PyPitch

Release 2.0

Jul 21, 2023

User guide

PyPitch is a C++-extension in Python to analyse audio streams for pitch.

CHAPTER 1

Installation

1.1 Install package with pip

To install with pip:

```
$ pip install pypitch
```

1.2 Install from source

To install pypitch from the latest source, first obtain the source code:

```
$ git clone https://github.com/fofix/python-pypitch
$ cd python-pypitch
```

then build the extension:

```
$ python setup.py build_ext --inplace
```

then install with:

```
$ pip install .
```

or:

```
$ pip install -e .
```

for development.

1.3 Dependencies

1.3.1 Python Packages

To use this package, you will need `Cython`.

1.3.2 Compiler Toolchain

The same compiler toolchain used to build the CPython interpreter should also be available. Refer to the CPython Developer's Guide for details about the compiler toolchain for your operating system.

CHAPTER 2

Usage

2.1 Import

To use `PyPitch` in your project:

```
from pypitch import pypitch
```

2.2 Modules

There a two modules available:

- `Analyzer`: to analyze input audio and transform it into usable data
- `Tone`: to represent tones.

CHAPTER 3

History

This module was extracted from FoFiX, and was inspired from Performous.

First, it was a pure-Python script named PitchAnalyzer. This script is available in the version 3 of FoFiX.

To speed up this FoFiX, this script were adapted in C++ in the version 4.

CHAPTER 4

Release notes

4.1 2.0 (2023-07-20)

- build: drop python 3.6 support
- build: add python 3.10 support
- build: add python 3.11 support
- build: remove *pytest-runner* from setup requirements

4.2 1.3 (2020-12-17)

- Python 3.9 compatibility
- Setup: update contact information
- CI: replace Travis with Github Actions
- Code: remove the *sqrLT* function

4.3 1.2 (2020-04-12)

- Python 3 compatible (CI, code, tests)
- Code: use *std::fmod* instead of *std::remainder* only for Visual C++ 9.0 (Windows & python 2.7)
- Doc: add a release section
- Setup: use `bumpversion` to bump the version easily

4.4 1.1 (2020-04-10)

- Doc: initialise some basics
- Setup: correctly use the relative path of the README file
- Setup: declare ‘docs’ extra for the documentation
- Setup: declare ‘tests’ extra for tests
- Setup / osx: use libc++ to compile the extension (xcode >= 10)
- Tests: restrict the pytest version (< 5)

4.5 1.0 (2020-03-15)

Initial release. Extracted from FoFiX.

CHAPTER 5

How to release

5.1 Create a release

Here is the release process for `PyPitch`:

1. bump the version (will commit):

```
python -m pip install bumpversion
bumpversion release
```

2. update the changelog date (in the same commit):

```
vim CHANGELOG.md
git add CHANGELOG.md
git commit --amend
```

3. make a PR
4. if tests pass on Travis, merge the PR
5. make a tag:

```
git tag VERSION
```

6. make the source distribution:

```
python setup.py build_ext --inplace --force
python setup.py sdist
```

7. make wheels with AppVeyor
8. upload sources and wheels on GitHub in a new release.

5.2 Upload to PyPI

You need to upload sources and wheels to PyPI:

```
python -m pip install twine
twine upload -r pypi dist/*
```

Note: First, upload on TestPyPI:

```
$ twine upload -r pypitest dist/*
```

Then, you need to test the installation:

```
pushd $(mktemp -d)
mktmpenv
pip install -i https://test.pypi.org/simple pypitch
python -c "from pypitch import pypitch; print(pypitch.__version__)"
deactivate
popd
```

5.3 Prepare the next release

To prepare the next release:

1. bump the version (with PART in [major, minor, patch], will commit):

   ```
   bumpversion PART
   ```

2. update the changelog to add the unreleased subsection (in the same commit):

   ```
   vim CHANGELOG.md
   git add CHANGELOG.md
   git commit --amend
   ```

3. make a PR.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search