MORSE, the essential ingredient to bring your robot to real life

Gilberto Echeverría
gechever@laas.fr

Laboratoire d'Analyse et d'Architecture des Systèmes
Toulouse, France

April 15, 2011

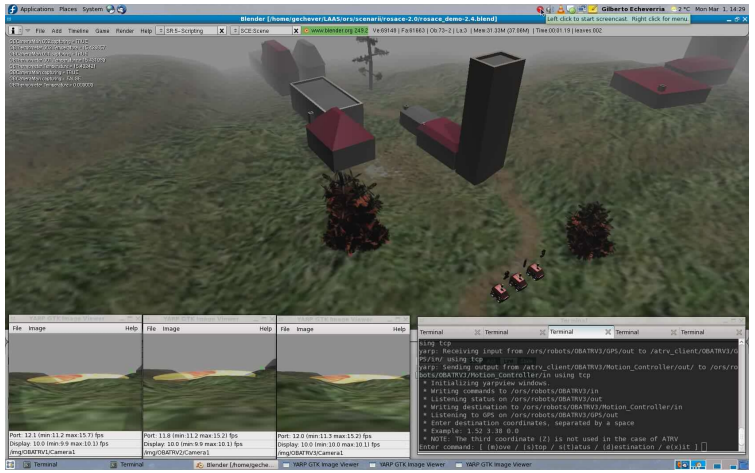
Review of MORSE

- Project started in August 2009
- General system robotics simulator
- Based on the Blender 3D software
- Modular architecture
- Middleware independent
- Multiple heterogeneous robot systems
- Communication and multi-agent simulation

Modular
Open
Robots
Simulation
Engine

March 2010 version

- Prototype version
- Restricted architecture
- Scripts stored inside Blender files
- Tightly linked with YARP
- No other middlewares usable

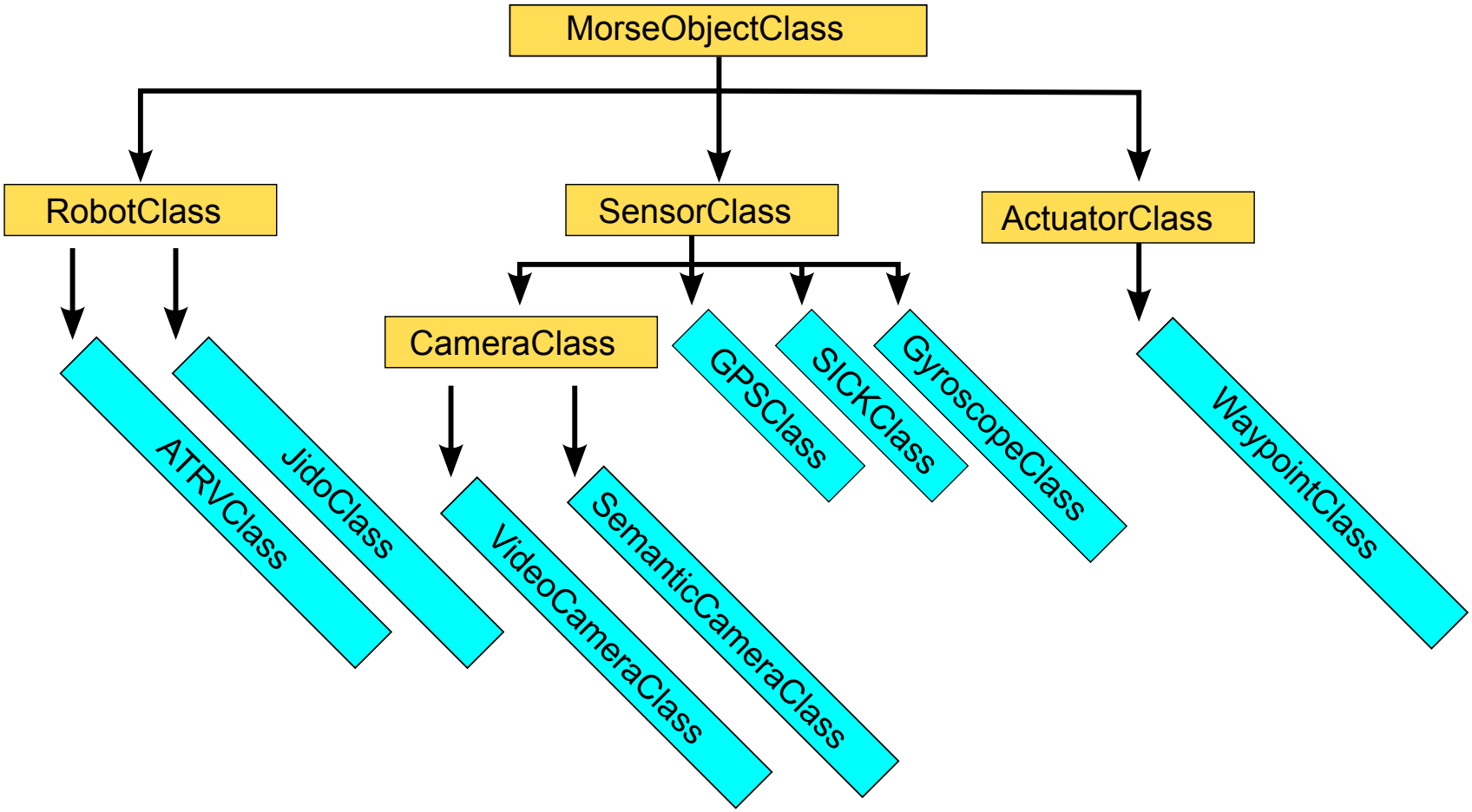


Outline

Main architecture principles

- Library of individual components
- Components consist of Python and Blender files
- Object oriented scripts for each component
- Robotic components are completely middleware independent
- Middlewares and modifiers are additional modules
- Completely extensible with plug-in methods
- Configurable binding of modules and middlewares

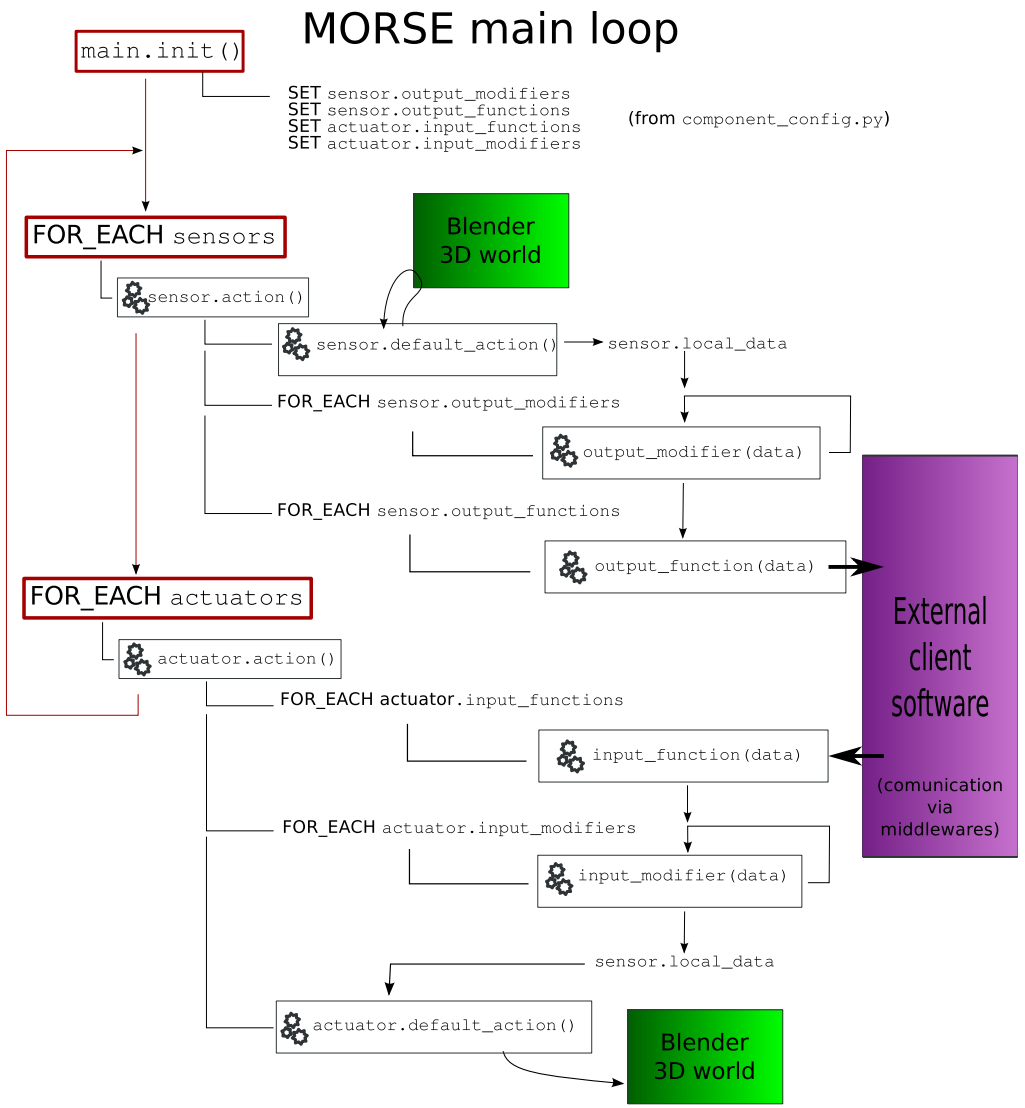
Object class categories

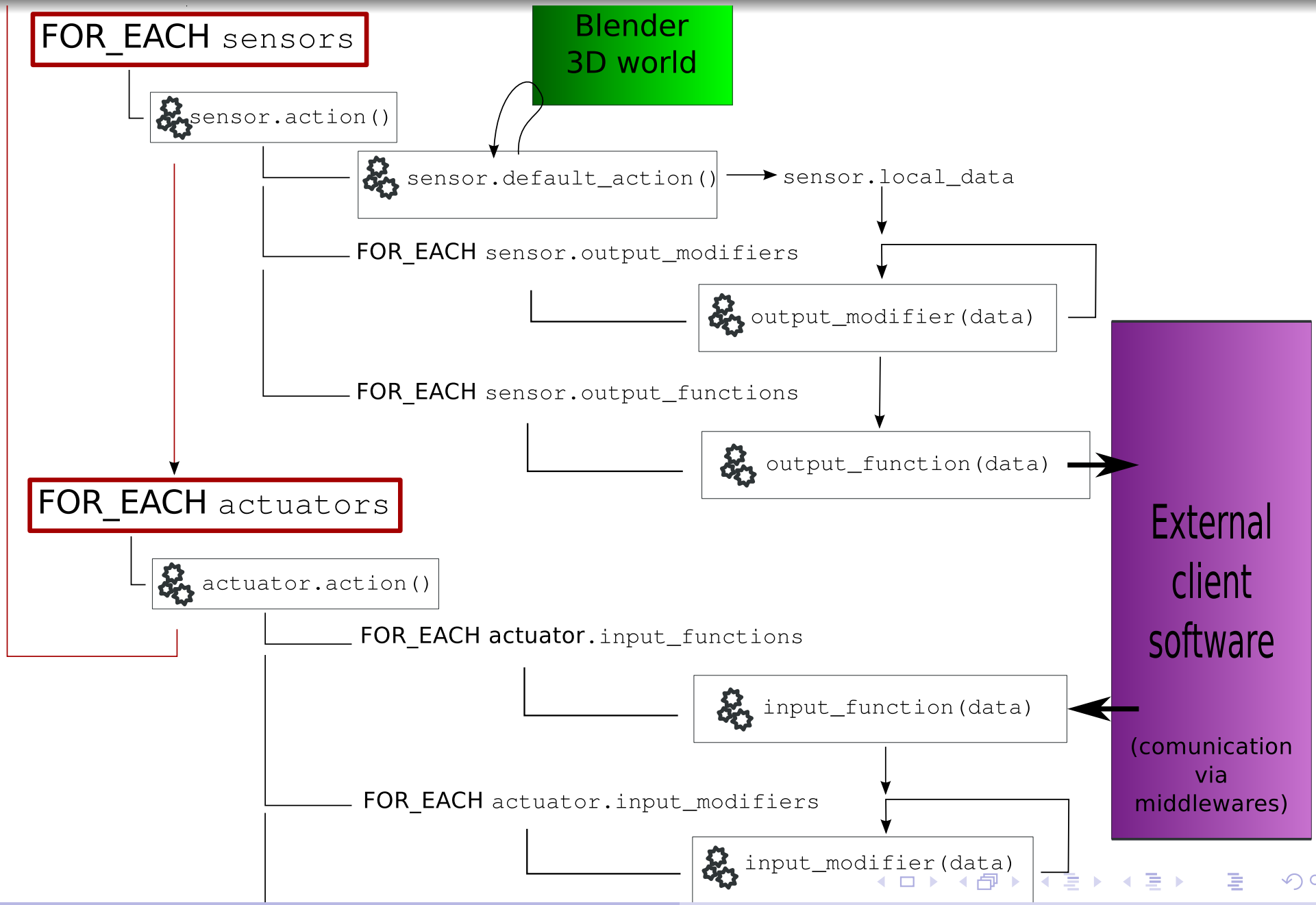


Middlewares and modifiers

- Middleware modules provide bindings for MORSE components to communicate the data they use with the outside world
- Multiple middlewares can be used in the same simulation
- They encapsulate data in the format required

- Modifier modules alter the data of a sensor/actuator before it is exported outside the simulator.
- Simulated data is "perfect", real data is imprecise and noisy
- Modify the reference frame, scale, units, etc. and noise

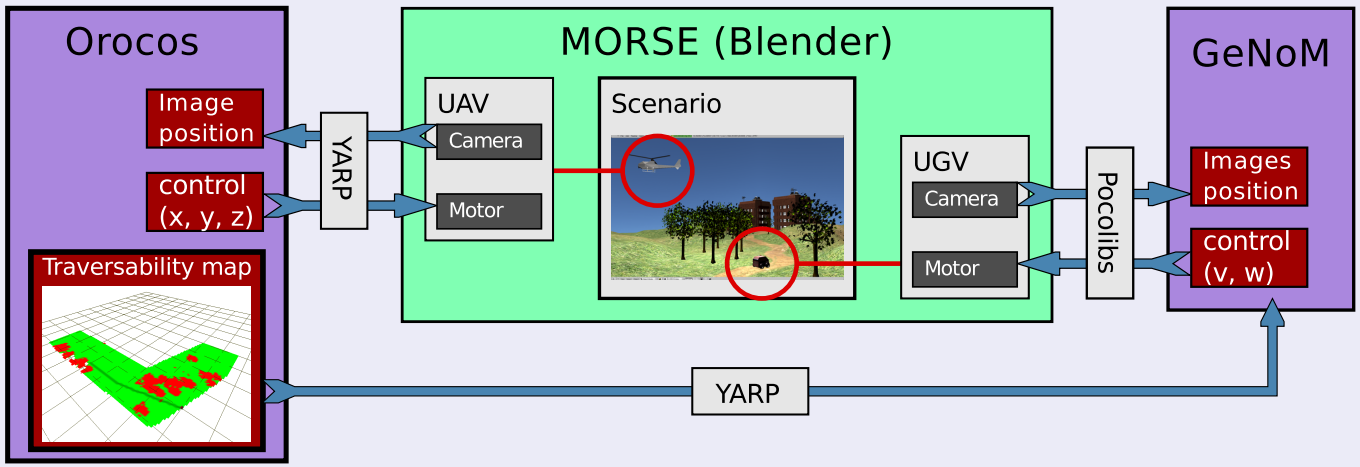

MORSE main loop
main.init()
SET sensor.output_modifiers
SET sensor.output_functions
SET actuator.input_functions
SET actuator.input_modifiers
(from component_config.py)
FOR_EACH sensors
Blender 3D world
sensor.action()
sensor.default_action()
sensor.local_data
FOR_EACH sensor.output_modifiers
output_modifier(data)
FOR_EACH sensor.output_functions
output_function(data)
External client software
(comunication via middlewares)
FOR_EACH actuators
actuator.action()
FOR_EACH actuator.input_functions
input_function(data)
FOR_EACH actuator.input_modifiers
input_modifier(data)
sensor.local_data
actuator.default_action()
Blender 3D world


FOR_EACH sensors
Blender 3D world
sensor.action()
sensor.default_action()
sensor.local_data
FOR_EACH sensor.output_modifiers
output_modifier(data)
FOR_EACH sensor.output_functions
output_function(data)
FOR_EACH actuators
actuator.action()
FOR_EACH actuator.input_functions
input_function(data)
FOR_EACH actuator.input_modifiers
input_modifier(data)
External client software
(comunication via middlewares)

Distributed control of the simulation

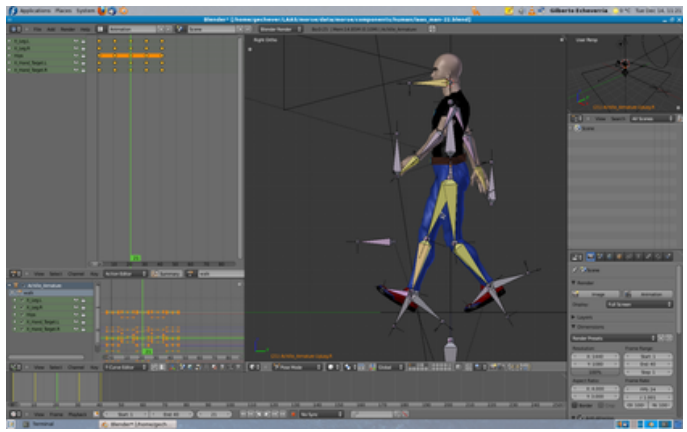
- Robot software connects to simulator through middlewares
- MORSE can run on separate CPUs from the robot software
- A single instance of MORSE controls the simulation

Distributed control of heterogeneous robots

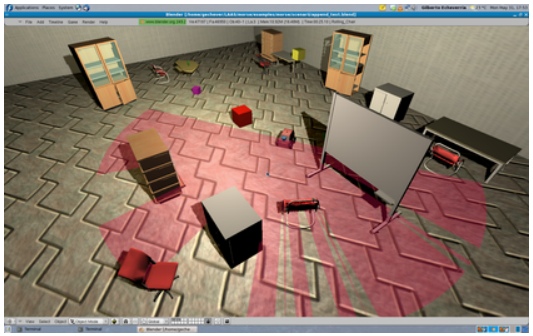
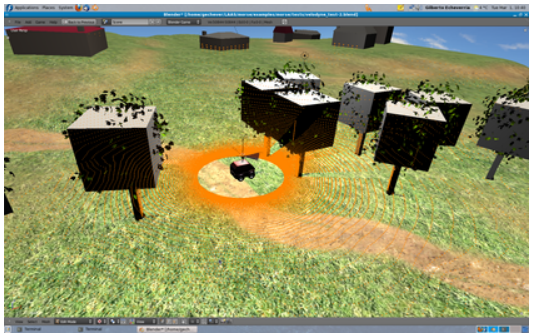
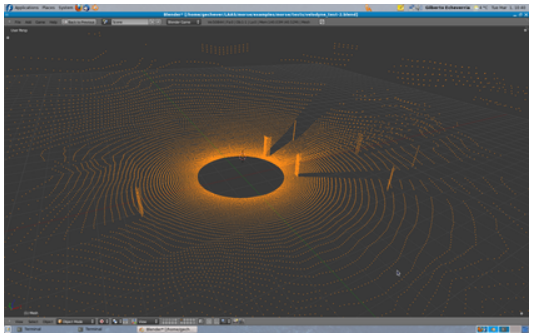


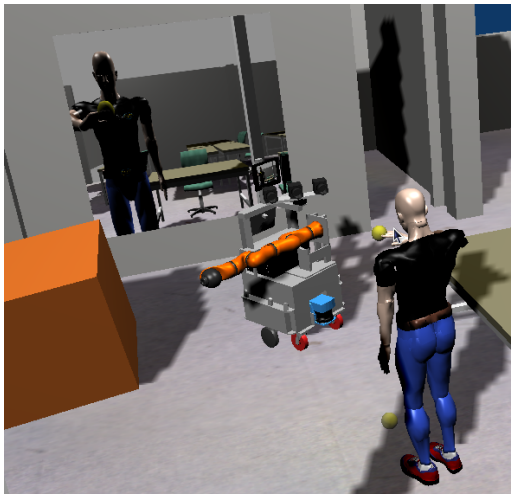
Switch to Blender 2.5x and Python 3.1

- Code ported for compatibility with Python 3
- Improved performance
- Faster image processing
- Better graphics display when using shaders
- Inverse kinematics for human models and arms
- Control panels configurable with Python



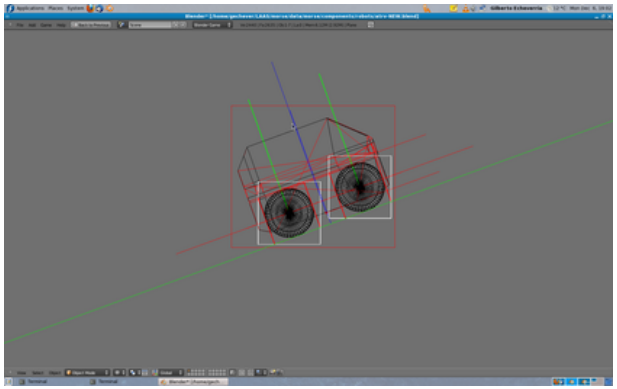
New sensors

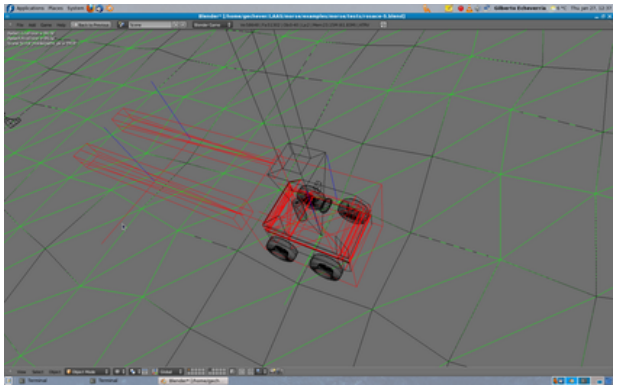






- Calibration of simulated camera properties
- Stereo cameras
- Semantic cameras
- Robot proximity sensors
- Sick and Velodyne range sensors
- Detection of human posture

Control



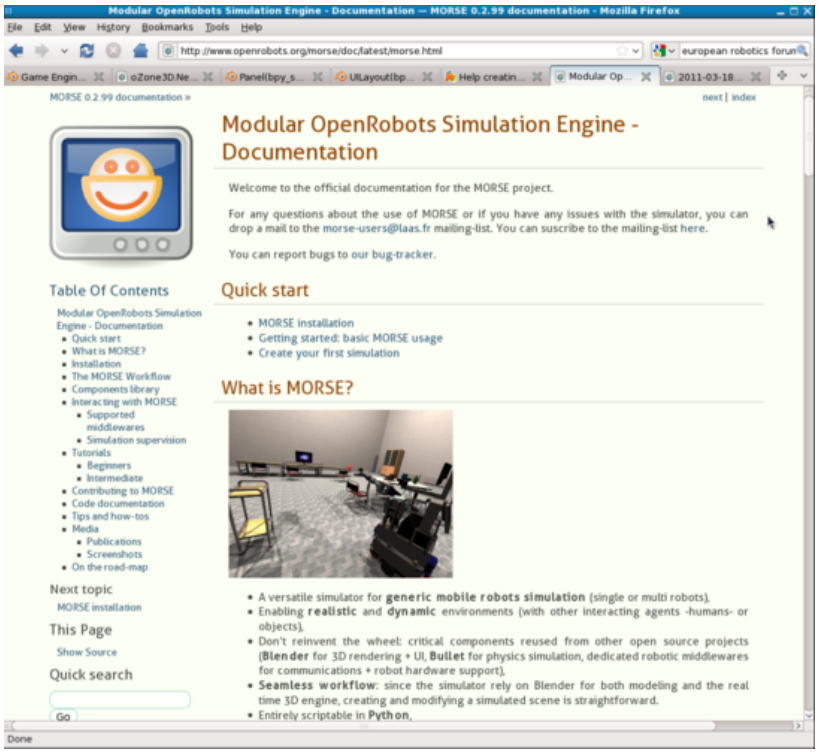


- Camera pant-tilt units
- Simple obstacle avoidance
- Correction of physical properties for robots and environment
- Switch cameras in simulation
- Implement internal clock in MORSE to measure elapsed time in simulation

Middlewares support

- YARP
- Pocolibs
- ROS (developed at TUM)
- Sockets
- Request manager
  (Sockets and YARP)

Release of version 0.3

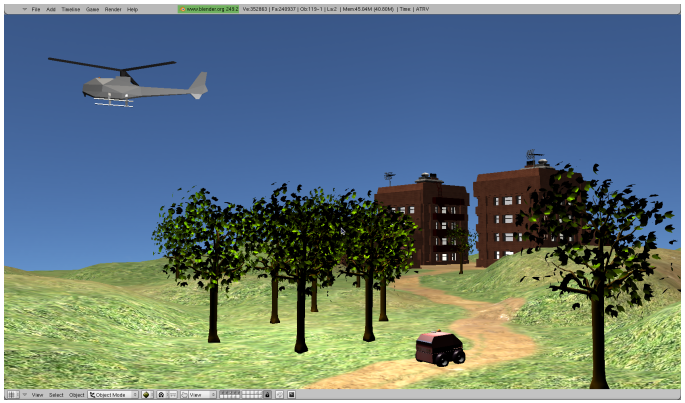


- Initial ROS support
- Large documentation effort, switch to restructured text format
- Add support for Blender 2.57
- Add component services: synchronous or asynchronous commands
- Tracking of elapsed time in the simulation

Outline

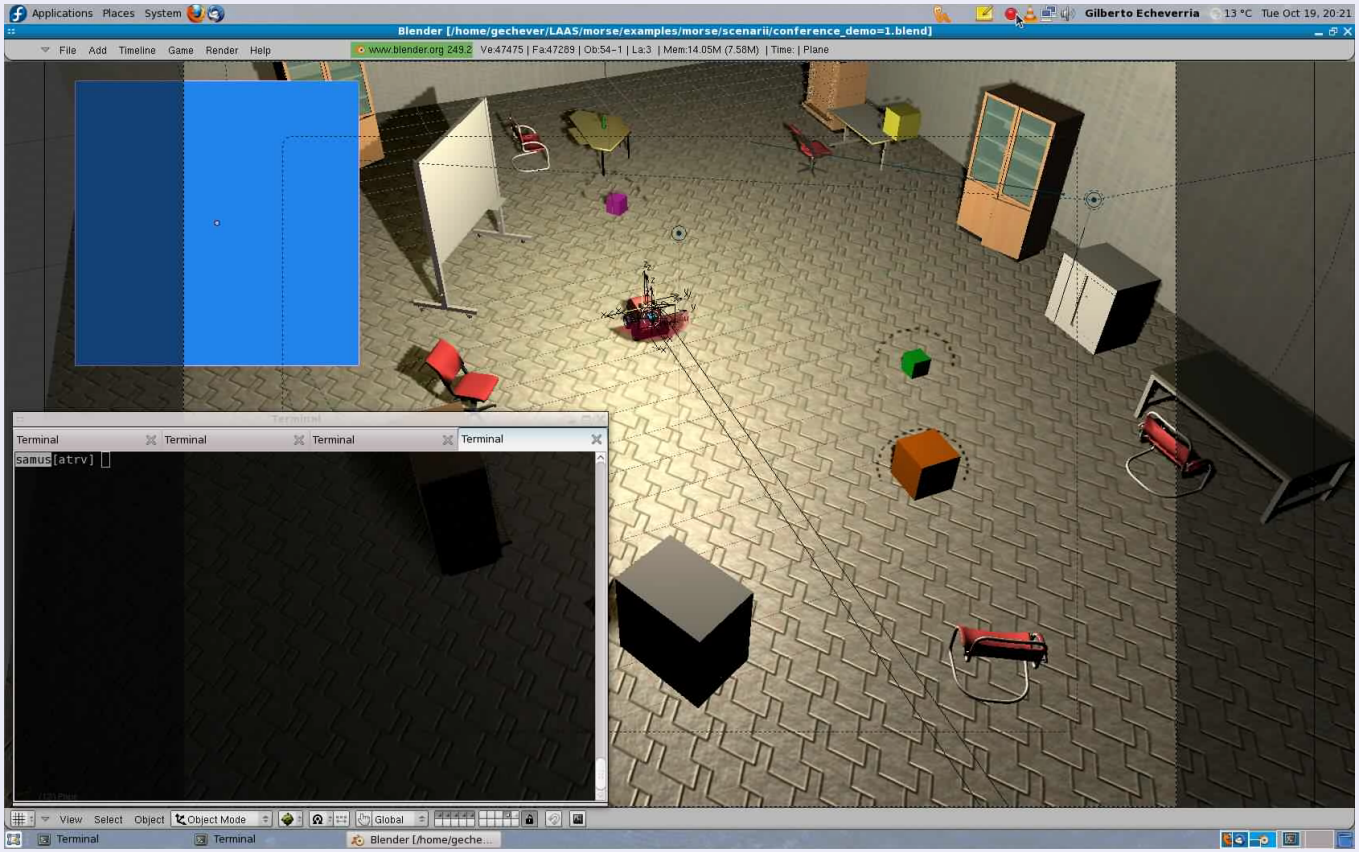
Current results

- Data input and output tested with real software
- Complete experiments possible using existing robot software
- Currently being used to debug navigation and task planning in various projects
- MORSE active developers at LAAS (France), UK Leuven (Belgium) and TUM (Germany)
- Large number of interested users



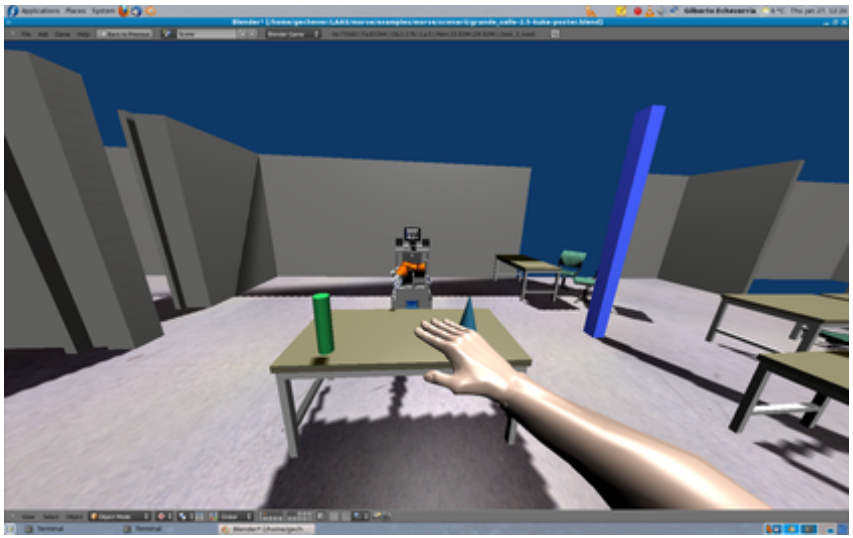
Simulation examples

Single robot simulation

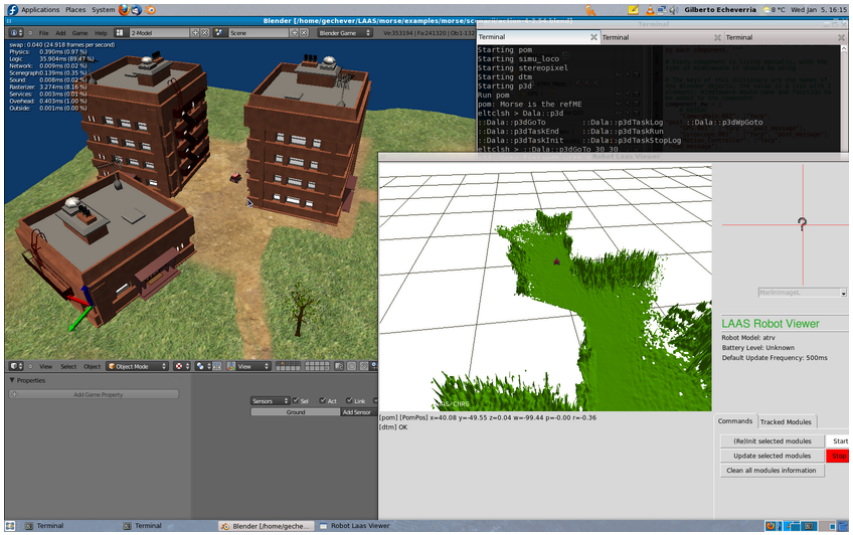


Projects using MORSE at LAAS

- CHRIS:
  Human robot interaction



- ACTION:
  Multi robot navigation

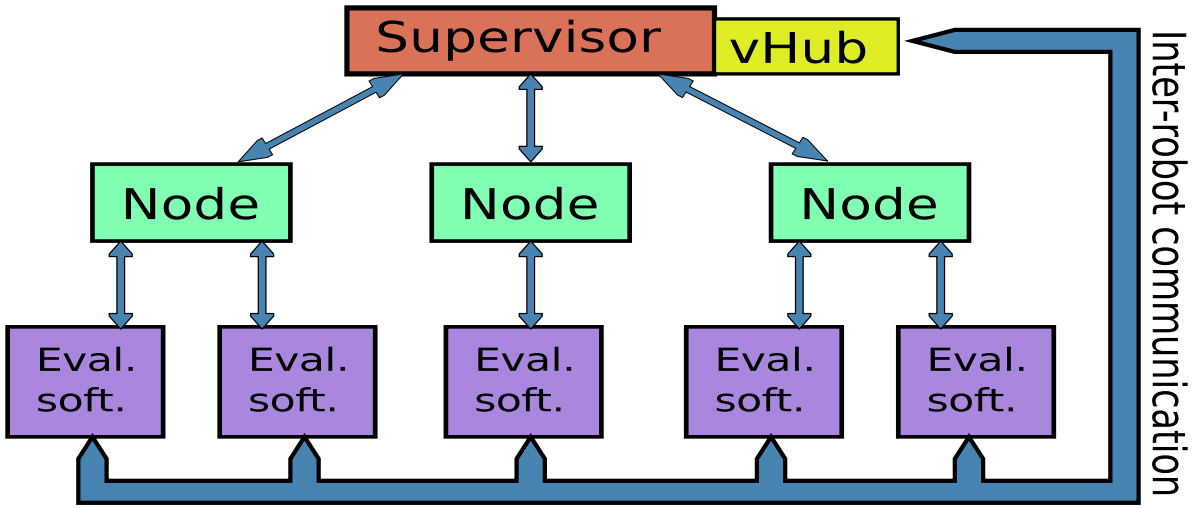


Dissemination activities

- MORSE training courses at LAAS and ONERA, Toulouse
- International events
  - Euron / Europ meeting (Mar 2010, San Sebastian, Spain)
  - BlenderVen Simposium (Oct 2010, San Cristobal, Venezuela)
  - Blender Conference (Oct 2010, Amsterdam, Netherlands)
  - ICRA 2011 Paper accepted for the conference, Shanghai, May 2011
- Hackathon (Jan 2011, Toulouse) Participation of researchers from ONERA, LAAS, Katholieke Universiteit Leuven (Belgium) and Technische Universität München (Germany)

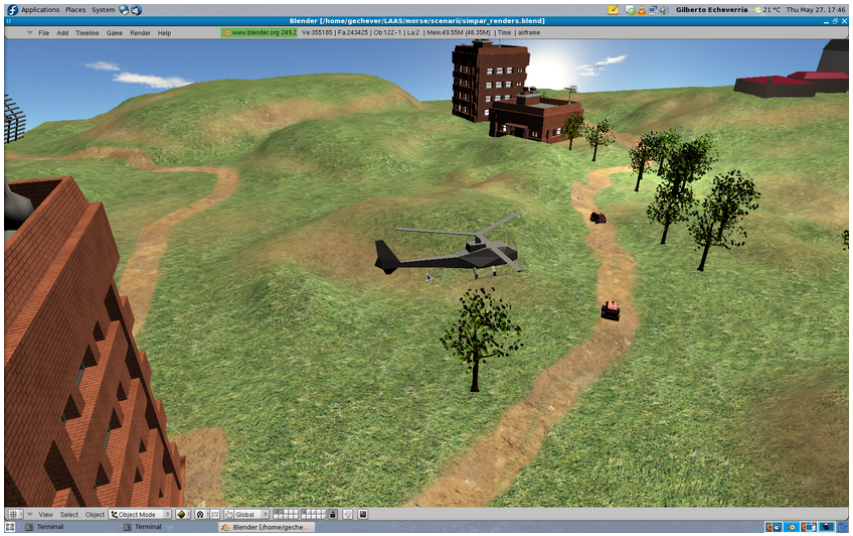
Extending MORSE

- Implementation of complex depth image and other sensors
- Dedicated GUI for scene creation and simulation control
- Full support for PR2 robot and the ROS middleware
- Simulation supervisor to manage custom events
- Virtual Hub (vHub) to simulate robot communications
- Distributed architecture for multiple simulation nodes.



Summary

- MORSE can be used in various robotics developments thanks to a modular structure and middleware compatibility
- Support for multiple heterogeneous robots
- Recent developments allow for a more robust architecture
- Active development community



Contact, questions, more information:

MORSE is an Open Source project.
Download, learn and get involved:

- Users mailing list: morse-users@laas.fr
- Developers mailing list: morse-dev@laas.fr
- Website: http://morse.openrobots.org

THANKS FOR YOUR ATTENTION!!!