Volkan OKBAY
1741479

EE636 Project User Manual
Tracking in High Density Crowd

This manual consists of parameter explanations, how to run project, folder content of submitted disk, simulation system features and short description of work done after preliminary report.

Folder Content

The submitted files include:

- Two MATLAB .m scripts, namely initialize.m and real_time.m,
- Sample video in \Vid subfolder,
- Necessary functions introduced in late versions of MATLAB (\Aux_func)
- The main followed paper and some of the important reference papers in \Document folder
- Preliminary report submitted earlier located in \Document folder
- Finally, this manual file. (OpenOffice or PDF extension)

Code Usage

To run, open 'initialize.m' and 'real_time.m' with MATLAB (at least 2015b ver). Copy a video file under Vid subfolder, in case it is desired to try a different file other than sample video. Enter parameters and the name of this file with its expansion to 'video_file' parameter (see below). Then, simply, run 'initialize.m' script. This will take some moment between app. 30 sec and 1.5 mins.

When 'DONE' text appers initialization would be completed. Then user may move to 'real_time.m' and run it. This action will take some time before opening a new window. Actually, it is a GUI, including a draggable point on the first frame of the video. Drag and drop the given point on an object to be traced. When in correct position, double-click on the point and observe the tracking phase on a second figure which pops-up immediately.

To select another object, cancel the current process or wait for it to finish. Run 'real_time.m' again and select next object point. On the second run, GUI will open much faster as it is already initialized.

NOTE: In order to trace the algorithm or to debug, user may see comments in the design files, visualize steps (see commented out code lines) and run sections by ctrl+Enter combination. Section trace and visualising 3-D plots (especially 'maraton.mov' video) will help user to grasp Floor Field concepts better. (see VISUALIZE sections in code scripts)

NOTE 2: Please note that, all the timing intervals are given according to simulation on below system.

Nvidia GeForce GTX980M graphic card with 8 GB memory
16 GB RAM (1866 Mhz)
Intel(R) Core i7-4710MQ CPU (2.50 Ghz)
MATLAB 2015b on Windows 10 (x64)

Parameters

There two .m files; as a result, two clusters of parameters. One is implemented once for each video file; while other regards real time process that can be altered before each tracking operation.

1-) *Initial Parameters:* Those are located in the beginning of initial.m file. Implemented once for each video file.

```
%----------% Main parameter
video_file = 'maraton.mov'; % Full name of the video file located in either
project folder or in \Vid subfolder
%----------% SFF: Optical Flow
frameCount = 20; % First N frames to consider in SFF calculation (def: 20)
frameNumber_average = 5; % To eliminate first N frames due to camera calibration
distortions in the beginning (def: 5)
%----------% SFF: Sink-seek
block_y = 10; % Grid size in y (for faster implementation) (def: 10)
block_x = 10; % Grid size in x (for faster implementation) (def: 10)
bwarea_para = 3000; % Area threshold (to get edge map masking) (def: 3000
pixels, pixels in a frame/100)
h = 10; % Default value for h (see paper, Kernel Density formula) (def: 10)
window_size = 3; % Kernel density window size (see paper, Kernel Density
formula) (def: 3)
sink_step_limit = 500; % Maximum step size (prevent infinite sink steps) (def:
500 steps)
%----------% BFF
bwarea_para_BFF = 3000; % Thresholding optical flow field to find boundaries and
barriers (def:3000 pixels, typically same with bwarea_para)
% Note that, for further adjustments, two strels under BFF section can be
% changed in order to debug BFF or to test BFF in different cases.
```

2-) *Real Time Parameters:* Those are located in the beginning of real_time.m file. Implemented once for each object path tracing sequence.

```
real_time_duration = maraton.Duration*maraton.FrameRate; % Total frame count in
the video file (def: maraton.Duration*maraton.FrameRate, video duration)
%----------% DFF
frameCountDFF = 10; % Time window for DFF (def: 10)
osize = [3 7]; % Object size (def:3 7 - 6x15 window)
window_size = 8; % Search window for next position (def:8 - 17x17 window)
DFF_factor = 5; % Gain of effect of DFF on the next position vector (def: 5)
%----------% Main probabilistic equation (see the main equation in the paper)
Ks = 0.02; % Weight of SFF (def: 0.2)
Kb = 0.02; % Weight of BFF (def: 0.2)
Kd = 0.2; % Weight of DFF (def: 0.2)
Kr = 10; % Weight of R (def: 1, not mentioned in the paper)
```

Work Done (After Preliminary Report)

In the end, Dynamic Floor Field, similarity matching and final probabilistic equations are implemented. The code is divided into two parts working initially and real time. With a GUI, object selection is eased. Parameters are adjusted to some anchor values.

To overcome difficulty in DFF calculation, instead of advection solution, the average of optical flow fields in time window and in spatial neighborhood is taken. On the tip of average vector resulted from the previous step, a Gaussian distribution is place 10x10 size and 5 deviation factor (see imfilter function for detail). The matrix produced becomes our DFF.