with possible variations of the target.

C) *Matching with constraints*: sparse optimization.

D) *Using discriminative classification*: Based on foreground-background differentiation, building a classifier for the foreground (target) that is updated continuously.

With respect to the features used to build the models, some examples are spatio-temporal features, gradient features, colour, texture, or the fusion of multiple features.

Finally, some of the issues when performing visual tracking are: change of perspective, noise, changes in the illumination, complex motion, non-rigid targets, occlusions, real-time performance and number of targets (scalability).

4.1.3 Extension within FitOptiVis

Current methods for tracking achieve accuracy of more than 80% in average, but when dealing with occlusions, multiple targets, or changes in illumination this accuracy drops to near 40% - 50% (understood in deviation of bounding boxes of less than 5-7 pixels for VGA resolution. With respect to the performance, methods drop to 7 fps for person tracking [KUO11]. We expect to achieve accuracies of 65% - 70% when dealing with occlusions while reaching up to 25 fps for person tracking. Although the number of targets is not taken into account here, our expectation is to perform person tracking when dealing with up to 5 targets, while maintaining similar accuracy and performance levels (with an estimated margin of 10%).

<u>TRL@M0:</u>
TRL1 - Only the model was defined, no other library/code was available.

FitOptiVis extensions:
At M12, the component is developed and it is able to track up to 3 targets reaching state of the art performance, while achieving around 50% of accuracy. No robust behaviour is implemented yet (no occlusion management, no illumination change invariation).

TRL@M36:
TRL4-6 - We expect a tested library for robust indoor person tracking for, included in the system demonstrators for the Smart Grid and the Habit Tracking UCs.

4.1.4 Prospective adoption FitOptiVis UC

Applied in FitOptiVis to the habit tracking UC and the Smart Grid UC.
Other fields of application not considered in the project: Intelligent Transportation Systems, pedestrian traffic management, tourist flow estimation, search and rescue applications, automatic counting systems, or human-robot interaction.

4.2 Behaviour classifier [UGR]

4.2.1 Short description

This component is a library for action classification using machine-learning techniques that recognizes a human action performed over a video sequence that contains a complete action execution. The component is intended to be trained indoors for monitoring the elderly at their own house (Habit Tracking UC) and a different instance outdoors for surveillance applications (Smart Grid UC).
Target:
In order to perform in real-time, the component is developed for a GPU (Graphic Processing Unit), using NVIDIA Cuda C++ for high-performance computation, and Machine Learning toolsets such as Tensor Flow.
Input:
The behaviour classifier input is a set of consecutive frames also called a video snippet that shows the complete execution of an action.
Output:
The output is a label that describes the recognized human action. The actions are determined depending on the task to be performed: for the Habit Tracking UC potential actions are watching TV, preparing coffee, cooking, and potential interesting action to trigger alarms such as accidental fall, fainting, or forgetting to turn off the stove; for the Smart Grid UC potential actions are walking, exercising/running, walking a dog, or potential risks such as trespassing, or lurking.

The action classifier is not run constantly; it requires sequences of frames to the action inference.

4.2.1.1 Qualities, tunable set-points and available design-time and run-time parameters

With respect to the qualities, we are taking into account accuracy, performance, and Precision-Recall curves (of F-measure). Accuracy using P-R curves define the relationship between the precision or fraction of positive labels that are correct, and recall or fraction of real positive were correctly labelled. A combination of precision and recall is given by the F-measure or F-score, basically defined as a harmonic mean. The performance is given by the number of actions labelled per second, depend on the size

of the video input streams and the number of labels or classes initially considered, since they determine the complexity of the network.
With respect to the set points, some of the potential features to be tuned are given by the following trade-offs: a) precision vs recall; b) classification accuracy vs performance; c) number of considered classes or labels.

4.2.1.2 Design-time support

In order to develop this component, general design and development tools are used to implement the CPU and GPU code.

4.2.1.3 Run-time support

- Monitoring: At least two different profiles that will be driven by the environment: outdoors (video surveillance application), and indoors (action – microaction classification of the elders at home)
- Programming APIs: NVIDIA GPU boards, and CUDA package C++ programming language, tensor flow package for machine learning application

4.2.2 Related state of the art

Human actions are performed pursuing some purposes. Understanding human actions through a vision system and determining its purpose is quite relevant for many applications. The definition for action ranges from a simple hand movement to complex actions composed by multiple simple actions (or microactions), performed over a sequence of consecutive frames that usually last a few seconds.
Action recognition entails 1) the action representation that extracts features from the video sequence and 2) the action recognition or classification that is fed with the action vector and infers an action label.
The action representation component requires features that extract discriminative information while ensuring invariation against illumination changes, camera perspective, pose, and speed. Some methods use holistic representations such as Motion History Volumes, shape features or silhouette; others work on local representations defining local motion in space-time such as 3D Histograms of Oriented Gradients, histograms of optical flow, or Motion Boundary Histograms. With respect to the action classifiers, some examples include nearest-neighbour clustering, Hidden Markov Models, Support Vector Machines. However, the most successful approaches are based on deep learning approaches require large datasets instead of hand-crafter features. In this case, both components are merged into the same networks, and are substituted by the convolutional operation to extract features frame by frame, and the temporal modelling. There a three main approaches: 1) applying 3D convolutions to consecutive frames capturing the temporal dynamics from them; 2) modelling temporal dynamics using a stream for optical flow (flow net) while the other captures 2D spatial features; 3) using temporal pooling to capture temporal information and use 2D convolutions for the spatial features, aggregating by using an LSTM model on top of a 2D ConvNet.

4.2.3 Extension within FitOptiVis

Current methods for action classification reach accuracies between 40-50% in average, but only when trained in specific datasets with large number of examples for each class, and with actions that are generic and easily discriminated.

With respect to the performance, it ranges from 1-2 fps for large datasets with a dozens of classes. We expect accuracies of 50% - 60% when dealing with the actions specific to the use cases for which the component is developed. We also expect higher time performance around 25 fps.

TRL@M0:
TRL1 – At M0 only the model was defined, no other library/code was available. Only state of the art pre-trained models are available, but not tailored.

FitOptiVis Extensions:
At M12, the first version of the learning network is setup and partially trained on different datasets. It reaches an accuracy close to 50%. With respect to the performance, it is still not running in real time.

TRL@M0:
TRL4/6 - At M36, we expect a tested library for action classification for different scenarios, included in the system demonstrators for the Smart Grid and the Habit Tracking UCs.

4.2.4 Prospective adoption FitOptiVis UC

Applied in FitOptiVis to the habit tracking UC and the smart grid UC.
Other fields of application not considered in the project: human-robot interaction, video retrieval or video scripting, gaming industry, or autonomous driving vehicles.

4.3 Salmi AR platform [HURJA]

Hurja will implement in the FitOptiVis project Salmi AR Platform capable of assisting brain damage patients and elderly people in their daily tasks, monitoring daily activities of users, and monitoring vital signs of users.

4.3.1 Short description

Brain damage patients or elderly people are wearing AR-glasses (such as Magic Leap or HoloLens 2) that assist/remind them on different important daily tasks, such as taking medicines, time to eat/sleep, how to make tea/coffee, how to brush teeth as well as having nurse/doctor/relative calls directly through AR-glasses, communicating with other peers (i.e., other brain damage patients or elderly people), monitoring users health condition, and automatically alerting nurse/relatives in case of emergency. Wireless indoor positioning will be utilized to enable more accurate information to be shown on AR-glasses.
AR-glasses will be UI for Salmi AR service and we plan to utilize IBM Watson based talking AI bot for voice communication between user and service. Service will be used through AR-based mobile application running on tablet or smart phone.

Salmi AR Platform component has three subcomponents:

- Extent Mobile Application,
- Salmi AR Mobile Application
- Salmi MAPS Website.

4.3.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:
Near real-time (soft real-time) performance, quality, and optimal energy usage requirements
Set Points:
Near real-time performance vs Quality vs Energy usage vs Data amount
Run-time performance monitoring:
Yes
Available model: Algorithms
Reconfiguration in the model:
Different configurations are possible

4.3.1.2 Design-time support

Salmi AR Platform will be developed using Unity 3D, C#, JSON, and PHP. Generic IDEs (Eclipse, Netbeans, etc.) will be used to handle component design, development, and verification.

4.3.1.3 Run-time support

Mobile application called Extent downloads on request a JSON packet that consist of list (descriptions) of wakeup images/objects/entities/actions. The request can come from Salmi MAPS website, Salmi AR mobile application, or directly from Extent mobile application if “free roam”-state has been switched on (requires GPS).

Run-time adaptivity support:
End-users have possibility to switch “free roam”-state off anytime they want and when this happens, *Extent mobile application* downloads new contents only by the request of an external source (at the moment only Salmi system related sources are available). *Extent mobile application* downloads all needed wakeup images, 3D-models, textures, audio files, videos, etc. based on the instructions received via JSON packet. To optimize the run-time performance of the Salmi AR system, all of these packets can be downloaded in advance. All files will be saved locally into end-users mobile device (smart phone or tablet) and those will be shown to end-users based on instructions received via JSON packets as soon as matching wakeup image/object/entity/action has been found or when an end-user is within a certain pre-defined distance from the target. Free roam data will be removed on-the-fly from end-users’ devices when each session ends. *Extent mobile application* is currently being developed using C# programming language on top of Unity 3D and back-end side is currently being developed using PHP. During our early testing phase, all description packets are in JSON format.

Monitoring:
Includes measured performance and energy usage, which can be handled by a generic data model. Relevant metrics to be monitored/evaluated are the following:

- Near real-time (soft real-time) performance: System performance can be monitored/evaluated in terms of frames-per-second or kilobits-per-second, but AR-feature robustness/performance depends highly on the selected AR-glass model. We plan to utilize Magic Leap and/or HoloLens 2 in the beginning of the development to make sure that all possible use cases can be implemented easily and later on we can possibly utilize also some other cheaper and less powerful AR-glass options that require more optimization for the system code for achieving same level of performance than with these high-end state-of-the-art AR-glasses. Furthermore, we plan to utilize smart feature extraction, segmentation, and classification algorithms meant for low-power mobile platforms, such as our Salmi AR Platform, in order to reduce bandwidth usage by only sending the necessary parts of images/videos and thus achieving near real-time (soft real-time) performance.

- Optimal energy usage: It is not an easy task to calculate the initial energy usage for the whole Salmi AR system before the first MVP version is fully implemented, but continuous camera feed and required advanced algorithms will produce us a challenge in terms of optimizing the energy usage of the system as a whole. As soon as the first MVP version is ready, we will perform extensive measurements on power usage and based on the achieved results, we will make needed adjustments to the implemented algorithms for enabling optimal energy usage of Salmi AR system.

4.3.2 Related state of the art

N/A

4.3.3 Extension within FitOptiVis

TRL@M0: TRL 2-3:

Potential application validated in terms of market needs and development of initial version of Salmi AR service Proof-of-Concept started.

Expected Developments/New Features:

Brain damage patients can perform their daily tasks, such as making tea/coffee or brushing teeth, which would not be normally possible without help of assistive technologies. Elderly people can live several years longer at home with help of Salmi AR Platform. Our platform provides context-awareness (i.e., what exactly brain damage patients or elderly people are doing at any given moment) and vital signs monitoring with automatic alerts in case of emergency, thus relieving workload of nurses/relatives. Video calls (point-to-point and group calls) with relatives/nurses/doctors/peers when needed (and even by automatic/forced manner in case patient or elderly people is not in good enough condition to make calls themselves). Salmi AR Platform provides: 1) Huge cost savings through increased efficiency of human resources utilization and significant reduction in hospitalization times! 2) Significant improvement in rehabilitation rates! 3) Better quality-of-life for brain damage patients and elderly people! 4) Extremely easy/convenient to use, motivating, and gives patients/users WOW Effect!

TRL@M36: TRL 7:

A complete Salmi AR service will be available at M36, tested in real world environments with real end-user pilots and thus Salmi AR service will be ready for production use with several different customers when the project ends.

4.3.4 Prospective adoption

Hurja will contribute to both habit tracking and VR/AR use cases with its Salmi AR platform. In addition to assisted living / remote health-care services for brain damage patients and elderly people, Salmi AR service can be used for any kind of domain-independent remote consultation services by using its video connection (point-to-point and group calls) and advanced AR/MR/XR features.

4.4 Distributed video coding solutions [UnivAQ]

Current video compression standards, (i.e., MPEGx or H.26x) are characterized by higher computational complexity at the encoder than at the decoder. The asymmetry design is tuned to broadcasting or streaming video-on-demand scenarios in which the video is compressed once and then decoded several times. In other scenarios (i.e., wireless video sensors for surveillance), a different distribution of the computational complexity is requested.

4.4.1 Short description

This software tool implements an algorithm for Distributed Video Coding (DVC) in order to exploit the correlation among videos recorded by multiple cameras.

Target:
Target technology will be a function of the chosen video HW components (cameras), which will also identify the Language. MATLAB will be used at the decoder.
Input:
Videos
Output:
Compressed video

4.4.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Quality: reduced computational complexity

4.4.1.2 Design-time support

MATLAB

4.4.1.3 Run-time support

There are no adaptive run time parameters for this component.

4.4.2 Related state of the art

There are two main approaches in literature to DVC:
1) Berkley (PRISM) [1]: block based with motion estimation at the decoder.

- Enhancement to PRISM is based on multi-level coset codes as proposed in [3].

2) Standford (pixel domain Wyner–Ziv architecture) [2]: frame based, which has gained much popularity because of its comparatively better rate-distortion performance, the most famous version is called DISCOVER.

- Enhancements are focused on modules such as
  - rate-control, [4,5]
  - decoding algorithm, [6,7,8,9]
  - side information generation, [10]
  - correlation noise modelling, [11].

Based on the analysis performed in [14] relevant topics related to DVC are:

- *Coding efficiency*: as already stated, DVC outperforms H264/AVC intra coding method except for scenes characterized by complex motion. Nevertheless, the performance remains significant lower than a full H264/AVC codec. In literature, different solutions have been proposed: spatial smoothing, refinement of motion vectors, use of forward and backward motion vectors, subpixel accuracy concept, exploitation of bit plane decoding for motion vectors remaining.

- *Complexity*: several iterations are required at the decoder to converge to a solution. In order to solve this issue, some solutions encompass a hybrid encoder-decoder, a parallel implementation of various channel decoding algorithms, and a change in the complexity of operations at the viable node.
- *Robust transmission*: the decoding should be performed to be successful even if packet losses occur. Among the solutions available in literature, the encoding of auxiliary frames only for some frames, exploitation of unequal error protection mechanisms.
- Scalability: due to the absence of any closed loop in its design, DVC supports codec independent scalability that can be achieved through the development of an enhancement layer, posing the problem as a variant of the WZ side information problem.
- Multiview: compared to monoview the difference is that the SI can be coded not only form the previously decoded frames in the same view, but also from other views. Many solutions have already been proposes like disparity compensation view prediction (DCVP), Multiview Motion Estimation (MVME), exploitation of a homography model, and View Synthesis Prediction (VSP).

In [12] a comparison between H264/AVC and DVC encoders power consumption is presented with promising results reporting that, for available DVC implementations and considering the power consumption/compression efficiency ratio, when comparing to compression algorithms based on differential frame coding (with zero search radius for ME) the advantages of DVC are not so evident.

A comparison in terms of rate-distortion curves between H264/AVC and DISCOVER (a well-known DVC solution) is carried on in [13]. The results show that the DVC codec may outperform the traditional intra codec; however, the performance is still far behind that of the inter codec (H.264/AVC SP). Another interesting observation is that the performance highly depends on the characteristics of video sequences. DVC may outperforms H.264/AVC No Motion for a sequence where the global motion cannot be addressed well by the simple zero motion vector scheme; however, for a sequence coming from a static camera for surveillance applications, H.264/AVC No Motion outperforms DVC and H.264/AVC Intra codec. These results show that existing DVC systems can be further improved. The power consumption analysis is also carried on in [13] where both processor-based and ASIC-based platforms are analysed. The analysis of the results of the processor-based platform shows that DVC technique can provide a power-efficient solution. Concerning ASIC-based platform, it shows that although the power consumption of a DVC encoder is only 7% of an H.264/AVC Simple Profile encoder, the power efficiency of DVC is still very similar to that of H.264/AVC No Motion codec because of the poorer coding efficiency.
It seems that the DISCOVER codec is one of the best performing DVC schemes reported in literature, [14].

4.4.3 Extension within FitOptiVis

<u>Expected Extensions:</u>
DVC solutions will be tested within FitOptiVis framework with the main purpose of guaranteeing efficient and reliable information exchange. In particular the analysis of WSNs for video monitoring based on DVC will be carried on in order to reduce requirements about: processing, bandwidth and power consumption.

4.4.4 Prospective adoption FitOptiVis UC

Water supply (Use Case 1)

4.4.5 Related documents

[1] Puri R, Ramchandran K (2002) PRISM: a new robust video coding architecture based on distributed compression principles. In: Proceedings of the 40th Allerton Conference on Communication, Control and Computing, Allerton, IL, USA, 2–4 October 2002.

[2] Girod B, Aaron AM, Rane S, Rebollo-Monedero D. Distributed video coding. Proc IEEE. 2005;93(1):71–83. doi: 10.1109/JPROC.2004.839619.

[3] Majumdar A, Ramchandran K (2004) PRISM: an error-resilient video coding paradigm for wireless networks. In: Proceedings of the First International Conference on Broadband Networks, San Jose, CA, USA, 25–29 October 2004

[4] Du B, Shen, H (2009) Encoder rate control for pixel-domain distributed video coding without feedback channel. In: Proceedings of the Third International Conference on Multimedia and Ubiquitous Engineering, Qingdao, China, 4–6 June 2009

[5] Morbee M, Prades-Nebot J, Pizurica A, Philips W (2007) Rate allocation algorithm for pixel-domain distributed video coding without feedback channel. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Honolulu, HI, USA, 15–20 April 2007

[6] Tagliasacchi M, Trapanese A, Tubaro S, Ascenso J, Brites C, Pereira F (2006a) Intra mode decision based on spatio-temporal cues in pixel domain Wyner–Ziv video coding. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Toulouse, Italy, 14–19 May 2006

[7] Roca A, Morbee M, Prades-Nebot J, Delp EJ (2007) A distortion control algorithm for pixel-domain Wyner– ZIV vide coding. In: Proceedings of the 26th Picture Coding Symposium, Lisbon, Portugal, 7–9 November 2007

[8] Avudainayagam A, Shea JM, Wu D. Hyper-trellis decoding of pixel-domain Wyner–Ziv video coding. IEEE Trans Circuits Syst Video Technol. 2008;18(5):557–568. doi: 10.1109/TCSVT.2008.918785.

[9] Trapanese A, Tagliasacchi M, Tubaro S, Brites C, Pereira F (2005a) Embedding a block-based intra mode in frame-based pixel domain Wyner–Ziv video coding. In: Proceedings of the 9th International Workshop on Very Low Bitrate Video Coding (VLBV), Sardinia, Italy, 15–16 September 2005

[10] Natário L, Brites C, Ascenso J, Pereira F (2005) Extrapolating side information for low-delay pixel-domain distributed video coding. In: Proceedings of the 9th International Workshop on Very Low Bitrate Video Coding (VLBV), Sardinia, Italy, 15–16 September 2005

[11] Brites C, Ascenso J, Pereira F (2006) Studying temporal correlation noise modeling for pixel based Wyner–Ziv video coding. In: Proceedings of the IEEE International Conference on Image Processing, Atlanta, GA, USA, 8–11 October 2006

[12] A. Ukhanova, E. Belyaev and S. Forchhammer, "Encoder power consumption comparison of distributed video codec and H.264/AVC in low-complexity mode," Software, Telecommunications and Computer Networks (SoftCOM), 2010 International Conference on, Split, Dubrovnik, 2010, pp. 66-70.

[13] S. Y. Chien et al., "Power Consumption Analysis for Distributed Video Sensors in Machine-to-Machine Networks," in IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 3, no. 1, pp. 55-64, March 2013.

[14] Distributed Video Coding: Trends and Perspectives, Frederic Dufaux, Wen Gao, Stefano Tubaro and Anthony Vetro EURASIP Journal on Image and Video Processing 2010 009:508167.

4.5 Virtual reality demonstrator [Nokia]

4.5.1 Short description

The Virtual Reality (VR) demonstrator can process point cloud data in an efficient way. It consists of an encoder, a streaming part, as well as decoder blocks. The main challenge is to process the huge amount of data in real system with high quality. Key software and hardware elements of the VR demonstrator have also been identified to be as follows: 1) point cloud encoder 2) video encoder / decoder 3) point cloud file encapsulation 4) point cloud streamer 5) point cloud renderer and 6) mobile collaboration application.

Type - SW
Target - GPU and OpenGL ES 3.2 software
Input - Video streams and point cloud data
Output - Video streams and point cloud data
Communication capabilities - Point cloud data with different parameters

4.5.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities - Performance, quality, real time requirements, reduced computational complexity
Set Points - Performance vs Quality vs Data amount
Available model - Algorithms
Reconfiguration in the model - Different configurations are possible

4.5.1.2 Design-time support

DT tools - General purpose tools (C++, Matlab)
Availability - The results can be available via open standards

4.5.1.3 Run-time support

RT adaptivity – This will be studied in the later research phase
Programmability - This will be studied in the later research phase
Programming API - OpenGL
Run time monitored events - These features will be designed later

4.5.2 Related state of the art

Advances in 3D sensing and capturing technology have unleashed a new wave of innovation in Virtual/Augmented/Mixed reality (VR/AR/MR) content creation and communication, as well as 3D sensing for smart city, robotics and automated driving applications. There is now a huge interest from the virtual reality market in being able to represent digitally the real world in three dimensions, thus enabling the end-user to navigate freely in this digital representation.

Volumetric visual data describes a 3D scene and objects with its geometry (shape, size, position in 3D-space) and respective attributes (e.g., colour, opacity, reflectance, albedo), plus any temporal changes. Such data is typically computer-generated from 3D models, or is captured from real-world scenes using a variety of solutions such as multiple cameras or a combination of video and dedicated geometry sensors.

Common representation formats for such volumetric data are polygon meshes or point clouds. Temporal information is included in the form of individual capture instances, similar to frames in a 2D video, or by other means, e.g., position of an object as a function of time. Because volumetric video describes a complete 3D scene or object, such data can be visualized from any viewpoint. Therefore, volumetric video is a key enabling technology for any AR, VR, or MR applications, especially for providing Six Degrees of Freedom (6DoF) viewing capabilities.

4.5.3 Extension within FitOptiVis

TRL@M0 - TRL2
Developments and testing of new features for point cloud processing
TRL@M36 - TRL5-6, the demonstrator will be tested on real case(s)

4.5.4 Prospective adoption FitOptiVis UC

Virtual Reality Demonstrator use case;
Fields of application - Smart cities, smart modelling, smart visualization

4.5.5 Related documents

[1] Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A. Chou, Robert A. Cohen, Maja Krivoku´ca, S´ebastien Lasserre, Zhu Li, Joan Llach, Khaled Mammou, Rufael Mekuria, Ohji Nakagami, Ernestasia Siahaan, Ali Tabatabai, Alexis M. Tourapis, Vladyslav Zakharchenko, “Emerging MPEG Standards for Point Cloud Compression,” IEEE JETCAS Special Issue on Immersive Video Coding and Transmission, 2019.

4.6 Point localization from stereovision [UWB, REX]

4.6.1 Short description

The component is a computation module that computes real-world 3D coordinates of points of interest. For these points are known 2D image coordinates (coordinates in perspective projection image) in 2 images taken from 2 different viewpoints – stereovision. Additionally, a set of reference points with known real-world 3D coordinates as well as coordinates from the 2 perspective projections is supplied.

Input (function arguments passed in shared memory or over TCP/IP stream)

N_R	Number of reference points (RP)
N_P	Number of points of interest (POI)
RP_{World}	Matrix N_Rx3 with [x,y,z] real-world coordinates of the reference points
RP_{Img1}	Matrix N_Rx2 with [x,y] image coordinates of the reference points in the first projection view (extracted from 1st camera image)
RP_{Img2}	Matrix N_Rx2 with [x,y] image coordinates of the reference points in the second projection view (extracted from 2nd camera image)
POI_{Img1}	Matrix N_Px2 with [x,y] image coordinates of the POIs in the first projection view (extracted from 1st camera image)
POI_{Img2}	Matrix N_Px2 with [x,y] image coordinates of the POIs in the second projection view (extracted from 2nd camera image)

Output (results passed out in shared memory or TCP/IP stream)

POI_{World}	Matrix N_Px3 with [x,y,z] real-world coordinates of the POIs
R	The location and number of reference points and location of the 2 cameras involve the precision and stability of the computing algorithm (dependability of the system of equations). This result measures the quality of the produced output coordinates.

4.6.1.1 Qualities, tunable set-points and available design-time and run-time parameters

With respect to the module objectives, accuracy, stability and performance measures are taken into account. At first, depending on application preferences, (precision vs performance) computing using double-precision floating point numbers (64-bit) or single-precision floats (32-bit) can be used.
As second - the form of used algorithms is not determined yet, but probably iterative algorithms will be used. In this case, conditions on convergence, step deltas or number of iteration cycles can be applied.

4.6.1.2 Design-time support

The first version of the component is to be designed and implemented in Microsoft Visual Studio and compiled with the provided MS C++ compiler or with GNU C++ compiler (for ARM based HW platforms).
Additionally, the used REX block SDK provides framework and base class implementation for the component. Only functional parts of the component have to be implemented - functions Init(), Exec() and Exit() (called upon system initialization, periodically during system run and upon system shutdown).

4.6.1.3 Run-time support

In run-time, the module is configured and observed via configuration window accessible through RexDraw (newly re-branded to REXYGEN Studio) - GUI based application that configures and inspects configuration of the real-time control application.
Component set points can be inspected and adjusted here, the results of the last evaluation (POIs coordinates and result reliability) are also available.

4.6.2 Related state of the art

Nowadays most of the 3D localization procedures use key-point detection algorithms executed above stereo image data and followed by point registration algorithms to pair corresponding image points. The subsequent 3D localization algorithms must count with inaccurate registration, as it is the most problematic part of the processing chain here. Using the active synchronized LED markers and differential imaging it is possible to ensure 100% correspondence between point pairs in stereo image and use straight computing algorithms to localize the POIs in 3D.

4.6.3 Extension within FitOptiVis

TRL at M0:
TRL 0.
TRL at M36:
UWB expects to deliver designed and fully implemented component that is tested on real industrial robots in lab as well as in controlled industrial environment. TRL 5 - 6

4.6.4 Prospective adoption FitOptiVis UC

The component will be used within the FitOptiVis demo pool as one of the principal components in the robot calibration use case (UC8).
Outside the project scope, it can be used in any application where 3D point localization from stereo images is required. A needed prerequisite is successful and 100% accurate point registration, which can be acquired for example using the other proposed component – synchronized LED markers, and differential imaging.

4.6.5 Related documents

[1] „Getting started with REXYGEN and Raspberry Pi", REX Controls s.r.o., https://www.rexygen.com/doc/ENGLISH/MANUALS/RexygenGettingStarted_RasPi/RexygenGettingStarted_RasPi_ENG.html
[2] „REXYGEN Studio User Guide", REX Controls s.r.o., https://www.rexygen.com/doc/ENGLISH/MANUALS/RexygenStudio/RexygenStudio_ENG.html
[3] Olivier Faugeras "Three-Dimensional Computer Vision : A Geometric Viewpoint", MIT Press Ltd, 1994, ISBN10: 0262061589, ISBN13: 9780262061582
[4] Rimon Elias, Robert Laganière: "Projective Geometry for Three-Dimensional Computer Vision", School of Information Technology and Engineering University of Ottawa, Ottawa, Ontario, Canada, K1N6N5
[5] Chien-Ping Lu, Gregory D. Hager, Eric Mjolsness: "Fast and Globally Convergent Pose Estimation from Video Images", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, No. 6, June 2000

4.7 Video content analysis (VCA) platform [Aitek]

Video content analysis is the capability of automatically analysing video to detect and determine temporal and spatial events, which can be used to trigger context switch, more-advanced processing in the systems or, in some cases, specific alarms.

4.7.1 Short description

The VCA platform includes a C++ based software modules suite implementing Video Analytics techniques for real-time detection of different events involving the safety and security of the monitored areas focusing on people, vehicles and static objects. These software modules are able to process live images from video-surveillance cameras and to perform post-event analyses of recorded footage and acquire statistical data.
Input:
Images recorded by cameras (including the ones installed on a drone)
Output:
Alarms after each event detection. Still images, video streams and related metadata that show the detected event
Composability:
The platform is composed by the following sub-components

- IP / analogue cameras;
- IP connection towards the Network Video Recorder (NVR).
- NVR/HDVR devices hosting video-surveillance and video-analytics SW;
- Other optional components like video management application, decoders and client.[1]

[1]These components will be described in the WP6 as they could be used in the demonstrator but they are not developed/improved during FitOptiVis.

It is worth noticing that VCA software can be executed on the camera, on the NVR or partially on camera partially on NVR, following a hybrid approach. In this last case, the camera takes care of preliminary processing steps, generating metadata processed by the NVR to detect the events.

4.7.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Quality of the video stream is a fundamental parameter for the VCA platform. It is expressed mainly in terms of frame rate and frame quality. As a matter of fact, these parameters has a dramatic impact on the event detection accuracy.
As explained in the next section, the offered load (i.e. the bandwidth needed to transmit the video flow) is another important set-point, which has to be considered according to the bandwidth available in the channel.
Considering the Pareto theory, the vector *<frame rate and frame quality>* represents the space in which the Pareto analysis should be done. For each point in this space corresponds an offered load that has to be lower or equal to the available bandwidth (on the network) which represents a boundary for the feasibility region that is an upper limit for frame rate and frame quality.
At the same time, frame rate and frame quality should be enough to guaranty sufficient VCA performance, defining another boundary of the feasibility region. Similar considerations (regarding the Pareto analysis) can be done for the computational resources needed to execute the processing algorithms, which may affect the supported frame rate and the frame quality.

4.7.1.2 Design-time support

The VCA platform, or at least some of its components, can be used as a test case for the accelerating tools developed by other FitOptiVis partners.

4.7.1.3 Run-time support

Monitoring:
Monitoring of the offered load compared to available bandwidth. This information is used to adapt frame rate and frame quality to maintain offered load lower than available bandwidth, as previously explained.
Runtime Adaptivity:
Run-Time adaptivity is currently under investigation. In particular, the focus is on the definition of mechanisms to adapt the characteristics of the video flows in terms of frame rate and frame quality at runtime. The system periodically monitors the offered load: if it exceeds the available bandwidth, the system triggers a modification of the video stream in order to reduce the offered load. Moreover, we are also studying similar approaches considering the computational resources available and resources required by video processing algorithms.
Programming APIs:
Some **programming APIs** are supported by the system. In particular, it has JSON-REST API (https). It has also an SDK in C. Moreover, similar API JSON-REST (https) are available specifically to configure the Video Content Analysis algorithms.

4.7.2 Related state of the art

There are two different approaches for video processing:

- Traditional algorithms, in which the video flow is analysed considering a sequence of correlated frames. Currently used approach includes a low level processing that consists in background and foreground separation, a middle level processing that produces as output a list of detected targets and a final processing that generates alarms analysing the characteristics of such targets.
- Innovative algorithms based on deep learning theory. In this case, the analysis can be done frame by frame independently. In practice, events detection can be done analysing just a single frame. Some examples include Single Shot Detection (SSD), Convolutional Neural Network (CNN), Recursive Neural Network (RNN), YOLO are some example of deep learning approaches currently used for image processing. This new approaches are targeted as possible improvement of the VCA platform within FitOptiVis.

4.7.3 Extension within FitOptiVis

TRL@M0

TRL4: Baseline on-going experimentations in lab in order to assess innovative video processing approaches based on deep learning algorithms.

Expected Extensions

Progresses during FitOptiVis, regard the integration of such new algorithms in our video monitoring solutions. These new processing algorithms may require a quite large amount of computational resources and re-configurability capabilities in case of insufficient resources available.

TRL@M36

TRL6/7: The goal is to integrate such new video processing approaches in our video security system in order to test them in relevant environments (i.e. the water supply system provided by SAT).

4.7.4 Prospective adoption

The proposed component will be used in water supply system demonstrator in UC1. The goal is to define and implement (at prototype level) an advanced security system able to detect intruders in dangerous areas, trying also to infer what he/she is doing, at least in a quite preliminary way. As explained in the previous section, this component will be generic enough to be used in other different applicative domains.

4.8 Face and behavioural analysis (FBA) platform [UNISS]

Face detection is a fundamental step in the "traditional" face analysis pipeline (including alignment, normalization and classification of images), and established techniques are available. Effective and efficient recognition and behavioural analysis are different and more complex functionalities.

4.8.1 Short description

This software component, mainly developed in C/C++, will be aimed at analysing live video sequences characterized by narrow to medium field of view and controlled or partly controlled light conditions. The main purpose of the component will be the detection and the automatic recognition of faces; more in detail

Principal key features and properties

- Efficient and multiple face detection under a large variety of poses;
- Face watch-list in order to verify whether a face is part of a known set of people;

- Basic behavioural analysis based on the temporal track of faces.

Processed inputs and produced outputs

The module will process video frames with arbitrary resolution. Detected faces and specific behaviours will be annotated through metadata (bounding boxes and labels) that will constitute the output of the component.

Composability

The module will involve a number of sub elements (processing cores) useful to extract specific image features. Most of these elements will be organized in form of C/C++ libraries.

4.8.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:

The component will manage a number of tunable features:

- frame rate (per second) and frame quality (image resolution);
- maximum number of tracked faces;
- upper/lower limits for valid face-box dimensions;
- quick change detection (QCD) based sensitivity
- image watch-list.

All the above qualities will be tunable at run-time, except for the watch-list that will be loaded at start-up.

Set-Point(s) and Parameters:

The component will usually work at full frame rate and full frame resolution. Different set points will correspond to integer sub-sampling in time or space.

Face-box dimensions will be given in percentage with respect to full frame resolution while sensitivity will be expressed in percentage with respect to optimal MAP estimate.

4.8.1.2 Design-time support

Generic IDEs (Eclipse, Netbeans) will be used to handle component design, development and verification.

4.8.1.3 Run-time support

Run-time adaptivity support:

Triggers based on the execution time of the component will be used for adaptation. To this purpose, output metadata will include timestamps associated to each processed frame. Timestamps analysis will offer to third party software the possibility to redefine set points and parameters in order to counterbalance low computational resources.

Monitoring:

The component will monitor continuously the time required to complete the processing of a single frame.

Programmability and Programming API:

Programmability support of the component will be offered by C/C++ APIs.

4.8.2 Related state of the art

Face detection has made significant progress in the last decade and a large number of face segmentation algorithms based on different assumptions and applications has been reported in the literature. For example, template matching methods use standard face patterns and derive from correlation a unique index stating the possible presence of a face. These methods are commonly unable to deal with variation in scale, pose and shape but can perform well in controlled applications like airport gates. Bottom-up methods are feature-based approaches that start from basic features (colour, contours, salient points) in order to infer the presence of a face. They usually can better manage for variation in scale but still suffer for variation in pose and shape. Holistic (appearance based) methods rely on templates and statistical analysis techniques; in this case,

however, templates are learned from example images. Neural networks and Support Vector Machines (SVM) belong to this class so as the Viola-Jones face detector, a well-known technique based on integral images that has made possible face detection in real time.
Face recognition share a number of basic tools with face detection. Feature based methods use features, geometric rules, graph matching and deformable templates. Many approaches have been proposed to localize and describe facial features (SIFT, SURF, GaborJets, LBPs among others) with very interesting results also in case of significant variation of scale and pose. Holistic methods use the whole image and a set of statistical and mathematical techniques (PCA, ICA, LDA, SVM) useful to better separate classes and improve the robustness to noise and discretization. Recent CNNs and DeepCNNs are holistic approaches based on networks with multiple convolution and pooling layers and complex learning techniques. They proved to perform very well in most practical cases, but they require long learning times and large number of training images. Innovation brought by FitOptiVis has to do with three main aspects:

- design and development of a fast face detection module inspired by Viola Jones but more related to facial features and more accurate for limited rotations of the head
- design and development of a face recognition module based on a new concept of similarity between facial features;
- design and development of a basic classifier able to learn from continuous observation and to correctly identify a limited number of behaviours.

4.8.3 Extension within FitOptiVis

TRL@M0: TRL3
A preliminary study and simple experimental proof-of-concept demonstrated the possible development of face detection and recognition algorithms deeply involving facial features. Current accuracy for face detection depends on the reference database used. For complex databases like LFW or IJB-C overall accuracy is under 90%. By fixing standard FAR (0.001) the verification rate is usually under 75%.
Current accuracy for face recognition depends on the reference database used and on the test protocol adopted. For complex databases like LFW or IJB-C and image-restricted protocols, accuracy ranges between 92% and 97% (on pre-detected faces). At the standard FAR (0.001) the verification rate is usually under 55%.
Extensions:
Development of a new algorithm for face detection/recognition, exploiting facial features and pose estimation. Full evaluation of the computational complexity of such new algorithm. Integration of
the algorithm in the video monitoring solutions of the company (AITEK) and test in the foreseen UC
TRL@M36:
6/7.
Technology integrated in the video security system of the company. Technology demonstrated in a relevant UC.

- Expected accuracy for face detection on complex databases like LFW or IJB-C over 90%.
- Expected verification rate at standard FAR (0.001) over 80%.
- Expected accuracy for face recognition on complex databases like LFW or IJB-C over 95%.
- Expected verification rate at standard FAR (0.001) over 60%.

4.8.4 Prospective adoption

The component will be adopted in Use Case 1: water supply system.
The component is generic enough to be used for a number of potential applications like:

- *Soft monitoring of public areas.*
- *Critical infrastructures monitoring (gates and physical access).*
- *Attendance registration.*
- *Smart Factories.*

4.9 Pose estimation engine [HIB]

4.9.1 Short description

This is a software component that takes as an input video that contains recordings of persons and generates an estimation of the pose (position of the extremities such as legs, arms and head) of the persons present.

Target:

The software is based on CMU Openpose, mostly coded using C++ and Python. We have also a wrapper so it can be used (with degraded performance) using Javascript from a regular internet browser.

Input:

Video streams from cameras. As of the moment, for the UC3 system being deployed for FitOptiVis, we are using Foscam FI9900EP Full HD POE cameras that we have been analysing during the first year.

Output:

A 2D image overlay of the points defining the pose of persons in a frame of video plus an array detailing these points as shown in the figure below.

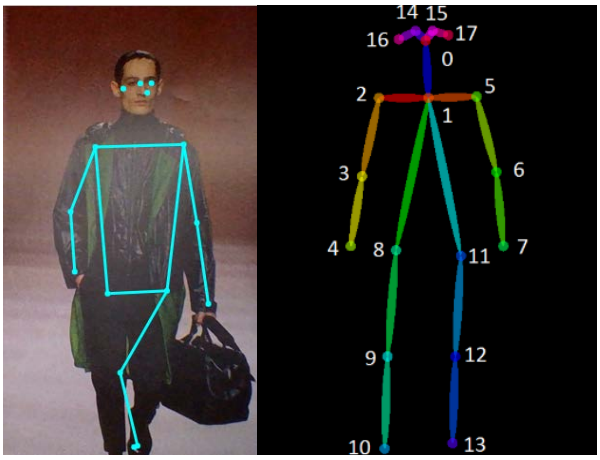

Figure 4: Output pose estimation

4.9.1.1 Qualities, tunable set-points and available design-time and run-time parameters

The quality parameters of the image and detection systems are as follow:

- Frames per second: the number of frames per second that are processed to deliver the results. From 15fps, acceptable results can be had but for fine movements up to 60fps are useful. Beyond that value, human movement and pose estimation does not get ant advantages, but could be used for more rapidly moving objects.

- Resolution: the resolution of the frames passed over to the system. We are working from VGA (640x480) to 2K images (1920x1080). The required resolution depends on the spatial resolution achieved from the placement of the camera.
- Number of tracked points: the person model is based on a series of joints for the skeleton (e.g., knee, wrist, elbow) and points for the face in the case of facial recognition. Higher number of points yields finer grained results but require more computational power to be processed in real time.
- Number of tracked persons: The system can detect features for more than one simultaneous person in the field of view. This can be configured and is subject of the same limitations as the complexity of each model.

The Set-Points for the system are to be released (will be based on the work in UC3), but are combinations of different number of detection points (complexity of single person model combined with number of persons to be recognized) and different resolution/framerate of video.

The planned adaptivity to be delivered using results in WP4 is the throttling the input video resolution and framerate in combination with the model complexity to adapt to performance set points.

4.9.1.2 Design-Time Support

At design time, during the Y1 we have just used general IDEs (Eclipse, etc.). It is expected that this can be complemented with variability-based tools to better describe valid set-point and resources combinations (see description in deliverable D3.1).

4.9.1.3 Run-Time Support

The component is now programmed using regular Deep Learning libraries (Tensorflow on Linux/Android targets plus experiments with Tensorflow using a Javascript backend). This uses GPU optimization by default but we are still investigating how to alter its functioning to match real-time changing resources at runtime. We are now investigating using OpenCL for GPU computing (as part of the activities of D4.1).

4.9.2 Related State of the Art

For detection of persons and their actions in video streams, we use the following assets:

- Frameworks: Keras[2], CNTK[3], TensorFlow[4]
- Models: CNN, LSTM[5]

[2] "Keras Documentation." https://keras.io/. Accessed 12 Jun. 2017.
[3] "Microsoft/CNTK - GitHub." https://github.com/Microsoft/CNTK. Accessed 12 Jun. 2017.
[4] TensorFlow: https://www.tensorflow.org/xla
[5] "Recurrent and Recursive Nets - Deep Learning Book." http://www.deeplearningbook.org/contents/rnn.html. Accessed 12 Jun. 2017.

- Datasets: Violent Flows datasets[6], UCF Web Abnormality [7], UMN Abnormal Crowd datasets, Stanford Question Answering Dataset[8]

The ability to detect anomalies in real-time is very valuable, so that appropriate actions can be taken as soon as it is detected to avoid or reduce negative consequences.

Most research works focus on hand engineering features for particular scenes or datasets, but these features need to be manually tuned each time a different scenario is introduced. We propose an automated video surveillance system to accurately determining pose of persons in a scene.

Body position detection state of the art is OpenPose[9] (Caffe and C / C ++), which also has a python wrapper to facilitate its use (PyOpenPose[10]). It is the combination that we currently have in our commercial LIFEonLive surveillance system (see section **Error! Reference source not found.**), with which we get 18 points as maximum in the model. There is also an alternative implementation of Tensorflow models in Javascript (tfjs-posenet[11]). With this, recognition can be done entirely in-browser.

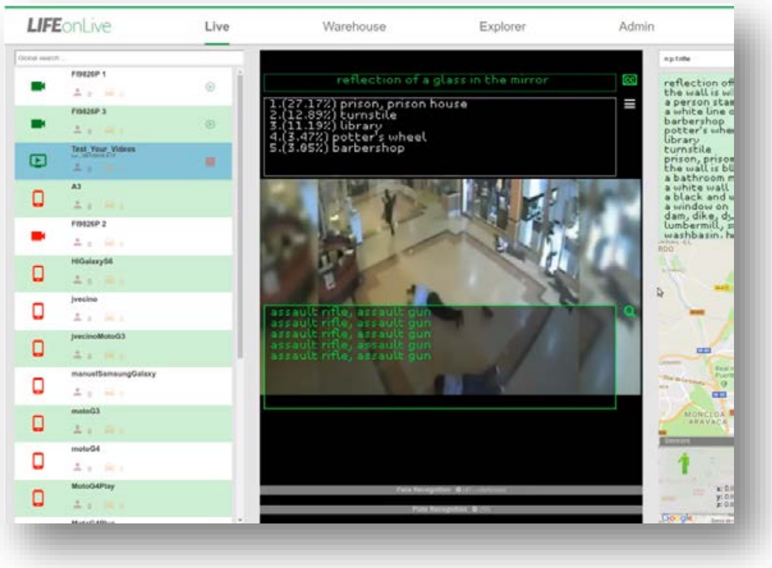

Figure 5: LifeOnLive surveillance screenshot

- LifeOnLive (https://www.youtube.com/watch?v=NFadp8N8KV4) is a mobile surveillance solution that enables security agents wearing bodycams to stream their activities to a data centre automatically. Besides, for purely storing the videos, there are analysis stages for automatically detecting features of interest

6 "Violent-Flows - Crowd Violence \ Non-violence Database and ...." http://www.openu.ac.il/home/hassner/data/violentflows/. Accessed 9 Jun. 2017.
7 "Abnormal Crowd Behavior Detection using Social Force Model - CRCV." http://crcv.ucf.edu/projects/Abnormal_Crowd/. Accessed 9 Jun. 2017.
8 "SQuAD." https://rajpurkar.github.io/SQuAD-explorer/. Accessed 12 Jun. 2017.
9 OpenPose Network: https://github.com/CMU-Perceptual-Computing-Lab/openpose
10 PyOpenPose wrapper: https://github.com/FORTH-ModelBasedTracker/PyOpenPose
11 tfjs Posenet Javascript wrapper: https://github.com/tensorflow/tfjs-models/tree/master/posenet

such as faces, license plates and others. Currently we are exploring the addition of pose estimation to provide more fine grained analysis of attitudes or behaviours of interest for security (e.g., raising the arms, pointing at something, aiming a gun).

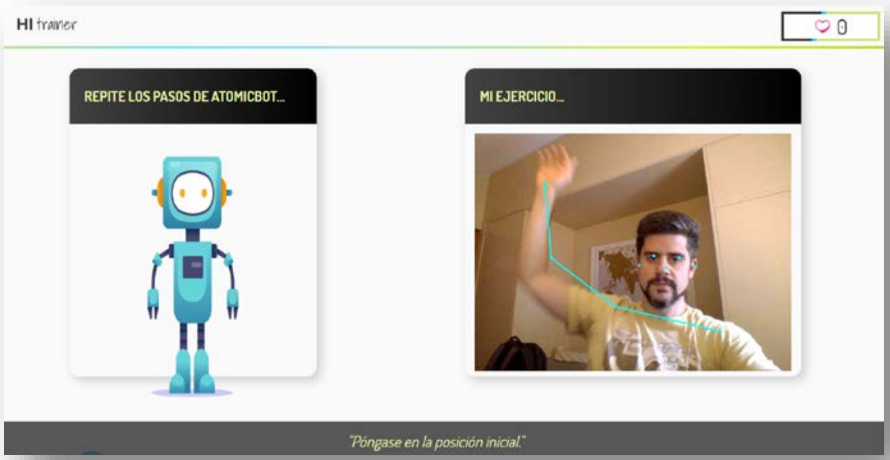

Figure 6: HI Trainer screenshot

- HI Trainer is an experimental module for our e-health products including REVITA (http://revita.hi-iberia.es/en/) in which a user (usually a senior citizen or person undergoing rehabilitation or physical therapy) is commanded to follow a series of exercises and then evaluated on their performance. This is a pre-commercial for now and not integrated in commercial releases of REVITA, but technology from FitOptiVis will be used in future evolutions.

4.9.3 Extension within FitOptiVis

Expected Extensions:

The particular features of the pose recognition system that we will pursue as part of FitOptiVis are the following:

- Adaptivity for changing context of operation (e.g., scaling down performance in low energy scenarios) and distribution of video analytics tasks on various devices.
- Execution in closer to market, more performance-constrained devices (ARM devices, in-camera hardware).

4.9.4 Prospective adoption FitOptiVis UC

HIB is implementing this component as part of the video processing technology in UC4: habit tracking. The target usage is the monitoring of persons in their own homes to detect situations of interest (falls, abnormal patterns of activity) that can be used by family members to check on their relatives' wellbeing and even by doctors remotely as clues of mild cognitive impairments being developed.

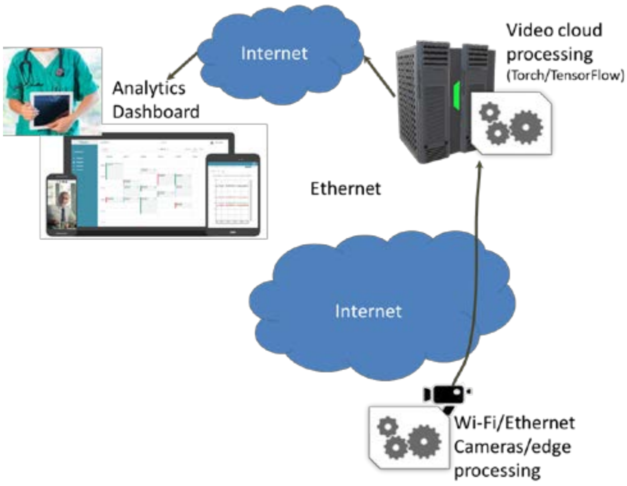

Figure 7: HIB architecture of distributed pose estimation processing in UC3

4.9.5 Related Documents

Additional Documents:
The aspects related to the processing system for pose estimation have been discussed in further length in D3.1 (for the modelling and design time tools explored) and in D4.1 (for the video analytics distribution and the run-time operations for resource management).

4.10 Reconfigurable video capture and processing [UC]

4.10.1 Short description

This module integrates video capture with some video processing algorithms. It supports several configurations and accelerator mapping. At runtime, some part of the module could be re-allocate to different computational resources. For example, the video capture module could be allocated in HW and supports board camera inputs or in SW and supports USB camera inputs. The component also includes a library of video processing algorithms that provides several functions such as feature detection. Some library functions also support HW and SW implementations that can be selected at runtime.

Type:
SW with HW accelerators.
Target:
Zynq UltraScale+ (or a similar FPGA) and an external computing platform.
Input:
Camera video frames.
Output:
Video image and results of the video processing pipeline.
Available model – The SW implementation includes a model for software simulation.
(Re-)configuration in the model – The component support several configurations in with the video source and video processing algorithms are defined. These configurations also define the HW/SW accelerator mapping. The configuration can be modified at runtime.

Composability - It can be involved in larger components.

4.10.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:
The qualities depends on the video processing pipeline. All of them have image resolution and frame rate qualities.
Set Points:
Every point defines, at least, video source input, video processing pipeline and module implementation (HW/SW mapping).

4.10.1.2 Design-time support

DT tools – Xilinx SDSoC and reVISION library. UC runtime support library. OpenMP with UC extensions. UC modeling framework (S3D).

4.10.1.3 Run-time support

Runtime reconfigurable modules and runtime manager.

4.10.2 Related state of the art

The HW camera component is based on the Xilinx revision module. It has been adapted for runtime SW/HW reconfiguration.

4.10.3 Extension within FitOptiVis

TRL @ M0:
Xilinx reVISION interface.

TRL @ M36:
First complete version. Runtime reconfigurable component that can be integrated in the autonomous vehicle use case.

4.10.4 Prospective adoption FitOptiVis UC

Usage in FitOptiVis use cases: UC10 and UC2 (optionally).
Fields of application – Mainly satellite image processing application

5 Communication-oriented components

This section presents FitOptiVis communication-oriented IPs that are intended as gateways and on chip traffic managers.

5.1 Time sensitive networking [7SOLS]

Time Sensitive Networking (TSN) guarantees bounded low latency, low packet delay variation and low packet loss in the context of mixed-criticality applications. TSN, a set of updates of IEEE Ethernet standards, brings backbone for real-time distributed control systems (Surveillance of smart-grid critical infrastructure UC) or edge-cloud distributed processing and performance monitoring (Habit tracking UC), among other possible applications.

5.1.1 Short description

The TSN component is a mixed software and IP-core-based library to enable Time Sensitive Networking switching and forwarding capabilities in Xilinx FPGA SoCs. The TSN component is composed by two interdependent functional sub-modules:

- the Hybrid Communication Manager subsystem
- the Network Timing subsystem.

The Hybrid Communication Manager (HCM) subsystem provides differentiated or hybrid RT-QoS (bounded latency, guaranteed bandwidth) to user-defined traffics, given by layer 2 (Ethernet), 3 (IP) or 4 (TCP/UDP) protocol header patterns. To this end, mechanisms defined on IEEE 802.3 and IEEE 802.1Q standards are adopted. Regular network sockets on software or AXI4-Stream on programmable logic are used to transfer mixed time-critical traffics to Network User layer applications. Traffic differentiation and prioritization is addressed through VLAN tagging (ID and Prio fields), conforming TSN streams. Traffics are dispatched according to the TSN stream priority, minimizing interferences from lower-priority traffics. To this end, a Time-Aware traffic Shaper (TAS) of the output bandwidth is adopting, enabling desired RT-QoS. This mechanism consists on a strict time-driven cyclic schedule. In order to assure coordination between TAS along the network, this component requires knowledge of the network timing, provided by the Network Timing Component.

The Network Timing Subsystem (NTS) offers network time synchronization to both the IP-cores present on the programmable logic and the processing system internal clock. To this end, IEEE 802.1AS standard-defined mechanisms, namely generalized Precision Time Protocol (gPTP), are implemented. The NTS collaborate on the Best Master Clock Algorithm, and therefore capable of electing the network time reference. Furthermore, the NTS provides time-aware bridge capability, as it can handle multiple interfaces to receive redundant time synchronization information and to redistribute time synchronization to neighbour peers. Protocol mechanisms and control of the local PTP Hardware Clock (PHC) entity is addressed through the gPTP software module. Sub-microsecond synchronization accuracy is provided thanks to the hardware TimeStamping Unit (TSU) present on every gPTP-capable interface. Network time synchronization is provided by the PHC to programmable logic, whereas a second-leap interrupt line is provided to synchronize internal processing system clock. Finally, a network user API allows configuration and runtime monitoring of the NTS.

5.1.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Hybrid communication data manager

Qualities:

The qualities that can be considered are link layer effective bandwidth, traffic differentiation and prioritization, and deterministic delivery for the most critical traffics. Traffic differentiation and prioritization are provided through VLAN encapsulation, conforming TSN streams. The deterministic delivery can be quantified through the end-to-end latency deviation for the most critical traffics, provided that both talker and listener nodes have knowledge of the network timing. Moreover, the link layer effective bandwidth, defined as the bit rate per second between user applications in absence of traffic interferences can be measured.

Set-Points:

With respect to Set-Points, TSN traffic manager IP is configurable at network deployment, attending to the application and network topology requirements. In particular:

- The traffic types object of differentiated RT-QoS should be defined by a combination of layer 2, layer 3 or layer 4 protocol headers.
- A priority is defined for each traffic type.
- A scheduling table should be defined for the Time-Aware traffic Shaper.

The scheduling table is a traffic priority-based arbitration scheme of the output bandwidth. Same-priority traffics compete for the same bandwidth resources and are dispatched following a FIFO scheme. The scheduling table determines which priorities are dispatched at a given time. Therefore, there is a trade-off between the guaranteed bandwidth for highest priorities and available bandwidth for the lowest traffics. In other words, deterministic deliveries are performed at the cost of available bandwidth for the rest of traffics.

Network timing subsystem

Qualities:

The qualities of the NTS are the synchronization accuracy, besides capabilities such as the synchronization of the processing system and the Best Master Clock Algorithm (BMCA), including the time-aware bridge capability.

The synchronization accuracy quality can be quantified through the comparison between Pulse-Per-Second signals generated from the PHC at different stations.

The Best Master Clock Algorithm can be evaluated considering each time-aware station configuration and the network topology. The BMCA should detect the time reference, determine the preferred path to the time reference, redundant paths to this time reference and the attached stations requiring retransmission of synchronization event messages. Furthermore, the synchronization of the Processing system can be verified comparing the internal clock and the current PHC time, which can be accessed through a memory-mapped register.

5.1.1.2 Design-time support

We used DT tools - Xilinx Vivado 2017.3 design suite for IP-core development, and GCC 7.3.0 for IP-core API programming.

5.1.1.3 Run-time support

Hybrid communication data manager

Monitoring:
Run-time monitored events: Per traffic priority congestion indicators. Network media status.
Programming API
Traffic types object of differentiation and prioritization, defined as a combination of protocol headers. Time-driven cyclic schedule for output bandwidth arbitration.

Network timing subsystem
Monitoring:
Run-time monitored events: Time synchronization drift, current network time reference (grandmaster), current path to grandmaster (pathTrace), the role of each gPTP-capable interface.
Programming API.
Time-aware station eligibility as grandMaster (standard-defined attributes). gPTP active interfaces, protocol message periodicity.

5.1.2 Related state of the art

Time Sensitive Networking is an update of IEEE Ethernet Standards for the implementation of synchronized, distributed control and real-time systems. Currently, TSN is applied on professional audio/video, automotive and industrial distributed systems.
Conventional Ethernet networks (i.e. 1000 Base-X, IEEE 802.3-1999) cannot be considered deterministic because of potential collisions and absence of traffic scheduling. On one hand, 1000 Base-X standard allows the physical medium to be shared by multiple stations (i.e. hub or bus network topologies). The Medium Access Control implements the CSMA/CD mechanism to avoid packet collisions by deferring transfers until the physical media is available. On the other hand, packets are dispatched following a FIFO scheme. To sum up, predictability or determinism for time-critical traffics cannot be guaranteed.
Existing deterministic Ethernet standards alter the 1000 Base-X physical and datalink layers or restrict the network architecture. Alternatives to TSN, such as EtherNet/IP or Ethernet PowerLink consist on application layers over 1000 Base-X and do not require specific hardware. However, network topology is restricted. Other alternatives, such as EtherCAT or PROFInet require specific hardware, thus restricting compatible devices. However, all of them provide optional synchronization support by means of specific IEEE 1588 profiles.

All these alternatives are vendor-locking and are based on protocol enhancements that are incompatible with each other. Conversely, TSN is an IEEE Ethernet standard. Therefore, TSN stations are fully interoperable with legacy Ethernet stations (i.e. 1000-Base-X) and some of the aforementioned protocols such as Ethernet/IP or PowerLink can take advantage of the RT-QoS provided by TSN. Furthermore, TSN is ready to support Industry 4.0 requirements, such as can communicate directly with is compatible with prior IEEE Ethernet standard releases, such as 1000-Base-X and is interoperable with the aforementioned protocols, such as Ethernet/IP or PowerLink can take advantage of RT-QoS provided by TSN at lower layers. Furthermore, TSN is able to support Industry 4.0 novel requirements, such as increased bandwidth and higher information transparency between lower and user application layers.

5.1.3 Extension within FitOptiVis

TRL@M0:

TRL-3 - “Experimental Proof of Concept”. At this moment, the VLAN module and TAS module are being implemented. VLAN tagging/untagging and prioritization attending to the protocol fields specified by the user. Time driven cyclic scheduling of the different traffic priorities per output channel. Seamless redundancy of user specified traffics. Frame pre-emption of user-defined priorities.

TRL@M36:

TRL-5 - “Technology validation in relevant environment”. VLAN and TAS module fully implemented. Besides, seamless redundancy and frame pre-emption functionalities will be added to the basic VLAN and TAS functionalities

5.1.4 Prospective adoption FitOptiVis UC

Use Case 5. Habit-tracking

Use Case 9. Surveillance of smart-grid critical infrastructure.

Other fields of application: Industrial IoT, automotive, avionics and any applications requiring networking able to manage low priority and deterministic traffics simultaneously, and/or time synchronization.

5.2 HSR implementation in RTU [SCHN]

In critical infrastructure, communications are gaining relevance and importance every day. The number of interconnected devices has increased, as well as the quality requirements, in which it is of paramount importance the capability to recover from communication failures in no time and in a transparent way for the other interconnected devices. In this environment, this High-availability Seamless Redundancy (HSR) component would help to meet those demands for Remote Terminal Units (RTU) in control systems used in electric substations.

5.2.1 Short description

This is supposed to be the communication component for the Surveillance of Smart grid critical infrastructure use case, being responsible for ensuring the exchange of information between different RTUs present in the smart grid network that allow HSR in the communications with other elements of the computing platform or implemented systems. The aim of this component is to allow communications in a more reliable way by using ring redundancy and enabling zero-time recovery against single point of failures. It involves both hardware and software elements.

Regarding hardware, it's necessary to select the physical interfaces through which the communication will take place (RJ45 Ethernet interface, Fiber optic, etc.). The RTU component includes the AM33xx Sitara microprocessors from the ARM Cortex-A8 family, and with their PRUs components will be possible to develop the HSR functionalities thanks to their specific features.

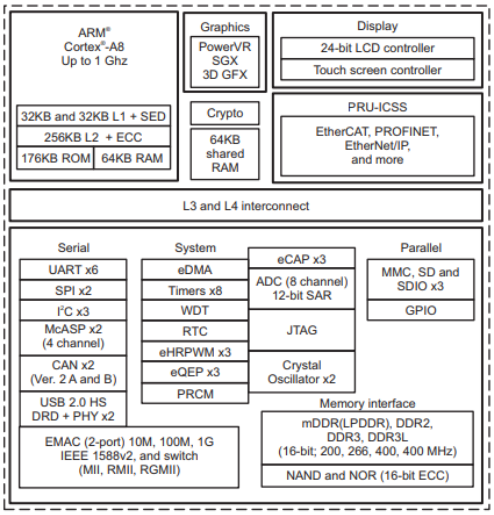

Figure 8: AM335x Functional Block Diagram [Ref: AM335x Sitara Processors Datasheet, Texas Instrument]

As per the software, it's necessary to select which protocols are used to achieve the information exchange over the physical interfaces already defined.

The system is designed to transmit all information through two communication paths in a ring topology, and to discard duplicated frames. Should there be any problem in one of the communication paths, only one frame would be delivered, and the system would be able to work without any problem and no delays or packets loss due to single point failures. In contrast, other ring redundancy methods would block one path, and would only use the primary one, dropping some packets until recovered from the fault.

Input (inputs to be received by the HSR component)
Incoming communication with smart grid frames as per HSR communication specification conforming a ring.
Output (outputs to be produced by the HSR component)
Outgoing communication with smart grid frames as per HSR communication specification conforming a ring.
Communication (signals or data packets to be exchanged with the HSR component, i.e., inputs and outputs)
Status/commands of binary signals (ON/OFF). Value of analogue registers or set-points. Any other calculated value through PLC configuration (transformer overload, number of switching operations, etc.)

5.2.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Qualities:

- Quality of Service, QoS, equal to a standard Ethernet communication:
    - Transmission delay. Delay based on the data-rate of the link.
    - Availability. Probability that a network is operational.
    - Jitter. Variance of the time delay between data packets over a network.
    - Packet loss. Percentage of the packets that fail to reach their destination with respect to the all packets sent.
    - Bit rate. Number of bits processed in a system per unit of time.

    - Throughput. Amount of data moved successfully from one point of the network to another in a period.
    - Latency. Sum of all possible time delays in a network.
- In addition, by HSR communication, we would have the following qualities:
    - Convergence time when a node, link or switch fails.
    - Number of failure points allowed

Set-Points:

- Physical interface in the ring configuration
- HSR RTU Configuration

Available model:

- HSR standard IEC 62439-3. High-availability Seamless Redundancy is based on the Ethernet or Fiber Optic technology. This network protocol provides seamless recovery against single failure of a bridge in the network.

5.2.1.2 Design-time support

- Specific development tools for Remote Terminal Units.
- Arago Texas Instrument and Yocto Project to generate Linux Kernel.
- Concept HDL and Pin Muxing Texas Instrument for schematic design.
- Allegro for PCB routing.

5.2.1.3 Run-time support

Monitoring:

- It will be monitored that messages are delivered within an accepted latency, and that every time there is a failure in any of the communication links, the information can be transmitted successfully through the other communication path.
- Hit/miss messages.
- The RTU with HSR technology installed in the Ring topology Configuration are provided with a web server that allow to monitor the variables involved in the communications between the Smart Grid equipment and the Surveillance equipment through the TSN network.

5.2.2 Related state of the art

For critical infrastructure and high-performance networks, the interruption in the communications, even for minimal times, are unacceptable. Ethernet is a technology with broadcast functionality but it could happen that there was only one active link between the source and the destination. So, to have redundancy we would need additional links to compensate and a redundancy control protocol to administrate the links.
A great number of redundancy control protocol exists. A few of them are defined as standards, such as:

- Media Redundancy Protocol MRP (IEC 62439-2)
- Parallel Redundancy Protocol PRP (IEC 62439-3)
- High-availability Seamless Redundancy HSR (IEC 62439-3)
- Cross-network Redundancy Protocol CRP (IEC 62439-4)
- Beacon Redundancy Protocol BRP (IEC 62439-5)
- Distributed Redundancy Protocol DRP (IEC 62439-6)

- Ring-based Redundancy Protocol RRP (IEC 62439-7)

The selection of redundancy standards for different smart grid applications would depend on the requirements of those applications.

5.2.3 Extension within FitOptiVis

TRL at M0:
TRL2 - There is a technology concept formulation about how HSR should be implemented in a RTU and how it can help to improve the reliability of control systems.
TRL at M36:
TRL6 - At the end of the project is expected to have implemented HSR in a RTU, integrated in the surveillance of smart-grid critical infrastructure use case

5.2.4 Prospective adoption FitOptiVis UC

The HSR technology will be used in surveillance of smart-grid critical infrastructure use case. The HSR brings reality to the use case making it closer to the real environments used in the smart-grid.

Furthermore, the most recent equipment that are being installed in some parts of the smart-grid try to reinforce availability of the communications due to the criticality of the data that are exchanged, using redundancy protocols and structures on different occasions. With this concept, HSR gives the use case a high availability for the critical data exchanged between the Surveillance system and the RTUs that manage a part of the smart-grid. In addition, a TSN network will be included between the RTUs and the Surveillance system to be able to assign priorities on the network traffic depending on its criticality.

5.2.5 Related documents

- IEC 62439-3:2016. Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR).
- Industrial Communication Protocols Supported on Sitara.
- PRU Assembly Language Tools Use´s Guide.

5.3 Fletcher big data communication interface [TUD]

FPGA accelerators are becoming commonplace accelerator solutions. Interfacing efficiently with FPGA accelerators, however, still remains a challenge. This is especially true for applications written in high-level languages that want to make use of FPGA accelerators. Such applications are often developed on top of language-specific interpreters, virtual machines, or compilers. These language specific tools are often designed for ease-of-use/abstraction, control-heavy or computational workloads, but not for data-intensive workloads and application interoperability with accelerators. Therefore, the data containers are often sub-optimal w.r.t. integration with FPGA accelerators, and require a significant serialization step in preparing the data to be workable by FPGA.

5.3.1 Short description

Fletcher is an FPGA acceleration framework targeting data-intensive applications. The Apache Arrow project defines a smarter, data-centric container format for 11 different languages. For this format, Fletcher can generate highly efficient hardware structures that interface with the Arrow data containers. This allows FPGA accelerators to efficiently integrate with high-level languages, unlocking end-to-end accelerator-application interface throughput limited by system bandwidth only.

Target:
FPGA
Input:
Typically big data collections from storage, memory or network
Output:
Typically reduced or filtered data

5.3.1.1 Qualities, tunable set-points and available design-time and run-time parameters

The are a number of qualities that can be used as parameters for interfaces created by Fletcher. These parameters, and their specific pre-defined set-points are as follows

1. Bit width: 2, 4, 8, 16, .., 2^n bits
2. Data types: strings, lists of integers, lists of lists

5.3.1.2 Design-time support

Vendor agnostic component library in VHDL

- Structure generation tool (Fletchgen) - generates vendor agnostic components.
- Fletcher designs
  - Simulation verified in
    - GHDL (open source, free)
    - Xilinx XSIM
    - Mentor Graphics QuestaSim
  - Synthesis verified in
    - Xilinx Vivado
    - Intel Quartus

5.3.1.3 Run-time support

- Software:
  - C++ run-time library
  - Python run-time library
  - Easily extendable to 11 other languages through Apache Arrow.
- Hardware:
  - Amazon EC2 F1 platform
  - OpenPOWER CAPI SNAP FPGA accelerator cards
  - Xilinx Alveo platform

5.3.2 Related state of the art

At the moment, there are no systems that allow automatic generation of data communication interfaces on FPGA hardware.

5.3.3 Extension within FitOptiVis

TRL@M0:
The TRL level at the beginning of the project is at technology development stage (TRL4).

TRL@M36:
During the project, we will extend Fletcher by creating a demonstrator for its big data communication capabilities (TRL6). Then this demonstrator will be used to create systems to use Fletcher (TRL7). By the end of the project (M36), these systems will be tested and made operational in practice (TRL9).

5.3.4 Prospective adoption FitOptiVis UC

The Fletcher data communication interface generation system represents an important component for big data applications that can benefit from integrating FPGA hardware accelerators to improve the performance of their kernels. Therefore, there are a number of possible commercial opportunities where Fletcher can capture a commercial market share. One commercial opportunity is to use Fletcher as a technology component to integrate into existing hardware accelerator solutions. This allows Fletcher to be commercialized as a service solution for other solutions in the field. A second commercial opportunity is to provide dedicated big data FPGA accelerated solutions for various computationally intensive kernels.

The Fletcher big data communication interface will be discussed with various FitOptiVis project partners in order to identify its potential for accelerating the use cases developed in the project.

5.3.5 Related documents

[1] https://github.com/johanpel/fletcher
[2] Peltenburg J., van Straten J., Brobbel M., Hofstee H.P., Al-Ars Z., "Supporting Columnar In-memory Formats on FPGA: The Hardware Design of Fletcher for Apache Arrow". In Applied Reconfigurable Computing (ARC), 2019. Lecture Notes in Computer Science, vol 11444.

5.4 Multi-sensor gateway [ABI]

The multi-sensor gateway, through an embedded Linux distribution will ensure a secure, safe and cooperative embedded system integrated in a system-of-systems environment.

5.4.1 Short description

The multi-sensor gateway will allow the runtime configurable multi-context support modifying data priority transmission upon the scenario requirements. It will be the orchestration actor of the heterogeneous network of sensors and actuators (valves, temperature, alarms, liquid and gas sensors), Drones and completely integrated in a distributed video processing pipeline.

Input:
The gateway will process inputs from smart cameras (a subset of Video Content Analysis and Face Recognition) and drone.
Output:
The gateway will provide output to smart cameras, PLC and other actuators and sensors.

The basic architecture of system with interfaces can be depicted in the figure below.

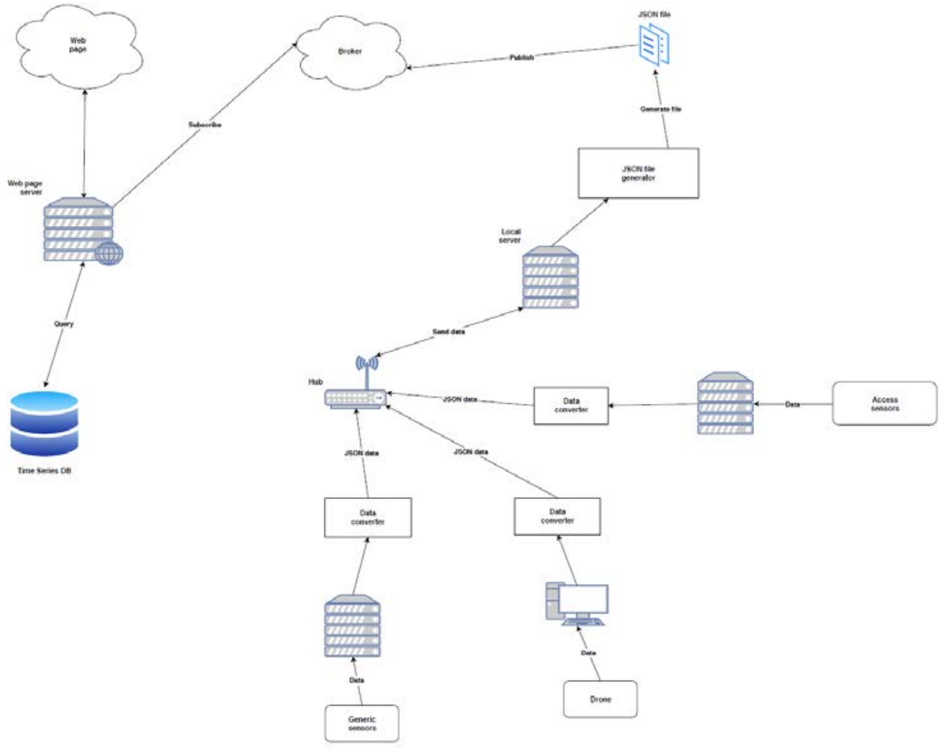

Figure 9: Multi-Sensor gateway architecture.

In order to monitor sensor data and set up the configuration of the communication system a dedicated dashboard will be developed

5.4.1.1 Qualities, tunable set-points and available design-time and run-time parameters

There are no specific predefined set-points for the gateway.

5.4.1.2 Design-time support

The multisensory gateway is not meant to be used to handle components design, analysis, verification and deployments.

5.4.1.3 Run-time support

At run time, the multi-sensor gateway will orchestrate eventually the re-configuration of the network. Upon reception of a specific trigger or event, the gate will send a sequence of commands to adjust the system configuration for the specific case (for example, sampling frequencies of data image, switch on a different set of cameras etc). It is important to note that the gateway is not mean to manage the video streams, but data already processed by something between the camera and the gateway, for example: detected unidentified person, detected employee, detected suspicious activities.
The dashboard is meant to be the entry point in which to configure the reactions of the gateway according to the inputs, for example, a camera detects a movement and the

gateway requires the camera and the other sensors to increase sampling frequency / frame rate / speed etc. and/ or launch an alarm, send email, call the police, etc.

5.4.2 Related state of the art

There is no investigation of the state-of-the-art done for this component.

5.4.3 Extension within FitOptiVis

TRL@M0:
The multisensory gateway will be completely developed in the FitOptiVis project. Some proof of concept with the basic function has demonstrated the potential of the proposed gateway and the dashboard. At M0 of the project, the TRL level of the component is 4.
TRL@M36:
At M36, we expect to test the final prototype in a realistic environment (water supply demonstrator) at TRL 6-7

5.4.4 Prospective adoption

The multi-sensor gateway will support the scenario needs in the water supply demonstrator guarantee the communication aspects with a customizing of the multi-sensor gateway.

6 Miscellaneous (Sensor/actuators/composed-composed)

This section describes components other than HW, SW or communication.

6.1 Wireless LED SYNC light [UWB, REX]

6.1.1 Short description

The wirelessly synchronized LED markers belongs to the actuator class of components. It is a distributed compound HW+FW solution consisting of a set of LED markers (MCU-based devices) bound to the central FPGA/SoC/MCU/CPU based platform over wireless link. The marker's flash synchronized with camera shutter allows to take 2 images of the scene - one with all the markers off and second one with the selected marker on. Differential image of these 2 captures allows fast, exact and robust identification and localization of markers in the image.

Input:

- LED timing definition table
- Definition table for camera capture timing
- Camera strobe pin specification

Output:

- Flashing of spatially scattered LED markers according to the definition table
- Camera STROBE pin driven according to camera capture timing table and in synchronization with the LED markers flashes

6.1.1.1 Qualities, tunable set-points and available design-time and run-time parameters

Timing precision throughout the complete distributed system (RF LED markers + central controller) is expected to be below 10 µs. It is more than sufficient with respect to the ordinary exposition times (~5ms).

The component's working space depends on RF range of the wireless circuitry, which goes in hand with RF TX power and antenna construction. It will be dimensioned for circa 10 m range.

The component does not require any design-time parameters. As run-time parameters serve input arguments listed in the previous paragraph.

6.1.1.2 Design-time support

The wireless LED markers are based on Silicon Labs EFR32 SoC. The GNU C++ compiler together with Gecko SDK (C/C++ library for EFR32 MCUs) is used to implement the markers' functionality.

The controller-side part of the component is to be designed and implemented in Microsoft Visual Studio and compiled with the provided MS C++ compiler or with GNU C++ compiler (for ARM based HW platforms).

Additionally, the used REX block SDK provides framework and base class implementation for the component. Only functional parts of the component have to be implemented - functions Init(), Exec() and Exit() (called upon system initialization, periodically during system run and upon system shutdown).

6.1.1.3 Run-time support

In run-time, the module is configured and observed via configuration window accessible through RexDraw (newly re-branded to REXYGEN Studio) - GUI based application that configures and inspects configuration of the real-time control application.
Component configuration (inputs) can be inspected and adjusted here. Component status information is available at the same place. It includes RF packet reception details (signal strength, signal quality) and an aggregated flag indicating synchronization loss (no signal for given period).

6.1.2 Related state of the art

Currently there is no system similar to the proposed component available.
Core part of the component is wireless time synchronization. Products allowing wireless time synchronization with the stated accuracy are rare as well. The typical NTP accuracy is about 1 ms on LAN (we speak about wired Ethernet, Wi-Fi is even worse) and thus far away from the requested precision. Better performance advertise proprietary solutions from several vendors - for example EchoRing from R3 Communications (tens of µs) or V-MON from Inertia Technology (100 ns). However, these solutions are not focused on image processing. They target on process control and vibration monitoring and time synchronization is only minor part of their functionality.
An alternative for the intended use of the component can be motion capture system from Vicon. It is based on passive motion capture markers; therefore, point registration has to be performed. There exist also motion capture systems that use active LED markers. Here, however, either the LED color identifies the marker (Biosense Medical) or the LEDs are lighted on sequentially (Qualisys), no wireless synchronization with the camera shutter is performed. Also, the price excludes the above listed motion capture systems from common use.

6.1.3 Extension within FitOptiVis

TRL at M0:
TRL 0.
TRL at M36:
UWB expects to deliver designed and fully implemented component that is tested in lab as well as in controlled industrial environment. TRL 5 - 6

6.1.4 Prospective adoption FitOptiVis UC

The component will be used within the FitOptiVis demo pool as one of the principal components in the robot calibration use case (UC8).
Outside the project scope, it can be used in any application where object tracking or localization is required. It can be used both for 2D as well as 3D localization tasks. A big advantage of this method is high robustness and possibility to exclude point registration procedures from the processing chain.

6.1.5 Related documents

[1] „Getting started with REXYGEN and Raspberry Pi“, REX Controls s.r.o., https://www.rexygen.com/doc/ENGLISH/MANUALS/RexygenGettingStarted_RasPi/RexygenGettingStarted_RasPi_ENG.html
[2] „REXYGEN Studio User Guide“, REX Controls s.r.o., https://www.rexygen.com/doc/ENGLISH/MANUALS/RexygenStudio/RexygenStudio_ENG.html
[3] „Network Time Protocol“, https://en.wikipedia.org/wiki/Network_Time_Protocol
[4] http://www.echoring.com

[5] Inertia Technology: „V-MON 4000“, http://inertia-technology.com/product/vibration-monitoring-vmon-4000/
[6] https://www.vicon.com/
[7] Biosense Medical: „Active Marker Motion Capture“, https://biosensemedical.com/active-marker-motion-capture/
[8] Qualisys: „Short range active marker“, https://www.qualisys.com/hardware/accessories/active-markers/short-range-active-marker/

6.2 Edge Capturer [ITI]

6.2.1 Short description

The Edge Capturer component is an abstraction that encapsulates an arbitrary number of cameras, the low power execution boards attached to each one of those cameras, and the software running on those boards, including a work dispatcher.
Managing all these subcomponents as a single one facilitates its management in the system architecture. In this way, the Edge Capturer is treated as a sole component that is in charge of capturing, pre-processing, segmenting, and transferring the images that the other elements of a system will process. The architecture of the Edge Capturer is depicted in the figure below.

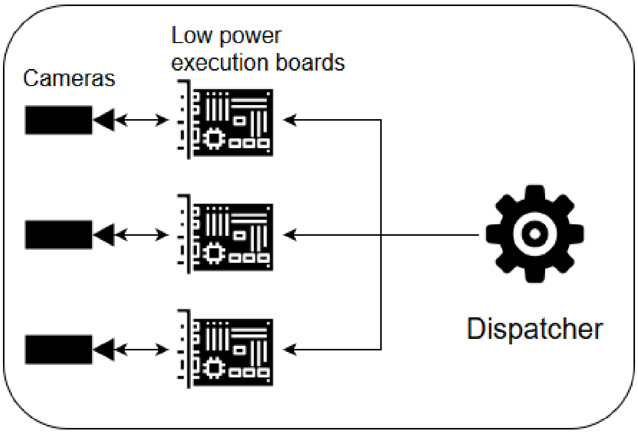

Figure 10: Edge Capturer architecture

The pipeline of the captured images is as follows. The Edge Capturer receives a request for a new capture. It then triggers the cameras so all of them take an image at the same time (given some jitter). Each camera transfers the captured image to the attached execution board. These execution boards are the first computational layer in the system and are installed as close to the cameras as possible. Each board pre-processes the image (undistortion, debayering, and so on) and then segments it to find eg. the Region Of Interest (ROI), which is the smallest rectangular region that contains the object of interest in the image, and to separate background and object of interest. If at this point a board detects an incorrect capture, it signals the other boards and a new capture of the same object is retaken. The most common incorrect capture happens when the entirety of the object is not inside the field of view of the camera. Once the image is pre-processed and segmented, each execution board transfers the image to the agent who requested the capture.

To reduce costs and exploit the distributed qualities of the Edge Capturer, this component uses low cost electronic boards (e.g. Raspberry Pi or Expressobin) which are installed close to the cameras.

6.2.1.1 Qualities, tunable set-points and available design-time and run-time parameters

With respect to qualities, we employ three metrics: throughput, latency, and bandwidth usage.

- Throughput is measured counting the number of captures that are processed by the component in a unit of time.
- Latency is the time that the system takes to process a new capture.
- Finally, the last metric is bandwidth usage, which is the quantity of Mbits per second that the communications from the Edge Capturer to the rest of the system use.

Several parameters can be adjusted for the Edge Capturer during design and run-time. In design-time, the main parameter is the number of cameras, however it is not planned to investigate this within the scope of the FitOptiVis project. Regarding runtime support, the Edge Capturer offers a workload dispatcher functionality.

Traditionally, cameras are connected to a single computing node, in this way, even when that node is not responsible for processing the images captured by the cameras; all the images have to travel through it, increasing bandwidth usage. With the dispatcher, the images are sent to the exact node that will continue processing the images. Moreover, since the Edge Capturer has to deal with a wide variety of objects and materials, the segmentation algorithm is highly adaptable.

Further, if several segmentation attempts for a single part fail, the Edge Capturer can try different segmentation settings (which are predefined by the operator) to obtain a valid image. Finally, the output can also be adapted to the task at hand. If the colour and texture qualities of the object are irrelevant for the task, the output of the Edge Capturer can be reduced to the silhouette of the object in a binary image. This reduces even further the bandwidth usage.

6.2.1.2 Design-time support

In order to develop this component, general design and development tools are used to implement the code.

6.2.1.3 Run-time support

Runtime Adaptivity:

At run-time, when an object is incorrectly segmented employing the default segmentation settings, the Edge Capturer is capable of employing several predefined segmentation configurations to adapt to the characteristics of that object. Moreover, the Edge Capturer can also detect incorrect captures (e.g., the object was not positioned inside the field of view of the cameras) and take a new capture without external output.

Monitoring:

We plan to monitor performance at run time, for doing that a set of candidate components will be selected, those artefacts will enable the run-time support, so parameters can be changed on the fly. The following metrics will be monitored:

- Throughput
- Average bandwidth usage
- Correct / incorrect capture ratio
- Average processing time per capture.

To optimize the hardware capabilities of the execution boards as much as possible, the pre-process and segmentation algorithms are implemented in C using the following APIs: OpenCV, OpenMP, ZMQ, and Aravis.

6.2.2 Related state of the art

Currently, in industrial computer vision solutions, the pre-processing and segmentation of an image is usually performed in a computational node that is not close to the camera. Moreover, this computational node is the same that performs the rest of the image processing, for example, analysis of the image contents and classification of the detected object.

As explained by Prajapati and Sanjay [PRAJAPATI11], low-level image processing (as segmentation) can be divided into several computation resources to reduce computation time. Moreover, as shown in [WU05], distributed image processing can reduce energy consumption. Based on these ideas, we propose the Edge Capturer component, which aims at addressing these two challenges (computation time and energy consumption) by means of image processing distribution on low-power execution boards.

As pointed out by Shi et al. [SHI16], in a system where several images taken by a number of devices have to travel to a single node, an edge-computing approach can reduce latency and bandwidth usage, while increasing throughput. Even though a wide number of applications of edge computing on image processing are focused on mobile devices [MAO2017], we believe that an edge computing approach can also benefit industrial applications. Therefore, Edge Capturer takes advantage of edge computing by locating the execution boards close to the cameras. In this way, the first computation layer is as close to the sensor as possible.

6.2.3 Extension within FitOptiVis

Edge capturer will be completely developed and tested within the project. It is expected to take advantage of the proposed edge architecture.

These are the indicators expected to be achieved:

- Current bandwidth usage is 28.7Mbytes per capture (using 16 cameras and one central node, no dispatcher), it is expected less than 9Mbytes per capture.
- Current segmentation time of capture made up by 16 images (there are 16 images in the UC where the Edge Capturer will be integrated) is 735 milliseconds, it is expected that the Edge Capturer with each board segmenting an image in parallel will take less than 200 milliseconds.
- In qualitative terms, we expect that the Edge Capturer, with its adaptive segmentation capabilities, generates less incorrect captures due to segmentation problems.

In terms of TRL, these are the levels at the beginning and end of the project:

TRL at M0

TRL2 - At M0, only the potential application of the Edge Capturer is validated based on the current capturing method employed in Zero Gravity (one product where Edge Capturer can be embedded). However, the specific execution boards are not decided and their software is not developed.

TRL at M36

TRL4/5 - At M36, we expect a tested component that can quickly deliver pre-processed and segmented images to the worker agents, synchronize all the cameras and boards, and deal with incorrect captures.

6.2.4 Prospective adoption FitOptiVis UC

In the context of FitOptiVis, the edge capturer is integrated into the 3D industrial inspection system use case.

The Edge Capturer can be applied to any system that has several capture devices (cameras) that can benefit from obtaining those images already pre-processed and segmented. For example, robot calibration and autonomous driving vehicles would be two candidate application areas where it can fit.

6.2.5 Related documents

[PRAJAPATI11] H. B. Prajapati and S. K. Vij, "Analytical study of parallel and distributed image processing," in International Conference on Image Information Processing, Shimla, 2011, pp. 1-6.

[WU05] H. Wu, A. Abouzeid, “Energy efficient distributed image compression in resource-constrained multihop wireless networks” in Computer communications, vol. 28, no. 14, pp. 1658-1668, 2005

[SHI16] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," in IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, 2016.

[MAO2017] Y. Mao, C. You, J. Zhang, K. Huang and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," in IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2322-2358, Fourthquarter 2017.

Furthermore, the development is a candidate technology to be embedded into the Zero Gravity 3D industrial inspection system (https://www.zerogravity3d.com/).

7 Analysis and conclusions

This report presented the components to be developed by the FitOptiVis partners in WP5. All 22 partners of WP5 contributed a total of 28 of their respective component developments. The components were classified into 4 different categories as summarized in the list below.

Sec 3. Processing and acceleration components
3.1. TUT: Customized parallel soft-cores
3.2. UTIA: Optimized HW/SW cores
3.3. UniCA: NEURAghe
3.4. UnivAQ: AIPHS
3.5. TASE: Image collection interface
3.6. TASE: Space image processing chain
3.7. TASE: Image transmission interface
3.8. CAMEA: Licence plate detector
3.9. UNISS, UNICA: MDC-generated coarse grained reconfigurable HW accelerators
3.10. UTU: Low energy dynamic voltage and frequency scaling
3.11. PHL: Multistream video image scaler and compositor
3.12. FUT: Efficient magnetic field generators

Sec 4. Software components
4.1. UGR: Behavior classifier
4.2. UGR: Person tracking software
4.3. HURJA: Salmi AR Platform
4.4. UnivAQ: Distributed Video Coding Solutions
4.5. Nokia: Virtual Reality Demonstrator
4.6. UWB, REX: Point localization from stereovision
4.7. Aitek: Video Content Analysis (VCA) platform
4.8. UNISS: Face and behavioural analysis (FBA) platform
4.9. HIB: Pose Estimation Engine
4.10. UC: Reconfigurable video capture and processing

Sec 5. Communication-oriented components
5.1. 7SOLS: Time Sensitive Networking
5.2. SCHN: Communication Element for Smart Grid Surveillance
5.3. TUD: Fletcher big data communication interface
5.4. ABI: Multi-sensor gateway

Sec 6. Miscellaneous (Sensor/actuators/composed-composed)
6.1. UWB, REX: Wireless LED SYNC light
6.2. ITI: Edge Capturer

This deliverable shows that the components developed in FitOptiVis cover all the use cases targeted by the project. The table below lists the use case coverage of the components discussed in this report. The table shows that for each project use case, there is at least one component developed in the project. Some components, on the other hand, cover more than one use case in the project, indicating project collaboration.

UC	HW IPs	SW IPs	Comm. IPs	Others	Total
Water supply	3.3 UniCA 3.4 UnivAQ 3.9 UNISS, UNICA	4.4 UnivAQ 4.7 Aitek 4.8 UNISS	5.4 ABI		8
3D inspection	3.10 UTU			6.2 ITI	2
Virtual reality	3.1 TUT	4.5 Nokia			2
Habit tracking		4.1 UGR 4.2 UGR 4.3 HURJA 4.9 HIB	5.1 7SOLS		5
Road traffic surveillance	3.8 CAMEA				1
Multi-source streaming composition	3.11 PHL				1
Sustainable MRI	3.12 FUT				1
Robot calibration	3.2 UTIA	4.6 UWB		6.1 UWB	3
Surveillance of smart-grid critical infrastructures		4.1 UGR 4.2 UGR	5.1 7SOLS 5.2 SCHN		4
Autonomous exploration	3.2 UTIA 3.5 TASE 3.6 TASE 3.7 TASE	4.10 UC			5