Summary of the takeover readiness code

Shreyas Bhat

May 2020

1 Introduction

This document presents an overview of the code to collect data and process it for computing eye-tracking and GSR metrics, for offline model training and online prediction of driver's readiness to takeover control from an autonomous vehicle when it reaches its limit. The sections below describe the code files and how to run them.

2 Offline metrics computation

The workflow for the offline metrics computation is as follows:

1. Run the map_and_send.py file on the iMotions computer
2. Ask the participant to focus their gaze on the calibration marker to calibrate the glasses
3. Check the calibration (mapping). If good enough, press 'y', else press 'ctrl-c' and repeat from beginning
4. Start recording on iMotions (make sure it is connected to Tobii wifi)
5. Ignore the calibration messages on iMotions
6. Export the sensor data from iMotions after the trial is complete
7. Use the offline metrics code files to compute and store the metrics for model training

The details of the code files involved in the offline metrics computations are present in the subsections below

2.1 map_and_send.py

This file maps the 3d coordinates given by Tobii to a background image of the setup. It then assigns an AOI to each gaze point and sends over the data to iMotions over UDP. The data transmission rate is 30 Hz, the same as the fps of the installed webcam.

The following things may need to be changed in the code according to the setup:

- Background image
- Number of apriltags in the background image
- Definition of AOIs in the background image
- IP address of the iMotions computer

2.2 offline_metrics_from_txt.py

This code file computes the following metrics from the data file exported by iMotions:

- Number and duration of fixations in each AOI
- Number and duration of saccades in each AOI
- Number of blinks in each AOI
- Gaze duration in each AOI
- Average pupil diameter in each AOI
- Gaze dispersion in x- and y- directions in meters
- Scan pattern between AOIs
- GSR indices - mean, max and standard deviation of GSR in phasic component
- HR indices - mean, min, max and standard deviation of heart rate
- IBI indices - mean, min, max and standard deviation of inter-beat interval
- Number of GSR peaks and peak rise times

It then saves the metrics computed in each time window as a python list in a file which can later be loaded for training the model

The following variables may need to be changed in the file during training:

- Filename of the data file
- Filename for the file where the metrics should be saved

- Time window width
- the interval by which the time window slides between each iteration
- Velocity threshold for classifying fixations
- The duration for identifying blinks
- The corresponding data column numbers in the data file
- The pixels to meters variable, which is used to calculate the distance between mapped gaze points in the background image
- The filepath for saving and loading the data.mat file for Ledalab analysis

The code for training the model would be written later.

2.3 load_pickle.py

An example code file for loading the metrics data from the saved file

3 Online metrics computation

A Linux environment with ROS installed is required for this part.

The ROS framework is described by the figure below:

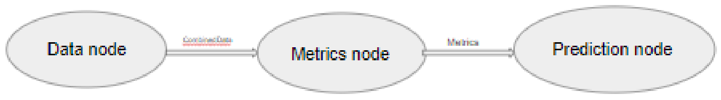

Figure 1: The ROS framework in the online computation of metrics

The network works in the following way:

- The data node collects data continuously and sends a window of data over the CombinedData topic
- The metrics node computes the gsr and eye-tracking metrics whenever new data is published on the combinedData topic and publishes it on the Metrics topic
- The prediction node can predict the driver's readiness based on the metrics on the Metrics channel. It can also play a sound to alert the driver to be ready if they are not

All the metrics described in the offline metrics computation section are calculated in real-time for the online prediction task. The sliding window width can be specified in the code, as can be the frequency of predictions.

The following are the steps for running the ROS framework:

1. run catkin_make in the catkin_ws folder if running for the first time
2. In a terminal run 'roscore' to start the master ros server
3. Open 3 terminals in the catkin_ws folder. Run 'source ./devel/setup.bash' in all
4. Make sure the system is connected to the Tobii WiFi
5. In one of the terminals, run 'rosrun takeover_readiness data_node.py'
6. In another terminal, run 'rosrun takeover_readiness metrics_computation_node.py'
7. In another terminal, run 'rosrun takeover_readiness prediction_node.py'
8. In the iMotions computer, run 'python3 map_and_send.py' and do the calibration procedure
9. Start the iMotions recording

The sections below describe the files associated with the ROS framework

3.1 data_node.py

This file collects data for a specified time window from iMotions in real-time. It then sends this data over the combinedData topic. It continously collects data and the frequncy of sending the data can be adjusted through the sliding width variable.

Data collected:

- Timestamp of the eye data point
- 3d gaze coordinate
- 2d mapped coordinates on the background image
- AOI
- Pupil diameter
- Left eye pupil position
- Right eye pupil position
- Timestamp of the shimmer data point
- GSR (micro siemens)
- Heart Rate
- Inter-beat interval

3.2 metrics_computation_node.py

This node subscribes to the CombinedData topic. Whenever new data is published, it computes all the metrics corresponding to that window of data and publishes them on the Metrics topic

The Metrics topic contains the following data:

- The timestamp of the first eye-tracker data point in the window
- Number and duration of fixations in each AOI
- Number and duration of saccades in each AOI
- Duration of gaze in each AOI
- Number of blinks in each AOI
- Average pupil diameter in each AOI
- Gaze dispersion in x- and y- directions
- Scan pattern between AOIs
- The timestamp of the first shimmer data point in the window
- GSR Indices
- HR Indices
- IBI Indices
- GSR Peaks
- GSR peak rise times

3.3 prediction_node.py

This node subscribes to the Metrics topic. Whenever new data is available on that topic, a prediction can be made using that data. The driver could be alerted that he/she is not ready to takeover using the display on the secondary screen and/or with an audio message.

4 Metrics computation details

The following sections describe the method for computing all the metrics stated above

4.1 Eye-tracking metrics

The following sections describe the method used to calculate the eye-tracking related metrics

4.1.1 Fixations and saccades

An implementation of the Tobii I-VT filter is used to classify gaze points as fixations. The algorithm works in the following way:

1. Each gaze point is compared to its previous gaze point. The 3d gaze point location vectors are used to calculate the gaze angle between the two points. The velocity is calculated using this angle and the difference in timestamps between these two points. If this velocity is less than a set threshold (30 deg/sec), this gaze point is classified as a fixation and it is added to a list representing the current fixation. If a saccade is encountered in between, this list is treated as one fixation and the duration of the fixation even is measured. If this duration is long enough (¿60 ms), the number of fixations in the AOI corresponding to the gaze points is incremented by one. The fixation duration in that AOI is incremented by the duration of this fixation event.

2. Each gaze point not classified as belonging to a fixation is classified as a saccade. This gaze point is then added to a list representing the current saccade. Once a fixation is detected again, this list is used to increment the saccade durations and the saccade number in the corresponding AOI is incremented by one.

3. If the fixation even duration is less than 60ms, it is classified as being neither a fixation nor a saccade. This is the same as the discard short fixations function in the Tobii I-VT filter.

4.1.2 Gaze duration

The AOI of each gaze point is checked and the time difference between this gaze point and the previous gaze point is added to the gaze duration of this AOI.

4.1.3 Blinks

If the eyes are not found by the tobii glasses, we get the same timestamp from the sensor until it can again detect the eye. If such an event occurs, the timestamp is added to a list until the eyes are detected again. If the time difference between the first timestamp when the eyes are detected again and the first timestamp when eyes were not detected is more than 75ms, the number of blinks in the last computed AOI is increased by one. This is the value used by Tobii in their implementation of the I-VT filter.

4.1.4 Average pupil diameter

The average of the pupil diameter is calculated over the time window in each AOI

4.1.5 Gaze dispersion

The standard deviation of the mapped gaze points over the time window along the x- and y- directions. The value is first computed in pixels and then can be converted to meters by using the apriltags and the distance between them in the image.

4.1.6 Scan pattern

The probability of shifting from one AOI to another. This is calculated by the ratio of switches from one AOI to another in the time window.

4.2 GSR metrics

The raw data from the Shimmer sensor is smoothed using the Breaking Point function. It is then stored in a 'data.mat' file which is then processed using the Ledalab software in MATLAB. The following sections describe the methods used for computing the GSR metrics:

4.2.1 GSR Indices

The analysed result from Ledalab contains the phasic and tonic components of the GSR signal. The GSR Indices are the maximum, mean, and standard deviation of the phasic component in the time window

4.2.2 HR Indices

The raw data from Shimmer contains heart rate data. The HR indices are the minimum, maximum, mean, and standard deviation of the heart rate in the given window.

4.2.3 IBI Indices

The raw data from Shimmer contains data about the inter-beat interval. The IBI indices are the minimum, maximum, mean, and standard deviation of the IBI in the given window.

4.2.4 GSR Peaks and peak rise times

The analysed result from Ledalab have information on GSR peaks in the given data. The number of peaks are calculated from the length of the peak times variable. The peak rise times are calculated as the difference between the peak times and the onset variables.