# Exercise 2 – Retiming

188.346 Videoverarbeitung UE 2016W

## 1. Framework

In this exercise, the framework consists of the two sub-directories */fg_frames* and */src*. In */fg_frames* you find all extracted frames of the foreground 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.e.) described below. In order to accomplish this, you can find hints in the source files, such as this one:

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

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

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

## 2. Submission

The deadline for exercise2 is **9.12.2015, 23:55.**

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 60 output frames in a directory */output*: this are the files frame0069.png to frame00129.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.

The structure of the submission should look like this:

**UE_GROUPx_EXERCISEy.zip**

- **/output**
  - frame0069.png
  - frame0070.png
  - …
  - frame00129.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! Also please make sure that your debugging code lines, like *"figure"*, *"imshow"* or "*subplot*" are not enabled in your final solution! Otherwise you will lose points.**

## 3. General Task

Input frames 11 and 12

Output frame

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 43 frames, the background has 129 frames. It is therefore necessary to add new frames to the foreground in order to result in an equal playing time. Within this task you duplicate the foreground frames so that the foreground video is played two times. Then 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: Duplicate video sequence

Implementation in: exercise2.m
Input for this task: foreground frames
Output of this task: duplicate foreground frames

In the first task you have to duplicate the foreground video frames so that the sequence appears twice in the final video. This means you have to copy each frame and name it so that the original naming concept is continued. This means your foreground image sequence should look like this:

**/fg_frames**
frame0043.png
frame0044.png --- copy of frame0001
frame0045.png --- copy of frame0002
...
frame0086.png --- copy of frame0043

Of course this must be done through the implemented code and not manually. Therefore the commands "sprintf" and "copyfile" could be useful. The new frames should be saved in the fg_frames directory so that they can be used in the next steps. If task a is completed, you can put the code in a comment block so that the frame-copying will not be executed when implementing the tasks b-d. In the final submission however the code must be uncommented.

**Task b: 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 c: 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 d: 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.

**Task e: Generate foreground map for new foreground video**

After you completed the tasks a-d of this exercise you have 129 foreground frames which should appear in the final video. But therefore we need the foreground maps for each of those frames. This is made by copying the output frames of this exercise in the fg_frames-folder of your solution of exercise 1 and run the framework again. You can use the reference frames with the scribbles from the first exercise for this step. You don't have to add the resulting foreground maps to your submission because exercise 1 has already been graded, but you will need them for the final exercise.

## Theoretical Questions

a. What is the goal of Optical Flow Estimation, and for which applications can it be used for?

b. Which assumption is causing problems when Block-Matching techniques are used for the Optical Flow Estimation?

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

d. What does it mean if the Data Term in the Horn & Schunck Energy Function is very large?