Poročilo - Vaja 1

Pri tej vaji smo opravili ocenjevanje homografije s pomočjo nevronske mreže.

Nalaganje in predobdelava slik:

```
class ImageDataSet(Dataset):
    def __init__(self, dirpath):
        self.dirpath = dirpath
        if not file_exists(dirpath+"\\offsets.npy"):
            self.offsets = calculate_frame_offsets(dirpath)
            write_to_file(self.offsets, dirpath+"\\offsets.npy")
        else:
            self.offsets = read_from_file(dirpath+"\\offsets.npy")
    def __len__(self):
        return len(self.offsets)
    def __getitem__(self, index):
        path = self.dirpath+"\\"+self.offsets[index][0]
        img = cv2.imread(path)
        img = cv2.resize(img, (320, 240))
        img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        orig_img = img.copy()
        x = random.randint(0+48,320-48)
        y = random.randint(0+48,240-48)
        corners = np.array([[x-32,y-32],[x+32,y-32],[x+32,y+32],[x-32,y+32]], dtype=np.int32)
        transformed_corners = np.array([[x-32+random.randint(-16,16),y-32+random.randint(-16,16)],
                                        [x+32+random.randint(-16,16),y-32+random.randint(-16,16)],[x+32+random.randint(-16,16),y+32+random.randint(-16,16)],
                                        [x-32+random.randint(-16,16),y+32+random.randint(-16,16)]], dtype=np.int32)
        homography_matrix, _ = cv2.findHomography(corners,transformed_corners)
        transformed_img = cv2.warpPerspective(img, homography_matrix, (320,240),flags=cv2.WARP_INVERSE_MAP)

        transformed_img = transformed_img[y-32:y+32,x-32:x+32]

        corner_offsets = transformed_corners - corners
        corner_offsets = corner_offsets.flatten()
        corner_offsets = corner_offsets.astype(np.float32)
        corner_offsets += 16
        corner_offsets /= 32

        img = img[y-32:y+32,x-32:x+32]

        '''cv2.imshow("img",img)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
        cv2.imshow("img",transformed_img)
        cv2.waitKey(0)
        cv2.destroyAllWindows()'''
        img = cv2.merge((img,transformed_img))
        img = img/255
        return np.array(img).astype(np.float32),corners,transformed_corners,homography_matrix,corner_offsets,orig_img
```

Struktura:

==========

==========

Layer (type:depth-idx) Output Shape Param #

==========

==========

HomoNet [64, 8] 1,376,424

├─ResNet: 1-1 [64, 64, 64, 64] --

```
|    └─Sequential: 2-1                   [64, 64, 64, 64]          --
|    |    └─Conv2d: 3-1                  [64, 64, 64, 64]          (128)
|    |    └─BatchNorm2d: 3-2             [64, 64, 64, 64]          (128)
|    └─Conv2d: 2-2                       [64, 64, 64, 64]          (1,152)
|    └─BatchNorm2d: 2-3                  [64, 64, 64, 64]          (128)
|    └─ReLU: 2-4                         [64, 64, 64, 64]          --
|    └─Conv2d: 2-5                       [64, 64, 64, 64]          (36,864)
|    └─BatchNorm2d: 2-6                  [64, 64, 64, 64]          (128)
|    └─ReLU: 2-7                         [64, 64, 64, 64]          --
├─ResNet: 1-2                            [64, 64, 64, 64]          --
|    └─Identity: 2-8                     [64, 64, 64, 64]          --
|    └─Conv2d: 2-9                       [64, 64, 64, 64]          (36,864)
|    └─BatchNorm2d: 2-10                 [64, 64, 64, 64]          (128)
|    └─ReLU: 2-11                        [64, 64, 64, 64]          --
|    └─Conv2d: 2-12                      [64, 64, 64, 64]          (36,864)
|    └─BatchNorm2d: 2-13                 [64, 64, 64, 64]          (128)
|    └─ReLU: 2-14                        [64, 64, 64, 64]          --
├─MaxPool2d: 1-3                         [64, 64, 32, 32]          --
├─ResNet: 1-4                            [64, 64, 32, 32]          (recursive)
|    └─Identity: 2-15                    [64, 64, 32, 32]          --
|    └─Conv2d: 2-16                      [64, 64, 32, 32]          (recursive)
|    └─BatchNorm2d: 2-17                 [64, 64, 32, 32]          (recursive)
|    └─ReLU: 2-18                        [64, 64, 32, 32]          --
```

| └─Conv2d: 2-19 [64, 64, 32, 32] (recursive)

| └─BatchNorm2d: 2-20 [64, 64, 32, 32] (recursive)

| └─ReLU: 2-21 [64, 64, 32, 32] --

├─ResNet: 1-5 [64, 64, 32, 32] (recursive)

| └─Identity: 2-22 [64, 64, 32, 32] --

| └─Conv2d: 2-23 [64, 64, 32, 32] (recursive)

| └─BatchNorm2d: 2-24 [64, 64, 32, 32] (recursive)

| └─ReLU: 2-25 [64, 64, 32, 32] --

| └─Conv2d: 2-26 [64, 64, 32, 32] (recursive)

| └─BatchNorm2d: 2-27 [64, 64, 32, 32] (recursive)

| └─ReLU: 2-28 [64, 64, 32, 32] --

├─MaxPool2d: 1-6 [64, 64, 16, 16] --

├─ResNet: 1-7 [64, 128, 16, 16] --

| └─Sequential: 2-29 [64, 128, 16, 16] --

| | └─Conv2d: 3-3 [64, 128, 16, 16] (8,192)

| | └─BatchNorm2d: 3-4 [64, 128, 16, 16] (256)

| └─Conv2d: 2-30 [64, 128, 16, 16] (73,728)

| └─BatchNorm2d: 2-31 [64, 128, 16, 16] (256)

| └─ReLU: 2-32 [64, 128, 16, 16] --

| └─Conv2d: 2-33 [64, 128, 16, 16] (147,456)

| └─BatchNorm2d: 2-34 [64, 128, 16, 16] (256)

| └─ReLU: 2-35 [64, 128, 16, 16] --

├─ResNet: 1-8 [64, 128, 16, 16] --

| └Identity: 2-36 [64, 128, 16, 16] --

| └Conv2d: 2-37 [64, 128, 16, 16] (147,456)

| └BatchNorm2d: 2-38 [64, 128, 16, 16] (256)

| └ReLU: 2-39 [64, 128, 16, 16] --

| └Conv2d: 2-40 [64, 128, 16, 16] (147,456)

| └BatchNorm2d: 2-41 [64, 128, 16, 16] (256)

| └ReLU: 2-42 [64, 128, 16, 16] --

├─MaxPool2d: 1-9 [64, 128, 8, 8] --

├─ResNet: 1-10 [64, 128, 8, 8] (recursive)

| └Identity: 2-43 [64, 128, 8, 8] --

| └Conv2d: 2-44 [64, 128, 8, 8] (recursive)

| └BatchNorm2d: 2-45 [64, 128, 8, 8] (recursive)

| └ReLU: 2-46 [64, 128, 8, 8] --

| └Conv2d: 2-47 [64, 128, 8, 8] (recursive)

| └BatchNorm2d: 2-48 [64, 128, 8, 8] (recursive)

| └ReLU: 2-49 [64, 128, 8, 8] --

├─ResNet: 1-11 [64, 128, 8, 8] (recursive)

| └Identity: 2-50 [64, 128, 8, 8] --

| └Conv2d: 2-51 [64, 128, 8, 8] (recursive)

| └BatchNorm2d: 2-52 [64, 128, 8, 8] (recursive)

| └ReLU: 2-53 [64, 128, 8, 8] --

| └Conv2d: 2-54 [64, 128, 8, 8] (recursive)

| └BatchNorm2d: 2-55 [64, 128, 8, 8] (recursive)

│ └─ReLU: 2-56 [64, 128, 8, 8] --

├─Linear: 1-12 [64, 512] (4,194,816)

├─Linear: 1-13 [64, 8] (4,104)

==========

Total params: 6,213,424

Trainable params: 0

Non-trainable params: 6,213,424

Total mult-adds (G): 50.27

==========

Input size (MB): 2.10

Forward/backward pass size (MB): 1812.21

Params size (MB): 19.35

Estimated Total Size (MB): 1833.65

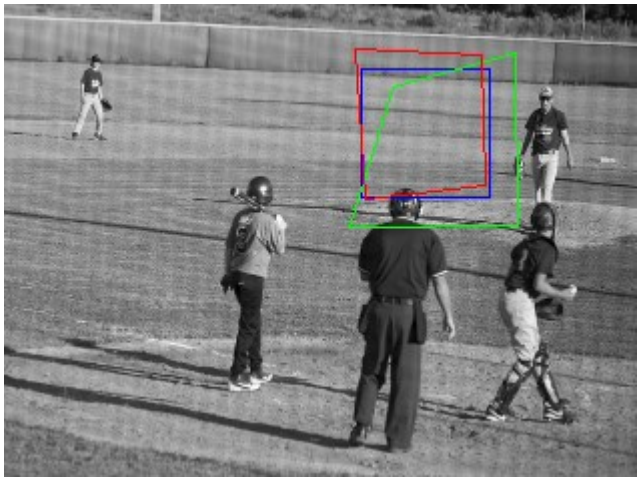
==========

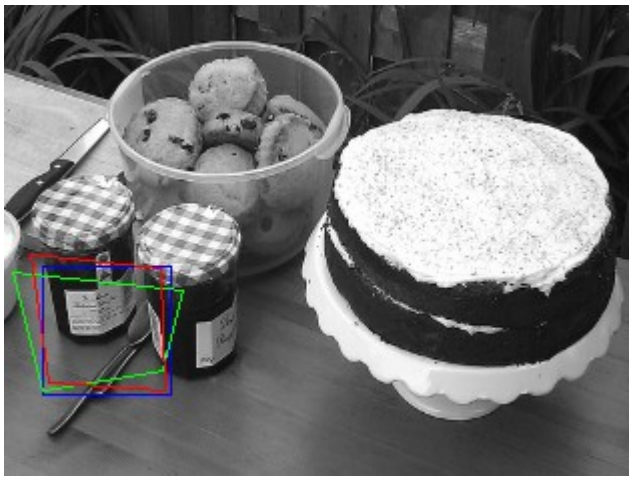
Učni primeri so transformirane sivinske slike katerih transformacijsko matriko iščemo kot ground trouth.

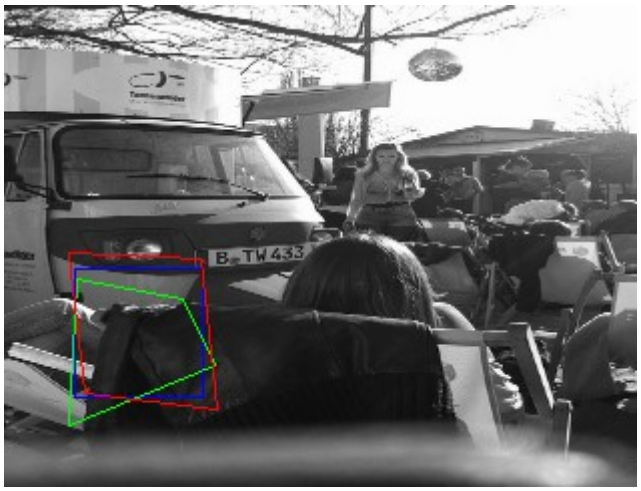
Originalna homologija

Transformirana homologija

Predvidevana homologija







Rezultati testiranja so pokazali da je pri učenju očitno prišlo do overfittinga, saj so predvidevanja dokaj natančna na traning setu, loss pa se nenadno poveča ko ga preizkusimo nad novimi podatki.

Končna MSE napaka nad regresisjsko glavo in učno množico: 0.03284

Končna MSE napaka nad regresisjsko glavo in testno množico: 0.17921