

gz-unitree: Reinforcement learning en robotique
avec validation par moteurs de physique
multiples pour le robot *H1v2* d'Unitree

Gwenn Le Bihan <gwenn.lebihan@etu.inp-n7.fr>

16 Novembre 2025

Reinforcement Learning

Et son application à la robotique

Bases du RL

Agent

Environnement

Score

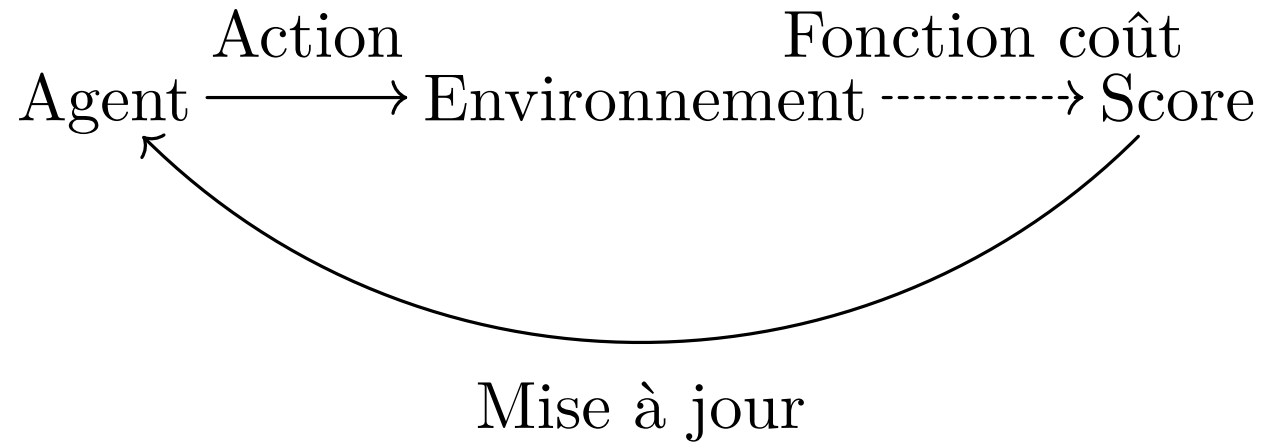
Bases du RL



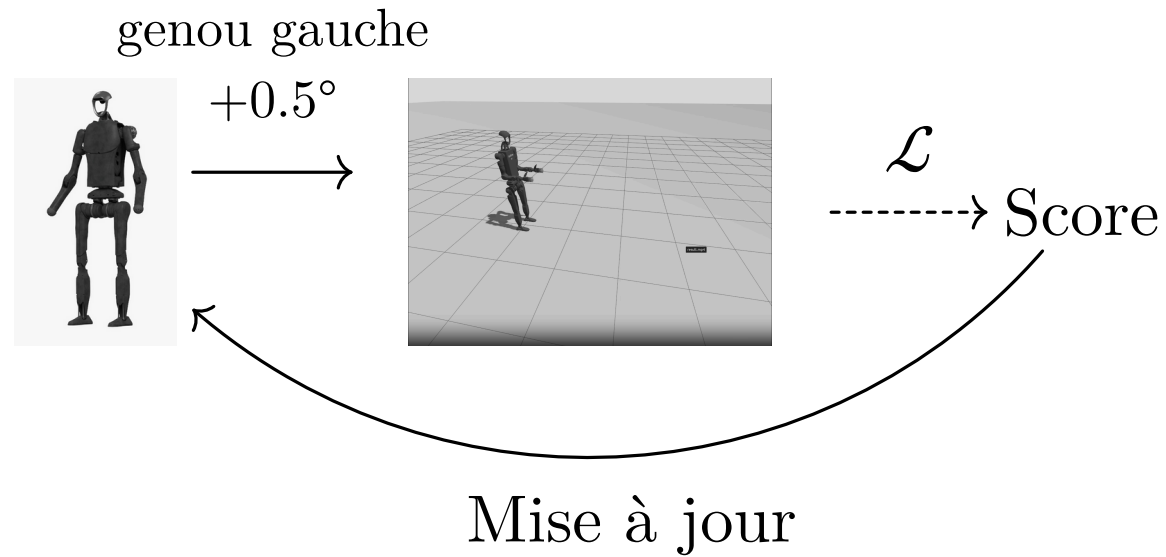
Bases du RL



Bases du RL



RL en robotique



C'est quoi \mathcal{L} ?

C'est très simple:

$$\mathcal{L}_r(\pi', \pi) := \mathbb{E}_{(s_t, a_t)_{t \in \mathbb{N}} \in \mathcal{C}} \sum_{t=0}^{\infty} \frac{Q_{\pi}(s_t, a_t)}{Q_{\pi'}(s_t, a_t)} A_{\pi, r}(s_t, a_t)$$

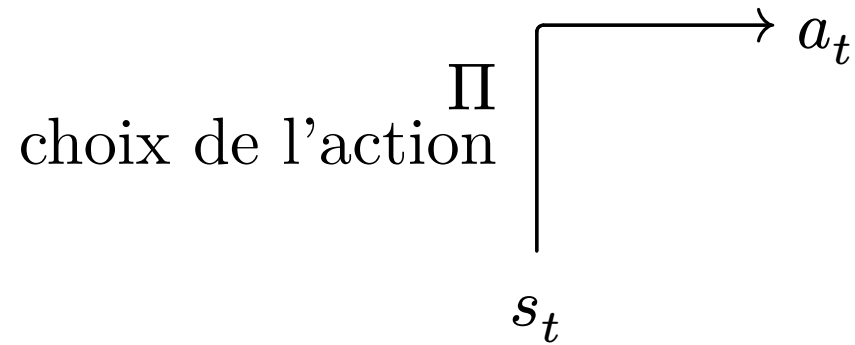
Comparaison des politiques

En Reinforcement Learning

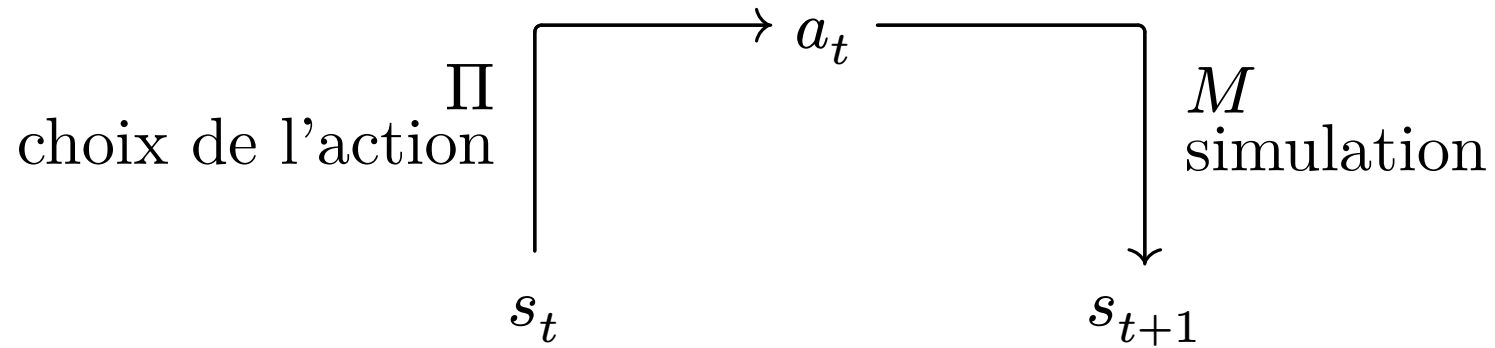
Comparaison des politiques

$$s_t$$

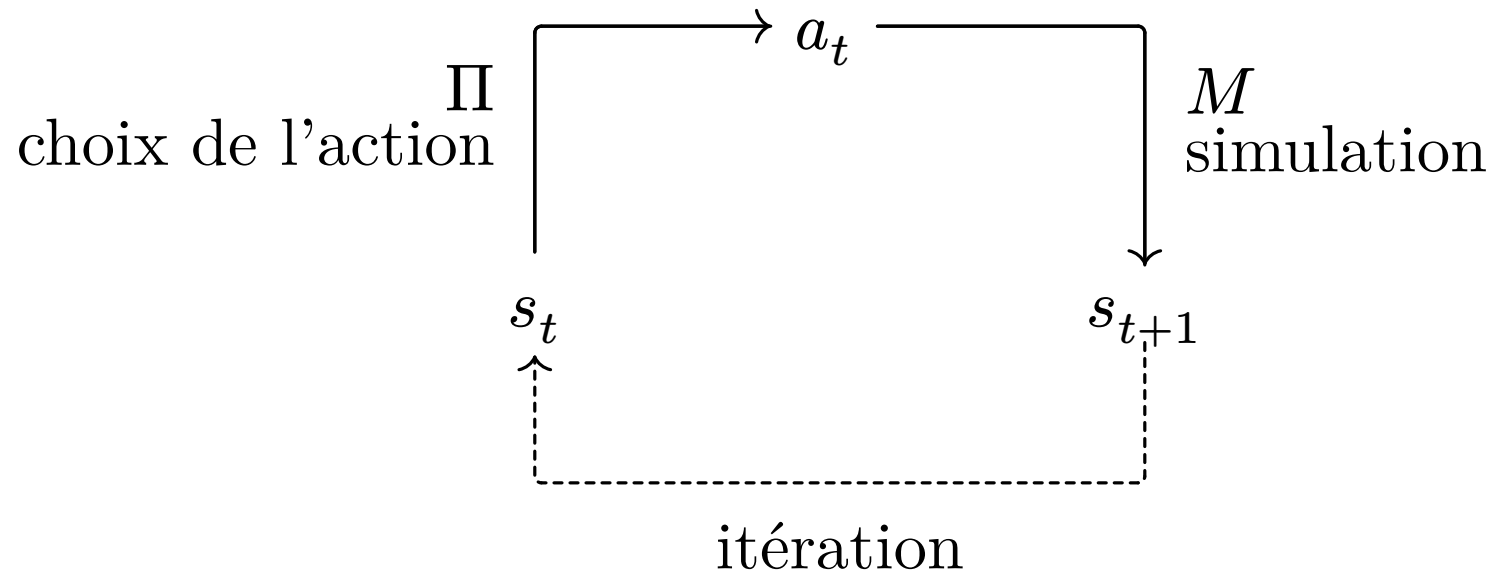
Comparaison des politiques



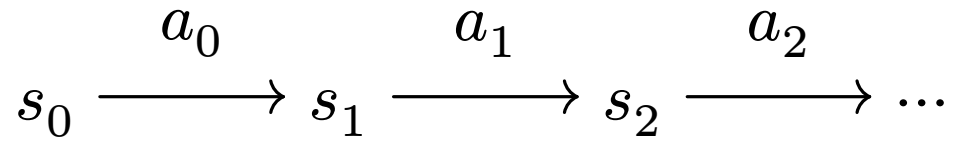
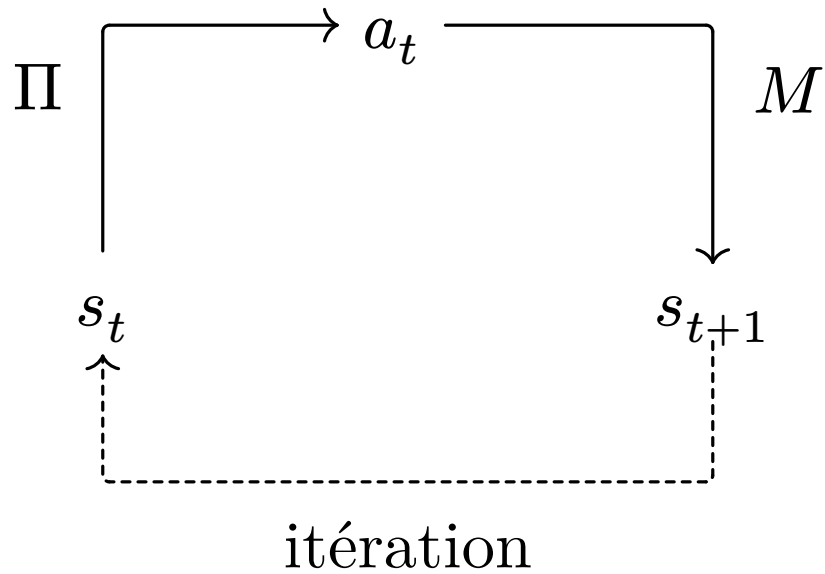
Comparaison des politiques



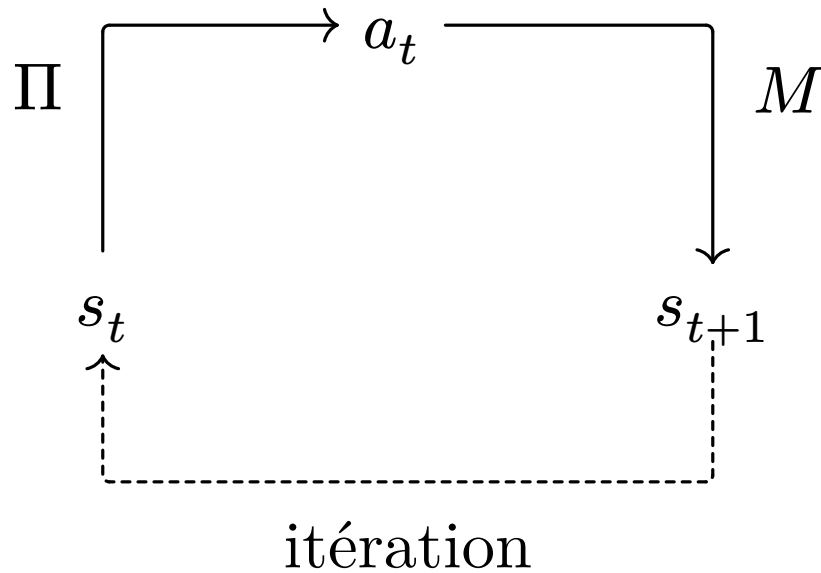
Comparaison des politiques



Comparaison des politiques



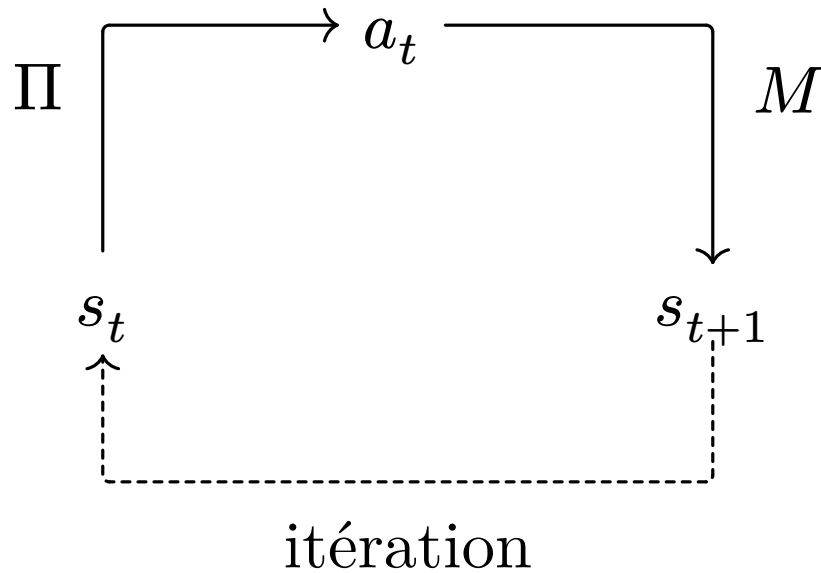
Comparaison des politiques



$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots$$

$$((s_0, a_0), (s_1, a_1), (s_2, a_2), \dots)$$

Comparaison des politiques



$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots$$

$$((s_0, a_0), (s_1, a_1), (s_2, a_2), \dots) \in \mathcal{C}$$

Comparaison des politiques

A := actions possibles

S := états possibles

$$\mathcal{C} := \left\{ \left\{ \begin{array}{ll} c_0 & = (s_0, a_0) \\ \forall t \in \mathbb{N} & c_{t+1} = (M(c_t), a_t) \end{array} \right. \middle| (s_0, a) \in S \times A^{\mathbb{N}} \right\}$$

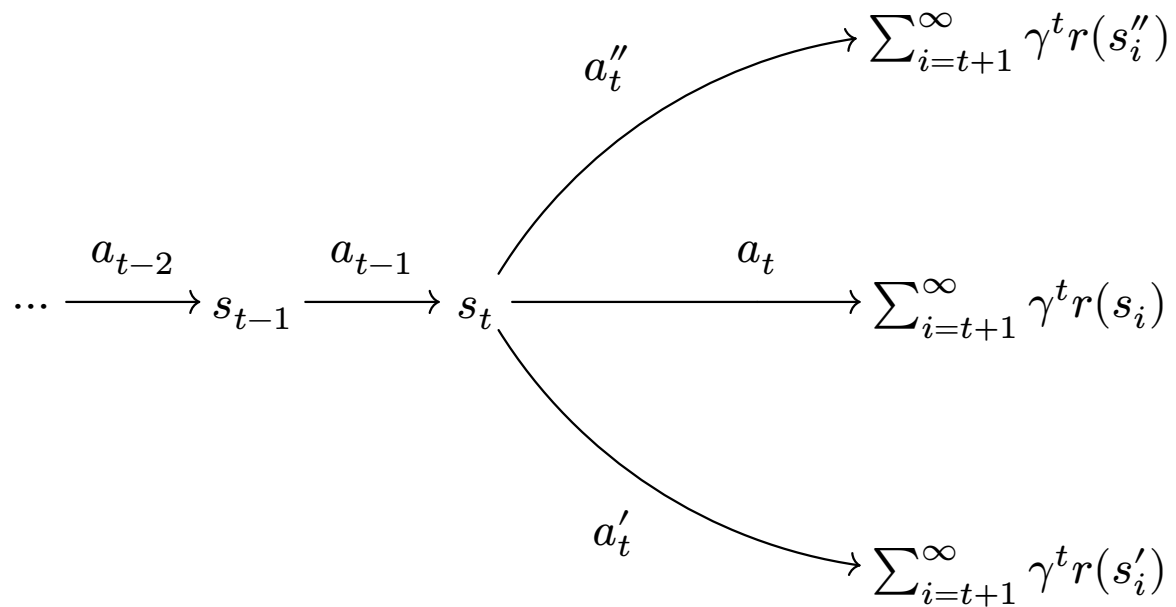
Comparaison des politiques: **Avantage A**

À quel point est-il mieux de choisir a_t plutôt qu'une autre action?

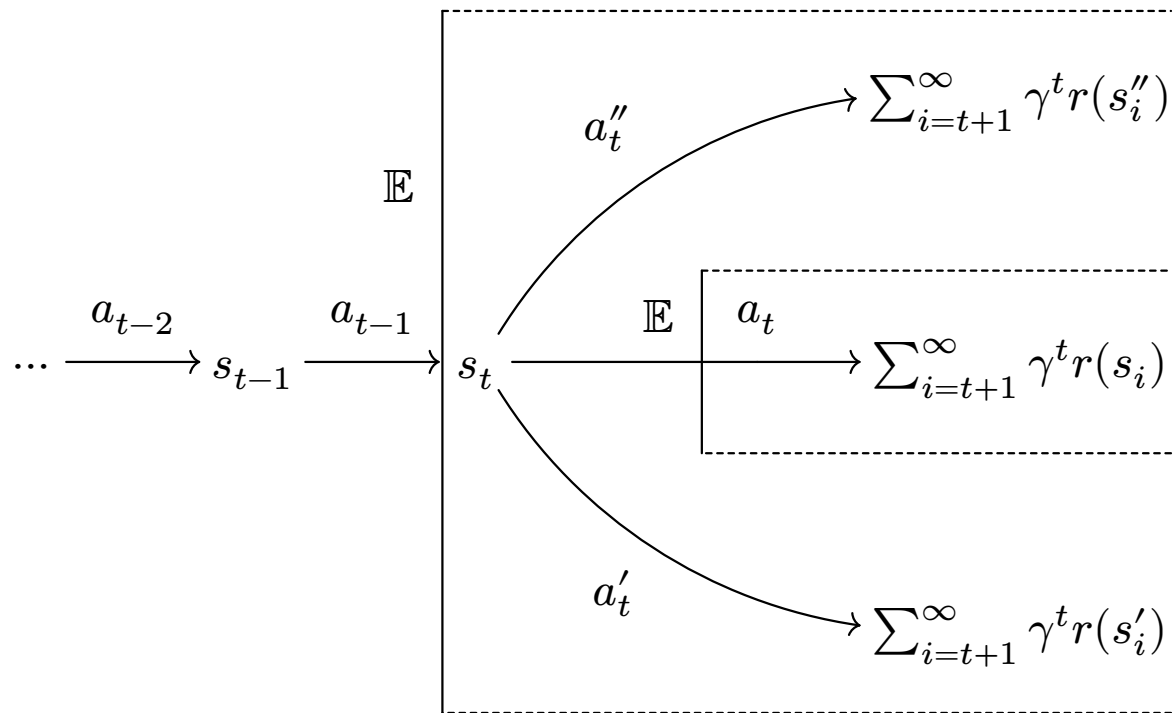
Comparaison des politiques: *Avantage A*

$$\dots \xrightarrow{a_{t-2}} s_{t-1} \xrightarrow{a_{t-1}} \dots$$

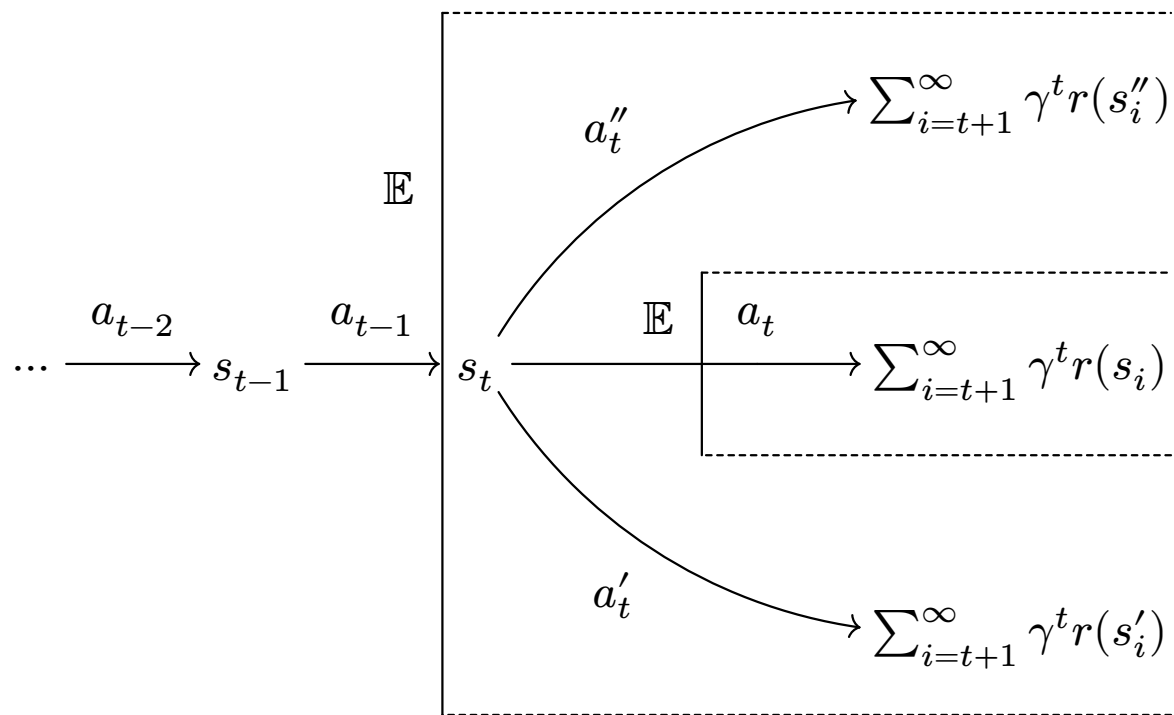
Comparaison des politiques: Avantage A



Comparaison des politiques: Avantage A



Comparaison des politiques: Avantage A



$$A_{\pi,r}(s, a) := \mathbb{E}(\text{avec } a_t) - \mathbb{E}(\text{\`a } t - 1)$$

C'est quoi \mathcal{L} ?

$$\mathcal{L}_r(\pi', \pi) :=$$

C'est quoi \mathcal{L} ?

$$\mathcal{L}_r(\pi', \pi) := \mathbb{E}_{(s_t, a_t)_{t \in \mathbb{N}} \in \mathcal{C}}$$

C'est quoi \mathcal{L} ?

$$\mathcal{L}_r(\pi', \pi) := \mathbb{E}_{(s_t, a_t)_{t \in \mathbb{N}} \in \mathcal{C}} \sum_{t=0}^{\infty}$$

C'est quoi \mathcal{L} ?

$$\mathcal{L}_r(\pi', \pi) := \mathbb{E}_{(s_t, a_t)_{t \in \mathbb{N}} \in \mathcal{C}} \sum_{t=0}^{\infty} \frac{Q_{\pi}(s_t, a_t)}{Q_{\pi'}(s_t, a_t)} A_{\pi, r}(s_t, a_t)$$

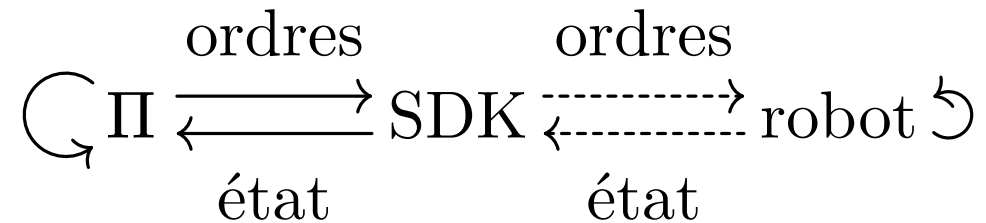
Mise à jour de Π

$$\Pi' = \begin{cases} \operatorname{argmax}_{\pi} \mathcal{L}_r(\pi, \Pi) \\ \text{s.c. } \text{distance}(\Pi', \Pi) < \delta \end{cases}$$

Le *SDK*¹ d'Unitree

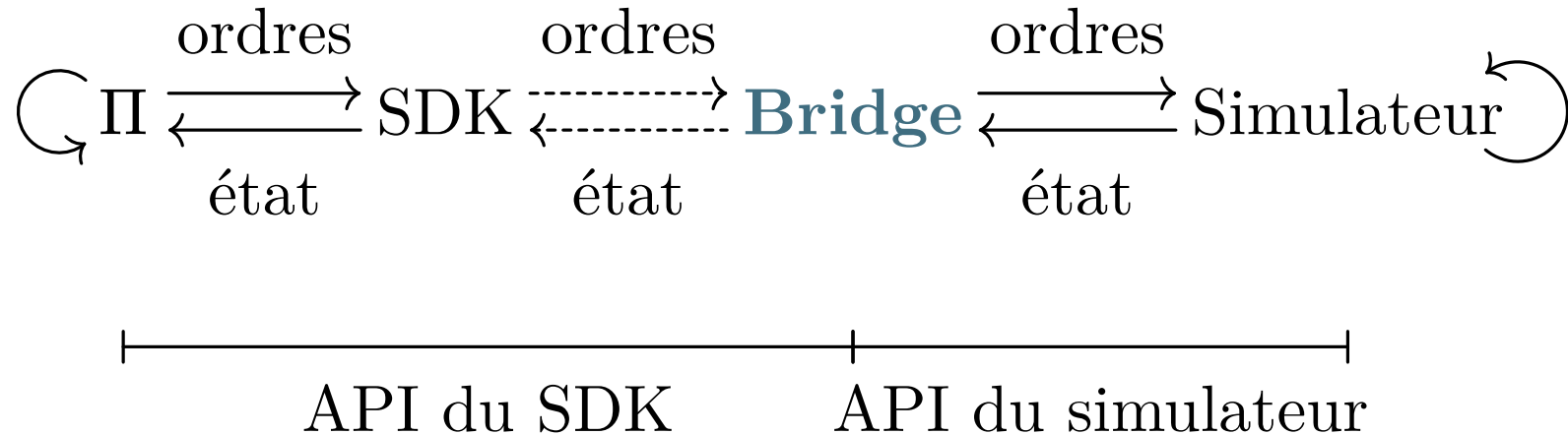
¹Software Development Kit

Le SDK d'Unitree

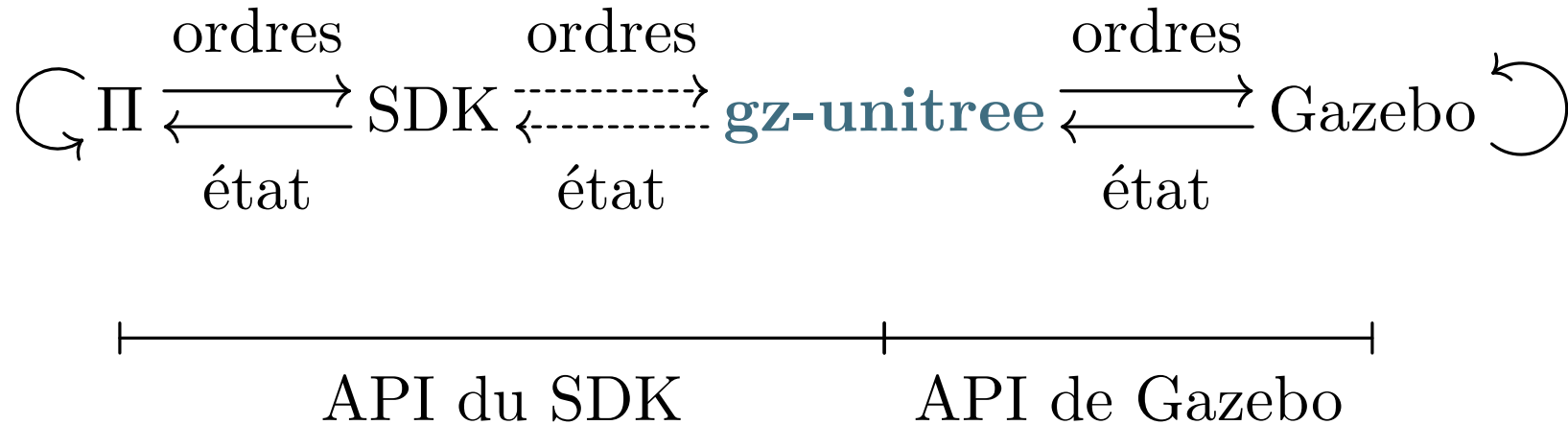


-----> Message DDS

Le SDK d'Unitree



Le SDK d'Unitree





II

gz::sim::System

::Configure

::PreUpdate

II

Unitree SDK

gz::sim::System

::Configure

::PreUpdate

ChannelFactory
domaine 1, interface 1o

ChannelPublisher
canal rt/lowstate

ChannelSubscriber
canal rt/lowcmd

II

Unitree SDK

gz::sim::System

::Configure

::PreUpdate

ChannelFactory
domaine 1, interface 1o

ChannelPublisher
canal rt/lowstate

ChannelSubscriber
canal rt/lowcmd

::LowStateWriter

::CmdHandler

State buffer
statebuf

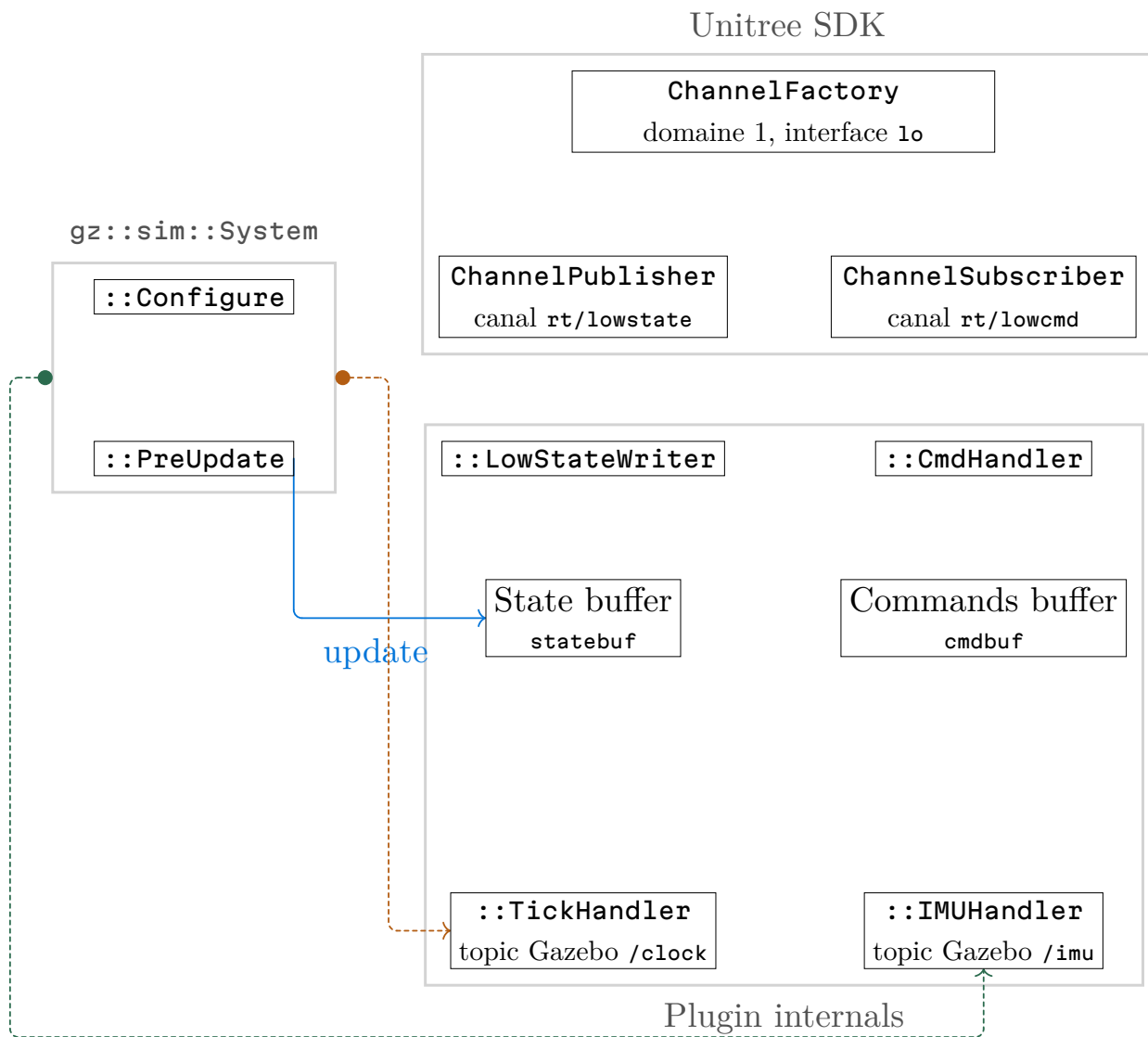
Commands buffer
cmdbuf

::TickHandler
topic Gazebo /clock

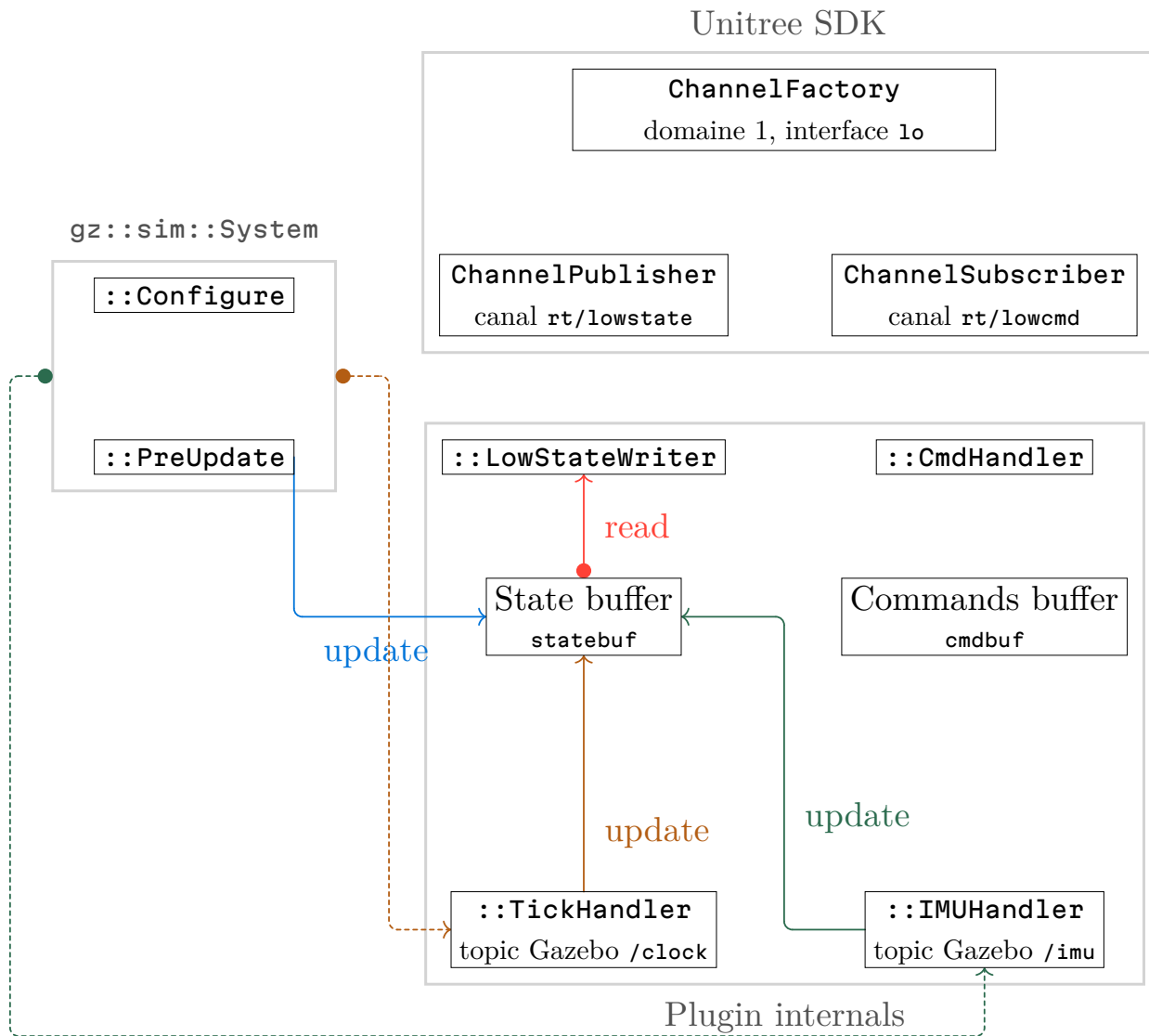
::IMUHandler
topic Gazebo /imu

Plugin internals

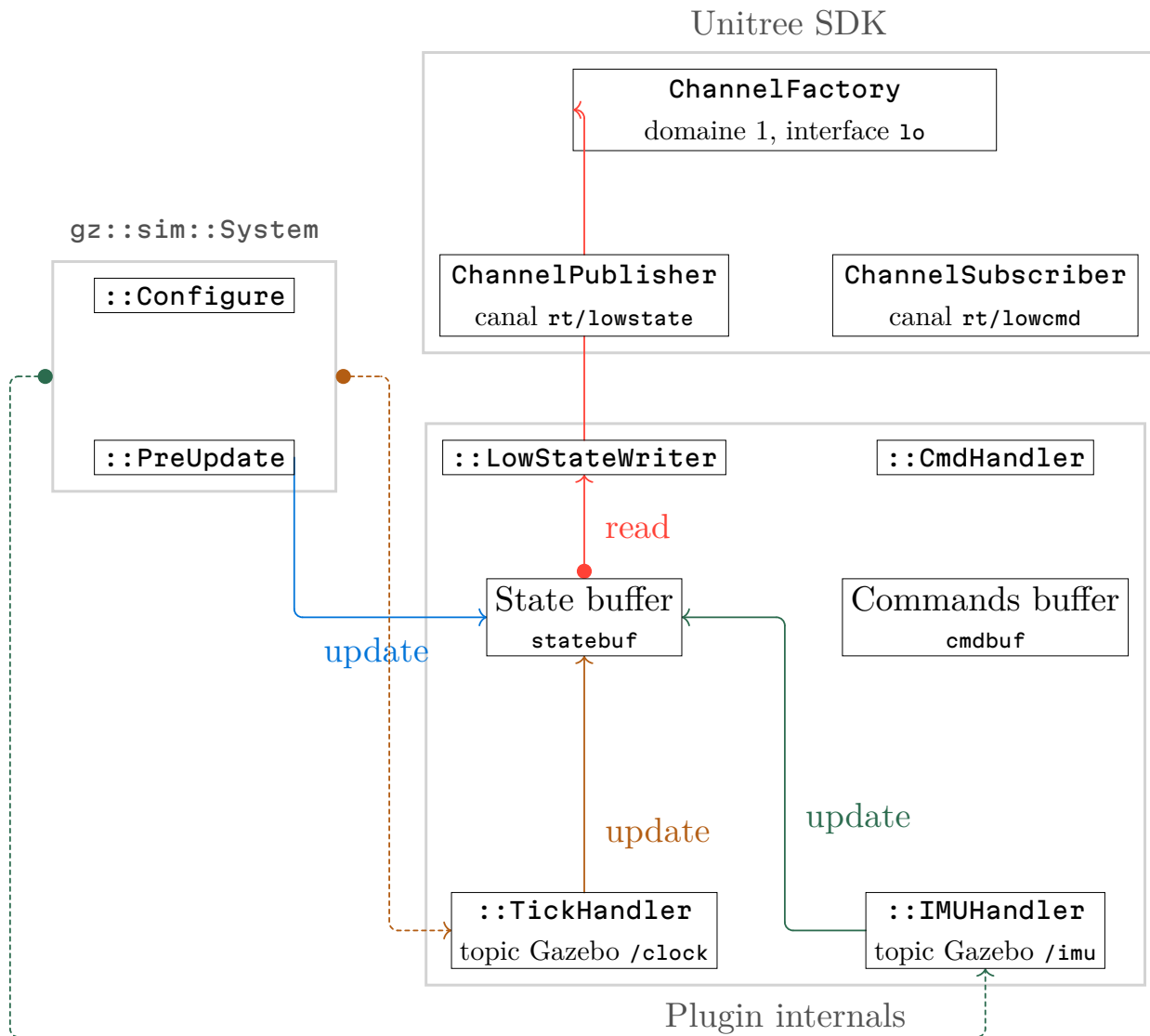
II

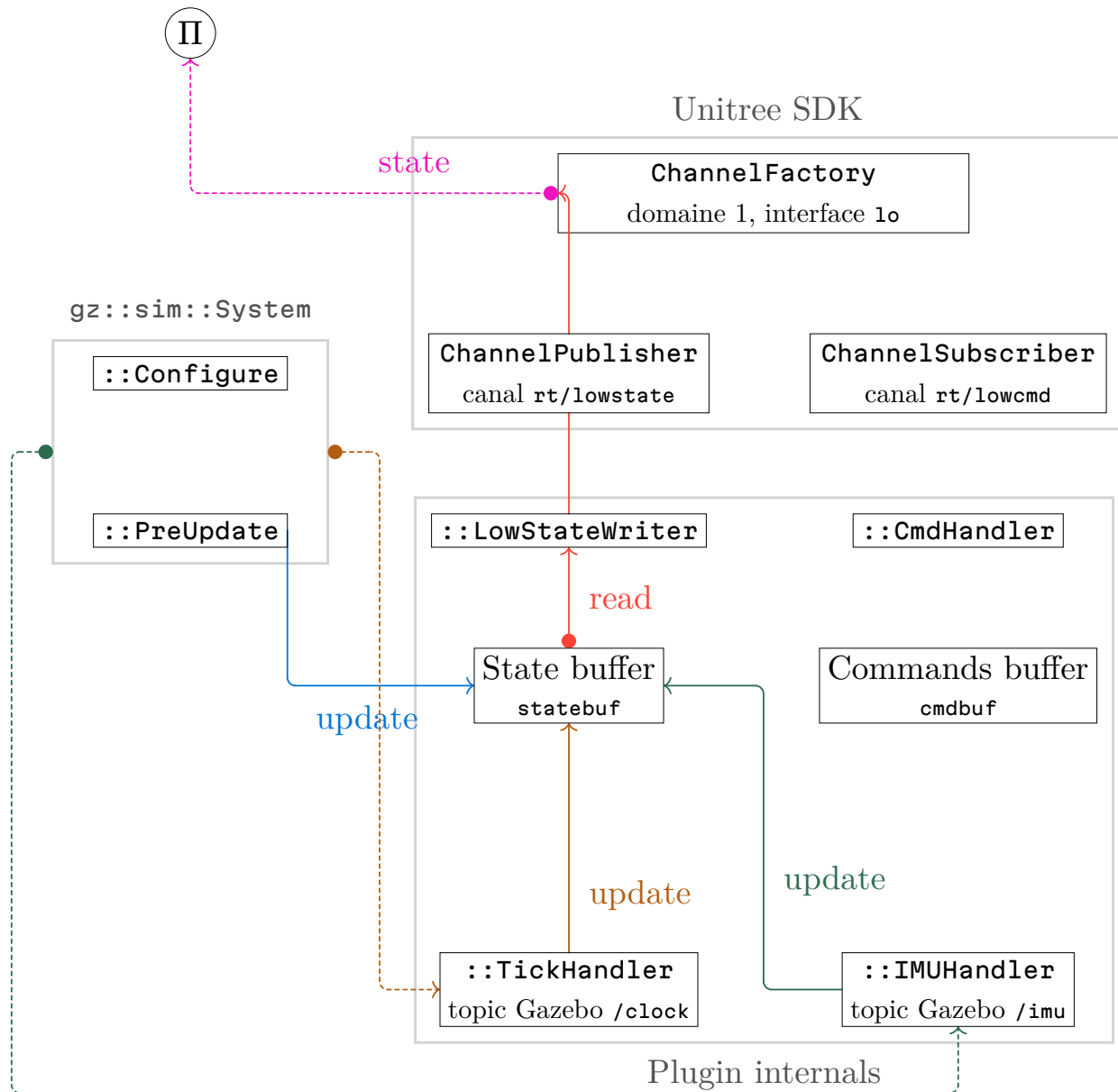


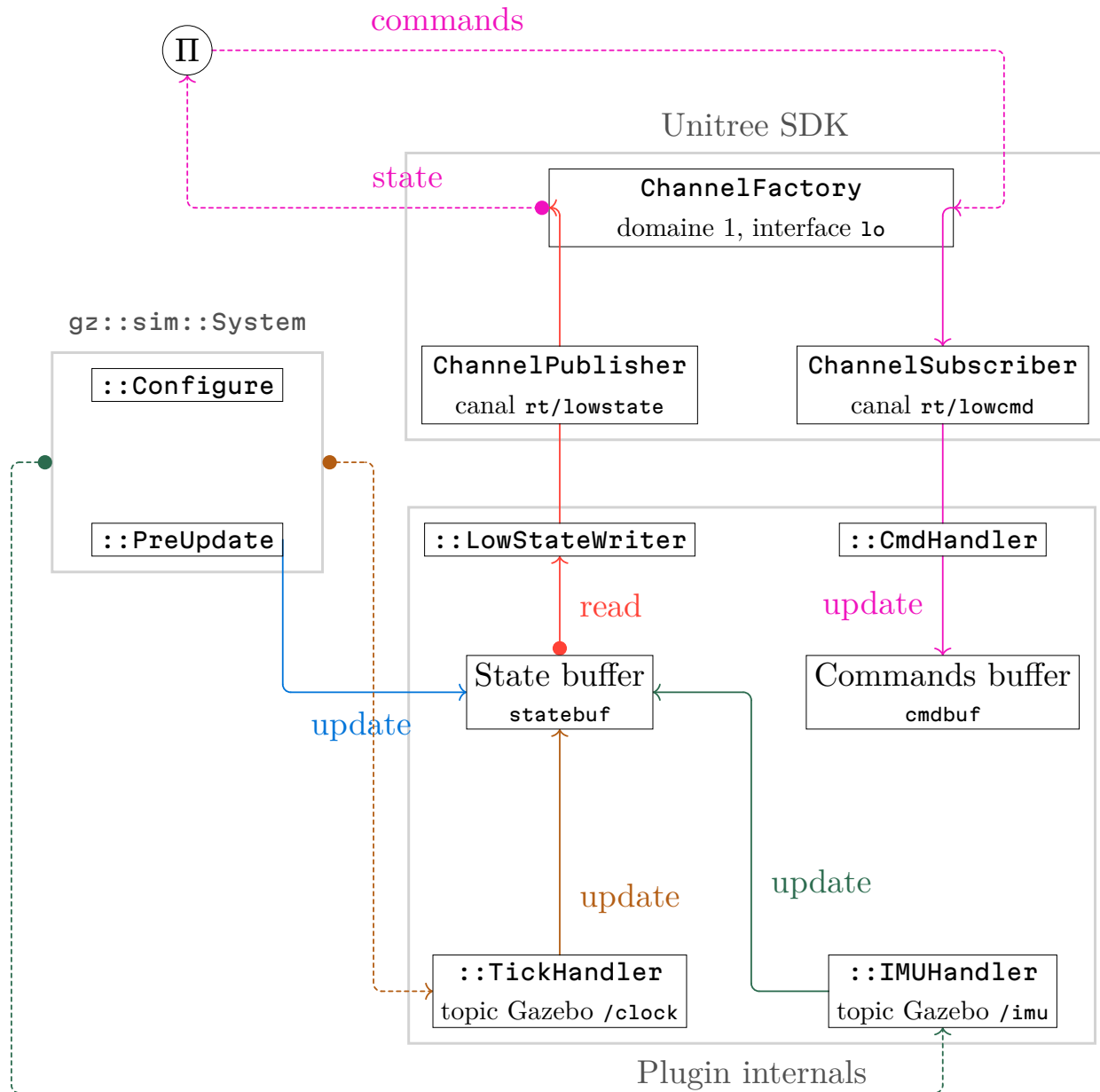
II

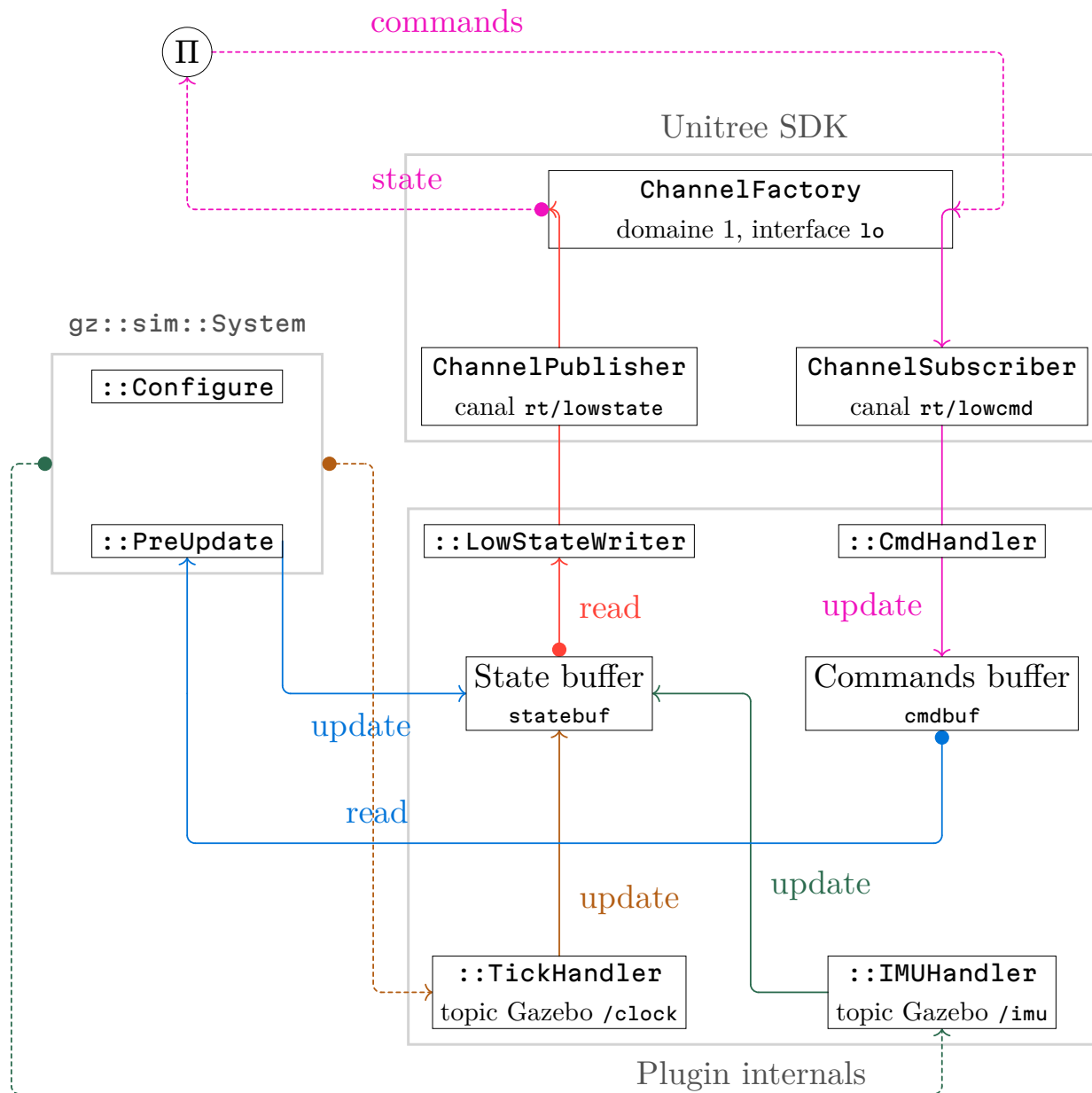


II









Reproductibilité

Avec Nix

Reproductibilité

```
from datetime import date

def f(a):
    return date.today().year + a
```

Reproductibilité

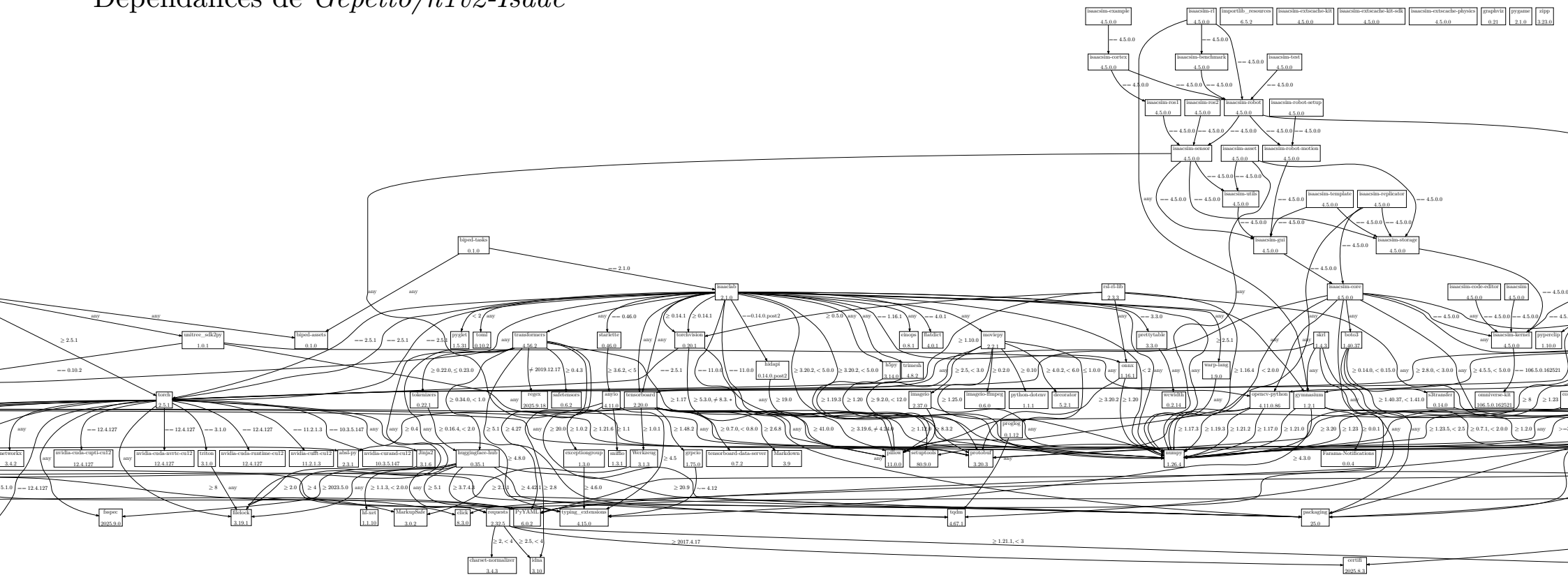
```
from datetime import date

def f(a):
    return date.today().year + a
```

Reproductibilité

Au compile-time

Beaucoup de dépendances!

Dépendances de *Gepetto/h1v2-Isaac*

Reproductibilité

```
./program.c  
int main()  
{  
    printf("Built at %s\n", BUILT_AT);  
}
```

puis

```
gcc -DBUILT_AT="\$(date)" program.c
```

Nix, le langage

Pour des paquets reproductibles

Définition de gz-unitree

```
{ lib, stdenv, fetchFromGitHub, cmake, eigen }:  
  
stdenv.mkDerivation {  
  pname = "unitree-sdk2";  
  version = "0.1.0";  
  
  src = fetchFromGitHub {  
    owner = "unitreerobotics";  
    repo = "unitree_sdk2";  
    rev = version;  
    hash = "sha256-r05zwhZW36+VOrIuTCr2HLf2R23csmnj33JFzUqz62Q=";  
  };  
  
  nativeBuildInputs = [ cmake ];  
  buildInputs = [ eigen ];  
  meta = { ... };  
}
```

Définition de gz-unitree

```
{ lib, stdenv, fetchFromGitHub, cmake, eigen }:
```

```
stdenv.mkDerivation {  
  pname = "unitree-sdk2";  
  version = "0.1.0";  
  
  src = fetchFromGitHub {  
    owner = "unitreerobotics";  
    repo = "unitree_sdk2";  
    rev = version;  
    hash = "sha256-r05zwhZW36+V0rIuTCr2HLf2R23csmnj33JFzUqz62Q=";  
  };  
  
  nativeBuildInputs = [ cmake ];  
  buildInputs = [ eigen ];  
  meta = { ... };  
}
```

Définition de gz-unitree

```
{ lib, stdenv, fetchFromGitHub, cmake, eigen }:
```

```
stdenv.mkDerivation {  
  pname = "unitree-sdk2";  
  version = "0.1.0";  
  
  src = fetchFromGitHub {  
    owner = "unitreerobotics";  
    repo = "unitree_sdk2";  
    rev = version;  
    hash = "sha256-r05zwhZW36+V0rIuTCr2HLf2R23csmnj33JFzUqz62Q=";  
  };  
  
  nativeBuildInputs = [ cmake ];  
  buildInputs = [ eigen ];  
  meta = { ... };  
}
```

Définition de gz-unitree

```
{ lib, stdenv, fetchFromGitHub, cmake, eigen }:
```

```
stdenv.mkDerivation {  
  pname = "unitree-sdk2";  
  version = "0.1.0";  
  
  src = fetchFromGitHub {  
    owner = "unitreerobotics";  
    repo = "unitree_sdk2";  
    rev = version;  
    hash = "sha256-r05zwhZW36+V0rIuTCr2HLf2R23csmnj33JFzUqz62Q=";  
  };  
  
  nativeBuildInputs = [ cmake ];  
  buildInputs = [ eigen ];  
  meta = { ... };  
}
```

Définition de nixpkgs#eigen

```
{ lib, stdenv, fetchFromGitLab, cmake }:
```

```
stdenv.mkDerivation {
  pname = "eigen";
  version = "3.4.0-unstable-2022-05-19";

  src = fetchFromGitLab {
    owner = "libeigen";
    repo = "eigen";
    rev = "e7248b26a1ed53fa030c5c459f7ea095dfd276ac";
    hash = "sha256-uQ1YYV3ojbMVfHdqjXRyUymRPjJZV3WHT36PTxPRius=";
  };

  nativeBuildInputs = [ cmake ];
  patches = [ ./include-dir.patch ];
  postPatch = ''substituteInPlace Eigen/src/SVD/BDCSVD.h --replace-fail "if (l == 0) {" "if (i
    >= k && l == 0) {''';
  meta = { ... };
}
```

Définition de nixpkgs#cmake

```
{
  lib,
  stdenv,
  fetchurl,
  replaceVars,
  buildPackages,
  bzip2,
  curlMinimal,
  expat,
  libarchive,
  libuv,
  ncurses,
  openssl,
  pkg-config,
  ps,
  sysctl,
  rhash,
  sphinx,
  texinfo,
  xz,
  zlib,
  darwin,
  isBootstrap ? null,
  isMinimalBuild ? (
    if isBootstrap != null then
      lib.warn "isBootstrap argument is deprecated and will be
        removed; use isMinimalBuild instead" isBootstrap
    else
      false
  ),
  useOpenSSL ? !isMinimalBuild,
  useSharedLibraries ? (!isMinimalBuild && !
    stdenv.hostPlatform.isCygwin),
  uiToolkits ? [ ], # can contain "ncurses" and/or "qt5"
  buildDocs ? !(isMinimalBuild || (uiToolkits == [ ])),
  libsForQt5,
  gitUpdater,
}:
```