mint数据流程

# 一、从原始数据（视频、音乐）到标定数据（aist\_plusplus\_final）

## 1、最原始数据

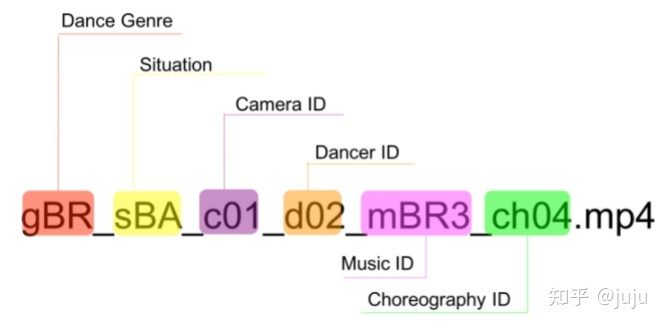
### 1.1、视频

* 下载

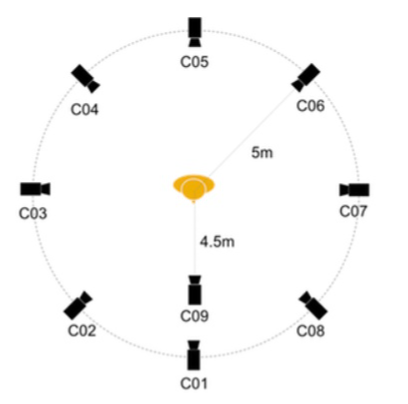
https://github.com/google/aistplusplus\_api根目录下有一个[downloader.py](https://github.com/google/aistplusplus_api/blob/main/downloader.py" \o "downloader.py)，如下使用。

python video-downloader.py --download\_folder="保存目录" --num\_processes=同时下载进程数量

* 视频文件名说明

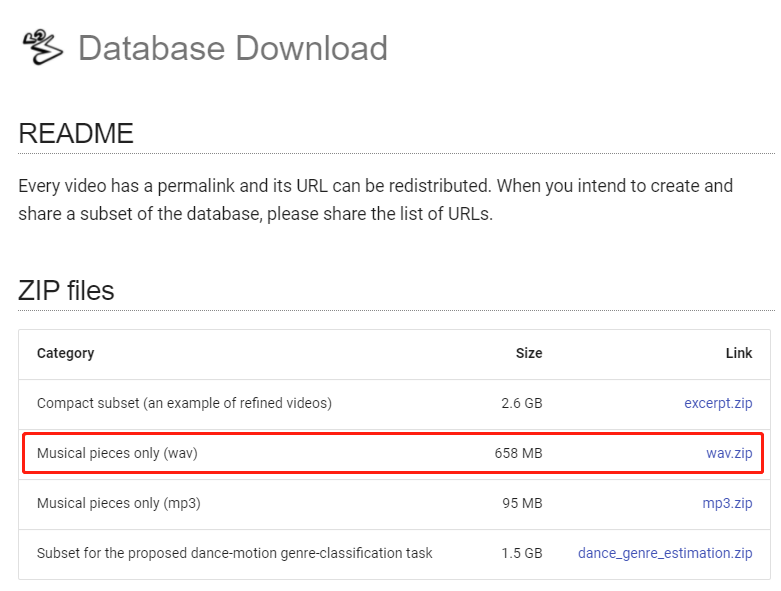


* 相机



### 1.2、音频

https://aistdancedb.ongaaccel.jp/database\_download/



## 2、标定方法（未知）

## 3、标定后的格式

以gBR\_sBM\_c\*\_d04\_mBR0\_ch01.mp4这些视频为例，\*从01到09共9个文件，对应9个相机位。每个视频长度是12秒，每秒60帧，共720帧。

对应的音频是mBR0.wav。

### 3.1、Keypoints3d

对应文件aist\_plusplus\_final/keypoints3d/gBR\_sBM\_cAll\_d04\_mBR0\_ch01.pkl。

import pickle

file\_3d = "I:/mint/data/aist\_plusplus\_final/keypoints3d/gBR\_sBM\_cAll\_d04\_mBR0\_ch01.pkl"

with open(file\_3d, 'rb') as f:

data = pickle.load(f)

#print(data)

keypoints3d = data['keypoints3d']

print(type(keypoints3d))

print(keypoints3d.shape)

print(keypoints3d[100])

keypoints3d\_optim = data['keypoints3d\_optim']

print(keypoints3d\_optim.shape)

print(keypoints3d\_optim[100])

数据格式

* keypoints3d
* 维度是(720, 17, 3)，数据类型是numpy.ndarray
* 720是视频的总帧数
* 17是COCO-format keypoint，[https://cocodataset.org/#home](https://cocodataset.org/" \l "home)，定义为

[

"nose",

"left\_eye", "right\_eye", "left\_ear", "right\_ear", "left\_shoulder","right\_shoulder",

"left\_elbow", "right\_elbow", "left\_wrist", "right\_wrist", "left\_hip", "right\_hip",

"left\_knee", "right\_knee", "left\_ankle", "right\_ankle"

]

* 3是坐标xyz还是yaw、pitch、roll？相对什么的坐标？单位？
* 第100帧的数据如下

[[ 9.68922340e+00 2.01815773e+02 2.40637835e+01]

[ 1.21819877e+01 2.05298870e+02 2.28144592e+01]

[ 5.62108039e+00 2.04960360e+02 2.19849321e+01]

[ 1.77721585e+01 2.05891705e+02 1.27942155e+01]

[ 1.35138106e+00 2.04535064e+02 1.23618802e+01]

[ 2.90139913e+01 1.93188706e+02 4.00846875e+00]

[-4.04994384e+00 1.87404834e+02 6.69162593e-03]

[ 5.21302428e+01 1.84772922e+02 -5.98155728e+00]

[-2.51256101e+01 1.73051141e+02 -8.95421579e+00]

[ 7.62920746e+01 1.81118582e+02 -3.76618662e+00]

[-4.85605746e+01 1.63625278e+02 -8.40008554e+00]

[ 3.28367072e+01 1.39531099e+02 2.88862486e+00]

[ 1.09067628e+01 1.37675100e+02 5.85805081e+00]

[ 3.05345828e+01 1.02664634e+02 9.73807533e+00]

[ 2.18232473e+01 1.05175799e+02 2.33976901e+01]

[ 1.51058700e+01 7.71230962e+01 -1.53018173e+01]

[ 3.33364472e+01 7.23303933e+01 3.49360900e+01]]

* keypoints3d\_optim
* 维度与keypoints3d相同
* 第100帧的数据如下，与keypoints3d基本相等

[[ 9.43480994 201.64837474 23.66209551]

[ 12.42986771 205.55980326 22.10199697]

[ 5.77068165 205.01589027 21.84044016]

[ 17.78743408 205.82878741 12.79111237]

[ 1.34092704 204.41543018 12.37614009]

[ 29.32325243 193.32517151 3.28566179]

[ -3.8819109 187.15061044 -0.55219018]

[ 52.21091117 184.57974249 -6.05777004]

[-25.0998953 172.94921812 -9.0448049 ]

[ 75.47635381 181.02038305 -4.0350102 ]

[-48.0830076 163.45869951 -8.30228443]

[ 32.62942076 139.71244807 2.55996631]

[ 11.335914 137.56727895 5.74867186]

[ 29.89228763 102.15763572 8.99558915]

[ 21.74458496 105.11553734 23.23924157]

[ 15.49369324 77.18673356 -14.84061659]

[ 33.32274017 72.33500391 34.82042212]]

### 3.2、Keypoints2d

对应文件aist\_plusplus\_final/keypoints2d/gBR\_sBM\_cAll\_d04\_mBR0\_ch01.pkl。

file\_2d = "I:/mint/data/aist\_plusplus\_final/keypoints2d/gBR\_sBM\_cAll\_d04\_mBR0\_ch01.pkl"

with open(file\_2d, 'rb') as f:

data = pickle.load(f)

#print(data)

keypoints2d = data['keypoints2d']

print(keypoints2d.shape)

print(keypoints2d[0][100])

det\_scores = data['det\_scores']

print(det\_scores.shape)

print(det\_scores[0][100:110])

timestamps = data['timestamps']

print(timestamps.shape)

print(timestamps[0:10])

数据格式

* keypoints2d
* 维度是(9, 720, 17, 3)
* 9是9个相机位
* 720是视频的总帧数
* 17是COCO-format keypoint
* 3是坐标xyz或yaw、pitch、roll？
* 第0个相机的第100帧数据如下

[[9.8100000e+02 4.7900000e+02 9.4721729e-01]

[9.8600000e+02 4.6800000e+02 9.7761357e-01]

[9.7000000e+02 4.7000000e+02 9.7738636e-01]

[1.0030000e+03 4.7000000e+02 9.0606987e-01]

[9.5900000e+02 4.7600000e+02 8.1568462e-01]

[1.0360000e+03 5.0900000e+02 7.8445530e-01]

[9.4500000e+02 5.3100000e+02 9.0304375e-01]

[1.1000000e+03 5.3400000e+02 7.9722464e-01]

[8.8400000e+02 5.6700000e+02 8.2262009e-01]

[1.1690000e+03 5.4200000e+02 9.0778506e-01]

[8.1700000e+02 5.9200000e+02 8.5662198e-01]

[1.0470000e+03 6.6700000e+02 7.3388356e-01]

[9.8900000e+02 6.7300000e+02 8.1191504e-01]

[1.0390000e+03 7.7800000e+02 7.2347325e-01]

[1.0200000e+03 7.7500000e+02 7.9804802e-01]

[9.9500000e+02 8.3000000e+02 4.5519388e-01]

[1.0560000e+03 8.8300000e+02 7.6395023e-01]]

* det\_scores
* 维度是(9, 720)
* 9是9个相机位
* 720是视频的总帧数
* 数值含义未知
* 第0个相机的[100:110]帧数据如下

[0.8224816 0.82239044 0.8432699 0.8369364 0.82911783 0.8433778

0.8451235 0.8412858 0.84833246 0.8496712 ]

* timestamps
* 维度是(720)
* 720是视频的总帧数
* 数据含义是取样点的时间，单位是微秒（1秒=1000000微秒）
* 第[0:10]帧数据如下

[ 0 16667 33333 50000 66667 83333 100000 116667 133333 150000]

### 3.3、Motions

对应文件aist\_plusplus\_final/motions/gBR\_sBM\_cAll\_d04\_mBR0\_ch01.pkl。

file\_motion = "I:/mint/data/aist\_plusplus\_final/motions/gBR\_sBM\_cAll\_d04\_mBR0\_ch01.pkl"

with open(file\_motion, 'rb') as f:

data = pickle.load(f)

#print(data)

smpl\_loss = data['smpl\_loss']

print(smpl\_loss)

smpl\_poses = data['smpl\_poses']

print(smpl\_poses.shape)

print(smpl\_poses[100])

smpl\_scaling = data['smpl\_scaling'] # (1,)

print(smpl\_scaling.shape)

print(smpl\_scaling)

smpl\_trans = data['smpl\_trans'] # (N, 3)

print(smpl\_trans.shape)

print(smpl\_trans[100])

数据格式

* smpl\_loss
* 标量，此处是1.5907235145568848
* smpl\_poses
* 维度是(720, 72)
* 720是视频的总帧数
* 72=24\*3，24的含义是SMPL-format body joint（<https://smpl.is.tue.mpg.de/>，具体如下），3是以弧度为单位的3个转动角度，应该是yaw、pitch、roll的某种顺序。

[

00、"root(pelvis)",

01、"lhip",

02、"rhip",

03、"belly(lowerback、spine1)",

04、"lknee",

05、"rknee",

06、"spine(upperback、spine2)",

07、"lankle",

08、"rankle",

09、"chest(spine3)",

10、"ltoes(lfoot)",

11、"rtoes(rfoot)",

12、"neck(lowerneck)",

13、"linshoulder(lclavicle、lcollar)",

14、"rinshoulder(rclavicle、rcollar)",

15、"head(upperneck)",

16、"lshoulder",

17、"rshoulder",

18、"lelbow",

19、"relbow",

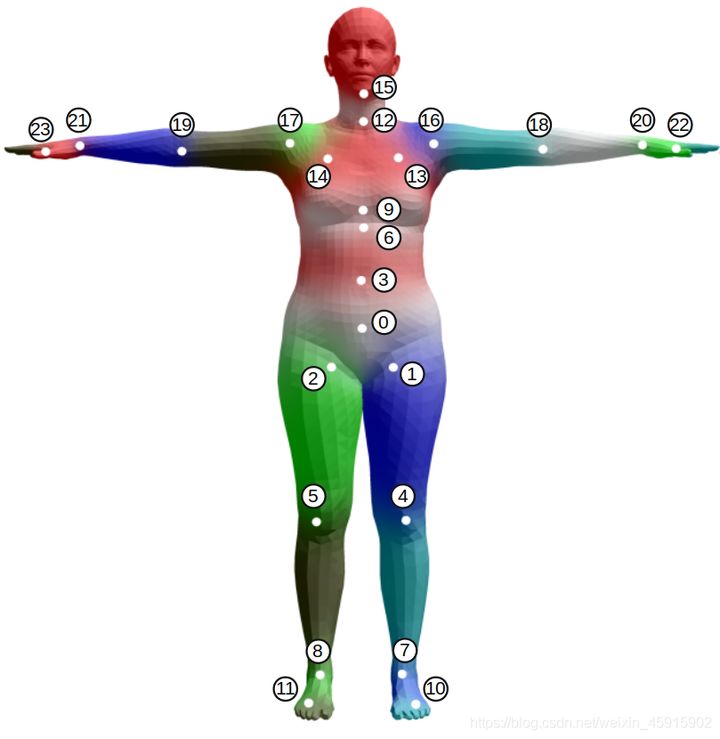
20、"lwrist",

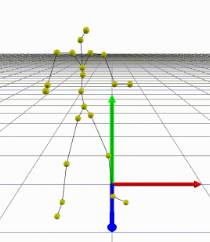
21、"rwrist",

22、"lhand",

23、"rhand",

]





* 第100帧的数据如下

[-1.82135865e-01 9.26153809e-02 8.03668723e-02 2.67502349e-02

-8.97292793e-02 -1.19448788e-01 -3.51499766e-01 4.84395437e-02

1.27387628e-01 1.62065193e-01 -1.14960253e-01 1.85872883e-01

7.59757400e-01 -2.07929350e-02 -4.07633156e-01 8.68370906e-02

-6.00908091e-03 4.78418469e-02 1.62435055e-01 -8.38840902e-02

1.82709936e-02 -1.13788748e-03 3.85971530e-03 -8.64725746e-03

1.65434566e-03 -7.23202131e-04 -4.99662617e-03 1.59184813e-01

-7.12527856e-02 -4.28883731e-02 8.39340128e-03 3.59594449e-03

9.21254605e-03 3.71604902e-03 -2.59635883e-04 1.88080140e-03

1.50836989e-01 4.35418896e-02 -1.12608679e-01 -7.08964765e-02

2.11541921e-01 -1.17979735e-01 -1.63503155e-01 -1.62990823e-01

-1.37378350e-02 -1.74262196e-01 1.32350260e-02 -7.63404369e-02

-6.41738027e-02 9.08346921e-02 -4.97144789e-01 -8.27528536e-02

1.26391605e-01 3.35001618e-01 -2.18952075e-02 -3.55378449e-01

1.34846389e-01 -8.24974757e-03 2.35292599e-01 -9.19892937e-02

1.01782056e-02 -2.56616040e-03 2.34512123e-03 -1.50089432e-03

7.49865954e-04 -5.74794458e-03 3.02228285e-03 3.02417157e-03

-1.63461466e-03 -3.38298292e-03 -2.35320232e-03 2.41386099e-03]

* smpl\_scaling
* 是个一维数组，此处只有一个值93.77886
* smpl\_trans
* 维度是(720, 3)
* 720是视频的总帧数
* 3是坐标xyz
* 第100帧的数据如下，含义应该是整体的平移，并且是放大了smpl\_scaling倍的

[ 21.675306 169.16722 1.4816978]

### 3.4、Cameras

配置，说明了各个视频的相机配置，一个相机配置如下

[

{

"name": "c01",

"size": [1920, 1080],

"matrix": [[1310.486088045505, 0.0, 960.0], [0.0, 1310.486088045505, 540.0], [0.0, 0.0, 1.0]],

"distortions": [-0.11413276507062059, 0.0, 0.0, 0.0, 0.0],

"rotation": [3.119905637132615, 0.00793503727830112, -0.024953662352903517],

"translation": [-3.104077194098298, 182.54559013217388, 453.39325012127074]

},

{

"name": "c02",

"size": [1920, 1080],

"matrix": [[1522.436019149514, 0.0, 960.0], [0.0, 1522.436019149514, 540.0], [0.0, 0.0, 1.0]],

"distortions": [0.06580022138027902, 0.0, 0.0, 0.0, 0.0],

"rotation": [2.8843352355123546, 0.010369166642977863, 1.1906199150206112],

"translation": [-0.8424770995171424, 184.24424242336184, 475.88675657738145]

},

{"name": "c03", ……},

……

{"name": "c09", ……}

]

### 3.5、Splits

配置，把视频归为3类：train、evaluate、test。

# 二、从标定数据（aist\_plusplus\_final）到训练数据（tf\_sstables）

在mint目录下，使用如下命令生成

python tools/preprocessing.py --anno\_dir="aist\_plusplus\_final的路径-audio\_dir="音频目录的路径" --split=train

音频数据，是使用librosa来处理的，以mWA4.wav为例，长度32秒，生成的feature数据维度是(1921,35)，1921≈60\*32=1920。

动作数据，使用的是标定数据中的motions部分，以gWA\_sFM\_cAll\_d25\_mWA4\_ch05为例，smpl\_poses的维度是(1919, 72)，处理代码如下，把每组1\*3的3个弧度制角度转变成对应的3\*3的矩阵。

smpl\_poses = R.from\_rotvec(smpl\_poses.reshape(-1, 3)).as\_matrix().reshape(smpl\_poses.shape[0], -1)

过程中维度变化过程是(1919, 72)->(46056=1919\*72/3, 3)->(46056, 3, 3)->(1919, 216=72/3\*9)。

然后把除以smpl\_scaling后的维度是(1919, 3)的smpl\_trans，和(1919, 216)的smpl\_poses拼接，smpl\_trans在前，smpl\_poses在后，变成(1919, 219)。

然后把音频和动作数据flatten成一维，连同名字和原维度信息存入dict，再转成tf.train.Example，随机写入一个20个文件种的1个。

# 三、训练

输入120帧（代码中写的是sec，是错误），目标20帧

## 1动作motion

feature\_dim是219，219 = 24个节点\*(3x3的矩阵) + 3个平移。

sample\_rate是1。

后续fact\_preprocessing会把平移也转成3\*3的矩阵，feature\_dim会再增加6，从tf.pad(example["motion\_sequence"], [[0, 0], [6, 0]])可知，在整个(1919, 219)数据的左侧增加了6列0，也即用[0, 0, 0, 0, 0, 0,x,y,z]表示平移。这种行为应该是没有意义的，3x3的矩阵表示不了旋转+平移，应该用4x4。

训练时，每个记录，随机取140帧数据，前120帧作为example["motion\_input"]【120\*225】，后20帧作为example["target"]【20\*225】。对应的音频数据则取240帧放入example["audio\_input"]【240\*35】。

连续取batch\_size个。

## 2音频audio

feature\_dim是35

sample\_rate是2

# 四、评估