Exercise 2 – Retiming

188.346 Videoverarbeitung UE 2014W

1. Framework

In this exercise, the framework consists of the two sub-directories */bg_frames* and */src*. In */bg_frames* you find all extracted frames of the background video. The */src* directory includes the following files:

- exercise2.m
- get_inbetween_image.m
- get_opticalflow.m

You are supposed to complete this framework by implementing the tasks (3.a.-3.c.) described below. In order to accomplish this, you can find hints in the source files, such as this one:

```
%--------------------------------------------------------------------------
% Task a: Compute optical flow vectors
%--------------------------------------------------------------------------
```

You can start the framework by calling the function "exercise2" in the command window. For example: *exercise2('../bg_frames/','../output/','png');*

The framework automatically takes care of reading and saving the frames.

2. Submission

The deadline for exercise2 is **21.12.2014**

Your submission must include:

- the */src* directory including the commented matlab source files with the implemented tasks
- a pdf file with your answers on the three theoretical questions listed below
- the last 70 output frames in a directory */output*: this are the files frame00300.png to frame00369.png
  HINT: Don't delete the other output frames! You will need them in the third exercise!

All files have to be uploaded before the deadline as a .zip file (UE_GROUPx_EXERCISEy.zip) on TUWEL. Only one submission per group is needed.

To avoid any misunderstandings, the structure of the submission should look like this:

UE_GROUPx_EXERCISEy.zip
/output
frame00300.png

frame00301.png
…
frame003069.png

/src

exercise2.m
get_inbetween_image.m
get_opticalflow.m

UE_GROUPx_EXERCISEy_theory.pdf

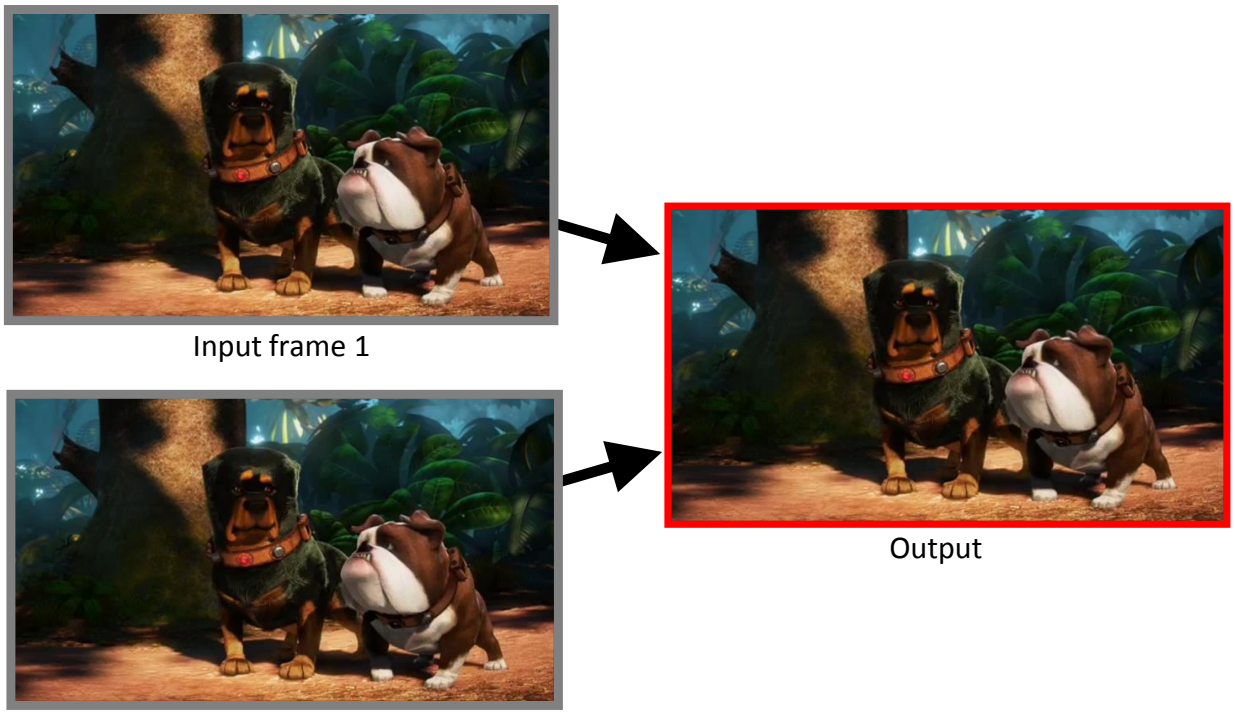
You will get points for:

- your implementation + **meaningful comments**
- answers on theoretical questions
- output images

NOTE: Please make sure that your submission includes ALL required files because points will be given only on the submitted files!

NOTE: Please make sure that your debugging code lines, like *"figure"*, *"imshow"* or "*subplot*" are not enabled in your final solution! Otherwise you will loose points.

3. General Task



In the second exercise, you will create additional frames to synchronize the playing time of a foreground and background video. The foreground video consists of 369 frames, the background has just 246 frames. It is therefore necessary to add new frames to the background in order to result in an equal playing time. Within this task you have to calculate the optical flow between two consecutive frames and with this motion vectors you can generate one new frame that reflects the motion between the two original frames. Therefore, you have to complete the optical flow function and compute new frames with the resulting flow vectors.

Task a. Compute optical flow vectors

Implementation in: get_opticalflow.m, exercise2.m (call function *"get_opticalflow")*
Input for this task: two consecutive frames
Output of this task: flow vectors

The principal aim of the optical flow estimation is to compute an approximation of the motion field of a video sequence. For this you have to call the function *"get_opticalflow"* with two consecutive gray-scaled frames and meaningful values for the parameters "*alpha*" and "*iterations*". The function *"get_opticalflow"* should estimate the optical flow vectors on the basis of the method by Horn & Schunck. To complete this function, you have to compute the flow vectors constrained by its local average and the optical flow constraints. You can find the information on how to calculate these vectors on the "optical flow" slides of the lecture part. After this calculation, you are asked to make an additional median filtering on

the two flow vectors to improve the estimation result in each iteration. You can use the MATLAB function "medfilt2" to perform the median filtering.

Task b. Generate new x- and y-values of new frame

Implementation in: get_inbetween_image.m, exercise2.m (call function *"get_ inbetween_image")*
Input for this task: first frame of the two consecutive frames, the computed flow vectors from task a.
Output of this task: new x- and y-values

One of the two computed vectors contains the pixel offsets between two frames in x-direction and the other one the offsets in y-direction. Thanks to these two vectors you know the motion offset of two consecutive frames for each pixel. If you take just half of every offset of each pixel you can create a new frame which fits between the two original frames. To compute the new x- and y-values you should take a closer look at the MATLAB function *"meshgrid"*. The resulting arrays of this function can be used to add the offsets.

HINT: Make sure that the new values don't exceed the size of the frame.

Task c. Generate new frame

Implementation in: get_inbetween_image.m, exercise2.m (call function *"get_ inbetween_image")*
Input for this task: first frame of the two consecutive frames, new x- and y-values
Output of this task: new generated image

With the new x- and y-values for each pixel we can interpolate the color, which should be used for this pixel. To compute this you can use the MATLAB function *"interp2"* for each RGB channel.

Theoretical Questions

a. What means “filling-in” in the context of the method by Horn & Schunck and what are the benefits of this effect?

b. What would be the result if you add the full motion vectors instead of half of them?

c. Which line of code computes the first derivation in “opticalflow.m”? And what is the meaning of the chosen parameter values?