README file for the Teleoperation System

Jan Ondras

May 19, 2017

1 Project overview

In this Part II Project on *Replicating Human Facial Emotions and Head Movements on a Robot Avatar*, I developed a novel automatic teleoperation system. This system can replicate facial expressions of emotions (namely, neutral, disgust, happiness, surprise) and head movements on the humanoid robot Nao in real-time.

The system consists of two independent subsystems:

- Emotion replication subsystem – Action Unit Detector (AUD) and Emotion Classifier (EC)
- Head pose replication subsystem – Head Pose Detector (HPD) and Head Pose Filter (HPF)

So there are two main folders `EmotionReplication` and `HeadPoseReplication` that contain the system files associated with these subsystems.

For details about project files (both modified and newly created) see Section 5.

2 Requirements

2.1 System

The following system requirements have to be met:

- Ubuntu Linux OS 14.04
- GCC version 4.8.4
- OpenCV library version 2.4.8
- Aldebaran Python SDK
- NaoQi and Choregraphe (optional, necessary only when simulating on virtual robot)
- Two web cameras.

2.2 Models

AUD and HPD require large files of models (far beyond 15 MB limit) that are not included in this package. Therefore, in order to run the system you need to download a .zip file from `https://www.cl.cam.ac.uk/~hg410/files/research/Part-II/JOndras/m.zip`, unpack it and place the content of `AUDmodels` folder into `./EmotionReplication/data/`, and the content of `HPDmodels` folder into `./HeadPoseReplication/models`.

2.3 Robot

The Aldebaran Nao robot is required. The system was tested with NaoQi version 2.1, head version 4.0 and body version 25.

3 Configuration

All the running modes can be set and configured at the beginning of `runAll.sh` file. In particular, there are following options:

1. **Displaying mode**
   If you want to use simulated robot in Choregraphe application, set `DISPLAY_MODE=VIRTUAL` and corresponding IP address and port `IP_PORT_VIRTUAL`.
   By default `IP_PORT_VIRTUAL=127.0.0.1:9559`.
   If you use real robot, set `DISPLAY_MODE=REAL` and `IP_PORT_REAL`.
   By default `IP_PORT_REAL=169.254.42.173:9559`.
2. **Rebuild emotion replication subsystem**
   To rebuild Action Unit Detector set `REBUILD_AUD=true` (default) or `false` otherwise.
3. **Rebuild head pose replication subsystem**
   To rebuild Head Pose Detector set `REBUILD_HPD=true` (default) or `false` (default) otherwise.
4. **Run emotion replication subsystem**
   To run emotion replication subsystem set `RUN_AUD=true` (default) or `false` otherwise.
5. **Run head pose replication subsystem**
   To run head pose replication subsystem set `RUN_HPD=true` (default) or `false` otherwise.
6. **Video stream input for emotion replication**
   To run emotion replication subsystem using web-camera (i.e. in online/live mode) set `AUD_LIVE=true` (default) or `false` otherwise. If set to `false`, the system will use the video file specified in `VIDEO_IN_AUD`.
7. **Video stream input for head pose replication**
   To run head pose replication subsystem using web-camera (i.e. in online/live mode) set `HPD_LIVE=true` (default) or `false` otherwise. If set to `false`, the system will use the video file specified in `VIDEO_IN_HPD`.

Finally, for simulating on virtual robot, it is necessary to set the paths to NaoQi and Choregraphe. These are set in `NAOQI_PATH` and `CHOREGRAPHE_PATH` respectively.

4 Running

The whole system is run by executing `./runAll.sh` on command line. Each subsystem can be terminated individually by pressing ESC key – the window of the subsystem you wish to close must have the focus.

5 Project files – details

Files of AUD (in `./EmotionReplication/`) modified by me:

- `AURecogniser.cpp` – calls Python functions of HPF (pass detected action units).
- `AURecogniser.hpp`
- `livefeatures.cpp` – modified input processing.
- `AUDetector.pro` – included additional libraries and settings.

Files of HPD (in `./HeadPoseReplication/`) modified by me:

- `/src/DemoTracker.cpp` – calls Python functions of HPF (pass measured angles and time step).
- `Makefile` – included additional libraries and settings.

New files created:

- For real-time replication:
  - `./EmotionReplication/AUDdisplay.py` – inference of emotions based on detected action units from AUD and sending LED commands to the robot.
  - `./EmotionReplication/weights.npz` - trained weights.
  - `./HeadPoseReplication/HPDdisplay.py` – filtering of head movements measured by HPD and sending head movement commands to the robot.
  - `./HeadPoseReplication/KFClass.py` – Kalman filter class.
- For preparation, pre-processing, training and evaluation. Please, see `ABOUT.txt` files in corresponding subdirectories for details.
  - Scripts for data processing and extraction from databases: `./OtherProjectFiles/DataProcessing/`
  - Emotion classifier training & cross-validation: `./OtherProjectFiles/EmotionClassifierTraining/`
  - Scripts for evaluation of Emotion Classifier and whole system evaluation (web-survey): `./OtherProjectFiles/UserStudyEvaluation/`
  - Scripts for estimating measurement and process noises (of Kalman filter), and Head Pose Filter evaluation, and comparison with FilterPy: `./OtherProjectFiles/HeadPoseFilterEvaluation/`

- Script for evaluation of replication latency measurements:
  `./OtherProjectFiles/LatencyEvaluation/`
- Basic robot testing (head movements and LEDs):
  `./OtherProjectFiles/RobotTest/`