Detection-pipeline

Zhuang Liu

June 26, 2017

Contents

1 Face detection

We use the VIPL_FACE_SDK to detect the face in each frame of the rgb video.

1.1 Details

(1). The whole project is in *Detection/FaceDetection_Iso*. The project is a sample to show you how to get the face detection result of each video. The result will be saved in *Detection/FaceDetection_Iso/sample/FacePosition/IsoRGBtrainFacePosition.txt*. Each row is in the form of X X XXX XXX XXX XXX, which represents the video_id, which frame the face has been detected, tlx, tly, brx and bry.

(2). If you want to get all the result, just modify the 'filePath' and 'Trainlist' in the main.cpp. And also change the name of the saved txt.

(3). Note: The current project is in release mode, you can configure the debug version yourself refer to the release configuration.

(4). We simply evaluate the face detection time for all the videos, it will take about 4 hours to get all the results. We have already uploaded the detection result in github. All the files are compressed in the *IsofacePosition.zip*.

2 Hand detection

2.1 An introduction to the method

We use Faster R-CNN based method to detect the hand in the video. The method is called *two-stream Faster R-CNN*. Figure 1 shows the hand detection pipeline of two-stream Faster R-CNN.

2.2 Detection procedure

This section will show the detection procedure of our method, which include 'Pre-requirement', 'Training' and 'Detection'.

2.2.1 Pre-requirement

(1). Install Faster R-CNN from

`https://github.com/rbgirshick/py-faster-rcnn`

(2). Modify files to support two-stream Faster R-CNN. (This part will show in **Details** section)

2.2.2 Training

(1). We use VGG16 based model and the end2end training strategy mentioned in Faster R-CNN to train our hand detection model. Since our goal is

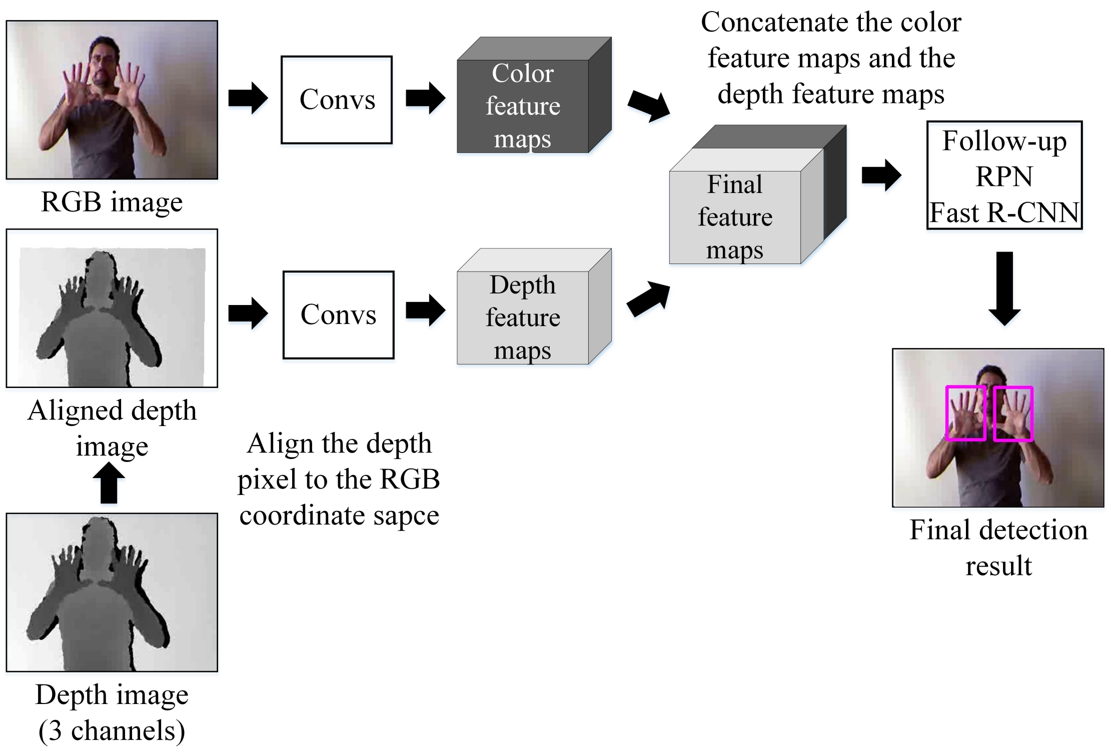

Figure 1: The detection pipeline of two-stream Faster R-CNN

just to detect hand in each frame, we change the network configuration files to match two classes (hand and background). (See .prototxt files in *models/pascal_voc/VGG16/faster_rcnn_end2end/rgbd_streams*)

(2). We use about 50,000 labeled frames from isolated RGB videos in training set to prepare the data according to the VOC2007 format.

(3). Data per-processing. To use the two-stream Faster R-CNN, we need to align the depth image to the RGB space first. (See project *CalibrationVideoDepth*, and this part will show in **Details** section)

(4). Train the model. The final model we get is:

vgg16_faster_rcnn_rgbd_streams.caffemodel .

Since the model is oversize for the github, we upload it in Baidu Netdisk. The url is:

`http://pan.baidu.com/s/1gftHQhH` code:8aac

2.2.3 Detection

(1). Data per-processing. For each depth video, we should align its frames to the corresponding rgb space.

(2). Use the vgg16_faster_rcnn_rgbd_streams.caffemodel to detect hand in rgb video, save the detection result in txt file.

For example:

For detecting the hand in M_00001.avi, we use M_00001.avi and the aligned K_00001.avi as input, then we get the result file which is named Label_M_00001.txt. In the result file, each row is in the form of XXXX XXX XXX XXX XXX ..., first XXXX means the frame number of the video, the followed XXX XXX XXX XXX (tlx, tly, brx, bry) means a candidate area of hand.

(3). Since we first aligned the depth video to rgb space, the label can also used in the aligned depth video to get depth feature.

2.3 Details

2.3.1 Pre-step: align depth video

We use camera calibration method to align the depth video to its corressponding rgb space. The whole project is in *CalibrationVideoDepth*. The project will create an executable file (.exe) in windows operating system. There is a sample in *CalibrationVideoDepth/sample*, just run the CalibrationVideoDepth.exe file. Result will be saved in *CalibrationVideoDepth/sample/CalibrationDepth*.

Note: The current project is in release mode, you can configure the debug version yourself refer to the release configuration.

Since this progress will take a lot of time. We have already uploaded the aligned depth video in Baidu Netdisk. The url is:

`http://pan.baidu.com/s/1skYyJd7` code:d7jx

2.3.2 Detection

(1). Install Faster R-CNN from

`https://github.com/rbgirshick/py-faster-rcnn`

(Suppose the path is *XX/py-faster-rcnn*)

Make sure run the *./tools/demo.py* successfully.

Make sure you have a 12G memory GPU (for run two-stream Faster R-CNN model).

(2). Copy the folders in *Detection/py-faster-rcnn-rgbd-streams* to your faster-rcnn path (which is *XX/py-faster-rcnn/* mentioned above). Replace the file if it already exists.

(a) rename your install 'py-faster-rcnn' to 'py-faster-rcnn-rgbd-streams'

(b) copy *Detection/py-faster-rcnn-rgbd-streams* to your faster-rcnn installed path (which is *XX* mentioned above). Replace the file if it already exists.

(c) download the model mentioned in subsection *Training*, and copy it to your *XX/py-faster-rcnn-rgbd-streams/data/faster_rcnn_model*.

(3). In *XX/py-faster-rcnn-rgbd-streams/tools/* there are some .py files, which means:

(a) demo_for_chalearn_rgb_show.py
A demo to show the result for using the two-stream model to detect hand in rgb video.
Usage:
There is a prepared video pair in your *XX/py-faster-rcnn-rgbd-streams/ChaLearn2017/sample* path.
$∼: cd XX/py-faster-rcnn-rgbd-streams
$∼/XX/py-faster-rcnn-rgbd-streams: python tools/demo_for_chalearn_rgb_show.py
The detection result will be saved in *XX/py-faster-rcnn-rgbd-streams/ChaLearn2017/sample*.

(b) chalearn_iso_rgb.py
Use two-stream model to detect hand in isolated rgb video. This script is used to detect hand for a batch videos. As mentioned before (in the pre-step, section 3.1), we have already aligned the depth video to rgb space. Since this detection process will take a lot of time, we suggest to use many GPUs to run the script.
Usage:
$∼: cd XX/py-faster-rcnn-rgbd-streams
$∼/XX/py-faster-rcnn-rgbd-streams: python tools/chalearn_iso_rgb .py --gpu 0 --path ./ChaLearn2007/IsoGD/ --dataset IsoGD_Phase_1 /train --begin 1 --end 1
--path: set the root path of IsoGD
--dataset: set the dataset (train/valid/test) of IsoGD
--begin: begin subfolder (001/002/...) of the dataset
--end: end subfolder (001/002/...) of the dataset
The result will be saved in the path of *./ChaLearn2017/IsoGD/DetectionLabel*

(4). After using *py-faster-rcnn-rgbd-streams* to detect hands in the videos, we can finally get the rgb results of each video.
For example: M_00001.avi $\longrightarrow$ Label_M_00001.txt

(5). We simply evaluate the detection time for all the videos, it will take about 3 days in a sigle Titan X GPU to get all the results. We have already uploaded the detection result in github. All the files are compressed in the *IsoHandLabel.rar*

3 Note: get all results

All samples mentioned above have just process a few of data, you should change the relative path to get the full results.

We have already uploaded the face and hand detection results in *IsofacePosition.zip* and *IsoHandLabel.rar*.

Another processing data from here is the aligned depth videos. We also uploaded all the aligned depth video in Baidu NetDisk, the url is:

`http://pan.baidu.com/s/1skYyJd7` code:d7jx