Lehrstuhl für Medieninformatik
Institut für Information und Medien, Sprache und Kultur
Universität Regensburg

Interaction Technology and Techniques Assignment 9: Recognizing Gaits Using a WiiMote and a Machine Learning Classifier

Summer semester 2014

Submission due: Sunday, 13. July 2014, 23:55

Hand in in groups of max. two.

Your task is to implement a pipeline for analyzing the sensor data of the WiiMote and recognizing certain movements.

9.1: Test a machine learning classifier on movement data

Download `Wiimote - FFT - SVM.ipynb` from GRIPS. Use this notebook as a blueprint for your own experiment:

- collect accelerometer data for at least three different categories (e.g., walking, standing, sitting, running, swinging the WiiMote like a sword, using it as a drumstick, …). For each category, collect at least twelve samples (alltogether) from at least 2 persons.
- modify the notebook so that the raw data from all three categories is preprocessed appropriately
- use two thirds of the data for training a machine learning classifier (e.g. an SVM classifier).
- evaluate how well the classifier works by testing it on the remaining third of samples you collected.
- document the results of your evaluation in the notebook (in a normal text field).
- collect all raw data files and the notebook in a .zip file.

Hand in the following file:

ml-test.zip: a zip archive containing the iPython notebook and accompanying raw data.

Points

- **1** The solution has been submitted, is not empty, and does not print out error messages.
- **1** The classification categories are chosen well.
- **1** The notebook can be executed without errors.
- **1** The notebook is well structured and documented
- **1** The classifier is able to correctly classify a majority of test samples.

9.2: A custom PyQtGraph flowchart node for the classifier

Implement the pipeline from Assignment 9.1 as a real-time PyQtGraph flowchart. Read the documentation for `wiimote.py` in GRIPS and the PyQtGraph documentation[1]. Write a small Python application `classify.py` that takes a Bluetooth MAC address as its only parameter and automatically connects to this Wiimote. This application should generate a PyQtGraph flowchart with the following elements:

[1]http://pyqtgraph.org/documentation/

- a WiiMoteNode
- a BufferNode (as described in previous assignments)
- a FileReaderNode that reads data from a file and outputs it as arrays. The file name should be configurable in a control widget but have a default value (see e.g. the BT address field in the WiiMoteNode). Files are read once when the node is instantiated *and* whenever the *read file* button in the control widget is clicked.
- an SvmClassifierNode implementing an SVM classifier. It receives training data and categories (from the FileReaderNode) on one terminal and data to classify on another terminal. It also has an output terminal that outputs to which category the input data belongs.
- a CategoryVisualizerNode that shows the output of the SVM classifier node either as a graphical image or as text (i.e., not just the number of the category).
- further nodes needed for transforming input data, e.g. an FFT node, etc.

You may split the classes into several files. You need to choose a reasonable file format for the classification data and document this format in the source code. The classifier pipeline should work in (more or less) realtime on startup without further intervention.

Hand in the following file:

classifier.zip: a .zip archive containing all necessary files to implements this flowchart (including the files with the training data and a short description of the movements that can be distinguished)

Points

- **1** The solution has been submitted, is not empty, and does not print out error messages.
- **2** The script correctly implements and displays a flowchart.
- **2** The FileReaderNode correctly reads files data and outputs it as arrays or dicts.
- **2** The SvmClassifierNode correctly classifies WiiMote movements.
- **3** The pipeline is modular, i.e. using individual nodes for each important processing step.
- **1** The script is well-structured and follows the Python style guide (PEP 8).

Submission

Submit via GRIPS until the deadline

All files should use UTF-8 encoding and Unix line breaks. Python files should use spaces instead of tabs.

`Have Fun!`