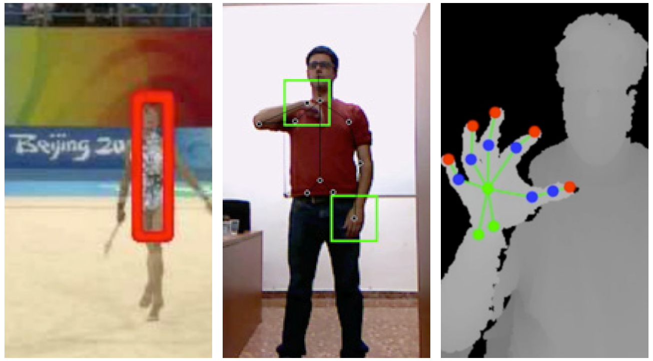
Characteristics

The software platform consists of three parts:

- The 'Starling' software, which allows to implement vision algorithms graphically, without programming, and to obtain a functional application within a few minutes. The generated code can be edited and modified.
- An SVN which regroups the code of all modules of the platform.
- A Wiki which describes the role of each module and its usage.

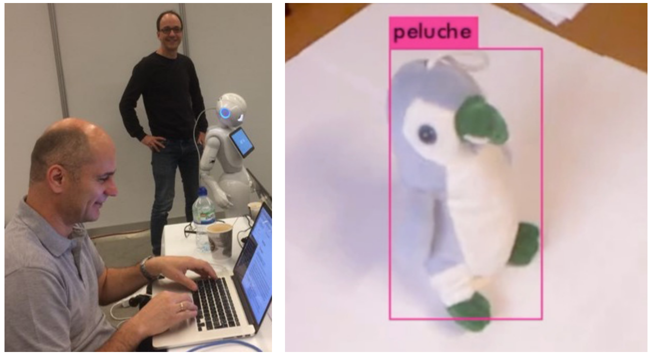
The hardware infrastructure includes:

- Three mobile robots, each equipped with an RGB-D camera, odometers, distance and contact sensors, and Android tablets to control them remotely.
- An «Optitrack» motion capture system based on 8 infrared cameras plus cabling and connection hardware.
- A GPU computing cluster with 16 Nvidia Titan-X GPUs distributed on 4 compute vertices and a file server, run by job scheduling software.

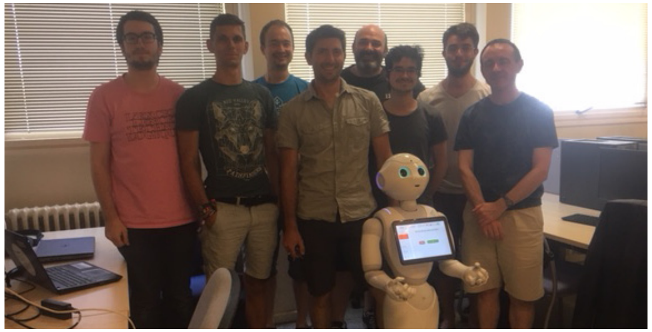


Examples for Software Modules

LIRIS-VISION is part of the software of the EMOX Robot by Awabot (here at the Nvidia Stand at GTC Europe).



LIRIS-VISION is part of the software platform of the LyonTech team participating in the finals of the Robocup @ Home 2018 competition in Montreal and 2019 in Sidney (together with Inria, CITI, CPE)

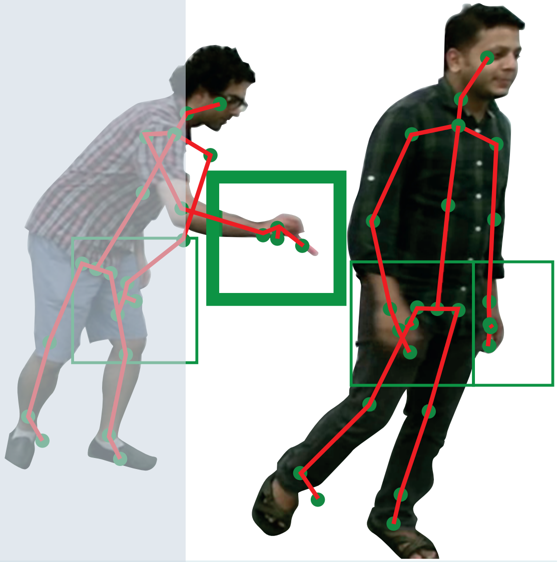


Contacts

Christian WOLF
Christian.Wolf@liris.cnrs.fr
Eric Lombardi
Eric.Lombardi@liris.cnrs.fr

LIRIS-VISION

Library and tools for vision applications



LIRIS

Robotics software platform

- Easy access to all robot sensors (color and depth, odometers, distance sensors, etc.)
- API in C ++ allowing very easy integration of new recognition or detection modules.
- Import of modules from the Starling software
- Simulation mode: development without physical access to the robot hardware
- Robot control in via WIFI
- Execution of vision algorithms on the robot's CPU/GPU or remote mode, ie. on a remote PC from videos transmitted over WIFI or wired connection.

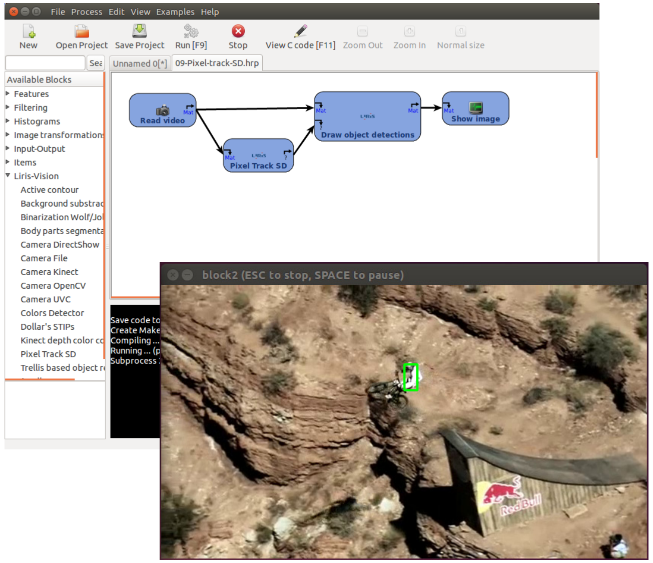
Motion Capture Platform

Fleet of mobile robots

GPU Computing Cluster

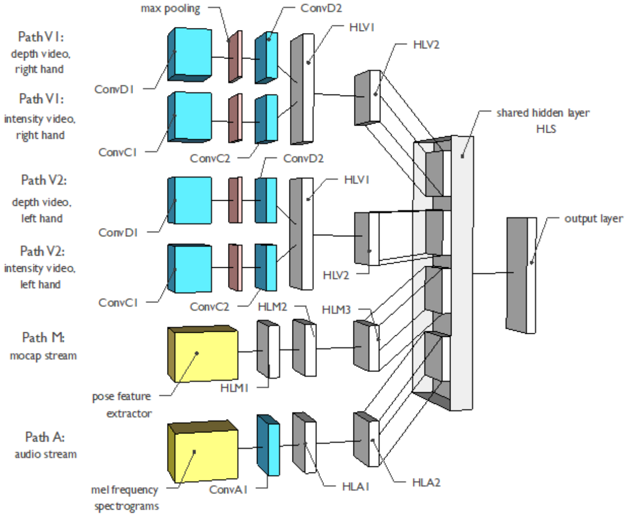
Starling

Starling allows to implement vision algorithms graphically, without programming. It is very simple to add new modules to STARLING: any algorithm developed in C ++ with OpenCV can be integrated by adding a configuration file. The Starling interface then makes it possible to connect the various modules together in order to obtain a more complex module and a functional application within a few minutes. This application can be executed directly, but Starling also allows to edit its code and modify it. The figure on the right gives an overview of the capabilities of the software.

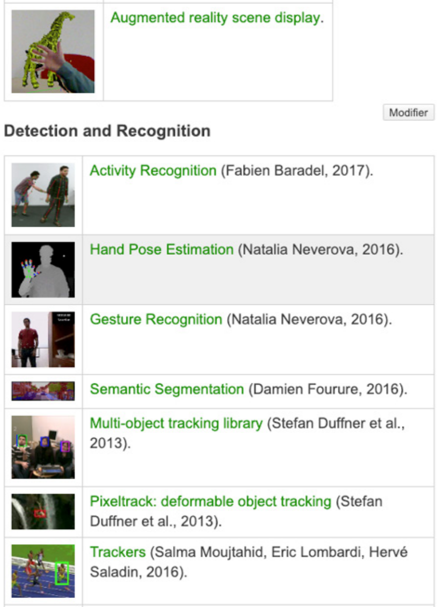


```
Save code to
Create Make
Compiling ...
Running ... (p
Subprocess
```

Example of a Starling application: detector / tracking of moving objects implemented by connecting 4 blocks: video capture, detection, bounding box drawing and display.



Methodologies: Deep Learning and structured models



	Augmented reality scene display.

Modifier

Detection and Recognition

	Activity Recognition (Fabien Baradel, 2017).
	Hand Pose Estimation (Natalia Neverova, 2016).
	Gesture Recognition (Natalia Neverova, 2016).
	Semantic Segmentation (Damien Fourure, 2016).
	Multi-object tracking library (Stefan Duffner et al., 2013).
	Pixeltrack: deformable object tracking (Stefan Duffner et al., 2013).
	Trackers (Salma Moujtahid, Eric Lombardi, Hervé Saladin, 2016).

A collection of modules designed and implemented by the team members

Platform website

http://liris.cnrs.fr/liris-vision/