In [20]:
```
# import Section
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cv2
from os import listdir
from os.path import isfile, join, isdir
import re
from matplotlib import pyplot as plt
import seaborn as sns
import glob
import re
from scipy.interpolate import interp1d
from tqdm import tqdm, tqdm_notebook
```

In [73]:
```
# Function Section
def calculate_pad(brightness, saturation):
    p = 0.69*brightness + 0.22*saturation
    a = -0.31*brightness + 0.6*saturation
    d = 0.76*brightness + 0.32*saturation
    return [p,a,d]

def calculate_pad_scene(scene):
    pads = []
    for img in scene:
        temp_b = mean_brightness(img)
        temp_s = mean_saturation(img)
        pads.append(calculate_pad(temp_b, temp_s))
    return np.mean([x[0] for x in pads]), np.mean([x[1] for x in pads]), np.mean

def calculate_blur(img):
    return cv2.Laplacian(img, cv2.CV_64F).var()

def calculate_blur_scene(scene):
    blurs = []
    for img in scene:
        blurs.append(calculate_blur(img))
    return np.mean(blurs)

def mean_brightness(img):
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #convert it to hsv
    return np.mean(hsv[:,:,2])

def mean_saturation(img):
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #convert it to hsv
    return np.mean(hsv[:,:,1])

def calculate_opticalFlow(img1, img2):
    prev = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
    forward = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
    mask = np.zeros_like(prev)
    mask[..., 1] = 255
    flow = cv2.calcOpticalFlowFarneback(prev, forward, flow=None, pyr_scale=0.5,
    magnitude, angle = cv2.cartToPolar(flow[..., 0], flow[..., 1])
    return cv2.normalize(magnitude, None, 0, 255, cv2.NORM_MINMAX)[0]

def calculate_opticalFlow_scene(scene):
    first = calculate_opticalFlow(scene[0], scene[1])
    second = calculate_opticalFlow(scene[1], scene[2])
    return np.mean([first, second])

def isjpg(filepath):
    return re.search(".jpg$", filepath)
```

In [74]:
```
# constant
base = 'data\\scenes'
movies = [x for x in listdir(base) if isdir(join(base, x)) and x != 'incredibles_
movies_paths = [join(base, x) for x in movies]
display(movies)
display(movies_paths)
img_paths = {}
for i in range(len(movies)):
    movie = movies[i]
    movie_path = movies_paths[i]
    files = [join(movie_path, f) for f in listdir(movie_path) if isjpg(join(movi
    img_paths[movie] = files
display([len(x) for x in img_paths.values()])
```

```
['big_hero_6', 'cars_3', 'incredible_2', 'toy_story_4', 'up', 'wall_e_']

['data\\scenes\\big_hero_6',
 'data\\scenes\\cars_3',
 'data\\scenes\\incredible_2',
 'data\\scenes\\toy_story_4',
 'data\\scenes\\up',
 'data\\scenes\\wall_e_']

[90, 90, 90, 90, 90, 90]
```

In [78]:
```
# data preprocessing
scene_names = []
scene_avg_ps = []
scene_avg_as = []
scene_avg_ds = []
scene_avg_blurs = []
scene_avg_optical_flows = []
scene_movies = []

for movie in tqdm_notebook(img_paths.keys()):
    display('preprocessing scenes in {m}'.format(m = movie))
    lst = img_paths[movie]
    for i in tqdm_notebook(range(0, 90, 3)):
        scene_num = lst[i][-8:-6].replace('-', '')
        scene_names.append(movie + scene_num)
        temp_imgs = []
        flag = False
        for j in range(3):
            img = cv2.imread(lst[i+j])
            if type(img) != type(None):
                img = cv2.resize(img, (320, 768))
                temp_imgs.append(img)
            else:
                flag = True
        temp_pad = calculate_pad_scene(temp_imgs)
        scene_avg_ps.append(temp_pad[0])
        scene_avg_as.append(temp_pad[1])
        scene_avg_ds.append(temp_pad[2])
        scene_avg_blurs.append(calculate_blur_scene(temp_imgs))
        scene_movies.append(movie)
        if not flag:
            scene_avg_optical_flows.append(calculate_opticalFlow_scene(temp_imgs
        else:
            scene_avg_optical_flows.append(np.nan)
```

```
                                                    100% 6/6 [01:39<00:00, 17.65s/it]

'preprocessing scenes in big_hero_6'

                                                    100% 30/30 [00:19<00:00, 1.64it/s]

'preprocessing scenes in cars_3'

                                                    100% 30/30 [00:13<00:00, 2.23it/s]

'preprocessing scenes in incredible_2'

                                                    100% 30/30 [00:12<00:00, 2.33it/s]

'preprocessing scenes in toy_story_4'

                                                    100% 30/30 [00:12<00:00, 2.36it/s]
```

In [83]:
```
clean = pd.DataFrame()
clean['scene_name'] = scene_names
clean['scene_avg_p'] = scene_avg_ps
clean['scene_avg_a'] = scene_avg_as
clean['scene_avg_d'] = scene_avg_ds
clean['scene_avg_blur'] = scene_avg_blurs
clean['scene_avg_optical_flow'] = scene_avg_optical_flows
clean['scene_movie'] = scene_movies
display(clean.shape)
display(clean.head())
clean.to_csv('clean_df.csv',  index=False)
```

```
(180, 7)
```

	scene_name	scene_avg_p	scene_avg_a	scene_avg_d	scene_avg_blur	scene_avg_optical_flow
0	big_hero_60	64.405538	72.981351	82.268261	204.102944	4.284929e-08
1	big_hero_61	76.658546	45.019868	93.268124	2058.214860	4.400276e-02
2	big_hero_610	93.844222	48.897762	113.486587	319.563208	7.499705e-03
3	big_hero_611	104.689522	68.900088	128.197153	344.318350	3.311605e-06
4	big_hero_612	52.825514	48.692155	66.235224	302.668920	1.479887e-03