Project #5: Video Stitching and Processing

CS445: Computational Photography - Spring 2020

Setup

In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

In [ ]:
```
# modify to where you store your project data including utils
datadir = "/content/drive/My Drive/cs445_projects/proj5/"
# datadir = "."

utilfn = datadir + "utils.py"
!cp "$utilfn" .
imagesfn = datadir + "images"
!cp -r "$imagesfn" .
```

In [2]:
```
!pip uninstall opencv-python -y
# downgrade OpenCV a bit to use SIFT
!pip install opencv-contrib-python==3.4.2.17 --force-reinstall
!pip install ffmpeg-python # for converting to video

import ffmpeg
import cv2
import numpy as np
import os
from numpy.linalg import svd, inv
import utils
%matplotlib inline
from matplotlib import pyplot as plt
```

Part I: Stitch two key frames

This involves:

1. compute homography H between two frames;
2. project each frame onto the same surface;
3. blend the surfaces.

Check that your homography is correct by plotting four points that form a square in frame 270 and their projections in each image.

In [3]:
```
def score_projection(pt1, pt2):
    '''
    Score corresponding to the number of inliers for RANSAC
    Input: pt1 and pt2 are 2xN arrays of N points such that
    pt1[:, i] and pt2[:,i] should be close in Euclidean distance
    if they are inliers
    Outputs: score (scalar count of inliers) and inliers (1xN logical array)
    '''
    diff_x = pt2[0, :] - pt1[0, :]
    diff_y = pt2[1, :] - pt1[1, :]
```

```
    inliers = np.sqrt(diff_x ** 2 + diff_y ** 2) < 1
    score = sum(inliers)

    return score, inliers


def auto_homography(Ia,Ib, homography_func=None,normalization_func=None):
    '''
    Computes a homography that maps points from Ia to Ib

    Input: Ia and Ib are images
    Output: H is the homography

    '''
    if Ia.dtype == 'float32' and Ib.dtype == 'float32':
        Ia = (Ia*255).astype(np.uint8)
        Ib = (Ib*255).astype(np.uint8)

    Ia_gray = cv2.cvtColor(Ia,cv2.COLOR_BGR2GRAY)
    Ib_gray = cv2.cvtColor(Ib,cv2.COLOR_BGR2GRAY)

    # Initiate SIFT detector
    sift = cv2.xfeatures2d.SIFT_create()

    # find the keypoints and descriptors with SIFT
    kp_a, des_a = sift.detectAndCompute(Ia_gray,None)
    kp_b, des_b = sift.detectAndCompute(Ib_gray,None)

    # BFMatcher with default params
    bf = cv2.BFMatcher()
    matches = bf.knnMatch(des_a,des_b, k=2)

    # Apply ratio test
    good = []
    for m,n in matches:
        if m.distance < 0.75*n.distance:
            good.append(m)

    numMatches = int(len(good))

    matches = good

    # Xa and Xb are 3xN matrices that contain homogeneous coordinates for the N
    # matching points for each image
    Xa = np.ones((3,numMatches))
    Xb = np.ones((3,numMatches))

    for idx, match_i in enumerate(matches):
        Xa[:,idx][0:2] = kp_a[match_i.queryIdx].pt
        Xb[:,idx][0:2] = kp_b[match_i.trainIdx].pt

    ## RANSAC
    niter = 1000
    best_score = 0
    n_to_sample = 4  # ???? # Put the correct number of points here

    for t in range(niter):
        # estimate homography
        subset = np.random.choice(numMatches, n_to_sample, replace=False)
        pts1 = Xa[:,subset]
        pts2 = Xb[:,subset]

        H_t = homography_func(pts1, pts2, normalization_func) # edit helper code bel
```

```
        # score homography
        Xb_ = np.dot(H_t, Xa) # project points from first image to second using H

        score_t, inliers_t = score_projection(Xb[:2,:]/Xb[2,:], Xb_[:2,:]/Xb_[2,:])

        if score_t > best_score:
            best_score = score_t
            H = H_t
            in_idx = inliers_t

    print('best score: {:02f}'.format(best_score))

    # Optionally, you may want to re-estimate H based on inliers

    return H
```

In [4]:
```
def computeHomography(pts1, pts2, normalization_func=None):
    '''
    Compute homography that maps from pts1 to pts2 using SVD. Normalization is optio

    Input: pts1 and pts2 are 3xN matrices for N points in homogeneous
    coordinates.

    Output: H is a 3x3 matrix, such that pts2~=H*pts1
    '''
    N = pts1.shape[1]

    A = np.zeros((2*N, 9))
    for i in range(N):
      p_1, p_2 = pts1[:, i], pts2[:, i]
      u_1, v_1 = p_1[0], p_1[1]
      u_2, v_2 = p_2[0], p_2[1]
      A[2*i, :] = [-u_1, -v_1, -1, 0, 0, 0, u_1*u_2, v_1*u_2, u_2]
      A[2*i+1, :] = [0, 0, 0, -u_1, -v_1, -1, u_1*v_2, v_1*v_2, v_2]

    u, s, vh = np.linalg.svd(A)
    H = vh[-1, :].reshape((3, 3))

    return H
```

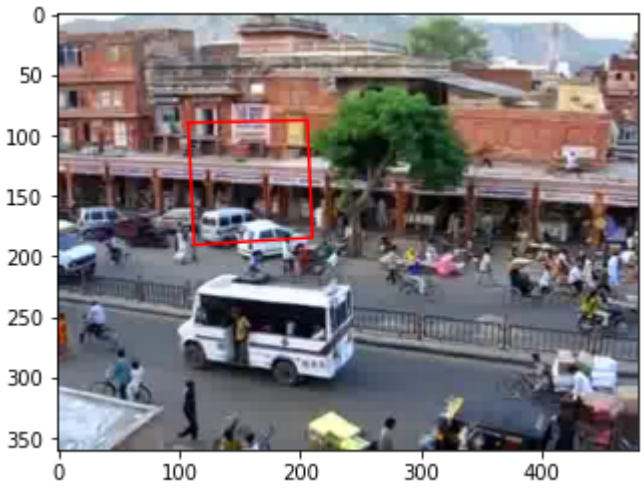
In [6]:
```
# images location
im1 = './images/input/frames/f0270.jpg'
im2 = './images/input/frames/f0450.jpg'

# Load an color image in grayscale
im1 = cv2.imread(im1)
im2 = cv2.imread(im2)

H = auto_homography(im1,im2, computeHomography)
print(H/H.max())

# plot the frames here
box_pts = np.array([[300, 400, 400, 300, 300], [100, 100, 200, 200, 100], [1, 1, 1,
plt.figure()
plt.imshow(im1[:,:,[2,1,0]])
plt.plot(box_pts[0,:], box_pts[1, :], 'r-')
```

```
best score: 156.000000
[[ 1.00000000e+00  5.29654357e-02 -2.05137393e+02]
 [ 1.33237574e-02  9.56042308e-01 -1.63008326e+01]
 [ 3.86867704e-04  4.38704151e-05  8.11410423e-01]]
```

Out[6]:
```
[<matplotlib.lines.Line2D at 0x7fd7e3eb9f50>]
```

In [ ]:
```
# project points into im2 and display the projected lines on im2
new_pts = H @ box_pts
new_pts /= new_pts[2]
plt.figure()
plt.imshow(im2[:,:,[2,1,0]])
plt.plot(new_pts[0,:], new_pts[1, :], 'r-')
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x7f2ddf48a050>]
```



In [ ]:
```
projectedWidth = 1600
projectedHeight = 600
Tr = np.array([[1, 0, 660], [0, 1, 120], [0, 0, 1]])

# warp and blend the two images
warp_1 = cv2.warpPerspective(im1, Tr @ H, (projectedWidth, projectedHeight))
warp_2 = cv2.warpAffine(im2, Tr[:2].astype(float), (projectedWidth, projectedHeight)
blend = utils.blendImages(warp_1, warp_2)
plt.figure(figsize=(15, 5))
plt.imshow(blend[:,:,[2,1,0]])
```

Out[ ]:
```
<matplotlib.image.AxesImage at 0x7f2ddee3f210>
```

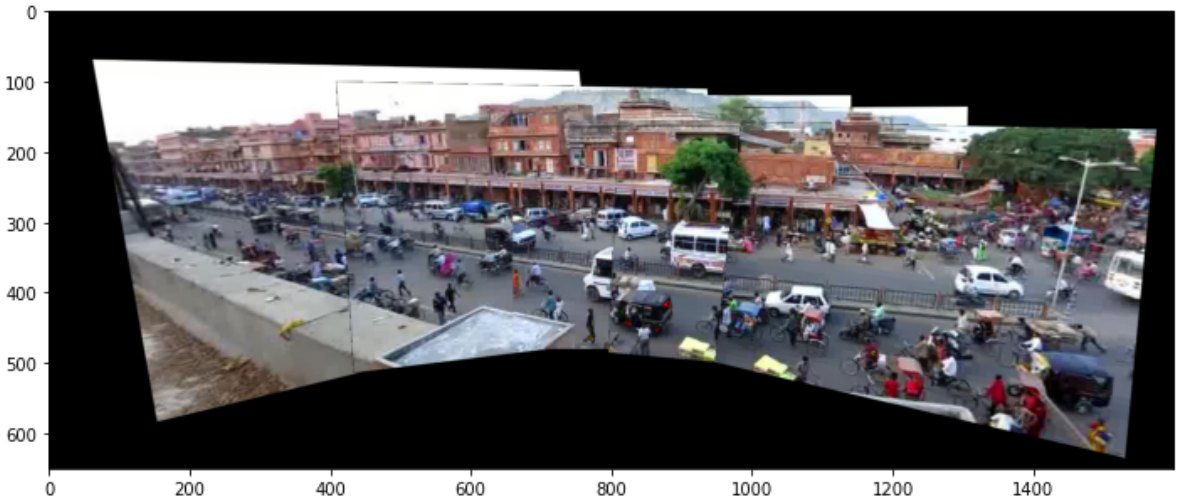
Part II: Panorama using five key frames

Produce a panorama by mapping five key frames [90, 270, 450, 630, 810] onto the same reference frame 450.

In [ ]:
```
key_frames_idx = np.array([90, 270, 450, 630, 810]) - 1

frames = np.zeros((len(key_frames_idx), im1.shape[0], im1.shape[1], im1.shape[2]),d
for n in range(len(key_frames_idx)):
    frames[n] = cv2.imread("./images/input/frames/f0{num}.jpg".format(num=str(key_fr


Hs = [
      auto_homography(frames[0], frames[1], computeHomography),
      auto_homography(frames[1], frames[2], computeHomography),
      auto_homography(frames[3], frames[2], computeHomography),
      auto_homography(frames[4], frames[3], computeHomography),
]

projectedWidth = 1600
projectedHeight = 650
Tr = np.array([[1, 0, 660], [0, 1, 120], [0, 0, 1]])

warps = [
         cv2.warpPerspective(frames[0], Tr @ (Hs[0] @ Hs[1]), (projectedWidth, proje
         cv2.warpPerspective(frames[1], Tr @ Hs[1], (projectedWidth, projectedHeight
         cv2.warpAffine(frames[2], Tr[:2].astype(float), (projectedWidth, projectedH
         cv2.warpPerspective(frames[3], Tr @ Hs[2], (projectedWidth, projectedHeight
         cv2.warpPerspective(frames[4], Tr @ (Hs[3] @ Hs[2]), (projectedWidth, proje
]

blend = warps[0]
for _, each in enumerate(warps[1:]):
    blend = utils.blendImages(blend, each)

plt.figure(figsize=(15, 5))
plt.imshow(blend[:,:,[2,1,0]])
```

```
best score: 212.000000
best score: 156.000000
best score: 145.000000
best score: 99.000000
```

Out[ ]:
```
<matplotlib.image.AxesImage at 0x7f2ddec93d90>
```



Part 3: Map the video to the reference plane

Project each frame onto the reference frame (using same size panorama) to create a video that shows the portion of the panorama revealed by each frame

In [5]:
```
# read all the images
import os
dir_frames = 'images/input/frames'
filenames = []
filesinfo = os.scandir(dir_frames)

filenames = [f.path for f in filesinfo if f.name.endswith(".jpg")]
filenames.sort(key=lambda f: int(''.join(filter(str.isdigit, f))))

frameCount = len(filenames)
frameHeight, frameWidth, frameChannels = cv2.imread(filenames[0]).shape
frames = np.zeros((frameCount, frameHeight, frameWidth, frameChannels),dtype='uint8'

for idx, file_i in enumerate(filenames):
    frames[idx] = cv2.imread(file_i)
```

In [ ]:
```
# part 3 solution
import os
out_path = os.path.join(datadir, 'images/output/frames')
if not os.path.exists(out_path):
    os.makedirs(out_path)

# create your video (see tips)
key_frames_idx = np.array([90, 270, 450, 630, 810]) - 1
key_frames = np.zeros((len(key_frames_idx), im1.shape[0], im1.shape[1], im1.shape[2]
for n in range(len(key_frames_idx)):
    key_frames[n] = cv2.imread("./images/input/frames/f0{num}.jpg".format(num=str(ke

projectedWidth = 1800
projectedHeight = 900
Tr = np.array([[1, 0, 660], [0, 1, 120], [0, 0, 1]])

Hs = [
      auto_homography(key_frames[0], key_frames[1], computeHomography),
      auto_homography(key_frames[1], key_frames[2], computeHomography),
      auto_homography(key_frames[3], key_frames[2], computeHomography),
      auto_homography(key_frames[4], key_frames[3], computeHomography),
]

for i, each_frame in enumerate(frames):
```

```
    if i < 180:
        each_H = auto_homography(each_frame, key_frames[0], computeHomography) @ (Hs
        im_warp = cv2.warpPerspective(each_frame, Tr @ each_H, (projectedWidth, proj
    elif i < 360:
        each_H = auto_homography(each_frame, key_frames[1], computeHomography) @ Hs[
        im_warp = cv2.warpPerspective(each_frame, Tr @ each_H, (projectedWidth, proj
    elif i < 540:
        each_H = auto_homography(each_frame, key_frames[2], computeHomography)
        im_warp = cv2.warpPerspective(each_frame, Tr @ each_H, (projectedWidth, proj
    elif i < 720:
        each_H = auto_homography(each_frame, key_frames[3], computeHomography) @ Hs[
        im_warp = cv2.warpPerspective(each_frame, Tr @ each_H, (projectedWidth, proj
    else:
        each_H = auto_homography(each_frame, key_frames[4], computeHomography) @ (Hs
        im_warp = cv2.warpPerspective(each_frame, Tr @ each_H, (projectedWidth, proj

    cv2.imwrite(os.path.join(out_path, f'{i}.jpg'), im_warp[:, :, [2, 1, 0]])

    # print(cv2.imread(os.path.join(out_path, f'{i}.jpg')))
    # break
```

best score: 202.000000
best score: 157.000000
best score: 140.000000
best score: 99.000000
best score: 245.000000
best score: 225.000000
best score: 251.000000
best score: 252.000000
best score: 221.000000
best score: 217.000000
best score: 211.000000
best score: 218.000000
best score: 239.000000
best score: 246.000000
best score: 217.000000
best score: 277.000000
best score: 234.000000
best score: 249.000000
best score: 269.000000
best score: 253.000000
best score: 199.000000
best score: 250.000000
best score: 240.000000
best score: 202.000000
best score: 240.000000
best score: 228.000000
best score: 242.000000
best score: 237.000000
best score: 231.000000
best score: 240.000000
best score: 248.000000
best score: 260.000000
best score: 266.000000
best score: 243.000000
best score: 256.000000
best score: 244.000000
best score: 239.000000
best score: 255.000000
best score: 267.000000
best score: 306.000000
best score: 268.000000
best score: 257.000000
best score: 230.000000
best score: 263.000000
best score: 249.000000
best score: 256.000000
best score: 272.000000
best score: 268.000000
best score: 260.000000
best score: 253.000000
best score: 260.000000
best score: 261.000000
best score: 295.000000
best score: 315.000000
best score: 301.000000
best score: 266.000000
best score: 248.000000
best score: 293.000000
best score: 291.000000
best score: 328.000000
best score: 318.000000
best score: 306.000000
best score: 334.000000
best score: 294.000000

best score: 284.000000
best score: 257.000000
best score: 245.000000
best score: 286.000000
best score: 263.000000
best score: 254.000000
best score: 309.000000
best score: 324.000000
best score: 277.000000
best score: 281.000000
best score: 306.000000
best score: 321.000000
best score: 289.000000
best score: 312.000000
best score: 352.000000
best score: 370.000000
best score: 333.000000
best score: 315.000000
best score: 309.000000
best score: 373.000000
best score: 379.000000
best score: 391.000000
best score: 395.000000
best score: 418.000000
best score: 409.000000
best score: 412.000000
best score: 446.000000
best score: 436.000000
best score: 681.000000
best score: 2816.000000
best score: 420.000000
best score: 408.000000
best score: 383.000000
best score: 373.000000
best score: 321.000000
best score: 308.000000
best score: 308.000000
best score: 290.000000
best score: 281.000000
best score: 315.000000
best score: 350.000000
best score: 352.000000
best score: 362.000000
best score: 340.000000
best score: 327.000000
best score: 295.000000
best score: 296.000000
best score: 298.000000
best score: 307.000000
best score: 273.000000
best score: 292.000000
best score: 282.000000
best score: 314.000000
best score: 318.000000
best score: 294.000000
best score: 268.000000
best score: 272.000000
best score: 278.000000
best score: 262.000000
best score: 272.000000
best score: 302.000000
best score: 287.000000
best score: 286.000000
best score: 272.000000

best score: 263.000000
best score: 277.000000
best score: 291.000000
best score: 297.000000
best score: 256.000000
best score: 273.000000
best score: 278.000000
best score: 254.000000
best score: 269.000000
best score: 259.000000
best score: 262.000000
best score: 267.000000
best score: 277.000000
best score: 242.000000
best score: 248.000000
best score: 246.000000
best score: 255.000000
best score: 256.000000
best score: 240.000000
best score: 260.000000
best score: 249.000000
best score: 228.000000
best score: 260.000000
best score: 251.000000
best score: 247.000000
best score: 240.000000
best score: 226.000000
best score: 243.000000
best score: 241.000000
best score: 249.000000
best score: 229.000000
best score: 229.000000
best score: 240.000000
best score: 207.000000
best score: 224.000000
best score: 223.000000
best score: 217.000000
best score: 260.000000
best score: 243.000000
best score: 238.000000
best score: 259.000000
best score: 226.000000
best score: 227.000000
best score: 212.000000
best score: 199.000000
best score: 221.000000
best score: 227.000000
best score: 217.000000
best score: 221.000000
best score: 236.000000
best score: 259.000000
best score: 253.000000
best score: 221.000000
best score: 184.000000
best score: 211.000000
best score: 228.000000
best score: 187.000000
best score: 169.000000
best score: 183.000000
best score: 198.000000
best score: 182.000000
best score: 215.000000
best score: 189.000000
best score: 170.000000

best score: 201.000000
best score: 180.000000
best score: 166.000000
best score: 178.000000
best score: 192.000000
best score: 170.000000
best score: 168.000000
best score: 190.000000
best score: 197.000000
best score: 196.000000
best score: 185.000000
best score: 210.000000
best score: 209.000000
best score: 207.000000
best score: 208.000000
best score: 192.000000
best score: 202.000000
best score: 207.000000
best score: 191.000000
best score: 200.000000
best score: 205.000000
best score: 190.000000
best score: 209.000000
best score: 187.000000
best score: 180.000000
best score: 222.000000
best score: 203.000000
best score: 175.000000
best score: 188.000000
best score: 195.000000
best score: 214.000000
best score: 186.000000
best score: 184.000000
best score: 178.000000
best score: 191.000000
best score: 187.000000
best score: 203.000000
best score: 206.000000
best score: 208.000000
best score: 199.000000
best score: 204.000000
best score: 209.000000
best score: 184.000000
best score: 166.000000
best score: 202.000000
best score: 193.000000
best score: 184.000000
best score: 191.000000
best score: 208.000000
best score: 196.000000
best score: 178.000000
best score: 188.000000
best score: 211.000000
best score: 202.000000
best score: 205.000000
best score: 203.000000
best score: 199.000000
best score: 223.000000
best score: 268.000000
best score: 273.000000
best score: 236.000000
best score: 259.000000
best score: 281.000000
best score: 255.000000

best score: 235.000000
best score: 265.000000
best score: 255.000000
best score: 274.000000
best score: 365.000000
best score: 312.000000
best score: 338.000000
best score: 240.000000
best score: 295.000000
best score: 304.000000
best score: 432.000000
best score: 324.000000
best score: 328.000000
best score: 311.000000
best score: 398.000000
best score: 457.000000
best score: 639.000000
best score: 2829.000000
best score: 547.000000
best score: 384.000000
best score: 340.000000
best score: 323.000000
best score: 327.000000
best score: 290.000000
best score: 287.000000
best score: 271.000000
best score: 270.000000
best score: 269.000000
best score: 265.000000
best score: 409.000000
best score: 244.000000
best score: 266.000000
best score: 216.000000
best score: 238.000000
best score: 308.000000
best score: 232.000000
best score: 243.000000
best score: 215.000000
best score: 225.000000
best score: 213.000000
best score: 218.000000
best score: 220.000000
best score: 205.000000
best score: 211.000000
best score: 200.000000
best score: 213.000000
best score: 191.000000
best score: 217.000000
best score: 234.000000
best score: 209.000000
best score: 217.000000
best score: 197.000000
best score: 193.000000
best score: 178.000000
best score: 199.000000
best score: 185.000000
best score: 205.000000
best score: 208.000000
best score: 195.000000
best score: 176.000000
best score: 184.000000
best score: 204.000000
best score: 174.000000
best score: 193.000000

best score: 169.000000
best score: 175.000000
best score: 190.000000
best score: 173.000000
best score: 181.000000
best score: 167.000000
best score: 205.000000
best score: 201.000000
best score: 171.000000
best score: 190.000000
best score: 169.000000
best score: 196.000000
best score: 156.000000
best score: 159.000000
best score: 170.000000
best score: 141.000000
best score: 165.000000
best score: 134.000000
best score: 170.000000
best score: 156.000000
best score: 169.000000
best score: 153.000000
best score: 176.000000
best score: 164.000000
best score: 172.000000
best score: 173.000000
best score: 175.000000
best score: 163.000000
best score: 171.000000
best score: 173.000000
best score: 166.000000
best score: 178.000000
best score: 159.000000
best score: 171.000000
best score: 172.000000
best score: 171.000000
best score: 159.000000
best score: 145.000000
best score: 172.000000
best score: 180.000000
best score: 144.000000
best score: 143.000000
best score: 162.000000
best score: 130.000000
best score: 166.000000
best score: 176.000000
best score: 171.000000
best score: 182.000000
best score: 185.000000
best score: 187.000000
best score: 178.000000
best score: 175.000000
best score: 200.000000
best score: 186.000000
best score: 196.000000
best score: 176.000000
best score: 195.000000
best score: 206.000000
best score: 205.000000
best score: 156.000000
best score: 211.000000
best score: 201.000000
best score: 199.000000
best score: 189.000000

best score: 179.000000
best score: 202.000000
best score: 197.000000
best score: 215.000000
best score: 245.000000
best score: 204.000000
best score: 197.000000
best score: 204.000000
best score: 206.000000
best score: 182.000000
best score: 203.000000
best score: 161.000000
best score: 189.000000
best score: 216.000000
best score: 188.000000
best score: 199.000000
best score: 199.000000
best score: 211.000000
best score: 198.000000
best score: 197.000000
best score: 186.000000
best score: 174.000000
best score: 207.000000
best score: 236.000000
best score: 208.000000
best score: 229.000000
best score: 216.000000
best score: 210.000000
best score: 235.000000
best score: 233.000000
best score: 212.000000
best score: 264.000000
best score: 246.000000
best score: 248.000000
best score: 204.000000
best score: 210.000000
best score: 195.000000
best score: 280.000000
best score: 256.000000
best score: 274.000000
best score: 269.000000
best score: 232.000000
best score: 241.000000
best score: 212.000000
best score: 233.000000
best score: 226.000000
best score: 214.000000
best score: 237.000000
best score: 246.000000
best score: 271.000000
best score: 258.000000
best score: 243.000000
best score: 247.000000
best score: 246.000000
best score: 240.000000
best score: 305.000000
best score: 300.000000
best score: 253.000000
best score: 264.000000
best score: 240.000000
best score: 257.000000
best score: 275.000000
best score: 270.000000
best score: 303.000000

best score: 303.000000
best score: 320.000000
best score: 386.000000
best score: 430.000000
best score: 686.000000
best score: 2951.000000
best score: 718.000000
best score: 351.000000
best score: 315.000000
best score: 274.000000
best score: 274.000000
best score: 257.000000
best score: 267.000000
best score: 236.000000
best score: 255.000000
best score: 217.000000
best score: 211.000000
best score: 254.000000
best score: 205.000000
best score: 221.000000
best score: 231.000000
best score: 216.000000
best score: 228.000000
best score: 250.000000
best score: 234.000000
best score: 185.000000
best score: 207.000000
best score: 198.000000
best score: 205.000000
best score: 195.000000
best score: 240.000000
best score: 225.000000
best score: 199.000000
best score: 196.000000
best score: 207.000000
best score: 201.000000
best score: 193.000000
best score: 192.000000
best score: 191.000000
best score: 167.000000
best score: 187.000000
best score: 204.000000
best score: 198.000000
best score: 187.000000
best score: 195.000000
best score: 192.000000
best score: 182.000000
best score: 186.000000
best score: 182.000000
best score: 172.000000
best score: 186.000000
best score: 182.000000
best score: 165.000000
best score: 182.000000
best score: 178.000000
best score: 185.000000
best score: 183.000000
best score: 184.000000
best score: 183.000000
best score: 191.000000
best score: 193.000000
best score: 190.000000
best score: 188.000000
best score: 206.000000

```
best score: 202.000000
best score: 195.000000
best score: 189.000000
best score: 171.000000
best score: 182.000000
best score: 194.000000
best score: 168.000000
best score: 192.000000
best score: 211.000000
best score: 189.000000
best score: 175.000000
best score: 184.000000
best score: 166.000000
best score: 195.000000
best score: 173.000000
best score: 178.000000
best score: 179.000000
best score: 162.000000
best score: 173.000000
best score: 148.000000
best score: 172.000000
best score: 163.000000
best score: 153.000000
best score: 164.000000
best score: 165.000000
best score: 159.000000
best score: 175.000000
best score: 176.000000
best score: 169.000000
best score: 179.000000
best score: 144.000000
best score: 151.000000
```

In [ ]:
```
utils.imageFolder2mpeg(out_path, os.path.join(datadir, 'images/output/part3.mp4'))
```

Part 4: Create background panorama

Create a background panorama based on the result from Part 3.

In [ ]:
```
# part 4
frameWidth = 1800
frameHeight = 900
import os
p4_path = os.path.join(datadir, 'images/output/frames')
p4_out = np.zeros((frameHeight, frameWidth, frameChannels))
```

In [ ]:
```
p4_frames = np.zeros((900, frameHeight, frameWidth, frameChannels))
```

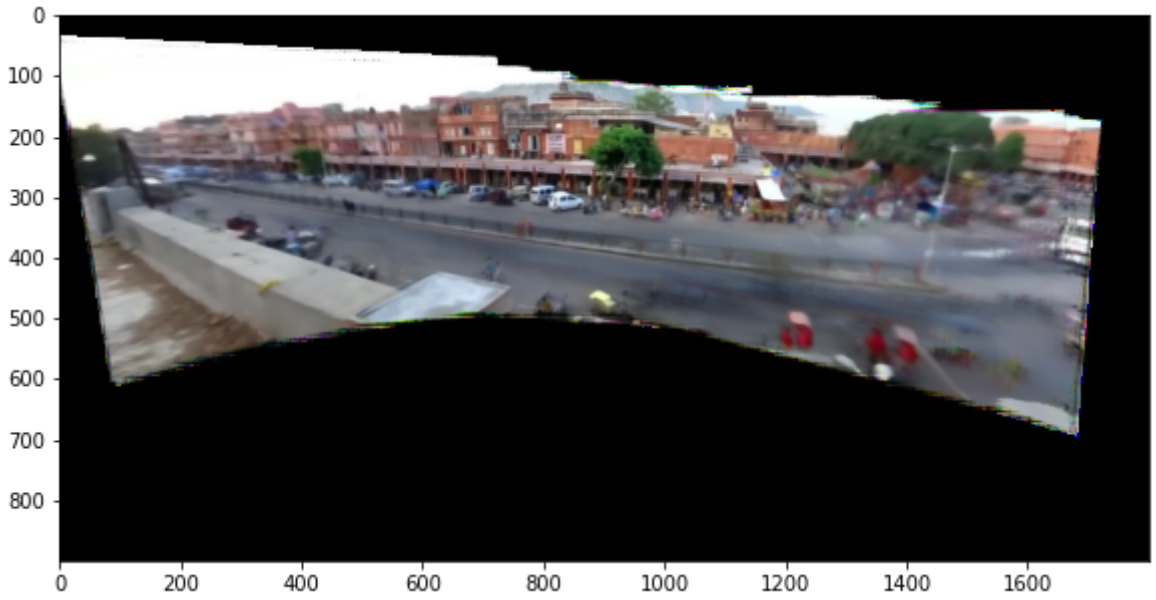
In [ ]:
```
# read images
for i in range(900):
    p4_frames[i] = cv2.imread(os.path.join(p4_path, f"{i}.jpg"))
```

In [ ]:
```
zero_to_nan = np.where(p4_frames == 0.0, np.nan, p4_frames)
p4_frames_medians = np.nanmedian(zero_to_nan, axis=0)
del zero_to_nan

# p4_frames_medians = np.median(p4_frames, axis=0)
print(p4_frames_medians.shape)
```

```
C:\Users\Eric\gitUIUC\cs445-sp22-mp5\venv\lib\site-packages\numpy\lib\nanfunctions.py:1096: RuntimeWarning: All-NaN slice encountered
  result = np.apply_along_axis(_nanmedian1d, axis, a, overwrite_input)
(900, 1800, 3)
```

In [ ]:
```
plt.figure(figsize=(15, 5))
plt.imshow(p4_frames_medians / 255)
```

Out[ ]:
```
<matplotlib.image.AxesImage at 0x2787c7671f0>
```



Part 5: Create background movie

Generate a movie that looks like the input movie but shows only background pixels. For each frame of the movie, you need to estimate a projection from the panorama to that frame. Your solution can use the background image you created in Part 4 and the per-frame homographies you created in Part 3.

In [11]:
```
# part 5
# USE FRAMES FROM PART 3
key_frames_idx = np.array([90, 270, 450, 630, 810]) - 1
key_frames = np.zeros((len(key_frames_idx), im1.shape[0], im1.shape[1], im1.shape[2]
for n in range(len(key_frames_idx)):
    key_frames[n] = cv2.imread("./images/input/frames/f0{num}.jpg".format(num=str(ke

projectedWidth = 1800
projectedHeight = 900
Tr = np.array([[1, 0, 660], [0, 1, 120], [0, 0, 1]])
```

In [14]:
```
Hs_pre = [
      auto_homography(key_frames[0], key_frames[1], computeHomography),
      auto_homography(key_frames[1], key_frames[2], computeHomography),
      auto_homography(key_frames[3], key_frames[2], computeHomography),
      auto_homography(key_frames[4], key_frames[3], computeHomography),
]

Hs = np.zeros((900, 3, 3))

for i, each_frame in enumerate(frames):
    if i < 180:
        Hs[i] = auto_homography(each_frame, key_frames[0], computeHomography) @ (Hs_
    elif i < 360:
```

```
        Hs[i] = auto_homography(each_frame, key_frames[1], computeHomography) @ Hs_p
    elif i < 540:
        Hs[i] = auto_homography(each_frame, key_frames[2], computeHomography)
    elif i < 720:
        Hs[i] = auto_homography(each_frame, key_frames[3], computeHomography) @ Hs_p
    else:
        Hs[i] = auto_homography(each_frame, key_frames[4], computeHomography) @ (Hs_
```

best score: 214.000000
best score: 162.000000
best score: 149.000000
best score: 101.000000
best score: 376.000000
best score: 372.000000
best score: 385.000000
best score: 375.000000
best score: 396.000000
best score: 390.000000
best score: 406.000000
best score: 401.000000
best score: 410.000000
best score: 429.000000
best score: 436.000000
best score: 397.000000
best score: 421.000000
best score: 400.000000
best score: 428.000000
best score: 430.000000
best score: 439.000000
best score: 430.000000
best score: 426.000000
best score: 425.000000
best score: 419.000000
best score: 428.000000
best score: 427.000000
best score: 436.000000
best score: 452.000000
best score: 431.000000
best score: 425.000000
best score: 439.000000
best score: 446.000000
best score: 446.000000
best score: 460.000000
best score: 455.000000
best score: 458.000000
best score: 445.000000
best score: 437.000000
best score: 462.000000
best score: 451.000000
best score: 452.000000
best score: 447.000000
best score: 481.000000
best score: 476.000000
best score: 471.000000
best score: 457.000000
best score: 457.000000
best score: 459.000000
best score: 429.000000
best score: 475.000000
best score: 470.000000
best score: 453.000000
best score: 468.000000
best score: 472.000000
best score: 477.000000
best score: 471.000000
best score: 480.000000
best score: 478.000000
best score: 453.000000
best score: 500.000000
best score: 520.000000
best score: 475.000000
best score: 497.000000

best score: 472.000000
best score: 501.000000
best score: 510.000000
best score: 509.000000
best score: 515.000000
best score: 513.000000
best score: 537.000000
best score: 555.000000
best score: 512.000000
best score: 522.000000
best score: 518.000000
best score: 517.000000
best score: 574.000000
best score: 524.000000
best score: 549.000000
best score: 546.000000
best score: 549.000000
best score: 537.000000
best score: 606.000000
best score: 641.000000
best score: 608.000000
best score: 553.000000
best score: 619.000000
best score: 613.000000
best score: 705.000000
best score: 708.000000
best score: 797.000000
best score: 813.000000
best score: 955.000000
best score: 2750.000000
best score: 926.000000
best score: 825.000000
best score: 656.000000
best score: 600.000000
best score: 655.000000
best score: 554.000000
best score: 574.000000
best score: 494.000000
best score: 535.000000
best score: 527.000000
best score: 497.000000
best score: 492.000000
best score: 469.000000
best score: 538.000000
best score: 460.000000
best score: 413.000000
best score: 438.000000
best score: 376.000000
best score: 451.000000
best score: 376.000000
best score: 400.000000
best score: 378.000000
best score: 394.000000
best score: 429.000000
best score: 386.000000
best score: 441.000000
best score: 370.000000
best score: 359.000000
best score: 368.000000
best score: 326.000000
best score: 362.000000
best score: 350.000000
best score: 321.000000
best score: 332.000000

best score: 325.000000
best score: 340.000000
best score: 308.000000
best score: 326.000000
best score: 329.000000
best score: 308.000000
best score: 308.000000
best score: 267.000000
best score: 319.000000
best score: 250.000000
best score: 269.000000
best score: 256.000000
best score: 268.000000
best score: 273.000000
best score: 248.000000
best score: 292.000000
best score: 294.000000
best score: 257.000000
best score: 291.000000
best score: 270.000000
best score: 271.000000
best score: 266.000000
best score: 254.000000
best score: 265.000000
best score: 283.000000
best score: 267.000000
best score: 254.000000
best score: 247.000000
best score: 255.000000
best score: 253.000000
best score: 265.000000
best score: 269.000000
best score: 255.000000
best score: 249.000000
best score: 262.000000
best score: 258.000000
best score: 257.000000
best score: 286.000000
best score: 270.000000
best score: 284.000000
best score: 278.000000
best score: 262.000000
best score: 233.000000
best score: 251.000000
best score: 235.000000
best score: 265.000000
best score: 240.000000
best score: 251.000000
best score: 257.000000
best score: 255.000000
best score: 254.000000
best score: 248.000000
best score: 253.000000
best score: 237.000000
best score: 240.000000
best score: 211.000000
best score: 187.000000
best score: 202.000000
best score: 201.000000
best score: 206.000000
best score: 186.000000
best score: 221.000000
best score: 210.000000
best score: 214.000000

best score: 200.000000
best score: 206.000000
best score: 234.000000
best score: 224.000000
best score: 215.000000
best score: 229.000000
best score: 219.000000
best score: 241.000000
best score: 217.000000
best score: 226.000000
best score: 209.000000
best score: 212.000000
best score: 231.000000
best score: 205.000000
best score: 214.000000
best score: 227.000000
best score: 216.000000
best score: 208.000000
best score: 241.000000
best score: 229.000000
best score: 217.000000
best score: 234.000000
best score: 242.000000
best score: 202.000000
best score: 224.000000
best score: 240.000000
best score: 213.000000
best score: 243.000000
best score: 245.000000
best score: 233.000000
best score: 232.000000
best score: 226.000000
best score: 277.000000
best score: 240.000000
best score: 265.000000
best score: 246.000000
best score: 233.000000
best score: 233.000000
best score: 244.000000
best score: 234.000000
best score: 231.000000
best score: 245.000000
best score: 220.000000
best score: 243.000000
best score: 201.000000
best score: 244.000000
best score: 222.000000
best score: 218.000000
best score: 221.000000
best score: 225.000000
best score: 253.000000
best score: 243.000000
best score: 236.000000
best score: 253.000000
best score: 259.000000
best score: 284.000000
best score: 272.000000
best score: 254.000000
best score: 278.000000
best score: 280.000000
best score: 291.000000
best score: 302.000000
best score: 294.000000
best score: 289.000000

best score: 291.000000
best score: 302.000000
best score: 287.000000
best score: 313.000000
best score: 364.000000
best score: 329.000000
best score: 349.000000
best score: 302.000000
best score: 282.000000
best score: 347.000000
best score: 372.000000
best score: 331.000000
best score: 361.000000
best score: 387.000000
best score: 387.000000
best score: 441.000000
best score: 477.000000
best score: 2819.000000
best score: 495.000000
best score: 462.000000
best score: 441.000000
best score: 422.000000
best score: 405.000000
best score: 323.000000
best score: 368.000000
best score: 327.000000
best score: 308.000000
best score: 306.000000
best score: 303.000000
best score: 287.000000
best score: 328.000000
best score: 305.000000
best score: 298.000000
best score: 311.000000
best score: 285.000000
best score: 291.000000
best score: 309.000000
best score: 278.000000
best score: 282.000000
best score: 261.000000
best score: 283.000000
best score: 285.000000
best score: 277.000000
best score: 253.000000
best score: 250.000000
best score: 248.000000
best score: 253.000000
best score: 264.000000
best score: 251.000000
best score: 244.000000
best score: 226.000000
best score: 241.000000
best score: 231.000000
best score: 261.000000
best score: 249.000000
best score: 242.000000
best score: 258.000000
best score: 275.000000
best score: 243.000000
best score: 279.000000
best score: 246.000000
best score: 256.000000
best score: 234.000000
best score: 245.000000

best score: 259.000000
best score: 251.000000
best score: 250.000000
best score: 233.000000
best score: 246.000000
best score: 235.000000
best score: 244.000000
best score: 242.000000
best score: 243.000000
best score: 245.000000
best score: 243.000000
best score: 241.000000
best score: 234.000000
best score: 227.000000
best score: 271.000000
best score: 210.000000
best score: 215.000000
best score: 223.000000
best score: 229.000000
best score: 212.000000
best score: 231.000000
best score: 243.000000
best score: 228.000000
best score: 196.000000
best score: 201.000000
best score: 202.000000
best score: 207.000000
best score: 183.000000
best score: 204.000000
best score: 201.000000
best score: 202.000000
best score: 200.000000
best score: 189.000000
best score: 217.000000
best score: 190.000000
best score: 198.000000
best score: 214.000000
best score: 221.000000
best score: 188.000000
best score: 220.000000
best score: 216.000000
best score: 214.000000
best score: 197.000000
best score: 203.000000
best score: 252.000000
best score: 226.000000
best score: 234.000000
best score: 262.000000
best score: 210.000000
best score: 225.000000
best score: 214.000000
best score: 223.000000
best score: 229.000000
best score: 242.000000
best score: 228.000000
best score: 272.000000
best score: 239.000000
best score: 252.000000
best score: 261.000000
best score: 262.000000
best score: 205.000000
best score: 266.000000
best score: 240.000000
best score: 210.000000

best score: 236.000000
best score: 229.000000
best score: 234.000000
best score: 243.000000
best score: 237.000000
best score: 228.000000
best score: 256.000000
best score: 249.000000
best score: 238.000000
best score: 250.000000
best score: 256.000000
best score: 261.000000
best score: 236.000000
best score: 246.000000
best score: 275.000000
best score: 307.000000
best score: 277.000000
best score: 266.000000
best score: 237.000000
best score: 273.000000
best score: 253.000000
best score: 259.000000
best score: 281.000000
best score: 248.000000
best score: 256.000000
best score: 252.000000
best score: 227.000000
best score: 247.000000
best score: 298.000000
best score: 323.000000
best score: 296.000000
best score: 291.000000
best score: 246.000000
best score: 272.000000
best score: 328.000000
best score: 326.000000
best score: 322.000000
best score: 321.000000
best score: 322.000000
best score: 298.000000
best score: 275.000000
best score: 265.000000
best score: 286.000000
best score: 275.000000
best score: 273.000000
best score: 269.000000
best score: 329.000000
best score: 312.000000
best score: 298.000000
best score: 269.000000
best score: 289.000000
best score: 317.000000
best score: 287.000000
best score: 334.000000
best score: 346.000000
best score: 384.000000
best score: 354.000000
best score: 331.000000
best score: 330.000000
best score: 386.000000
best score: 412.000000
best score: 381.000000
best score: 422.000000
best score: 408.000000

best score: 407.000000
best score: 419.000000
best score: 459.000000
best score: 465.000000
best score: 664.000000
best score: 2816.000000
best score: 431.000000
best score: 409.000000
best score: 381.000000
best score: 380.000000
best score: 327.000000
best score: 305.000000
best score: 300.000000
best score: 292.000000
best score: 274.000000
best score: 314.000000
best score: 354.000000
best score: 363.000000
best score: 330.000000
best score: 357.000000
best score: 316.000000
best score: 294.000000
best score: 307.000000
best score: 290.000000
best score: 325.000000
best score: 275.000000
best score: 295.000000
best score: 275.000000
best score: 299.000000
best score: 284.000000
best score: 283.000000
best score: 280.000000
best score: 275.000000
best score: 269.000000
best score: 283.000000
best score: 263.000000
best score: 312.000000
best score: 306.000000
best score: 285.000000
best score: 288.000000
best score: 257.000000
best score: 285.000000
best score: 274.000000
best score: 283.000000
best score: 261.000000
best score: 263.000000
best score: 293.000000
best score: 259.000000
best score: 276.000000
best score: 256.000000
best score: 251.000000
best score: 247.000000
best score: 270.000000
best score: 250.000000
best score: 265.000000
best score: 240.000000
best score: 234.000000
best score: 250.000000
best score: 231.000000
best score: 242.000000
best score: 221.000000
best score: 231.000000
best score: 260.000000
best score: 233.000000

best score: 235.000000
best score: 235.000000
best score: 238.000000
best score: 248.000000
best score: 234.000000
best score: 256.000000
best score: 238.000000
best score: 232.000000
best score: 240.000000
best score: 208.000000
best score: 213.000000
best score: 233.000000
best score: 239.000000
best score: 251.000000
best score: 256.000000
best score: 252.000000
best score: 255.000000
best score: 227.000000
best score: 225.000000
best score: 213.000000
best score: 209.000000
best score: 228.000000
best score: 223.000000
best score: 214.000000
best score: 221.000000
best score: 235.000000
best score: 254.000000
best score: 270.000000
best score: 234.000000
best score: 187.000000
best score: 202.000000
best score: 223.000000
best score: 177.000000
best score: 185.000000
best score: 198.000000
best score: 191.000000
best score: 174.000000
best score: 208.000000
best score: 176.000000
best score: 176.000000
best score: 203.000000
best score: 159.000000
best score: 170.000000
best score: 166.000000
best score: 181.000000
best score: 172.000000
best score: 176.000000
best score: 184.000000
best score: 194.000000
best score: 177.000000
best score: 183.000000
best score: 209.000000
best score: 201.000000
best score: 212.000000
best score: 216.000000
best score: 190.000000
best score: 211.000000
best score: 218.000000
best score: 196.000000
best score: 209.000000
best score: 198.000000
best score: 172.000000
best score: 202.000000
best score: 190.000000

best score: 187.000000
best score: 208.000000
best score: 191.000000
best score: 166.000000
best score: 185.000000
best score: 194.000000
best score: 187.000000
best score: 193.000000
best score: 181.000000
best score: 177.000000
best score: 191.000000
best score: 188.000000
best score: 201.000000
best score: 196.000000
best score: 214.000000
best score: 186.000000
best score: 207.000000
best score: 206.000000
best score: 198.000000
best score: 177.000000
best score: 203.000000
best score: 197.000000
best score: 204.000000
best score: 172.000000
best score: 205.000000
best score: 189.000000
best score: 191.000000
best score: 176.000000
best score: 204.000000
best score: 207.000000
best score: 199.000000
best score: 210.000000
best score: 215.000000
best score: 203.000000
best score: 259.000000
best score: 267.000000
best score: 254.000000
best score: 254.000000
best score: 262.000000
best score: 275.000000
best score: 252.000000
best score: 259.000000
best score: 265.000000
best score: 295.000000
best score: 354.000000
best score: 319.000000
best score: 327.000000
best score: 266.000000
best score: 279.000000
best score: 310.000000
best score: 417.000000
best score: 340.000000
best score: 316.000000
best score: 355.000000
best score: 383.000000
best score: 473.000000
best score: 616.000000
best score: 2829.000000
best score: 507.000000
best score: 349.000000
best score: 334.000000
best score: 340.000000
best score: 321.000000
best score: 284.000000

best score: 282.000000
best score: 266.000000
best score: 248.000000
best score: 253.000000
best score: 282.000000
best score: 414.000000
best score: 242.000000
best score: 262.000000
best score: 222.000000
best score: 232.000000
best score: 306.000000
best score: 230.000000
best score: 236.000000
best score: 220.000000
best score: 228.000000
best score: 228.000000
best score: 208.000000
best score: 224.000000
best score: 222.000000
best score: 222.000000
best score: 189.000000
best score: 218.000000
best score: 191.000000
best score: 226.000000
best score: 234.000000
best score: 210.000000
best score: 229.000000
best score: 199.000000
best score: 193.000000
best score: 184.000000
best score: 197.000000
best score: 178.000000
best score: 189.000000
best score: 206.000000
best score: 187.000000
best score: 192.000000
best score: 175.000000
best score: 188.000000
best score: 168.000000
best score: 197.000000
best score: 177.000000
best score: 191.000000
best score: 208.000000
best score: 185.000000
best score: 173.000000
best score: 184.000000
best score: 203.000000
best score: 188.000000
best score: 164.000000
best score: 176.000000
best score: 170.000000
best score: 194.000000
best score: 166.000000
best score: 162.000000
best score: 168.000000
best score: 158.000000
best score: 157.000000
best score: 143.000000
best score: 158.000000
best score: 144.000000
best score: 171.000000
best score: 146.000000
best score: 171.000000
best score: 174.000000

best score: 166.000000
best score: 180.000000
best score: 184.000000
best score: 166.000000
best score: 167.000000
best score: 168.000000
best score: 167.000000
best score: 176.000000
best score: 142.000000
best score: 169.000000
best score: 170.000000
best score: 176.000000
best score: 166.000000
best score: 165.000000
best score: 165.000000
best score: 180.000000
best score: 153.000000
best score: 154.000000
best score: 160.000000
best score: 140.000000
best score: 173.000000
best score: 193.000000
best score: 171.000000
best score: 182.000000
best score: 175.000000
best score: 176.000000
best score: 176.000000
best score: 188.000000
best score: 202.000000
best score: 187.000000
best score: 186.000000
best score: 178.000000
best score: 191.000000
best score: 205.000000
best score: 214.000000
best score: 160.000000
best score: 212.000000
best score: 207.000000
best score: 198.000000
best score: 188.000000
best score: 175.000000
best score: 187.000000
best score: 207.000000
best score: 212.000000
best score: 252.000000
best score: 210.000000
best score: 203.000000
best score: 176.000000
best score: 209.000000
best score: 181.000000
best score: 194.000000
best score: 148.000000
best score: 180.000000
best score: 217.000000
best score: 191.000000
best score: 194.000000
best score: 187.000000
best score: 204.000000
best score: 202.000000
best score: 197.000000
best score: 200.000000
best score: 192.000000
best score: 204.000000
best score: 234.000000

best score: 222.000000
best score: 219.000000
best score: 211.000000
best score: 201.000000
best score: 221.000000
best score: 236.000000
best score: 220.000000
best score: 242.000000
best score: 259.000000
best score: 251.000000
best score: 205.000000
best score: 212.000000
best score: 199.000000
best score: 262.000000
best score: 263.000000
best score: 269.000000
best score: 276.000000
best score: 230.000000
best score: 258.000000
best score: 227.000000
best score: 228.000000
best score: 240.000000
best score: 238.000000
best score: 243.000000
best score: 237.000000
best score: 258.000000
best score: 258.000000
best score: 244.000000
best score: 227.000000
best score: 243.000000
best score: 230.000000
best score: 306.000000
best score: 319.000000
best score: 247.000000
best score: 274.000000
best score: 263.000000
best score: 261.000000
best score: 277.000000
best score: 293.000000
best score: 298.000000
best score: 301.000000
best score: 336.000000
best score: 396.000000
best score: 408.000000
best score: 696.000000
best score: 2951.000000
best score: 744.000000
best score: 337.000000
best score: 318.000000
best score: 269.000000
best score: 261.000000
best score: 276.000000
best score: 251.000000
best score: 249.000000
best score: 241.000000
best score: 242.000000
best score: 232.000000
best score: 246.000000
best score: 244.000000
best score: 221.000000
best score: 230.000000
best score: 225.000000
best score: 236.000000
best score: 252.000000

best score: 241.000000
best score: 210.000000
best score: 205.000000
best score: 201.000000
best score: 207.000000
best score: 194.000000
best score: 245.000000
best score: 226.000000
best score: 203.000000
best score: 208.000000
best score: 203.000000
best score: 195.000000
best score: 205.000000
best score: 179.000000
best score: 190.000000
best score: 168.000000
best score: 190.000000
best score: 211.000000
best score: 209.000000
best score: 186.000000
best score: 201.000000
best score: 192.000000
best score: 186.000000
best score: 183.000000
best score: 157.000000
best score: 187.000000
best score: 186.000000
best score: 177.000000
best score: 156.000000
best score: 190.000000
best score: 180.000000
best score: 185.000000
best score: 179.000000
best score: 177.000000
best score: 178.000000
best score: 192.000000
best score: 183.000000
best score: 194.000000
best score: 183.000000
best score: 186.000000
best score: 183.000000
best score: 193.000000
best score: 187.000000
best score: 176.000000
best score: 176.000000
best score: 195.000000
best score: 167.000000
best score: 181.000000
best score: 214.000000
best score: 187.000000
best score: 175.000000
best score: 182.000000
best score: 182.000000
best score: 187.000000
best score: 155.000000
best score: 180.000000
best score: 182.000000
best score: 156.000000
best score: 168.000000
best score: 158.000000
best score: 172.000000
best score: 162.000000
best score: 138.000000
best score: 155.000000

```
best score: 160.000000
best score: 152.000000
best score: 186.000000
best score: 176.000000
best score: 169.000000
best score: 173.000000
best score: 144.000000
best score: 155.000000
```

In [16]:
```
np.save(os.path.join(datadir, 'p5_Hs'), Hs)
```

In [57]:
```
Hs = np.load(os.path.join(datadir, 'p5_Hs.npy'))
```

In [58]:
```
p4_frames_medians = np.load(os.path.join(datadir, 'p4_frames_medians.npy'))
```

In [21]:
```
import os
if not os.path.exists(os.path.join(datadir, 'images/output/part5')):
    os.makedirs(os.path.join(datadir, 'images/output/part5'))
```

In [62]:
```
Tr = np.array([[1, 0, 660], [0, 1, 120], [0, 0, 1]])
for i in range(900):
    this_H = Hs[i]
    im_warp = cv2.warpPerspective(p4_frames_medians, np.linalg.inv(Tr @ this_H), (48
    cv2.imwrite(os.path.join(datadir, 'images/output/part5', f'{i}.jpg'), im_warp[:,
```

In [63]:
```
utils.imageFolder2mpeg(os.path.join(datadir, 'images/output/part5'), os.path.join(da
```

Part 6: Create foreground movie

In the background video, moving objects are removed. In each frame, those pixels that are different enough than the background color are considered foreground. For each frame determine foreground pixels and generate a movie that emphasizes or includes only foreground pixels.

In [66]:
```
# part 6
p6_frames = np.zeros((900, 360, 480, 3))
for i in range(900):
    this_out = np.zeros((360, 480, 3))
    # BGR to RGB
    a = cv2.imread(os.path.join(datadir, "images/input/frames/f0{num}.jpg".format(nu
    b = cv2.imread(os.path.join(datadir, 'images/output/part5', f"{i}.jpg"))

    cum_abs_diff = np.abs(a - b).sum(axis=2)

    for h in range(360):
        for w in range(480):
            if cum_abs_diff[h, w] > 450:
                this_out[h, w] = a[h, w]

#     plt.figure(figsize=(15, 5))
#     plt.imshow(this_out / 255)

    cv2.imwrite(os.path.join('images/output/part6', f'{i}.jpg'), this_out[:,:,::-1])
#     break

# image diff
```

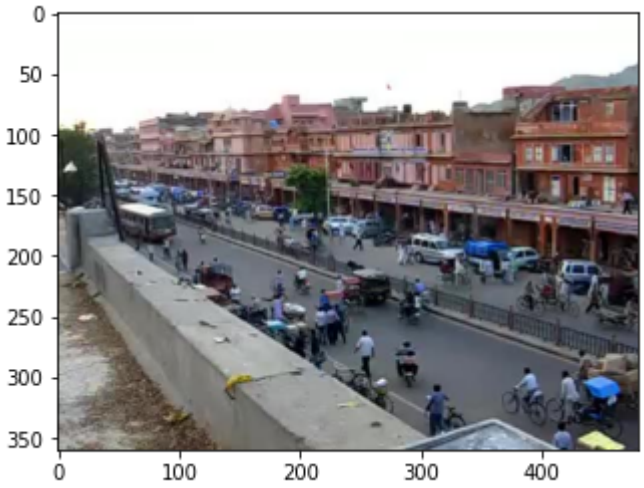
In [67]:
```
utils.imageFolder2mpeg(os.path.join(datadir, 'images/output/part6'), os.path.join(da
```

Unused

In [31]:
```
p5_Hs = np.load('p5_Hs.npy')
```

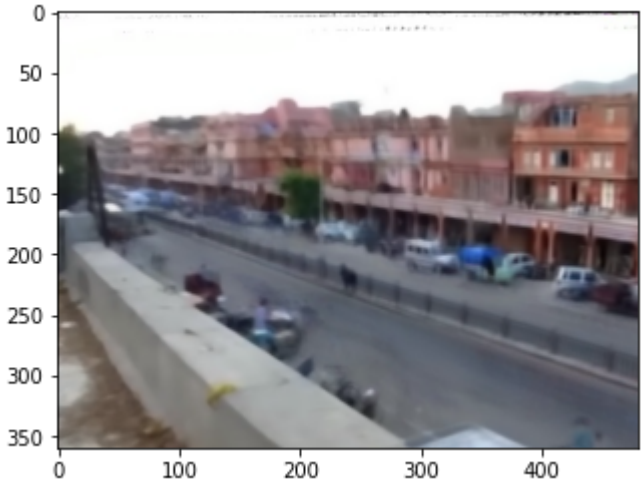
In [35]:
```
a = cv2.imread(os.path.join(datadir, "images/input/frames/f0{num}.jpg".format(num=s
plt.imshow(a)
```

Out[35]:
```
<matplotlib.image.AxesImage at 0x13a78ac06a0>
```



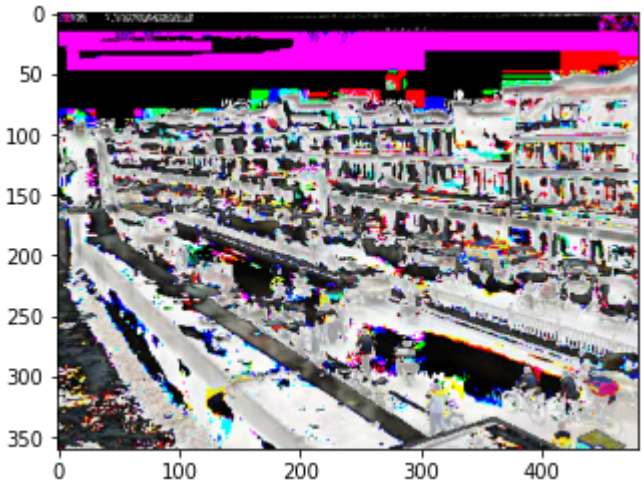
In [37]:
```
b = cv2.imread(os.path.join(datadir, 'images/output/part5', f"{0}.jpg"))
plt.imshow(b)
```

Out[37]:
```
<matplotlib.image.AxesImage at 0x13a78b7e9e0>
```



In [42]:
```
plt.imshow(a - b)
```

Out[42]:
```
<matplotlib.image.AxesImage at 0x13a79d46a40>
```



In [50]:
```
!pip3 install tqdm
from tqdm.notebook import tqdm
```

```
Requirement already satisfied: tqdm in c:\users\eric\gituiuc\cs445-sp22-mp5\venv\lib
\site-packages (4.64.0)
Requirement already satisfied: colorama in c:\users\eric\gituiuc\cs445-sp22-mp5\venv
\lib\site-packages (from tqdm) (0.4.4)
```

In [43]:
```
# part 6
p6_frames = np.zeros((900, 360, 480, 3))
for i in range(900):
    # BGR to RGB
    p6_frames[i] = cv2.imread(os.path.join(datadir, "images/input/frames/f0{num}.jpg
        - cv2.imread(os.path.join(datadir, 'images/output/part5', f"{i}.jpg"))

# image diff
```

In [55]:
```
del p6_out
```

In [56]:
```
frameWidth = 1800
frameHeight = 900
p6_out = np.zeros((900, frameHeight, frameWidth, frameChannels))
```

In [57]:
```
projectedWidth = 1800
projectedHeight = 900
Tr = np.array([[1, 0, 660], [0, 1, 120], [0, 0, 1]])

for i, each_frame in tqdm(enumerate(p6_frames), total=len(p6_frames)):
    if i < 180:
        im_warp = cv2.warpPerspective(each_frame, Tr @ p5_Hs[i], (projectedWidth, pr
    elif i < 360:
        im_warp = cv2.warpPerspective(each_frame, Tr @ p5_Hs[i], (projectedWidth, pr
    elif i < 540:
        im_warp = cv2.warpPerspective(each_frame, Tr @ p5_Hs[i], (projectedWidth, pr
    elif i < 720:
        im_warp = cv2.warpPerspective(each_frame, Tr @ p5_Hs[i], (projectedWidth, pr
    else:
        im_warp = cv2.warpPerspective(each_frame, Tr @ p5_Hs[i], (projectedWidth, pr

#     cv2.imwrite(os.path.join('images/output/part6', f'{i}.jpg'), im_warp[:, :, :])
    p6_out += im_warp
```

```
  0%|          | 0/900 [00:00<?, ?it/s]
```

In [58]:
```
zero_to_nan = np.where(p6_frames == 0.0, np.nan, p6_frames)
p6_frames_mean = np.nanmean(zero_to_nan, axis=0)
del zero_to_nan
```

```
print(p6_frames_mean.shape)
```

```
(360, 480, 3)
```

In [60]:
```
np.save('E:\p6_tmp', p6_out)
```

Bells and whistles

In [ ]:
```

```