

Humanoid Path Planner

Florent Lamiraux

CNRS-LAAS, Toulouse, France

Humanoid Path Planner

Introduction

Description of the software

Outline

Introduction

Description of the software

Path Planning

Given

- ▶ A robot (kinematic chain),
- ▶ obstacles,
- ▶ constraints,
- ▶ an initial configuration and
- ▶ goal configurations,

Compute a collision-free path satisfying the constraints from the initial configuration to a goal configuration.

Path Planning

Given

- ▶ A robot (kinematic chain),
- ▶ obstacles,
- ▶ constraints,
- ▶ an initial configuration and
- ▶ goal configurations,

Compute a collision-free path satisfying the constraints from the initial configuration to a goal configuration.

Path Planning

Given

- ▶ A robot (kinematic chain),
- ▶ obstacles,
- ▶ constraints,
- ▶ an initial configuration and
- ▶ goal configurations,

Compute a collision-free path satisfying the constraints from the initial configuration to a goal configuration.

Path Planning

Given

- ▶ A robot (kinematic chain),
- ▶ obstacles,
- ▶ constraints,
- ▶ an initial configuration and
- ▶ goal configurations,

Compute a collision-free path satisfying the constraints from the initial configuration to a goal configuration.

Path Planning

Given

- ▶ A robot (kinematic chain),
- ▶ obstacles,
- ▶ constraints,
- ▶ an initial configuration and
- ▶ goal configurations,

Compute a collision-free path satisfying the constraints from the initial configuration to a goal configuration.

Historical perspective

- ▶ **1998: Move3D,**
- ▶ 2001: Creation of Kineo-CAM, transfer of Move3D,
- ▶ 2006: Release of KineoWorks-2, development of HPP based on KineoWorks-2,
- ▶ 2013: kineo-CAM is bought by Siemens,
- ▶ December 2013: development of HPP open-source.

Historical perspective

- ▶ 1998: Move3D,
- ▶ 2001: Creation of Kineo-CAM, transfer of Move3D,
- ▶ 2006: Release of KineoWorks-2, development of HPP based on KineoWorks-2,
- ▶ 2013: kineo-CAM is bought by Siemens,
- ▶ December 2013: development of HPP open-source.

Historical perspective

- ▶ 1998: Move3D,
- ▶ 2001: Creation of Kineo-CAM, transfer of Move3D,
- ▶ 2006: Release of KineoWorks-2, development of HPP based on KineoWorks-2,
- ▶ 2013: kineo-CAM is bought by Siemens,
- ▶ December 2013: development of HPP open-source.

Historical perspective

- ▶ 1998: Move3D,
- ▶ 2001: Creation of Kineo-CAM, transfer of Move3D,
- ▶ 2006: Release of KineoWorks-2, development of HPP based on KineoWorks-2,
- ▶ 2013: kineo-CAM is bought by Siemens,
- ▶ December 2013: development of HPP open-source.

Historical perspective

- ▶ 1998: Move3D,
- ▶ 2001: Creation of Kineo-CAM, transfer of Move3D,
- ▶ 2006: Release of KineoWorks-2, development of HPP based on KineoWorks-2,
- ▶ 2013: kineo-CAM is bought by Siemens,
- ▶ December 2013: development of HPP open-source.

Outline

Introduction

Description of the software

Overview of the architecture

Modular: collection of packages

- ▶ package dependencies tracked by `pkg-config`,
- ▶ installation managed by `cmake` and a `git` submodule:

```
git://github.com/jrl-umi3218/jrl-cmakemodules.git,
```

- ▶ programmed in `C++`,
- ▶ controlled via `python`

Overview of the architecture

Modular: collection of packages

- ▶ package dependencies tracked by `pkg-config`,
- ▶ installation managed by `cmake` and a `git` submodule:

```
git://github.com/jrl-umi3218/jrl-cmakemodules.git,
```

- ▶ programmed in `C++`,
- ▶ controlled via `python`

Overview of the architecture

Modular: collection of packages

- ▶ package dependencies tracked by `pkg-config`,
- ▶ installation managed by `cmake` and a `git` submodule:

```
git://github.com/jrl-umi3218/jrl-cmakemodules.git,
```

- ▶ programmed in `C++`,
- ▶ controlled via `python`

Overview of the architecture

Modular: collection of packages

- ▶ package dependencies tracked by `pkg-config`,
- ▶ installation managed by `cmake` and a `git` submodule:

`git://github.com/jrl-umi3218/jrl-cmakemodules.git,`

- ▶ programmed in `C++`,
- ▶ controlled via `python`

Overview of the architecture

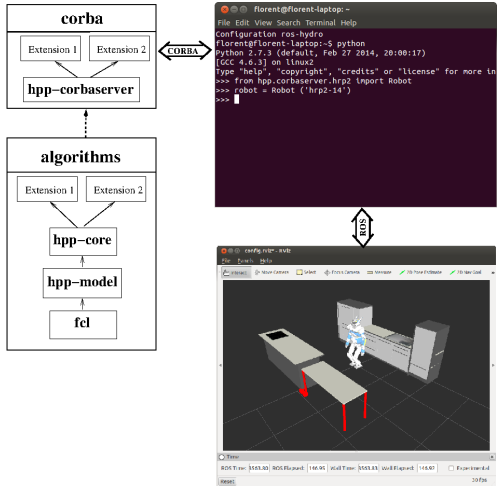
Modular: collection of packages

- ▶ package dependencies tracked by `pkg-config`,
- ▶ installation managed by `cmake` and a `git` submodule:

```
git://github.com/jrl-umi3218/jrl-cmakemodules.git,
```

- ▶ programmed in `C++`,
- ▶ controlled via `python`

Overview of the architecture



Software Development Kit

Packages implementing the core infrastructure

- ▶ Kinematic chain with geometry
 - ▶ `hpp-model`: implementation of kinematic chain with geometry,
 - ▶ tree of joints (Rotation, Translation, SO3: unit-quaternions),
 - ▶ moving `fcl::CollisionObjects`,
 - ▶ forward kinematics,
 - ▶ joint Jacobians,
 - ▶ center of mass and Jacobian.
- ▶ Path planning
 - ▶ `hpp-core`: definition of basic classes,
 - ▶ path planning problems,
 - ▶ path planning solvers (RRT),
 - ▶ constraints (locked dofs, numerical constraints)
 - ▶ path optimizers (random shortcut),
 - ▶ steering methods (straight interpolation)

Software Development Kit

Packages implementing the core infrastructure

- ▶ Kinematic chain with geometry
 - ▶ `hpp-model`: implementation of kinematic chain with geometry,
 - ▶ tree of joints (Rotation, Translation, SO3: unit-quaternions),
 - ▶ moving `fcl::CollisionObjects`,
 - ▶ forward kinematics,
 - ▶ joint Jacobians,
 - ▶ center of mass and Jacobian.
- ▶ Path planning
 - ▶ `hpp-core`: definition of basic classes,
 - ▶ path planning problems,
 - ▶ path planning solvers (RRT),
 - ▶ constraints (locked dofs, numerical constraints)
 - ▶ path optimizers (random shortcut),
 - ▶ steering methods (straight interpolation)

Extensions

Packages implementing other algorithms

- ▶ `hpp-model-urdf`: construction of robots and objects by parsing urdf/srdf files.
- ▶ `hpp-wholebody-step`: whole-body and walk planning using sliding path approximation,
- ▶ `hpp-manipulation`: manipulation planning (under development, not yet available).

Extensions

Packages implementing other algorithms

- ▶ `hpp-model-urdf`: construction of robots and objects by parsing urdf/srdf files.
- ▶ `hpp-wholebody-step`: whole-body and walk planning using sliding path approximation,
- ▶ `hpp-manipulation`: manipulation planning (under development, not yet available).

Extensions

Packages implementing other algorithms

- ▶ `hpp-model-urdf`: construction of robots and objects by parsing urdf/srdf files.
- ▶ `hpp-wholebody-step`: whole-body and walk planning using sliding path approximation,
- ▶ `hpp-manipulation`: manipulation planning (under development, not yet available).

Python control

hpp-corbaserver: python scripting through CORBA

- ▶ **embed** `hpp-core` into a CORBA server and expose services through 3 `idl` interfaces:
 - ▶ `Robot` load and initializes robot,
 - ▶ `Obstacle` load and build obstacles,
 - ▶ `Problem` define and solve problem.
- ▶ Implement python classes to help user call CORBA services
 - ▶ `Robot` automatize robot loading,
 - ▶ `Problem` definition problem helper.

Python control

`hpp-corbaserver`: python scripting through CORBA

- ▶ `embed hpp-core` into a CORBA server and expose services through 3 `idl` interfaces:
 - ▶ `Robot` load and initializes robot,
 - ▶ `Obstacle` load and build obstacles,
 - ▶ `Problem` define and solve problem.
- ▶ Implement python classes to help user call CORBA services
 - ▶ `Robot` automatize robot loading,
 - ▶ `Problem` definition problem helper.

Python control

Extensions

- ▶ `hpp-wholebody-step-corba`: control of humanoid specific constraints and algorithms,
- ▶ `hpp-manipulation-corba`: control of manipulation planning specific classes and algorithms (under construction, not yet available).

Python control

Extensions

- ▶ `hpp-wholebody-step-corba`: control of humanoid specific constraints and algorithms,
- ▶ `hpp-manipulation-corba`: control of manipulation planning specific classes and algorithms (under construction, not yet available).

Visualization through ROS/rviz

Implemented by package `hpp_ros`.

Documentation

Entry point on Gepetto home page:

The screenshot shows the website for the Gepetto Team. The main header features the team name and navigation links. The main content area is titled 'Humanoid Path Planner' and includes sections for Description, Installation, Documentation, and Tutorial. A right-hand sidebar contains a 'News' section with a list of recent events and a 'Site map' section with a list of site pages. The sidebar also includes a search bar and an 'edit SideBar' link.

Gepetto Team

Movement of Anthropomorphic Systems - LAAS - CNRS

home members publications job offers news

Humanoid Path Planner

Description

HPP is a C++ Software Development Kit implementing path planning for kinematic chains in environments cluttered with obstacles. Collision checking is performed by the Flexible Collision Library developed at University of North Carolina.

A bridge with ROS/iviz is provided to visualize obstacles, robot configurations and paths. URDF model can also be parsed to build robots.

python scripting is implemented via CORBA servers and idl interfaces in a seamless way.

It is a collection of software packages handled by cmake and pkg-config.

Installation

To install HPP under ubuntu-12.04 using ros-groovy, follow the instruction provided by the following [README.md](#) file.

Documentation

The [documentation](#) generated by doxygen is automatically installed with each software package.

Tutorial

A tutorial is proposed at the bottom of the documentation page to help you get familiar with the software.

News

- 2014 Apr 23 ISAE visit
- 2014 Apr 04 The behaviors of things
- 2014 Mar 17 PAL Robotics visit
- 2014 Feb 6 Saphari visit
- 2011 Sep Jean-Paul Laumond professor at the Collège de France

See more...

Site map

- HomePage
- Members
- Publications
- Software
 - Humanoid Path Planner
- Demonstrations
- Job Offers
- News

edit SideBar

Search

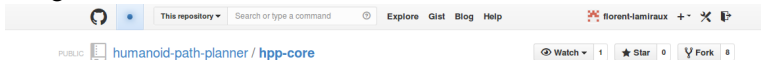
Installation

Go to

`https://github.com/humanoid-path-planner/hpp-doc`
and follow the installation instructions.

Keep informed

- ▶ Mailing list `hpp@laas.fr` to discuss issues related to the software,
- ▶ github notifications for issues related to individual packages



Demonstration