UNIVERSITÀ DEGLI STUDI DI PADOVA

Analisi E Classificazione Di Azioni Facciali

di

Gianluca Donato

Relatore:

Chiar.mo Prof. G.A. Mian

Controrelatore:

Chiar.mo Prof. R. Frezza

Alla memoria di mamma e papà.
INDICE

Dedica ........................................................................ iii
Indice ....................................................................... iv
Lista delle Figure ........................................................ vi
Lista delle Tavole ........................................................ ix
Ringraziamenti ........................................................... x

1 INTRODUZIONE ......................................................... 1
1.1 Perché l’Analisi di Azioni Facciali? ......................... 1
1.2 Gli Algoritmi ......................................................... 2
  1.2.1 Il flusso ottico ................................................. 2
  1.2.2 L’analisi olistica ............................................ 2
  1.2.3 L’analisi localizzata ....................................... 4
1.3 Risultati ............................................................... 4
1.4 Ringraziamenti ..................................................... 5

2 FACIAL EXPRESSIONS AND THE F.A.C.S. .................. 6
2.1 The Facial Action Coding System ............................ 7
2.2 Analysis of facial signals by computer .................... 9
  2.2.1 Facial action codes versus emotion categories .......... 10
  2.2.2 Feigned expressions of emotion ....................... 11

3 ANALYSIS OF FACIAL MOTION ................................... 14
3.1 Model-based techniques ....................................... 15
3.2 Feature-based approaches .................................. 16
3.3 Holistic analysis ................................................ 18
3.4 Local spatial analysis .......................................... 20
3.5 Overview of Approach ....................................... 20
3.6 Image Database ................................................ 22

4 OPTIC FLOW ANALYSIS ........................................... 25
4.1 Gradient-based optic flow .................................... 26
4.2 Correlation-based optic flow ............................... 28
4.3 Local smoothing ............................................... 29

iv
2.1 The Facial Action Coding System decomposes facial motion into component actions. The upper facial muscles corresponding to action units 1, 2, 4, 6 and 7 are illustrated. Adapted from Ekman & Friesen (1978).

2.2 Example action sequences from the database. The example sequences show two subjects demonstrating AU1 starting from a null expression and ending with a high magnitude example of the action. Frame 2 is a low magnitude example of the action, frames 3 and 4 are medium magnitude examples, and frames 5 and 6 are high magnitude.

3.1 List of facial actions classified in this study. From left to right: example cropped image of the highest magnitude action, the $\delta$-image obtained by subtracting the neutral frame (the first image in the sequence), Action Unit number, Action Unit name, and number of subjects examined.

3.2 Effect of logistic thresholding on the images. On top the original image and its corresponding histogram of pixel intensities, on the bottom the processed image.

4.1 A simple example showing the relation between the motion field and the optic flow.

4.2 A 1-dimensional example of the flow extraction using the intensity conservation equation.

4.3 Optic flow for AU9+AU25 extracted by the gradient–based technique.

4.4 A simple example of the process of flow extraction using the correlation–based algorithm. For each pixel $P(x, y)$ (top, center) in the image corresponding to $t = t_0$ we consider a small window $W_p$ centered in $P$ (in red). We then compute the correlation between $W_p$ and the bigger search areas $W_s$ (in blue) in the two other images (corresponding to time $t = t_0 - 1$ and $t = t_0 + 1$). The response is finally used to calculate the two matrices $R^{-1}$ and $R^{+1}$ (green) using Eq. (4.5). Notice that the two location highlighted in the $R$ matrices (in red) are those with correlation equal to 1, while all other displacements will yield lower responses.

4.5 Optic flow for AU1 extracted using local velocity information by the correlation-based technique, with no spatial smoothing.
4.6 Top: optic flow for AU9+AU25 extracted using local velocity information by the correlation-based technique, with no spatial smoothing. Bottom: the same flow after the local smoothing.

5.1 First 8 principal components of the difference images for the upper face actions (top), and lower face actions (bottom). Components are ordered left to right, top to bottom.

5.2 Projected data for the 3 actions classification example. \( W_{opt} = W_{pca} : \mathbb{R}^{1350} \rightarrow \mathbb{R}^2 \).

5.3 Projected data for the 3 actions classification example. \( W_{opt} = W_{pca} W_{flid} : \mathbb{R}^{1350} \rightarrow \mathbb{R}^2 \).

5.4 Example 2-D data distribution and corresponding principal component and independent component axes. Figure inspired by Lewicki & Sejnowski (submitted).

5.5 Optimal information flow in sigmoidal neurons. Left: The input \( x \) is passed through a nonlinear function, \( g(x) \). The information in the output density \( f_y(y) \) depends on matching the mean and variance of \( f_x(x) \) to the slope and threshold of \( g(x) \). Right: \( f_y(y) \) is plotted for different values of the weight, \( w \). The optimal weight, \( w_{opt} \) transmits the most information. Figure from Bell & Sejnowski(1995).

5.6 TOP: Image synthesis model for the ICA representation. BOTTOM: The ICA representation.

5.7 Example independent component images of the upper face (top) and lower face actions (bottom).

5.8 An original \( \delta \)-image on the left, with its corresponding LFA representation \( O(x) \) on the right.

5.9 On the left the input covariance corresponding to pixel \( \zeta(13,9) \), on the right the covariance of the output \( C'(\zeta, x) \). The filter is now very sparse and the redundancy is almost eliminated.

5.10 The first 155 points selected by the sparsification algorithm superimposed to the mean images of the upper and lower face actions.

6.1 The 144 principal components of 7750 random patches extracted from all the \( \delta \)-images.

6.2 Enlargement of the first 10 principal components of Figure 6.1. Components are ordered left to right, top to bottom.

6.3 Local principal components of fixed 15 × 15 patches of the upper face images. From top to bottom: Principal components 1–3.

6.4 Top: Original \( \delta \)-image. Bottom two rows, from left to right: Gabor kernels (low and high frequency), the imaginary part and magnitude of the filtered image.
6.5 *PCA Jets.* Left: two kernels corresponding to low and high frequencies (patches size $49 \times 49$ and $9 \times 9$). Right: the result of the convolution with the $\delta$-image of Figure 6.4. . . . . . . . . . . . . . 59

7.1 Projected data for the 3 actions classification example. From left to right and top to bottom the data for different magnitudes. Bottom right shows the projection of the means of all images. Notice how the plot corresponding to Delta 4 is the same as Figure 5.2. . . . . . . . . . . . . . 64
LISTA DELLE TAVOLE

7.1 Best performance for each classifier. FLD: Fisher’s linear discriminant. ICA: Independent component analysis. LFA: Local feature analysis. ................................................. 68

8.1 Summary of automated facial expression analysis systems. These models were tested on different data sets with differing levels of difficulty. Classification categories were feigned expressions of emotion, except where otherwise indicated. aClassified facial actions bClassified images from Pictures of Facial Affect (Ekman & Friesen, 1976) in which trained actors performed the muscle contractions empirically associated with certain emotional states. ............... 75
RINGRAZIAMENTI

È sempre difficile riuscire a condensare in poche righe la gratitudine verso tutti coloro che in modo diretto o indiretto mi sono stati vicini in questi anni e mi hanno aiutato a raggiungere questo traguardo. Partendo dalla fine, da questa tesi, vorrei ringraziare tutto l’ambiente di San Diego e i miei colleghi del laboratorio di neurobiologia computazionale del Salk Institute che mi hanno spronato a realizzare questo lavoro: il Professor Sejnowski prima di tutto per i suoi consigli sempre appropriati, Marni e Mike per l’aiuto datomi e Tony, compagno di nottate di lavoro insieme. Ma molte altre persone in quel periodo mi sono state d’aiuto nello svolgimento di questo lavoro: il Professor Poggio del MIT e il Professor Movellan, Dhara, Glen, Leslie, Rosemary. Dal punto di vista umano non posso dimenticare Clara —che è riuscita a sopportarmi per nove mesi— Ed, Ross e Grace. Voglio poi ringraziare anche tutti gli amici che ho avuto accanto durante tutti questi anni di università: Dario, Bruno, Gabriele, Nino, Milko, Gianpiero e tutti i compagni di Padova. Un pensiero particolare a chei di Udin: Federico, Fabrizio, Gianni, Marco, Roberto, Riccardo, sempre con me nei momenti più difficili, e Lorena e Carlo e tutto il NoFun. E la mia famiglia: Zia Carla e tutti gli zii e le nonne che mi hanno fatto sempre sentire l’aria di casa. Infine —last but not least— un pensiero speciale a Federica e Giovanni che si sono sopportati le mie peggiori crisi.

E naturalmente grazie a te, Alisa, che hai fatto di questi ultimi due anni i più belli e coinvolgenti e anche i più pazzi e confusi della mia vita.

 Lucaは、ありがたいです

Grazie a tutti di cuore.
Chapter 1

INTRODUZIONE

1.1 Perchè l’Analisi di Azioni Facciali?

Questa tesi affronta il problema di automatizzare un’attività umana di altissima specializzazione ovvero la classificazione delle espressioni facciali e più in particolare l’analisi delle “azioni” facciali così come sono state definite e classificate da Ekman [25]. L’ispirazione è prettamente psicologica: l’analisi dettagliata delle espressioni facciali è usata specialmente negli studi di psicologia comportamentale dove è fondamentale riconoscere ogni sfumatura di espressione che può aiutare a distinguere —ad esempio— un sorriso vero da uno falso [26], oppure i diversi stati psicologici che si celano dietro espressioni solo superficialmente identiche. Oltre a questo però l’automatizzazione dell’analisi usando il FACS (Facial Action Coding System) permette una soluzione più precisa del problema più generale del riconoscimento delle espressioni facciali che ha applicazioni che spaziano dallo sviluppo di nuove interfacce uomo-computer, alla biometrica. In particolare questo lavoro ha avuto l’attenzione della C.I.A. (Central Intelligence Agency) statunitense come possibile aiuto nella determinazione dei segni di un comportamento mendace.

1.2 Gli Algoritmi

A mia disposizione avevo un database di sequenze di immagini già classificate da operatori esperti \(^2\) da cui ho selezionato un sottoinsieme di 20 soggetti che compiono 12 azioni distinte (o sovrapposizione delle stesse) su cui testare i diversi algoritmi da me sviluppati. Questi si dividono in tre tipi principali:

1. basati sull’estrazione esplicita del movimento facciale usando il flusso ottico
2. basati su un’analisi statistica dell’intera immagine
3. basati su un’analisi di aree ridotte dell’immagine.

1.2.1 Il flusso ottico

Nel caso del flusso ottico (Capitolo 4) sono stati testati tre diversi metodi: il primo —più semplice— ipotizza che le variazioni dell’intensità dei singoli pixel siano dovute esclusivamente ai movimenti dell’immagine [40] e ricava delle stime di velocità orizzontale e verticale attraverso il calcolo di semplici gradienti. Il secondo metodo parte da ipotesi meno restrittive e ricava le stime delle velocità calcolando la correlazione tra la finestra intorno a ogni pixel e quelle immediatamente adiacenti nelle immagini che precedono e seguono l’immagine in una sequenza continua. Il terzo infine è un raffinamento del precedente che cerca di migliorare la stima imponendo un’ipotesi di continuità del flusso che si concretizza in una sorta di filtro passa-basso applicato selettivamente sulle zone dell’immagine dove la prima stima risulta meno accurata.

1.2.2 L’analisi olistica

Gli algoritmi degli altri due tipi utilizzano un approccio diverso nel senso che invece di tentare di estrarre esplicitamente l’informazione di movimento oper-

\(^2\)Il database è tuttora di proprietà esclusiva del Salk Institute for Biological Studies di La Jolla – U.S.A., e in particolare del Computational Neurobiology Laboratory.
ano un’analisi statistica dell’intero insieme di immagini nel tentativo di trovare una rappresentazione ottimale dei dati che riducendo la dimensionalità permetta una migliore discriminabilità delle diverse azioni con algoritmi di classificazione. Ho provato alcuni algoritmi classici della ricerca sul riconoscimento facciale quali l’analisi a componenti principali [75] e le wavelet di Gabor [48] insieme ad algoritmi sviluppati appositamente per questa applicazione (e.g. PCA Jets) o altri ancora relativamente nuovi come l’analisi a componenti indipendenti che rimane uno dei metodi più promettenti.

Tra gli algoritmi che utilizzano l’intera immagine (Capitolo 5) il più conosciuto è senza dubbio l’analisi a componenti principali (PCA) (Sezione 5.1) che sfrutta la matrice correlazione dell’intero insieme di immagini per determinare le direzioni di massima varianza. Una volta selezionate le $p$ direzioni che danno conto della maggior parte dell’intera varianza dei dati, è possibile proiettare i dati su queste direzioni e quindi rappresentarli usando solo $p$ parametri.

L’analisi a componenti indipendenti (ICA) (Sezione 5.3) si differenzia dalla PCA in quanto cerca di usare relazioni statistiche di ordine superiore al secondo (la matrice correlazione usata dalla PCA) per ottenere una rappresentazione che garantisca la massima indipendenza statistica ai dati. L’ICA non impone vincoli di ortogonalità agli assi su cui vengono proiettati i dati, e permette in questo modo di estrarre più informazione e quindi ottenere risultati migliori. L’algoritmo usato è una variante dell’originale proposto da Bell [7] e [8] che incorpora alcuni consigli dell’autore ed è particolarmente pensato per risolvere il problema della classificazione oltre che rappresentazione delle immagini dei movimenti facciali.

L’ultimo, ma più tradizionale, metodo è l’analisi multidimensionale o discriminante lineare di Fisher (FLD) (Sezione 5.2), in cui si parte dall’ipotesi che immagini di soggetti diversi che compiano lo stesso movimento giacciono su un unico iperpiano e che quindi la classificazione possa avvenire linearmente una volta effettuata una semplice proiezione su questo spazio. Lo spunto deriva dal lavoro di Belhumeur [6] dedicato al riconoscimento di facce. Come si vedrà il sistema è poco efficace nel generalizzare a immagini di nuovi soggetti non inseriti nelle immagini di training,
pur ottenendo la migliore performance assoluta su questi ultimi.

1.2.3 L’analisi localizzata

Per quanto riguarda gli algoritmi locali (Capitolo 6) tre sono derivati dalla PCA applicata a aree ristrette dell’immagine, e l’ultimo sfrutta le wavelet di Gabor e le loro proprietà di localizzazione nel dominio spaziale e spettrale.

Il primo algoritmo basato sulla PCA (vedi Sezione 6.2) utilizza una moltitudine di piccole zone quadrate scelte a caso su tutte le immagini per estrarre le componenti principali da usare poi come filtri per costruire una rappresentazione più compatta. Il secondo algoritmo (vedi Sezione 6.3) si discosta dal precedente per il fatto che le zone dell’immagine stavolta sono fissate e le componenti principali sono quindi più correlate con l’immagine di partenza (vedi Figura 6.3). La rappresentazione Jet (vedi Sezione 6.4) ha la particolarità di essere gerarchica, si ottiene infatti utilizzando le wavelet di Gabor su diverse frequenze e orientazioni spaziali per simulare il tipo di preelaborazione svolto inizialmente dalla cortecchia cerebrale nel riconoscimento dei dati visivi in arrivo dal nervo ottico. Questa rappresentazione è molto ridondante e ottiene ottimi risultati oltre a essere biologicamente plausibile. Per studiare l’eventuale somiglianza tra la rappresentazione a Jet e la PCA locale ho sviluppato infine l’algoritmo che ho chiamato *PCA Jets* (Sezione 6.5) che usa la PCA locale per ottenere una rappresentazione gerarchica analoga ai Jets di Gabor e che però non raggiunge gli stessi risultati suggerendo che le particolarità delle wavelet siano le vere responsabili dell’ottima performance.

1.3 Risultati

L’ICA e la rappresentazione a Jets di Gabor raggiungono i migliori risultati con una percentuale del 96% di generalizzazione sugli 20 soggetti e le 12 azioni considerate. Il lavoro è tutt’altro che concluso, ma i buoni risultati ottenuti promettono ottime prospettive di sviluppo dal punto di vista pratico e applicativo e offrono
nuove conferme alla validità di alcune teorie sul funzionamento del cervello umano e del sistema visivo in particolare (vedi Capitolo 8).

1.4 Ringraziamenti

Questo lavoro è stato svolto nell’arco dell’intero 1997 presso il Computational Neurobiology Laboratory del “Salk Institute for Biological Studies”, durante la mia permanenza a San Diego presso l’Università di California. Il responsabile è stato il Professor Terrence J. Sejnowski direttore del laboratorio. Ringrazio Marian Stewart Bartlett autrice del primo articolo su questo progetto [4] per il suo appoggio e il Prof. Sejnowski per la disponibilità e accoglienza offertami.
Chapter 2

FACIAL EXPRESSIONS AND THE F.A.C.S.

Facial expressions provide information not only about affective state, but also about cognitive activity, temperament and personality, truthfulness, and psychopathology. Facial measurement is used as a behavioral measure of such internal processes in studies of emotion, social interaction, communication, anthropology, personality, and child development (for reviews see [29] [27] [28]). The leading method for measuring facial movement is the Facial Action Coding System [25]. The Facial Action Coding System (FACS) provides an objective means for describing facial signals in terms of component motions, or “facial actions.” FACS is presently performed by highly trained human experts. Recent advances in image analysis open up the possibility of automatic measurement of facial signals. An automated system would make facial expression measurement more widely accessible as a tool for research and assessment in behavioral science and medicine. Such a system would also have application in human-computer interaction tools. This thesis explores and compares methods for classifying Facial Actions in image sequences of faces.
2.1 The Facial Action Coding System

The Facial Action Coding System (FACS) was developed by Ekman and Friesen [25] to investigate the relationship between facial behavior and internal state. It was developed in order to study which facial motions are produced under what conditions, how consistently are they produced, and whether the same facial motions are produced under the same conditions across cultures. Examples of the kind of questions that can only be answered by measuring the facial behavior itself include: Are there differences in facial behavior when people are telling the truth as compared to when they are lying? Do different patterns of central nervous system activity accompany different facial movements?

The Facial Action Coding System (FACS) [25] allows precise specification of the morphology (the specific facial actions which occur) and the dynamics (duration, onset and offset time) of facial movement. FACS was developed by determining from palpation, knowledge of anatomy, and videotapes, how the contraction of each of the facial muscles changed the appearance of the face. Ekman and Friesen defined 46 Action Units, or AUs, to correspond to each independent motion of the face. A trained human FACS coder decomposes an observed expression into the specific AUs that produced the expression. FACS is coded from video, and the code includes the onset, start of apex, end of apex, and offset of each facial action. More than 300 people worldwide have achieved inter-coder agreement on the Facial Action Coding System.

Dynamic information in facial behavior is a critical component of FACS. Timing, for example, contains information for discriminating genuine, emotional smiles from false ones produced voluntarily such as when smiling for the camera [26]. In addition to upward movement of the mouth corners (AU 12), a genuine smile includes an eye scrunch produced by contraction of the outer portion of the sphincter muscle around the eye (AU 6). Genuine smiles typically feature apex coordination, with AU 6 reaching its maximum intensity at the same time as AU 12, and are also shorter and smoother in execution than anxious or false smiles. Huang
Figure 2.1: The Facial Action Coding System decomposes facial motion into component actions. The upper facial muscles corresponding to action units 1, 2, 4, 6 and 7 are illustrated. Adapted from Ekman & Friesen (1978).

[41] demonstrated that the relative timing of mouth and eye motion in synthesized expressions can shift perception of a smile from a genuine smile, to a voluntary smile, to a sneer.

In recent years a number of studies have appeared showing the rich variety of information that can be obtained by using this tool. Examples include evidence of a facial signal for embarrassment [46]; differences between genuine and simulated pain [18]; cross cultural differences in how infants respond to restraint [12]; and signs of psychopathology [73]. Experiments using FACS, for example, have shown significant differences between the facial signals of suicidal and non-suicidally depressed patients [38]. When responding to the question, “Do you still wish to take your own life?” patients who had recently attempted to kill themselves displayed brief facial signals associated with contempt and disgust, whereas none of the non-suicidal patients displayed those signals. (See [29] for a report of another eighteen studies which measured facial behavior with FACS on these and related topics).

Promising as these findings are, a major impediment remains: The time required to both train human experts and to manually score the video tape. It
takes over 100 hours of training to achieve minimal competency on FACS, and each minute of video tape takes approximately one hour to score. Automating the FACS would make it more widely accessible as a research tool, and it would provide a good foundation for human-computer interaction tools. Some success has been achieved for automatic detection of facial actions by tracking the positions of dots attached to the face [39] [43]. A system that detects facial actions from image sequences without requiring application of dots to the subjects face would have much broader utility. Efforts have recently turned to measuring facial actions by image processing of video sequences [4] [14].

Components of FACS have been incorporated into computer graphic systems for synthesizing facial expressions (e.g. Toy Story [45]), and for parameterizing facial movement [69] [54] [79]. There appears to be some confusion in the computer vision literature between the Facial Action Coding System itself and computer graphic systems that implement aspects of FACS. For example, a number of criticisms of FACS such as ignoring temporal information (cf. [30]), are actually criticisms of some computer graphic implementations of FACS, and not of FACS itself. Another clarification is that although there are clearly defined relationships between FACS and the underlying facial muscles, FACS is an image-based method. Facial actions are defined by the motion and image changes they produce in video sequences of face images.

2.2 Analysis of facial signals by computer

Recent advances have been made in computer vision for automatic recognition of facial expressions in images. The approaches that have been explored include analysis of facial motion [54] [79] [68] [30], measurements of the shapes of facial features and their spatial arrangements [49], holistic spatial pattern analysis using techniques based on principal component analysis [16] [61] [49] and methods for relating face images to physical models of the facial skin and musculature [54] [74] [52] [30]. A number of methods that have been developed for representing faces for
identity recognition may also be powerful for expression analysis. These include Gabor wavelets [19] [48], linear discriminant analysis [6], local feature analysis [62], and independent component analysis [3] [2]. These approaches are reviewed below, but first we present some considerations for building an automatic facial expression analysis system that will be of practical use when presented with the diversity of facial signals that occur during natural communication.

2.2.1 Facial action codes versus emotion categories.

Most systems for automatic facial expression analysis attempt to classify expressions into a few broad categories of emotion, such as happy, sad, or surprised, often quoting behavioral evidence for seven universal facial expressions (see [23] for a review of universals in facial expression). This is a reasonable starting point, but there tends to be a basic misunderstanding of these studies in the computer vision literature. The universal expression studies demonstrate consistent facial signals across cultures for seven discrete emotions, but do not imply that all facial expressions are subsumed by one of these categories.

Real facial signals consist of thousands of distinct expressions, for many of which a gross category assignment would be impossible, misleading, or simply insufficient [35]. Signals of two or more emotions may occur in the same facial expression, such as blends of happiness and disgust which produce “smug”, or blends of happiness and sadness which produce “nostalgia” [24]. For applications such as user-interfaces, computer games, or TV ratings, it may be important to distinguish “happy-surprise” from “fearful-surprise” or “horrified-surprise” which implies disgust as well. Within an emotion category such as anger, there are variations in the intensity of the emotion, such as annoyance and fury, and variants of the emotion such as vengeance and resentment. Most facial movements shown during a social interaction are not relevant to emotion at all, but are what Ekman called “conversational signals” [22]. A variety of facial movements provide different kinds of emphasis to speech and can also provide information about syntax. This
information is important for understanding the nature of conversation, identifying fluctuations in the speaker's level of involvement with what he or she is saying, and could augment a speech recognition system. Finally, there are facial tics, mannerisms, and other peculiarities in facial movement which may be relevant to personality, psychopathology, and/or brain lesions.

If automated facial measurement were, then, to be constructed simply in terms of seven elementary emotional categories, much important information would be lost: blends of two emotions, variations within an emotional category, variations in intensity, conversational signals, and idiosyncratic facial movements. Such information need not be lost if the automated system is based on the Facial Action Coding System, which provides a description of the basic elements of any facial movement, analogous to a set of phonemes for facial expression. Facial Action codes provide a rich description of facial behavior, and a system that outputs Facial Action codes instead of, or in addition to, emotion category labels will be a more powerful tool for applications both in industry and behavioral science.

For basic science inquiries into the relationship of facial behavior and internal state, objective measurements of facial behavior are required and a system that only produced emotion category labels could not be used for that purpose. Some computer vision systems do provide explicit descriptions of facial movement [54] [79][30], but it is not known whether these descriptions are sufficient for describing the full range of facial behavior. These descriptions are not readily interpretable in terms of Facial Action codes, and there is not yet any empirical data establishing relationships between these motion descriptions and internal state. A large body of behavioral data already exists establishing the relationship of Facial Action codes to emotions, emotion intensity, variations, blends, and conversational signals.

2.2.2 **Feigned expressions of emotion.**

Another important issue for developing an automated facial expression analysis system is obtaining a reliable set of training images from which to build the sys-
tem [35]. Most systems have utilized datasets of voluntary expressions, in which subjects were asked to “look happy”, “sad”, or “surprised”. While the use of such feigned expressions is again a reasonable starting point, there are limitations to such a datasets that must be recognized. Voluntary expressions differ from spontaneous expressions in response to actual emotion. There is evidence that they are mediated by different neural substrates [67]. When subjects produce voluntary expressions, some actions tend to be exaggerated, some actions can be incorrectly included, while some actions that are usually present when the emotion is experienced are omitted [22]. There is a tendency to exaggerate facial motions that are used as conversational signals, such as a full brow raise, and omit other motions for which we have less practice at cognitive control during speech, such as many signals in the eye region [24]. It can be important to detect these latter signals, since they are also less successfully masked or neutralized when attempting to hide one’s emotional state [24].

More reliable datasets could be obtained a number of ways, including recording images of subjects as they watch emotive videos. FACS provides another means of obtaining a reliable dataset. Subjects can be instructed to perform specific Facial Actions, and the veracity of the dataset would not depend on the subjects’ acting abilities, but rather on the FACS score describing the actual facial behavior in the images. Figure 2.2 shows image sequences from our database of two subjects performing an individual Facial Action, the inner brow raiser.
Figure 2.2: Example action sequences from the database. The example sequences show two subjects demonstrating AU1 starting from a null expression and ending with a high magnitude example of the action. Frame 2 is a low magnitude example of the action, frames 3 and 4 are medium magnitude examples, and frames 5 and 6 are high magnitude.
Chapter 3

ANALYSIS OF FACIAL MOTION

The majority of work on facial expression recognition has focused on facial motion analysis through optic flow estimation (see 4 for more details). In an early exploration of facial expression recognition, Mase [54] used optic flow to estimate the activity in 12 of the 44 facial muscles. For each muscle he defined a window in the face image and an axis along which each muscle expands and contracts. The mean similarity of the flow vectors inside the window to this axis provided a coarse estimate of the activity of the muscle. Mase also explored a statistically driven technique for recognizing facial expressions from optic flow. Means and covariances of optic flow in local regions of the face provided a high-dimensional feature vector, of which the 15 measures with the highest ratios of between-class to within-class variability were used for expression classification.

Yacoob & Davis [79] combined tracking of major facial features with local analysis of optic flow. They constructed a mid-level representation of facial motion by first locating and tracking prominent facial features, and then quantizing the optic flow within subregions of each feature into eight principal directions. The mid-level representation consisted of such descriptions as “right mouth corner raises.” These descriptions were then classified into one of six facial expressions using a set
of heuristic rules.

Rosenblum, Yacoob & Davis [68] expanded this work to analyze facial expressions using the full temporal profile of the expression, from initiation, to apex, and relaxation. They trained radial basis function neural networks\(^1\) to estimate the stage of an expression from a facial motion description similar to [79], and constructed separate networks for each expression. Radial basis functions approximate nonlinear mappings by Gaussian interpolation of examples and are well suited to modeling systems with smooth transitions between states. The output of each expression network over the full time-course of the expression was then analyzed, and a set of heuristic rules were established to determine whether to accept or reject the image sequence as an example of that expression.

Cohn et al. [14] are developing a system for facial action coding based on human-computer interaction. Over 40 feature points were manually located in the initial face image, and the displacements of these feature points were estimated by optic flow. Discriminant functions were employed to classify facial actions from the set of 40 displacements.

### 3.1 Model-based techniques.

Several facial expression recognition systems have employed explicit physical models of the face [54] [74] [52] [30]. Mase’s approach described above, for example, employed an extremely simple physical model. Essa & Pentland [30] extended a detailed anatomical and physical model of the face developed by Terzopoulos and Waters [74] applied it to both recognizing and synthesizing facial expressions. The model included 44 facial muscles, their points of attachment to the skin, and the

---

\(^1\)Neural networks are used to approximate multivariate nonlinear functions using sparsely sampled data from the function to be approximated. The Radial Basis Function Network (RBFN) uses the classical 3-layer architecture of a neural network (input, hidden and output) but the units composing the hidden layer are driven by radial basis activation functions (usually gaussians) called receptive fields. This way the RBFN implements a form of template matching in which a template contributes to the generation of the output proportional to the degree that the template matches the input and by the significance of that template to the output.
elastic properties of the skin modeled in a geometric mesh. Images of faces were mapped onto the physical model by image warping based on the locations of six points on the face. Motion estimates from optic flow were refined by the physical model in a recursive estimation and control framework, and the estimated forces were used to classify the facial expressions. In a variation on this work, Essa and Pentland generated templates of 2-D motion energy by back-projecting the “corrected” motion into the 2-D image. Facial expressions were recognized by template matching in the 2-D image space, bypassing the more time consuming physical model.

In a model-based system, classification accuracy is limited by both the accuracy of mapping image onto the model and the validity of the model itself. There are numerous factors that influence the motion of the skin following muscle contraction, and it would be difficult to accurately account for all of them in a deterministic model. In this thesis we take an image-based approach in which Facial Action classes are learned directly from example image sequences of the actions, bypassing the physical model.

Beymer, Shashua, and Poggio [10] demonstrated the potential of example-based approaches for analysis and synthesis of face images. They trained radial basis function neural networks to learn the transformation from optic flow fields to pose and expression coordinates, and from pose and expression coordinates back to optic flow fields. The estimated optic flow fields could be used to synthesize new poses or expressions from an example image by image warping techniques.

3.2 Feature-based approaches.

One of the earliest techniques for recognizing facial identity in images was based on the computation of a set of geometrical features of the face such as nose length, chin shape, and distance between the eyes [44] [11]. Geometrical features relevant to facial expression analysis might include mouth shape, eye to eyebrow distance, or the magnitude of wrinkles in specified locations on the face. A difficulty
with feature-based approaches to image analysis is that the specific image features relevant to the classification may not be known in advance, and the selected set of features may fail to capture sufficient discriminative information.

An alternative to feature-based image analysis is template matching. The two approaches differ significantly in that the template matching doesn't require the explicit extraction and/or definition of any salient feature to be identified in order to classify the image. The whole image is used to build the templates which could be as trivial as the average of all the training images belonging to a particular class to be identified in a pattern recognition task. This approach is usually faster, less computationally intensive and less prone to misconceptions in the models used since little a priori assumptions are made on the nature of the data. Templates capture information about configuration and shape that can be difficult to parameterize. Template matching has been shown to outperform feature-based methods for face recognition [11] [49]. In previous work on this project [4], Marian Bartlett explored classification of upper facial actions using feature measurements such as the increase of facial wrinkles in specific facial regions and a measure of eye opening. She found that these feature measurements were less reliable indicators of facial actions than template-based classifiers. The poor performance of the feature measures was attributed to differences in patterns of facial wrinkling across subjects due not only to age but to variations in facial morphology as well.

Feature-based and template-based methods need not be mutually exclusive. Lanitis, Taylor, & Cootes, [49] recognized identity, gender, and facial expressions by measuring shapes and spatial relationships of a set of facial features using a flexible face model. Performance improved by augmenting this set of features with parameters containing information about modes of variation in grayscale images, using the principal component analysis techniques described in Section 3.3.
3.3 Holistic analysis.

One form of holistic analysis is to classify facial expressions directly from the image pixels training a neural network through back-propagation. Kobayashi, Tang, and Hara [47] obtained 90% accuracy using a neural network to recognize six basic expressions from selected columns of the graylevel image pixels. Neural network approaches to image analysis can be advantageous for face processing since the physical properties relevant to the classification need not be specified in advance. The weights are equivalent to a learned set of templates. The network learns the relevant “features” from the statistics of the image set, where these features could be either local or involve relationships among multiple image locations. This is particularly useful when the specific features relevant to the classification are unknown. (See [76] for a review of connectionist approaches to processing images of faces.)

Another holistic spatial representation is based on the principal components of the image pixels [17] [75]. The principal components are the eigenvectors of the pixelwise covariance matrix of the set of images (i.e. Karhunen-Loève transform) (see refsec:pca). A low-dimensional representation of the face images with minimum reconstruction error is obtained by projecting each image onto the first few principal component axes. The position and scaling of the faces is critical to the success of such image-based approaches, the alignment of the training images is determinant. Principal components analysis has been applied successfully to recognizing both facial identity [17] [75], and facial expressions [16] [61] [4]. Feedforward networks taking such holistic representations as input can also successfully classify gender from face images [16] [33]. A class-specific linear projection of a principal components representation has recently been shown to give highly accurate identity recognition performance when other examples of the same faces are available to calculate the projection weights [6]. New views of a face can be synthesized from a sample view using principal components representations of face shape and texture [77]. Principal component representations of face images have also been
shown to account well for human perception of distinctiveness and recognizability [60] [37].

Representations based on principal components analysis address the second-order statistics of the image set, and do not address high-order statistical dependencies such as the relationships among three or more pixels. In a task such as facial expression recognition, much of the important information may be contained in the high-order relationships among the image pixels. Independent component analysis (ICA, refsec:ica) is a generalization of PCA which uses the high-order moments of the input in addition to the second-order moments. An unsupervised learning algorithm for performing ICA was recently developed [7]. This algorithm has proven successful for separating a set of randomly mixed auditory signals, known as the cocktail party problem [7], and also has been applied to separate EEG signals [53], fMRI signals [55], and find image filters that give independent outputs from natural scenes [8]. A representation for face recognition based on the independent components of face images has recently been developed [3] [2], and was found to be superior to the PCA representation for classifying facial identity across changes in expression and changes in pose.

Penev and Atick [62] recently developed a topographic representation based on principal component analysis. The representation is based on a set of kernels that are optimally matched to the second-order statistics of the input ensemble. The kernels were obtained by performing zero phase whitening of the principal components, followed by a rotation to topographic correspondence with pixel location. Penev and Atick call the technique local feature analysis (LFA) because the resulting kernels contain spatially local regions of nonzero value, but in this thesis we class this technique as holistic, as with ICA, since the image-dimensional kernels result from statistical analysis over the whole image. Atick’s group obtained the highest recognition performance so far on the FERET face recognition test [63]. Although the actual techniques employed during this test have not been disclosed, it has been implied that they included some of the principles embodied in LFA.
3.4 Local spatial analysis

Representations based on local spatial filters may be superior to spatially global representations for image classification. Padgett & Cottrell [61] improved the performance of their facial expression recognition system by performing PCA on subimage patches of $30 \times 30$ pixels of the face images, and using these $30 \times 30$ principal component images as convolution kernels. This finding is supported by Gray, Movellan & Sejnowski [34] who also obtained better performance for visual speechreading using the principal components of subimage patches as convolution kernels over a representation based on the principal components of the full images.

Gabor filters, obtained by convolving a 2-D sine wave with a Gaussian envelope, are local filters that resemble the responses of visual cortical cells [19]. Representations based on the outputs of these filters at multiple spatial scales, orientations, and spatial locations, have been shown to be useful for recognizing facial identity [48]. In a direct comparison of face recognition algorithms, Gabor filter representations gave better face recognition performance than representations based on principal components analysis [81].

3.5 Overview of Approach

This thesis explores and compares methods for classifying facial actions in image sequences of faces. We examine a number of techniques that have been presented in the literature for processing images of faces. We had to adapt many algorithms that were not intended for use on images of faces or were designed for facial recognition rather than for the classification of facial actions, and in the process we realized that these images present enough peculiarities to justify the development of ad hoc methods. Finally we develop some new algorithms and study the performance of all methods on this particular image analysis task.

1. Optic Flow. Two methods for estimating optic flow were implemented and compared: a fast gradient-based method for calculating flow between pairs
of images based on [40], and a correlation-based method [72]. Local smoothing is commonly imposed on flow fields to clean up the signal. We also examined the effects of local smoothing on classification of facial motion.

2. **Holistic analysis.** Next, we examined classification performance based on holistic spatial analysis of the graylevel images. We compared four holistic representations: principal component analysis (PCA), which finds a set of image-dimensional kernels that decorrelate the second order statistics of the dataset; independent component analysis (ICA), which minimizes the higher order dependencies in addition to the covariance; Local feature analysis (LFA), which is a topographic representation based on principal component analysis; and Fisher’s linear discriminants (FLD), which computes a class-specific linear projection of the PCA representation onto lower dimensions.

3. **Local filters.** In addition, we examined representations based on the outputs of local filters. A local filter consisting of a simple $15 \times 15$ Gaussian provided a benchmark for classification performance using local filters. Three local filters based on principal components analysis were then compared. The first filter consisted of the principal component eigenvectors of $15 \times 15$ subimage patches taken from random locations throughout the images. These principal component eigenvectors were used as convolution kernels for filtering the entire face image. The second local representation consisted of the principal components of $15 \times 15$ subimage patches at fixed locations in each image. These PCA-based local representations were compared to a Gabor wavelet representation for classifying facial action. The local-PCA random representation is related to the Gabor representation in that it provides the local amplitude spectrum of the image. In order to further compare the Gabor and local-PCA representation, we devised an intermediate representation based on local-PCA that contained multiscale, hierarchical properties corresponding to the Gabor filter bank. This representation matched the Gabor representation for both spatial scale and number of filters.

4. **Human Subjects.** The ability of naive human subjects to classify facial actions in the same images that were presented to the classification algorithms
provided a benchmark for the performance of those systems. Since the long-term goal of this project is to replace human expert coders with an automated system, a second benchmark was provided by the agreement rate of expert coders on these images.

3.6 Image Database

We collected a database of image sequences of subjects performing specified facial actions. The full database contains over 1100 sequences containing over 150 distinct actions, or action combinations, and 24 different subjects. Trained FACS experts provided demonstrations and instructions to subjects on how to perform each action. The selection of images was based on FACS coding of stop motion video. The images were coded by three experienced FACS coders certified with high intercoder reliability. The criterion for acceptance of images was that the requested action and only the requested action was present. For this investigation, we used data from 20 subjects and attempted to classify 12 actions: 6 upper face actions and 6 lower face actions. See Figure 3.1 for a summary of the actions examined. The actions were divided into upper and lower-face categories because facial actions in the lower face have little influence on facial motion in the upper face, and vice versa [25] which allowed us to treat them separately.

We obtained examples of the 12 actions from 20 subjects. Each example action in the database consisted of a sequence of six images, beginning with a neutral expression and ending with a high magnitude \(^2\) muscle contraction (Figure 2.2).

The first image in each sequence was located by manually marking three points: the centers of the eyes and mouth. These locations were then used throughout the sequence. The performance of some methods depended on obtaining highly accurate normalization of the faces. The three coordinates were then used to center, rotate, scale, and finally crop a window of 60 × 90 pixels around the region

\(^2\)The term “magnitude” replaces the term “intensity” used in FACS to avoid confusion with image intensity.
<table>
<thead>
<tr>
<th>Upper Face</th>
<th>Action Unit</th>
<th>Subjects</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td>1 Inner brow raiser</td>
<td>9</td>
</tr>
<tr>
<td><img src="image2.png" alt="Image" /></td>
<td>2 Outer brow raiser</td>
<td>10</td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td>4 Brow lowerer</td>
<td>18</td>
</tr>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td>5 Upper lid raiser</td>
<td>20</td>
</tr>
<tr>
<td><img src="image5.png" alt="Image" /></td>
<td>6 Cheek raiser</td>
<td>5</td>
</tr>
<tr>
<td><img src="image6.png" alt="Image" /></td>
<td>7 Lid tightener</td>
<td>18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lower Face</th>
<th>Action Unit</th>
<th>Subjects</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image7.png" alt="Image" /></td>
<td>17 Chin raiser</td>
<td>8</td>
</tr>
<tr>
<td><img src="image8.png" alt="Image" /></td>
<td>18 Lip puckerer</td>
<td>4</td>
</tr>
<tr>
<td><img src="image9.png" alt="Image" /></td>
<td>9 Nose wrinkle</td>
<td>4</td>
</tr>
<tr>
<td><img src="image10.png" alt="Image" /></td>
<td>25 Lips part</td>
<td></td>
</tr>
<tr>
<td><img src="image11.png" alt="Image" /></td>
<td>10 Upper lip raiser</td>
<td>5</td>
</tr>
<tr>
<td><img src="image12.png" alt="Image" /></td>
<td>25 Lips part</td>
<td></td>
</tr>
<tr>
<td><img src="image13.png" alt="Image" /></td>
<td>16 Lower lip depressor</td>
<td>4</td>
</tr>
<tr>
<td><img src="image14.png" alt="Image" /></td>
<td>25 Lips part</td>
<td></td>
</tr>
<tr>
<td><img src="image15.png" alt="Image" /></td>
<td>20 Lip stretcher</td>
<td>6</td>
</tr>
<tr>
<td><img src="image16.png" alt="Image" /></td>
<td>25 Lips part</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.1: List of facial actions classified in this study. From left to right: example cropped image of the highest magnitude action, the $\tilde{\delta}$-image obtained by subtracting the neutral frame (the first image in the sequence), Action Unit number, Action Unit name, and number of subjects examined.
of interest (eyes or mouth). To control the variation in lighting between frames of the same sequence and in different sequences, we applied a logistic filter whose parameters were chosen to match the statistics of the grayscale levels of each sequence [56]. Every image was thresholded by the function

$$l(\mu, \sigma) = \frac{1}{1 - e^{-K \frac{\sigma}{\sqrt{2\pi}}(x-\mu)}}$$

where \(\mu\) and \(\sigma\) are respectively the average and the standard deviation of each entire image sequence (see Figure 3.2. The parameter \(K\) controls the sharpness of the logistic function. When \(K \approx 1\) the function approximates histogram equalization; following [56] we set \(K = 1.2\).

![Figure 3.2: Effect of logistic thresholding on the images. On top the original image and its corresponding histogram of pixel intensities, on the bottom the processed image.](image)
Chapter 4

OPTIC FLOW ANALYSIS

To define the optic flow let’s consider a situation like the one depicted in Figure 4.1. The picture shows a 3D scene projected onto a 2D surface. If a 3D object moves with respect to this surface with a velocity $\vec{v}$, its projected image varies accordingly with an apparent velocity $\vec{v}_x + \vec{v}_y$. The movement of the projection of each point in the world is called motion field. The optic flow is an approximation of the motion field obtained measuring the apparent motion of local regions from one frame to another.

Figure 4.1: A simple example showing the relation between the motion field and the optic flow.
We compared two methods for calculating flow, a fast gradient-based method for calculating flow between pairs of images based on [40], and a correlation-based method [72]. We also examined the contribution of local smoothing of the flow signal to action classification.

4.1 Gradient-based optic flow

Local estimates of motion in the direction of the image gradient were obtained from pairs of images by an algorithm based on the intensity conservation equation [40]:

\[ 0 = \frac{dI(x, y, t)}{dt} \Rightarrow 0 = \frac{\partial x}{\partial t} \frac{\partial I(x, y, t)}{\partial x} + \frac{\partial y}{\partial t} \frac{\partial I(x, y, t)}{\partial y} + \frac{\partial I(x, y, t)}{\partial t} \]  (4.1)

where \( u \triangleq \frac{\partial x}{\partial t} \) and \( v \triangleq \frac{\partial y}{\partial t} \) are the local velocities in the \( x \) and \( y \) directions. This equation assumes that there is no overall gain or loss of brightness in the image over time, and that any local changes in brightness can be accounted for by shifts in spatial position. The image velocity is defined in terms of the spatial and temporal gradients of the image, \( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \) and \( \frac{\partial I}{\partial t} \). These image gradients can be estimated directly from the pair of images, and we can solve for \( u \) and \( v \) to obtain an estimate of the velocity in the direction of the image gradient.

Images were smoothed by a 5 \( \times \) 5 Gaussian kernel. Estimates of the spatial gradients, \( \Delta x \) and \( \Delta y \), were obtained with horizontal and vertical Sobel edge filters. The temporal gradient was estimated by

\[ \Delta t = I(x, y, t_1) - I(x, y, t_0) \]  (4.2)

We took as our local estimates of image velocity
Figure 4.2: A 1-dimensional example of the flow extraction using the intensity conservation equation.

\[ \dot{u} = \frac{\Delta x}{\Delta t} \]

\[ \dot{v} = \frac{\Delta y}{\Delta t} \]  

(4.3)

We can see more clearly how the algorithm works in the simple one-dimensional example shown in Figure 4.2 where the curves represent the functions $I(x, t_0)$ and $I(x, t_1)$. The estimate for $u$ can be seen immediately on the plot.

Gradient-based techniques for estimating optic flow such as this one give reliable estimates of velocity only at points where the spatial and temporal gradient of the image sequence is high. We therefore retained only the velocities from those locations. Velocity estimates were set to zero at locations at which

\[ r = \Delta x^2 + \Delta y^2 \leq 0.2 \]  

(4.4)

where $r$ is the total edge measure. One of the advantages of this simple local estimate of flow was speed. It took 0.25 seconds on a Dec Alpha 2100a processor to compute one flow field. Unfortunately the spatial resolution was very low as shown in Figure 4.3, and this led us to investigate other methods.
4.2 Correlation-based optic flow

The second estimate of optic flow was obtained by correlation-based extraction of local velocity information [72]. As in the gradient-based approach, this algorithm assumes a luminance conservation constraint. We start with a sequence of three images at time $t = t_0 - 1, t_0, t_0 + 1$ and use it to recover all the velocity information available locally. For each pixel $P(x, y)$ in the central image ($t = t_0$):

1. a small window $W_p$ $3 \times 3$ pixels is formed around $P$;
2. a search area $W_s$ $5 \times 5$ pixels is considered around location $(x, y)$ in the other two images;
3. the correlation between $W_p$ and each corresponding window in $W_s$ is computed, thus giving the matching strength, or response, at each pixel in the search window $W_s$.

Instead of selecting the pixel in $W_s$ with the highest response as the correct match for $P$, we can visualize $W_s$ as covered with a “response distribution” $R^{t-t_0}$ [72].

$$R^{+1}(u, v) = \exp\{-k\{ \sum_{i=-2}^{2} \sum_{j=-2}^{2} [I(x + i, y + j, t_0) - I(x + i + u, y + j + v, t_0 + 1)]^2 \}\}$$

$$R^{-1}(u, v) = \exp\{-k\{ \sum_{i=-2}^{2} \sum_{j=-2}^{2} [I(x + i, y + j, t_0) - I(x + i + u, y + j + v, t_0 - 1)]^2 \}\}$$

(see also Figure 4.4 for an example of the process). The parameter $k$ is chosen to be 0.95 to avoid numerical problems [72]. To incorporate information in $R^{-1}$ and $R^{+1}$, Singh suggests the use of the simple heuristics

$$R = R^{+1} + R^{-1}$$

(4.6)

where $R^{-1}$ indicates a “rotation” of $\pi$ of the responses in $R^{-1}$. 
\( R \) is a frequency distribution in velocity space: The response at a point gives the frequency of occurrence—or likelihood—of the corresponding value of each velocity. Using this interpretation we can compute an estimate of the true velocity using least squares weighted by the response itself.

\[
\hat{u} = \frac{\sum_u \sum_v R(u, v) u}{\sum_u \sum_v R(u, v)} \\
\hat{v} = \frac{\sum_u \sum_v R(u, v) v}{\sum_u \sum_v R(u, v)} \quad u, v \in [-2, 2]
\]

(4.7)

Since the true velocity is a point in \((u, v)\) space, the displacement from any point in \(W\) is the error associated to the corresponding velocity. Assuming the errors are additive, independent and zero-mean, a covariance matrix \(S_{cc}\) is naturally associated to the estimate \((\hat{u}, \hat{v})\):

\[
S_{cc} = \begin{bmatrix}
\sum_u \sum_v R(u, v) (u-u_{cc})^2 & \sum_u \sum_v R(u, v) (u-u_{cc})(v-v_{cc}) \\
\sum_u \sum_v R(u, v) (u-u_{cc})(v-v_{cc}) & \sum_u \sum_v R(u, v) (v-v_{cc})^2 
\end{bmatrix}
\]

(4.8)

The need for \(S_{cc}\) will be clear in the next section. Figure 4.5 shows an example flow field obtained by this algorithm.

### 4.3 Local smoothing

To refine the conservation constraint estimate \(U_{cc} = (\hat{u}, \hat{v})\) for the velocity of each pixel \(P\), we estimate the velocity at \(P\) from the velocities in a neighborhood around \(P\). Assuming no spatial discontinuities in the motion, the velocity \(U'_{cc} = (\hat{u}', \hat{v}')\) at each \(P'\) in a neighborhood of \(P\) can be thought of as a measurement of the velocity of \(P\). The local neighborhood estimate of velocity, \(\overline{U}\), is a weighted sum of the velocities at \(P'\) using a \(5 \times 5\) Gaussian mask.

An optimal estimate \(U\) of \((u, v)\) should combine the two estimates \(U_{cc}\) and \(\overline{U}\), from the conservation and local smoothness constraints respectively. Since \(U\) is
a point in \((u,v)\) space, its distance from \(\overline{U}\), weighted by its covariance matrix \(\overline{S}\), represents the error in the smoothness constraint estimate. Similarly, the distance between \(U\) and \(U_{cc}\) weighted by \(\overline{S_{cc}}\) represents the error due to conservation constraints. Computing \(U\) then, amounts to simultaneously minimizing the two errors:

\[
U = \arg \min \{ ||U - U_{cc}||_{S_{cc}} \wedge ||U - \overline{U}||_{\overline{S}} \} \tag{4.9}
\]

Since we do not know the true velocity, this estimate must be computed iteratively. To update the field we use the equations [72]:

\[
U^0 = U_{cc} \\
U^{k+1} = [S_{cc}^{-1} + \overline{S}^{-1}]^{-1}[S_{cc}^{-1}U_{cc} + \overline{S}^{-1}\overline{U}^k]
\tag{4.10}
\]

where \(\overline{U}^k\) is the estimate derived from smoothness constraints at step \(k\). The iterations stop when

\[
||U^{k+1} - U^k|| < \varepsilon \tag{4.11}
\]

with \(\varepsilon \propto 10^{-4}\).
Figure 4.3: Optic flow for AU9+AU25 extracted by the gradient-based technique.
Figure 4.4: A simple example of the process of flow extraction using the correlation-based algorithm. For each pixel $\mathcal{P}(x, y)$ (top, center) in the image corresponding to $t = t_0$ we consider a small window $\mathcal{W}_p$ centered in $\mathcal{P}$ (in red). We then compute the correlation between $\mathcal{W}_p$ and the bigger search areas $\mathcal{W}_s$ (in blue) in the two other images (corresponding to time $t = t_0 - 1$ and $t = t_0 + 1$). The response is finally used to calculate the two matrices $\mathcal{R}^{-1}$ and $\mathcal{R}^{+1}$ (green) using Eq. (4.5). Notice that the two location highlighted in the $\mathcal{R}$ matrices (in red) are those with correlation equal to 1, while all other displacements will yield lower responses.
Figure 4.5: Optic flow for AU1 extracted using local velocity information by the correlation-based technique, with no spatial smoothing.
Figure 4.6: Top: optic flow for AU9+AU25 extracted using local velocity information by the correlation-based technique, with no spatial smoothing. Bottom: the same flow after the local smoothing.
Chapter 5

HOLISTIC ANALYSIS

The holistic spatial analysis algorithms each found a set of $n$-dimensional data-driven image kernels, where $n$ is the number of pixels in each image. The analyses were performed on the difference (or $\delta$) images (Figure 3.1), obtained by subtracting the first image in a sequence (neutral frame) from all of the subsequent frames in each sequence. Advantages of difference images include robustness to changes in illumination, removal of surface variations between subjects, and emphasis of the dynamic aspects of the image sequence [56]. Holistic kernels for the upper and lower-face subimages were calculated separately.

We began with the zero-mean data matrix $X$ where the $\delta$-images were stored as row vectors $x_j$:

$$
X = \begin{bmatrix}
x_{11} & x_{12} & \cdots & x_{1n} \\
x_{21} & x_{12} & \cdots & x_{2n} \\
\vdots \\
x_{N1} & x_{N2} & \cdots & x_{Nn}
\end{bmatrix} = \begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_N
\end{bmatrix}
$$

(5.1)

In the following descriptions, $n$ is the number of pixels in each image, $N$ is the number of training images and $p$ is the size of the final representation. For all the algorithms based on principal component analysis $p$ is also the number of principal components retained to build the representation.
5.1 Principal Component Analysis: “EigenActions”

This approach is based on [16] and [75], with the primary distinction in that we performed principal components analysis on the dataset of difference images. The principal components were obtained by calculating the eigenvectors of the pixelwise covariance matrix, $S$ of size $n \times n$, of the matrix of all $\delta$-images, $X$. The eigenvectors were found by decomposing $S$ into the orthogonal matrix $P$ and diagonal matrix $D$:

$$S = PD P^T$$

(5.2)

Examples of the eigenvectors are shown in Figure 5.1. The zero-mean $\delta$-frames of each sequence were then projected onto the first $p$ eigenvectors in $P$, producing a vector of $p$ coefficients for each image.

5.1.1 Case study: classification of 3 actions.

In order to understand the power of PCA to reduce the dimensionality of the data, let’s analyze a simple example where we consider only three actions: AU17, AU18 and AU25. We determined with PCA an optimal projection matrix $W_{opt} = W_{pca}$ to reduce the dimensionality of the data down to just 2 dimensions, suitable to be simply plotted. We considered 16 images (8 subjects performing AU17 and 4 subjects performing AU18 and AU25) downsampled to $30 \times 45$ pixels. All images were used to calculate $W_{pca}$ so there’s no distinction between training and test ensembles. The result is shown in Figure 5.2. The different symbols mark the three classes. It’s clear how the the projections for each class are clustered together and sufficiently spread out to allow classification with a simple linear classifier (equivalent to the assumption of a multivariate gaussian distribution for the data).
Figure 5.1: First 8 principal components of the difference images for the upper face actions (top), and lower face actions (bottom). Components are ordered left to right, top to bottom.

Figure 5.2: Projected data for the 3 actions classification example. $W_{opt} = W_{pca} : \mathbb{R}^{1350} \mapsto \mathbb{R}^2$. 
5.2 “Fisher Actions”

This approach is based on the original work by Belhumeur and others [6] that showed that a class-specific linear projection of a principal components representation of faces improved identity recognition performance. The method is based on Fisher’s linear discriminant (FLD) [32], which projects the images into a subspace in which the classes are maximally separated. FLD assumes linear separability of the classes. For identity recognition, the approach relied on the assumption that images of the same face under different viewing conditions lie in an approximately linear subspace of the image space, an assumption which holds true for changes in lighting if the face is modeled by a Lambertian surface [70] [36]. In our dataset, the lighting conditions are fairly constant and most of the variation is suppressed by the logistic filter. The linear assumption for facial expression classification is that the δ-images of a facial action across different faces lie in a linear subspace.

Fisher’s Linear Discriminant is a projection into a subspace that maximizes the between-class scatter while minimizing the within-class scatter of the projected data. Let

\[ \chi \triangleq \{ \chi_1, \chi_2, \ldots, \chi_c \} \]  

be the set of all \( N = |\chi| \) data, divided into \( c \) classes. Each class \( \chi_i \) is composed of a variable number of images \( x_i \in \mathbb{R}^n \). The between-class scatter matrix \( S_B \) and the inter-class scatter \( S_W \) are defined as

\[ S_B \triangleq \sum_{i=1}^{c} |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T \]

and

\[ S_W \triangleq \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \]
where $\mu_i$ is the mean image of class $\chi_i$ and $\mu$ is the mean of all data. The optimal transformation is given by the matrix $W_{opt}$ and is such that [6]

$$W_{opt} : \mathbb{R}^n \mapsto \mathbb{R}^{c-1}$$

and

$$W_{opt} = \arg\max_W J(W)$$

$$= \arg\max_W \frac{\det(W^T S_B W)}{\det(W^T S_W W)}$$

$$= \{w_1, w_2, \ldots, w_{c-1}\}$$

The $\{w_i\}$ are the solution of the generalized eigenvalues problem

$$S_B w_i = \lambda_i S_W w_i \quad \text{for} \quad i = 1, \ldots, c-1$$

Performing first PCA on the total scatter matrix $S_T = S_W + S_B$ to project the feature space to $\mathbb{R}^{N-c}$ greatly simplifies the calculations [6]. Calling $W_{pca}$ the matrix of the transformation,

$$W_{pca} \triangleq \arg\max_W |W^T S_T W|$$

we can project $S_W$ and $S_B$:

$$\tilde{S}_B \triangleq W_{pca}^T S_B W_{pca}$$

and

$$\tilde{S}_W \triangleq W_{pca}^T S_W W_{pca}$$

The original FLD problem can thus be reformulated as:

$$W_{fld} = \arg\max_W J(W)$$

$$= \arg\max_W \frac{\det(W^T \tilde{S}_B W)}{\det(W^T \tilde{S}_W W)}$$

$$= \{w'_1, w'_2, \ldots, w'_{c-1}\}$$
From (5.5) and (5.9)

\[ W_{opt} = W_{pca} W_{fld} \]  \hspace{1cm} (5.10)

and the \( \{w'_i\} \) can now be calculated using

\[ \bar{S}_W^{-1} \bar{S}_B w'_i = \lambda_i w'_i \]  \hspace{1cm} (5.11)

where \( \bar{S}_W \) is now full-rank.

### 5.2.1 Classification of 3 actions.

As we did in Section 5.1.1 we want to evaluate the performance of the FLD algorithm in the problem of classifying just 3 actions. As before we have: \( |\chi_{17}| = 8, \ |\chi_{18}| = 4, \ |\chi_{25}| = 4 \) and \( n = 1350 \). This time though \( W_{opt} = W_{pca} W_{fld} \). The results are shown in Figure 5.3. Even though the resolution of the plot doesn’t show it clearly, what appears as a single symbol is actually a cluster like the one in Figure 5.2, showing that the inter-class spread is virtually eliminated at the cost of a somewhat smaller between-class spread (notice the \( x \) and \( y \) scale of the two plots). A linear classifier is still able to perfectly separate the three classes.

### 5.3 Independent Component Analysis

Representations such as “Eigenfaces” [75], “Holons” [17] and “Local Feature Analysis” [62] are based on principal components analysis and are optimally matched to the second-order statistics of the image set, the pixelwise covariances, but are insensitive to the high-order statistics of the image set. In a task such as facial expression analysis, much of the relevant information may be contained in the high-order relationships among the image pixels. Independent component analysis (ICA) provides an image representation that is sensitive to higher order statistical dependencies in the image set, not just the covariances.
Figure 5.3: Projected data for the 3 actions classification example. $W_{opt} = W_{pca}W_{flu} : \mathbb{R}^{1350} \rightarrow \mathbb{R}^2$.

### 5.3.1 ICA vs. PCA

The difference between ICA and PCA is illustrated as follows. Consider a set of data points derived from two underlying distributions as shown in Figure 5.4. Principal component analysis encodes second order dependencies in the data by rotating the axes to correspond to directions of maximum covariance. PCA models the data as a multivariate Gaussian and would place an orthogonal set of axes such that the projections of the two distributions would be completely overlapping. Independent component analysis does not constrain the axes to be orthogonal, and attempts to place them in the directions of statistical dependencies in the data. Each weight vector in ICA attempts to encode a portion of the dependencies in the input, so that the dependencies are removed from between the elements of the output. The projection of the two distributions onto the ICA axes would have less overlap, and the output distributions of the two weight vectors would be
5.3.2 Bell and Sejnowski’s Algorithm

Bell and Sejnowski’s ICA algorithm is an unsupervised learning rule that was derived from the principle of optimal information transfer through sigmoidal neurons [50, 7]. Consider the case of a single input, $x$, and output, $y$, passed through a nonlinear squashing function, $g(\cdot)$.

\[
y = g(u) \triangleq \frac{1}{1 + e^{-u}}
\]
\[
u = wx + w_0
\]

In which the input $x$ is multiplied by a weight $w$ and added to a bias weight $w_0$. As illustrated in Figure 5.5, the optimal weight $w$ on $x$ for maximizing information transfer is the one that best matches the probability density of $x$ to the slope of the

---

1The kurtosis is the 4th order standardized cumulant of a probability distribution function.
nonlinearity. The optimal $w$ produces the flattest possible output density, which in other words, maximizes the entropy of the output.

The optimal weight is found by stochastic gradient ascent on the entropy of the output, $y$ with respect to $w$. When there are multiple inputs $X$ and outputs $Y$, maximizing the joint entropy of the output “encourages” the individual outputs to move towards statistical independence. When the form of the nonlinear transfer function $g(\cdot)$ is the same as the cumulative density functions of the underlying independent components (up to scaling and translation) it can be shown that maximizing the mutual information between the input and the output also minimizes the mutual information between the inputs $u_i$ [57, 8]. Many natural signals, such as sound sources, have been shown to have a super-Gaussian distribution, meaning that the kurtosis of the probability distribution exceeds that of a Gaussian [7]. For mixtures of super-Gaussian signals, the logistic transfer function has been found to be sufficient to separate the signals [7].

Figure 5.5: Optimal information flow in sigmoidal neurons. Left: The input $x$ is passed through a nonlinear function, $g(x)$. The information in the output density $f_y(y)$ depends on matching the mean and variance of $f_x(x)$ to the slope and threshold of $g(x)$. Right: $f_y(y)$ is plotted for different values of the weight, $w$. The optimal weight, $w_{opt}$ transmits the most information. Figure from Bell & Sejnowski(1995).
The update rule for the weight matrix, $W$, for multiple inputs and outputs is given by [8]

$$\Delta W = (I + y'u^T)W$$ (5.13)

where

$$y' = \frac{\partial}{\partial y_i} \frac{\partial y_i}{\partial u_i} = \frac{\partial}{\partial u_i} \ln \frac{\partial y_i}{\partial u_i}$$ (5.14)

We employed the logistic transfer function

$$g(u) \triangleq \frac{1}{1 + e^{-u}}$$ (5.15)

giving

$$y' = (1 - 2y_i)$$ (5.16)

The algorithm includes a “sphering” step prior to learning [8]. The row means are subtracted from the dataset, $X$, and then $X$ is passed through the zero-phase whitening filter, $W_z$, which is twice the inverse square root of the covariance matrix:

$$W_z \triangleq 2(XX^T)^{-\frac{1}{2}}.$$ (5.17)

This removes both the first and the second-order statistics of the data; both the mean and covariances are set to zero and the variances are equalized. The full transform from the zero-mean input was calculated as the product of the sphering matrix and the learned matrix

$$W_I = WW_z$$ (5.18)

The pre-whitening filter in the ICA algorithm has the Mexican-hat shape of retinal ganglion cell receptive fields which remove much of the variability due to lighting [8].
5.3.3 The Image Representation

The independent component representation was obtained by performing “blind separation” on the set of face images [3] [2]. The independent components of the face images were separated according to the image synthesis model of Figure 5.6. The $\delta$-images in $X$ were assumed to be a linear mixture of an unknown set of statistically independent source images $S$, where $A$ is an unknown mixing matrix. The sources were recovered by a matrix of learned filters, $W$, which produced statistically independent outputs, $U$.

Figure 5.6: TOP: Image synthesis model for the ICA representation. BOTTOM: The ICA representation.
The ICA filters, $W$, were found using the Bell and Sejnowski’s algorithm [7]. The ICA outputs, $U$, provided a set of independent basis images for the expression images. These basis images can be considered a set of statistically independent image features, where the pixel values in each feature image were statistically independent from the pixel values in the other feature images. The ICA representation consisted of the coefficients, $b$, for the linear combination of independent basis images, $u$, that comprised each face image $x$ (Figure 5.6, Bottom). These coefficients were obtained from $W^{-1}$ (see [2]).

Examples of the independent components of the expression images are shown in Figure 5.7. The ICA basis images were spatially local, i.e. each component extracted shows only one or few features localized in space as shown in the examples. Two factors contributed to the local property of the ICA basis images: Most of the statistical dependencies were in spatially proximal image locations, and secondly, the ICA algorithm produces sparse outputs [8].

Unlike PCA, there is no inherent ordering to the independent components of the dataset. We therefore selected as an ordering parameter the class discriminability of each component. Let $x_{ij}$ be i-th image of subject j, $\bar{x}$ the overall mean of a coefficient, and $\bar{x}_j$ be the mean for person $j$. The ratio of between-class to within-class variability, $r$, for each coefficient is defined as

$$r \overset{\Delta}{=} \frac{\sigma_{between}}{\sigma_{within}}$$

(5.19)

Where

$$\sigma_{between} \overset{\Delta}{=} \sum_j (\bar{x}_j - \bar{x})^2$$

(5.20)

is the variance of the $j$ class means, and

$$\sigma_{within} \overset{\Delta}{=} \sum_j \sum_i (x_{ij} - \bar{x}_j)^2$$

(5.21)

is the sum of the variances within each class. The first $p$ components selected by class discriminability comprised the independent component representation.
Figure 5.7: Example independent component images of the upper face (top) and lower face actions (bottom).
5.4 Local Feature Analysis (LFA)

Local Feature Analysis (LFA) defines a set of topographic, local kernels that are optimally matched to the second-order statistics of the input ensemble [62]. The kernels are derived from the principal component axes, and consist of “sphering” the PCA coefficients to equalize their variance [1], followed by a rotation to pixel space.

As in the global PCA representation (Section 5.1), we begin with the zero-mean matrix of $\delta$—images, $X$ (5.1), and calculate the principal component eigenvectors $P$ according to $S = PD^TP$. Penev & Atick [62] defined the following set of topographic kernels based on $P$ and $D$, where “topographic” indicates that the kernels are indexed by spatial location:

$$K = PV^TP^T$$  \hspace{1cm} (5.22)

where

$$V = D^{-\frac{1}{2}} =$$

$$= \text{diag}(\frac{1}{\sqrt{\lambda_i}}) \quad i = 1, \ldots, p$$  \hspace{1cm} (5.23)

and $\lambda_i$ are the eigenvalues of $S$. The rows of $K$ contain kernels with spatially local properties. Note that the matrix $V$ is the inverse square root of the principal components covariance matrix. This transform spheres the principal component coefficients (normalizes their output variance to unity) and minimizes correlations in the LFA output. The kernel matrix $K$ transforms $X$ to the LFA output $O$ (see Figure 5.8.1)

$$O = KX^T = \begin{bmatrix} O(x_1) \\ O(x_2) \\ \vdots \\ O(x_N) \end{bmatrix} \quad O(x_j) \in \mathbb{R}^{1 \times n}$$  \hspace{1cm} (5.24)

The original images $X$ can be reconstructed from $O$ by $X^T = K^{-1}O$. 

Figure 5.8: An original δ-image on the left, with its corresponding LFA representation \( O(x) \) on the right.

Figure 5.9: On the left the input covariance corresponding to pixel \( \zeta(13,9) \), on the right the covariance of the output \( C(\zeta, x) \). The filter is now very sparse and the redundancy is almost eliminated.
5.4.1 Sparsification of LFA

LFA produces an $n$ dimensional representation, where $n$ is the number of pixels in the images. Since we have $n$ outputs described by $p \ll n$ linearly independent variables, there are residual correlations in the output. Penev & Atick proposed an algorithm for reducing the dimensionality of the representation by choosing a subset $\mathcal{M}$ of outputs that were as decorrelated as possible. The sparsification algorithm was an iterative algorithm based on multiple linear regression. At each time step, the output point that was predicted most poorly by multiple linear regression on the points in $\mathcal{M}$ was added to $\mathcal{M}$.

Penev & Atick [62] presented methods for image representation, but did not address the application of local feature analysis to image recognition. The sparsification algorithm proposed by Penev & Atick selected a different set of points, $\mathcal{M}$, for each image, which is problematic for recognition. In order to make the representation amenable to recognition, we diverged from their sparsification algorithm by finding a single set of $\mathcal{M}$ points for all images. At each time step, the point with the largest mean reconstruction error across all of the images was added to $\mathcal{M}$.

At each step, the point added to $\mathcal{M}$ is chosen as

$$\arg \max \langle \|O - O^{rec}\|^{2} \rangle$$

(5.25)

where $O^{rec}$ is a reconstruction of the complete output, $O$, using a linear predictor on the subset $\mathcal{M}$ of the outputs $O$. The linear predictor is of the form:

$$\mathcal{Y} = \beta \mathcal{X}$$

(5.26)

where $\mathcal{Y} = O^{rec}$, $\beta$ is the vector of the regression parameters, and $\mathcal{X} = O(\mathcal{M}, N)$. Here $O(\mathcal{M}, N)$ denotes the subset of $O$ corresponding to the points in $\mathcal{M}$ for all $N$ images.  \footnote{$O(\mathcal{M}, N) = O(i,j), \forall i \in \mathcal{M}, \forall j = 1, \ldots, N.$}
\( \beta \) is calculated from:

\[
\beta = \frac{\mathcal{Y} \mathcal{X}}{(\mathcal{X}^T \mathcal{X})} = \frac{(O^{\text{vec}})^T O(\mathcal{M}, N)}{O(\mathcal{M}, N)^T O(\mathcal{M}, N)}
\]  
\[
\text{(5.27)}
\]

Equation (5.27) can also be expressed in terms of the correlation matrix of the outputs, \( C = O^T O \), as in [62]:

\[
\beta = C(\mathcal{M}, N)C(\mathcal{M}, \mathcal{M})^{-1}
\]  
\[
\text{(5.28)}
\]

The termination condition can be either \( |\mathcal{M}| = N \) or \( \|O - O^{\text{vec}}\|^2 \leq \varepsilon \). We calculated all possible \( N \) points and then tested for optimal classification performance. Figure 5.10 shows the locations of the points selected by the sparsification algorithm. The algorithm converges to the least-squares minimum error. If \( K \) is calculated retaining all eigenvectors then the reconstruction is perfect and the error for \( |\mathcal{M}| = N \) is exactly zero.

The final representation is given by the set of values of image intensity at the points in \( \mathcal{M} \) and is therefore of size \( N \ll n \).
Figure 5.10: The first 155 points selected by the sparsification algorithm superimposed to the mean images of the upper and lower face actions.
Chapter 6

LOCAL REPRESENTATIONS

The approaches described so far were all “global” meaning that the kernels for the representation were derived from the entire image. We explored five different kinds of local representations based on filters that act on small spatial regions within the images. Three of the filters are based on PCA, whereas one relies on a biologically inspired wavelet decomposition.

6.1 Gaussian kernel

A simple benchmark for the local filters consisted of a single Gaussian kernel. δ-images were convolved with a $15 \times 15$ Gaussian kernel and the output was downsampled by 0.25. The dimensionality of the final representation was thus $\frac{n}{4}$.

6.2 Local PCA: Random Patches

A local representation based on the principal components of subimage patches (local PCA) outperformed the representation based on the principal components of the full images (global PCA) for classifying facial expressions [61]. Local basis functions were obtained from the principal component eigenvectors of image patches selected from random image locations. These results were supported by
[34] for lipreading. A set of more than 7000 patches of size $15 \times 15$ was taken from random locations in the $\delta$-images and decomposed using PCA. The first $p$ principal components were then used as convolution kernels to filter the full images. The outputs were subsequently downsampled by a factor of 4, such that the final dimensionality of the representation was isomorphic to $\mathbb{R}^{p \times n/4}$. The local PCA filters obtained from the set of lower-face $\delta$-images are shown in Figure 6.1. Principal component analysis of randomly selected image patches, in which the image statistics are stationary over the patch, describes the amplitude spectrum of the patches [31] [66].

### 6.3 Local PCA: Fixed Patches

Instead of performing PCA on patches selected from random locations in the images, we divided the images into $m \ll \frac{n}{4}$, $15 \times 15$ fixed regions and calculated the principal components of each region separately. Each image was thus represented by $p \times m$ coefficients. The final representation consisted of $p = 10$ principal components of $m = 48$ image regions. The first three principal component vectors of the $15 \times 15$ fixed subregions are shown in Figure 6.3.

### 6.4 Gabor wavelet representation

We next investigated classification performance with an image representation based on the outputs of local filters based on the Gabor wavelet representation [58]. These filters closely model the receptive field properties of cells in the primary visual cortex [65] [42] [20] [19]. Such filters remove most of the variability in images due to variation in lighting and contrast. Representations of faces based on Gabor wavelets have proven successful for recognizing facial identity in images [48].

Given an image $I(x)$ (where $x = (x, y)$), the transform $\mathcal{J}$ is defined as a
Figure 6.1: The 144 principal components of 7750 random patches extracted from all the $\delta$-images.

Figure 6.2: Enlargement of the first 10 principal components of Figure 6.1. Components are ordered left to right, top to bottom.
Figure 6.3: Local principal components of fixed $15 \times 15$ patches of the upper face images. From top to bottom: Principal components 1–3.
convolution

\[ \mathcal{J}_i = \int \mathcal{I}(x)\psi_i(x - x')d^2x' \]

\[ = a_i e^{i\phi_i} \]  

(6.1)

with a family of Gabor kernels \( \psi_i \):

\[ \psi_i(x) = \frac{\| \mathbf{k}_i \|^2}{\sigma^2} e^{-\frac{\| \mathbf{k}_i \|^2}{2\sigma^2}} \left[ e^{i\mathbf{k}_i x - \frac{x^2}{\sigma^2}} \right] \]  

(6.2)

Each \( \psi_i \) is a plane wave characterized by the vector \( \mathbf{k}_i \) enveloped by a Gaussian function, where the parameter \( \sigma = 2\pi \) determines the ratio of window width to wavelength. The first term in the square brackets determines the oscillatory part of the kernel, and the second term compensates for the DC value of the kernel [48],

\[ \mathbf{k}_i = \begin{pmatrix} f_\nu \cos \varphi_\mu \\ f_\nu \sin \varphi_\mu \end{pmatrix} \]  

(6.3)

where

\[ f_\nu = 2^{-\frac{\nu + 2}{2}} \pi, \quad \varphi_\mu = \mu \frac{\pi}{8} \]

The parameters \( \nu \) and \( \mu \) define the frequency and orientation of the kernels. We used 5 frequencies (\( \nu = 0 - 4 \)) and 8 orientations, (\( \mu = 1 - 8 \)) in the final representation, as in [48]. The Gabor filters were applied to the \( \delta \)-images. The outputs \( \{ \mathcal{J}_i \} \) of the 40 Gabor filters were downsampled by a factor \( q \) to reduce the dimensionality to \( 40 \times \frac{n}{q} \), and normalized, which performed a divisive contrast normalization. (See Figure 6.4 for an example.) We tested the performance of the system using \( q = 1, 4, 16 \) and found that \( q = 16 \) yielded the best generalization rate. To determine which frequency ranges contained more information for action classification, we reran the tests using subsets of high frequencies (\( \nu = 0, 1, 2 \)), and low frequencies, (\( \nu = 2, 3, 4 \)) which showed that the high frequencies contain information more relevant to the expression recognition task than the lower frequency band.
6.5  *PCA Jets*

The principal components of random subimage patches are related to the Gabor representation in that they provide the amplitude spectrum of local image regions. In order to further understand the Gabor representation we developed a new representation that endowed local PCA with the multidimensionality and hierarchical properties of the Gabor wavelets (see Figure 6.5). Instead of doing PCA on random patches of fixed size, we chose five patch sizes to match the Gaussian enveloping the plane wave in each Gabor kernel. Patch sizes were chosen as ±3σ, yielding the following set: [9×9, 15×15, 23×23, 35×35, and 49×49]. The number of filters was matched to the Gabor representation by retaining 16 principal components at each scale, for a total of 80 filters. The filter outputs were downsampled as in the Gabor representation.
Figure 6.4: Top: Original δ-image. Bottom two rows, from left to right: Gabor kernels (low and high frequency), the imaginary part and magnitude of the filtered image.

Figure 6.5: PCA Jets. Left: two kernels corresponding to low and high frequencies (patches size 49 × 49 and 9 × 9). Right: the result of the convolution with the δ-image of Figure 6.4.
Chapter 7

PERFORMANCE

7.1 Human subjects

7.1.1 Naive subjects

A benchmark for the performance of the image classification systems was provided by the performance of naive human subjects on the same set of images. Subjects were ten adult volunteers with no prior knowledge of facial expression measurement. The task was a paper and pencil task in which images of faces were printed using a high resolution HP Laserjet 4si printer with 600 dpi. Face images were cropped and scaled identically to how they had been presented to the automated classification systems. The upper and lower face actions were tested separately. Subjects were provided with a guide sheet containing an example image of each of the six actions along with a written description of each action and a list of image cues important for detecting and discriminating the actions from [25]. Each subject was given a training session in which the facial actions were described and demonstrated, and then the image cues listed on the guide sheet were reviewed and indicated on the example images. The subjects kept the guide sheet as a reference during the task.

Face images were presented in pairs, with the null image and the test image
presented side by side. Subjects were instructed to compare the test image with the null image and decide which of the actions the subject had performed in the test image. Ninety-three image pairs were presented in both the upper and lower face tasks, including low, medium, and high magnitude examples of each action. Subjects were allowed to take as much time as they needed to perform the task, which ranged from 30 minutes to one hour.

7.1.2 Expert coders

A second benchmark for the performance of the image classification systems was to compare it to that of trained FACS experts, since the objective is to ultimately replace the human experts. Subjects were four certified FACS coders. The task was a paper and pencil task as above, with the following exceptions: Expert subjects were not given a guide sheet or additional training, and performance of the experts was measured for images in which the complete face was visible, as it would normally be during FACS scoring.

The face images used in the task were cropped to display the full face from the top of the forehead to the bottom of the chin. Images were scaled to $125 \times 100$ pixels, with 45 pixels between the eyes, matching the spatial resolution of the upper-face images presented to the computational algorithms. One hundred and fourteen upper-face image pairs and ninety-three lower-face image pairs were presented. Subjects were instructed to take as much time as needed to complete the scoring, which ranged from 20 minutes to 1 hour and 15 minutes. Because the images were originally labeled by two expert coders with access to stop-motion video, the performance of the expert subjects provided a measure of inter-coder agreement for coding static images and stop-motion video.
7.2 Classification Procedures

The output of each image analysis algorithm produced a feature vector that was used for classification. Although we use the term “feature vector” it should be emphasized that the features were not necessarily local in nature. We employed a simple nearest neighbor classifier in which the similarity $S$ of a training feature vector, $f^t$, and a novel feature vector, $f^n$, was measured as the cosine of the angle between them:

$$S(f^n, f^t) = \frac{\langle f^n, f^t \rangle}{\|f^n\| \cdot \|f^t\|} \in [-1, 1]$$ (7.1)

The algorithms were trained and tested using leave-one-out cross-validation, also known as the jack-knife procedure, which makes maximal use of the available data for training. In this procedure, the image representations were calculated multiple times using data from all but one subject, where all of the images of that subject were reserved for testing. This procedure was repeated for each of the 20 subjects, and mean classification accuracy was calculated across all of the test cases.

The image representations were calculated using low, medium, and high magnitude facial actions. Classification performances are presented for medium magnitude facial actions, the fourth frame in each sequence. This choice was made after different tests comparing performance for each magnitude. To help understand why the medium intensity is more informative we can go back to the simple example of classification of 3 actions (Section 5.1.1). Figure 7.1 shows the projected data for each magnitude and the projection of the means of all images. The plots show how the first two $\delta$-images are not informative enough to correctly classify, while the others contain almost the same information. This example fails to show —though— that there are other actions where the images of the highest intensity are less distinguishable (i.e. the highest magnitude images are more similar across subjects than across actions) and therefore the use of medium intensity images is
more effective.

Classification performance was also evaluated using Euclidean distance instead of cosine as the similarity measure and template matching instead of nearest neighbor as the classifier, where the templates consisted of the mean feature vector for the training images. The results shown in Table 7.1 are for the similarity measure and classifier that gave best performance, as presented below.

7.3 Optic Flow Analysis

Flow fields were centered on the third image in the sequence, where the facial action is at medium magnitude. For the gradient-based algorithm, flow fields were calculated between the neutral image and the third image. Best performance was obtained using the cosine similarity measure and template matching.

The correlation-based flow algorithm outperformed the fast gradient-based algorithm for image classification, with 85.6% and 55.8% correct classification performances, respectively. There was a substantial difference in processing speed between the two algorithms. The correlation based method required 57 seconds to calculate a single flow field on a Dec Alpha 2100a processor compared to 0.25 seconds for the gradient based method. The addition of spatial smoothing after the first step in the correlation-based flow fields did not improve performance, and instead degraded it to 53.1%. Classification of the correlation-based optic flow algorithm is comparable to the performance of other facial expression recognition systems based on optic flow (e.g. [79] [68] See Table 8.2).

7.4 Holistic Spatial Analysis

Principal Component Analysis. Best performance with the holistic principal component representation, 79.3% correct, was obtained with the first 30 principal components, using Euclidean distance and template matching. In some previous studies, classification performance was improved by discarding the first
Figure 7.1: Projected data for the 3 actions classification example. From left to right and top to bottom the data for different magnitudes. Bottom right shows the projection of the means of all images. Notice how the plot corresponding to Delta 4 is the same as Figure 5.2.
few principal components (c.f. [6]). For this dataset, discarding the first one and two principal components degraded performance.

**Fisher’s Linear Discriminant.** The dimensionality of the images was first reduced by selecting the first 30 principal components, and these were then projected down to 5 dimensions via the projection matrix, $W_{fd}$. Best performance of 75.7% correct was obtained with Euclidean distance and template matching. Classification performance was not improved over that obtained with the first 30 principal components by projecting into this low dimensional space. These results are consistent with other reports of poor generalization to novel subjects [13]. Good results have only been obtained with this technique when other images of the test subjects were used to calculate the projection matrix [6]. The low dimensionality appears to provide insufficient degrees of freedom for linear discrimination between classes of face images [13].

**Independent component analysis.** Independent component analysis performed the best of the three holistic representations. Note, however, that the independent component images in Figure 5.7 were local in nature. While the ICA algorithm analyzed the images as whole, the basis images that the algorithm settled upon were local. Best performance of 95.5% was obtained with the first 75 components selected by class discriminability, using the cosine similarity measure, and nearest neighbor classifier.

**Local Feature Analysis.** The local feature analysis representation [62] attained 81.1% correct classification using the first 155 kernels selected by the sparsification algorithm, using the cosine similarity measure and nearest neighbor classifier. LFA gave the same classification performance as global PCA.

### 7.5 Local Analysis

**Gaussian Kernel.** A simple benchmark for the local filters consisted of a single $15 \times 15$ Gaussian kernel, subsampled by 0.25. The output of this basic local filter was classified at 70.3% accuracy using Euclidean distance and template
matching.

**PCA Random.** The second set of local filters examined consisted of the principal component vectors of a set of 7000 15 × 15 subimage patches selected from random locations about the δ-images. Images were filtered using the first $p$ principal components, and the outputs were subsampled by a factor of 4. Performance improved by excluding the first principal component. Best performance of 73.4% was obtained with principal components 2–30, using Euclidean distance and template matching. Unlike the results obtained in [61], local PCA of random patches did not outperform global PCA. The difference in performance was not statistically significant.

**PCA Fixed.** The representation based on PCA of fixed 15 × 15 patches gave similar classification performance to that based on PCA of random patches. Classification performance was tested using up to the first 30 components of each patch. Best performance of 78.3% was obtained with the first 10 principal components of each image patch, using Euclidean distance and the nearest neighbor classifier. Performance with the local principal components of fixed patches was comparable to that with the global PCA representation.

**Gabor filters.** Best performance of 95.5% was obtained with the Gabor filter representation using a the cosine similarity measure and nearest neighbor classifier. This finding is supported by Zhang, Yan, & Lades [81] who found that face recognition with the Gabor filter representation was superior to that with a holistic principal component based representation.

To determine which frequency ranges contain more information for action classification, we reran the classification tests using only a subset of frequencies from the Gabor filter representation. Using high frequencies only ($\nu = 0, 1, 2$) the performance of 92.8% was almost the same as for $\nu = 0, \ldots, 4$, and was significantly higher (∼10% greater) than 83.8% yielded by the low frequencies only ($\nu = 2, 3, 4$). The finding that the higher spatial frequency bands of the Gabor filter representation contain more information than the lower frequency bands is consistent with our analysis of optic flow, above, in which reduction of the spatial resolution of the
optic flow through smoothing had a deleterious effect on classification performance. It appears that high spatial frequencies are important for this task.

**PCA jets.** We next investigated whether the multiscale property of the Gabor wavelet representation accounts for the difference in performance obtained using the Gabor representation and the local PCA representation. To test this hypothesis, we developed the multiscale version of the local PCA representation, PCA jets. A multiscale local PCA representation was obtained by performing PCA on random image patches at five different scales. Sixteen principal components were retained at each scale to match the number of filters in the Gabor jets. As for the Gabor representation, performance was tested using the cosine similarity measure and nearest neighbor classifier. Best results were obtained using eigenvectors 2 to 17 for each patch size. Performance was 64.9% for all four scales, 72.1% for the two smaller scales, and 62.2% for the two larger scales. The multiscale principal component analysis (PCA jets) did not improve performance over the single scale local PCA. The multiscale property of the Gabor representation does not account for the improvement in performance obtained with this representation over local representations based on principal component analysis.

**Error Analysis.** Classification errors were examined for the three top performing algorithms: correlation-based flow, ICA, and Gabor wavelets. Only one image was misclassified by all 3 algorithms. AU7 was consistently misclassified as AU6 for one subject. All of the expert coders classified this image as an AU7. There was otherwise a small trend for the algorithms to make errors on the same images. The conditionally dependent error rates for each of the three algorithms, given that an image was misclassified by one of the other algorithms, were all 0.4. Due to the small N, none of these conditionally dependent error rates was significantly higher than chance.
Table 7.1: Best performance for each classifier. FLD: Fisher’s linear discriminant. ICA: Independent component analysis. LFA: Local feature analysis.

**Optic Flow**

<table>
<thead>
<tr>
<th></th>
<th>Gradient</th>
<th>Correlation</th>
<th>Smoothed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>55.8% ± 4.7%</td>
<td>85.6% ± 3.3%</td>
<td>53.1% ± 4.7%</td>
</tr>
</tbody>
</table>

**Holistic Analysis**

<table>
<thead>
<tr>
<th></th>
<th>PCA</th>
<th>FLD</th>
<th>ICA</th>
<th>LFA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>79.3% ± 3.9%</td>
<td>75.7% ± 4.1%</td>
<td>95.5% ± 2.0%</td>
<td>81.1% ± 3.7%</td>
</tr>
</tbody>
</table>

**Local Analysis**

<table>
<thead>
<tr>
<th>Gaussian</th>
<th>PCA</th>
<th>PCA</th>
<th>PCA</th>
<th>Gabor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel</td>
<td>Random</td>
<td>Fixed</td>
<td>Jets</td>
<td>Jets</td>
</tr>
<tr>
<td></td>
<td>70.3 ± 4.3%</td>
<td>73.4% ± 4.2%</td>
<td>78.3% ± 3.9%</td>
<td>72.1% ± 4.2%</td>
</tr>
</tbody>
</table>

**Human**

<table>
<thead>
<tr>
<th></th>
<th>Naive</th>
<th>Expert</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>77.9% ± 2.5%</td>
<td>94.1% ± 2.1</td>
</tr>
</tbody>
</table>


Chapter 8

DISCUSSION AND CONCLUSIONS

8.1 Discussion of the new results

We have compared a number of different image analysis methods on a difficult classification problem, the classification of facial actions. Several of these approaches to facial expression analysis have been presented in the literature, but until now, there has been little direct comparison of these methods on a single dataset. These approaches include analysis of facial motion [54] [79] [68] [30], holistic spatial pattern analysis using techniques based on principal components analysis [16] [61] [49], and measurements of the shapes and facial features and their spatial arrangements [49]. This investigation compared facial action classification using optic flow, holistic spatial analysis, and local spatial representations. We also included in our comparison a number of representations that had been developed for facial identity recognition, and applied them for the first time to facial expression analysis. These representations included Gabor filters [48], Linear Discriminant Analysis [6], Local Feature Analysis [62], and Independent Component Analysis [2]. Finally, we developed new representations especially for this task, like the fixed local PCA, band-limited Gabor jets and the PCA jets.
Best performances were obtained with the local Gabor filter representation, and the Independent Component representation, which both achieved 96% correct classification. The performance of these two methods equaled the agreement level of expert human subjects on these images. The representations derived from the second-order statistics of the dataset (PCA and LFA) performed about as well as naive human subjects on this image classification task, in the 80% accuracy range. Correlation-based optic flow performed at a level between the two, at 86%. These results compare favorably with other systems developed for emotion classification, summarized in Table 8.2.

We obtained converging evidence that local spatial filters are important for analysis of facial expressions. The two representations that gave by far the best performance were based on local filters, the Gabor representation [48] and the Independent Component representation [2]. ICA was classified as a holistic algorithm, since the analysis was performed over the images as a whole. The basis images that the algorithm produced, however, were local. Our results also demonstrated that spatial locality of the image filters alone is insufficient for good classification. Local principal component representations such as LFA and PCA of subimage patches performed no better than the global PCA representation (Eigenfaces). This finding differs from [61], in which representations based on local basis functions from the principal components of local image patches outperformed global PCA. The success of the global principal component analysis in this implementation could be attributable to reduced variability due to the use of difference images, or to the smaller original image size than that in [61], such that 29 principal components accounted for a greater percentage of the variability. In addition, we employed a region of interest analysis, in which images were cropped to contain half of the face, which is similar to the “Eigenfeature” approach that gave Padgett & Cottrell better performance than global-PCA.

The ICA representation performed as well as the Gabor representation, despite a two order of magnitude difference in the number of basis functions. A large number of basis functions does not appear to confer an advantage for classification.
The PCA-jet representation, which was matched to the Gabor representation for number of basis functions as well as scale, performed at only 72% correct.

In addition to spatial locality, the ICA representation and the Gabor filter representation share the property of redundancy reduction. Relationships have been demonstrated between Gabor filters and statistical independence. Bell & Sejnowski [8] found that the filters that produced independent outputs from natural scenes were spatially local, oriented edge filters, similar to a bank of Gabor filters. It has also been shown that Gabor filter outputs of natural images are pairwise independent in the presence of divisive normalization similar to the length normalization in our representation [71].

The ICA representation also captures phase information. Spatial phase contains the structural information in images that drives human recognition much more strongly than the amplitude spectrum [59] [64]. A face image synthesized from the amplitude spectrum of face A and the phase spectrum of face B will be perceived as an image of face B. The pixelwise covariances in the image set correspond to the amplitude spectrum, but not the phase spectrum, whereas the high-order statistics contain the phase information [8]. Principal component-based representations therefore contain only the amplitude spectrum of the images, whereas the independent component representation is sensitive to both amplitude and phase.

The Gabor wavelets, PCA, and ICA each provide a way to represent face images as a linear superposition of basis functions. PCA models the data as a multivariate Gaussian, and the basis functions are restricted to be orthogonal [51]. ICA allows the learning of non-orthogonal bases and allows the data to be modeled with non-Gaussian distributions [15]. As noted above, there are a number of relationships between Gabor wavelets and the basis functions obtained with ICA. The Gabor wavelets are not specialized to the particular data ensemble, but would be advantageous when the number of data samples is small.

We also obtained two independent sources of evidence that high spatial frequencies are important for classifying facial actions. Spatial smoothing of optic flow degraded performance by more than 30%. Secondly, classification with only
the high frequencies of the Gabor representation was superior to classification using only the low spatial frequencies. A similar result was obtained with the PCA jets.

Another interesting finding was that contrary to the results obtained in [6], Fisher's Linear Discriminants did not improve upon classification with PCA, despite providing a much more compact representation of the data that optimized linear discrimination. This suggests that the linear subspace assumption was violated more catastrophically for our dataset than for the dataset in [6] which consisted of faces under different lighting conditions. Another reason for the difference in performance may be due to the problem of generalization to novel subjects. The FLD method achieved the best performance on the training data (close to 100%) but generalized poorly to new individuals. This is consistent with the findings of Chellappa [13] (also of H. Wechsler, personal communication) who reported that the FLD method performs well for novel images of subjects that were included in the training set, but poorly for subjects that were entirely novel. The limitation may be that FLD projects the data to a dimensionality that is too low. Class discriminations that are approximately linear in high dimensions may not be linear when projected down to as few as 5 dimensions.

Similarly, classification based on local feature analysis [62] also did not improve on performance with PCA representations. LFA was developed for image representation and compression, and has not been adapted for recognition. One of the authors of the method (J. Atick) obtained very high performance on the FERET face recognition test [63], but the recognition algorithm used for the test has not been disclosed. Although he has indicated that the recognition algorithm employed many of the same concepts as LFA, it should be noted that LFA was developed after the FERET competition.

Naive human subjects classified the facial actions at approximately the same accuracy as representations based on spatially global filters such as global-PCA, whereas the expert human subjects performance more closely matched that using spatially local filters. This supports other evidence of a shift in visual processing
strategies with familiarity and expertise, from configurational processing based on external features, to local processing based on internal features [80].

The image analysis presented here made use of a neutral frame. For applications in which neutral images are unavailable, principal component analysis could be performed on the original graylevel images. Methods based on principal component analysis have successfully classified static graylevel images of facial expressions [61]. The image analysis also required localization of the face in the image. For this study, the localization was carried out by making three mouse clicks, one at the center of each eye and one at the center of the mouth, in the first frame of the sequence. All other aspects of the systems were fully automated. Highly accurate eye and mouth location algorithms are available [9], and automating this step is a realistic option. The image alignment procedure ignored out-of-plane rotations. Out-of-plane rotations could be handled by methods for estimating the frontal view of a face from a nonfrontal view [10] [77].

8.2 Conclusions

The results of this comparison provided converging evidence for the importance of possessing local filters, high spatial frequencies, and statistical independence for classifying facial actions. The relevance of high spatial frequencies has implications for motion-based facial expression analysis. Since optic flow is a noisy measure, most flow-based expression analysis systems employ regularization procedures such as smoothing and quantizing to estimate a principal direction of motion within an image region. The motion in facial expression sequences is nonrigid and can be highly discontinuous due to the formation of wrinkles. It appears that smoothing across such boundaries is disadvantageous.

The majority of the approaches to facial expression recognition by computer have focused exclusively on analysis of facial motion. It should be noted that while human subjects can recognize facial expressions from motion signals alone [5], recognition rates are just above chance, and much lower than for static graylevel
images. In this comparison, we found that best performance was obtained with representations based on surface graylevels. A future direction of this work is to combine the best motion classifiers with the best classifiers for static graylevel images. Perhaps combining motion and graylevel information will ultimately provide the best facial expression recognition performance, as it does for human subjects [78].
<table>
<thead>
<tr>
<th>System</th>
<th>No. Faces</th>
<th>No. Classes</th>
<th>Method</th>
<th>Test Image Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Present Results</td>
<td>20</td>
<td>12&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Gabor filters, Independent components, Optic flow</td>
<td>96% 96% 86%</td>
</tr>
<tr>
<td>Mase [54]</td>
<td>1</td>
<td>4</td>
<td>Optic flow</td>
<td>86%</td>
</tr>
<tr>
<td>Yacoob &amp; Davis, [79]</td>
<td>32</td>
<td>7</td>
<td>Optic flow</td>
<td>87%</td>
</tr>
<tr>
<td>Kobayashi et al. [47]</td>
<td>30</td>
<td>6</td>
<td>Feedforward neural network, taking pixel graylevels as input</td>
<td>90%</td>
</tr>
<tr>
<td>Rosenblum, et al. [68]</td>
<td>32</td>
<td>2</td>
<td>Radial basis functions; optic flow</td>
<td>88%</td>
</tr>
<tr>
<td>Padgett &amp; Cottrall [61]</td>
<td>12</td>
<td>6&lt;sup&gt;b&lt;/sup&gt;</td>
<td>PCA of static graylevel images</td>
<td>86%</td>
</tr>
<tr>
<td>Essa &amp; Pentland [30]</td>
<td>8</td>
<td>5</td>
<td>Parameterized face model; optic flow</td>
<td>98%</td>
</tr>
<tr>
<td>Lanitis et al. [49]</td>
<td>30</td>
<td>7</td>
<td>Adaptive local features; PCA of static graylevel images</td>
<td>74%</td>
</tr>
<tr>
<td>Cohn et al. [14]</td>
<td>30</td>
<td>15&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Feature tracking based on optic flow</td>
<td>86%</td>
</tr>
</tbody>
</table>

Table 8.1: Summary of automated facial expression analysis systems. These models were tested on different data sets with differing levels of difficulty. Classification categories were feigned expressions of emotion, except where otherwise indicated. <sup>a</sup>Classified facial actions <sup>b</sup>Classified images from Pictures of Facial Affect (Ekman & Friesen, 1976) in which trained actors performed the muscle contractions empirically associated with certain emotional states.
Appendix A

SELECTED MATLAB CODE

Optic Flow

This is the code for the optic flow extraction. It includes both Singh’s method and the gradient-based method. It is fully configurable since most of the parameters in the algorithms can be specified when invoking the function. For example the size of the windows in the correlation-based method can be modified by specifying the parameters $n$ and $N$. The code provides also pre- and post-processing. Preprocessing included two different lowpass filters and two bandpass filters. These were included in development phase but turned out to be irrelevant during the real tests. Postprocessing tries to reduce the field to keep only the most informative part. To do this we used the confidence measure associated to the flow estimate, or simply set to zero the flow at points where it’s below a threshold or $2\sigma$ below the flow average. The outputs are the flow fields in $x$ and $y$ direction $U_{cc}$ and $V_{cc}$ respectively, together with the corresponding confidence measures $C_1$ and $C_2$.

function $[U_{cc},V_{cc},C_1,C_2]=...$

singh_flow(img1,img2,img3,prep,flow_type,n,N,nn,post,display)

% Gianluca Donato, February 1997
% Computational Neurobiology Lab, The Salk Institute.
Optical flow extraction using A. Singh’s algorithm


Input is sequence of 3 images (ideally 1 time step apart from each other)
Img1, img2, img3 stored as matrices.

INPUT ARGUMENTS: (I haven’t put any smart error detection so make sure you provide all of them)

prep is a string specifying preprocessing. options are:
'pyra' = 1st step of Burt’s Laplacian pyramid BP filtering
'gau3' = 3x3 gaussian LP filter
'gau5' = 5x5 gaussian LP filter
'spec' = BP filter .15-.45 Hz band
'none' = no prefiltering

flow_type is a string specifying the algorithm for the flow extraction:
'cons' = only conservation information extraction
'neig' = conservation info and neighborhood information extraction
'grad' = simple gradient based algorithm (uses only img1 and img3)

2n+1 is the size of the correlation window
2N+1 is the size of the search window
2nn+1 is the size of the neighborhood used in the second step
% post specifies type of postprocessing
% 'con' = postprocessing using confidence measures
% 'sma' = " " flow magnitude
% 'sta' = " " flow statistics
% 'all' = " " both previous methods
% 'non' = no postprocessing
%
% display specifies display options:
% 'disp' = outputs plot of final flow
% 'nodi' = no display
%
% Outputs are:
% Ucc, Vcc matrix of flow estimates
% C1, C2 confidence measures

[rows,cols]=size(img2);

k_exp=0.95;       % parameter for computing response distribution Rc

epsilon=1e-3;     % difference in flow between 2 steps of neighbor info
                 % extraction

MAX_STEP=3;       % maximum number of smoothing steps

img1=mat2gray(img1);
img2=mat2gray(img2);
img3=mat2gray(img3);

%if (all(prep=='none'))
% disp('no preprocessing')
If (all(flow_type=='cons'))|all(flow_type=='neig'))
Ucc=zeros([rows,cols]);
Vcc=zeros([rows,cols]);
Scc=zeros(2*([rows,cols]-6));
scc=zeros(2);
Sn=zeros(2*([rows,cols]-6));
sn=zeros(2);
C1=zeros([rows,cols]-6);
C2=zeros([rows,cols]-6);
Rn=[.25 .5 .25]*[.25 .5 .25];
sum_Rn=sum(sum(Rn));  % normalization factor for 2nd pass

% CONSERVATION INFORMATION EXTRACTION
% correlation window Wp is 2n+1 x 2n+1 and search window Ws is 2N+1 x 2N+1
% outputs are response distribution Rc, estimate Ucc and Vcc and covariance
% matrices Scc associated ti each pixel
%-------------------------------------------------------------------
% to avoid boundary effects we skip the first and last rows and cols
%-------------------------------------------------------------------
for i=N+n+1:rows-(N+n)  %loop on all pixels of original image
    for j=N+n+1:cols-(N+n)
        Wp=img2(i-n:i+n,j-n:j+n);  % window Wp 2n+1 x 2n+1
        Ws1=img3(i-(N+n):i+N+n,j-(N+n):j+N+n);  % windows Ws 2N+1 x 2N+1
        Ws_1=img1(i-(N+n):i+N+n,j-(N+n):j+N+n);
        Ec1=zeros(2*+1);      % error distributions
        Ec_1=zeros(2*+1);
        for u=1:(2*+1)
            for v=1:(2*+1)
                Ec1(u,v)=sum(sum((Wp-Ws1(u:2+n+u,v:2+n+v)).^2));
                Ec_1(u,v)=sum(sum((Wp-Ws_1(u:2+n+u,v:2+n+v)).^2));
            end
        end
\begin{verbatim}
Rc1 = exp(-k_exp*Ec1); \% sample response distribution
Rc_1 = exp(-k_exp*Ec_1);
\%
Rc = rot90(Rc_1) + Rc1;
Rc = fliplr(Rc_1) + Rc1;

ucc = 0;
vcc = 0;
for u = -N:N
    for v = -N:N
        ucc = ucc + Rc(u+N+1,v+N+1)*u;
        vcc = vcc + Rc(u+N+1,v+N+1)*v;
    end
end

junk = sum(sum(Rc));
Ucc(i,j) = ucc/junk;
Vcc(i,j) = vcc/junk;

u_sum = 0;
v_sum = 0;
uv_sum = 0;
for u = -N:N
    for v = -N:N
        u_sum = u_sum + Rc(u+N+1,v+N+1)*(u-Ucc(i,j))^2;
        v_sum = v_sum + Rc(u+N+1,v+N+1)*(v-Vcc(i,j))^2;
        uv_sum = uv_sum + Rc(u+N+1,v+N+1)*(u-Ucc(i,j))*(v-Vcc(i,j));
    end
end

scc(1,1) = u_sum/junk;
\end{verbatim}
scc(1,2)=uv_sum/junk;
scc(2,1)=scc(1,2);
scc(2,2)=v_sum;
Scc(2*(i-4)+1:2*(i-4)+2,2*(j-4)+1:2*(j-4)+2)=scc;
dumb=sort(eig(scc).^(-2)); % confidence measures
C1(i,j)=dumb(2);
C2(i,j)=dumb(1);
end
end

if (all(flow_type=='cons')&(all(post=='con')|all(post=='all')))
    m1=mean(C1(:));
    m2=mean(C2(:));
    sdi=std(C1(:));
    sd2=std(C2(:));
    junk=find((C1<(m1-2*sdi))&((C2<(m2-2*sd2))));
    Ucc(junk)=zeros(size(junk));
    Vcc(junk)=zeros(size(junk));
end

if (all(flow_type=='cons')&(all(post=='sma')|all(post=='all')))
    % let's disregard smaller components of the flow
    % and spurious flow at boundary
    Ucc(n+N+1,:)=zeros(1,cols);
    Ucc(rows-(n+N),:)=zeros(1,cols);
    Ucc(:,n+N+1)=zeros(rows,1);
    Ucc(:,cols-(n+N))=zeros(rows,1);
    Vcc(n+N+1,:)=zeros(1,cols);
    Vcc(rows-(n+N),:)=zeros(1,cols);
    Vcc(:,n+N+1)=zeros(rows,1);
    Vcc(:,cols-(n+N))=zeros(rows,1);
mag=(Ucc.^2+Vcc.^2).^5;
sdmag=std([mag(:),-mag(:)])/2;
junk=find(mag<sdmag);
Ucc(junk)=zeros(size(junk));
Vcc(junk)=zeros(size(junk));
end

if (all(flow_type=='cons')&(all(post=='sta')|all(post=='all')))
% let's disregard smaller components of the flow
  mu=mean(Ucc(:));
sdu=std(Ucc(:));
mv=mean(Vcc(:));
sdv=std(Vcc(:));
junk=find(((Ucc>(mu-2*sdu))&(Ucc>(mu+2*sdu)))|...
  ((Vcc>(mv-2*sdv))&(Vcc>(mv+2*sdv))));

  Ucc(junk)=zeros(size(junk));
  Vcc(junk)=zeros(size(junk));
end

if (all(flow_type=='cons')& all(display=='disp'))
  quiver(flipud(Ucc),flipud(Vcc),2);
end
end

% NEIGHBORHOOD INFORMATION EXTRACTION
% Wn is window 2n+1 x 2n+1 around (i,j)
% WU and WV are corresponding windows in (u,v) space
% ucc is vector with velocity estimate from conservation information
% U_  contains the estimates at each step
% At beginning Ucc and Vcc contain only the cons. info. estimate = U_0
% at each step U is the true velocity and U_ the estimate
% -------------------------------------------------------------------
if all(flow_type==’neig’)
    U_cc=Ucc;
    V_cc=Vcc;
    U=zeros(2,1);
    U_=zeros(2,1);
    sn=zeros(2);
    v_change=1+epsilon; step=0;
    while ((v_change > epsilon)&(step<MAX_STEP))
        step=step+1;
        for i=N+2:rows-(N+1)  % loop on all pixels of original image
            for j=N+2:cols-(N+1)
                Wn=img2(i-nn:i+nn,j-nn:j+nn);
                scc=Scc(2*(i-4)+1:2*(i-4)+2,2*(j-4)+1:2*(j-4)+2);
                scc_inv=svdinv(scc);
                %
                scc_inv=inv(scc);
                ucc=[Ucc(i,j),Vcc(i,j)]’;
                WU=Ucc(i-nn:i+nn,j-nn:j+nn);  % this windows must be
                WV=Vcc(i-nn:i+nn,j-nn:j+nn);  % updated at every step
                u_=0;  % estimates for velocity
                v_=0;
                for k=1:(2*nn+1)
                    for l=1:(2*nn+1)
                        u_=u_+Rn(k,l)*WU(k,l);
                        v_=v_+Rn(k,l)*WV(k,l);
                    end
                end
                U_(1)=u_/sum_Rn; U_(2)=v_/sum_Rn;
\[ u_{\text{sum}} = \text{sum}(Rn(:)) \cdot (WU(:) - u_{\_})^2; \]
\[ v_{\text{sum}} = \text{sum}(Rn(:)) \cdot (WV(:) - v_{\_})^2; \]
\[ uv_{\text{sum}} = \text{sum}(Rn(:)) \cdot ((WU(:) - u_{\_}) \cdot (WV(:) - v_{\_})); \]

\[ sn(1,1) = u_{\text{sum}} / \text{sum}_Rn; \]
\[ sn(1,2) = uv_{\text{sum}} / \text{sum}_Rn; \]
\[ sn(2,1) = sn(1,2); \]
\[ sn(2,2) = v_{\text{sum}} / \text{sum}_Rn; \]
\[ Sn(2(i-4)+1:2*(i-4)+2,2*(j-4)+1:2*(j-4)+2) = sn; \]
\[ sn_{\text{inv}} = \text{svdinv}(sn); \]
\[
\% \quad sn_{\text{inv}} = \text{inv}(sn);
\]

\[ U = \text{inv}(scc_{\text{inv}} + sn_{\text{inv}}) \cdot (scc_{\text{inv}} \cdot ucc + sn_{\text{inv}} \cdot U_{\_}); \]
\[ U_{\_cc}(i,j) = U_{\_}(1); \]
\[ V_{\_cc}(i,j) = U_{\_}(2); \]
\[ Ucc(i,j) = U(1); \]
\[ Vcc(i,j) = U(2); \]

\[ \text{dumb} = \text{sort}([\text{norm}(Ucc-U_{\_cc}) \text{ norm}(Vcc-V_{\_cc})]); \]
\[ \text{v\_change} = \text{dumb}(2); \]
\[ \text{fprintf}(1,'step %.0f, v\_change=\%g\n', \text{step}, \text{v\_change}); \]
\[ \text{U} = U; \]

\[ \% \text{while loop} \]

\[ \% \text{compute confidence measures for final velocity} \]

\[ \text{for i}=4:\text{rows}-3 \]
\[ \text{for j}=4:\text{cols}-3 \]
\[ sn = Sn(2(i-4)+1:2*(i-4)+2,2*(j-4)+1:2*(j-4)+2); \]
\[ scc = Scc(2(i-4)+1:2*(i-4)+2,2*(j-4)+1:2*(j-4)+2); \]
dumb = sort(eig(svdinv(sn) + svdinv(scc)));  
\%
\% dumb = sort(eig(inv(sn) + inv(scc)));
C1(i,j) = dumb(2);
C2(i,j) = dumb(1);
end
end

if (all(post == 'all') | all(post == 'con'))
  m1 = mean(C1(:));
  m2 = mean(C2(:));
  sd1 = std(C1(:));
  sd2 = std(C2(:));
  junk = find((C1 < (m1 - 2*sd1) & (C2 < (m2 - 2*sd2))));
Ucc(junk) = zeros(size(junk));
Vcc(junk) = zeros(size(junk));
  end

if (all(post == 'all') | all(post == 'sma'))
  mu = mean(Ucc(:));
  sdu = std(Ucc(:));
  mv = mean(Vcc(:));
  sdv = std(Vcc(:));
  junk = find(((Ucc > (mu - 2*sdu)) & (Ucc > (mu + 2*sdu))) | ...
              ((Vcc > (mv - 2*sdv)) & (Vcc > (mv + 2*sdv))));
Ucc(junk) = zeros(size(junk));
Vcc(junk) = zeros(size(junk));
  end

if all(display == 'disp')
  quiver(flipud(Ucc), flipud(Vcc), 2);
end
end % neighborhood info. extraction

if (all(flow_type=='grad'))
    load /home/luca/matlab/dxk.asc  % 3x3 x derivative filter
    load /home/luca/matlab/dyk.asc  % 3x3 y derivative filter
    load /home/luca/matlab/blr1d.asc % 1x5 gaussian blurring filter

    tmpA = conv2(img1, blr1d, 'same');
    tmpB = conv2(img2, blr1d, 'same');
    inAblr = conv2(tmpA, blr1d, 'same');
    inBblr = conv2(tmpB, blr1d, 'same');

    % temporal blur
    inBblr = 0.5 * (inAblr + inBblr);
    dt = inBblr - inAblr;
    dx = conv2(inAblr, dxk, 'same');
    dy = conv2(inAblr, dyk, 'same');

    vx = dx ./ dt;
    vy = dy ./ dt;

    % disregard locations with dt too small (threshold values can be adjusted):

    [ix,jx] = find((dt < -0.05) | (dt > 0.05));
    dt_cutoff = length (ix);
    sx = ones(1, (length(ix)));
    dtlocs = sparse(ix, jx, sx,rows,cols);
    vxdtth = dtlocs .* vx ;
    vydttth = dtlocs .* vy ;
% thresholding by r:

\[ r = dx.*dx + dy.*dy; \]
\[ [i, j] = \text{find}(r > 0.3); \]  // was: 0.3
\[ \text{edginess\_cutoff} = \text{length}\ (i); \]
\[ s = \text{ones}(1, \text{length}(i)); \]
\[ \text{rthresh\_05} = \text{sparse}(i, j, s, \text{rows}, \text{cols}); \]
\[ \text{Ucc} = \text{vxdtth}.* \text{rthresh\_05}; \]
\[ \text{Vcc} = \text{vydtth}.* \text{rthresh\_05}; \]

\[ C1=0; \ C2=0; \]  // we have no confidence estimates in this case

% the algorithm introduces spurious flow at the borders, let’s discard it
\[ \text{mask}=\text{ones}(\text{rows}, \text{cols}); \]
\[ \text{mask}(\text{ceil}(\text{rows}*.05):\text{ceil}(\text{rows}*.95),\text{ceil}(\text{cols}*.05):\text{ceil}(\text{cols}*.95))\ldots \]
\[ =\text{zeros}(\text{size}(\text{mask}(\text{ceil}(\text{rows}*.05):\text{ceil}(\text{rows}*.95),\ldots \]
\[ \text{ceil}(\text{cols}*.05):\text{ceil}(\text{cols}*.95))); \]
\[ \text{mask}=\sim\text{mask}; \]
\[ \text{Ucc}=\text{mask}.*\text{Ucc}; \]
\[ \text{Vcc}=\text{mask}.*\text{Vcc}; \]

\[ \text{if} \ (\text{all}(\text{display}==’\text{disp’})) \]
\[ \text{quiver}(\text{flipud(Ucc)},\text{flipud(Vcc)},3); \]
\[ \text{end} \]
\[ \text{end} \]
\[ \text{end} \]
LFA

Following is the code for the Local Feature Analysis, it is highly optimized to exploit the faster vectorial MATLAB functions. The input is a matrix data that contains all the δ-images used for training in row format, it outputs the set $M$.

% do SVD to get the eigenvectors and eigenvalues - better condition number
% than computing with eig since it's the square root of the eigenvalues

% 'data' needs to be ZERO-MEAN !!!
xbar=mean(data)';
mean_s=xbar(:,ones(size(data,1),1));
%clear xbar;
data=data-mean_s';
clear mean_s;

% 'data' is # images x # pixels (e.g. 155 images x 30*45 pixels)
A = data' ./ sqrt(155 - 1);

% compute SVD
% standard approach
[L M 0] = svd(A, 0);

% The columns of L are the eigenvectors of the covariance matrix
% of the original data.
% $M*M'$ are the corresponding eigenvalues

% $'M'$ gives the singular values - square roots of the eigenvalues

% put sqrt eigenvalues in a row vector, then invert them individually.
sqrteigvals = sum(M);
invsqrteigvals = 1 ./ sqrteigvals;
invsqrteigvals = diag(invsqrteigvals);

% Now V are the eigenvectors
V = L; clear L

% # of eigenvectors to use
N = 155;

Vshort = V(:, 1 : N);
invsqrteigvals = invsqrteigvals(1:N, 1:N);

% Make the Atick K matrix
K = Vshort * invsqrteigvals * Vshort';

% Transpose the data matrix again - make it pixels x images
data = data';

% Multiply Atick's K by original data to get new outputs -
% re-representation in terms of the Atick K basis
O = K * data;

% Now compute Atick's P - the covariance function of the outputs
P = O * O'/((155-1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Next step is to do the sparsification algorithm.

figure(1)
map=gray(256);
map(256,:)=[1 0 0];
colormap(map)

\% M is the maximum number of points we're going to consider
M = 155;

0_rec = zeros(size(0));

\% media is the mean image
media=xbar;
max_mean=max(media);

[junk index]=max(mean(P));
m=index(1);

for counter = 2 : M
  P_x_xl=P(:,m);
P_prime=inv(P(m,m));
A_M=P_x_xl*P_prime;
O_rec=A_M*O(m,:);
E=mean((O-O_rec)*(O-O_rec)');
[max_err index]=max(E);
skipped=0;
while any(index==m)
  skipped=skipped+1;
  E(index)=0;
  [junk index]=max(E);
end
figure(2)
plot(E), drawnow
m(counter)=index(1);
fprintf(1,'iter #:%.0d, point:%.0d, error:%.1f, ...
Gabor wavelet

The first function is used to produce a Gabor kernel $\psi$ of size $\text{nrows} \times \text{ncols}$ given the parameters $\mu = \omega_0$ and $\nu = \theta$. The FFT of these kernels are then stored to be quickly available when needed to build the final representation.

```matlab
function kernel=gabor(a,b,F_0,omega_0,theta,nrows,ncols)

kernel=zeros(nrows,ncols);

u_0 = F_0*cos(omega_0);
v_0 = -F_0*sin(omega_0);

center_v = nrows/2 + v_0*nrows - 1;
center_u = nrows/2 + u_0*nrows;

a_scaled = a*ncols;
b_scaled = b*nrows;

for i=1:nrows
    for j=1:ncols
        x = (j-center_u);
y = center_v-i;
        xt = x*cos(theta)+y*sin(theta);
yt = -x*sin(theta)+y*cos(theta);
```
xt = xt / a_scaled;
yt = yt / b_scaled;

dist = xt^2 + yt^2;
mask = exp(-dist*pi)/(a*b);
kernel(i,j) = mask;
end
end

The second function creates the complete jet representation for the input image with the desired orientations and frequency bands. It can also downsample the image if needed. It assumes the FFTs of the kernels are already available.

function J = jet(image,rows,cols,reduce)

% J = jet(image,rows,cols,reduce)
%
% rows and cols are optionals, they just speed everything up if supplied
% reduce is a switch to downsample the image 4 times before convolving
% the function works with upper and lower face images (standard sizes: 60x90
% and 66x96)

if nargin <= 2
[rows cols] = size(image);
end

error = 0;

if ~(rows == 60 | rows == 66)
disp(['Error!! '])
error = 1;
end

if error==0
    if (nargin==2 |nargin==4)
        rows=rows/2;
        cols=cols/2;
        image=imresize(image,[rows,cols],’near’);
    end
    
    dirname=['/usr/local/data/luca/kernels-',int2str(rows),’x’,int2str(cols),’/’];

    % get rid of garbage at the border
    % image(1,:)=image(2,:);
    % image(:,1)=image(:,2);
    % image(rows,:)=image(rows-1,:);
    % image(:,cols)=image(:,cols-1);

    img_star=fft2(mat2gray(image));

    J=zeros(40,rows*cols);

    count=0;
    for ni=4:-1:0
        for mu=0:7
            count=count+1;
            name=['ker_',int2str(ni),'_',int2str(mu)];
            eval(['load ',dirname,name]);
            temp=fftshift(ifft2(img_star.*psi_star));

            % jets are normalized for robustness against brightness variations
            temp=temp(:)/norm(temp(:));
            J(count,:)=temp;
        end
    end
end
Appendix B

USEFUL WWW RESOURCES

Facial Expressions and General Facial Recognition

General Resources

- Facial analysis resources:
  
  http://mambo.ucsc.edu/psl/fan1.html

- Facial animation resources:
  
  http://mambo.ucsc.edu/psl/fan.html


People

- Marian Stewart Bartlett’s [4, 3, 2] homepage:
  
  http://www.cn1.salk.edu/~marni/
- Michael Gray’s [34] homepage:

  http://www.cn1.salk.edu/~michael/

- Terrence J. Sejnowski’s [4, 3, 2, 7, 8, 21, 27, 33, 53, 55] homepage:

  http://www.salk.edu/faculty/sejnowski.html

- Javier Movellan’s [56] homepage:

  http://cogsci.ucsd.edu/~movellan

- Paul Belhumeur’s [6] homepage:

  http://www.eng.yale.edu/faculty/vita/belhumeur.html

- Tomaso Poggio’s [10, 11, 77] homepage:

  http://www.ai.mit.edu/projects/cbcl/web-pis/poggio/

- Alex “Sandy” Pentland’s [75] homepage:


- Paul Ekman’s [22, 23, 24, 25, 26, 27, 28, 29, 35] homepage:

  http://mambo.ucsc.edu/psl/joehager/pekm.html

- Joe Hager’s [4, 21, 27, 35] homepage:

  http://www.nirc.com/joeh.html
ICA and Blind Separation/Deconvolution

General Resources and Sample Code

- Tony Bell’s page on ICA:
  
  http://www.cn1.salk.edu/~tony/ica.html

- Te Won Lee’s page on ICA:
  
  http://www.cn1.salk.edu/~tewon/ica_cn1.html

Institutes

- Home page of the Computational Neurobiology Laboratory, part of The Salk Institute for Biological Studies, La Jolla, California:
  
  http://www.cn1.salk.edu/CNL/

- Home page for the Information Representation Group at RIKEN, headed by Prof. S. Amari, Saitama, Japan:
  
  http://www.bip.riken.go.jp/irl/Welcome.html

People

- Tony Bell’s [7, 8, 53, 55] homepage:
  
  http://www.cn1.salk.edu/~tony/

- Te Won Lee’s homepage:
  
  http://www.cn1.salk.edu/~tewon/
• Shun-ichi Amari’s homepage:

http://www.bip.riken.go.jp/irl/amari/amari.html

• Jean Paul Cardoso’s homepage:

http://sig.enst.fr:80/~cardoso/

• David MacKay’s homepage:

http://wol.ra.phy.cam.ac.uk/mackay/

LFA

• Homepage for the Computational Neuroscience Laboratory, Rockefeller University, New York:

http://venezia.rockefeller.edu/

• J.J. Atick’s homepage:

http://venezia.rockefeller.edu/atick/

• Homepage of Visionics Corporation, maker of the PC software FaceIt, winner of the FERET competition and the only known real world application of LFA:

http://www.faceit.com

\(^1\)Professor Amari was the first to introduce the idea of “information geometry” to model the space of probability functions and to use the “natural gradient” to speed up the convergence of ICA algorithms by taking into account the natural norm of the matrix space. We too used his technique in Eq. 5.13 where it avoided a matrix inversion of \(W\) for each step. See S.Amari (1997a) Information Geometry, Contemporary Mathematics and S. Amari (1997) Natural Gradient Works Efficiently in Learning, to appear.
Gabor Wavelets and Jets

• *An Introduction to Wavelets II The Neural Perspective:*

  http://www.fen.bris.ac.uk/engmaths/research/slide_show/slide_show.html

• Gabor Jets as I implemented them:

  http://www.cnl.salk.edu/~wiskott/Projects/Jet.html

• Laurenz Wiskott’s homepage:

  http://www.cnl.salk.edu/~wiskott/

More General Paper Repositories

• Papers from the Computational Neurobiology Laboratory:

  http://www.cnl.salk.edu/cgi-bin/pub-search

• Papers from MIT AI Laboratory:

  http://www.ai.mit.edu/pubs.html

• Cognitive Science papers:

  http://cogprints.soton.ac.uk/
Bibliografia


resentations for face recognition. In T. Rogowitz, B. & Pappas, editor, *Pro-
ceedings of the SPIE Symposium on Electronic Imaging: Science and Technol-
ogy; Human Vision and Electronic Imaging III*, volume 3299, San Jose, CA, 

[3] M.S. Bartlett and T.J. Sejnowski. Viewpoint invariant face recognition us-
ing independent component analysis and attractor networks. In M. Mozer, 
M. Jordan, and T. Petsche, editors, *Advances in Neural Information Process-

Classifying facial action. In D. Touretski, M. Mozer, and M. Hasselmo, editors, 

importance of upper and lower areas of the face. *Journal of Personality and 

[6] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. fisher-
faces: Recognition using class specific linear projection. *IEEE Transations on 

separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 
1995.


