Ubiquitous and Mobile Computing CS 528:Visage: A Face Interpretation Engine for Smartphone Applications

Amogh Raghunath

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

Mateus Amarante Araujo

Robotics Engineering Dept.

Worcester Polytechnic Institute (WPI)

Outline

- Introduction
- Related Work
- Design
- Architecture
- Implementation
- Evaluation
- Visage Applications
- Conclusion

Introduction

- Smart phones are embedded with sensors
- Users are increasingly using applications
  - Tweeting, Web surfing, texting
- Camera, capable of observing users as they interact with different application

Introduction

- Visage: A robust, real-time face interpretation engine for smart phones
  - Tracking user’s 3D head poses & facial expression
  - Fuse data from front-facing camera & motion sensor

Related Work

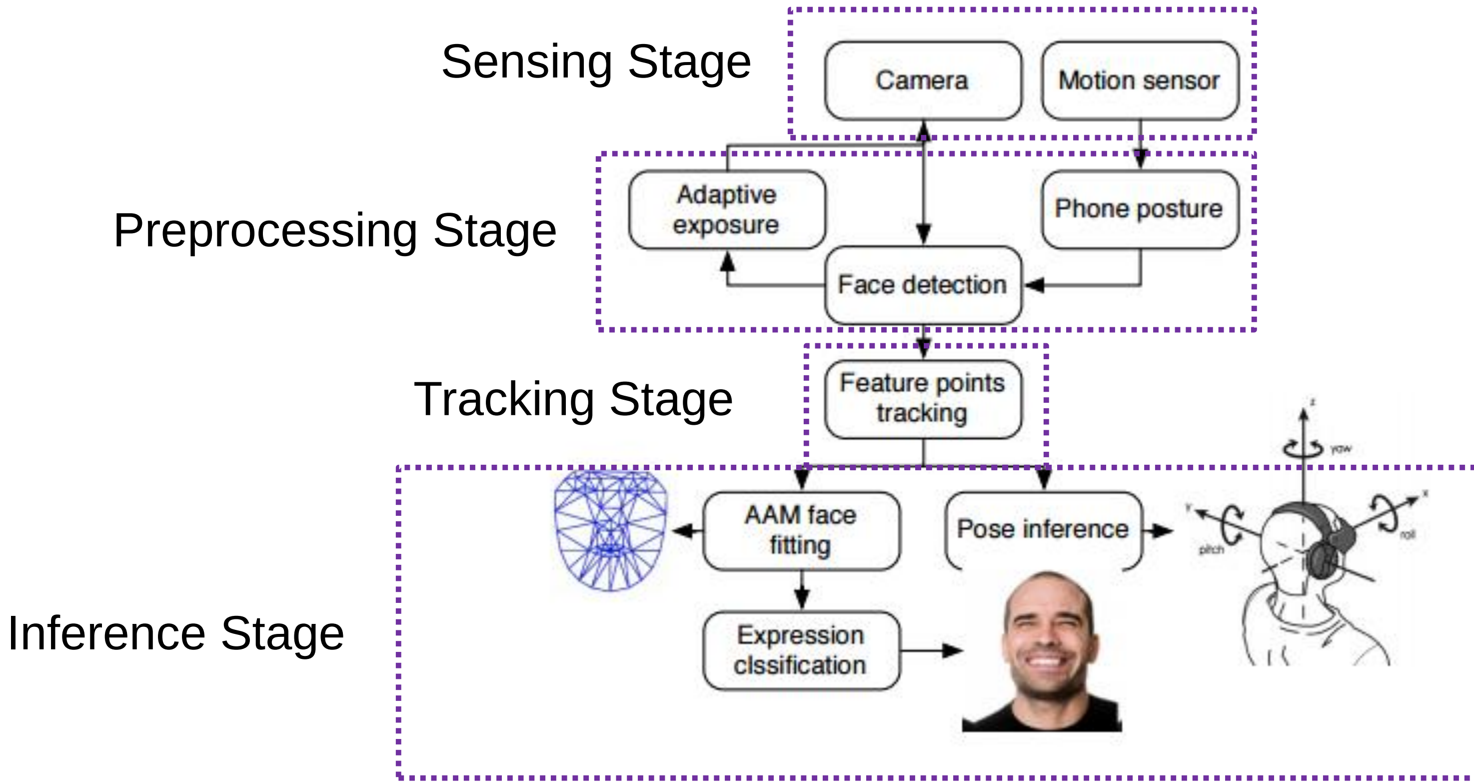
- Involves limited image processing
  - SenseCam
  - Recognizr
  - MoVi
- Simple tracking of 2D face representations
  - PEYE
- Visage: A robust, real-time face interpretation engine for smart phones

Design

Challenges

- User Mobility
  - Movement of the phone cause low image quality
  - Analyze exposure level of face region
- Limited Phone Resources
  - Operate in real-time

Architecture



Architecture

Sensing Stage

- Captures the video stream from the phone’s front-facing camera
- Raw motion data from accelerometer and gyro sensors on the phone.

Architecture
Preprocessing Stage

1. Phone posture component
2. Face detection with tilt compensation
3. Adaptive exposure component

Architecture
Preprocessing Stage

Phone posture component

- Identifies frames which contain user's face and monitors the phone posture
- Raw readings from accelerometer and gyroscope and estimates of direction of gravity
- Calculates mean and variance on each direction
- Gravity direction – mean of accelerometer data

Architecture

Preprocessing Stage

Face detection with tilt compensation

- AdaBoost object detector with tilt correction

$$\theta_g = \frac{180}{\pi} arctan \frac{a_x}{a_y}$$

- Image is tilted by:

$$I_r = \begin{bmatrix} cos\theta_g & -sin\theta_g \\ sin\theta_g & cos\theta_g \end{bmatrix} I_i$$

Architecture

Preprocessing Stage

Adaptive exposure component

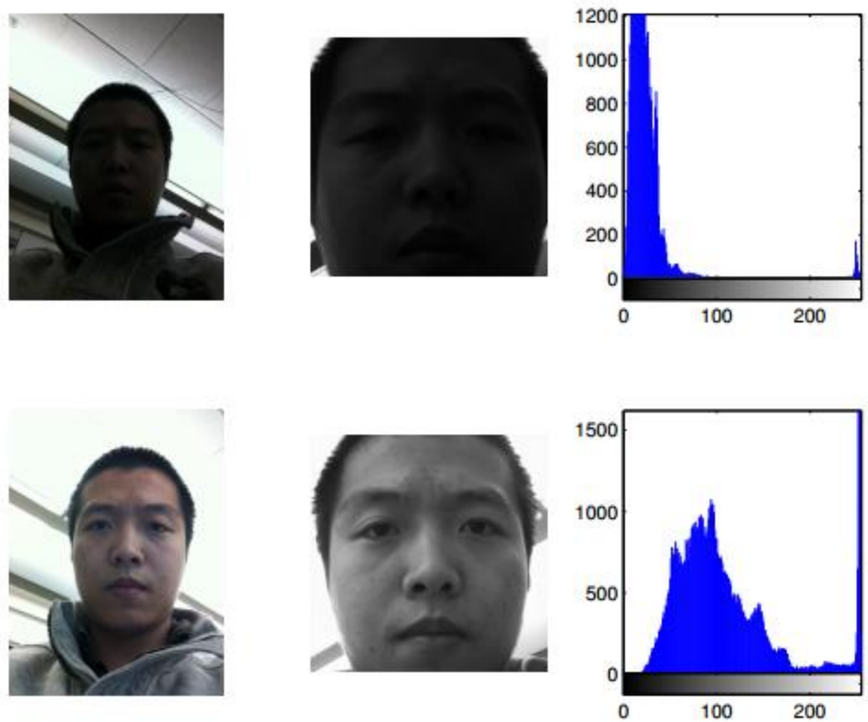
- A clear face region is critical for tracking and inference
- Visage uses the local lighting information within the detected face region to correct the camera hardware exposure level.
- Exposure level by computing the centroid of Hface:

$$C_{H_{face}} = \frac{\sum_{i=0}^{255} iH_{face}(i)}{\sum_{i=0}^{255} i}$$

Architecture

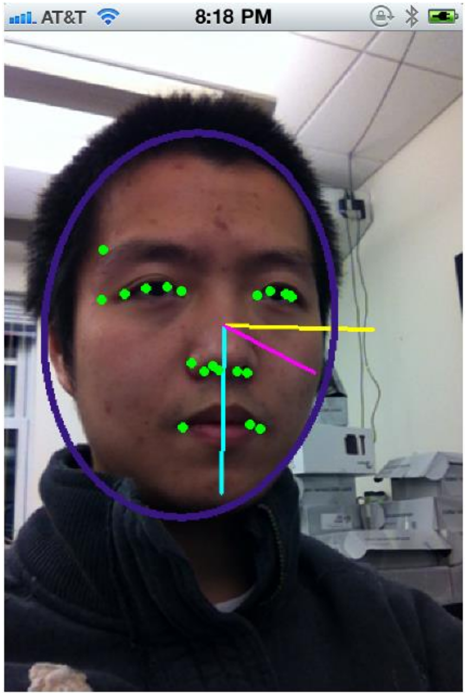
Preprocessing Stage

Adaptive exposure component



Architecture
Tracking Stage

- Feature Points Tracking Component
  - Select feature points (e.g. eye corners and edges of mouth): they are stable across frames

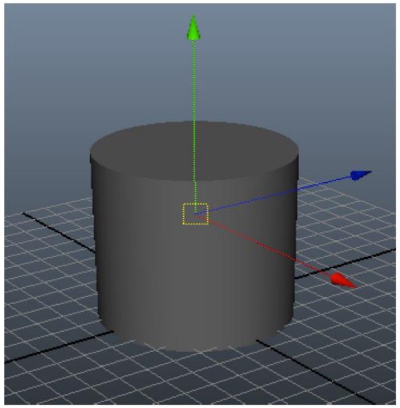


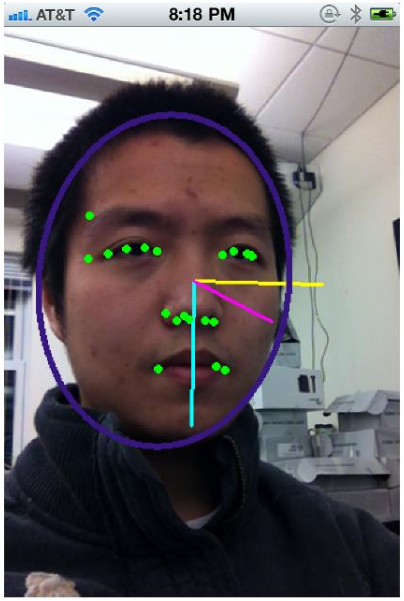
Methodology
Tracking Stage

- Feature Points Tracking Component
  - Lucas-Kanade (LK) tracking algorithm
  - CAMSHIFT

Methodology
Tracking Stage

- Pose Estimation Component
  - Pose from Orthography and Scaling with ITerations (POSIT) algorithm
    - 4 points in Image (2D) -> 3D pose estimation
    - Human head simplified to a rigid cylinder

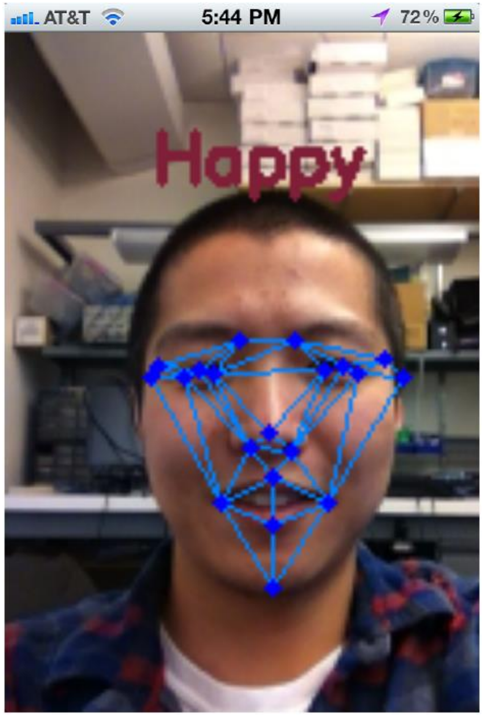


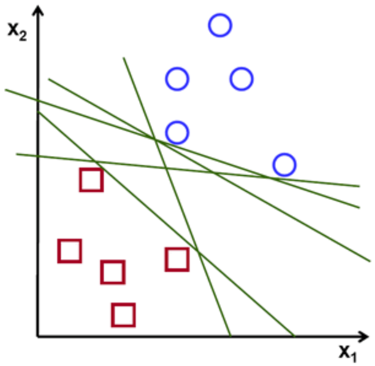


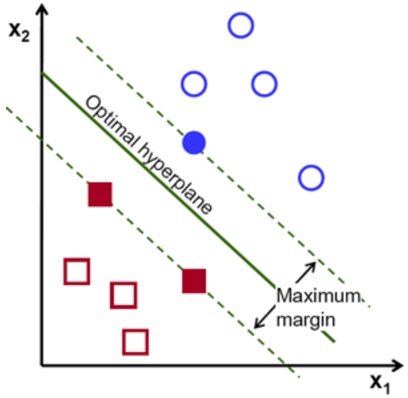
Methodology

Inference Stage

- Active Appearance Model
  - Generate appearance features for classification
  - Combine shape and texture models (more accurate)
- Expression Classification
  - Fisher Linear Discriminant Analysis (Fisherface) to reduce the dimension of the face feature vector
  - Support Vector Machine (SVM) classifier with LibSVM







Implementation

- iPhone 4
- OpenCV Library
- AAM from VOSM (Vision Open Statistical Models)

Resolution	Time (ms)
640×480	4090
480×360	2123
320×240	868
192×144	$\boxed{298}$
160×120	203
96×72	68
80×60	53

Evaluation

- Benchmarks

Tasks	Avg. CPU usage	Avg. memory usage
GUI only	< 1%	3.18MB
Pose estimation	58%	6.07MB
Expression inference	29%	4.57MB
Pose estimation & expression inference	68%	6.28MB

Table 2. CPU and memory usage under various task benchmarks

Component	Average processing time(ms)
Face detection	53
Feature points tracking	32
AAM fitting	92
Facial expression classification	3

Table 3. Processing time benchmarks

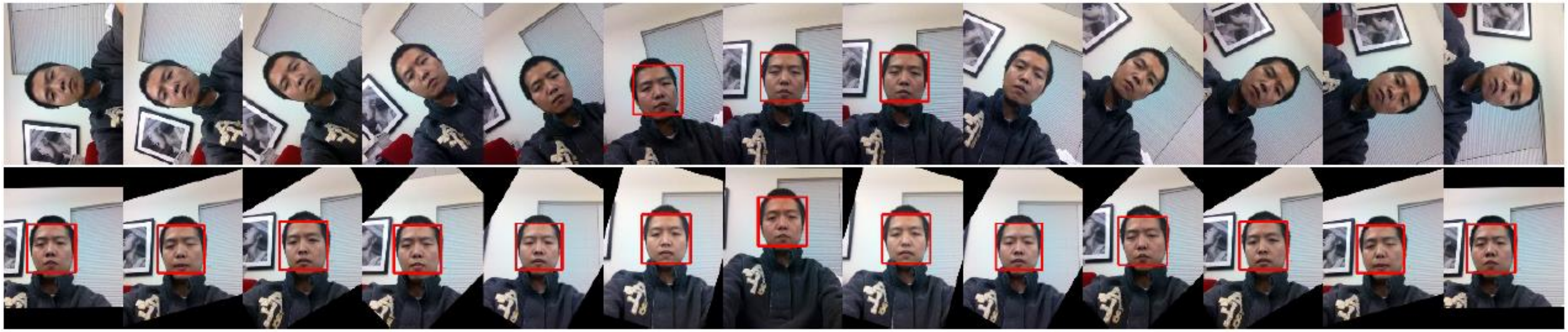
Evaluation

- Tilted Face Detection

Standard AdaBoost face detector

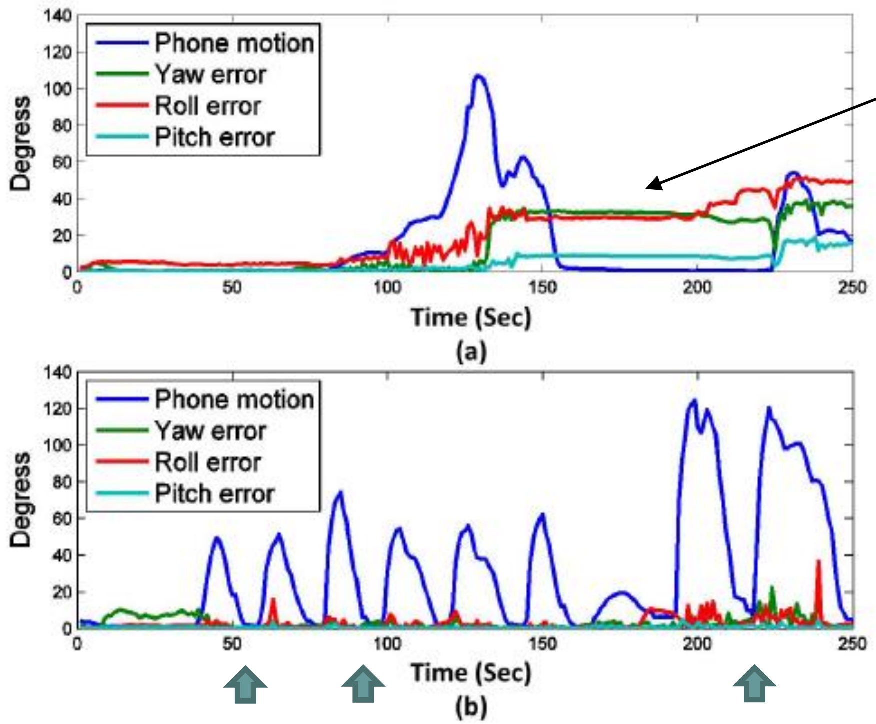
vs

Visage’s Detector



Evaluation

- Motion Based Reinicialization

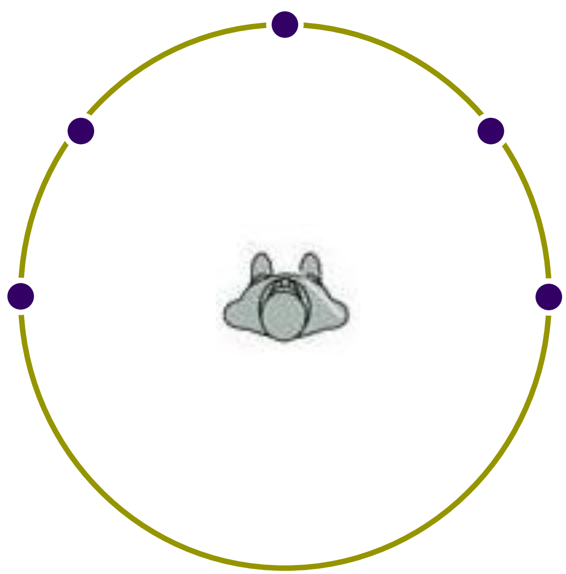


Estimation drifts

Reinitialize when variance is high

Evaluation

- Head Pose Estimation



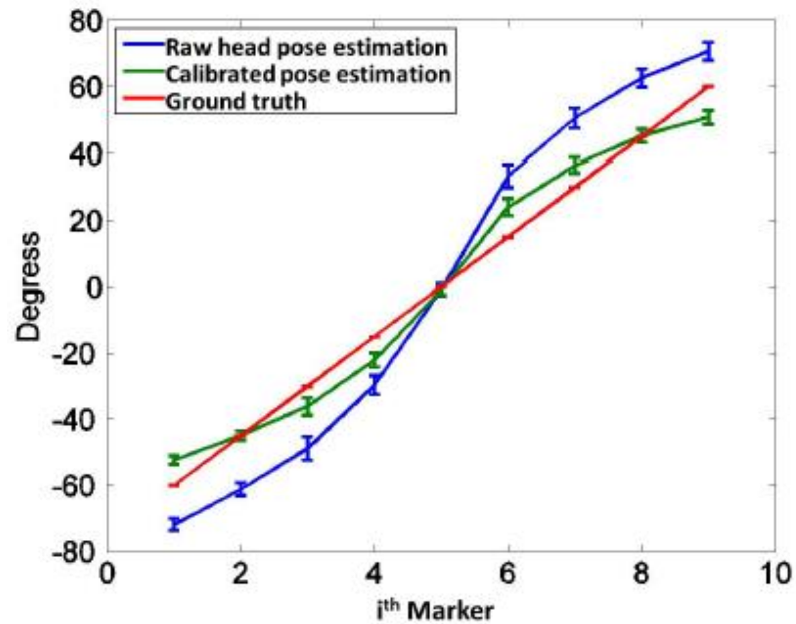


Fig. 8. Head pose estimation error

Evaluation

- Facial Expression Classification
  - Validation with The Japanese Female Facial Expression (JAFFE) Database

Expressions	Anger	Disgust	Fear	Happy	Neutral	Sadness	Surprise
Accuracy(%)	82.16	79.68	83.57	90.30	89.93	73.24	87.52

Expressions	Anger	Disgust	Fear	Happy	Neutral	Sadness	Surprise
Anger	93.33	6.67	0	0	0	0	0
Disgust	6.90	75.86	17.24	0	0	0	0
Fear	0	7.41	92.54	0	0	0	3.23
Happy	0	0	0	87.10	6.45	3.23	0
Neutral	0	0	0	0	90.00	10.00	0
Sadness	0	6.45	9.68	3.23	9.68	70.97	0
Surprise	0	0	3.33	3.33	0	0	93.33

Confusion Matrix

Applications

- Streetview+

(a) Streetview+ on the go

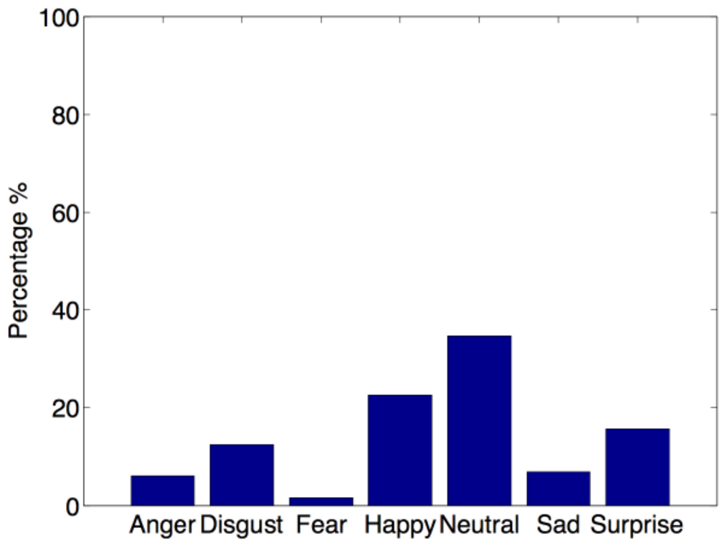
(b) Head facing front

(c) Head facing left

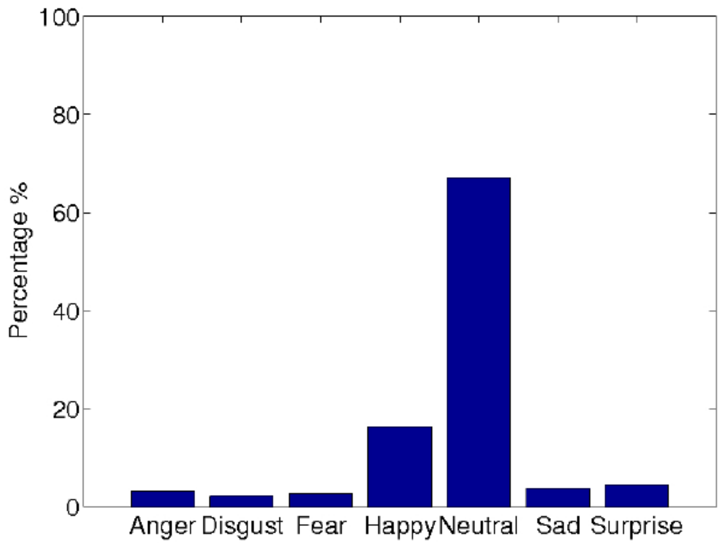
(d) Head facing right

Applications

- Mood Profiler



(a) YouTube



(b) Email

Conclusion

- **Face-aware applications**
- Designed for resource limited mobile phones
- Online processing at a lower computational cost multi-modality sensing
- Flexible and robust

References

- Yang, Xiaochao, et al. "Visage: A face interpretation engine for smartphone applications." *Mobile Computing, Applications, and Services*. Springer Berlin Heidelberg, 2012. 149-168.
- *http://cseweb.ucsd.edu/~yfreund/papers/IntroToBoosting.pdf (AdaBoost)*
- *http://docs.opencv.org/master/d7/d8b/tutorial_py_lucas_kanade.html#gsc.tab=0 (LK tracking algorithm)*
- *http://docs.opencv.org/master/db/df8/tutorial_py_meanshift.html#gsc.tab=0 (CAMSHIFT)*
- *http://makematics.com/research/posit/ (POSIT algorithm)*
- *http://www.visionopen.com/downloads/open-source-software/vosm/# (VOSM projects)*
- *http://www2.imm.dtu.dk/~aam/main/ (AAM algorithm)*
- *http://www.ics.uci.edu/~welling/classnotes/papers_class/Fisher-LDA.pdf (Fisher LDA)*
- *http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html (SVM algorithm)*
- *http://www.csie.ntu.edu.tw/~cjlin/libsvm/ (LibSVM)*
- *http://www.kasrl.org/jaffe.html (JAAFE Database)*

Thank you!