RESEARCH CENTER
Bordeaux - Sud-Ouest

FIELD

Activity Report 2012

Section Software

Edition: 2013-04-24

LFANT Project-Team

5. Software

5.1. Pari/Gp

Participants: Karim Belabas [correspondant], Bill Allombert, Henri Cohen, Andreas Enge.

http://pari.math.u-bordeaux.fr/

PARI/GP is a widely used computer algebra system designed for fast computations in number theory (factorisation, algebraic number theory, elliptic curves, ...), but it also contains a large number of other useful functions to compute with mathematical entities such as matrices, polynomials, power series, algebraic numbers, etc., and many transcendental functions.

- PARI is a C library, allowing fast computations.
- GP is an easy-to-use interactive shell giving access to the PARI functions.
- `gp2c`, the GP-to-C compiler, combines the best of both worlds by compiling GP scripts to the C language and transparently loading the resulting functions into GP; scripts compiled by `gp2c` will typically run three to four times faster.
- Version of PARI/GP: 2.5.3
- Version of `gp2c`: 0.0.7pl4
- License: GPL v2+
- Programming language: C

5.2. GNU MPC

Participants: Andreas Enge [correspondant], Mickaël Gastineau, Philippe Théveny, Paul Zimmermann [INRIA project-team CARAMEL].

http://mpc.multiprecision.org/.

GNUMPC is a C library for the arithmetic of complex numbers with arbitrarily high precision and correct rounding of the result. It is built upon and follows the same principles as GNUMPFR.

It is a prerequisite for the GNU compiler collection GCC since version 4.5, where it is used in the C and Fortran frontends for constant folding, the evaluation of constant mathematical expressions during the compilation of a program. Since 2011, it is an official GNU project.

2011 has seen the first release of the major version 1.0.

- Version: 1.0.1 *Fagus silvatica*
- License: LGPL v3+
- ACM: G.1.0 (Multiple precision arithmetic)
- AMS: 30.04 Explicit machine computation and programs
- APP: Dépôt APP le 2003-02-05 sous le numéro IDDN FR 001 060029 000 R P 2003 000 10000
- Programming language: C

5.3. MPFRCX

Participant: Andreas Enge.

http://mpfrcx.multiprecision.org/

MPFRCX is a library for the arithmetic of univariate polynomials over arbitrary precision real (MPFR) or complex (MPC) numbers, without control on the rounding. For the time being, only the few functions needed to implement the floating point approach to complex multiplication are implemented. On the other hand, these comprise asymptotically fast multiplication routines such as Toom-Cook and the FFT.

- Version: 0.4.1 *Cassava*
- License: LGPL v2.1+
- Programming language: C

5.4. CM

Participant: Andreas Enge.

http://cm.multiprecision.org/

The CM software implements the construction of ring class fields of imaginary quadratic number fields and of elliptic curves with complex multiplication via floating point approximations. It consists of libraries that can be called from within a C program and of executable command line applications. For the implemented algorithms, see [9].

- Version: 0.2 *Blindhühnchen*
- License: GPL v2+
- Programming language: C

5.5. AVIsogenies

Participants: Damien Robert [correspondant], Gaëtan Bisson, Romain Cosset [INRIA project-team CARAMEL].

http://avisogenies.gforge.inria.fr/.

AVISOGENIES (Abelian Varieties and Isogenies) is a MAGMA package for working with abelian varieties, with a particular emphasis on explicit isogeny computation.

Its prominent feature is the computation of (ℓ, ℓ)-isogenies between Jacobian varieties of genus-two hyperelliptic curves over finite fields of characteristic coprime to ℓ; practical runs have used values of ℓ in the hundreds.

It can also be used to compute endomorphism rings of abelian surfaces, and find complete addition laws on them.

- Version: 0.6
- License: LGPL v2.1+
- Programming language: Magma

5.6. Cubic

Participant: Karim Belabas.

http://www.math.u-bordeaux1.fr/~belabas/research/software/cubic-1.2.tgz

CUBIC is a standalone program that prints out generating equations for cubic fields of either signature and bounded discriminant. It depends on the PARI library. The algorithm has quasi-linear time complexity in the size of the output.

- Version: 1.2
- License: GPL v2+
- Programming language: C

5.7. Euclid

Participant: Pierre Lezowski.

http://www.math.u-bordeaux1.fr/~plezowsk/euclid/index.php

EUCLID is a C program to compute the Euclidean minimum of a number field. It uses the PARI library.

- Version: 1.0
- License: GPL v2+
- Programming language: C

5.8. KleinianGroups

Participant: Aurel Page.

http://www.normalesup.org/~page/Recherche/Logiciels/logiciels.html

KLEINIANGROUPS is a Magma package that computes fundamental domains of arithmetic Kleinian groups.

- Version: 1.0
- License: GPL v3+
- Programming language: Magma

ALEA Project-Team

5. Software

5.1. BiiPS software

BiiPS is a general software, developped by Adrien Todeschini, for Bayesian inference with interacting particle systems, a.k.a. sequential Monte Carlo (SMC) methods. It aims at popularizing the use of these methods to non-statistician researchers and students, thanks to its automated “black box" inference engine.

It borrows from the BUGS/JAGS software, widely used in Bayesian statistics, the statistical modeling with graphical models and the language associated with their descriptions.

Unlike MCMC methods used by BUGS/JAGS, SMC methods are more adapted to dynamic problems (tracking, signal filtering, etc).

A beta version of the software can be downloaded from the website of the BiiPS project. This software has been presented at the international workshop BayesComp in Kyoto, the international conference ISBA in Tokyo, the conference on Premières Rencontres R in Bordeaux, and the international workshop on efficient simulation in finance in Paris.

BACCHUS Team

5. Software

5.1. AeroSol

Participants: Damien Genêt [corresponding member for Bacchus], Maxime Mogé, Dragan Amenga-Mbengoué, François Pellegrini, Vincent Perrier [corresponding member], Mario Ricchiutto, François Rue.

The `Aerosol` software is jointly developed by teams `BACCHUS` and `Cagire`. It is a high order finite element library written in C++. The code design has been carried for being able to perform efficient computations, with continuous and discontinuous finite elements methods on hybrid and possibly curvilinear meshes. The distribution of the unknowns is made with the software `PaMPA`, developed within teams `BACCHUS` and `PUMAS`. Maxime Mogé has been hired on a young engineer position (IJD) obtained in the ADT `OuBa HOP` for participating in the parallelization of the library, and arrived on November, 1st 2011. On January 2012, Dragan Amenga-Mbengoué was recruited on the ANR `Realfluids`.

At the end of 2011, Aerosol had the following features:

- Development environement: use of `CMake` for compilation, `CTest` for automatic testing and memory checking, `lcov` and `gcov` for code coverage reports.
- In/Out: link with the XML library for handling with parameter files. Reader for `GMSH`, and writer to the VTK-ASCII legacy format.
- Quadrature formula: up to 11th order for Lines, Quadrangles, Hexaedra, Pyramids, Prisms, up to 14th order for tetrahedron, up to 21st order for triangles.
- Finite elements: up to fourth degree for Lagrange finite elements on lines, triangles and quadrangles.
- Geometry: elementary geometrical functions for first order lines, triangles, quadrangles.
- Time iteration: explicit Runge-Kutta up to fourth order, explicit Strong Stability Preserving schemes up to third order.
- Linear Solvers: link with the external linear solver UMFPack.
- Memory handling: discontinuous and continuous discretizations based on `PaMPA` for triangular and quadrangular meshes.
- Numerical schemes: continuous Galerkin method for the Laplace problem (up to fifth order) with non consistent time iteration or with direct matrix inversion. Scalar stabilized residual distribution schemes with explicit Euler time iteration have been implemented for steady problems.

This year, the following features were added:

- Development environement: development of a `CDash` server for collecting the unitary tests and memory checking. Beginning of the development of an interface for functional tests.
- General structure: Parts of the code were abstracted in order to allow for parallel development: Linear solvers (template type abstraction for generic linear solver external library), Generic integrator classes (integrating on elements, on faces with handling neighbor elements, or for working on Lagrange points of a given element), models (template abstraction for generic hyperbolic systems), equations of state (template-based abstraction for a generic equation of state).
- In/Out: Parallel `GMSH` reader, cell and point centered visualization based on VTK-legacy formats. XML paraview files on unstructured meshes (vtu), and parallel XML based files (pvtu).
- Quadrature formula: Gauss-Lobatto type quadrature formula.
- Finite elements: Hierarchichal orthogonal finite element basis on lines, triangles (with Dubiner transform). Finite element basis that are interpolation basis on Gauss-Legendre points for lines, quadrangles, and hexaedra. Lagrange, and Hierarchical orthogonal finite elements basis for hexaedra, prisms and tetrahedra.

- Geometry: elementary geometrical functions for first order three dimensional shapes: hexaedra, prisms, and tetrahedra.
- Time iteration: CFL time stepping, optimized CFL time schemes: SSP(2,3) and SSP (3,4)
- Linear Solvers: Internal solver for diagonal matrices. Link with the external solvers PETSc and MUMPS.
- Memory handling: parallel degrees of freedom handling for continuous and discontinuous approximations
- Numerical schemes: Discontinuous Galerkin methods for hyperbolic systems. SUPG and Residual Distribution schemes.
- Models: Perfect gas Euler system, real gas Euler system, scalar advection, Waves equation in first order formulation, generic interface for defining space-time models from space models.
- Numerical fluxes: centered fluxes, exact Godunov' flux for linear hyperbolic systems, and Lax-Friedrich flux.
- Parallel computing: Mesh redistribution, computation of Overlap with `PaMPA`. Collective asynchronous communications (`PaMPA` based). Tests on the cluster Avakas from MCIA, and on Mésocentre de Marseille. The library was also compiled on PlaFRIM.
- C++/Fortran interface: Tests for binding fortran with C++.

5.2. COCA

Participants: Mario Ricchiuto [corresponding member], Gérard Vignoles.

`COCA`(CodeOxydationCompositesAutocicatrisants) is a `Fortran 90` code for the simulation of the oxidation process in self-healing composite materials, developed in collaboration with the Laboratoire des Composites ThermoStructuraux in Bordeaux (UMR-5801 LCTS). This process involves the chemical oxidation of some of the matrix components of the composite, and the production of a liquid oxide that flows and fills material cracks, acting as a diffusion barrier against oxygen and thus protecting the ceramic fibers of the material. `COCA` simulates this process using a finite element discretization of the model equations. In its current version only transverse cracks are available. `COCA` makes use of `PaStiX` to solve the algebraic systems arising from the discretization.

5.3. RealfluiDS

Participants: Dante de Santis, Gianluca Geraci, Pietro Marco Congedo, Rémi Abgrall [corresponding member].

`RealfluiDS` is a software dedicated to the simulation of inert or reactive flows. It is also able to simulate multiphase, multimaterial, MHD flows and turbulent flows (using the SA model). There exist 2D and 3D dimensional versions. The 2D version is used to test new ideas that are later implemented in the 3D one. This software implements the more recent residual distribution schemes. The code has been parallelized with and without overlap of the domains. An Uncertainty Quantification library has been added to the software. A partitioning tool exists in the package, which uses `Scotch`. In the years to come, all the know-how of `RealfluiDS` will be transferred to `Aerosol`.

5.4. MMG3D

Participants: Cécile Dobrzynski [corresponding member], Algiane Froehly.

MMG3D is a tetrahedral fully automatic remesher. Starting from a tetrahedral mesh, it produces quasi-uniform meshes with respect to a metric tensor field. This tensor prescribes a length and a direction for the edges, so that the resulting meshes will be anisotropic. The software is based on local mesh modifications and an anisotropic version of Delaunay kernel is implemented to insert vertices in the mesh. Moreover, MMG3D allows one to deal with rigid body motion and moving meshes. When a displacement is prescribed on a part of the boundary, a final mesh is generated such that the surface points will be moved according this displacement. MMG3D is used in particular in GAMMA for their mesh adaptation developments, but also at EPFL (maths department), Dassault Aviation, Lemma (a french SME), etc. MMG3D can be used in FreeFem++ (http://www.freefem.org), a free software which eases the solving of PDEs and in Gmsh (http://geuz.org/gmsh/). More details can be found on http://www.math.u-bordeaux1.fr/~dobj/logiciels/mmg3d.php.

A new version of MMG3D is under development. The big novelty of this version is the modification of the surface triangulation. A. Froehly, ingenieer in the FUI Rodin, is working on this new version.

5.5. ORComp

Participants: Pietro Marco Congedo [Corresponding member], Rémi Abgrall, Nassim Razaaly, Dante De Santis, Maria-Giovanna Rodio.

The ORComp platform is a simulation tool permitting to design an ORC cycle. Starting from the solar radiation, this plateform computes the cycle providing the best performance with optimal choices of the fluid and the operating conditions. It includes RobUQ, a simulation block of the ORC cycles, the RealFluid code for the simulation of the turbine and of the heat exchanger, the software FluidProp (developed at the University of Delft) for computing the fluid thermodynamic properties.

5.6. PaMPA

Participants: Cédric Lachat, François Pellegrini [Corresponding member], Cécile Dobrzynski, Hervé Guillard [PUMAS], Laurent Hascoët [Tropics].

PaMPA ("Parallel Mesh Partitioning and Adaptation") is a middleware library dedicated to the management of distributed meshes. Its purpose is to relieve solver writers from the tedious and error prone task of writing again and again service routines for mesh handling, data communication and exchange, remeshing, and data redistribution. It is based on a distributed data structure that represents meshes as a set of *entities* (elements, faces, edges, nodes, etc.), linked by *relations* (that is, computation dependencies).

PaMPA interfaces with Scotch for mesh redistribution, and with MMG3D for parallel remeshing of tetrahedral elements. Other sequential remeshers can be plugged in order to handle other types of elements.

Version 0.2 allows users to declare a distributed mesh, to declare values attached to the entities of the meshes (e.g. temperature attached to elements, pressures to the faces, etc.), to exchange values between overlapping entities located at the boundaries of subdomains assigned to different processors, to iterate over the relations of entities (e.g. iterate over the faces of elements), to remesh the pieces of the mesh that need to, and to redistribute evenly the remeshed mesh across the processors of the parallel architecture.

PaMPA is already used as the data structure manager for two solvers being developed at Inria: Plato and Aerosol.

5.7. PLATO

Participants: Hervé Guillard [PUMAS], Laure Combe [PUMAS,contact], Cédric Lachat, Pierre Ramet [corresponding member].

The developement of Plato ("*A platform for Tokamak simulation*") (http://www-sop.inria.fr/pumas/plato.php) is being supported by an ADT action of the D2T. Plato is a suite of data and software dedicated to the geometry and physics of Tokamaks and its main objective is to provide the Inria large scale initiative FUSION teams working with plasma fluid models with a common development tool. The construction of this platform will integrate the following developments.

1. A (small) database corresponding to axi-symmetrical solutions of the equilibrium plasma equations for realistic geometrical and magnetic configurations (ToreSupra, JET and ITER). The construction of meshes is always an important time consuming task. Plato will provide meshes and solutions corresponding to equilibrium solutions that will be used as initial data for more complex computations.
2. A set of tools for the handling, manipulation and transformation of meshes and solutions using different discretisations (P1, Q1, P3, etc)
3. Numerical templates allowing the use of 3D discretization schemes using finite element schemes in the poloidal plane and spectral Fourier or structured finite volume representations in the toroidal one.
4. Several applications (Ideal MHD and drift approximation) used in the framework of the Inria large scale initiative FUSION.

5.8. RobUQ

Participants: Pietro Marco Congedo [Corresponding member], Rémi Abgrall, Gianluca Geraci, Julie Tryoen, Nassim Razaaly.

The RobUQ platform has been conceived to solve problems in uncertainty quantification and robust design. It includes the optimization code ALGEN, and the uncertainty quantification code NISP. It includes also some methods for the computation of high-order statistics, efficient strategies for robust optimization, the Simplex2 method. Some methods are developed in partnership with the Stanford University (in the framework of the associated team AQUARIUS). Other methods are developed in the context of ANR UFO.

5.9. Scotch

Participants: François Pellegrini [corresponding member], Sébastien Fourestier.

parallel graph partitioning, parallel static mapping, parallel sparse matrix block ordering, graph repartitioning, mesh partitioning.

Scotch (http://www.labri.fr/~pelegrin/scotch/) is a software package for parallel and sequential sparse matrix ordering, parallel and sequential graph partitioning, as well as sequential static mapping and remapping, without and with fixed vertices, and mesh and hypergraph partitioning.

The initial purpose of Scotch was to compute high-quality static mappings of valuated graphs representing parallel computations onto target architectures of arbitrary topologies. This allows the mapper to take into account the topology and heterogeneity of the target architecture in terms of processor speed and link bandwidth. This feature, which was meant for the NUMA machines of the 1980's, has not been widely used in the past because machines in the 1990's became UMA again thanks to hardware advances. Now, architectures become NUMA again, and these features are regaining popularity.

Version 5.0 of Scotch, released on August 2007, was the first version to comprise parallel routines. This extension, called PT-Scotch (for "*Parallel Threaded Scotch*"), is based on a distributed memory model, and makes use of the MPI and, optionally, Posix thread APIs. Version 5.1, released on September 2008, extended the parallel features of PT-Scotch, which can now compute graph partitions in parallel by means of a parallel recursive bipartitioning framework. Release 5.1.10 had made Scotch the first full 64-bit implementation of a general purpose graph partitioner.

Version `6.0`, released on December 2012, corresponding to the 20-year anniversary of `Scotch`, offers many new features: static mapping with fixed vertices, static remapping, and static remapping with fixed vertices. Several critical algorithms of the formerly sequential `Scotch` library can now run in a multi-threaded way. All of these features will be available for the parallel `PT-Scotch` library in the upcoming release `6.1`.

`Scotch` has been integrated in numerous third-party software, which indirectly contribute to its diffusion, e.g. OPENFOAM (fluid mechanics solver, OpenCFD ltd.), the CODE_ASTER LIBRE solver (thermal and mechanical analysis software developed by French state-owned electricity producer EDF), the ZOLTAN module of the TRILINOS software (SANDIA Labs), the parallel linear system solvers `MUMPS` (ENSEEITH/IRIT, LIP and LaBRI), `SuperLUDist` (U.C. Berkeley), `PaStiX` (LaBRI) and `HIPS` (LaBRI), etc. `Scotch` is natively available in several Linux and Unix distributions, as well as on some vendors platform (SGI, etc).

5.10. SLOWS

Participant: Mario Ricchiuto [corresponding member].

`SLOWS` ("*Shallow-water fLOWS*") is a `C`-platform allowing the simulation of free surface shallow water flows with friction. Arbitrary bathymetries are allowed, defined either by some complex piecewise analytical expression, or by xyz data files, the classical Manning model for friction is used, and an Exner model is implemented for sediment transport. The equations are discretized with a residual based approach which is an adaptation of the schemes developed for aeronautics applications. Due to the inherent unsteadiness of these flows, the time discretization plays an important role. Three different approaches are available, based on conditionally depth-positivity preserving implicit schemes, or on conditionally depth-positivity preserving genuinely explicit discretizations, or on an unconditionally depth-positivity preserving space-time approach.

CAGIRE Team

5. Software

5.1. AeroSol

Participants: Dragan Amenga-Mbengoué [Bacchus], Damien Genet [Bacchus], Maxime Mogé, Francois Pellegrini [Bacchus], Vincent Perrier [correspondant], Francois Rué [Bacchus], Mario Ricchiuto [Bacchus].

The software AeroSol is jointly developed in the team Bacchus and the team Cagire. It is a high order finite element library written in C++. The code design has been carried for being able to perform efficient computations, with continuous and discontinuous finite elements methods on hybrid and possibly curvilinear meshes. The distribution of the unknowns is made with the software PaMPA, developed within the team Bacchus and the team Pumas. This year, Dragan Amenga-Mbengoué was recruited on the ANR Realfluids, and François Rué (Service Experimentation et Développement) joined the team Bacchus for working on Aerosol.

At the end of 2011, Aerosol had the following features

- **development environement** use of CMake for compilation, CTest for automatic tests and memory checking, lcov and gcov for code coverage reports.
- **In/Out** link with the XML library for handling with parameter files. Reader for GMSH, and writer on the VTK-ASCII legacy format.
- **Quadrature formula** up to 11th order for Lines, Quadrangles, Hexaedra, Pyramids, Prisms, up to 14th order for tetrahedron, up to 21st order for triangles.
- **Finite elements** up to fourth degree for Lagrange finite elements on lines, triangles and quadrangles.
- **Geometry** elementary geometrical functions for first order lines, triangles, quadrangles.
- **Time iteration** explicit Runge-Kutta up to fourth order, explicit Strong Stability Preserving schemes up to third order.
- **Linear Solvers** link with the external linear solver UMFPack.
- **Memory handling** discontinuous and continuous discretizations based on PaMPA for triangular and quadrangular meshes.
- **Numerical schemes** continuous Galerkin method for the Laplace problem (up to fifth order) with non consistent time iteration or with direct matrix inversion. Scalar stabilized residual distribution schemes with explicit Euler time iteration have been implemented for steady problems.

This year, the following features were added

- **development environement** development of a CDash server for collecting the unitary tests and memory checking. Beginning of the development of an interface for functional tests.
- **General structure** Parts of the code were abstracted in order to allow for parallel development: Linear solvers (template type abstraction for generic linear solver external library), Generic integrator classes (integrating on elements, on faces with handling neighbour elements, or for working on Lagrange points of a given element), models (template abstraction for generic hyperbolic systems), equations of state (template-based abstraction for a generic equation of state).
- **In/Out** Parallel GMSH reader, cell and point centered visualization based on VTK-legacy formats. XML paraview files on unstructured meshes (vtu), and parallel XML based files (pvtu).
- **Quadrature formula** Gauss-Lobatto type quadrature formula.
- **Finite elements** Hierarchichal orthogonal finite element basis on lines, triangles (with Dubiner transform). Finite element basis that are interpolation basis on Gauss-Legendre points for lines, quadrangles, and hexaedra. Lagrange, and Hierarchical orthogonal finite elements basis for hexaedra, prisms and tetrahedra.

- **Geometry** elementary geometrical functions for first order three dimensional shapes: hexaedra, prisms, and tetrahedra.
- **Time iteration** CFL time stepping, optimized CFL time schemes: SSP(2,3) and SSP (3,4)
- **Linear Solvers** Internal solver for diagonal matrices. Link with the external solvers PETSc and MUMPS.
- **Memory handling** parallel degrees of freedom handling for continuous and discontinuous approximations
- **Numerical schemes** Discontinuous Galerkin methods for hyperbolic systems. SUPG and Residual Distribution schemes.
- **Models** Perfect gas Euler system, real gas Euler system, scalar advection, Waves equation in first order formulation, generic interface for defining space-time models from space models.
- **Numerical fluxes** centered fluxes, exact Godunov' flux for linear hyperbolic systems, and Lax-Friedrich flux.
- **Parallel computing** Mesh redistribution, computation of Overlap with PaMPA. collective asynchronous communications (PaMPA based). Tests on the cluster Avakas from MCIA, and on Mésocentre de Marseille, and PlaFRIM.
- **C++/Fortran interface** Tests for binding fortran with C++.

CONCHA Project-Team

5. Software

5.1. C++ library Concha

Participants: Roland Becker, Daniela Capatina, Robert Luce, David Trujillo.

The objectives of our library CONCHA are to offer a flexible and extensible software with respect to:

- Numerical methods and
- Physical models.

The aim is to have a flexible code which could easily switch between the different discretizations, in order to provide a toolbox for rapid testing of new ideas.

The software architecture is designed in such a way that a group of core developers can contribute in an efficient manner, and that independent development of different physical applications is possible. Further, in order to accelerate the integration of new members and in order to provide a basis for our educational purposes (see Section 8.1), the software proposes different entrance levels. The basic structure consists of a common block, and several special libraries which correspond to the different fields of applications described in Sections 4.1 –4.4 Hyperbolic solvers, Low-Mach number flow solvers, DNS, and viscoelastic flows. A more detailed description of each special library may be found below. In order to coordinate the cooperative development of the library, Concha is based on the Inria-Gforge.

5.2. User interface and python interface

Participants: Roland Becker, David Trujillo.

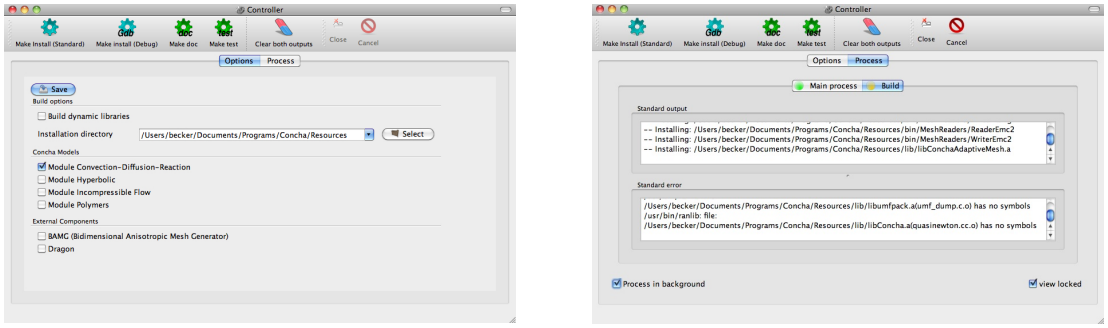

Figure 3. Graphical user interface: option panel (left) and process panel (right) of the install tool.

We are confronted with heterogenous backgrounds and levels of implication of the developers and users. It seems therefore crucial to be able to respond to the different needs. Our aim is to facilitate the development of the library, and at the same time, to make it possible that our colleagues involved in physical modeling can have access to the functionality of the software with a reasonable investment of time. Two graphical user interfaces have been developed: one for the installation of the library and another one for the building and execution of projects. They are based on common database and scripts written in python. The scripts can also be launched in a shell. In Figure 3 the user interface of the install tool is shown. The option panel allows to choose the components for conditional compilation and the compilation type (debug and release).

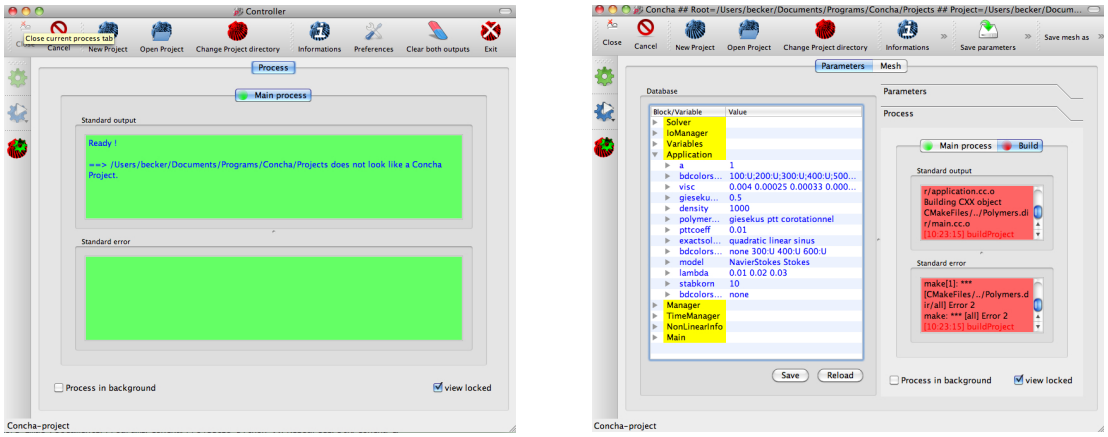

Figure 4. Graphical user interface: project build panel (left) and parameter panel (right) of the project tool.

In Figure 4 the user interface of the project tool is shown. A project consists of a number of sources files and a parameter file used by the C++-executable. The sources define classes derived from the library, which are used to specify certain data such as boundary conditions and employed finite element spaces. The parameter file contains algorithmic information and physical parameters. It is generated from a database by the python utilities.

The tools offered by this development platform are based on a python interface for the library, called pyConcha. It offers a common interface, based on a pluggin-system, which allows the devloppement of command line tools in parallel. This year the consolidation of the interface part of pyConcha has been an important task. The pyConcha library is now a framework rather than a simple interface to Concha C++ library. It allows now creation of plugins, so that each user-programmer can customize pyConcha to his own goals. Previously, two main programs where working: concha-install.py to install library, and concha-project.py for (semi-)end-users. Both are now plugins of pyConcha, and can be launched by pyConcha at startup. A plugin visualization could now be developped in an independant way, and launched by pyConcha on demand.

The structure of pyConcha framework is clearly splitted in various modules(layers): Command Line Interface module, Graphical User Interface module and Handlers modules, see Figure 5 . A great effort has been made for internationalization of pyConcha.

5.3. Euler equations

Participants: Roland Becker, Kossivi Gokpi, Robert Luce, Eric Schall, David Trujillo.

Based on the library CONCHA we have developed a solver for hyperbolic PDE's based on DGFEM. So far different standard solvers for the Euler equations such as Lax-Friedrichs, Steger-Warming, and HLL have been implemented for test problems. A typical example is the scram jet test case shown in Figure 6 .

5.4. Incompressible flow solvers

Participants: Roland Becker, Daniela Capatina, Robert Luce, David Trujillo.

We have started the validation of the implementation of different finite element methods for incompressible flows at hand of standard benchmark problems as the Stokes flow around a symmetric cylinder [65] and the stationary flow around a slightly non symmetric cylinder [70], see Figure 7 .

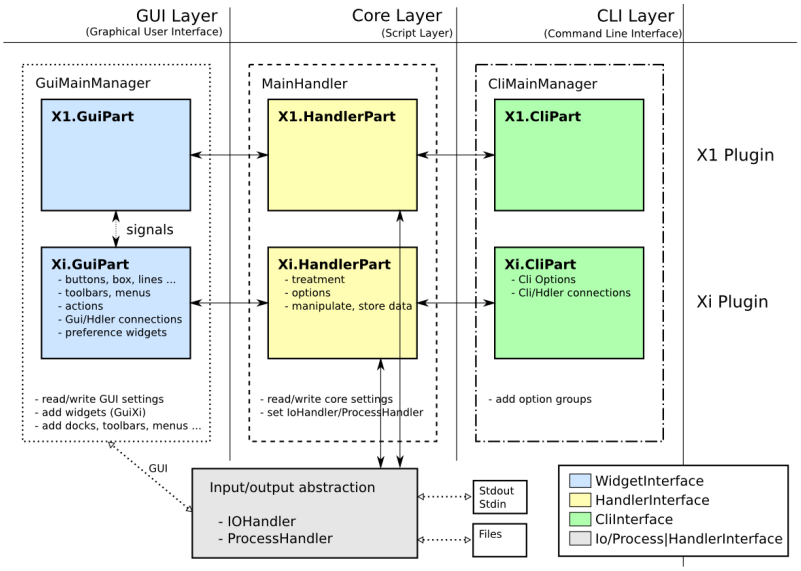

Figure 5. Structure of the pyConcha framework.

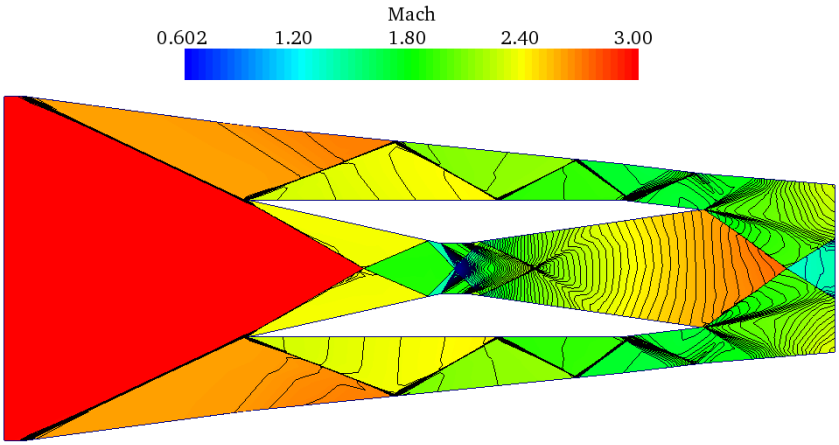

Figure 6. Computed Mach-number distribution for the Scramjet test problem.

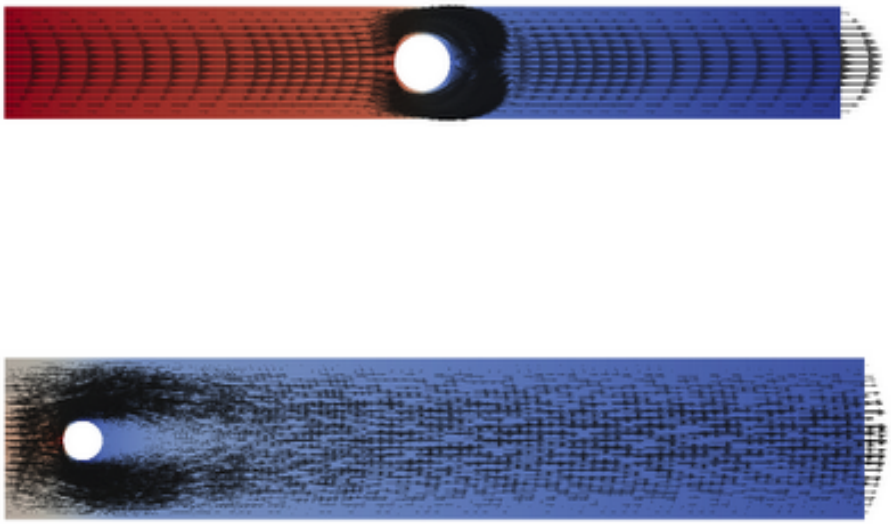

Figure 7. Flow fields for the Stokes (above) and Navier-Stokes (below) benchmark.

5.5. DNS

Participants: Roland Becker, David Trujillo.

For the direct numerical simulation of incompressible turbulent flows, we have started to develop a special solver based on structured meshes with a fast multigrid algorithm incorporating projection-like schemes. The main idea is to use non-conforming finite elements for the velocities with piecewise constant pressures, leading to a special structure of the discrete Schur complement, when an explicit treatment of the convection and diffusion term is used.

5.6. Validation and comparison with other CFD-software

Participants: Roland Becker, Didier Graebling, Eric Schall, David Trujillo.

Validation and comparison with other CFD-software is crucial in order to evaluate the potential of our numerical schemes concerning accuracy, computing time and other practical aspects.

We have compared the Concha library for incompressible and compressible flows. For incompressible flows, we have used a test case proposed by Hulsen and the well-known Schafer-Turek cylinder benchmark in order to validate the accuracy of the Stokes and Navier-Stokes solvers. The viscoelastic code has been compared with PolyFlow for different test configurations.

The compressible Euler code has been compared to the ELSA software developed by ONERA.

For further comparison and validation, it would be important to consider other commercial and research tools such as: *Aéro3* (Inria-Smash), AVBP (CERFACS), Fluent (ANSYS), and OpenFOAM (OpenCfd).

For this purpose we have proposed the ADT-project VALSE in collaboration with a small company involved in aerodynamics (EPSILON Toulouse), which unfortunately has been rejected by Inria.

CQFD Project-Team (section vide)

GEOSTAT Project-Team

5. Software

5.1. FluidExponents

Participants: Denis Arrivault [correspondant], Hussein Yahia, Joel Sudre.

Denis Arrivault has joined the team for a complete refoundation, rewriting, generalization and diffusion of the FluidExponents software. FluidExponents is a software implementation of the MMF, presently written in Java, in a cooperative development mode on the Inria GForge, deposited at APP in 2010. The new software is presently in the phase of specification, and will be rewritten in C++, using existing libraries for data containers, mathematical computation and user interface. Denis Arrivault is recruited for a 24 month period on FluidExponents ADT.
During the new development, researchers still make use of the current version of the FluidExponents software written in Java, version number 0.8. Contact: denis.arrivault@inria.fr.

MC2 Project-Team

5. Software

5.1. eLYSe

Participants: Olivier Saut [correspondant], Raphael Bahègne, Vincent Huber, Jean-Baptiste Lagaert, Mathieu Specklin.

eLYse is a numerical platform used for our computations in Biology (tumor growth), micro-fluidics and complex Newtonian fluid flows. The platform is divided in two libraries : one is devoted to the modelling equations and the other one includes the numerical solvers. For example, we are able to treat (in 2D and 3D) transport equations, diffusion equations, Navier-Stokes equations, Maxwell system and the interaction fluid-structure by level-set and penalization methods. The solvers are based on finite volume methods on cartesian grids and allow parallel computations. See also the web page http://www.math.u-bordeaux1.fr/~osaut/pages/eLYSe.html.

- Version: 0.4
- ACM: ACM J.2 J.3 G.1.8 G.1.10
- AMS: AMS65Z05 35Q92
- Keywords: Modélization and numerical simulations, Finite volume methods, Level Set approach, Penalization method
- APP: En cours
- Type of human computer interaction: console
- OS/Middelware: Platform developped on Mac OS X architecture.
- Required library or software: Petsc (http://www.mcs.anl.gov/petsc/petsc-as/)Vtk (http://www.vtk.org/)Blitz++ (http://c2.com/cgi/wiki?BlitzPlusPlus) (optionnel)Boost (http://www.boost.org/)
- Programming language: C++
- Documentation: doxygen.

5.2. Kesaco

Participants: Olivier Saut [correspondant], Raphael Bahègne, Damiano Lombardi, Mathieu Specklin.

Kesaco is a set of libraries and programs aiming at applications of mathematical modeling in clinical oncology. It features:

- A library of specialized mathematical model describing the growth of different types of cancers (secondary tumors in the lung, gliomas).
- A set of programs useful to validate mathematical models (compute the various behavior they can produce) and to build databases of numerical simulations.
- Segmentation and registration routines to use medical images directly in our numerical codes.
- Calibration methods to recover the parameters of the models using sequences of medical images. Three techniques are implemented (a genetic algorithm, a technique based on reduced order models, a sensitivity technique).

All these routines are adapted to run on a MP architecture. The webpage may be found at http://www.math.u-bordeaux1.fr/~osaut/pages/kesaco.html.

- Version: 0.1
- Keywords: Modélization and numerical simulations
- APP: En cours
- Type of human computer interaction: console
- OS/Middelware: Platform developped on Mac OS X architecture.
- Required library or software: eLYSe, Insight Toolkit (http://www.itk.org)
- Programming language: C++
- Documentation: doxygen.

5.3. NaSCar

Participant: Michel Bergmann [correspondant].

This code is devoted to solve 3D-flows in around moving and deformable bodies. The incompressible Navier-Stokes equations are solved on fixed grids, and the bodies are taken into account thanks to penalization and/or immersed boundary methods. The interface between the fluid and the bodies is tracked with a level set function or in a Lagrangian way. The numerical code is fully second order (time and space). The numerical method is based on projection schemes of Chorin-Temam's type. The code is written in C language and use Petsc (http://www.mcs.anl.gov/petsc/petsc-as/) library for the resolution of large linear systems in parallel.

NaSCar can be used to simulate both hydrodynamic bio-locomation as fish like swimming and aerodynamic flows such wake generated by a wind turbine.

- Version: 1
- Keywords: numerical analyse, fluid mechanics, langage C, PETSc
- Software benefit : simulate a flow around a deformable obstacle, moving into a fluid.
- APP: En cours
- Patent: non
- Type of human computer interaction: human for the moment
- OS/Middelware: unix, linux, mac os
- Required library or software: PETSc item Programming language: C
- Documentation: in progess

5.4. S-MPI-2D-3D

Participants: Charles-Henri Bruneau [correspondant], Khodor Khadra.

The software NS-MPI-2D-3D is a numerical platform devoted to the computation of the incompressible flow around bodies in two or three dimensions modelled by Stokes, Navier-Stokes or Oldroyd-B equations. It is based on finite differences or finite volumes approximations on cartesian grid using the volume penalization method to handle the obstacles. The resolution is achieved by means of the multigrid method. Dirichlet, periodic or artificial boundary conditions are implemented to solve various problems in closed or open domains.

- Version: 3
- Keywords: Numerical simulation of incompressible flows,
- Type of human computer interaction: console
- OS/Middelware: unix, linux, Mac OS X item Programming language: Fortran 95 and MPI
- Documentation: included

5.5. Other MC2 codes

- Penalization techniques on cartesian grids to solve incompressible Navier-Stokes equations
  - **Vortex**: sequential, Vortex In-Cell (VIC) scheme : hybrid vortex methods based on the combination of Lagrangian mesh-free schemes and Eulerian grid based schemes on the same flow region.
  - Unstructured body fitted meshes
  - **Richards** : 2D Unstructured finite element code, implicit solver, sequential, to solve the transport-diffusion equations through a porous media including tidal forcing and mechanisms of diagenesis.
  - development inside **FluidBox** software in collaboration with **BACCHUS**. 2D-3D unstructured meshes, Stabilized Finite Elements method (SUPG), RANS turbulence model, parallel: Domain Decomposition and MPI.
- Immersed boundary techniques for:
  - **Compressible flows** : 2D-3D finite volume scheme for compressible Euler equations with solid obstacles on cartesian grids. 3D code parallelized with MPI
  - **Elliptic problems :** 2 2D-3D finite difference scheme for elliptic interface problems, parallelized with PETSc
  - Electropermeabilization: 2D finite difference scheme, parallelized with PETSc to simulate the electropermeablization of biological cells

REALOPT Project-Team

5. Software

5.1. BaPCod – a generic Branch-and-Price Code

Participants: Romain Leguay [Software Engineer], Pierre Pesneau, Ruslan Sadykov, François Vanderbeck [correspondant].

BaPCod is a prototype code that solves Mixed Integer Programs (MIP) by application of a Dantzig-Wolfe reformulation technique. The reformulated problem is solved using a branch-and-price (column generation) algorithm. This software platform, made of C++ classes, offers a "*black-box*" implementation that does not require user input and is not application specific. The features are
(i) the automation of the Dantzig-Wolfe reformulation process (the user defines a mixed integer programming problem in a pseudo modeling language, defining variables and constraints, identifying subproblems. He can provide subproblem solvers if available, but he does not need to explicitly define the reformulation, the explicit form of the columns, their reduced cost, or the Lagrangian bounds.
(ii) a default column generation procedure with standard initialization and stabilization [1], [23] [25] [29] [32] and
(iii) a default branching scheme that is generic to all applications [9],
(iv) default primal heuristics specially developed for use in a decomposition framework [49], [27], [30].
The prototype software was/is used as background solver for 5 PhD thesis. It also served as the framework for our comparative study in a Inria collaborative research action [1]. It has been experimented by two of our industrial partners, Exeo Solutions (Bayonne), on an inventory routing problem, and Orange Lab (France Telecom, Paris) on network design problems. The prototype also enables us to be very responsive in our industrial contact.

See also the web page https://wiki.bordeaux.inria.fr/realopt/pmwiki.php/Project/BaPCod.

CARMEN Team (section vide)

MAGIQUE-3D Project-Team

5. Software

5.1. Hou10ni

This software, written in FORTRAN 90, simulates the propagation of acoustic waves in heterogeneous 2D and 3D media. It is based on an Interior Penalty Discontinuous Galerkin Method (IPDGM). The 2D version of the code has been implemented in the Reverse Time Migration (RTM) software of TOTAL in the framework of the Ph.D thesis of Caroline Baldassari. The 2D code allows for the use of meshes composed of cells of various order (p-adaptivity in space). For the time discretization, we used the local time stepping strategy described at section 3.2 , item **High-Order Schemes in Space and Time** which permits not only the use of different time-step, but also to adapt the order of the time-discretization to the order of each cells (hp-adaptivity in time).

The main competitors of Hou10ni are codes based on Finite Differences, Spectral Element Method or other Discontinuous Galerkin Methods (such as the ADER schemes). During her Ph.D thesis, Caroline Baldassari compared the solution obtained by Hou10ni to the solution obtained by a Finite Difference Method and by a Spectral Element Method (SPECFEM). To evaluate the accuracy of the solutions, we have compared them to analytical solutions provided by the codes Gar6more (see below). The results of these comparisons is: a) that Hou10ni outperforms the Finite Difference Methods both in terms of accuracy and of computational burden and b) that its performances are similar to Spectral Element Methods. Since Hou10ni allows for the use of meshes based on tetraedrons, which are more appropriate to mesh complex topographies, and for the p-adaptivity, we decided to implement it in the RTM code of TOTAL. Of course, we also used these comparisons to validate the code. Now, it remains to compare the performances of Hou10ni to the ADER schemes.

Recently, we have extended the 2D version of Hou10ni for computing the solution of the harmonic wave equation (Helmholtz). This new version is able to deal with both acoustic and elastodynamic media, but also to model elastoacoustic problems. The surfaces between the different media can be approximated by curved elements. We can use up to P^{15} elements when dealing with curved elements and element of arbitrary order (with of course a limitation depending on the machine precision) when dealing with non-curved elements.

5.2. Gar6more3D

Participant: Julien Diaz [correspondant].

This code computes the analytical solution of problems of waves propagation in two layered 3D media such as- acoustic/acoustic- acoustic/elastodynamic- acoustic/porous- porous/porous, based on the Cagniard-de Hoop method.
See also the web page http://web.univ-pau.fr/~jdiaz1/software.html.

The main objective of this code is to provide reference solutions in order to validate numerical codes. They have been already used by J. Tromp and C. Morency to validate their code of poroelastic wave propagation [87]. They are freely distributed under a CECILL licence and can be downloaded on the website http://web.univ-pau.fr/~jdiaz1/software.html. As far as we know, the main competitor of this code is EX2DELDEL (available on http://www.spice-rtn.org), but this code only deals with 2D acoustic or elastic media. Our codes seem to be the only one able to deal with bilayered poroelastic media and to handle the three dimensional cases.

- ACM: J.2
- AMS: 34B27 35L05 35L15 74F10 74J05
- Programming language: Fortran 90

MAGNOME Project-Team

5. Software

5.1. Inria Bioscience Resources

Participants: Olivier Collin [correspondant], Frédéric Cazals, Mireille Régnier, Marie-France Sagot, Hélène Touzet, Hidde de Jong, David James Sherman, Marie-Dominique Devignes, Dominique Lavenier.

Inria Bioscience Resources is a portal designed to improve the visibility of bioinformatics tools and resources developed by Inria teams. This portal will help the community of biologists and bioinformatians understand the variety of bioinformatics projects in Inria, test the different applications, and contact project-teams. Eight project-teams participate in the development of this portal. Inria Bioscience Resources is developed in an Inria Technology Development Action (ADT).

5.2. Magus: Collaborative Genome Annotation

Participants: David James Sherman [correspondant], Pascal Durrens, Natalia Golenetskaya, Florian Lajus, Tiphaine Martin.

As part of our contribution the Génolevures Consortium, we have developed over the past few years an efficient set of tools for web-based collaborative annotation of eukaryote genomes. The MAGUS genome annotation system integrates genome sequences and sequences features, *in silico* analyses, and views of external data resources into a familiar user interface requiring only a Web navigator. MAGUS implements the annotation workflows and enforces curation standards to guarantee consistency and integrity. As a novel feature the system provides a workflow for *simultaneous annotation* of related genomes through the use of protein families identified by *in silico* analyses; this has resulted in a three-fold increase in curation speed, compared to one-at-a-time curation of individual genes. This allows us to maintain Génolevures standards of high-quality manual annotation while efficiently using the time of our volunteer curators.

MAGUS is built on: a standard sequence feature database, the Stein lab generic genome browser [61], various biomedical ontologies (http://obo.sf.net), and a web interface implementing a representational state transfer (REST) architecture [39].

For more information see magus.gforge.inria.fr, the MAGUS Gforge web site. MAGUS is developed in an Inria Technology Development Action (ADT).

5.3. YAGA: Yeast Genome Annotation

Participants: Tiphaine Martin, Pascal Durrens [correspondant], Elisabeth Bon, Aurélie Goulielmakis.

With the arrival of new generations of sequencers, laboratories, at a lower cost, can be sequenced groups of genomes. You can no longer manually annotate these genomes. The YAGA (Yeast Automatic Genome Annotation) software's objective is to annotate a raw sequence syntactically and functionally as well as generate EMBL files for publication. The annotation takes into account data from comparative genomics, such as protein family profiles.

After determining the constraints of the annotation, the YAGA software can automatically annotate *de novo* all genomes from their raw sequences.The predictors used by the YAGA software can also take into account the data RNAseq to reinforce the prediction of genes.The current settings of the software are intended for annotation of the genomes of yeast, but the software is adaptable for all types of species, and has been trained and used for the annotation of bacterial genomes.

5.4. BioRica: Multi-scale Stochastic Modeling

Participants: David James Sherman [correspondant], Rodrigo Assar Cuevas.

BioRica is a high-level modeling framework integrating discrete and continuous multi-scale dynamics within the same semantics field. A model in BioRica node is hierarchically composed of nodes, which may be existing models. Individual nodes can be of two types:

- Discrete nodes are composed of states, and transitions described by constrained events, which can be non deterministic. This captures a range of existing discrete formalisms (Petri nets, finite automata, etc.). Stochastic behavior can be added by associating the likelihood that an event fires when activated. Markov chains or Markov decision processes can be concisely described. Timed behavior is added by defining the delay between an event's activation and the moment that its transition occurs.
- Continuous nodes are described by ODE systems, potentially a hybrid system whose internal state flows continuously while having discrete jumps.

The system has been implemented as a distributable software package

The BioRica compiler reads a specification for hierarchical model and compiles it into an executable simulator. The modeling language is a stochastic extension to the AltaRica Dataflow language, inspired by work of Antoine Rauzy. Input parsers for SBML 2 version 4 are curently being validated. The compiled code uses the Python runtime environment and can be run stand-alone on most systems [40].

For more information see biorica.gforge.inria.fr, the BioRica Gforge web site. BioRica was developed as an Inria Technology Development Action (ADT).

5.5. Pathtastic: Inference of whole-genome metabolic models

Participants: David James Sherman [correspondant], Pascal Durrens, Nicolás Loira, Tiphaine Martin, Anna Zhukova.

Pathtastic is a software tool for inferring whole-genome metabolic models for eukaryote cell factories. It is based on *metabolic scaffolds*, abstract descriptions of reactions and pathways on which inferred reactions are hung are are eventually connected by an interative mapping and specialization process. Scaffold fragments can be repeatedly used to build specialized subnetworks of the complete model.

Pathtastic uses a consensus procedure to infer reactions from complementary genome comparisons, and an algebra for assisted manual editing of pathways.

For more information see pathtastic.gforge.inria.fr, the Pathtastic Gforge web site.

5.6. Génolevures On Line: Comparative Genomics of Yeasts

Participants: Pascal Durrens [correspondant], Natalia Golenetskaya, Tiphaine Martin, David James Sherman.

The Génolevures online database provides tools and data for exploring the annotated genome sequences of more than 20 genomes, determined and manually annotated by the Génolevures Consortium to facilitate comparative genomic studies of hemiascomycetous yeasts. Data are presented with a focus on relations between genes and genomes: conservation of genes and gene families, speciation, chromosomal reorganization and synteny. The Génolevures site includes an area for specific studies by members of its international community.

Génolevures online uses the MAGUS system for genome navigation, with project-specific extensions developed by David Sherman, Pascal Durrens, and Tiphaine Martin. An advanced query system for data mining in Génolevures is being developed by Natalia Golenetskaya. The contents of the knowledge base are expanded and maintained by the CNRS through GDR 2354 Génolevures. Technical support for Génolevures On Line is provided the CNRS through UMR 5800 LaBRI.

For more information see genolevures.org, the Génolevures web site.

MNEMOSYNE Team

5. Software

5.1. Positioning

Our previous works in the domain of well-defined distributed asynchronous adaptive computations [32], [29], [34] have already made us define a library (DANA [3]), closely related to both the notion of artificial neural networks and cellular automata. From a conceptual point of view, the computational paradigm supporting the library is grounded on the notion of a unit that is essentially a (vector of) potential that can vary along time under the influence of other units and learning. Those units can organized into layers, maps and network.

More generally, we gather in the middleware EnaS (that stands for *Event Neural Assembly Simulation*; cf. http://gforge.inria.fr/projects/enas) our numerical and theoretical developments, allowing to simulate and analyze so called "event neural assemblies". Enas has been designed as a plug-in for our simulators (e.g. DANA or MVASpike) as other existing simulators (via the NeuralEnsemble meta-simulation platform) and additional modules for computations with neural unit assembly on standard platforms (e.g. Python or the Scilab platform).

We will also have to interact with the High Performance Computing (HPC) community, since having large scale simulations at that mesoscopic level is an important challenge in our systemic view of computational neuroscience. Our approach implies to emulate the dynamics of thousands, or even millions, of integrated computational units, each of them playing the role of a whole elementary neural circuit (e.g. the microcolumn for the cortex). Mesoscopic models are considered in such an integrative approach, in order to exhibit global dynamical effect that would be hardly reachable by compartment models involving membrane equations or even spiking neuron networks.

The vast majority of high performance computing softwares for computational neuroscience addresses sub-neural or neural models [19], but coarser grained population models are also demanding for large scale simulations, with fully distributed computations, without global memory or time reference, as it is specified in (*cf.* § 3.2).

5.2. Dana

Participant: Nicolas Rougier.

DANA [28] is a python framework (http://dana.loria.fr) whose computational paradigm is grounded on the notion of a unit that is essentially a set of time dependent values varying under the influence of other units via adaptive weighted connections. The evolution of a unit's value are defined by a set of differential equations expressed in standard mathematical notation which greatly ease their definition. The units are organized into groups that form a model. Each unit can be connected to any other unit (including itself) using a weighted connection. The DANA framework offers a set of core objects needed to design and run such models. The modeler only has to define the equations of a unit as well as the equations governing the training of the connections. The simulation is completely transparent to the modeler and is handled by DANA. This allows DANA to be used for a wide range of numerical and distributed models as long as they fit the proposed framework (e.g. cellular automata, reaction-diffusion system, decentralized neural networks, recurrent neural networks, kernel-based image processing, etc.).

5.3. ENAS: Event Neural Assembly Simulation

Participants: Frédéric Alexandre, Nicolas Rougier, Thierry Viéville.

EnaS (that stands for “Event Neural Assembly Simulation”) is a middleware implementing our last numerical and theoretical developments, allowing to simulate and analyze so called "event neural assemblies". The recent achievements include (in collaboration with the Neuromathcomp EPI): spike trains statistical analysis via Gibbs distributions, spiking network programing for exact event’s sequence restitution, discrete neural field parameters algorithmic adjustments and time-constrained event-based network simulation reconciling clock and event based simulation methods. It has been designed as plug-in for our simulators (e.g. DANA or Mvaspike) as other existing simulators (via the NeuralEnsemble meta-simulation platform) and additional modules for computations with neural unit assembly on standard platforms (e.g. Python or the Scilab platform).

5.4. Virtual Enaction

Participants: Frédéric Alexandre, André Garenne, Nicolas Rougier, Thierry Viéville.

The computational models studied in this project have applications that extend far beyond what is possible to experiment yet in human or non-human primate subjects. Real robotics experimentations are also impaired by rather heavy technological constraints; for instance, it is not easy to dismantle a given embedded system in the course of emerging ideas. The only versatile environment in which such complex behaviors can be studied both globally and at the level of details of the available modeling is a virtual environment, as in video games, Such a system can be implemented as «brainy-bot» (a programmed player based on our knowledge of the brain architecture) which goal is to survive in a complete manipulable environment.

In order to attain this rather ambitious objective we are going to both (i) deploy an existing open-source video game middleware in order to be able to shape the survival situation to be studied and (ii) revisit the existing models in order to be able to integrate them as an effective brainy-bot. It will consist of a platform associated to a scenario that would be the closest possible to a survival situation (foraging, predator-prey relationship, partner approach to reproduction) and in which it would be easy to integrate an artificial agent with sensory inputs (visual, touch and smell), emotional and somatosensory cues (hunger, thirst, fear, ..) and motor outputs (movement, gesture, ..) connected to a "brain" whose architecture will correspond to the major anatomical regions involved in the issues of learning and action selection (cortex areas detailed here, basal ganglia, hippocampus, and areas dedicated to sensorimotor processes). The internal game clock will be slowed down enough to be able to run non trivial brainy-bot implementations.

CEPAGE Project-Team

5. Software

5.1. SimGrid

Participants: Przemyslaw Uznanski, Lionel Eyraud-Dubois [correspondant].

SimGrid (http://simgrid.gforge.inria.fr/) SimGrid is a toolkit that provides core functionalities for the simulation of distributed applications in heterogeneous distributed environments. The specific goal of the project is to facilitate research in the area of parallel and distributed large scale systems, such as Grids, P2P systems and clouds. Its use cases encompass heuristic evaluation, application prototyping or even real application development and tuning. It is based on experimentally validated models, and features very high scalability, which allows to perform very large scale simulations. It is used by over a hundred academic users all over the world, and has been used in about one hundred scientific articles.

CEPAGE has contributed to this software by participating in the management of the project and in many design decisions. We also implemented simpler models, based on our works in this area, allowing a better scalability while keeping a reasonable precision.

Software assessment : A-4, SO-4, SM-4, EM-3, SDL-5.
Contribution : DA-2, CD-2, MS-2, TPM-3.

5.2. Hubble

Participants: Ludovic Courtès, Nicolas Bonichon [correspondant].

Hubble is implemented in Scheme, using GNU Guile version 2. Details of the simulation, such as keeping track of processor occupation and network usage, are taken care of by SimGrid, a toolkit for the simulation of distributed applications in heterogeneous distributed environments.

The input to Hubble is an XML description of the DAG of build tasks. For each task, a build duration and the size in bytes of the build output are specified. For our evaluation purposes, we collected this data on a production system, the http://hydra.nixos.org/ build farm hosted at the Technical University of Delft. The DAG itself is the snapshot of the Nix Package Collection (Nixpkgs) corresponding to this data. Hubble has its own in-memory representation of the DAG in the form of a purely functional data structure.

The Nixpkgs DAG contains fixed-output nodes, i.e., nodes whose output is known in advance and does not require any computation. These nodes are typically downloads of source code from external web sites. The raw data collected on http://hydra.nixos.org/ specifies a non-zero duration for these nodes, which represents the time it took to perform the download. This duration info is irrelevant in our context, since they don't require any computation, and Hubble views these nodes as instantaneous.

See also the web page http://hubble.gforge.inria.fr/.

Software assessment: A-3, SO-3, SM-2, EM-1, SDL-2.
Contribution: DA-4, CD-4, MS-4, TPM-4.

5.3. Gengraph

Participant: Cyril Gavoille [correspondant].

This is a command-line tool for generating graphs. There are several output formats, includes the dot format from GraphViz. It generates also .pdf files for visualization. Several graph algorithms have been implemented (diameter, connectivity, treewidth, etc.) which can be tested on the graphs. The software has been originally designed for teaching purpose so that students can test their project algorithms on many non trivial families like random geometric graphs, graphs of given density, given treewidth. It is also used for research purpose, in particular the exhaustive search results in the Emilie Diot's thesis are based on `gengraph`. The program can filter a list of graphs based to many criteria, as for instance it can extract all graphs of a given list that are 2-connected, of diameter at least four, and that exclude some minor (or some induced subgraph).

Currently, more than 100 parametrized graph families are implemented, supporting simple operators like complementation, random edge/vertex removal, and others. The source has more than 10,000 lines including a command-line documentation of 2,000 lines. The single source file is available at http://dept-info.labri.fr/~gavoille/gengraph.c

Software assessment: A-3, SO-3, SM-2, EM-2, SDL-2.
Contribution: DA-4, CD-4, MS-4, TPM-4.

5.4. Bedibe

Participants: Lionel Eyraud-Dubois [correspondant], Przemyslaw Uznanski.

Bedibe (Benchmarking Distributed Bandwidth Estimation) is a software to compare different models for bandwidth estimation on the Internet, and their associated instantiation algorithms. The goal is to ease the development of new models and algorithms, and the comparison with existing solutions. The development of this software is just starting.

See also the web page http://bedibe.gforge.inria.fr/.

Software assessment : A-1-up2, SO-3, SM-1-up2, EM-2, SDL-1-up2.

5.5. MineWithRounds

Participants: Sofian Maabout [correspondant], Nicolas Hanusse.

The software implements a parallel algorithm aiming at computing *Borders* that's sets of maximal/minimal subsets of objects satisfying some anti-monotone condition. It is implemented in `C++` together with the `openMP` library to exploit multi-core machines. In its current status, it outperforms state of the art implementations addressing the Maximal Frequent Itemsets problem.

Software assessment: A-2, SO-4, SM-2, EM-2, SDL-2.
Contribution: DA-4, CD-4, MS-4, TPM-4.

HIEPACS Project-Team

5. Software

5.1. Introduction

We describe in this section the software that we are developing. The first two (MaPHyS and ScalFMM) will be the main milestones of our project. The other software developments will be conducted in collaboration with academic partners or in collaboration with some industrial partners in the context of their private R&D or production activities. For all these software developments, we will use first the various (very) large parallel platforms available through CERFACS and GENCI in France (CCRT, CINES and IDRIS Computational Centers), and next the high-end parallel platforms that will be available via European and US initiatives or projects such that PRACE.

5.2. MaPHyS

MaPHyS (Massivelly Parallel Hybrid Solver) is a software package whose proptotype was initially developed in the framework of the PhD thesis of Azzam Haidar (CERFACS) and futher consolidated thanks to the ANR-CIS Solstice funding. This parallel linear solver couples direct and iterative approaches. The underlying idea is to apply to general unstructured linear systems domain decomposition ideas developed for the solution of linear systems arising from PDEs. The interface problem, associated with the so called Schur complement system, is solved using a block preconditioner with overlap between the blocks that is referred to as Algebraic Additive Schwarz.

The MaPHyS package is very much a first outcome of the research activity described in Section 3.3 . Finally, MaPHyS is a preconditioner that can be used to speed-up the convergence of any Krylov subspace method. We forsee to either embed in MaPHyS some Krylov solvers or to release them as standalone packages, in particular for the block variants that will be some outcome of the studies discussed in Section 3.3 .

5.3. EPSN

EPSN (Environment for Computational Steering) is a software environment for the steering of legacy parallel-distributed simulations with simple GUI or more complex (possibly parallel) visualization programs (see Figure 1). In order to make a legacy simulation steerable, the user annotates the sourcecode with the EPSN API. These annotations provide the EPSN environment with two kinds of information: the description of the program structure according to a Hierarchical Task Model (HTM) and the description of the distributed data that will be remotely accessible. EPSN provides a distributed data model, that handles common scientific objects such as parameters, structured grids, particles/atoms and unstructured meshes. It is then possible to dynamically connect EPSN with a client program, that provides a GUI with some visualization & interaction features, as for instance SIMONE (SImulation MONitoring for Epsn). Once a client is connected, it interacts with the simulation via EPSN API. It is possible : 1) to control the execution flow of the remote simulation; 2) to access/modify its data onthefly; and 3) finally to invoke advanced user-defined routines in the simulation. The current version of EPSN is fully based on CORBA for communication on heterogeneous system and VTK/Paraview for visualization. A new release of EPSN, that will be fully based on MPI to handle efficient communication, is currently under development. A prototype is already working.

EPSN has been supported by the ACI-GRID program (grant number PPL02-03), the ARC RedGRID, the ANR MASSIM (grant number ANR-05-MMSA-0008-03) and the ANR CIS NOSSI (2007). More informations are available on our web site: http://www.labri.fr/projet/epsn. This software is publicy available at Inria Gforge (http://epsn.gforge.inria.fr).

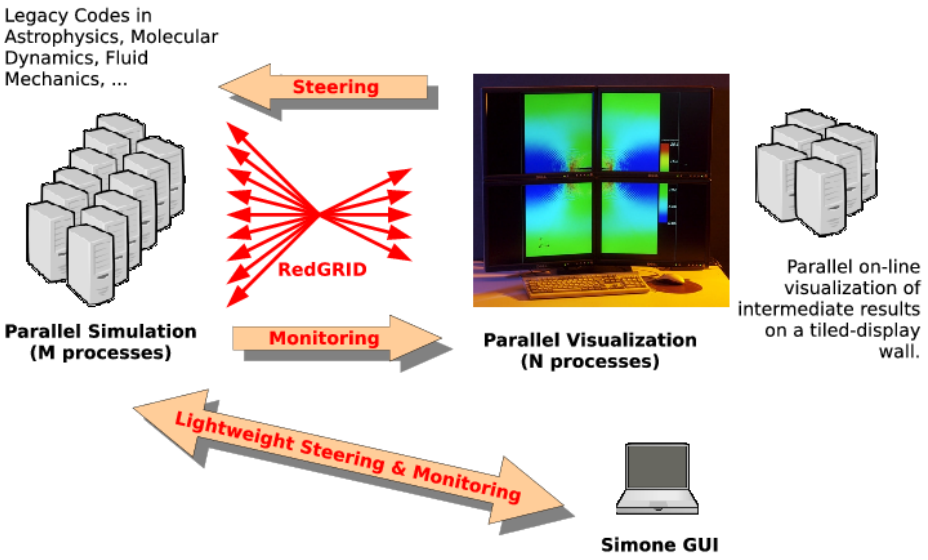

Figure 1. EPSN: software environment for $M \times N$ computational steering.

5.4. MPICPL

MPICPL (MPI CouPLing) is a software library dedicated to the coupling of parallel legacy codes, that are based on the well-known MPI standard. It proposes a lightweight and comprehensive programing interface that simplifies the coupling of several MPI codes (2, 3 or more). MPICPL facilitates the deployment of these codes thanks to the *mpicplrun* tool and it interconnects them automatically through standard MPI inter-communicators. Moreover, it generates the universe communicator, that merges the world communicators of all coupled-codes. The coupling infrastructure is described by a simple XML file, that is just loaded by the *mpicplrun* tool. Future releases will incorporate new features for checkpoint/restart and dynamic parallel code connection.

MPICPL was developed by the Inria HiePACS project-team for the purpose of the ANR CIS NOSSI. It uses advanced features of MPI2 standard. The framework is publicy available at Inria Gforge: http://mpicpl.gforge.inria.fr.

5.5. MONIQA

MONIQA (MONitoring graphic user Interface for QM/MM Applications) is a GUI specially designed for the monitoring & steering of the QM/MM application in the ANR CIS NOSSI project. It is based on Tulip, a graph visualization software http://tulip.labri.fr), used to display atoms and molecules. It proposes two working modes : offline or online. The offline mode is mainly used to load input files of DL_POLY & Siesta, and to prepare the quantum region for the QM/MM coupling. In online mode, the end-user can monitor & interact with the running QM/MM application thanks to EPSN. It is thus possible to visualize molecular and physical data (distances, angles, charges, energies), and to change simulation parameters on-the-fly, such as the target temperature of the system, thermo or barostat parameters, verbosity of output, ... MONIQA is based on QT4. It was developed specifically for the ANR NOSSI project and is available (restricted access) at Inria Gforge: http://nossi.gforge.inria.fr.

5.6. ScalFMM

`ScalFMM` intends to offer all the functionalities needed to perform large parallel simulations while enabling an easy customization of the simulation components: kernels, particles and cells. It works in parallel in a shared/distributed memory model using OpenMP and MPI. The software architecture has been designed with two major objectives: being easy to maintain and easy to understand. The code is extremely documented and the naming convention fully respected. Driven by its user-oriented philosophy, `ScalFMM` is using CMAKE as a compiler/installer tool. Even if `ScalFMM` is written in C++ it will support a C and fortran API soon.

`ScalFMM` (Parallel Fast Multipole Library for Large Scale Simulations) is a software library to simulate N-body interactions using the Fast Multipole Method.

The library offers two methods to compute interactions between bodies when the potential decays like $1/r$. The first method is the classical FMM based on spherical harmonic expansions and the second is the Black-Box method which is an independent kernel formulation (introduced by E. Darve @ Stanford). With this method, we can now easily add new non oscillatory kernels in our library. For the classical method, two approaches are used to decrease the complexity of the operators. We consider either matrix formulation that allows us to use BLAS routines or rotation matrix to speed up the M2L operator.

`ScalFMM` intends to offer all the functionalities needed to perform large parallel simulations while enabling an easy customization of the simulation components: kernels, particles and cells. It works in parallel in a shared/distributed memory model using OpenMP and MPI. The software architecture has been designed with two major objectives: being easy to maintain and easy to understand. There is two main parts: 1) the management of the octree and the parallelization of the method ; 2) the kernels. This new architecture allow us to easily add new FMM algorithm or kernels and new paradigm of parallelization.

The `ScalFMM` package is available at scalfmm.gforge.inria.fr.

5.7. Other software

These software packages are or will be developed in collaboration with some academic partners (LIP6, LaBRI, CPMOH, IPREM, EPFL) or in collaboration with industrial partners (CEA, TOTAL, EDF) in the context of their private R&D or production activities.

- For the materials physics applications, a lot of development will be done in the context of ANR projects (NOSSI and OPTIDIS, see Section 4.2) in collaboration with LaBRI, CPMOH, IPREM, EPFL and with CEA Saclay and Bruyère-le-Châtel.

PHOENIX Project-Team

5. Software

5.1. DiaSuite: a Development Environment for Sense/Compute/Control Applications

Participants: Charles Consel [correspondent], Julien Bruneau, Amélie Marzin, Damien Martin-Guillerez, Emilie Balland.

Despite much progress, developing a pervasive computing application remains a challenge because of a lack of conceptual frameworks and supporting tools. This challenge involves coping with heterogeneous devices, overcoming the intricacies of distributed systems technologies, working out an architecture for the application, encoding it in a program, writing specific code to test the application, and finally deploying it.

DIASUITE is a suite of tools covering the development life-cycle of a pervasive computing application:

- *Defining an application area.* First, an expert defines a catalog of entities, whether hardware or software, that are specific to a target area. These entities serve as building blocks to develop applications in this area. They are gathered in a taxonomy definition, written in the taxonomy layer of the DIASPEC language.
- *Designing an application.* Given a taxonomy, the architect can design and structure applications. To do so, the DIASPEC language provides an application design layer [40]. This layer is dedicated to an architectural pattern commonly used in the pervasive computing domain [30]. Describing the architecture application allows to further model a pervasive computing system, making explicit its functional decomposition.
- *Implementing an application.* We leverage the taxonomy definition and the architecture description to provide dedicated support to both the entity and the application developers. This support takes the form of a Java programming framework, generated by the DIAGEN compiler. The generated programming framework precisely guides the developer with respect to the taxonomy definition and the architecture description. It consists of high-level operations to discover entities and interact with both entities and application components. In doing so, it abstracts away from the underlying distributed technologies, providing further separation of concerns.
- *Testing an application.* DIAGEN generates a simulation support to test pervasive computing applications before their actual deployment. An application is simulated in the DIASIM tool, without requiring any code modification. DIASIM provides an editor to define simulation scenarios and a 2D-renderer to monitor the simulated application. Furthermore, simulated and actual entities can be mixed. This hybrid simulation enables an application to migrate incrementally to an actual environment.
- *Deploying a system.* Finally, the system administrator deploys the pervasive computing system. To this end, a distributed systems technology is selected. We have developed a back-end that currently targets the following technologies: Web Services, RMI, SIP and OSGI. This targeting is transparent for the application code. The variety of these target technologies demonstrates that our development approach separates concerns into well-defined layers.

This development cycle is summarized in the Figure 1 .

See also the web page http://diasuite.inria.fr.

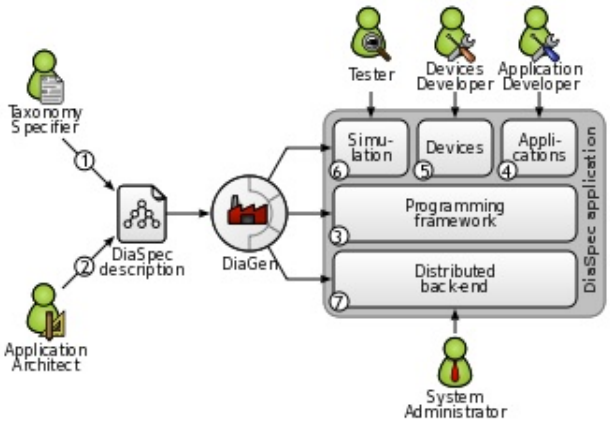

Figure 1. DIASUITE *Development Cycle*

5.1.1. DiaSpec: a Domain-Specific Language for Networked Entities

The core of the DIASUITE development environment is the domain specific language called DIASPEC and its compiler DIAGEN:

- DIASPEC is composed of two layers:
  - The *Taxonomy Layer* allows the declaration of entities that are relevant to the target application area. An entity consists of sensing capabilities, producing data, and actuating capabilities, providing actions. Accordingly, an entity description declares a data source for each one of its sensing capabilities. As well, an actuating capability corresponds to a set of method declarations. An entity declaration also includes attributes, characterizing properties of entity instances. Entity declarations are organized hierarchically allowing entity classes to inherit attributes, sources and actions. A taxonomy allows separation of concerns in that the expert can focus on the concerns of cataloging area-specific entities. The entity developer is concerned about mapping a taxonomical description into an actual entity, and the application developer concentrates on the application logic.
  - The *Architecture Layer* is based on an architectural pattern commonly used in the pervasive computing domain [30]. It consists of context components fueled by sensing entities. These components process gathered data to make them amenable to the application needs. Context data are then passed to controller components that trigger actions on entities. Using an architecture description enables the key components of an application to be identified, allowing their implementation to evolve with the requirements (*e.g.,* varying light management implementations in a controller component to optimize energy consumption).
- DIAGEN is the DIASPEC compiler that performs both static and runtime verifications over DIASPEC declarations and produces a dedicated programming framework that guides and eases the implementation of components. The generated framework is independent of the underlying distributed technology. As of today, DIAGEN supports multiple targets: Local, RMI, SIP, Web Services and OSGI.

5.1.2. DiaSim: a Parametrized Simulator for Pervasive Computing Applications

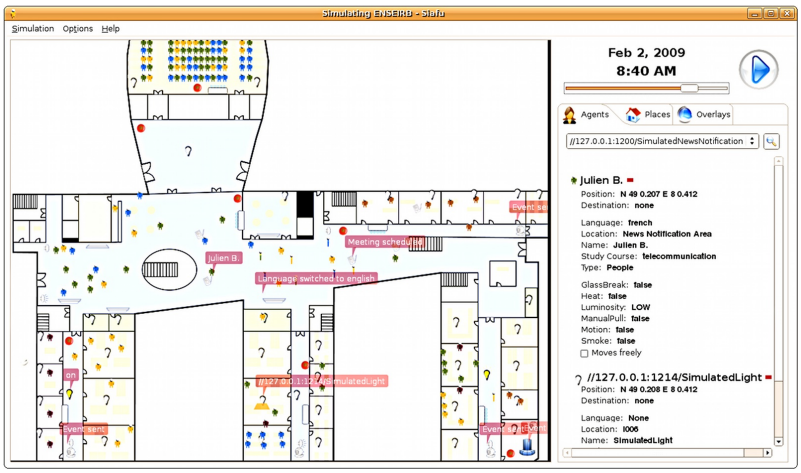

Figure 2. A screenshot of the DIASIM *simulator*

Pervasive computing applications involve both software and integration concerns. This situation is problematic for testing pervasive computing applications because it requires acquiring, testing and interfacing a variety of software and hardware entities. This process can rapidly become costly and time-consuming when the target environment involves many entities.

To ease the testing of pervasive applications, we are developing a simulator for pervasive computing applications: DIASIM. To cope with widely heterogeneous entities, DIASIM is parameterized with respect to a DIASPEC specification describing a target pervasive computing environment. This description is used to generate with DIAGEN both a programming framework to develop the simulation logic and an emulation layer to execute applications. Furthermore, a simulation renderer is coupled to DIASIM to allow a simulated pervasive system to be visually monitored and debugged. The simulation renderer is illustrated in Figure 2 .

5.2. DiaSuiteBox: an Open Service Platform

Participants: Julien Bruneau [correspondent], Damien Martin-Guillerez, Charles Consel, Emilie Balland.

The DiaSuiteBox platform runs an open-ended set of applications leveraging a range of appliances and web services. Our solution consists of a dedicated development environment, a certifying application store, and a lightweight runtime platform. This solution is based on the DIASUITE project.

The DiaSuiteBox platform can be embedded in a small plug-computer or deployed in the cloud. Thanks to the application store and the developer community, the platform is fed by a full offer of new innovative applications. During the submission process, an application is automatically analyzed and checked in order to be certified. The user is ensured of the behavior of its applications are innocuous and correct beside the provided information. This box relies on several technology standards like UPnP, Bluetooth, USB, etc. As shown in Figure 3 , this platform can be easily extended by plugging appliances directly on the box or by connecting devices on the local network.

See also the web page http://diasuitebox.inria.fr.

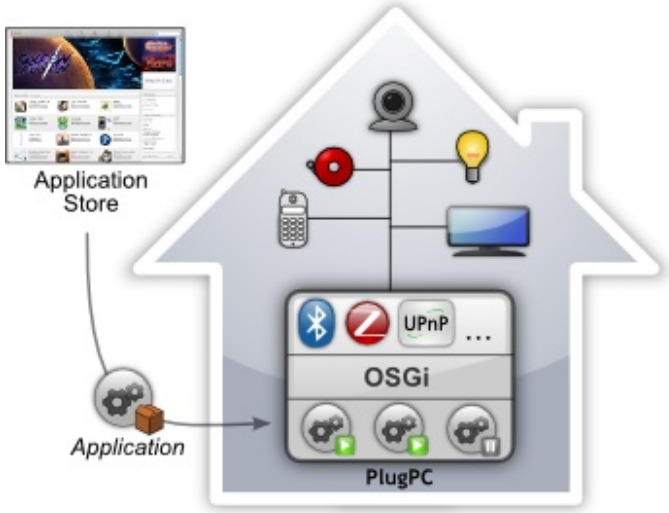

Figure 3. DiaSuiteBox platform architecture

RUNTIME Project-Team

5. Software

5.1. Common Communication Interface

Participant: Brice Goglin.

- The *Common Communication Interface* aims at offering a generic and portable programming interface for a wide range of networking technologies (Ethernet, InfiniBand, ...) and application needs (MPI, storage, low latency UDP, ...).
- CCI is developed in collaboration with the *Oak Ridge National Laboratory* and several other academics and industrial partners.
- CCI is in early development and currently composed of 19 000 lines of C.
- http://www.cci-forum.org

5.2. Hardware Locality

Participants: Brice Goglin, Samuel Thibault.

- *Hardware Locality* (HWLOC) is a library and set of tools aiming at discovering and exposing the topology of machines, including processors, cores, threads, shared caches, NUMA memory nodes and I/O devices.
- It builds a widely-portable abstraction of these resources and exposes it to the application so as to help them adapt their behavior to the hardware characteristics.
- HWLOC targets many types of high-performance computing applications [2], from thread scheduling to placement of MPI processes. Most existing MPI implementations, several resource managers and task schedulers already use HWLOC.
- HWLOC is developed in collaboration with the OPEN MPI project. The core development is still mostly performed by Brice GOGLIN and Samuel THIBAULT from the RUNTIME team-project, but many outside contributors are joining the effort, especially from the OPEN MPI and MPICH2 communities.
- HWLOC is composed of 40 000 lines of C.
- http://runtime.bordeaux.inria.fr/hwloc/

5.3. KNem

Participant: Brice Goglin.

- KNEM (*Kernel Nemesis*) is a Linux kernel module that offers high-performance data transfer between user-space processes.
- KNEM offers a very simple message passing interface that may be used when transferring very large messages within point-to-point or collective MPI operations between processes on the same node.
- Thanks to its kernel-based design, KNEM is able to transfer messages through a single memory copy, much faster than the usual user-space two-copy model.
- KNEM also offers the optional ability to offload memory copies on INTEL I/O AT hardware which improves throughput and reduces CPU consumption and cache pollution.
- KNEM is developed in collaboration with the MPICH2 team at the Argonne National Laboratory and the OPEN MPI project. These partners already released KNEM support as part of their MPI implementations.
- KNEM is composed of 7000 lines of C. Its main contributor is Brice GOGLIN.
- http://runtime.bordeaux.inria.fr/knem/

5.4. Marcel

Participants: Olivier Aumage, Yannick Martin, Samuel Thibault.

- MARCEL is the two-level thread scheduler (also called N:M scheduler) of the PM^2 software suite.
- The architecture of MARCEL was carefully designed to support a large number of threads and to efficiently exploit hierarchical architectures (e.g. multicore chips, NUMA machines).
- MARCEL provides a *seed* construct which can be seen as a precursor of thread. It is only when the time comes to actually run the seed that MARCEL attempts to reuse the resources and the context of another, dying thread, significantly saving management costs.
- In addition to a set of original extensions, MARCEL provides a POSIX-compliant interface which thus permits to take advantage of it by just recompiling unmodified applications or parallel programming environments (API compatibility), or even by running already-compiled binaries with the Linux NPTL ABI compatibility layer.
- For debugging purpose, a trace of the scheduling events can be recorded and used after execution for generating an animated movie showing a replay of the execution.
- The MARCEL thread scheduling library is made of 80 000 lines of code.
- http://runtime.bordeaux.inria.fr/marcel/
- Marcel has been supported for 2 years (2009-2011) by the Inria ADT Visimar.

5.5. ForestGOMP

Participants: Olivier Aumage, Yannick Martin, Pierre-André Wacrenier.

- FORESTGOMP is an OPENMP environment based on both the GNU OPENMP run-time and the MARCEL thread library.
- It is designed to schedule efficiently nested sets of threads (derived from nested parallel regions) over hierarchical architectures so as to minimize cache misses and NUMA penalties.
- The FORESTGOMP runtime generates nested MARCEL bubbles each time an OPENMP parallel region is encountered, thereby grouping threads sharing common data.
- Topology-aware scheduling policies implemented by BUBBLESCHED can then be used to dynamically map bubbles onto the various levels of the underlying hierarchical architecture.
- FORESTGOMP allowed us to validate the BUBBLESCHED approach with highly irregular, fine grain, divide-and-conquer parallel applications.
- http://runtime.bordeaux.inria.fr/forestgomp/

5.6. Open-MX

Participant: Brice Goglin.

- The OPEN-MX software stack is a high-performance message passing implementation for any generic ETHERNET interface.
- It was developed within our collaboration with Myricom, Inc. as a part of the move towards the convergence between high-speed interconnects and generic networks.
- OPEN-MX exposes the raw ETHERNET performance at the application level through a pure message passing protocol.
- While the goal is similar to the old GAMMA stack [35] or the recent iWarp [34] implementations, OPEN-MX relies on generic hardware and drivers and has been designed for message passing.
- OPEN-MX is also wire-compatible with Myricom MX protocol and interface so that any application built for MX may run on any machine without Myricom hardware and talk other nodes running with or without the native MX stack.
- OPEN-MX is also an interesting framework for studying next-generation hardware features that could help ETHERNET hardware become legacy in the context of high-performance computing. Some innovative message-passing-aware stateless abilities, such as multiqueue binding and interrupt coalescing, were designed and evaluated thanks to OPEN-MX [5].
- Brice GOGLIN is the main contributor to OPEN-MX. The software is already composed of more than 45 000 lines of code in the Linux kernel and in user-space.
- http://open-mx.org/

5.7. StarPU

Participants: Cédric Augonnet, Olivier Aumage, Nicolas Collin, Nathalie Furmento, Cyril Roelandt, Ludovic Stordeur, Samuel Thibault, Ludovic Courtès.

- StarPU permits high performance libraries or compiler environments to exploit heterogeneous multicore machines possibly equipped with GPGPUs or Cell processors.
- StarPU offers a unified offloadable task abstraction named codelet.In case a codelet may run on heterogeneous architectures, it is possible to specify one function for each architectures (e.g. one function for CUDA and one function for CPUs).
- StarPU takes care to schedule and execute those codelets as efficiently as possible over the entire machine. A high-level data management library enforces memory coherency over the machine: before a codelet starts (e.g. on an accelerator), all its data are transparently made available on the compute resource.
- StarPU obtains portable performances by efficiently (and easily) using all computing resources at the same time.
- StarPU also takes advantage of the heterogeneous nature of a machine, for instance by using scheduling strategies based on auto-tuned performance models.
- StarPU can also leverage existing parallel implementations, by supporting *parallel tasks*, which can be run concurrently over the machine.
- StarPU provides a *reduction* mode, which permit to further optimize data management when results have to be reduced.
- StarPU provides integration in MPI clusters through a lightweight DSM over MPI.
- StarPU comes with a plug-in for the GNU Compiler Collection (GCC), which extends languages of the C family with syntactic devices to describe StarPU's main programming concepts in a concise, high-level way.
- http://runtime.bordeaux.inria.fr/StarPU/

5.8. NewMadeleine

Participants: Alexandre Denis, François Trahay, Raymond Namyst.

- NewMadeleine is communication library for high performance networks, based on a modular architecture using software components.
- The NewMadeleine optimizing scheduler aims at enabling the use of a much wider range of communication flow optimization techniques such as packet reordering or cross-flow packet aggregation.
- NewMadeleine targets applications with irregular, multiflow communication schemes such as found in the increasingly common application conglomerates made of multiple programming environments and coupled pieces of code, for instance.
- It is designed to be programmable through the concepts of optimization *strategies*, allowing experimentations with multiple approaches or on multiple issues with regard to processing communication flows, based on basic communication flows operations such as packet merging or reordering.
- The reference software development branch of the NewMadeleine software consists in 90 000 lines of code. NewMadeleine is available on various networking technologies: Myrinet, Infiniband, Quadrics and Ethernet. It is developed and maintained by Alexandre Denis.
- http://runtime.bordeaux.inria.fr/newmadeleine/

5.9. PadicoTM

Participant: Alexandre Denis.

- PadicoTM is a high-performance communication framework for grids. It is designed to enable various middleware systems (such as CORBA, MPI, SOAP, JVM, DSM, etc.) to utilize the networking technologies found on grids.
- PadicoTM aims at decoupling middleware systems from the various networking resources to reach transparent portability and flexibility.
- PadicoTM architecture is based on software components. Puk (the PadicoTM micro-kernel) implements a light-weight high-performance component model that is used to build communication stacks.
- PadicoTM component model is now used in NEWMADELEINE. It is the cornerstone for networking integration in the projects "LEGO" and "COOP" from the ANR.
- PadicoTM is composed of roughly 60 000 lines of C.
- PadicoTM is registered at the APP under number IDDN.FR.001.260013.000.S.P.2002.000.10000.
- http://runtime.bordeaux.inria.fr/PadicoTM/

5.10. MAQAO

Participants: Denis Barthou, Andres Charif-Rubial.

- MAQAO is a performance tuning tool for OpenMP parallel applications. It relies on the static analysis of binary codes and the collection of dynamic information (such as memory traces). It provides hints to the user about performance bottlenecks and possible workarounds.
- MAQAO relies on binary codes and inserts probes for instrumention directly inside the binary. There is no need to recompile. The static/dynamic approach of MAQAO analysis is the main originality of the tool, combining performance model with values collected through instrumentation.
- MAQAO has a static performance model for x86 architecture and Itanium. This model analyzes performance of the predecoder, of the decoder and of the different pipelines of the x86 architecture, in particular for SSE instructions.
- The dynamic collection of data in MAQAO enables the analysis of thread interactions, such as false sharing, amount of data reuse, runtime scheduling policy, ...
- MAQAO is in the project "ProHMPT" from the ANR. A demo of MAQAO has been made in Jan. 2010 for SME/Inria days and in Nov. 2010 at SuperComputing, Inria Booth.
- http://www.maqao.org/

5.11. QIRAL

Participant: Denis Barthou.

- QIRAL is a high level language (expressed through LaTeX) that is used to described Lattice QCD problems. It describes matrix formulations, domain specific properties on preconditionings, and algorithms.
- The compiler chain for QIRAL can combine algorithms and preconditionings, checking validity of the composition automatically. It generates OpenMP parallel code, using libraries, such as BLAS.
- This code is developped in collaboration with other teams participating to the ANR PetaQCD project.

5.12. TreeMatch

Participants: Emmanuel Jeannot, Guillaume Mercier, François Tessier.

- TREEMATCH is a library for performing process placement based on the topology of the machine and the communication pattern of the application.
- TREEMATCH provides a permutation of the processes to the processors/cores in order to minimize the communication cost of the application.
- Important features are : the number of processors can be greater than the number of applications processes ; it assumes that the topology is a tree and does not require valuation of the topology (e.g. communication speeds) ; it implements different placement algorithms that are switched according to the input size.
- TREEMATCH is integrated into various software such as the Charm++ programming environment as well as in both major open-source MPI implementations: Open MPI and MPICH2.
- TREEMATCH is available at: http://treematch.gforge.inria.fr.

FLOWERS Project-Team

5. Software

5.1. Perception Tools

Participants: David Filliat [correspondant], Natalia Lyubova, Louis-Charles Caron, Alexander Gepperth.

5.1.1. Perception Abstraction Engine

Participants: David Filliat [correspondant], Natalia Lyubova.

PAE (Perception Abstraction Engine) is a C++ library developed to provide a uniform interface to existing visual feature detector such as SIFT, SURF, MSER, superpixels, etc... Its main goal is to be able to use these various feature detectors in a "bag of feature" approach for applications such as robot localisation and object recognition. Several approach are also implemented for the visual vocabularies, in particular the fast incremental vocabularies developed in the team.

The library provide common C++ interfaces to feature detectors, visual features and visual vocabularies. A factory approach make it possible to change the feature detectors and visual vocabularies types and parameters through configuration strings, without the need to recompile. Some applications are also included in the library, in particular topological robot localization (room recognition) and visual object recognition. An Urbi interface is also provided for these modules.

5.1.2. Incremental object discovery

Participants: Natalia Lyubova [correspondant], David Filliat.

This software makes it possible to detect, model and recognize objects in a scenario of interaction between a humanoid robot and a human teacher. It is based either on standard images, or on the kinect camera to take advantage of the depth information. The software is written in C++ and relies mainly on PAE and OpenCV.

The software implements several modules: candidate object segmentation based on motion information, keypoint-based object tracking, incremental object model construction integrating multiple features (keypoints + superpixels) and object categorisation based on mutual information with robot motors (making it possible to segment robot parts, objects and humans).

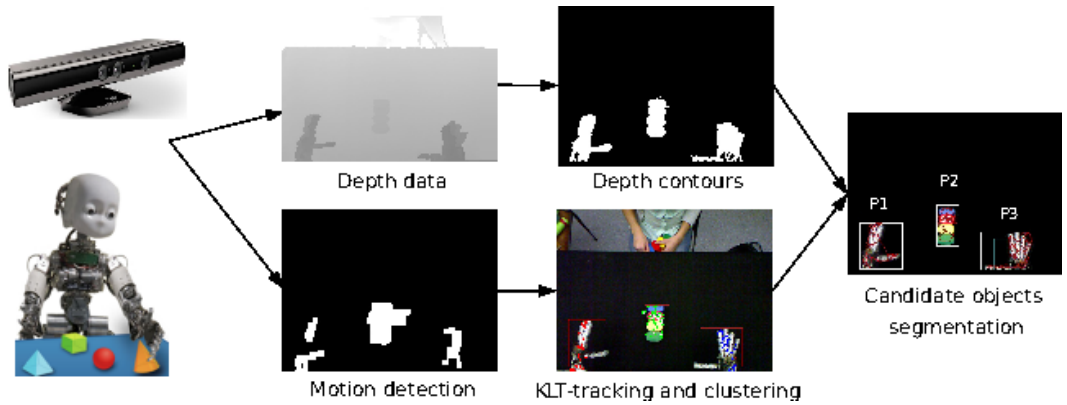

Figure 1. System Overview of the Incremental object discovery Software.

5.1.3. Object recognition from a 3-D point cloud

Participants: Louis-Charles Caron [correspondant], Alexander Gepperth, David Filliat.

This software scans the 3-D point cloud of a scene to find objects and match them against a database of known objects. The process consists in 3 stages. The segmentation step finds the objects in the point cloud, the feature extraction computes discriminating properties to be used in the classification stage for object recognition.

The segmentation is based on simple assumptions about the geometry of an indoor scene. Successive RANSACs are used to find large planes, which correspond to the floor, ceiling and walls. The cloud is stripped from the points belonging to these planes. The remaining points are clustered, meaning that close-by points are considered to form a single object.

Objects are characterized by their shape and color. Color histograms and SIFT features are computed, using the PAE library, to capture the visual appearance of the objects. Their shape is encoded by computing thousands of randomly chosen SURFLET features to construct a relative frequency histrogram.

An early classification is done using each of the 3 features separately. For the color features a bag of words approach (from PAE) is used. For the shape feature, the minimum squared distance between the object's histogram and that of all objects in the database is calculated. Classification scores are then fused by a feed-forward neural network to get the final result [39].

5.1.4. PEDDETECT: GPU-accelerated person detection demo

Participant: Alexander Gepperth [correspondant].

PEDDETECT implements real-time person detection in indoor or outdoor environments. It can grab image data directly from one or several USB cameras, as well as from pre-recorded video streams. It detects mulitple persons in 800x600 color images at frame rates of >15Hz, depending on available GPU power. In addition, it also classifies the pose of detected persons in one of the four categories "seen from the front", "seen from the back", "facing left" and "facing right". The software makes use of advanced feature computation and nonlinear SVM techniques which are accelerated using the CUDA interface to GPU programming to achieve high frame rates. It was developed in the context of an ongoing collaboration with Honda Research Institute USA, Inc.

5.2. Datasets

5.2.1. Choreography dataset

Participants: Olivier Mangin [correspondant], Haylee Fogg.

This database contains choreography motions recorded through a kinect device. These motions have a combinatorial structure: from a given set of primitive dance motions, choreographies are constructed as simultaneous execution of some of these primitive motions. Primitive dance motions are chosen from a total set of 48 motions and are spanned over one or two limbs, either the legs (e.g. walk, squat), left or right arm (e.g. wave hand, punch) or both arms (e.g. clap in hands, paddle). Complex choreographies are produced as the simultaneous demonstration of two or three of these primitive motion: either one for legs and one for both arm, or one for legs and one for each arm. The dataset has been used in the experiments from [52] for studying learning techniques allowing to identify dictionaries of motion primitives, and is publicly available at https://flowers.inria.fr/choreography_database.html.

5.3. Learning algorithms

5.3.1. RLPark - Reinforcement Learning Algorithms in JAVA

Participant: Thomas Degris [correspondant].

RLPark is a reinforcement learning framework in Java. RLPark includes learning algorithms, state representations, reinforcement learning architectures, standard benchmark problems, communication interfaces for three robots, a framework for running experiments on clusters, and real-time visualization using Zephyr. More precisely, RLPark includes:

- Online Learning Algorithms: Sarsa, Expected Sarsa, Q-Learning, On-policy and off-policy Actor-Critic with normal distribution (continuous actions) and Boltzmann distribution (discrete action), average reward actor-critic, TD, TD(λ), GTD(λ), GQ(λ), TDC
- State Representations: tile coding (with no hashing, hashing and hashing with mumur2), Linear Threshold Unit, observation history, feature normalization, radial basis functions
- Interface with Robots: the Critterbot, iRobot Create, Nao, Puppy, Dynamixel motors
- Benchmark Problems: mountain car, swing-up pendulum, random walk, continuous grid world

An example of RLpark running an online learning experiment on a reinforcement learning benchmark problem is shown in Figure 2 .

RLPark was started in spring 2009 in the RLAI group at the university of Alberta (Canada) when Thomas Degris was a postdoc in this group. RLPark is still actively used by RLAI. Collaborators and users include Adam White, Joseph Modayil and Patrick Pilarski (testing) from the University of Alberta.

RLPark has been used by Richard Sutton, a professor and iCORE chair in the department of computing science at the University of Alberta, for a demo in his invited talk *Learning About Sensorimotor Data* at the Neural Information Processing Systems (NIPS) 2011 [1]. Patrick Pilarski used RLPark for live demos on television (Breakfast Television Edmonton, CityTV, June 5th, 2012) and at TEDx Edmonton on Intelligent Artificial Limbs[2]. So far, RLPark has been used in more than a dozens of publications (see http://rlpark.github.com/publications.html for a list).

RLPark has been ported to C++ by Saminda Abeyruwan, a student of the University of Miami (United States of America). The Horde architecture in RLPark has been optimized for GPU by Clément Gehring, a student of the McGill University in Montreal (Canada).

Future developments include the implementation of additional algorithms (the Dyna architecture, back propagation in neural networks, ...). A paper is under review for the JMLR Machine Learning Open Source Software. Documentation and tutorials are included on the RLPark web site [3]. RLPark is licensed under the open source Eclipse Public License.

5.3.2. DMP-BBO Matlab library

Participant: Freek Stulp [correspondant].

The dmp_bbo (Black-Box Optimization for Dynamic Movement Primitives) Matlab library is a direct consequence of the insight that black-box optimization outperforms reinforcement learning when using policies represented as Dynamic Movement Primitives. It implements several variants of the PI^{BB} algorithm for direct policy search. It is currently being used and extended by several FLOWERS members (Manuel Lopes, Clement Moulin-Frier) and external collaborators (Jonas Buchli, Hwangbo Jemin of ETH Zurich). This code was used for the following publications: [63], [60], [62].

5.3.3. PROPRE: simulation of developmental concept formation using PYTHON

Participant: Alexander Gepperth [correspondant].

This simulation software implements the algorithms described in [24], [40]. It is available online under the URL www.gepperth.net/downloads.html. The simulation is implemented in PYTHON for easy use, yet the time-critical core functions are written in C.

[1] http://webdocs.cs.ualberta.ca/~sutton/Talks/Talks.html#sensorimotor
[2] http://www.youtube.com/watch?v=YPc-Ae7zqSo
[3] http://rlpark.github.com

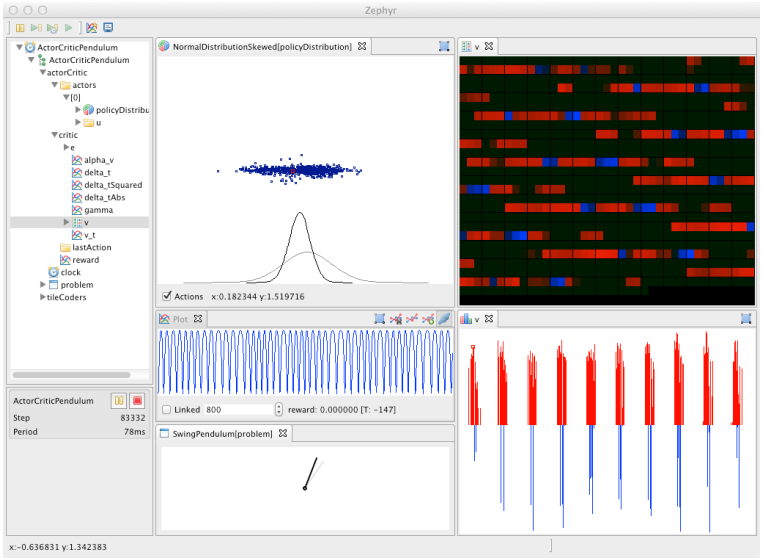

Figure 2. An example of an experiment in RLPark. Zephyr displays two views of a learned weight vector, an animation of the problem, the current policy distribution learned by the algorithm and the reward obtained by the algorithm. Videos are available at: http://rlpark.github.com.

5.3.4. pyStreamPlayer: synchronized replay of multiple sensor recordings and supplementary data

Participant: Alexander Gepperth [correspondant].

This Python software is intended to facilitate the application of machine learning algorithms by avoiding to work directly with an embodied agent but instead with data recorded in such an agent. Assuming that non-synchronous data from multiple sensors (e.g., camera, Kinect, laser etc.) have been recorded according to a flexible format defined by the pyStreamPlayer architecture, pyStreamPlayer can replay these data while retaining the exact temporal relations between different sensor measurements. As long as the current task does not involve the generation of actions, this software allows to process sensor data as if it was coming from an agent which is usually considerably easier. At the same time, pyStreamPlayer allows to replay arbitrary supplementary information such as, e.g., object information, as if it was coming from a sensor. In this way, supervision information can be stored and accessed together with sensory measurements using an unified interface. pyStreamPlayer has been used to facilitate real-world object recognition tasks, and several of the major databases in this field (CalTech Pedestrian database, HRI RoadTraffic traffic objects database, CVC person database, KITTI traffic objects database) have been converted to the pyStreamPlaer format and now serve as a source of training and test data for learning algorithms.

pyStreamPlayer has been integrated into a ROS node as well, allowing th replay and transmission across networks of distributed processes.

5.4. Software Platforms

5.4.1. Robust robotics manipulation - Object detection and tracking

Participants: Antoine Hoarau [ADT Engineer Since Nov. 2012], Freek Stulp [Supervisor], David Filliat [Supervisor].

Autonomous human-centered robots, for instance robots that assist people with disabilities, must be able to physically manipulate their environment. There is therefore a strong interest within the FLOWERS team to apply the developmental approach to robotics in particular to the acquisition of sophisticated skills for manipulation and perception. ENSTA-ParisTech has recently acquired a Meka (cf. 3) humanoid robot dedicated to human-robot interaction, and which is perfectly fitted to this research. The goal of this project is to install state-of-the-art software architecture and libraries for perception and control on the Meka robot, so that this robot can be jointly used by FLOWERS and ENSTA. In particular, we want to provide the robot with an initial set of manipulation skills.
The goal is to develop a set of demos, which demonstrate the capabilities of the Meka, and provide a basis on which researchers can base their experiments. As the robot is not yet available at ENSTA, initial work focused on the robot's environment, meaning ROS and the M3 software (provided by Meka Robotics, based on both C++ and Python scripts) and on trying to implement a simple ball-catching demo : the idea is to throw a ball toward the robot which catch it (basic human-robot interaction, combining both perception and control). Different tracking algorithms are being tried for the ball, such as Camshift, Hough Circles + Kalman Filter, or more complex LineMod (all included in OpenCV) to finally estimate its trajectory for the robot to catch it. The M3 software provided by Meka Robotics contains a simulation environment that allows us to work without the robot hardware (cf. 4 .

Figure 3. The Meka robot plateform acquired by ENSTA ParisTech

5.4.2. ErgoRobot/Flowers Field Software

Participants: Jérôme Béchu [correspondant], Pierre-Yves Oudeyer, Pierre Rouanet, Olivier Mangin, Fabien Benureau, Mathieu Lapeyre.

In the context of its participation to the exhibition "Mathematics: A Beautiful Elsewhere" at Fondation Cartier pour l'Art Contemporain in Paris (19th October 2011 to 18th March 2012), the team has elaborated and experimented a robotic experimental set-up called "Ergo-Robots/FLOWERS Fields". This set-up is not only a way to share our scientific research on curiosity-driven learning, human-robot interaction and language acquisition with the general public, but, as described in the Results and Highlights section, attacks a very important technological challenge impacting the science of developmental robotics: How to design a robot learning experiment that can run continuously and autonomously for several months?

The global scenario for the robots in the installation/experiment is the following. In a big egg that has just opened, a tribe of young robotic creatures evolves and explores its environment, wreathed by a large zero that symbolizes the origin. Beyond their innate capabilities, they are outfitted with mechanisms that allow them to learn new skills and invent their own language. Endowed with artificial curiosity, they explore objects

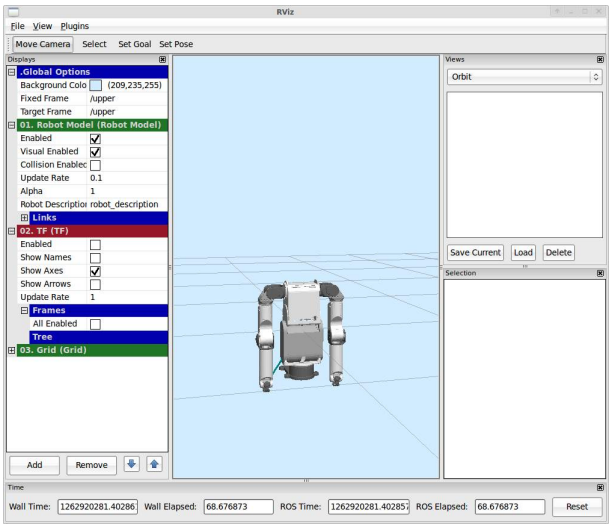

Figure 4. M3 simulation through ROS and Rviz

around them, as well as the effect their vocalizations produce on humans. Human, also curious to see what these creatures can do, react with their own gestures, creating a loop of interaction which progressively self-organizes into a new communication system established between man and ergo-robots.

We now outline the main elements of the software architectures underlying this experimental setup.

5.4.2.1. System components

The software architecture is organized to control the experiment at several levels, and in particular:

- **Scenes**: The organization of behavioural scenes, managing the behaviours that are allowed to each robot at particular times and in particular contexts;
- **Behaviours**: The individual behaviours of robots, also called stems, which are outlined in the next section;
- **stems**: The low-level actions and perceptin of robots while executing their behaviours, including motors control on the five physical stems, color and intensity of lights inside the stem head, production of sounds through speakers. Sensors are the kinect used to interact with visitors, and motor feedback capabilities.

In addition to that a video projector is used to display some artistic view of stem agents internal state.

5.4.2.2. Behaviours

A number of innate behaviours were designed and are used by the robots as elementary behaviours of more complex behaviours, including the three following learning behaviours.

The Naming Game is a behaviour played by stems two-by-two and based on computational models of how communities of language users can self-organize shared lexicons. In the naming game, stems interact with each other in a stylised interaction. Repeated interactions lead to the development of a common repertoire of words for naming objects. More precisely, object belong to meaning spaces. Two such spaces have been implemented for the exhibition. The first one is related to object spatial categorization and the second one is related to movement categorization. The object space contains stems, some hole in walls and the interaction zone. The movement space contains representations of small dances that stem can produce and reproduce.

Figure 5. Three importants concepts in ErgoRobots

Object Curiosity is a behaviour in controlling intrinsically motivated exploration of the physical environnement by the stems. A small wood object is present in the reachable physical environement of the stem, attached on the top of a spring so that it is guaranteed that it comes back to its original position. The stem uses a motor primitive to act on the object and motor feedback to detect movements of the object. The robot learns through active exploration what kind of parameters motor primitive will result in touching the object.

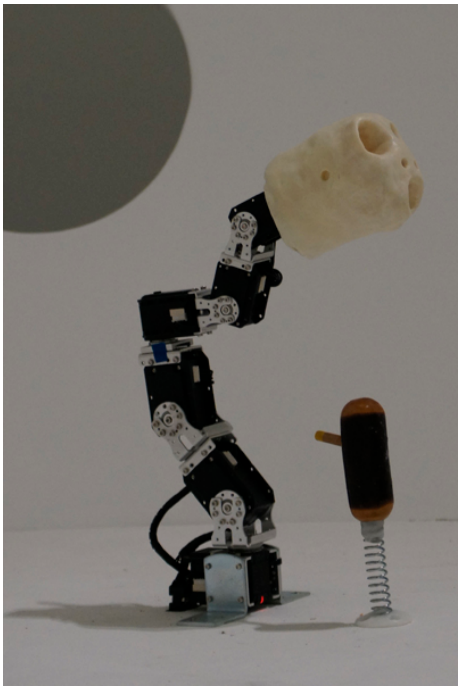

Figure 6. A Stem with the head designed by David Lynch and an Object

Birds Curiosity is a behaviour that drives robots to explore, through curiosity-driven learning, interaction with humans. One stem, generally the stem in the center, plays a sound, predicts the visitor reaction, look the interaction zone and wait the gesture of the visitor. To produce a sound the visitor have to make a gesture in space. In the next iterations, the robot chooses to produce sounds to human which produce most surprising

responses from the human (i.e. the robot is “interested” to explore sound interactions which are not easily predictable by itself).. As describe in the picture, the space is split in four. Each zone corresponding with a sound.

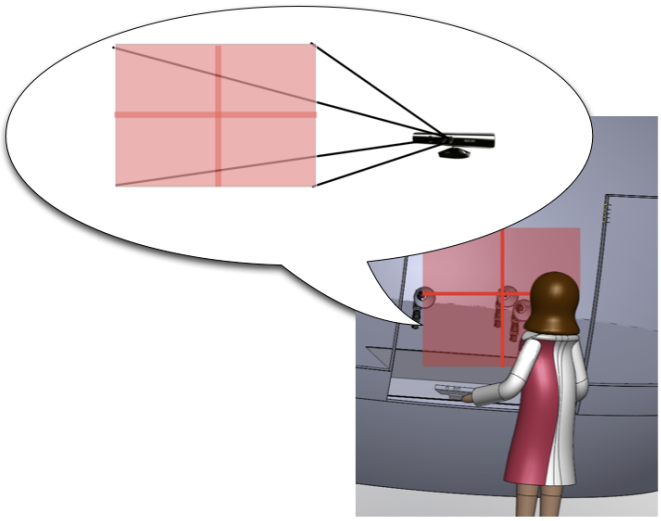

Figure 7. A virtual visitor interact with a virtual grid

5.4.2.3. Programming tools

The system is based on URBI and used some UObjects from UFlow. The most important part of the system is written in URBI script. Python and freenect [4] are used too.

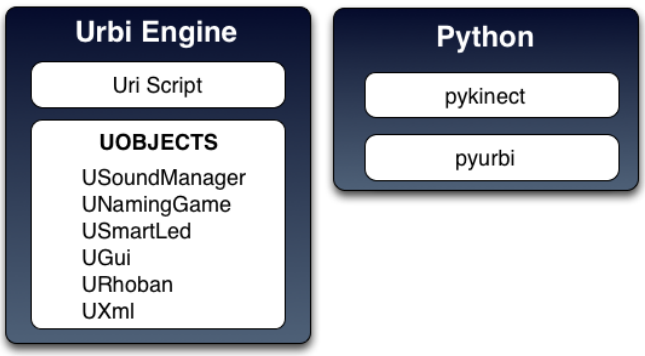

Figure 8. List of software used in ErgoRobots

[4] Kinect library

The system at the startup detects motors and lights. It create dynamically a list of Stem. A Stem is one robot with 6 motors as described in hardware part.

To interact with people, we used the freenect library to interface with the kinect, with a binding to python where detection and following of gestures is made.

For the display, we display an abstract rendering of the structure inside each ErgoRobot, using a python parser to read and parse log file from the ErgoRobot system, and the Bloom/Processing software to create and display the rendering. Currently, the system has three displays, one for the naming game, another one for birds curiosity and the last one for objects curiosity.

The sound system used the UObject USoundManager. It plays sounds when required by a behaviour, it also plays word sounds in Naming Game behaviour.

The Light system used Linkm technologies. In the head of each ErgoRobot we put two lights devices. Each light device is a RGB Light. We can control the intensity of each primary color through I2C control. To control lights we used LinkM USB Device. And finally we used an UObject dedicated to communicate with the USB Device.

5.4.2.4. Maintenance

A dedicate maintenance software is used to switch off, switch on the system. This software is written in Python (and Qt). The status of ErgoRobots is display on the graphical interface. Buttons are present too : Start, Stop, Reset and Take a video.

Recently we added a video system to have a visual feedback of motors usage and also to detect eventual problems. This is a screenshot of the application :

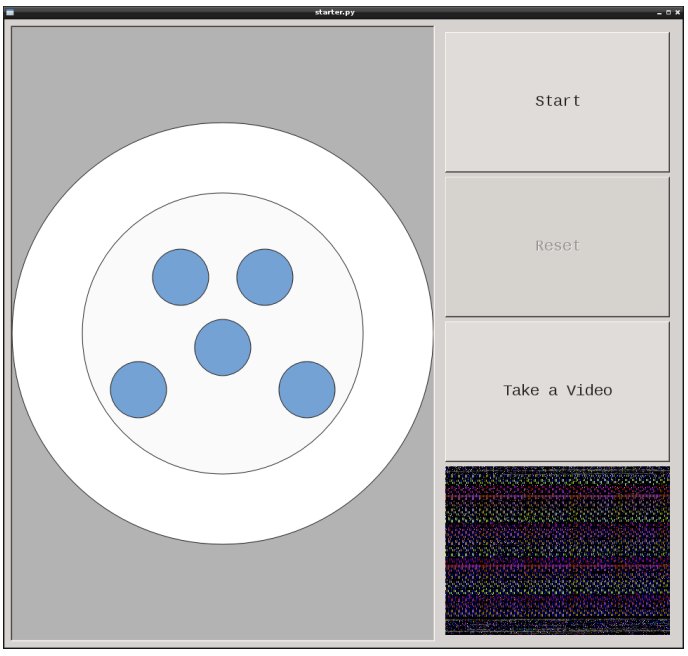

Figure 9. Maintenance Software for the ErgoRobots.

5.4.3. MonitorBoard - Complete solution for monitoring Rhoban Project robots

Participants: Paul Fudal [correspondant], Olivier Ly, Hugo Gimbert.

In collaboration with Rhoban Project/LaBRI/CNRS/Univ. Bordeaux I, the Flowers team took part in a project to exhibit robots at the International Exhibition in Yeosu - 2012 - South Korea (8 millions of visitors expected, from more than 100 countries). The installation consisted in three humanoids (one dancing, two playing on a spring) and five musicians (arms only) playing musical instruments (electric guitar, electric bass guitar, keytar, drums, DJ turntables). In order to increase the robustness of the robotic platform, a complete solution of software and hardware was build. The software solution aims to allow all robots to run safely during the whole exhibition (12 hours per days) and to provide an easy way to diagnose and identify potential electronic and mechanical failures. This software is able to monitor all robots at the same time, verify the health of each motors and each embedded systems. It is able to shutdown or reboot a robot if necessary using PowerSwitches (electric plugs controlled over network) and notify maintenance personal by email explaining the failure. All information is also logged for statistical use. This solution allows to monitor the whole platform without being present, and provides warning signs enabling preventive actions to be taken before an actual failures. It was entirely written in $C\#$ using Microsoft Visual Studio 2010 with .NET API and combined with the existing Rhoban Project API, extended and modified for this purpose. It also involved electric plugs controlled over a network connection.

5.4.4. Motor tracking system

Participants: Jérôme Béchu, Olivier Mangin [correspondant].

We developed a website interface to a database of motors used to build robots in the team. This system is designed for internal use in the team and was developed using the django web framework (https://www.djangoproject.com/) .

5.5. Visualization Tools

5.5.1. Zephyr - Realtime Visualization in JAVA

Participant: Thomas Degris [correspondant].

Zephyr is a software to visualize numeric variables and data structure in real time and at different time scale. Zephyr is practical because it requires only minimal changes in the code: it uses Java reflexivity to automatically detect variables in the code to monitor and data structure with an associated dedicated view. Zephyr can easily be extended with new plugins because it is based on the popular Eclipse Rich Client Platform. Consequently, Zephyr takes advantage of an already existing and fully operational Eclipse plugins for many of its functionalities. Finally, Zephyr is distributed with a Java python virtual machine named Jython and a lisp implementation named Clojure. An example of a Zephyr screen is shown in Figure 10 .

Zephyr was started in fall 2009 in the RLAI group at the university of Alberta (Canada) when Thomas Degris was a postdoc in this group. Zephyr is still actively used by RLAI. Users include Adam White, Joseph Modayil and Patrick Pilarski from the University of Alberta. Zephyr has been registered on the Eclipse marketplace since October 2011. Documentation about Zephyr is included on its website: http://zephyrplugins.github.com. Zephyr is licensed under the open source Eclipse Public License.

5.6. Hardware

5.6.1. Poppy Platform

Participant: Matthieu Lapeyre [correspondant].

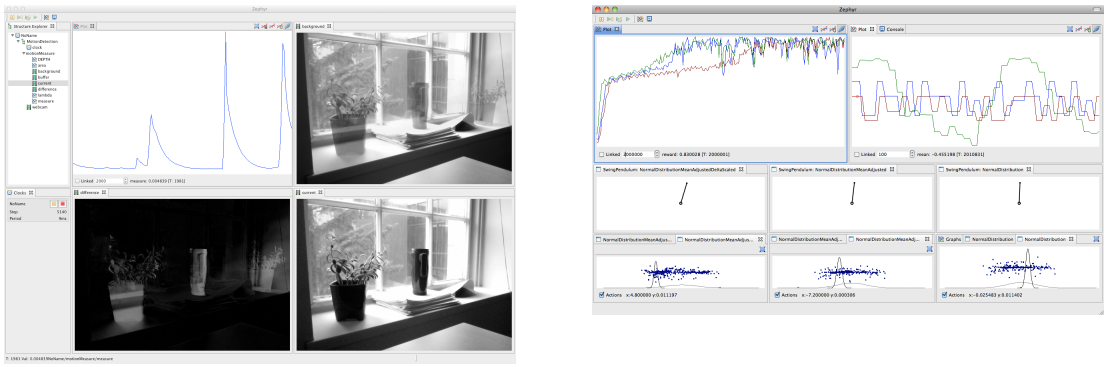

Figure 10. Left: Zephyr showing the different steps of a video processing pipeline in real-time. Right: Zephyr showing different data structure and variables of a reinforcement learning agent at different time scale. A video is available at: http://zephyrplugins.github.com.

5.6.1.1. Main goals :

No current platform (Nao [86], Darwin Op [87], Nimbro Op [117], HRP-2, ...) does offer both a adapted morphology in the sense of allowing physical interaction (safe, compliant, playful) and optimized for walking. So to explore these challenges we have decided to build a new bio-inspired humanoid robotic platform, called Poppy, which provides some of the software and hardware features needed to explore both social interaction and biped locomotion for personal robot. It presents the following main features to make it an interesting platform to study how the combination of morphology and social interaction can help the learning:

- Design inspired from the study of the anatomy of the human body and its bio-mechanic
- Dynamic and reactive: we try to keep the weight of the robot as low as possible (geometry of the pieces and smaller motors)
- Social interaction: screen for communication and permits physical interaction thanks to compliance
- Study of the morphology of the leg to improve the biped walking
- Practical platform: low cost, ease of use and easy to reproduce

5.6.1.2. Overview :

Poppy platform (Figure 11) is a humanoid, it is 84cm tall for 3 kg. It has a large sensor motors space including 25 dynamical motors (MX-28 and AX-12), force sensors under its feet and some extra sensors in the head: 2 HD-wide angle-cameras, stereo-micros and an inertial central unit (IMU 9DoF) plus a large LCD Screen (4 inch) for visual communication (e.g. emotions, instructions or debug). The mechanical parts were designed and optimized to be as light as possible while maintaining the necessary strength. For this, the choice of a lattice beam structure manufactured with 3Dprinting polyamide was used.

The poppy morphology is designed based on the actual human body. We have deeply studied the biomechanics of the human body and have extracted some interesting features for humanoid robotics. This inspiration is expressed in the whole structure (e.g. the limb proportions) and in particular in the trunk and legs.

Poppy uses the bio-inspired trunk system introduced by Acroban [98]. These five motors allow it to reproduce the main changes brought by the human spine. This feature allows the integration of more natural and fluid motion while improving the user experience during physical interactions. In addition, the spine plays a fundamental role in bipedal walking and postural balance by actively participating in the balancing of the robot.

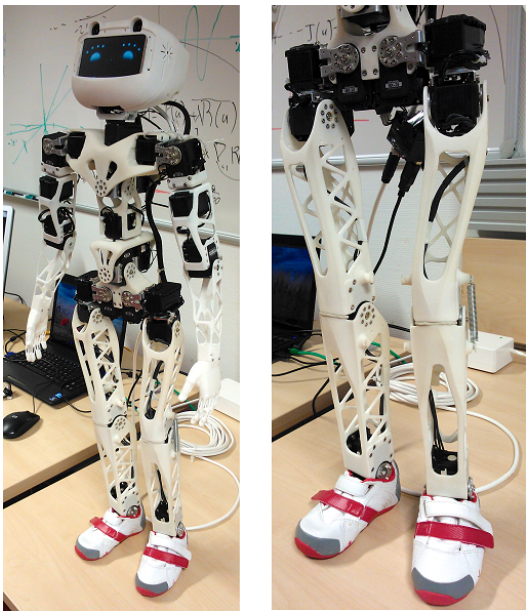

Figure 11. a. Global view of the Poppy platform. b. Zoom on legs design

The legs were designed to increase the stability and agility of the robot during the biped walking by combining bio-inspired, semi-passive, lightweight and mechanical-computation features. We will now describe two examples of this approach:

The architecture of the hips and thighs of Poppy uses biomechanical principles existing in humans. The human femur is actually slightly bent at an angle of about 6 degrees. In addition, the implantation of the femoral head in the hip is on the side. This results in a reduction of the lateral hip movement needed to move the center of gravity from one foot to another and a decrease in the lateral falling speed. In the case of Poppy, the inclination of its thighs by an angle of 6 degrees causes a gain of performance of more than 30% for the two above mentioned points.

Another example is Poppy's feet. Poppy has the particularity of having small feet compared to standard humanoids. It has humanly proportioned feet (ie about 15% of its total size).It is also equipped with compliant toes joints (see Figure 12 .a). We believe that this feet involve two keys features to obtain a human-like and efficient walking gait. However, that raises problems regarding balance because the support polygon is reduced. We decided to add pressure sensors under each foot in order to get accurate feedback of the current state of the robot (see Figure 12 .b).

5.6.1.3. Future works :

In our work, we explore the combination of both a bio-inspired body and bio-inspired learning algorithms. We are currently working on experiments involving Poppy to perform skill learning. First we would like to succeed in achieving an effective postural balance using the articulated spine, the feet pressure sensors and the IMU. Then, we would like to perform experiments on the learning of biped walking using algorithms such as the ones described in [95] or [83]. We are expecting to clearly reduce the learning time needed and increase the quality of the learned tasks thanks to the bio-inspired morphology of Poppy.

We are also interested in social interactions with non-expert users. We would like to conduct user study to evaluate how playful physical interactions and emotions could improve learning in robotics. We think that the poppy platform could be very suitable for such studies.

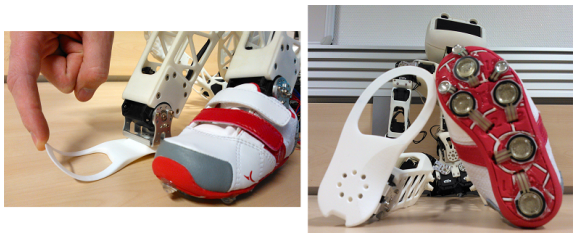

Figure 12. Poppy feet use actual children shoes combine with a compliant feet, toes (a.) and pressure sensors (b.)

5.6.2. Ergo-Robots/FLOWERS Fields: Towards Large-Scale Robot Learning Experiments in the Real World

Participants: Jerome Bechu, Fabien Benureau, Haylee Fogg, Paul Fudal, Hugo Gimbert, Matthieu Lapeyre, Olivier Ly, Olivier Mangin, Pierre Rouanet, Pierre-Yves Oudeyer.

In the context of its participation to the exhibition "Mathematics: A Beautiful Elsewhere" at Fondation Cartier pour l'Art Contemporain in Paris, starting from 19th October 2011 and to be held until 18th March 2012, the team, in collaboration with Labri/Univ. Bordeaux I, has elaborated and experimented a robotic experimental set-up called "Ergo-Robots/FLOWERS Fields" 13 . This set-up is not only a way to share our scientific investigations with the general public, but attacks a very important technological challenge impacting the science of developmental robotics: How to design a robot learning experiment that can run continuously and autonomously for several months? Indeed, developmental robotics takes life-long learning and development as one of its central objective and object of study, and thus shall require experimental setups that allow robots to run, learn and develop for extended periods of time. Yet, in practice, this has not been possible so far due to the unavailability of platforms adapted at the same time to learning, exploration, easy and versatile reconfiguration, and extended time of experimentation. Most experiments so far in the field have a duration ranging from a few minutes to a few hours. This is an important obstacle for the progress of developmental robotics, which would need experimental set-ups capable of running for several months. This is exactly the challenge explored by the Ergo-Robots installation, which we have approached by using new generations of affordable yet sophisticated and powerful off-the-shelf servomotors (RX Series from Robotis) combined with an adequately designed software and hardware architecture, as well as processes for streamlined maintenance. The experiment is now running for five months, six days a week, in a public exhibition which has strong constraints over periods of functioning and no continual presence of dedicated technicians/engineers on site. The experiment involved five robots, each with 6 degrees of freedoms, which are endowed with curiosity-driven learning mechanisms allowing them to explore and learn how to manipulate physical objects around them as well as to discover and explore vocal interactions with humans/the visitors. The robots are also playing language games allowing them to invent their own linguistic conventions. A battery of measures has been set up in order to study the evolution of the platform, with the aim of using the results (to be described in an article) as a reference for building future robot learning experiments on extended periods of time, both within the team and in the developmental robotics community. The system has been running during 5 months, 8 hours a day, with no major problems. During the two first months, the platform worked during 390h21mn, and was only stopped during 24h59mn (6 percent of time). After retuning the system based on what we learnt in the two first months, this performance was increased in the three last months: the platform worked for 618h23mn and was only stopped during 17h56mn (2.9 percent of time).
More information available at: http://flowers.inria.fr/ergo-robots.php and http://fondation.cartier.com/.

5.6.2.1. The Ergo-Robots Hardware Platform

Participants: Jerome Bechu [correspondant], Fabien Benureau, Haylee Fogg, Hugo Gimbert, Matthieu Lapeyre, Olivier Ly, Olivier Mangin, Pierre-Yves Oudeyer, Pierre Rouanet.

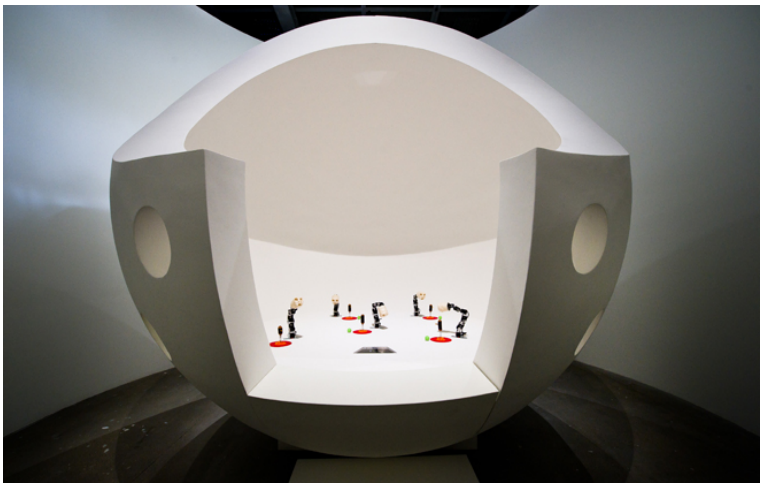

Figure 13. The Ergo-Robot experiment: robot learning experiment running continuously for 5 months at Fondation Cartier pour l'Art Contemporain, exhibition "Mathématiques: Un Dépaysement Soudain".

ErgoRobots 13 is a hardware platform for showcasing a number of curiosity and learning behaviours for the public to interact with. It was designed by the Flowers team in collaboration with Labri/Univ. Bordeaux I. The platform can also have future uses inside the lab for experiments that require more than one robot to complete. Although this system is entirely new this year, a very different previous version existed with the name FLOWERSField. It consists of five ErgoRobots, a control system, an interaction system, a display system, a sound system and a light system. There is an external system which monitors the ErgoRobots which contains a control system, a power system, a surveillance system and a metric capture system. The system has been running during 5 months, 8 hours a day, with no major problems. During the two first months, the platform worked during 390h21mn, and was only stopped during 24h59mn (6 percent of time). After retuning the system based on what we learnt in the two first months, this performance was increased in the three last months: the platform worked for 618h23mn and was only stopped during 17h56mn (2.9 percent of time).

The Ergo-Robot system: The robots themselves are each composed of six motors (see figure). Currently, the heads of the robots have been created in wax by David Lynch and the entire system is displayed at Fondation Cartier inside a large egg shaped orb as shown in the following diagram. The control system module contains both an MMNET1002 control board with an UART-RS485 breakout board which communicates with a ubuntu Linux PC via an ethernet cable. The mment board communicates with the motors, but all other ErgoRobot systems communicate with the PC directly. The sound system is currently externally provided and communicates with the PC. The light system is a series of two or three BlinkM RGB leds placed inside each ErgoRobot head that are controlled through two LinkM USB devices directly with the computer. A kinect placed in front of the system operates as the means for the public to interact with the platform and communicates directly through USB to the PC. The display system is currently an externally provided projector that projects visualisations of the field's current state behind the ErgoRobots.

The external system: This system allows anyone that is monitoring the system to externally control the ErgoRobots system. The PC with which the software control takes place is a Ubuntu Linux system which communicates with the ErgoRobot control system via an ethernet cable. The ErgoRobot harware system can be managed by an external power system which includes a 15.5V bench top power supply for the ErgoRobot motors, an external 12V plug in adapter for the mment board, an external 5V plug in adapter for the LED lights which are all controlled via an emergency stop button. The maintenance system can be located out of direct

view of the ErgoRobot field as it has a surveillance system: a kinect that can display the current state of the field. More surveillance is conducted through a metric capture system that communicates with the ErgoRobots to obtain various state values of the ErgoRobots through the motor sensors and other data. This surveillance is not entirely in place as of 2011 and will be implemented in early 2012.

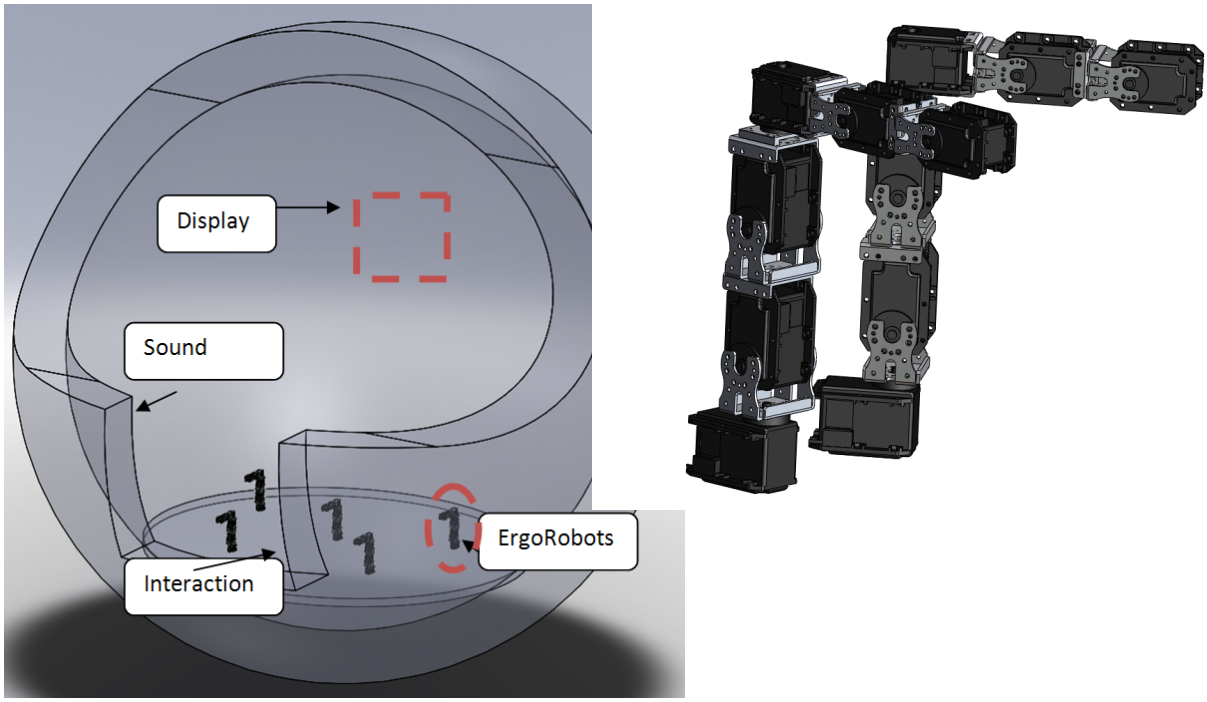

Figure 14. Ergo-Robots

5.6.2.2. Stem Platform for Affordances

Participant: Fabien Benureau [correspondant].

The Stem Platform for Affordances (figure 15 is a hardware platform that is intended for use in the lab for experiments. It features a 6 DOFs arm robot identical to the other robot stems present in the lab, and a physical platform intended for the interaction with objects. Our affordance experiments involves a lot of trials; there was the need for a platform that could reset itself after the robot interacted, as it is an assumption underlying our current algorithms. The stem platform provides exactly that, with the object position and orientation being reseted by the platform autonomously and in less than 10 seconds. This provides the potential to do more than 2000 independent interactions with an object over the course of 12 hours.

The platform also provides sensory capabilities, being able to track the position and orientation of the object at all time. On the hardware side, a camera is used. We investigated both a standard PSEye, that provides a high framerate (120Hz) with noise, and a high quality, firewire camera with professional optics, providing higher resolution, low noise at the expense of a low framerate (15Hz). The latter was kindly provided by the Potioc team. On the software side, computing the position is done by the open-source augmented reality library ARToolkitPlus. On the objects themselves, AR tags are placed.

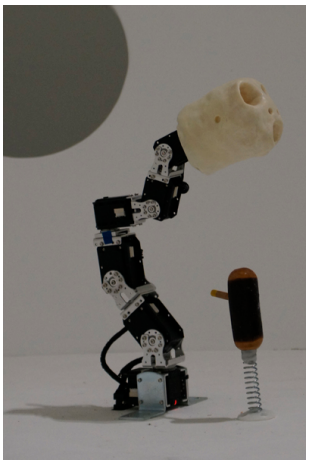

Figure 15. The Stem Platform

The platform is supported by a simulation that reproduce the setup in V-Rep. In order to be able to use the same algorithms for both the hardware and the simulation, a low-level interface was written for Pypot and V-Rep, using the work done by Paul Fudal on V-Rep Bridge.

The complete platform took roughly 3 weeks to make, with 3 additional weeks for the software. The team recently acquired material that would make possible to build a similar platform faster and in a more robust material (wood is used in the first platform). This platform, backed up by its simulation, will allow us to perform planned experiments in a reliable and statistically significant manner.

5.6.2.3. Humanoid Robot Torso

Participant: Haylee Fogg [correspondant].

The Humanoid Robot Torso is a hardware platform that is intended for use in the lab for either experiments or demonstrations 16 . It consists of a humanoid robot that contains just a torso, arms with shoulders and grippers, and head. It is entirely new this year, as a new design has been made, and a skeleton built with 3D printing technologies. The arms with the claws contain seven degrees of freedom (including 'grip'). The head consists of a smartphone for the face and an associated camera for the 'eyes' with the ability to move in two degrees (pitch and roll). The hardware is both robotis Dynamixel RX-28 and R-64 motors attached together with standard robotis frames and 3D printed limbs. A wiki has been built, documenting both the hardware and software platform.

5.6.2.4. NoFish platform

Participants: Mai Nguyen [correspondant], Paul Fudal [correspondant], Jérôme Béchu.

The NoFish platform is a hardware platform that is intended for use in the lab for experiments. It consists of an ErgoRobot with an attached fishing rod. The robot is fixed on a table and has in front of him a delimited area where to throw the fishing cap. This area is covered by a camera in order to track the fishing cap and to give its coordinates. The robot is managed by a software written using the Urbi framework. This program controls the robot using pre-programmed moves and also gives a way to uses the robot joint by joint. A second software written in C++ using OpenCV framework tracks the position of the fishing cap and sends the coordinates to the Urbi software controlling the robot. Finally, at the upper layer of the software architecture, MatLab is used to implement different learning algorithms. All MatLab code is able to receive informations from the Urbi part of the software (fishing cap coordinates, joints informations, etc) and also to send order to the robot (position joint by joint, preprogrammed moves, etc). To finish, and because the platform can run a learning algorithms

Figure 16. The Humanoid Robot Torso Platform

during a long time, an electric plug managed by the Urbi part of the software is added to the platform to shutdown the power if the robot is blocked or does not respond anymore.

MANAO Team

4. Software

4.1. EIGEN

Participants: G. Guennebaud, D. Nuentsa

Keywords : Linear algebra

Efficient numerical computation is central to many computer science domains. In particular, in computer graphics, space transformations and local regressions involve dense linear algebra, data interpolation and differential equations require sparse linear algebra, while more advanced problems involve non-linear optimization or spectral analysis. On the one hand, solutions such as MatLab are limited to prototyping. On the other hand, optimized libraries coming from the HPC (high performance computing) world are often tedious to use and more adapted for very large problems running on clusters. Moreover, all these solutions are very slow at handling very small problems which often arise in computer graphics, vision, or robotics. As a result, researchers of these domains used to waste a lot of time at either implementing their own half cooked solution, or dealing with dozens of complex to use libraries.

The objective of Eigen is to fill this gap by proposing an easy to use, efficient, and versatile C++ mathematical template library for linear algebra and related algorithms. In particular it provides fixed and dynamic size matrices and vectors, matrix decompositions (LU, LLT, LDLT, QR, eigenvalues, etc.), sparse matrices and solvers, some basic geometry features (transformations, quaternions, axis-angles, Euler angles, hyperplanes, lines, etc.), some non-linear solvers, automatic differentiations, etc. Thanks to expression templates, Eigen provides a very powerful and easy to use API. Explicit vectorization is performed for the SSE, AltiVec and ARM NEON instruction sets, with graceful fallback to non-vectorized code. Expression templates allow to perform global expression optimizations, and to remove unnecessary temporary objects.

Eigen is already a well established library with about 20000 unique visitors of the website per month. Eigen is co-developed and maintained with a couple of other researchers and occasional contributors spread over the world. Its development started in 2008, and the last release is the 3.1 version in June 2012. Eigen is currently supported by Inria through an ADT started in January 2012.

Facts:

- Web: http://eigen.tuxfamily.org/
- License: LGPL3+

POTIOC Team

5. Software

5.1. OpenViBE

Participants: Fabien Lotte [local correspondant], Alison Cellard [engineer].

As part of our research work on BCI, we contribute to the development of the OpenViBE [4] software, which is an open source platform dedicated to the design, evaluation and use of BCI for real and virtual applications. OpenViBE development is leaded by Inria, and Potioc will be one of the Inria team contributing to its evolution. Moreover, Potioc is implied in an Inria ADT (Technological Development Action) project that has just started, an that is dedicated to the development of OpenViBE together with 3 other Inria teams (Hybrid, Athena, Cortex).

5.2. Drile

Participant: Florent Berthaut.

As part of the research on Virtual Reality for Musical Performance, notably the Drile system, various software are being developped and made available to the community. These software pieces are the following:

- Pure-Data external to access data from the Virtual Reality Peripheral Network : https://github.com/scrime/vrpd
- Drile: http://hitmuri.net/index.php/Research/Drile

[4] http://openvibe.inria.fr