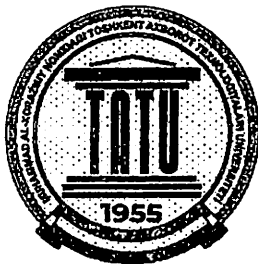


U 1406

**O'ZBEKISTON RESPUBLIKASI RAQAMLI TEXNOLOGIYALAR
VAZIRLIGI
MUHAMMAD AL-XORAZMIY NOMIDAGI
TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI**

**TELEVIZION TEXNOLOGIYALAR FAKULTETI
“AUDIOVIZUAL TEXNOLOGIYALAR” KAFEDRASI**



**“OVOZ VA TASVIRNI QAYTA ISHLASH” FANIDAN LABORATORIYA
ISHLARINI BAJARISH UCHUN USLUBIY QO'LLANMA**

TOSHKENT 2023

Sempling signallarini qayta ishlashi – bu uzluksiz signallarni diskret qiymatlar seryasiga o'zgartirilishidir. Diskretizatsiya chastotasi – bu vaqtning oralig'idagi sempillar sonidir. Diskretizatsiyaning yuqori chastotasi axborotni kam yo'qotilishiga, biroq katta hisoblash xarajatlariga olib keladi.

Ovozni qayta ishlash bo'yicha ilova

Unga quyidagilarni kiritish mumkin:

- Musiqali kolleksiyalarni ularning audiobelgilariga muvofiq indekslash.
- Radiokanallar uchun musiqa tavsiyasi.
- Audiofayllar uchun o'xshashlikni qidirish (Shazam).
- Ovozni qayta ishlash va sintez qilish – suhbat agentlari uchun sun'iy ovozni generatsiyalash.

Python dasturlash tili yordamida audioma'lumotlarni qayta ishlash

Ovoz chastota, o'tkazish chizig'i, detsibel va shu kabi parametrlar bilan audiosignal ko'rinishida taqdim qilingan. Oddiy audiosignalni amplituda va vaqt funksiyalari sifatida ifoda etish mumkin.

Ba'zi qurilmalar bu ovozlarni ushlab olishi va mashina o'qiy oladigan formatda taqdim qilishi mumkin. Quyida bu formatlarga misollar keltirilgan:

- wav (Waveform Audio File Format)
- mp3 (MPEG-1 Audio Layer 3)
- WMA (Windows Media Audio)

Ovozni qayta ishlash jarayoni bilimlarni birlashtirish, aniqlash va sinflashtirishdan iborat qaror qabul qilish sxemasini nazarda tutuvchi qo'yilgan vazifaga taalluqli akustik xarakteristikadagi ajratib olishni o'z ichiga oladi. Yaxshi tomoni shundaki, Python ba'zi kutubxonalari bu vazifani osonlashtirishga yordam beradi.

Python audio kutubxonalari

Biz audioni yig'ish va ishga tushurish uchun ikki kutubxonadan foydalanamiz:

1. Librosa

Librosa istalgan ovoz signallari bilan ishlashi mumkin, biroq asosan aynan musiqaga yo'naltirilgan. U to'la qonli musiqali axborotni ajratib olish tizimini (MIR) yaratish imkonini beradi. Modul to'liq hujjatlashtirilgan, bundan tashqari foydalanish bo'yicha ko'plab qo'llanmalarga ega.

Pyhton ni librosa kutubxonasini o'rnatish quyidagicha bo'ladi:

```
pip install librosa
```

yoki quyidagicha:

```
conda install -c conda-forge librosa
```

Siz audiosignallar kovertatsiyasi uchun tayyor yechimlarga ega `ffmpeg` modulini ham o'rnatish mumkin.

2. IPython.display.Audio

IPython.display.Audio bevosita audioni Jupyter Notebook da ishga tushurish imkonini beradi.

Audiofaylni yuklash

```
import librosa  
audio_path = './T08-violin.wav'  
x, sr = librosa.load(audio_path)  
print(type(x), type(sr))  
<class 'numpy.ndarray'> <class 'int'>  
print(x.shape, sr)  
(396688,) 22050
```

Bu kod audio vaqt qatorini 22 kGs diskretizatsiya chastotali (`sr`) NumPy massiviga aylantiradi. Defolt qiymatlarni o'zgartirish mumkin, masalan, 44,1 kGs ga.

```
librosa.load(audio_path, sr=44100)
```

yoki sempoillashni umuman o'chirish:

```
librosa.load(audio_path, sr=None)
```

Diskretizatsiya chastotasi – bu Gs yoki kGs larda o'Ichangan, soniyasiga jo'natiladigan, ovoz semllari (o'zgarishi) sonidir.

Ishga tushurish

Audion ishga tushurish uchun **IPython.display.Audio** dan foydalanamiz:

```
import IPython.display as ipd
```

```
ipd.Audio(audio_path)
```

Bu kod Jupyter Notebook da quyidagi vidjetni qaytaradi:



1-rasm. Ovozni vizuallashtirish

Ovozni vizuallashtirish

To'liqin shakli

librosa.display.waveplot foydalanib, audioma'lumotlar massivini vizuallashtirish mumkin:

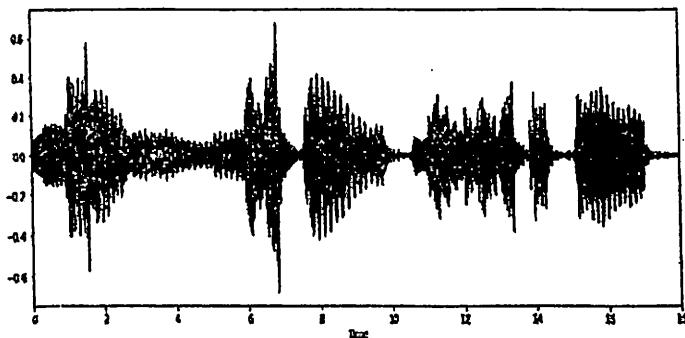
```
%matplotlib inline
```

```
import matplotlib.pyplot as plt
```

```
import librosa.display
```

```
plt.figure(figsize=(14, 5))
```

```
librosa.display.waveplot(x, sr=sr)
```



1.2-rasm. Amplitudali og‘uvchi signal grafigi

4. LABORATORIYA ISHIGA TOPSHIRIQ

Talaba tomonidan tanlangan asdiofayning ko‘rsatilgan formatlaridan biri uchun audiod faylni yuklash, ishga tushurish, vizuallashtirishni amalga oshirish.

5. LABORATORIYA ISHIGA KO‘RSATMA

Tanlangan audiod fayl uchun ko‘rsatilgan barcha amallarni bajarilgan so‘ng .py kengaytmali-kodni Google Disk dagi muvofiq papkaga jo‘natish.

6. HISOBOTGA QO‘YLADIGAN TALABLAR

Hisobot quyidagilardan iborat bo‘lishi kerak:

- talaba familiyasi ko‘rinishidagi nomga ega faylda saqlangan topshiriqni bajarilish natijalari.

7. NAZORAT SAVOLLARI

- 1) Ovoz nima?
- 2) Python dasturlash tili yordamida audioma'lumotlarni qayta ishlash
- 3) Audiod faylni yuklash, vizuallashtirish, ishga tushurish to‘g‘risida aytib bering.

LABORATORIYA ISHI №2

PYTHON DASTURLASH TILIDA SPEKTOGRAMMANI QAYTA ISHLASH

1. ISHDAN MAQSAD

- Ovoz va tasvirni tanib olishda qo'llaniladigan asosiy modellarni tahlillash to'g'risida tasavvurga ega bo'lish
- Python dasturlash tilida spektogrammani qayta ishlash asoslarini o'rganish

2. ISHGA TAYYORGARLIK

- Laboratoriya ishi tavsifini o'qib chiqish, Python ishchi muhitida fayllar bilan ishlashni o'rganish.
- Laboratoriya ishida keltirilgan savollarga javob berish.

3. NAZARIY QISM

Spektogramma – bu vaqt o'tishi bilan ovozli va boshqa signallarni chastota spektrini vizual ko'rinishidir. Ularni ba'zida sonogramma deb ham atashadi. Ikki o'lchamli grafikda birinchi o'q bo'yicha chastota, ikkinchi o'q bo'yicha esa vaqt beriladi.

Python da spektogramma yaratish uchun *librosa.display.specshow* foydalanamiz.

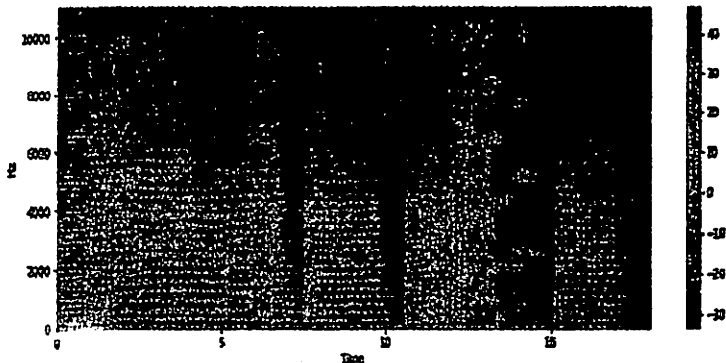
```
X = librosa.stft(x)
```

```
Xdb = librosa.amplitude_to_db(abs(X))
```

```
plt.figure(figsize=(14, 5))
```

```
librosa.display.specshow(Xdb, sr=sr, x_axis='time', y_axis='hz')
```

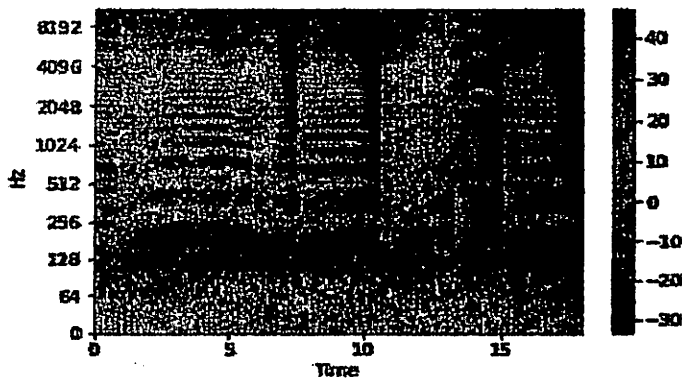
```
plt.colorbar()
```



2.1-rasm. Audiofayl spektrogrammasi

Vektikal o'q – bu chastota (0 dan 10 kGsgacha), gorizontal o'q esa klip vaqti. Barcha sezilarli o'zgarishlar spektrni pastki qismida ro'y berganligi sababli, chastota yoyini logorifmik ko'rinishda ifodalash mumkin.

```
librosa.display.specshow(Xdb, sr=sr, x_axis='time', y_axis='log')  
plt.colorbar()
```



2.2-rasm. Logorifmik o'qidan foydalanish

Audio yozuvi

librosa.output.write_wav NumPy massivni WAV-fayl qilib saqlaydi:

```
librosa.output.write_wav('example.wav', x, sr)
```

2. Ovoz signalini yaratish

Endi Python da 220 Gs chastotali ovozli signal yaratamiz. Bu Audio funksiyasiga beriladigan NumPy massivdir:

```
import numpy as np  
sr = 22050 # sample rate  
T = 5.0 # seconds  
t = np.linspace(0, T, int(T*sr), endpoint=False) # time variable  
x = 0.5*np.sin(2*np.pi*220*t) # pure sine wave at 220 Hz  
# Ishga tushurish  
ipd.Audio(x, rate=sr) # load a NumPy array  
  
# Saqlash  
librosa.output.write_wav('tone_220.wav', x, sr)  
[embed]https://soundcloud.com/parul-pandey-323138580/embed
```

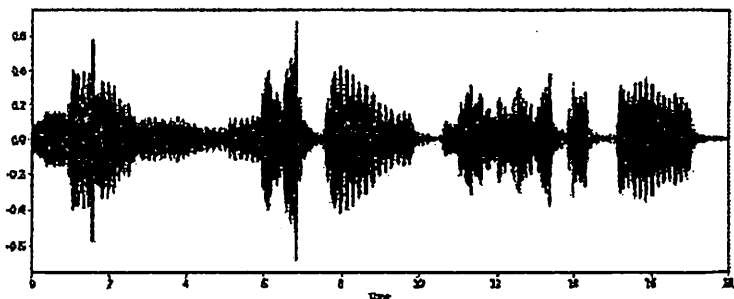
Mohiyatni ajratib olish

Har bir ovozli signal keraklilarini ajratib olish kerak bo'lgan ko'plab xarakteristikalariga ega. Tahlil uchun axborotni ajratib olish jarayoni obyektlarni ajratib olish yoki mohiyatni ajratib olish deb nomlanadi (feature extraction).

Nol orqali o'tish chastotasi

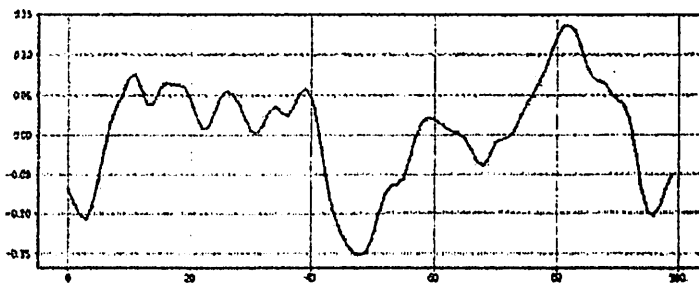
Nol kesishuvi chastotasi (zero crossing rate) – bu signal belgisini o'zgarish chastotasi, ya'ni signal musbatdan manfiyga va aksincha o'zgaradigan chastotadir. Bu funktsiya ovozni tanib olish uchun ham, musiqali axborotni ajratib olish uchun ham keng foydalaniladi. Metall va rok uchun bu parametr odatda katta miqdordagi zarbalar soni sababli boshqa janrlarga qaraganda yuqori.

Misolimiz uchun Python da nol orqali o'tish chastotasini hisblaymiz:



2.3-rasm. Nol orqali o'tish chastotasini aniqlash

```
x, sr = librosa.load('./T08-violin.wav')
plt.figure(figsize=(14, 5))
librosa.display.waveplot(x, sr=sr)
```



2.4-rasm. Signal grafigi

```
n0 = 9000
n1 = 9100
plt.figure(figsize=(14, 5))
plt.plot(x[n0:n1])
plt.grid()
```

Grafikda biz 6 ta nolni kesishuvini ko'rishimiz mumkin. Keling tekshiramiz:

```
zero_crossings = librosa.zero_crossings(x[n0:n1], pad=False)
print(sum(zero_crossings))
```

Spektral sentroid

Spektral sentroid ovoz “massa markazi” qayerda joylashganini ko‘rsatadi va barcha chastotalarni o‘rtacha o‘lchangan qiymati kabi hisoblanadi.

Blyuz kompozitsiyalarda chastota teng taqsimlangan va sentroid spektrni markazrog‘ida yotadi. Metalda kompozitsiya oxirida ifodalangan aralash chastota kuzatiladi, shu sababli spektroid spektrni oxiriga yaqin yotadi.

`librosa.feature.spectral_centroid` yordamida har bir fayl uchun spektral sentroidni hisoblaymiz:

```
spectral_centroids = librosa.feature.spectral_centroid(x, sr=sr)[0]
```

```
spectral_centroids.shape
```

```
(775,)
```

```
# Vizualizatsiya uchun vaqtni hisoblash
```

```
frames = range(len(spectral_centroids))
```

```
t = librosa.frames_to_time(frames)
```

```
# Spektral sentroidni normallashtirish
```

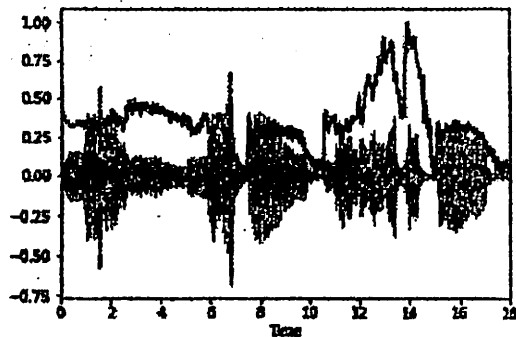
```
def normalize(x, axis=0):
```

```
    return sklearn.preprocessing.minmax_scale(x, axis=axis)
```

```
# Grafikni qurish
```

```
librosa.display.waveplot(x, sr=sr, alpha=0.4)
```

```
plt.plot(t, normalize(spectral_centroids), color='r')
```



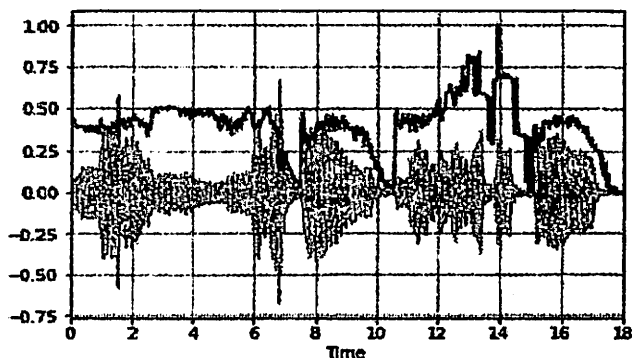
2.5-rasm. Spektr oxirida chastota o‘sishi ifodalangan grafik

Chatotani spektral qulashi

Bu umumiy spektral energiyani ma'lum foizida yotadigan chastota signalining shakli, masalan, 85% bo'ladi.

`librosa.feature.spectral_rolloff` har bir freym uchun chatota qulashi hisoblaydi:

```
spectral_rolloff = librosa.feature.spectral_rolloff(x+0.01, sr=sr)[0]  
librosa.display.waveplot(x, sr=sr, alpha=0.4)  
plt.plot(t, normalize(spectral_rolloff), color='r')
```

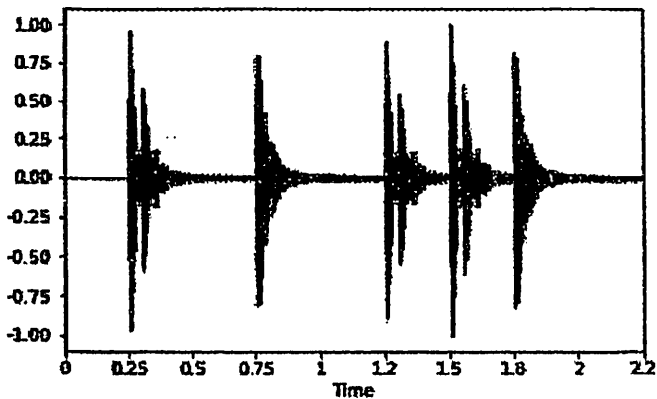


2.6-rasm. Chastota qulashi

Mel-chastotali kepstral koefitsientlar

Signalni mel-chastotali kepstral koefitsientlari (MFCC) – bu spektral og'ma umumiy shaklini siqilgan holda tavsiflovchi xarakteristikalarini uncha katta bo'lmagan to'plamidir. Bu parametr inson ovozi xarakteristikasini modellashtiradi.

Misol uchun sodda siklik to'liqini olamiz:



2.7-rasm. Siklik to'liqin

```
x, fs = librosa.load('./simple_loop.wav')
```

```
librosa.display.waveplot(x, sr=sr)
```

librosa.feature.mfcc yordamida bu koeffitsientlarni hisoblaymiz:

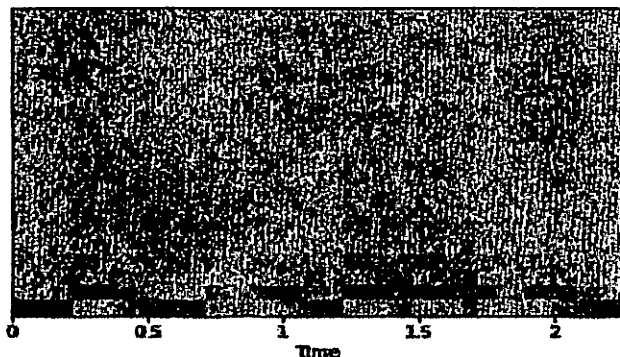
```
mfccs = librosa.feature.mfcc(x, sr=fs)
```

```
print mfccs.shape
```

```
(20, 97)
```

```
# Ko'rsatish
```

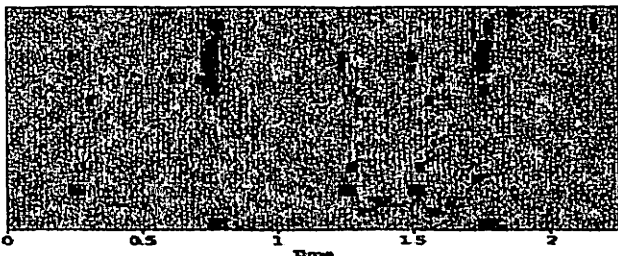
```
librosa.display.specshow(mfccs, sr=sr, x_axis='time')
```



2.8-rasm. Spektogammani ko'rsatish

Biz yana har bir koeffitsient o'lhovi nolli o'rta va yagona dispersiyaga ega bo'ladigan qilib masshtablashimiz mumkin:

```
import sklearn
mfccs = sklearn.preprocessing.scale(mfccs, axis=1)
print(mfccs.mean(axis=1))
print(mfccs.var(axis=1))
librosa.display.specshow(mfccs, sr=sr, x_axis='time')
```



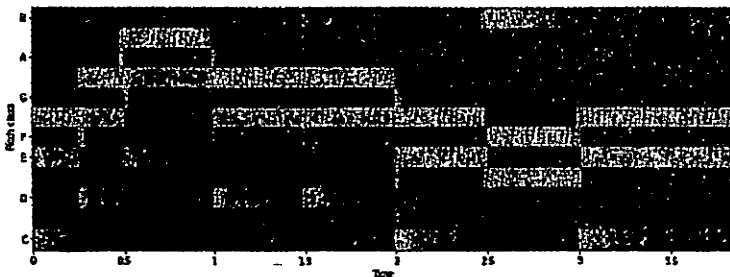
2.9-rasm. Rangbaranglik chastotasi

Rangbaranglik chastotasi

Rangbaranglik (chroma features) – bu butun spektr 12 turli musiqa oktavasini yarim tonlarini ifodalovchi 12 konteynerlarga proeksiyalanadigan musiqali ovoz uchun kuchli ko'rinishidir.

Hisoblash uchun `librosa.feature.chroma_stft` foydalaniladi:

```
# Fayl yuklanishi
x, sr = librosa.load('./simple_piano.wav')
hop_length = 512
chromagram = librosa.feature.chroma_stft(x, sr=sr, hop_length=hop_length)
plt.figure(figsize=(15, 5))
librosa.display.specshow(chromagram, x_axis='time', y_axis='chroma',
hop_length=hop_length, cmap='coolwarm')
```



2.10-rasm. Spektrni proeksiyalash

4. LABORATORIYA ISHIGA TOPSHIRIQLAR

1. Yozilgan audio bo'yicha ovoqli signal yarating
2. Tanlangan audio fayl uchun mohiyatni ajratishni amalga oshiring.
3. Nol orqali o'tish chastotasini hisoblang.
4. Grafik signalini oling.
5. Har bir freym uchun spektral sentroidni hisoblang.
6. Mel-chastotali kepral koeffitsientni hisoblang.
7. Rangbaranglik chastotasini hisoblang.

5. HISOBOTGA QO'YILADIGAN TALABLAR

Hisobot quyidagilardan iborat bo'lishi lozim:

- Joriy fayllar va topshiriq bajarilish natijalari talaba familiyasi ko'rinishidagi nomli faylda saqlangan bo'lishi lozim.

6. NAZORAT SAVOLLARI

- 1) Audioda spektogamma roli?
- 2) Ovoqli signal yaratish jarayonini tavsiflang?
- 3) Mohiyatni ajratib olish nima?

- 4) Nol orqali o'tish chastotasi nima?
- 5) Spektral sentroid?
- 6) Chastota spektral qulashi?
- 7) Mel chastotali kepstral koeffitsient?
- 8) Rangbaranglik chastotasi?

LABORATORIYA ISHI №3

QO'SHIQLARNI JANRLAR BO'YICHA SINFLASHTIRISH

1. ISHDAN MAQSAD

- Dastur ishlab chiqish ta'minlanishi uchun instrumental vositalarni qo'llash ko'nikmalarini olish;
- Qo'shiqlarni janrlar bo'yicha sinflashtirish imkoniyati o'rganish.

2. ISHGA TAYYORGARLIK

1. Python da janrlar bo'yicha qo'shiqlarni sinflashtirish imkoniyatini o'rganish va laboratoriya tavsifini o'qib chiqish.
2. Laboratoriya ishidagi savollarga javob berish.
3. Mazkur laboratoriya ishini bajarish uchun laboratoriya ishi №2 uchun yaratilgan ma'lumotlardan foydalaning.

3. NAZARIY QISM

Akustik signal tuzilmasi va musiqali axborotni ajratib olish jarayonini o'ziga xosliklari bilan tanishib olgandan so'ng ovoz bilan ishlash uchun Python kutubxonalarini ko'rib chiqamiz.

Musiqani janrli klassifikatorini modelleshtirishga urinib ko'ramiz. Bu agar ko'plab noma'lum mp3 fayllarni tartiblash kerak bo'lganda asqotadi.

Dastlabki ishlov

Sinflashtirish modelini o'qitishdan oldin ovozli tanlanmalardan qayta ishlanmagan ma'lumotlarni ma'noliroq narsaga aylantirish kerak. Python SoX moduli yordamida ishlay olishi uchun kliplarni wav formatiga o'tkazamiz.

sox input.au output.wav

Sinflashtirish

Mohiyatni ajratib olish

Endi audiofayldan barcha kerakli axborotni ajratib olamiz:

- Mel-chastotali kepstral koeffitsient,
- Spektral sentroid,
- Nol orqali o'tish chastotasi,
- Rangbaranglik chastotasi,
- Chastotani spektral qulashi.

Bu barcha funksiyalarni .csv faylda saqlaymiz.

Qo'shiqlarni janr bo'yicha taqsimlash uchun mavjud bo'lgan sinflashtirish algoritmidan foydalanishimiz yoki bevosita spektogrammadan foydalanishimiz yoki mohiyatni ajratib, unga sinflashtirish modellarini qo'llashimiz mumkin.

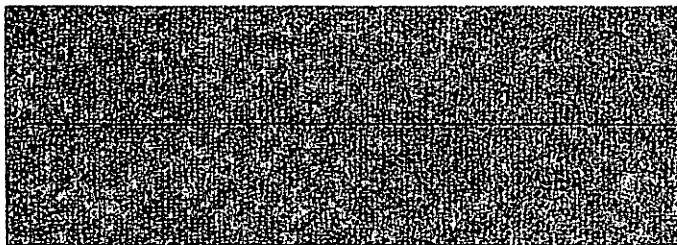
Janrli sinflashtirish – musiqali axborotni ajratib olishni ko'plab amaliy sohalarning biri xolos.

Biz musiqali signallar bilan Python da ishlash qanday tashkillashtirilishini bilib oldik. Bu bilimlarni ko'plab vazifalarni yechish uchun qo'llash mumkin: ritmni kuzatish, musiqa, tavsivaviy tizimni yaratish, instrumentlarni tanib olshi va shu kabilar.

Avvalambor savol beramiz: umuman kimga radioda reklamani tanib olish kerak o'zi? Bu o'zlarining reklama roliklarini real chiqishini kuzatish, kesilish va uzilish hollarini ushlash uchun reklama beruvchilarga foydali; radiostansiyalar hududlarda tarmoq reklamasini chiqishini monitoring qilishi mumkin. O'sha tanib olish vazifasi agar biz musiqa asarini o'ynalishini kuzatish yoki kichik fragmentdan qo'shigni tanib olishni istasak paydo bo'ladi,

Vazifa yanada qat'iyroq quyidagicha shakllanadi: bizda etalon audio-fragmentlarni (qo'shiq yoki reklama roligi) ba'zi to'plami mavjud va qaysidir fragmenti ijobiy ovoz beradigan eferning audio-yozuvi mavjud. Vazifa – eshitilgan barcha fragmentlarni topish, boshlanish momentini va o'ynatish davomiyligini aniqlash. Agar efir yozuvini tahlillasak, unda tizim real vaqtdan tezroq ishlashi kerak bo'ladi.

Hamma biladiki ovoz (tor ma'noda) – bu havoda tarqaladigan siqilgan va kesilgan to'liqlardir. Ovoz yozuvi, masalan wav fayl amplituda qiymatlarining ketma-ketligidir (fizik jihatdan siqilish darajasi yoki bosimiga teng). Agar siz audio-redaktorni ochsangiz, unda bu ma'lumotlarni vizualizatsiyasini ko'rgan bo'lsangiz kerak – amplitudani vaqtga bog'liqlik grafigi (fragment davomiyligi 0.025 soniya):



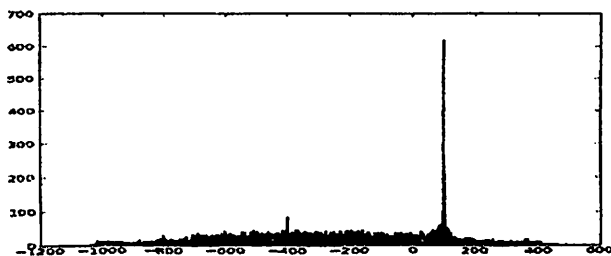
3.1-rasm. Havoda tarqaladigan siqilgan va kesilgan to'liqlar.

Biroq biz chastotani bu tebranishlarini bevosita ilg'amaymiz, turli chastota va tembrdagi ovozlarni eshitamiz. Shuning uchun odatda ovozni vizualizatsiyasini boshqa yo'lidan – gorizontali o'qqa vaqt, vertikal o'qqa chastota qo'yilgan, nuqta rangi esa amplitudani bildiradigan spektogrammadan foydalanamiz.

Efirda fragment qidiruvi vazifasini ikki qismga ajratish mumkin: avval etalon bo'lgan kp sonli nomzodlar orasidan fragmentlarni topish, keyin haqiqatdan ham nomzod joriy efirda ishtirok etishi va agar ishtirok etsa, ovoz qaerda boshlanishi va tugashi tekshiriladi. Ikkala operatsiya o'z ishi uchun ovoz fragmentini "izidan" foydalanadi. U shovqinga bardoshli va yetarlicha kompakt bo'ladi. Bu iz quyidagicha quriladi: spektogrammani qisqa vaqt bo'yicha qismlarga ajratamiz va har bir qismda maksimal amplitudali chastotalisini qidiramiz (aslida, turli diapazonlarda bir necha maksimumlarni qidirish ma'qul, biroq soddalik uchun eng hajmlisidan bitta maksimumni olamiz). Bunday chastotalar to'plami (yoki chastotalar indeksi) izni bildiradi. Qo'pol qilib aytganda, har bir vaqt momentida ovoz chiqaruvchi "notalardir".

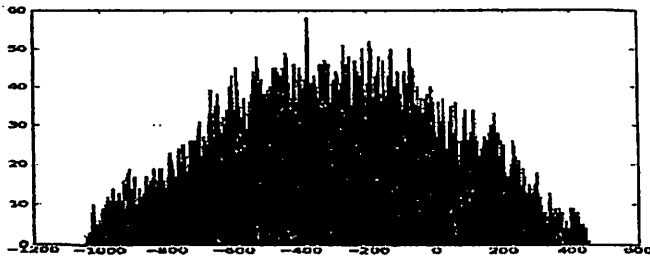
Ovoz fragmenti izini olish

Biz efir fragmenti izini va barcha etalon fragmentlarni olishimiz mumkin, faqatgina nomzodlarni tez qidirish va fragmentlarni taqqoslashni o'rganishimiz kerak. Avval taqqoslash vazifasini ko'ramiz. Shovqin va buzilish sababli izlar aniq mos kelmasligi mumkin. Biroq bunday noqulay chastotalar yetarlicha barcha buzilishlarni o'zidan o'tkazadi (chastotalar deyarli umuman "suzmaydi") va chastotani yetarlicha katta foizi aniq mos keladi – shunday qilib, ko'p moslikli ikki chastotalar ketma-ketliklari orasidan surilishlarni topish qoladi. Vizualashtirishni oddiy usuli – avval chastota bo'yicha mos kelgan barcha nuqtalar juftligini topish, keyin mos kelgan nuqtalar orasidan vaqt farqlarini gistogrammasini qurishdir. Agar ikkita fragment umumiy sohaga ega bo'lsa, unda gistogrammada yorqin ifodalangan uchni ko'rsatadi (uchni ko'rinishi mos keladigan fragment vaqt boshlanishi to'g'risidan darak beradi):



3.2-rasm. Vaqt bo'yicha farqlar gistogrammasi

Agar ikkita fragment hech qanday bog'likka ega bo'lmasa, unda hech qanday uch bo'lmaydi:



3.3-rasm. Fragmentlarda bog‘liqlik bo‘lmagandagi gistogramma

Nomzodlar qidiruvi muammosi odatda xeshlashdan foydalanish bilan yechiladi – fragment izi bo‘yicha ko‘p sonli xeshlar quriladi, bu ketma-ket yoki ma’lum masofa bilan keladigan izdan bir necha qiymatlardir. Neyron tarmoqlar bilan ishlash uchun Google Colab – GPU va TPU ni bepul bajarilish muhiti sifatida taqdim qiladigan bepul servisdur (Google Colab – bu Jupyter Notebook ni ishga tushurish imkonini beradigan servisdur, bunda agar `Runtime -> Change runtime type -> GPU` tanlansa, unda bitta GPU (Nvidia Tesla K80) ga ruxsat olinadi). Videokartadan sutkasiga 12 soat foydalanish mumkin, shu sababli qiyinroq masalalar uchun boshqa variantlarni ko‘rib chiqish mumkin.

Jupyter Notebook – bu python da interfaol hisoblash uchun komanda qobig‘idir.

Birinchi navbatda audio faylni PNG tasvir (spektrogramma) formatiga aylantirish lozim. Keyin ulardan ahamiyatli xarakteristikalarini ajratib olish lozim: MFCC, spektral sentroid, kesishish tezligi, rangbaranglik chastotasi, spektr qulashi.

Belgi ajratib olingach, ANN dan sinflashtirish uchun foydali sh mumkin bo‘lishi uchun CSV faylni qo‘shish mumkin.

1. Google Drive ga ma’lumotlarni ajratib olib yuklaymiz, keyin diskni Colabni ulaymiz.
2. Barcha kerakli kutubxonalarni import qilamiz.

```

import librosa
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import os
from PIL import Image
import pathlib
import csv
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
import keras
from keras import layers
from keras import layers
import keras
from keras.models import Sequential
import warnings
warnings.filterwarnings('ignore')

```

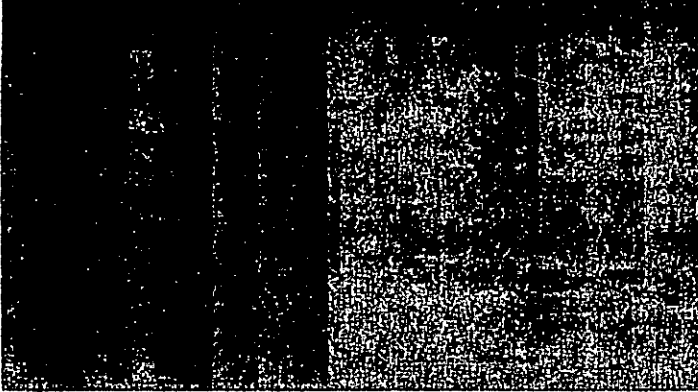
3. Audioma'lumotlar fayllarni PNG ga konvertlaymiz yoki har bir audio uchun spektogrammani ajratib olamiz.

```

cmap = plt.get_cmap('inferno')
plt.figure(figsize=(8,8))
genres = 'blues classical country disco hiphop jazz metal pop reggae
rock'.split()
for g in genres:
    pathlib.Path(f'img_data/{g}').mkdir(parents=True, exist_ok=True)
    for filename in os.listdir(f'./drive/My Drive/genres/{g}'):
        songname = f'./drive/My Drive/genres/{g}/{filename}'
        y, sr = librosa.load(songname, mono=True, duration=5)
        plt.specgram(y, NFFT=2048, Fs=2, Fc=0, noverlap=128,
            cmap=cmap, sides='default', mode='default', scale='dB');
        plt.axis('off');
        plt.savefig(f'img_data/{g}/{filename[:-3].replace(".", "")}.png')
    plt.clf()

```

Blyuz janridagi qo'shiq sempl spektogrammasi:



3.4-rasm. Qo'shiq semplici spektrogrammasi

Audiofayllarni mos spektrogrammalarga aylantirish funksiyalarni ajratib olishni soddalashtiradi.

CSV fayli uchun sarlavha yaratish

```
header = 'filename chroma_stft rmse spectral_centroid spectral_bandwidth
rolloff zero_crossing_rate'
for i in range(1, 21):
header += f' mfcc{i}'
header += ' label'
header = header.split()
```

4. Spektrogrammadan belgilarni ajratib olamiz: MFCC, spektral sentroid, nol kesishish chastotasi, ragbaranglik chastotasi va spektr qulashi.

```
file = open('dataset.csv', 'w', newline='')
with file:
writer = csv.writer(file)
writer.writerow(header)
genres = 'blues classical country disco hiphop jazz metal pop reggae rock'.split()
for g in genres:
for filename in os.listdir(f'/drive/My Drive/genres/{g}'):
songname = f'/drive/My Drive/genres/{g}/{filename}'
```

```

y, sr = librosa.load(songname, mono=True, duration=30)
rmse = librosa.feature.rmse(y=y)
chroma_sft = librosa.feature.chroma_sft(y=y, sr=sr)
spec_cent = librosa.feature.spectral_centroid(y=y, sr=sr)
spec_bw = librosa.feature.spectral_bandwidth(y=y, sr=sr)
rolloff = librosa.feature.spectral_rolloff(y=y, sr=sr)
zcr = librosa.feature.zero_crossing_rate(y)
mfcc = librosa.feature.mfcc(y=y, sr=sr)
to_append = f'{filename} {np.mean(chroma_sft)} {np.mean(rmse)}
{np.mean(spec_cent)} {np.mean(spec_bw)} {np.mean(rolloff)} {np.mean(zcr)}'
for e in mfcc:
    to_append += f' {np.mean(e)}'
to_append += f' {g}'
file = open('dataset.csv', 'a', newline='')
with file:

```

5. O'qitish va sinash uchun to'plamlarga ma'lumotlarni ajratish va belgilarni masshtablashtirish, metkalar yaratish, CSV ma'lumotlarini yuklashni o'z ichiga olgan ma'lumotlarni dastlabki qayta ishlashni amalga oshiramiz.

```

data = pd.read_csv('dataset.csv')
data.head()# Keraksiz ustunlarni o'chirish
data = data.drop(['filename'],axis=1)# Metkalar yaratish
genre_list = data.iloc[:, -1]
encoder = LabelEncoder()
y = encoder.fit_transform(genre_list)# Belglar ustunlarini masshtablashtirish
scaler = StandardScaler()
X = scaler.fit_transform(np.array(data.iloc[:, :-1], dtype = float))# Ma'lumotlarni
o'qitish va sinash to'plamlariga ajratish
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

```

6. ANN modelini yaratish

```
model = Sequential()
model.add(layers.Dense(256, activation='relu',
input_shape=(X_train.shape[1],)))
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10,
activation='softmax'))model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
```

7. Modelni ishga tushurish:

```
classifier = model.fit(X_train,
y_train,
epochs=100,
batch_size=128)
```

100 davrdan keyin aniqlik 0.67 ni tashkil etdi.

4. LABORATORIYA ISHIGA TOPSHIRIQ

1. Audiofayldan barcha kerakli axborotni ajratib olish:

- Mel-chastotali kepsstral koeffitsient,
- Spektral sentroid
- Nol orqali o'tish chastotasi
- Ragbaranglik chastotasi
- Spektral qulash chastotasi.

Bu barcha funksiyalarni csv-faylda saqlang.

2. Ovoz fragmentida izini oling.

3. O'qitish va sinash uchun to'plamlarga ma'lumotlarni ajratib va belgilarni masshtablashtirish, metkalar yaratish, CSV ma'lumotlarini yuklashni o'z ichiga olgan ma'lumotlarni dastlabki qayta ishlashni amalga oshiring.

4. ANN modelini yarating.

5. HISOBOTGA QO'YILADIGAN TALABLAR

Hisobot quyidagilardan iborat bo'lishi mumkin:

- Talaba familiyasi ko'rinishidagi nomga ega faylda saqlangan topshiriq bajarilish natijalari.

6. NAZORAT SAVOLLARI

1. Ovoz fragmenti izi qayday yo'l bilan olinadi?
2. Audiofayl nega PNG tasvir formatiga aylantiriladi?
3. Kerakli kutubxonalarni import qilish qanday amalga oshiriladi?
4. Audioma'lumotlarni PNG formatga konvertlash yoki har bir audio uchun spektrogramma ajratib olish jarayoni?
5. Spektrogrammadan belglarni ajratib olish: MFCC, spektral sentroid, nol bilan kesishish chastotasi, rangbaranglik chastotasi va spektr qulashi.
6. Ma'lumotlarni dastlabki qayta ishlash?

LABORATORIYA ISHI №4

PILLOW TASVIRLARI KUTUBXONALARI BILAN ISHLASH

1. ISHDAN MAQSAD

- tasvirlarni qayta ishlashda paydo bo'ladigan muammolarni yechish malakasini shakllantirish;
- PILLOW kutubxonasidan foydalanib, tasvir bilan ishlash imkoniyatlarini o'rganish.

2. ISHGA TAYYORGARLIK

1. Laboratoriya ishi tavsifini o'qib chiqish va PILLOW kutubxonasidan foydalanib tasvirlar bilan ishlash imkoniyatini o'rganish.
2. Laboratoriya ishida taqdim qilingan savollarga javob berish.

3. NAZARIY QISM

Python tasvirlar kutubxonasini yoki PIL (Python Imaging Library) Python dasturlash tilida grafikni qayta ishlash uchun kerak. Fredrik Lund Python ga bag'ishlangan eng yaxshi bloglarni muallifi hisoblanadi. Biroq u PIL oxirgi relizidan uncha uzoq vaqt o'tmasdan uzoq 2009 yildan buyon yangilanmay qo'ygan. Yaxshiyamki, Pillow nom ostida PIL forkini yaratib loyihani qo'llab-quvvatlovchi Python ishlab chiqaruvchilari topildi. Pillow PIL original kutubxonasini o'rni bosuvchi bo'ldi. Bundan tashqari, u PIL erisholmagan Python 3 ni ham qo'llab-quvvatlaydi.

PIL va Pillow bir vaqtning o'zida o'rnatilmasligiga e'tibor qarating.

Python da Pillow ni o'rnatish

Python da Pillow ni pip yoki easy_install orqali o'rnatish mumkin. Pip orqali o'rnatish quyidagicha amalga oshiriladi:

Shell

pip3 install pillow

Linux yoki Mac da ishlashda komandani sudo oraqli, ya'ni ma'mur nomidan ishga tushurish talab etilishi mumkin.

Pyhton da tasvirlarni Pillow bilan ochish

Pillow orqali tasvirlarni oson ochish mumkin va uni tashqi dastur orqali ekranda akslantirish mumkin. Misol:

Python

```
from PIL import Image
```

```
image = Image.open('jelly.jpg')
```

```
image.show()
```

show() usuli asosan to'g'rilash uchun foydalaniladi. Misolda Image moduli import qilinadi va ko'rsatilgan tasvir ochiladi.

Windows da tasvir BMP vaqtinchalik faylda saqlanadi va Paint ga o'xshash oddiy dastur orqali ochiladi.

Pillow orqali tasvir to'g'risida axborot olish

Pillow yordamida ham tasvir to'g'risida batafsil axborotni olish mumkin.

Kichik bir misol:

```
>>> from PIL import Image
```

```
>>> image = Image.open('jelly.jpg')
```

```
>>> r, g, b = image.split()
```

```
>>> histogram = image.histogram()
```

```
[384761, 489777, 557209, 405004, 220701, 154786, 55807, 35806, 21901,
```

```
16242]
```

```
>>> exif = image.getexif()
```

```
Exif
```

Bu misolda RGB (red, green, blue) tasvirlarni qanday ajratib olish ko'rsatilgan. Yana tasvir gistogrammasini olish ko'rsatilgan. Boshqa Python paketidan foydalanib, gistogramma grafik qurish mumkin - matplotlib.

Yuqorida keltirilgan misol tasvirdan EXIF axborotini ajratib olish ko'rsatilgan. Bu yerda usul, hozirgi momentda ahamiyatga ega emas ko'plab axborotlar bo'lganligi sababli bir qancha qisqargan.

Pillow (crop) orqali tasvirlarni qirqish

Pillow tasvirlarni qirqish uchun ham foydalaniladi. Bu yetarlicha oson jarayon bo'lsada urinish va xatoliklar orqali erishiladi. Image.crop() orqali rasmni qirqishga urinib ko'ramiz:

```
from PIL import Image
```

```
image = Image.open('jelly.jpg')
```

```
cropped = image.crop((0, 80, 200, 400))
```

```
cropped.save('/path/to/photos/cropped_jelly.png')
```

E'tibor qarating, bu yerda faqat tavsirni ochish, keyin esa crop() usulini chaqirish lozim. x/y koordinatalarni nimani qirqish kerakligini ko'rsatish talab etiladi, masalan (x1, y1, x2, y2).

Pillow da 0 pikseli yuqori chap sath hisoblanadi. X qiymatini oshishi bilan o'ngga surilish boshlanadi. y qiymatini oshishi bilan pastga surilish boshlanadi.

To'g'ri koordinatalarni olish uchun Gimp yoki Photoshop dan foydalanish mumkin. Bular quyidagi qarqim uchun koordinatalarni aniqlashga yordam beradi.

```
from PIL import Image
```

```
image = Image.open('jelly.jpg')
```

```
cropped = image.crop((177, 882, 1179, 1707))
```

```
cropped.save('/path/to/photos/cropped_jelly2.png')
```

Dastur rasmni qirqadi, keyin esa yangi ko'rinish diskda saqlanadi. crop() usuli piksellar koordinatalarini bildiradigan to'rtta elementni kortejini qabul qiladi (yuqori chap, yuqori o'ng, pastki chap, pastki o'ng).

Tasvirlarni burish – Pillow ni rotate() usuli
Image.rotate() tasvirni burilgan nusahasini qaytaradi.

```
from PIL import Image
import sys
try:
    tattras = Image.open("tattras.jpg")
except IOError:
    print("Unable to load image")
    sys.exit(1)
    rotated = tattras.rotate(180)
    rotated.save('tattras_rotated.jpg')
```

Mazkur dastur tasvirlarni 180 darajaga buradi va tavsimi yangi ko‘rinishda diskda saqlaydi.

Tkinter Python da rasmni akslantirish

Quyidagi kod Tkinter dasturida rasmni akslantirish uchun kerak.

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
from PIL import Image, ImageTk
from tkinter import Tk
from tkinter.ttk import Frame, Label
import sys
class Example(Frame):
    def __init__(self):
        super().__init__()
        self.loadImage()
        self.initUI()
    def loadImage(self):
        try:
            self.img = Image.open("tattras.jpg")
```

```

except IOError:
    print("Unable to load image")
    sys.exit(1)
def initUI(self):
    self.master.title("Label")
    tatra = ImageTk.PhotoImage(self.img)
    label = Label(self, image=tatra)
    # reference must be stored
    label.image = tatra
    label.pack()
    self.pack()
def setGeometry(self):
    w, h = self.img.size
    self.master.geometry(("0x%d+300+300") % (w, h))

def main():
    root = Tk()
    ex = Example()
    ex.setGeometry()
    root.mainloop()
if __name__ == '__main__':
    main()

```

Dastur Tkinter dan label tulkit vidjetini rasmini ko'rsatadi.

```
from PIL import Image, ImageTk
```

ImageTk Tkinter bilan birgalikdagi rasm hisoblanadi. Tkinter rasm obyektini olishni kutayotgan hamma joyda foydalanilishi mumkin.

```
tatra = ImageTk.PhotoImage(self.img)
```

Bu yerda rasmni hosil qilamiz.

```
label = Label(self, image=tatra)
```

Rasm label vidjeti image parametriga beriladi.

label.image = tatra

Axlat to'planib qolishdan qochish maqsadida rasmga havolalar saqlanishi lozim.

w, h = self.img.size

self.master.geometry(("dx%d+300+300") % (w, h))

Oyna hajmi rasm hajmi bilan mos tushadi.

URL dan Pillow ga tavsirni yuklash

Quyidagi misolda URL ko'rsatilgan holda tasvirlarni olish ko'rsatilgan.

from PIL import Image

import requests

import sys

url = 'https://l.vtimg.com/vi/vEYsdh6uiS4/maxresdefault.jpg'

try:

resp = requests.get(url, stream=True).raw

except requests.exceptions.RequestException as e:

sys.exit(1)

try:

img = Image.open(resp)

except IOError:

print("Unable to open image")

sys.exit(1)

img.save('sid.jpg', 'jpeg')

Kod tasvirlarni URL orqali o'qiydi va uni diskda saqlaydi.

import requests

Tasvirlarni yuklash uchun requests kutubxonasidan foydalanamiz.

resp = requests.get(url, stream=True).raw

Tasvir raw ma'lumotlari kabi o'qiydi.

img = Image.open(resp)

Rasm response javob obyektidan yaratiladi.

img.save('sid.jpg', 'jpeg')

Yakunda rasm saqlanadi.

Pillow da rasm yaratish

Pillow da 2D grafikni yaratish uchun bazaviy imkoniyatlar mavjud. *ImageDraw* moduli Image obyektlar uchun oddiy 2D grafikni hosil qiladi. Biz yangi rasm, ularga annotatsiya yaratishimiz, mavjud ftoni retushalashimiz, hamda birdaniga veb uchun grafikni generatsiyalashimiz mumkin.

```
from PIL import Image, ImageDraw
# Sozdaem belsy kvadrat
img = Image.new('RGBA', (200, 200), 'white')
idraw = ImageDraw.Draw(img)
idraw.rectangle((10, 10, 100, 100), fill='blue')
img.save('rectangle.png')
```

Misolda oq fonda ko'k to'g'ri to'rtburchak chizilgan yangi rasm yaratiladi.

```
img = Image.new('RGBA', (200, 200), 'white')
```

«RGBA» rejimli 200x200 hajmli va oq fondagi yangi rasm yaratiladi.

```
idraw = ImageDraw.Draw(img)
```

Rasmdan *ImageDraw* obyekti yaratiladi. Endi unda nimadir chizishimiz mumkin.

```
idraw.rectangle((10, 10, 100, 100), fill='blue')
```

rectangle() usuli yordamida yaratilgan rasm maydonida ko'k to'g'ri to'rtburchak chizamiz.

ImageFont – Pillow dan foydalangan holda tasvirda matn yozamiz

Keyingi kodda Pillow yordamida rasmda qanday qilib Python da matn yozishimiz mumkinligi ko'rsatilgan.

Python yordamida tasvirga suv bulgisini qo'yish.

```
from PIL import Image, ImageDraw, ImageFont
import sys
try:
    tatra = Image.open("tatra.jpg")
except:
    print("Unable to load image")
    sys.exit(1)
idraw = ImageDraw.Draw(tatra)
```



```
text = "High Tatras"  
font = ImageFont.truetype("arial.ttf", size=18)  
idraw.text((10, 10), text, font=font)  
tatras.save('tatras_watermarked.png')
```

Rasm yaratish uchun ImageDraw moduli foydalaniladi.

```
font = ImageFont.truetype("arial.ttf", size=18)
```

18 hajmli Arial shrift yaratiladi.

```
idraw.text((10, 10), text, font=font)
```

Matni o'zi *text()* usuli orqali kiritiladi. Sukut bo'yicha shrift rangi oq.

4. LABORATORIYA ISHIGA TOPSHIRIQ

Python da Pillow o'rnatish. Pillow bilan Python da ochiq tasvir hosil qiling. Tasvir to'g'risida axborot oling. Kerakli qirqimni amalga oshiring. Tasvirlarni zahira nushasini oling. Tkinter Python da rasm aksini yarating. Pillow da URL rasmni yuklang. Pillow da rasm yarating, rasmda matn yozing.

5. HISOBOTGA QO'YILADIGAN TALABLAR

Hisobot quyidagilardan iborat bo'lishi lozim:

- Talab familiyasi ko'rinishidagi nomli faylda saqlangan topshiriqni bajarilish natijalari.

6. NAZORAT UCHUN SAVOLLAR

1. Pillow bilan Python da tasvirni ochish
2. Pillow orqali rasm to'g'risida axborot olish
3. Pillow(crop) orqali tasvirni qirqish
4. Pillow ni rotate() usulida tasvirni burish
5. Tkinter Python da rasmni akslantirish
6. Pillow da URL dan rasmni yuklash
7. Pillow da rasmni yaratish
8. ImageFont – Pillow dan foydalanib, rasmda matn yozamiz.

LABORATORIYA ISHI №5
TASVIRLARNI QAYTA ISHLASH UCHUN FILTRLARDAN
FOYDALANISH

1. ISHDAN MAQSAD

- Tasvirlarni qayta ishlash kompyuter tizimi yaratish bo'yicha tasavvur hosil qilish;

- Filter() usulini asosiy modullari bilan tanishish.

2. ISHGA TAYYORGARLIK

1. Laboratoriya ishi tavsifini o'qib chiqish va filtrlar bilan ishlash imkoniyatini o'rganish.

2. Laboratoriya ishida ko'rsatilgan savollarga javob berish.

3. NAZARIY QISM

Pillow tasvirlarni qayta ishlash uchun turli filtrlar to'plamidan foydalanish imkonini beradi. Ular ImageFilter moduli qismi hisoblanadi. filter() usulidan foydalanishga bir nechta misollar ko'rib chiqamiz:

```
from PIL import ImageFilter  
from PIL import Image  
image = Image.open('jelly.jpg')  
blurred_jelly = image.filter(ImageFilter.BLUR)  
blurred_jelly.save('/path/to/photos/blurry_jelly.png')
```

Dastur ma'lum rasmni oladi, u asosida *ImageFilter.BLUR* foydalanib xira rasmni yaratadi va *save()* usuli yordamida olingan natijani diskka saqlaydi.

Xira rasm

Biroq ko'p hollarda rasmni xiralashtirishga hojat yo'q, aksincha – aniqlikni oshirish talab etiladi. Pillow rasm aniqligini quyidagi tarzda o'zgartiradi:

```
from PIL import ImageFilter  
from PIL import Image  
image = Image.open('/path/to/photos/jelly.jpg')  
blurred_jelly = image.filter(ImageFilter.SHARPEN)  
blurred_jelly.save('/path/to/photos/sharper_jelly.png')
```

Bundan tashqari tasvir aniqligini oshirish uchun Python da **ImageEnhance** modulidan foydalanish mumkin.

Boshqa filtrlardan foydalanish ham mumkin - **DETAIL**, **EDGE_ENHANCE**, **EMBOSS**, **SMOOTH** va shu kabilar. Kodda bitta tasvir uchun bir vaqtning o'zida bir nechta filtrdan foydalanish mumkin.

JPG dan PNG ga konvertatsiyalash

Python Pillow da *save()* usuli rasmni boshqa formatga konvertatsiyalash imkonini beradi.

```
from PIL import Image  
import sys  
try:  
    tatras = Image.open("tatras.jpg")  
except IOError:  
    print("Unable to load image")  
    sys.exit(1)  
tatras.save('tatras.png', 'png')
```

Dastur JPG tasvirlarni hisoblaydi va PNG formatiga konvertatsiyalaydi. Bu quyidagicha amalga oshiriladi:

```
tatras.save('tatras.png', 'png')
```

save() usulini ikkinchi parametri rasmni natijaviy formatini aniqlashtirish uchun kerak.

Pillow da GrayScale oq-qora tasvirini yaratish

Image.convert() usuli yordamida original oq-qora rasm yasash mumkin.

```
from PIL import Image  
import sys
```

```
try:  
    tatras = Image.open("tatras.jpg")  
except IOError:
```

```

print("Unable to load image")
sys.exit(1)
grayscale = tattras.convert('L')
grayscale.show()

```

Dastur rasmni o'qiydi va uni oq-qoraga transformatsiyalaydi. Bunga quyidagi qator javob beradi:

```
grayscale = tattras.convert('L')
```

convert() usulini birinchi parametri mod hisoblanadi. "L" modi oq-qora variantni taqdim qiladi.

Pillow da rasm hajmini o'zgartirish - resize()

resize() usuli yordamida rasm uzunligi va kengligini o'zgartirish mumkin.

Bu misolda o'lchamni o'zgartirishni uchta misoli ko'rsatiladi:

- Rasmni kengligi va balandligini bilgan holda o'zgartirish;
- Kenglikni balandlik uchun proporsiyani hisobga olib o'zgartirish;
- Balandlikni kenglikka proporsional o'zgartirish.

Rasmni kengligi va balandligini bilgan holda o'zgartirish

```
from PIL import Image
```

```
# Rasm o'lchamini yangisiga o'zgartirish.
```

```
tattras = Image.open("tattras.jpg")
```

```
tattras = tattras.resize((100, 100), Image.ANTIALIAS)
```

Kenglikni rasmni yangi balandligi uchun proporsiyalarni hisobga olib o'zgartirish

```
from PIL import Image
```

```
tattras = Image.open("tattras.jpg")
```

```
width, height = tattras.size
```

```
new_width = 680 # shirina
```

```
new_height = int(new_width * height / width)
```

```
tattras = tattras.resize((new_width, new_height),
```

Image.ANTIALIAS)

tatras.show()

Rasm balandligini yangilanayotgan kenglikka proporsional o'zgartirish

from PIL import Image

tatras = Image.open("tatras.jpg")

width, height = tatras.size

new_height = 680 # Visota

*new_width = int(new_height * width / height)*

tatras = tatras.resize((new_width, new_height),

Image.ANTIALIAS)

tatras.show()

4. LABORATORIYA ISHIGA TOPSHIRIQ

1. Tanlab olingan rasm bilan quyidagi amallarni bajaring:

- xira rasm hosil qiling;
- rasm aniqligini oshiring;
- rasmni JPG dan PNG formatiga konvertatsiyalang;
- Pillow da GrayScale oq-qora rasm yarating;
- Pillow da rasm o'lchamini o'zgartiring – resize().

5. HISOBOTGA QO'YILADIGAN TALABLAR

Hisobot quyidagilardan iborat bo'lishi lozim:

- Talab familiyasi ko'rinishidagi nomli faylda saqlangan topshiriqni bajarilish natijalari.

6. NAZORAT UCHUN SAVOLLAR

1. Python da Pillow tasvirini ochish.
2. Pillow orqali tasvir to'g'risida axborot olish
3. Pillow (crop) orqali tasvirlarni qirqish.
4. Pillow ni rotate() usulida tasvirlarni buring.
5. Tkinter Python da tasvirlarni akslantirish
6. Tasvirlarni URL dan Pillow ga yuklang.
7. Pillow da rasm yarating.
8. ImageFont – Pillow dan foydalanib, rasmga matn yozing.

LABORATORIYA ISHI №6

OPENCV (OPEN SOURCE COMPUTER VISION) IMKONIYATLARIDAN FOYDALANISH

1. ISHDAN MAQSAD

-tasvirni qayta ishlash usullari asosida dasturiy ta'minot ishlab chiqish malakasini olish

-Kompyuter ko'rishi va mashinali o'qitish vazifalari uchun kutubxonalardan foydalanishni o'rganish.

2. ISHGA TAYYORGARLIK

1. Laboratoriya ishi tavsifini o'qing hamda kompyuter ko'rishi va mashinali o'qitish vazifalari uchun kutubxonalar imkoniyatlarini o'rganish.

2. Laboratoriya ishidagi savollarga javob berish

3. NAZARIY QISM

OpenCV (Open Source Computer Vision) – kompyuter ko'rishi va mashinali o'qitish vazifasi uchun keng qo'llaniladigan kutubxonadir. OpenCV original realizatsiyasi C++ da joylashgan. Python C/C++ ga qaraganda sekinroq, biroq boshqa tomondan uni bu tillar yordamida oson kengaytirish mumkin. Bu o'ziga xoslik C++ da hisoblashda intensiv, Python modullari sifatida foydalanish uchun Python da o'ramalarda kodlarni yaratish imkonini berdi. OpenCV – Python bu OpenCV ni Python dagi realizatsiyasidir. Kutubxona kross-platformali va ochiq kod sifatida foydalanish mumkin.

Bundan tashqari OpenCV ga NumPy hamda Matplotlib kabi ba'zi yordamchilar kerak, ularni quyidagilardan foydalanib o'rning:

```
python -m pip install --user numpy scipy matplotlib ipython jupyter pandas  
sympy nose
```

```
>>> import cv2
```

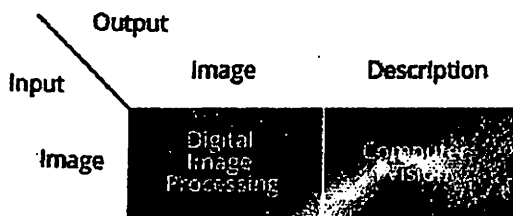
```
>>> import numpy
```

```
>>> import matplotlib
```

Agar import qilishda xatolik bo'lmasa, siz ishga tayyorsiz.

Tasvirlarni qayta ishlash yoki kompyuter ko'rishi nima va nima uchun kerak?

Tasvirlarni qayta ishlash (odatda “Tasvirlarni raqamli qayta ishlash” deb nomlanadi) va Computer Vision – bu qandaydir axborotni olish maqsadida tasvir yoki video bilan ishlaydigan algoritmlar to‘plamiga ega kompyuter ilmining sohasidir. Tasvirlarni qayta ishlash tasvirlarni tasvirga aylantirish faoliyati bo‘lgan bir paytda, ya’ni qayta ishlovni kirish va chiqishi tasvir hisoblanadi, kompyuter ko‘rishi kompyuterni tushunishga yoki axborotni olishga yoki raqamli tasvir yoki videodan yuqori darajali axborot olishga majbur qilishga tegishli fanlararo sohadir. Quyidagi xarita yuqorida ta’kidlanganlarni vizual tushuntirilishini beradi.



6.1-rasm. Tasvirlarni qayta ishlash va kompyuter ko‘rishini vizual namoyishi

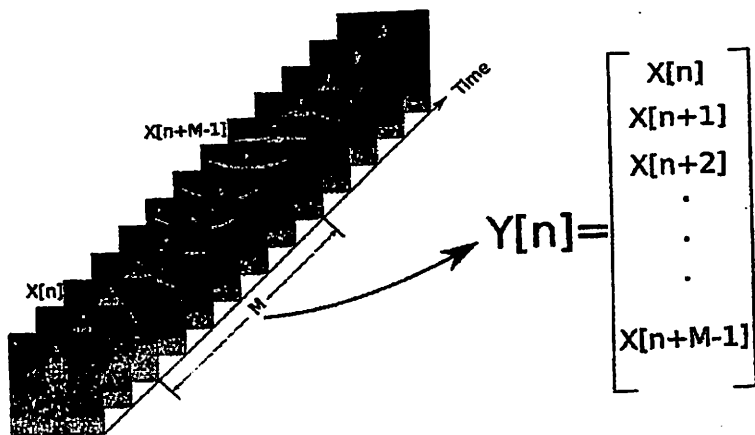
Tasvirlarni qayta ishlash algoritmi kiruvchi ma’lumot sifatida tasvir yoki videoni qabul qiladi, ularni qayta ishlaydi va qayta ishlash natijasida tasvir yoki video qoladi. Biroq kompyuter ko‘rishi algoritmi tasvir yoki videoni olib, ularni qayta ishlaydi va undan yaqqol hamda tarkiblii tavsif yaratadi.

Tasvir nima?

Tasvir ikki o‘lchamli $f(x, y)$ funksiya kabi ifodalanishi mumkin, bu yerda X ham Y ham muhit (tekislik) koordinatalari va istalgan koordinata juftligini amplitudasi (x,y) intensivlik yoki bu nuqtadagi kulrang tasvir darajasi deb nomlanadi. (x,y) va f amplituda yakuniy diskret qiymat hisoblansa, tasvirlarni raqamli tasvir deb ataymiz.

Video nima?

Videoni asosiy ta’rifi – bu vaqt o‘qi bo‘ylab joylashtirilgan tasvirlardir. Video tomonlar nisbati, kadrlar chastotasi, qator bo‘ylab yoki progressiv kengaytirish, rang modeli, siqish usuli va shu kabilar bilan xarakterlanishi mumkin.



6.2-rasm. Video xarakteristikasi

Video M fragmentini hisobga olib, $X[n]$ birinchi kadr, $X[n+M-1]$ oxirgi kadr, $Y[n]$ fragmentni to'liq videoga shakllantirish uchun ularni barchasini tartiblovchi vektor hisoblanadi.

Tasvirlarni qayta ishlash va kirish/chiqish

OpenCV kirish/chiqish uchun tasvir va videoni ko'plab formatlarini qo'llab-quvvatlaydi. Bugungi kamera yozuvi (video) kadrlar tasvir hisoblangan soniyasiga 30-60 kadr tezlikda tasvirlanadigan kadrlarga olib kelinadi. Shunday qilib, tasvir qayta ishlovi va video tahlili katta qismi bir xil usullardan foydalanamiz.

cv2.imread() uchun quyidagi formatlar qo'llab-quvvatlanadi:

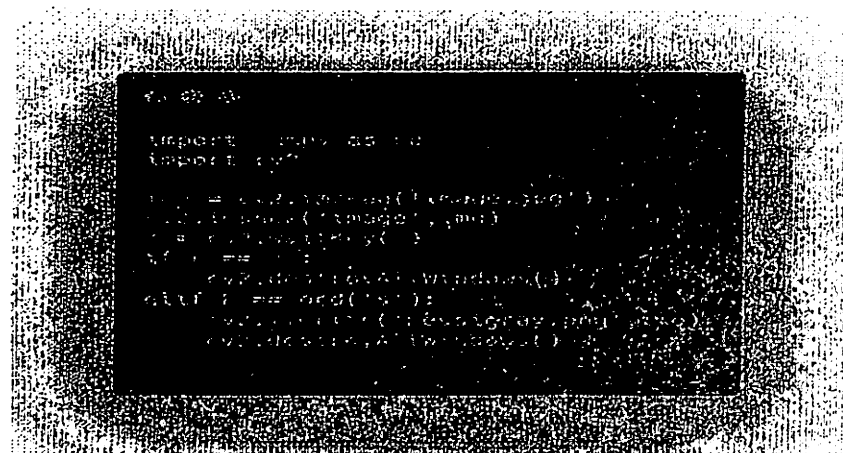
- Windows rastri tasviri - * .bmp, * .dib
- JPEG fayllar - * .jpeg, * .jpg, * .jpe
- JPEG 2000 fayllar - * .jp2
- Portativ tarmoq grafikasi - * .png
- Portativ tasvir formatlari - * .pbm, * .pgm, * .ppm
- Quyosh rasrlari - * .sr, * .ras
- TIFF fayllari - * .tiff, * .tif

cv2.VideoCapture() uchun AVI fayli - * .avi format to'liq qo'llab-quvvatlanadi, chunki AVI krossplatformali qo'llab-quvvatlanadigan yagona yaxshi format hisoblanadi.

Quyida OpenCV da tasvir va video kirish/chiqishiga ba'zi misollar keltirilgan:



6.3-rasm. Kirish uchun imread() da foydalanish



6.4-rasm. Tugma bosilishi bilan akslanayotgan tasvirlarni yopish

```

import cv2

cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)

while (cap.isOpened()):
    frame = cap.read()
    if frame == True:
        cv2.imshow('frame', frame)
    else:
        break
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()

```

6.5-rasm. Mavjud videoni yuklash va ishga tushurish

```

import cv2
import sys

url = 'cv2.VideoCapture(0)'
cap = cv2.VideoCapture(url, cv2.CAP_DSHOW)
ret, frame = cap.read()

while True:
    cv2.imshow('frame', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()

```

6.6-rasm. Veb-kasetadan video yozuvi va “q” tugmasini bosishdan keyin diskda saqlash

```

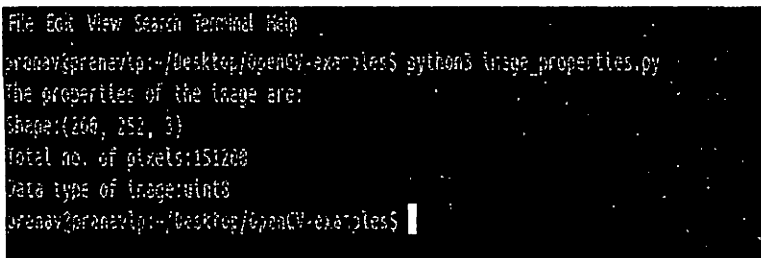
>>> import cv2
>>> img = cv2.imread('test.jpg')
>>> print(type(img))
<class 'numpy.ndarray'>
>>>

```

Tasvir toifasi Numpy's –"ndarray" (N – o'lchamli massiv) ekanligini ko'rish mumkin, tasvirlar ba'zi manipulyatsiyalarini ko'ramiz (massivlar manipulyatsiyasi), ba'zilarini esa OpenCV modulisiz ko'rib chiqamiz.

1. Tasvirlarni asosiy xususiyatlarini olamiz

```
import cv2print("Shape:" + str(img.shape))  
print("Total no. of pixels:" + str(img.size))  
print("Data type of image:" + str(img.dtype))
```



```
File Edit View Search Terminal Help  
pranav@pranavip:~/Desktop/openCV-examples$ python3 imgae_properties.py  
The properties of the image are:  
Shape:(200, 252, 3)  
Total no. of pixels:151200  
Data type of image:uint8  
pranav@pranavip:~/Desktop/openCV-examples$
```

6.7-rasm. Yuqoridagi kod natijasi

2. Tasvir pikselarini ruxsati va o'zgarishi

Alohida pikselga uning kulrang/intensivlik darajasi bo'yicha ruxsati:

```
import cv2  
import numpy as npm = cv2.imread("test1.jpeg")height,width,depth =  
np.shape(m)y = 1 # y coordinate(across height)  
x = 1 # x coordinate(across width)print("Value at (1, 1, 0) = " + str(m[y][x][0]))  
# This will print the pixel value at given coordinates at depth  
zero(blue)print("Value at (1, 1, 1) = " + str(m[y][x][1]))  
# This will print the pixel value at given coordinates at depth  
one(green)print("Value at (1, 1, 2) = " + str(m[y][x][2]))  
# This will print the pixel value at given coordinates at depth two(red)
```

```

File Edit View Search Terminal Help
pranav@pranavlp:~$ cd Desktop/OpenCV-examples/
pranav@pranavlp:~/Desktop/OpenCV-examples$ python3 accesspixels.py
Value at (1, 1, 0) = 241
Value at (1, 1, 1) = 235
Value at (1, 1, 2) = 233
pranav@pranavlp:~/Desktop/OpenCV-examples$

```

6.8-rasm. Yuqoridagi kod natijasi

Shunchaki tasvirlarni barcha piksellarini olish uchun quyidagi koddan foydalanish mumkin:

```

import cv2
import numpy as npm = cv2.imread("test1.jpeg")height, width, depth=
np.shape(m)# iterate over the entire image.
for y in range(0, height):
    for x in range(0, width):
        print(m[y][x])

```

Bu terminalingizad ko'p raqamlar chiqaradi!
Piksel qiymatini o'zgartirish uchun:

```

import cv2
import numpy as npm = cv2.imread("test1.jpeg")height, width, depth =
np.shape(m)for py in range(0, height):
    for px in range(0, width):
        m[py][px][0] = 0cv2.imshow('matrix', m)
cv2.imwrite('output2.png', m)
cv2.waitKey(0)
cv2.destroyAllWindows()

```



6.9-rasm. Birinchi tasvir – joriy. Boshqalari ma'lum kanallarning barcha piksellarini 0 ga teng qilib manipulyatsiyalandi.

3. Tasvir kanallarini ajratish

RGB tasvir 24 bitli rang chuqurligi to'g'risidagi ma'lumotlarga, ya'ni RGB ma'lumotlar - uchta 8 razryadli kanallarga ega. Bu kanallar intensivlik darajasi 0 dan 255 gacha bo'lgan ko'k, yashil va qizil ranglardir. OpenCV dagi tasvirlarni

`cv2.split()` dan foydalanib ajratish mumkin, biroq bu usul katta hisoblash sarfini talab etadi, shu sababli Numpy indekslashdan foydalanishimiz lozim, chunki u samaraliroq va mumkin bo'lsa foydalanilishi kerak.

`cv2.split()` usuli yordamida ajratish juda sodda:

```
b,g,r = cv2.split(img)
```

OpenCV da kanallar talabi BGR RGB emas.

Indeksatsiyadan foydalanib ajratish:

```
import cv2
```

```
import numpy as npm = cv2.imread("test1.jpeg")blue = m[:, :, 0]
```

```
green = m[:, :, 1]
```

```
red = m[:, :, 2]
```

4. LABORATORIYA ISHIGA TOPSHIRIQ: OpenCV dan foydalanib, tasvirlarni qayta ishlash imkoniyatlarini o'rganish.

5. HISOBOTGA QO'YILADIGAN TALABLAR

Hisobot quyidagilardan iborat bo'lishi lozim:

- Talab familiyasi ko'rinishidagi nomli faylda saqlangan topshiriqni bajarilish natijalari.

6. NAZORAT UCHUN SAVOLLAR

1. Tasvirlarni qayta ishlash va kiritish/chiqarish.
2. tugma bosilishi bilan akslanayotgan tasvirlarni yopish
3. Mavjud videoni yuklash va ishga tushurish
4. Veb-kameradan videoni yozish va "q" tugmasini bosish bilan disskka saqlash.
5. Tasvirni asosiy xususiyatlarini olish
6. Tasvir piksellerini o'zgartirish va ruxsat
7. Tasvir kanallarini ajratish

ADABIYOTLAR RO'YHATI

1. Trevor Cox, The Sound Book: The Science of the Sonic Wonders of the World, ООО «Издательская Группа «Азбука-Аттикус», 2018.
2. Roschesho Davide. Sound processing, 2012, 244
3. Burger W., Burge M. J. Digital Image Processing: An Algorithmic Introduction using Java. –New York: Springer, 2008. –564 p.
4. Gonzales R., Vude R. цифровая обработка изображений. Издание 3-е, испр. и доп. – М.: Техносфера, 2012. – 1104 с.
5. Гашников М. В., Глумов Н. И., Илясова Н. Ю. и др. Методы компьютерной обработки изображений. Изд. 2. Под ред. В.А. Сойфера. - М.: Физматлит, 2003. -784 с.

QO'SHIMCHA ADABIYOTLAR

1. Указ Президента Республики Узбекистан от 7 февраля 2018 года №УП-4947 «О стратегии действий по дальнейшему развитию Республики Узбекистан».
2. Mirziyoyev Sh.M. Azmu shijoatli xalqimiz tariximizning yangi sahifasini yaratishga qodir. O'zbekiston Respublikasi Prezidenti Shavkat Mirziyoyevning oliy majlisga murojaatnomasi. // Xalq so'zi gazetasi. 2020 yil 25 yanvar, №19.
3. Абламейко С.В., Лагуновский Д.М. Обработка изображений: технология, методы, применение. - Мн.: Амалфея, 2000.
4. Яне Б. цифровая обработка изображений. - М.: Техносфера, 2007. – 584 с.
5. Грузман И.С., Киришук В.С., Косых В.П. и др. цифровая обработка изображений в информационных системах. – Новосибирск: НГТУ, 2002. – 352 с.
6. Визилтер Йу.В. Обработка и анализ изображений в задачах машинного зрения. – М.: Физмат книга, 2010. – 672 с.
7. Pratt W.K. Digital Image Processing. – New York: Wiley, 2001. – 792 p
8. Jahne B. Digital Image Processing. - New York: Springer, 2005. – 585 p.

9. Chi Zh., Yan H., Pham T. Fuzzy algorithms : with applications to image processing and pattern recognition. New Jersey: World Scientific, 1996. – 230 p.

10. Gonzalez R., Woods R. Digital Image Processing. 3rd Edition - New Jersey: Prentice-Hall, 2008. – 954 p

11. Загуменнов А. Компьютерная обработка звука. – М.: Радио и связь, 2008. – 496 с.

12. Olli Niemitalo. Digital sound processing tutorial for the braindead-2012,-562 r.

13. Кинцел Т. Руководство программиста по работе со звуком. Пер. с англ. – М.: ДМК Пресс, 2000. – 432 с.

14. Лоянич А.А. Записи и обработка звука на компьютере–М.: Горячая линия , 2014.- 462 с.

15. Секунов Н.Ю. Обработка звука на ПС. – СПб.: БХВ-Петербург, 2001. – 1238 с.

16. Furu S. Digital Speech Processing, Synthesis and Recognition. New York: Marcel Dekker, 2001. - 452 p.

17. Гашников М. В., Глумов Н. И., Илясова Н. Ю. и др. Методы компьютерной обработки изображений. Изд. 2. Под ред. В.А. Сойфера. - М.: Физматлит, 2003. -784 с.

Axborot manbalari

1. Интеллектуальный анализ данных.

<http://www.seas.harvard.edu/courses/cs283/>

MUNDARIJA

1-Laboratoriya ishi. PYTHON DASTURLASH TILIDA OVOZNI QAYTA ISHLASH	3
2-Laboratoriya ishi. PYTHON DASTURLASH TILIDA SPEKTOGRAMMANI QAYTA ISHLASH.....	8
3-Laboratoriya ishi. QO'SHIQLARNI JANRLAR BO'YICHA SINFLASHTIRISH.....	18
4-Laboratoriya ishi. PILLOW TASVIRLARI KUTUBXONALARI BILAN ISHLASH.....	28
5-Laboratoriya ishi. TASVIRNI QAYTA ISHLASH UCHUN FILTRLARDAN FOYDALANISH.....	36
6-Laboratoriya ishi. OPENCV (OPEN SOURCE COMPUTER VISION)IMKONIYATLARIDAN FOYDALANISH.....	41
Foydalanilgan adabiyotlar.....	49

2800

**“OVOZ VA TASVIRNI QAYTA ISHLASH” FANIDAN LABORATORIYA
ISHLARINI BAJARISH UCHUN USLUBIY QO‘LLANMA**

5350200 - Televizion texnologiyalar (Audiovizual
texnologiyalar) mutaxassisligi talabalari
uchun uslubiy qo‘llanma

AVT kafedrasining 2021 yil 30 noyabr (14-sonli
bayonnoma) majlisida ko‘rib chiqildi va chop etishga
tavsiyalandi

TT fakultetining ilmiy-uslubiy Kengashida ko‘rib
chiqildi va chop etishga tavsiyalandi
2021 yil 28-dekabr 4-sonli bayonnoma

TATU ilmiy-uslubiy Kengashida ko‘rib
chiqildi va chop etishga tavsiyalandi
2022 yil 02. 23. 7(153)-sonli bayonnoma

Mualliflar: Sh.T.Kasimova
B.Boymurodov
Sh. Chulliyev

Taqrizchilar: N.Mirzayev
S.Beknazarova

Mas’ul muharrir: A.Muxamadiyev
Musahhah: N.X.Raximova

Bichimi 60x84 1/16. Bosma tabog’i 3,25
Adadi 20. Buyurtma № 26
Al Xorazmiy nomidagi

Toshkent axborot texnologiyalari universiteti
«Taxririyl nashriyot» bo’limida chop etildi.
Toshkent sh. Amir Temur ko‘chasi 108-uy.