Pacer: Modular, real-time software for legged robot planning and control

Samuel Zapolsky, Evan Drumwright
The George Washington University, Washington, DC, USA
{samzapo,drum}@gwu.edu

1 Introduction

Integrating robotic systems for planning, estimation, and control is a widely interdisciplinary task, drawing on the expertise of hardware engineers, software developers, and control theorists (among others). Researchers often have to collaborate across multiple institutions to create fully functional robots. The ease with which roboticists can communicate by passing techniques and software between research groups is a significant factor in how quickly robot systems can be prototyped, built, and controlled. Cooperation and standardization help properly assess work and permit effective adoption and development upon existing research in the field. We present a software package, *Pacer*, that provides an up-to-date, stable code base for locomotion. *Pacer* allows researchers to contribute directly to the state of the art in locomotion, while enabling them to appraise, modify, and adopt existing software.

Similar compilations of robotics planning and control tools have been made publicly available, including the work of the Gepetto team at LAAS CNRS and *Drake* from the Robot Locomotion Group at MIT[3]. *Pacer* is a framework package developed for real-time perception, planning, and control of simulated and physical robots. *Pacer* provides:

An up to date, stable code base for state of the art planning (motion, path, footstep) and control (error feedback, operational space and joint space, inverse dynamics) software for robot locomotion.

A high level control interface that abstracts planning and control for legged robot locomotion to high level commands (*e.g.*, Dubins car, planar robot).

A plugin robot interface for seamless transition of control from *in situ* to simulated robots (also used by Schaal [2]).

A plugin-based control architecture that permits hot-swapping controllers while a robot is active.

The software we have developed is publicly available at `https://github.com/PositronicsLab/Pacer`.

1.1 High level control interface for legged robots

Pacer provides a simple front end to semi-autonomous control in locomotion by abstracting legged robot footstep planning and control to more simplistic commands (*e.g.*, planar robot degrees of freedom (x, y, θ), the base position differential of the robot in $SE(2)$). Using this abstraction, an operator can drive a robot like a car, or have the robot base follow a planned trajectory between waypoints in the environment. This abstraction can be used as an accessible method for controlling legged robots by researchers not involved directly in locomotion research; as examples, consider problems like image stabilization from walking robots, image recognition on situated legged robots, path and footstep planning algorithms for walking robots, and design of new legged machines. We hope for *Pacer* to catalyze projects such as these.

1.2 Plugin robot interface

Our software maintains and updates an internal model of the robot being controlled. *Pacer* parses the inertial and dynamic properties from standard robot description file types (SDF, URDF) and uses this model to generate kinematic (Jacobian), and dynamic (generalized inertia tensor, momentum) data independently of any particular simulator's kinematics and dynamics representations. This independence from a particular simulator allows for ready communication between groups that conduct research within different simulators. We have demonstrated our system controlling real and simulated robots with no alteration to the underlying controllers (see Figure 1).

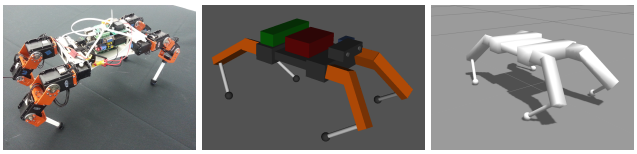

Figure 1: The quadruped robot *R. Links* (left), in *Moby* (center), and *Gazebo* (right).

1.3 Modular planning and control structure

The control architecture we use adopts a plugin framework (also used to manage robot controllers in *Gazebo* [1]). We modularize each controller and planner into its own encapsulated unit. Though some systems may bridge planning and control into a single system (kinodynamic planning), the active systems on a robot often are segmented into distinct, sequential categories: perception, planning, and control. However, one may use further categorizations, including global and local planning, reactive control, stabilization, balance, inverse kinematics, inverse dynamics control, and error feedback control. *Pacer* ensures that sequential processes (*e.g.*, footstep planning then inverse kinematics) are run in order, while allowing non-interdependent processes—such as global path planning—to run in parallel. We further extend plu-

gin scheduling by adding a real-time factor to each control module, permitting a controller to run for n iterations of the real time system before it is queried for a control input to the robot. This layered architecture is illustrated in Figure 2.

References

[1] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2149–2154, Sendai, Japan, Sept 2004.

[2] S. Schaal. The SL simulation and real-time control software package. Technical report, Univ. Southern California, 2009.

[3] R. Tedrake. Drake: A planning, control, and analysis toolbox for nonlinear dynamical systems, 2014.

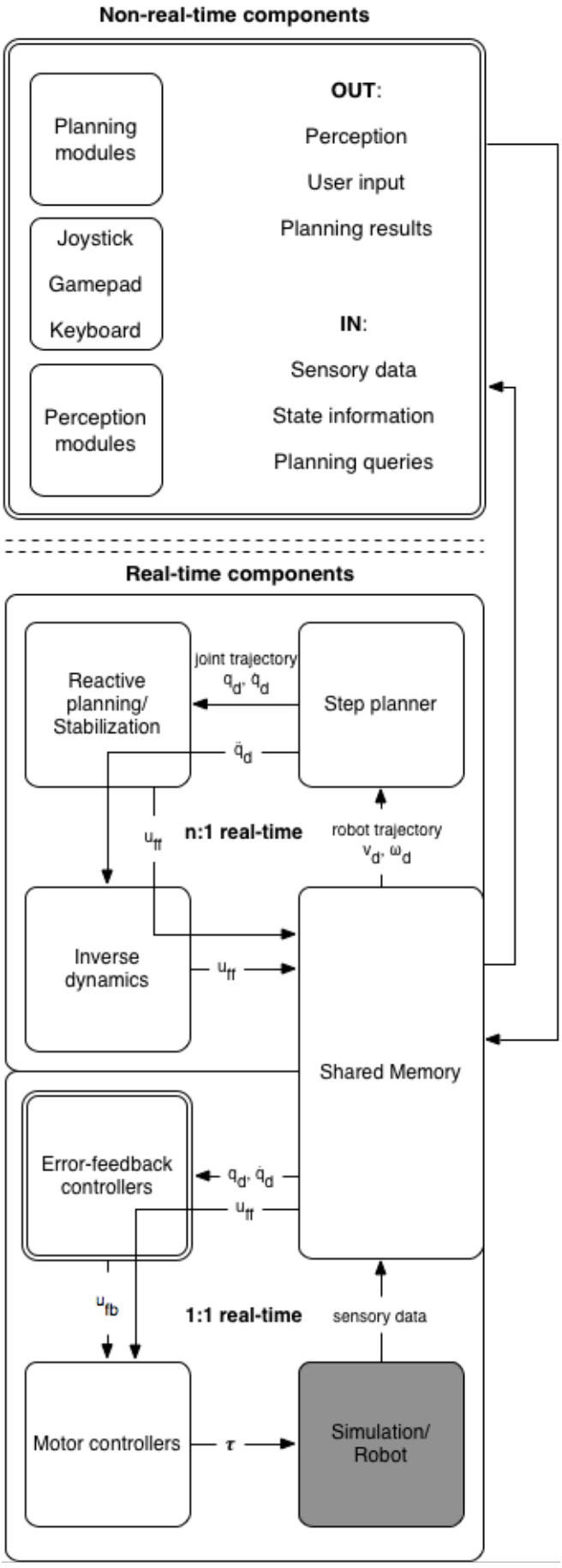

Figure 2: Diagram of layered control architecture in *Pacer*