Taichi

https://github.com/taichi-dev/taichi

March 6, 2021

Content

Installation

Taichi can be installed via pip on 64-bit Python 3.6,3.7,3.8

```
pip install taichi
```

- Taichi supports Windows, Linux, and OS X
- Taichi runs on both CPUs and GPUs (CUDA,OpenGL,Apple Metal).

Content

Initialization

```
ti.init(arch=ti.cuda)
```

- ti.x64[arm,cuda,opengl,metal]: stick to a certain backend.
- ti.cpu (default)
- ti.gpu[cuda,metal,opengl]

Content

Data types

ti.i8/i16/i32/i64
ti.u8/u16/u32/u64
ti.f32/f64

- tensors
  - scalar: ti.field
  - vector: ti.Vector
  - matrix: ti.Matrix

$eg : ex01.py$

Content

Kernels

Computation resides in kernels.

- compiled,
- statically-typed,
- parallel
- differentiable

$eg : ex02.py$

Content

Functions

- Taichi functions can be called by Taichi kernels and other Taichi functions
- Taichi functions will be force-inlined

$eg : ex03.py$

Content

Loops

- Range-for Loops:which are no different from Python for loops, except that it will be parallelized when used at the outermost scope. Range-for loops can be nested.

- Struct-for loops: which iterates over (sparse) tensor elements

For loops at the outermost scope in a Taichi kernel are automatically parallelized.

- range-for loop : $eg : ex04.py$
- struct-for loop: $eg : ex05.py$

Content

Atomic Operations

In Taichi, augmented assignments are automatically atomic

```
x[i] += 1
x[i] -= 1
x[i] *= 2
x[i] /= 2
...
```

$eg : ex06.py$

Content

Taichi-scope v.s. Python-scope

- Taichi-scope: Everything decorated with ti.kernel and ti.func.
- Python-scope: Code outside the Taichi-scope.

1. Code in Taichi-scope will be compiled by the Taichi compiler and run on parallel devices.
2. Code in Python-scope is simply Python code and will be executed by the Python interpreter.