MISSION:
Deliver packages to locations before they expire (hopefully with a minimal amount of robot motion)

BASIC ASSUMPTIONS:

- Known map of Houses, Roads(Intersections).
- Known Beacon locations
- Package states are fully observable(target,location time)
- Robot can carry any number of packages at once

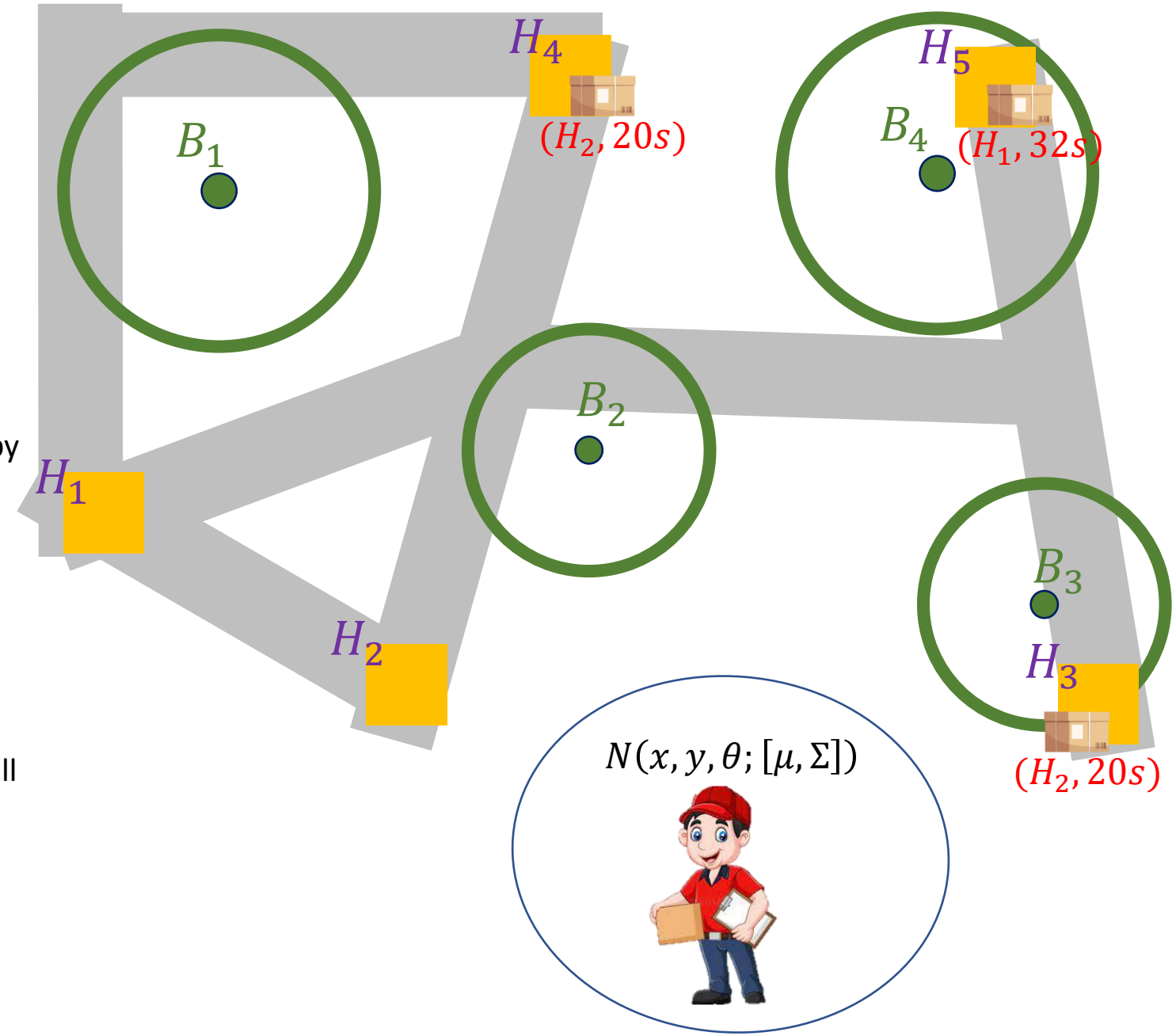
ROBOT STASTE:

- Robot state is partially observable and estimated by a gaussian (μ, Σ)
- Beacons provide stochastic measurements of bearing/range
- Stochastic odometry measurements allow for propagating a motion model

MORE CONSTRAINTS:

- Robot can pickup a package only if $\mathrm{trace}(\Sigma)$ is small enough and is close enough to the package

→We can't ignore the beacons in planning.



Full Planning:

I – *robot state, packages state*

G – *packages at goals not expired*

ACTIONS:

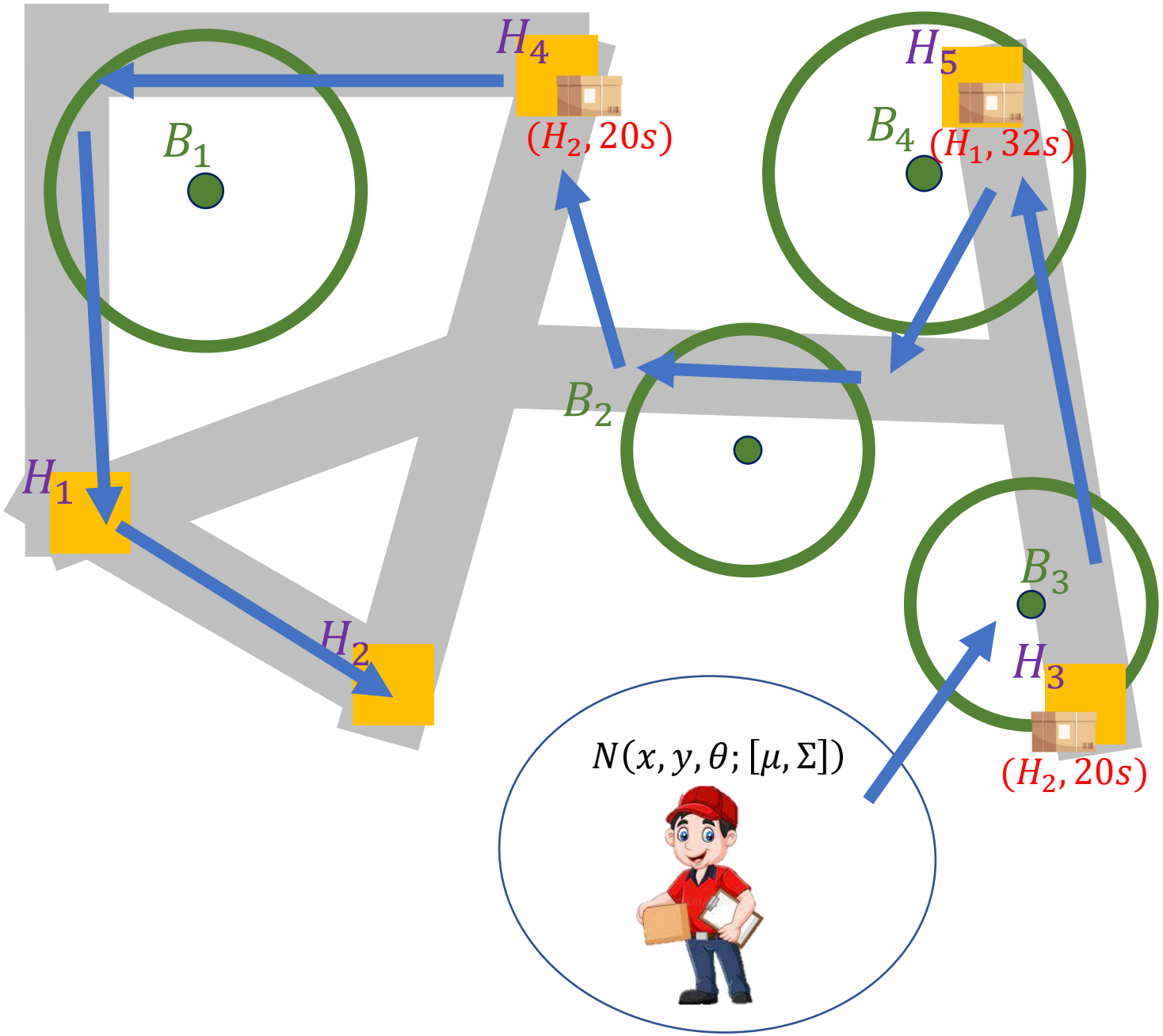
- Pick-up
- Drop-down
- Move to (costs time depending on distance)

NOTE:

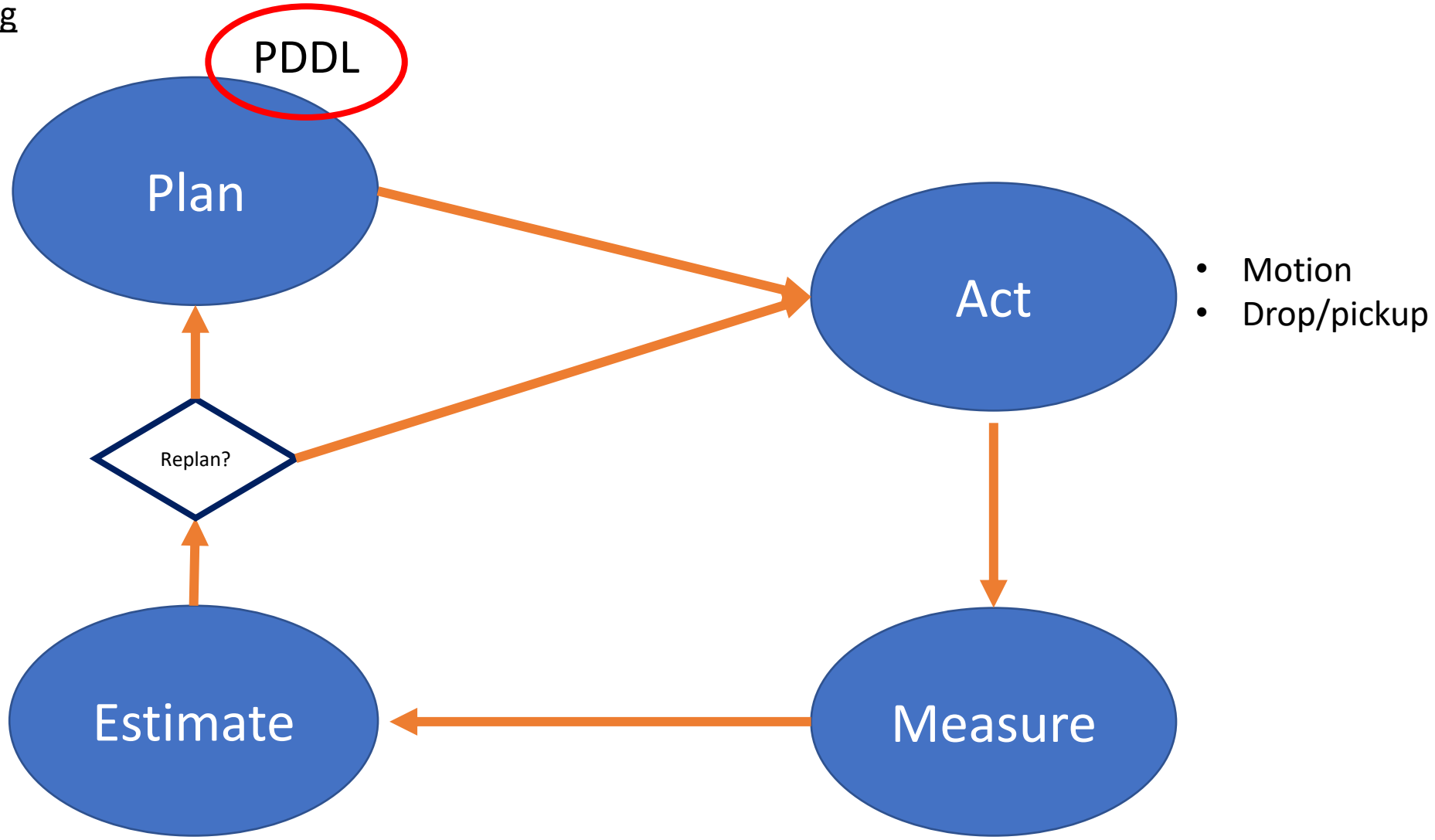
- Problems with no possible solutions can be formulated. We can/should report failure.

Consecutive Planning

- Due to the stochasticity of the motion model, a plan made can be no longer feasible.
  Example: flat tire.
  In that case we will want to fix the tire and replan.



Consecutive Planning



- Motion
- Drop/pickup

- EKF or similar

- Beacons

PROBLEM

Our classic PDDL solver is not capable of POMDP!

Solution:
We make further assumptions when planning

What are our options? What are we dealing with?

PDDL2.1

Additional features:

- numeric expressions;
- metrics;
- durative actions (both discretised and continuous).

The differences between the PDDL2.1 syntax and the PDDL (McDermott, 2000) syntax:

- operations increase, decrease and assign are used instead of $+$, $-$ and change;
- numeric expressions are not allowed to appear as terms in the language (that is, as arguments to predicates or values of action parameters);
- functions are restricted to be of type $Object^n \rightarrow \mathbb{R}$.

Suggestion:

Action: $Move(x_k \rightarrow x_{k+1})$

Effect: Increase/Decrease uncertainty estimator $U \in R^1$

$$trace(\Sigma_{k+1}) = U_{k+1} \approx \left(U_k + g(x_k, x_{k+1})\right) \sum_{x_{beacon}} h(x_k, x_{k+1}, x_{beacon})$$

Exploring proper g, h is part of the project

Tools of the trade

https://github.com/IBM/pddl-in-python

PDDL in Python – Python DSL for writing a PDDL

A minimal implementation of a DSL which allows people to write PDDL in python. Based on parsing python's AST.

Author: Masataro Asai

License: MIT.

Example in examples/blocksworld.py:

```
class Blocksworld(Domain):
    def move_b_to_b(bm, bf, bt):
        if clear[bm] and clear[bt] and on[bm, bf]:
            clear[bt]  = False
            on[bm, bf] = False
            on[bm, bt] = True
            clear[bf]  = True

    def move_b_to_t(bm, bf):
        if clear[bm] and on[bm, bf]:
            on[bm, bf]   = False
            on_table[bm] = True
            clear[bf]    = True

    def move_t_to_b(bm, bt):
        if clear[bm] and clear[bt] and on_table[bm]:
            clear[bt]    = False
            on_table[bm] = False
            on[bm, bt]   = True

print(Blocksworld())
```

```
(domain blocksworld
  (:requirement :strips)
  (:types)
  (:predicates
    (clear ?x0)
    (on ?x0 ?x1)
    (on-table ?x0))
  (:action move-b-to-b :parameters (?bm ?bf ?bt)
   :preconditions
   (and
     (clear ?bm)
     (clear ?bt)
     (on ?bm ?bf))
   :effects
   (and
     (not (clear ?bt))
     (not (on ?bm ?bf))
     (on ?bm ?bt)
     (clear ?bf)))
  (:action move-b-to-t :parameters (?bm ?bf)
   :preconditions
   (and
     (clear ?bm)
     (on ?bm ?bf))
   :effects
   (and
     (not (on ?bm ?bf))
     (on-table ?bm)
     (clear ?bf)))
  (:action move-t-to-b :parameters (?bm ?bt)
   :preconditions
   (and
     (clear ?bm)
     (clear ?bt)
     (on-table ?bm))
   :effects
   (and
     (not (clear ?bt))
     (not (on-table ?bm))
     (on ?bm ?bt))))
```

https://github.com/caelan/SS-Replan

SS-Replan

Online observation, estimatation, planning, and control for a Franka Panda Robot operating in NVIDIA SRL's simulated kitchen environment.

Installation

SS-Replan supports both Python 2 and Python 3.

- Install Git LFS
- `$ pip install numpy scipy pybullet sklearn`
- `$ git lfs clone --branch master --recurse-submodules https://github.com/caelan/SS-Replan.git`
- `$ cd SS-Replan`
- `SS-Replan$ git lfs install`
- `SS-Replan$ ./pddlstream/FastDownward/build.py release64`
- `SS-Replan$ cd ss-pybullet/pybullet_tools/ikfast/franka_panda`
- `SS-Replan/ss-pybullet/pybullet_tools/ikfast/franka_panda$ ./setup.py`

It's also possible to use TRAC-IK instead of IKFast; however it requires installing ROS (`$ sudo apt install ros-kinetic-trac-ik`).

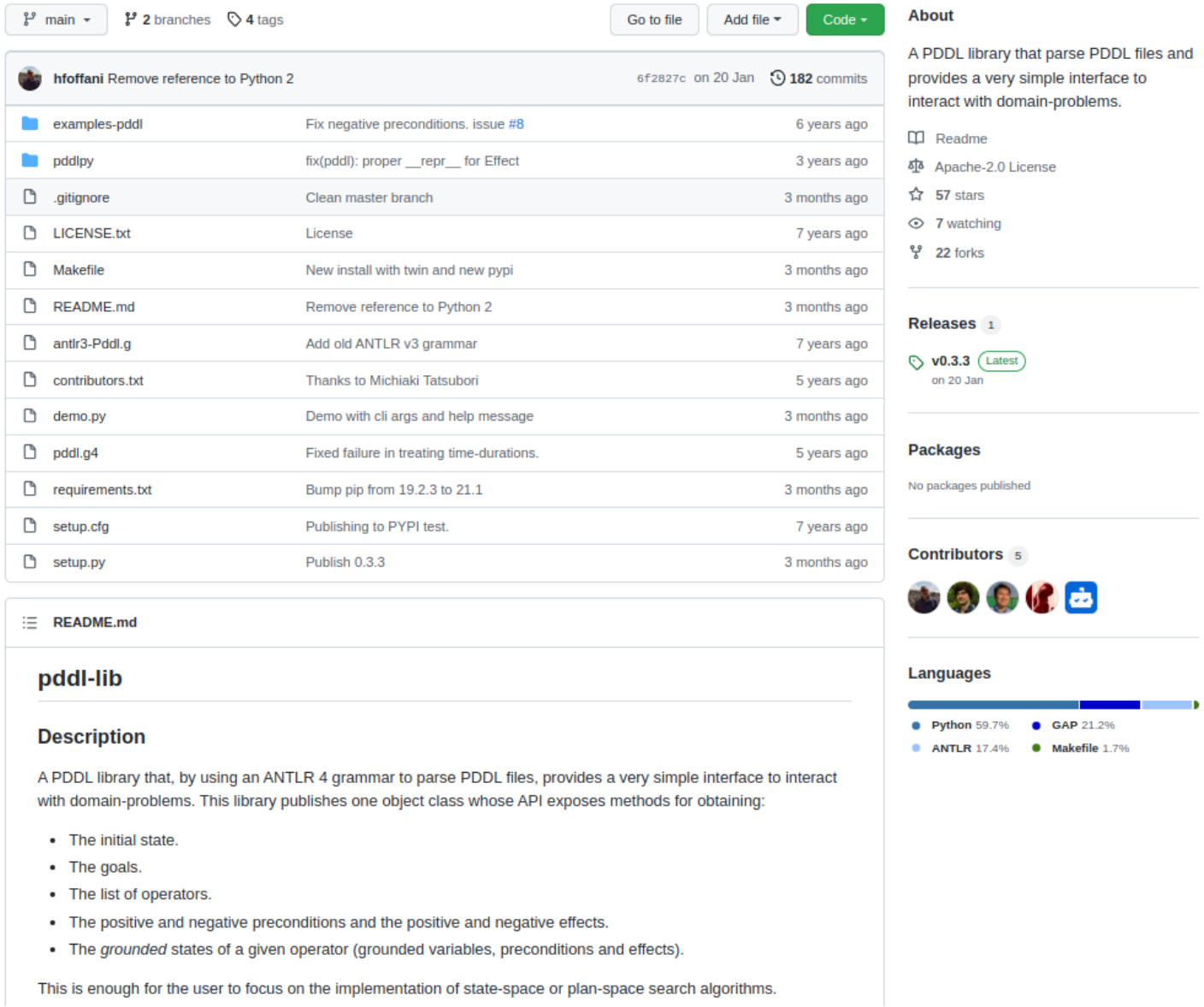
PyBullet Examples

- `SS-Replan$ git pull`
- `SS-Replan$ git submodule update --init --recursive`
- `SS-Replan$ ./run_pybullet.py [-h]`

Languages

Python 99.9% Other 0.1%

https://github.com/hfoffani/pddl-lib



main · 2 branches · 4 tags · Go to file · Add file · Code

hfoffani Remove reference to Python 2 — 6f2827c on 20 Jan — 182 commits

examples-pddl	Fix negative preconditions. issue #8	6 years ago
pddlpy	fix(pddl): proper __repr__ for Effect	3 years ago
.gitignore	Clean master branch	3 months ago
LICENSE.txt	License	7 years ago
Makefile	New install with twin and new pypi	3 months ago
README.md	Remove reference to Python 2	3 months ago
antlr3-Pddl.g	Add old ANTLR v3 grammar	7 years ago
contributors.txt	Thanks to Michiaki Tatsubori	5 years ago
demo.py	Demo with cli args and help message	3 months ago
pddl.g4	Fixed failure in treating time-durations.	5 years ago
requirements.txt	Bump pip from 19.2.3 to 21.1	3 months ago
setup.cfg	Publishing to PYPI test.	7 years ago
setup.py	Publish 0.3.3	3 months ago

README.md

pddl-lib

Description

A PDDL library that, by using an ANTLR 4 grammar to parse PDDL files, provides a very simple interface to interact with domain-problems. This library publishes one object class whose API exposes methods for obtaining:

- The initial state.
- The goals.
- The list of operators.
- The positive and negative preconditions and the positive and negative effects.
- The *grounded* states of a given operator (grounded variables, preconditions and effects).

This is enough for the user to focus on the implementation of state-space or plan-space search algorithms.

About

A PDDL library that parse PDDL files and provides a very simple interface to interact with domain-problems.

Readme
Apache-2.0 License
57 stars
7 watching
22 forks

Releases 1

v0.3.3 Latest
on 20 Jan

Packages

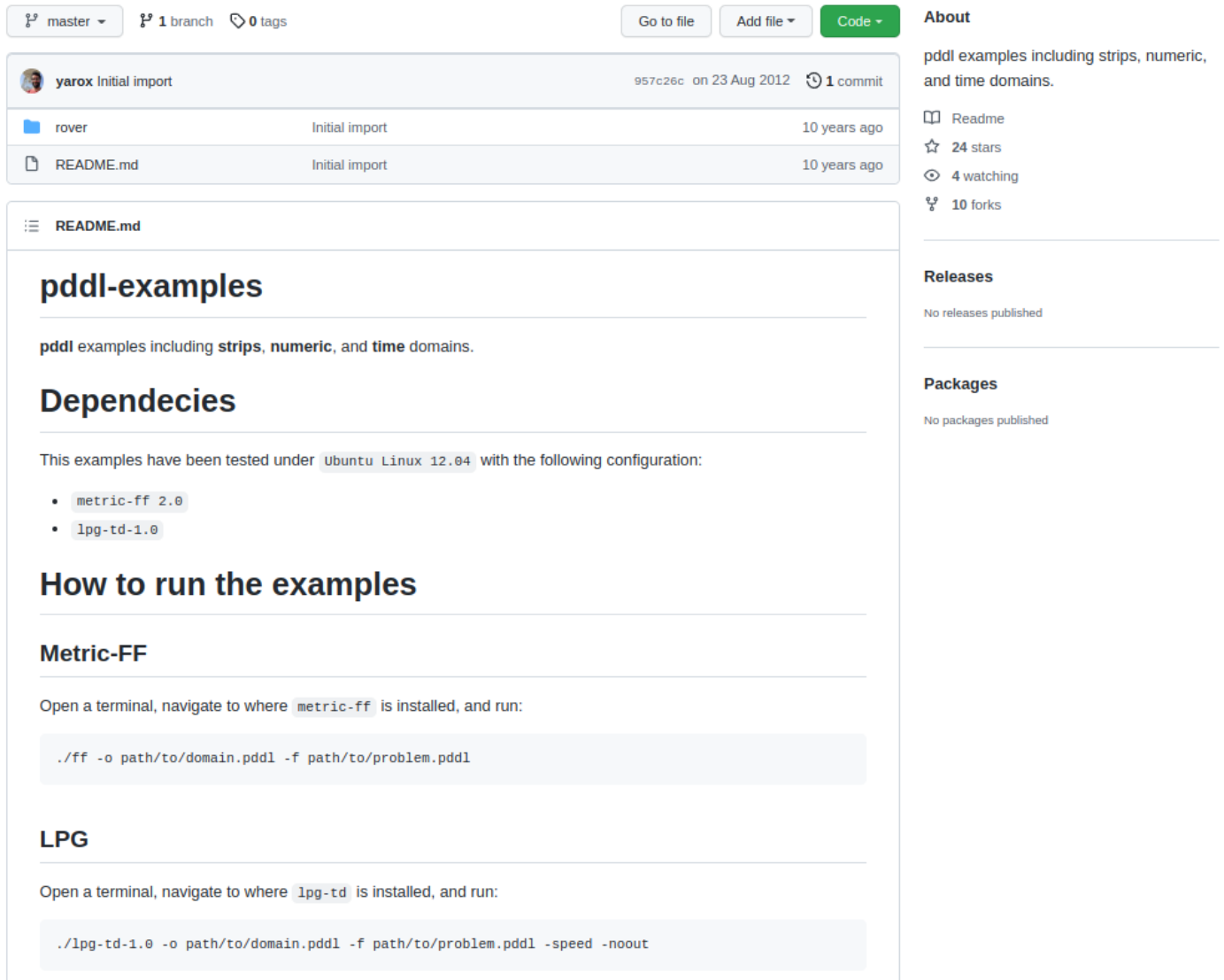
No packages published

Contributors 5

Languages

Python 59.7% · GAP 21.2% · ANTLR 17.4% · Makefile 1.7%

https://github.com/yarox/pddl-examples



master · 1 branch · 0 tags · Go to file · Add file · Code

yarox Initial import — 957c26c on 23 Aug 2012 — 1 commit

rover	Initial import	10 years ago
README.md	Initial import	10 years ago

README.md

pddl-examples

pddl examples including **strips**, **numeric**, and **time** domains.

Dependecies

This examples have been tested under `Ubuntu Linux 12.04` with the following configuration:

- `metric-ff 2.0`
- `lpg-td-1.0`

How to run the examples

Metric-FF

Open a terminal, navigate to where `metric-ff` is installed, and run:

```
./ff -o path/to/domain.pddl -f path/to/problem.pddl
```

LPG

Open a terminal, navigate to where `lpg-td` is installed, and run:

```
./lpg-td-1.0 -o path/to/domain.pddl -f path/to/problem.pddl -speed -noout
```

About

pddl examples including strips, numeric, and time domains.

Readme
24 stars
4 watching
10 forks

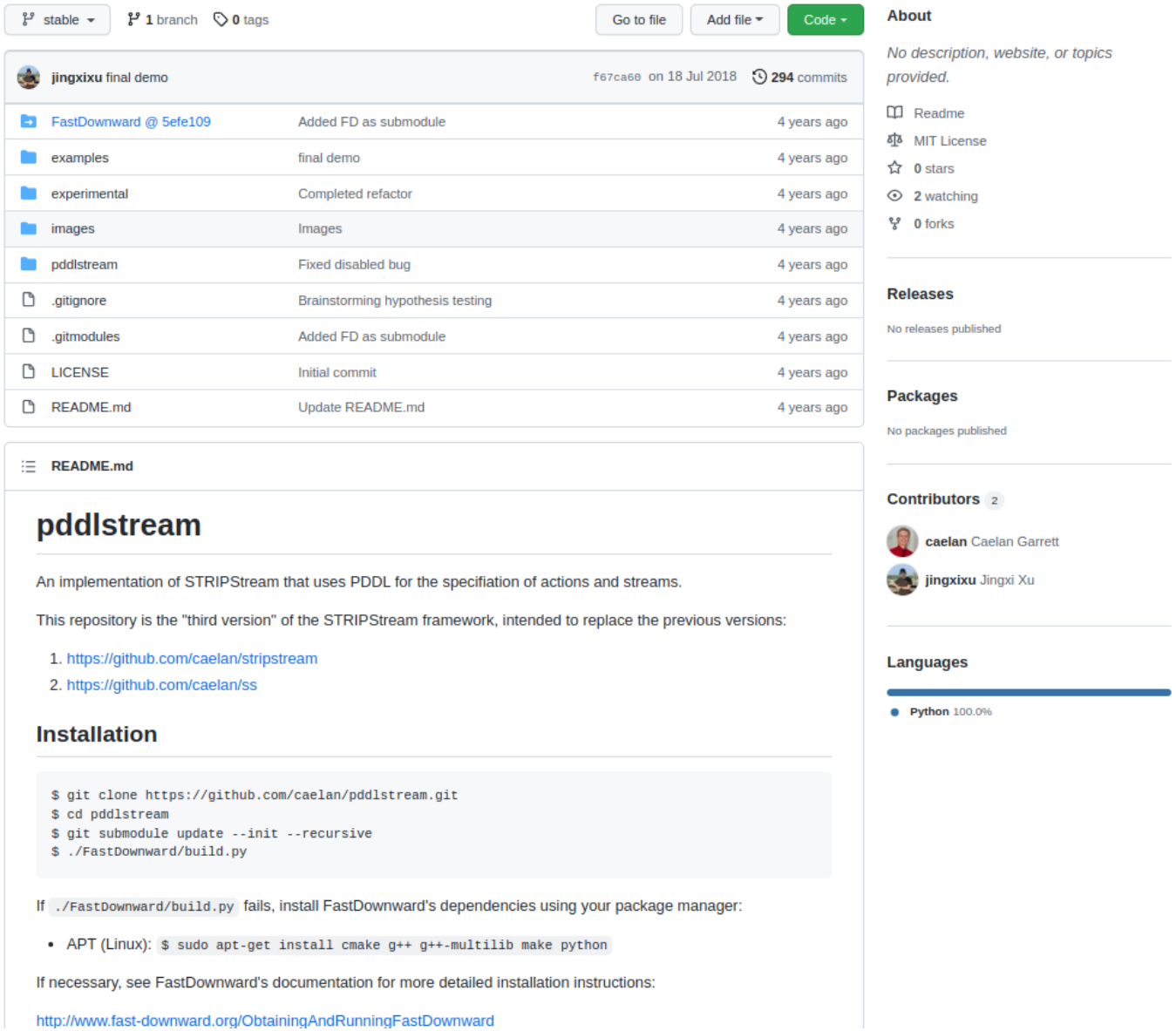
Releases

No releases published

Packages

No packages published

https://github.com/jingxixu/pddlstream



stable · 1 branch · 0 tags · Go to file · Add file · Code

jingxixu final demo — f67ca60 on 18 Jul 2018 — 294 commits

FastDownward @ 5efe109	Added FD as submodule	4 years ago
examples	final demo	4 years ago
experimental	Completed refactor	4 years ago
images	Images	4 years ago
pddlstream	Fixed disabled bug	4 years ago
.gitignore	Brainstorming hypothesis testing	4 years ago
.gitmodules	Added FD as submodule	4 years ago
LICENSE	Initial commit	4 years ago
README.md	Update README.md	4 years ago

README.md

pddlstream

An implementation of STRIPStream that uses PDDL for the specifiation of actions and streams.

This repository is the "third version" of the STRIPStream framework, intended to replace the previous versions:

1. https://github.com/caelan/stripstream
2. https://github.com/caelan/ss

Installation

```
$ git clone https://github.com/caelan/pddlstream.git
$ cd pddlstream
$ git submodule update --init --recursive
$ ./FastDownward/build.py
```

If `./FastDownward/build.py` fails, install FastDownward's dependencies using your package manager:

- APT (Linux): `$ sudo apt-get install cmake g++ g++-multilib make python`

If necessary, see FastDownward's documentation for more detailed installation instructions:

http://www.fast-downward.org/ObtainingAndRunningFastDownward

About

No description, website, or topics provided.

Readme
MIT License
0 stars
2 watching
0 forks

Releases

No releases published

Packages

No packages published

Contributors 2

caelan Caelan Garrett
jingxixu Jingxi Xu

Languages

Python 100.0%