PILOT: An Actor-oriented Learning and Optimization Toolkit for Swarm Applications

Ilge Akkaya, Shuhei Emoto, Edward A. Lee

Programming The Swarm Workshop
University of California, Berkeley

May 28, 2015
Table of Contents

1 Introduction

2 PILOT Workflow
   - PILOT Actor Library
   - State-space Modeling

3 Case Study
   - Designing Cooperative Robotic Applications using PILOT
Motivation

Distributed system design is prone to errors:

- time and concurrency often not addressed by programming abstractions
- algorithm design is not compositional

In a mobile sensor network setting, design requirements are even more complex

Existing software abstractions based on imperative code fall short on providing

- Scalability
- Structured, repeatable code for deterministic behavior
- Flexible interfaces for variable computational resources
We present PILOT (Ptolemy Inference, Learning, and Optimization Toolkit) that presents an actor library for structured design of robotic sensor network applications

**Goal:** Designing sensor-to-actuator learning and control applications for robotic sensor networks

- Reusable, *actor-oriented component abstractions* that are less error-prone and can be deployed on variable network resources
- Inference and optimization tasks can be defined by the state-space model of the problem domain
Actor-Oriented Design

Ptolemy II
- Ptolemy II is an open source platform for modeling and simulation of systems.
- Actor-oriented, hierarchical, heterogeneous design for composing models of computation
- Accessors: Actors that provide access to sensors, actuators and remote services

The Decorator design pattern
- PILOT utilizes the decorator design pattern based on Ptolemy to define hierarchical association relations between state-space and measurement models
- Accessors: Actors that provide access to sensors, actuators and remote services
Table of Contents

1 Introduction

2 PILOT Workflow
   - PILOT Actor Library
   - State-space Modeling

3 Case Study
   - Designing Cooperative Robotic Applications using PILOT
System Architecture

Figure: State-Space Aware System Architecture in PILOT
PILOT: Ptolemy Inference, Learning and Optimization Toolkit

- Machine Learning
  - Bayesian parameter estimation, decoding and classification (Hidden Markov, Hidden Semi-Markov and Gaussian Mixture Models)
- Particle Filters
- Optimization (Convex Optimization and Gradient Descent)
State-Space Modeling in PILOT

A general Bayesian state-space model is given by

\[
\begin{align*}
    x_0 &\sim \pi_X(x_0) \quad \text{(prior)} \\
    z_t | x_t &\sim g(x_t, u_t, t) \quad \text{(measurement model)} \\
    x_{t+1} | x_t &\sim f(x_t, u_t, t) \quad \text{(state dynamics)}
\end{align*}
\]

Example:

\[
\begin{align*}
    X_t &= \begin{bmatrix} x_t; y_t \end{bmatrix} \\
    x_0, y_0 &\sim \text{Uniform}(-100, 100) \\
    X_{t+1} &= X_t + \eta_t \\
    z_t &= \sqrt{x_t^2 + y_t^2} + \omega_t \\
    \omega_t &\sim \mathcal{N}(0, \sigma^2) \\
    \eta_t &\sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma\right)
\end{align*}
\]
Table of Contents

1 Introduction

2 PILOT Workflow
   - PILOT Actor Library
   - State-space Modeling

3 Case Study
   - Designing Cooperative Robotic Applications using PILOT
Case Study: Cooperative RSN Control

Figure: Top-Level Model for Range-Only Target Localization
Problem Definition

- A network of mobile sensor nodes
- Range-only sensors used to sense the position of mobile target(s)
- A cloud based application (centralized or decentralized) which takes range measurements and computes future robot trajectories to achieve a control goal
  - Localize target as fast as possible
  - Pursue the target
  - A multi-objective control goal
- Subject to environmental constraints
  - Obstacle/Collision avoidance
  - Speed and acceleration constraints
Figure: Model of a Robot Equipped with Range Sensor

- Heterogeneous models: Continuous and Discrete-Event
- Sensing an actuation can be simulated or
- the control logic can be integrated with the physical sensor network via Accessors
PILOT Control Workflow

Introduction

PILOT Workflow

PILOT Library

State-space Modeling

Case Study

Cooperative Apps

UC Berkeley

Case Study

Ilge Akkaya
State-Space Oriented Learning Models

Figure: Sample particle output for state estimation using range-only sensing
CompositeOptimizer Workflow

\[ \mathbf{x}^* = \text{minimize} \ f(\mathbf{x}, q_1, q_2) \]
\[ \mathbf{x} \in \mathbb{R}^n \]

subject to \( g(\mathbf{x}, q_1, q_2) \geq 0 \),

Algorithm

CompositeOptimizer

Input: \( Q \leftarrow Q_i \)

Output: \( \mathbf{x}^* \) that is a local optimum of \( f(\cdot) \)

define \( P \): An actor that implements SDF, has inputs: \( \mathbf{x}, Q \) and outputs: \( f, g \)

while \( k < k_{\text{MAX}} \) &

!CompositeOptimizer.converged() do

\( \mathbf{x}(k) \leftarrow \text{OptimizerDirector.getNextX}() \);
\( P.\text{readInputs}(\mathbf{x} \leftarrow \mathbf{x}(k), Q \leftarrow Q_i) \);
\( P.\text{execute}() \);
\( P.\text{writeOutput}(f(\mathbf{x}(k), Q_i) \Rightarrow f^{(k)}), g(\mathbf{x}(k), Q_i) \Rightarrow g^{(k)}) \);

OptimizerDirector.\text{computeNextX}(f^{(k)}, g^{(k)})
end while

\( \mathbf{x}^* \leftarrow \text{CompositeOptimizer.getOptimalX}() \)
Experiments: Direct Pursuit

\[ u_t^* = \arg \min_{u_t \in U^M} \| R_{t+1} - x_{t+1} \| \]

s.t. \( \| u^{(i)}_t \| \leq V_{max}, \ i = 1, 2, \ldots, M \)
Experiments: MI Maximization

\[ u_t^* = \arg \max_{u_t \in U^M} I(z_{t+1}; x_{t+1}) \]

\[ \text{s.t} \|u_t^{(i)}\| \leq V_{max}, \ i = 1, 2, \ldots, M \]
Experiments: MI Maximization + Pursuit

Figure: Sample trajectory for MI Maximization with Single Pursuer

$$u_t(i)^* = \begin{cases} 
\arg \min_{u_t(i) \in U} \| R_{t+1} - x_{t+1} \| \\
\text{s.t.} \| u_t(i) \| \leq V_{\text{max}}, \ i = 1, 2, \ldots, M 
\end{cases}$$

if $d_t(i) < d_t(j), \ \forall j \neq i$

$$d_t(i) := \| R_{i_t} - x_t \|, \ i \in \{1, 2, \ldots, M\}$$
Dynamic Performance Evaluation

Figure: Comparison of Algorithmic Accuracy
Environmental Fault Modeling: Binary Erasure Channel

Figure: Comparison of Cooperative Localization Error with Probabilistic Channel Loss
Robotic Sensor Network Control
Work in Progress: Distributed Optimization Enabled by Vert.x (or ROS?)

**Figure**: Model-predictive control of flocking robots on a ring network topology
Thanks!

- PILOT is shipped as part of the open-source Ptolemy II Project
- RSN control demos can be accessed online at:
  - http://ptolemy.eecs.berkeley.edu