بسمه تعالی

پروژه پایانی درس سیگنال ها و سیستم ها

آیسا میاهی نیا ۹۹۵۲۲۱۴۹

استاد درس:

دکتر محمدی

الگوریتم های متعددی برای **time stretching** وجود دارد در حالی که فرکانس اساسی آن را تغییر ندهد یا به عبارتی **pitch** را حفظ کند. هر کدام از الگوریتم ها به صورت خلاصه توضیح داده شده اند. منبع این لینک می باشد.

- Overlap-Add (OLA)
  - OLA is the most simple TSM algorithm that changes the length of the signal through modifying the hop size between analysis frame and synthesis frame.
- Pitch Synchronous Overlap-Add (TD-PSOLA)
  - TD-PSOLA is the algorithm that analyzes the original waveforms to create pitch-synchronous analysis windows and synthesize the output signal both for modifying time-scale and pitch-scale.
- Waveform Similarity Overlap-Add (WSOLA)
  - WSOLA maximizes the waveform similarity by allowing timing tolerance to analysis frame to find the most similar position through cross correlation.
- Phase Vocoder (PV)
  - Phase vocoder estimates instantaneous frequency, and it is used to update phases of input signal's frequency components in short-time Fourier transform. Although TSM results with phase vocoder has high phase continuity, it causes an transient smearing for percussive audio sources and a coloring artifact called phasiness.
- TSM based on harmonic-percussive source separation (HPTSM)
  - A novel TSM algorithm that applying phase vocoder to only harmonic sources and applying OLA to only percussive sources.

اما کاری که من برای تغییر سرعت صوت انجام دادم به این صورت می باشد که ابتدا فرکانس صوت خوانده شده را ضربدر آن ضریبی که به عنوان ورودی داده شده است میکنم و آرایه دیتا جدید با فرکانس هایی که ضربدر ضریب شدند حاصل می شود. اما اگر همین دیتا را با استفاده از دستور **write**یک فایل خروجی بدهیم و آن را گوش کنیم میبینیم که صدا (با توجه به ضریب) زیر یا بم تر می شود که خب این خوب نیست برای اینکه این اتفاق نیوفتد لازمه تا شیفت فرکانسی به اندازه لازم داده شود.از آنجایی که **Pitch**گوینده باید مقدارش تغییر نکنه پس ما به

اندازه ای شیفت میدیم که مقدار **pitch**جدید با قدیم برابر شول پس به اندازه تفاضل این دو شیفت فرکانسی انجام میدهیم. اما نکته ای که باید به آن توجه کرد این است که در **numpy** آرایه ها از صفر شروع می شوند ولی دیتا ما باید نسبت پی متقارن باشد پس باید یکبار که شیفت به راست میدهیم و یکبار هم به چپ آرایه دیتا اصلی را شیفت دهیم بعد این دو را کنار هم قرار دهیم. برای اینکه بخواهیم شیفت فرکانسی بدهیم لازمه که دیتا ما در حوزه فرکانس باشد پس از تبدیل فوریه برای اینکار استفاده میکنیم. البته میتوان از خواص شیفت فرکانسی هم استفاده کرد و در همان حوزه زمان هم کار کرد، همان کاری که در کد صورت گرفته است. آن را **write** میکنیم با سمپل ریت جدید که برابره با سمپل ریت قبلی ضربدر **speed**.

برای محاسبه **pitch** از تابع **pitch_finder** استفاده شده است که نحوه پیاده سازی آن از این لینک مشاهده کرد.