

# Computer Vision

## Head Pose estimation

Philippe Tanghe      Li Quan

1 juli 2011

### Inhoudsopgave

<b>1. Inleiding</b>	<b>2</b>
<b>2. Specificaties</b>	<b>2</b>
<b>3. Overzicht aanpakken</b>	<b>3</b>
3.1. Appearance template . . . . .	3
3.2. Detector array . . . . .	3
3.3. Nonlinear regression . . . . .	3
3.4. Manifold embedding . . . . .	4
3.5. Flexible model . . . . .	4
3.6. Geometric model . . . . .	4
3.7. Tracking . . . . .	4
3.8. Hybride . . . . .	4
<b>4. Hoofdpose schatting</b>	<b>5</b>
4.1. Keuze aanpak . . . . .	5
4.2. Overzicht . . . . .	5
4.2.1. Voorstelling (feature data) . . . . .	6
4.2.2. Leren van een model . . . . .	6
4.2.3. Voorspellen van poses . . . . .	6
4.3. Evaluatie . . . . .	7
4.3.1. Neurale netwerken . . . . .	7
4.3.2. Support Vector Machines . . . . .	9
4.3.3. Vergelijking . . . . .	10
<b>5. Gezichtsherkenning</b>	<b>10</b>
5.1. Implementatie . . . . .	10
5.2. Evaluatie . . . . .	11
<b>6. Conclusie</b>	<b>11</b>
<b>A. Labeling van testdata</b>	<b>13</b>

# 1. Inleiding

Dit verslag bespreekt het project voor het vak Computer Vision. Dit project omvat het ontwerpen van een *head pose estimation* systeem: de pose van een gegeven 2D-afbeelding van een hoofd moet zo nauwkeurig mogelijk geschat worden. Optioneel volgt na het schatten van de pose ook een systeem voor gezichtsherkenning.

## Overzicht

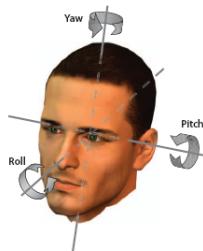
Eerst situeren we het probleem en geven de specificaties waaraan het systeem moet voldoen. Vervolgens geven we een korte samenvatting van de verschillende mogelijkheden om dit probleem aan te pakken.

Hierna bespreken we onze werkwijze, de daarbij gekozen trade-offs en de bijbehorende resultaten. Ten slotte geven we mogelijke verbeteringen en uitbreidingen mee.

# 2. Specificaties

De pose van het hoofd bevat heel wat nonverbale informatie [2, 9]. 2D gezichtsherkenningssystemen werken minder goed als geen rekening gehouden wordt met de pose [8, 13].

De bedoeling is dus om een systeem te ontwikkelen dat de hoofdpose op een efficiënte manier kan schatten. We beperken ons echter tot 1 vrijheidsgraad (i.e., ofwel yaw, ofwel pitch, zie figuur 1) en beschouwen het probleem als een classificatieprobleem.



**Figuur 1:** De vrijheidsgraden voor het head pose estimation systeem zijn in dit project beperkt tot ofwel yaw, ofwel pitch. [8]

De gegeven trainingset bestaat uit gelabelde, hoge-resolutie afbeeldingen (inclusief een aantal landmarks) van de hoofden van een twintigtal personen, elkeen in ieder van volgende poses<sup>1</sup>:

- $0^\circ$  voor yaw en pitch (neutrale poses).
- $10^\circ, 20^\circ, 30^\circ, \pm 45^\circ$  en  $\pm 90^\circ$  voor yaw.
- $\pm 5^\circ$  en  $\pm 10^\circ$  voor pitch.

---

<sup>1</sup>Hierbij komen positieve waarden overeen met een rechtse (yaw) respectievelijk bovenwaartse (pitch) kijkrichting.

(De asymmetrie tussen de linkse en rechtse kijkrichting van de trainingset kan eenvoudig verholpen worden door gebruik te maken van de respectieve gespiegelde versies. In totaal verkrijgen we dus 304 trainingsafbeeldingen.)

### 3. Overzicht aanpakken

Eerst geven we een overzicht van de verschillende aanpakken en hun voor- en nadelen, zoals aangegeven in de overview paper [8].

#### 3.1. Appearance template

Deze methode vergelijkt een nieuwe afbeelding van een hoofd met een verzameling van gelabelde afbeeldingen en zoekt de meest gelijkaardige pose. Dit is een zeer eenvoudig concept en is te gebruiken voor zowel hoge en lage resoluties.

Het grote nadeel van deze aanpak is dat ze allicht te naïef is. Ze is namelijk gebaseerd op de foutieve veronderstelling dat gelijkaardigheid van afbeeldingen gelijkaardigheid van poses impliceert. Een ander nadeel is dat wanneer de trainingset uitgebreid wordt dat er in principe met steeds meer afbeeldingen moet worden vergeleken, waardoor het systeem steeds rekenintensiever wordt.

#### 3.2. Detector array

Een detector array traant een reeks head detectors die elk afgesteld zijn op een bepaalde pose. De nieuwe afbeelding krijgt dan als schatting de pose die overeenstemt met de detector(s) die de beste support heeft (hebben). Deze aanpak werkt voor hoge en lage resoluties en kan de uitzicht-eigenschappen (niet pose gerelateerd) van het hoofd leren negeren.

Deze aanpak is discreet van aard en is dus enkel aangewezen voor een beperkt aantal mogelijkheden. (De rekencomplexiteit stijgt lineair met het aantal detectors). Het trainen van de verschillende detectors is vrij moeizaam. Als twee detectors ingesteld zijn voor twee heel gelijkaardige poses, moeten de positieve trainingsvoorbeelden voor de ene negatieve zijn voor de andere (en vice versa). Verder moet er voor elke detector een trainingset zijn die voldoende groot is.

#### 3.3. Nonlinear regression

Deze methode gebruikt niet-lineaire regressietools om een mapping te zoeken van de afbeelding (of de daaruit afgeleide feature data) tot een pose. De meest gebruikte zijn neurale netwerken en recentelijk ook support vector machines [3].

Het is niet altijd duidelijk of een goede mapping kan geleerd worden, maar uit de literatuur blijkt dat deze regressietools zeer goed presteren, zowel qua performantie en nauwkeurigheid. Deze systemen zijn wel vrij gevoelig aan de positie van het hoofd in de afbeelding.

### **3.4. Manifold embedding**

Bij manifold embedding zoekt men laagdimensionale manifolds die de variatie in pose modelleren. Het is dus — meer dan in de vorige methode — belangrijk een goede dimensionaliteitsreductie te vinden die zoveel mogelijk eigenschappen van de pose behoudt en zo weinig mogelijk karakteristieke eigenschappen. Dit is echter niet vanzelfsprekend: methodes hiervoor zijn bijvoorbeeld PCA, KPCA, LDA en KLDA.

Nieuwe afbeeldingen worden dan geprojecteerd in deze laagdimensionale manifolds waarop vervolgens gelijkaardige technieken als voorgaand worden gebruikt.

### **3.5. Flexible model**

Deze methode maakt gebruik van landmarks: dit zijn aanduidingen op de afbeelding die overeenstemmen met gezichtspecifieke posities, zoals de ooghoeken. Deze set landmarks vormen een niet-rigide model van het hoofd, dat een bepaalde graad van variatie toelaat.

De pose van een nieuwe afbeelding kan dan geschat worden door de verschillende modellen te vergelijken aan de hand van bijvoorbeeld hun parameterinstantiatie. Een bijkomend voordeel van deze methode is dat ze ook toelaat om virtuele gezichten te genereren (bijvoorbeeld een frontaal gezicht creëren aan de hand van een profelfoto).

### **3.6. Geometric model**

Geometrische modellen gebruiken de locatie van een beperkt aantal gezichtskenmerken (bijvoorbeeld ogen en neus). Uit hun relatieve positie kan dan vrij eenvoudig de pose geschat worden. Hierbij kan dus rechtstreeks gebruik gemaakt worden van bepaalde gekende eigenschappen van verschillende hoofdposes.

Voor deze methode moeten de gezichtskenmerken aangeduid zijn of kunnen gedetecteerd worden. Deze methode is dus problematisch wanneer bepaalde gezichtskenmerken moeilijk zichtbaar zijn.

### **3.7. Tracking**

Tracking gebruikt videoframes om de verandering in pose te schatten. Deze techniek is dus niet relevant voor dit project waarbij de pose van statische afbeeldingen moeten geschat worden.

### **3.8. Hybride**

Hybride methodes combineren meerdere van bovenstaande technieken om zoveel mogelijk de voordelen van verschillende systemen te bundelen en/of nadelen van één bepaald systeem te beperken. Het nadeel van deze methode is dat ze meer rekenkracht vergt en dat bij tegensprekende resultaten van verschillende systemen er een keuze gemaakt moet worden.

## 4. Hoofdpose schatting

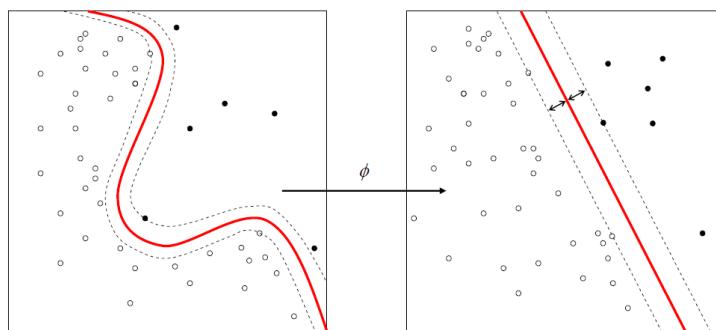
Voor het maken van de head pose estimator hebben we uiteindelijk gekozen voor het niet-lineaire regressiemodel.

### 4.1. Keuze aanpak

De belangrijkste motivatie voor onze keuze is dat de studies erop wezen dat neurale netwerken (NN) en support vector machines (SVM) heel goede resultaten behalen, zowel op vlak van performantie als nauwkeurigheid.

NNs worden reeds lang succesvol gebruikt in diverse pattern recognition taken. SVMs zijn recentelijk aan een grote opmars bezig — niet alleen in de Computer Vision wereld. Bijgevolg hebben deze tools een goede ondersteuning. De nadelen die opgesomd werden, zijn weinig relevant voor dit project (zo zijn de locaties van de hoofden in de afbeeldingen consistent).

In tegenstelling tot neurale netwerken hebben SVMs een betere theoretische basis [1] en is het basisconcept heel elegant, zoals figuur 2 illustreert.



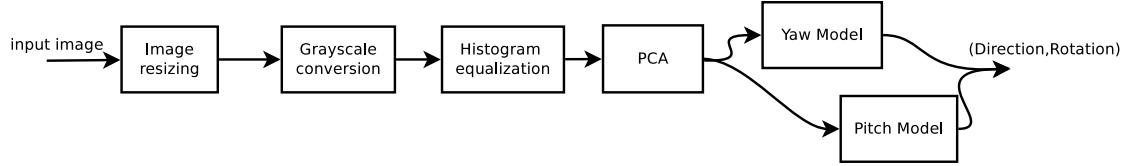
**Figuur 2:** Het principe van een Support Vector Machine. De transformatie naar de hoogdimensionale ruimte (via de kernel trick) laat toe om eenvoudig een hypervlak te zoeken dat de twee klassen lineair scheidt. [12]

### 4.2. Overzicht

We hebben zowel een systeem gebaseerd op neurale netwerken als een op SVMs ontwikkeld. De implementatie is gebaseerd op volgende papers: [4] en [7]. Voor elk van deze methodes zijn 3 belangrijke onderdelen te onderscheiden:

- Representatie van de feature data (feature selection/extraction).
- Het leren van het model uit de trainingset: dit omvat de keuze van allerhande parameters.
- Het maken van voorspellingen aan de hand van het model.

Een schematische voorstelling van het systeem is te vinden in figuur 3.



**Figuur 3:** Schematische voorstelling van het head pose estimation systeem.

#### 4.2.1. Voorstelling (feature data)

De trainingsafbeeldingen worden steeds geherscaleerd naar een vaste, kleinere resolutie. Verder worden de afbeeldingen ook omgezet naar grijswaarden.

Hierna wordt er histogramequalisatie toegepast op de afbeelding. Dit zorgt ervoor dat nadelige belichtingseffecten verminderd worden en zorgt voor een globaal beter contrast.

Vervolgens wordt Principal Component Analysis (PCA) [6] toegepast. Dit is een statistische methode voor dimensionaliteitsreductie waarbij van  $n$  oorspronkelijke (mogelijk gecorreleerde) variabelen,  $m$  orthogonale variabelen (de zogenaamde principal components, waarbij meestal  $m \ll n$ ) worden gezocht. De eerste principale component verklaart maximaal de covariantie-variantiestructuur, de tweede heeft het tweede grootste aandeel, enzovoort. Geometrisch komt dit overeen met een datatransformatie op een orthogonale basis.

Het selecteren van het aantal componenten is niet heel eenvoudig; deze komen ruwweg overeen met een trade-off tussen rekенcomplexiteit en nauwkeurigheid. Het aantal principale componenten werd empirisch vastgelegd op 20, net als in [7].

#### 4.2.2. Leren van een model

Na het verwerken van de afbeeldingen van de trainingset wordt deze gebruikt voor het leren van een model. Bij de neurale netwerk-aanpak wordt er gebruik gemaakt van de ingebouwde neural network (NN) toolbox van Matlab. Voor yaw en pitch werd gekozen voor twee afzonderlijke multilayer perceptron NNs, getraind via backpropagation.

Voor (LS-)SVM werd LS-SVMlab v1.7 gebruikt [10]. Ook hier werden de yaw en pitch afzonderlijk behandeld. Als kernel werd voor beide een (Gaussiaanse) RBF kernel gekozen,  $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\sigma^2}\right)$ , waarbij  $\sigma^2$  de squared bandwith is (deze bepaalt de smoothness van de beslissingsoppervlakken [3]). Verder is er nog de regularisatieparameter  $\gamma$  die de kost van misclassificaties beïnvloedt. Deze parameters kunnen bepaald worden via cross-validation met een grid search of coupled simulated annealing gevolgd door een simplex search.

#### 4.2.3. Voorspellen van poses

Het voorspellen van nieuwe afbeeldingen wordt eenvoudig gedaan door ze te transformeren in dezelfde ruimte als de trainingdata (zie feature data) en deze als invoer te geven aan de niet-lineaire regressiemodellen.

Aangezien de resultaten van de yaw beter zijn dan die van de pitch, hebben zij voorrang bij de uiteindelijke bepaling van de pose. Merk op dat we hier gebruik maken van het feit dat we slechts een vrijheidsgraad beschouwen.

### 4.3. Evaluatie

#### 4.3.1. Neurale netwerken

Voor neurale netwerken hebben we zowel met als zonder deze extra afbeeldingen getraind. Dit vormt geen probleem aangezien de testafbeeldingen ook binnen die beperkte trainingsrange vallen. Er worden 2 netwerken gebruikt: een voor yaw en een voor pitch.

Tabel 1 toont parameterinstellingen met de best bekomen resultaten voor enkele iteraties. De parameters zijn volgende:

- PCA: geeft aan of PCA is toegepast op de afbeelding; indien niet wordt de ruwe afbeelding als feature data gebruikt (na herscalering, omzetting in grijswaarde en hisogramequalisatie).
- gespiegelde images: geeft aan of de gespiegelde images mee gebruikt worden voor de training of niet.
- hidden layer (yaw/pitch): geeft de grootte<sup>2</sup> van de hidden layer in het netwerk voor yaw/pitch.
- train ratio: geeft het percentage van voorbeelden in de trainingset voor crossvalidatie (het complement is het percentage voorbeelden in de validation set).

De training gebeurt met een backpropagation conjugate gradient algoritme. De resultaten voor de neurale netwerken variëren sterk in kwaliteit van iteratie tot iteratie. Er zit met een vaste parameterinstelling nog willekeurigheid om tot een concreet netwerk te komen (zoals bijvoorbeeld welke voorbeelden nu net als validation set worden gekozen).

Het netwerk lijkt (met de geteste parameterinstellingen) over het algemeen het beste te werken zonder de extra afbeeldingen en geeft ook vaker goede resultaten indien we de afbeelding als feature data gebruiken. Maar zonder PCA hebben we een input-grootte van 1024 ( $32 \times 32$ ), wat computationally wel wat zwaarder is dan met een input-grootte gelijk aan 20 (het gekozen aantal principale componenten).

Wanneer we de resultaten beter bekijken merken we dat soms ook grote classificatiefouten voorkomen. Een voorbeeld voor het model dat overeenstemt met de eerste kolom van tabel 1 wordt gegeven in figuur 4.

Goede resultaten kunnen bekomen worden met deze aanpak. Het probleem is dat er meerdere iteraties nodig zijn door de aanwezige randomheid. Ook is het wellicht beter een grotere testset te gebruiken om sluitende conclusies omtrent accuracy te kunnen trekken.

---

<sup>2</sup>Een vuistregel voor de grootte van de hidden layer is dat ze ligt tussen input- en outputgrootte, [5].

	$\times$	$\times$	$\times$	$\times$	$\checkmark$	$\checkmark$
PCA	$\times$	$\times$	$\times$	$\times$	$\checkmark$	$\checkmark$
gespiegelde images	$\times$	$\times$	$\checkmark$	$\checkmark$	$\times$	$\checkmark$
hidden layer (yaw)	22	22	22	22	22	22
hidden layer (pitch)	25	500	22	22	33	22
train ratio	0.67	0.67	0.67	0.70	0.67	0.70
training accuracy (yaw)	0.967	0.963	0.878	0.957	0.959	0.868
training accuracy (pitch)	0.861	0.824	0.845	0.918	0.795	0.776
training accuracy (combined)	0.828	0.787	0.724	0.875	0.7541	0.645
test accuracy (yaw)	0.9	0.95	0.9	0.8	1	0.8
test accuracy (pitch)	0.85	0.85	0.9	0.8	0.85	0.85
test accuracy (combined)	0.75	0.8	0.8	0.6	0.85	0.65

**Tabel 1:** Parameters voor NN en training en test accuracies.



**Figuur 4:** Pose uit de testset die door een NN met een grote fout wordt geklassificeerd (schatting pitch -10° in plaats van 10°).

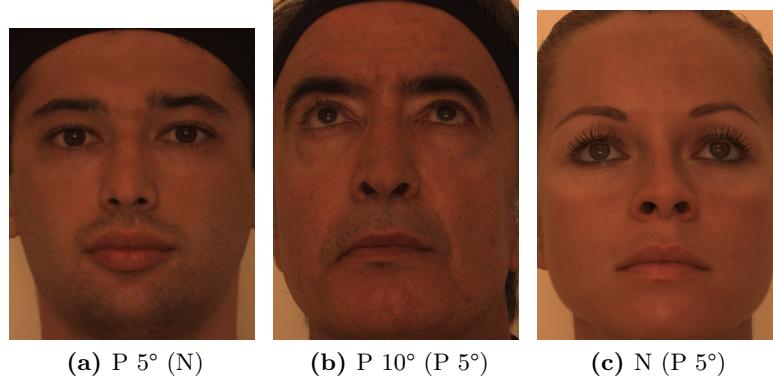
### 4.3.2. Support Vector Machines

Tabel 2 toont de parameters voor de kernel van de SVM verkregen via coupled simulated annealing gevolgd door simplex search, en de trainings- en testsetaccuracien. Het is duidelijk dat de nauwkeurigheden op zowel de training- als de testdata algemeen vrij goed zijn.

	$16 \times 16$	$32 \times 32$	$64 \times 64$
$\gamma_{yaw}$	0.416	2.028	3.821
$\sigma_{yaw}^2$	7.891	17.587	27.788
$\gamma_{pitch}$	165.805	1201.045	387.463
$\sigma_{pitch}^2$	37.059	169.826	73.507
training accuracy (yaw)	0.990	0.987	0.977
training accuracy (pitch)	0.931	0.944	0.954
training accuracy (combined)	0.921	0.931	0.931
test accuracy (yaw)	0.95	1	1
test accuracy (pitch)	0.80	0.85	0.85
test accuracy (combined)	0.75	0.85	0.85

**Tabel 2:** Parameters voor SVM en training en test accuracien.

Figuur 5 toont de 3 foutief geklassificeerde testimages (volgens onze eigen labeling, zie tabel 3). Pitch lijkt moeilijker in te schatten dan yaw, zoals ook blijkt uit tabel 2. Dit is inherent een moeilijker probleem, aangezien de verschillen vergeleken met yawrotaties eerder subtel zijn.



**Figuur 5:** Poses uit de testset die “verkeerd” worden voorspeld met SVM. Tussen haakjes staan de “correcte” labels.

Het is belangrijk om op te merken dat we geen rekening gehouden met de gradatie van fouten: volledig foute misclassificaties zouden bijvoorbeeld zwaarder kunnen afgestraft worden. Bij onze experimenten is het duidelijk dat als er foute classificaties werden gemaakt, deze slechts een klasse verschilden van de correcte.

#### 4.3.3. Vergelijking

Over het algemeen zijn de resultaten van de SVMs dan die van de NNs. De SVMs zijn eenvoudiger te trainen en we bekomen consistent betere nauwkeurigheden voor zowel yaw en pitch (training- en testaccuracy) dan de NNs.

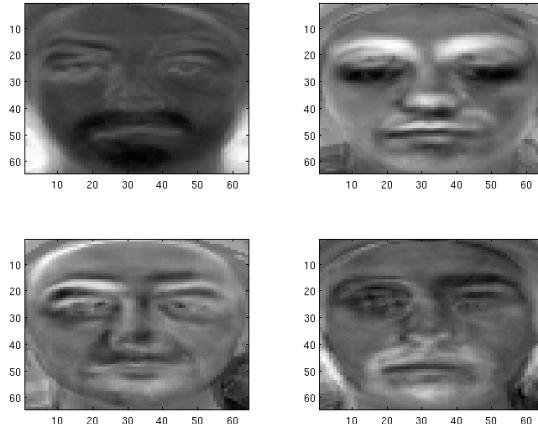
### 5. Gezichtsherkenning

Gezichtsherkenning houdend met posevariatie is geen eenvoudige taak: een overzicht van verschillende aanpakken en hun voor- en nadelen zijn beschreven in [13]. Deze paper deelt de methodes in ruwweg 3 categorieën: algemene methodes (holistische, lokale algoritmen), 2D (transformatie in pose ruimte) en 3D (gebaseerd op 3D modellen van het hoofd) aanpakken.

#### 5.1. Implementatie

Voor het optionele gezichtsherkenning gedeelte hebben we als proof of concept een heel eenvoudig systeem ontwikkeld, gebaseerd op het bekende eigenfaces algoritme [11].

De stappen zijn in principe heel gelijkaardig aan die voor de pose-estimatie. Hier gebruiken we enkel de neutrale gezichten. Deze worden op analoge manier gepreprocessed en na PCA verkrijgen we een set eigenfaces (zie figuur 6). We kunnen een nieuw gezicht dan voorstellen als een lineaire combinatie van deze eigenfaces.



**Figuur 6:** Enkele eigenfaces.

## 5.2. Evaluatie

Voor het voorspellen van gezichten wordt als dissimilariteitsscore gewoon de Euclidische afstand gebruikt (een betere metriek is de Mahalanobis [6]). Hiermee verkrijgen we een herkenningsrate van 45% op de gegeven testafbeeldingen. De relatief lage nauwkeurigheid ligt aan het feit dat we momenteel nog geen rekening houden met posevariatie.

Een manier om deze wel in rekening te brengen, is om na het voorspellen van de pose, een virtueel neutraal gezicht te maken (of omgekeerd, de neutrale images omzetten naar de gegeven pose) en deze te vergelijken op bovenstaande manier (een andere mogelijkheid zou kunnen zijn om opnieuw niet-lineaire regressietools te gebruiken). Dit is mogelijk met bijvoorbeeld 3D modellen of Active Appearance Models. Wegens tijdsbeperkingen werden deze extra functionaliteiten echter niet geïmplementeerd.

Een bijkomstige beperking was overigens dat volgens de specificaties elke te testen afbeelding precies overeenkwam met één afbeelding in de database. Dit is uiteraard een naïeve veronderstelling en maakt het volledige probleem (personen die niet in de database zitten, posevariatie, onderscheid tussen hoofd en niet hoofd, ...) heel wat complexer.

## 6. Conclusie

We hebben niet-lineaire regressie gebruikt voor het schatten van de hoofdposes. Als feature data gebruiken we voor een image de geherscaleerde afbeelding ( $32 \times 32$ ), omgezet in grijswaarden en een histogramequalisatie toegepast en ten slotte PCA projectie (20 principale componenten). We hebben een implementatie met neurale netwerken en een met support vector machines vergeleken.

*De support vector machine-aanpak behaalde de beste resultaten*, met een trainingset accuracy van 98% en 95% en testset accuracy van 100% en 85% voor respectievelijk yaw en pitch. De gemaakte fouten hierbij waren allen bij pitch en verschilden telkens 1 klasse ten opzichte van de juiste respectieve klasse.

Voor het gezichtsherkenningssysteem hebben we een simpele aanpak gekozen gebaseerd op het eigenfaces algoritme, zonder rekening te houden met de posevariatie. Hiermee krijgen we een gezichtsherkenningssaccuracy van 45%.

Het volledige systeem is uiteraard nog vrij beperkt in mogelijkheden aangezien heel wat vereenvoudigingen werden gemaakt. Zo is er het beperkt aantal rotatievrijheden, de veronderstelling dat alle afbeeldingen hoofden zijn, die reeds mooi werden bijgesneden en van hoge resolutie zijn, zonder hinder van occlusies, ...

Het lijkt ons interessant om het systeem hiervoor uit te breiden door meer trainingsimages te gebruiken, misschien om heel fijne schattingen te kunnen maken met behulp van bijvoorbeeld Support Vector Regression, evenals Active Appearance Models te gebruiken om gezichten te kunnen synthetiseren zodat het gezichtsherkenningssysteem hier expliciet gebruik kan van maken. Ook kunnen real-time systemen onderzocht worden.

Ook hybride vormen die eventueel gebruik maken van de landmarks kunnen mogelijk de moeilijkheden bij het schatten van de pitch verhelpen.

## Referenties

- [1] AHMAD, A. R., KHALID, M., YUSOF, R., AND VIARD-GAUDIN, C. The comparative study of svm tools for data classification, 2009.
- [2] ASTERIADIS, S., TZOUVELI, P., KARPOUZIS, K., AND KOLLIAS, S. Non-verbal feedback on user interest based on gaze direction and head pose. *International Workshop on Semantic Media Adaptation and Personalization* (2007), 171–178.
- [3] BLOCKEEL, H. Machine learning and inductive inference, 2010.
- [4] BROWN, L. M., AND TRAN, Y.-L. Comparative study of coarse head pose estimation. In *Proceedings of the Workshop on Motion and Video Computing* (Washington, DC, USA, 2002), MOTION '02, IEEE Computer Society, pp. 125–135.
- [5] FAQS.ORG. How many hidden units should I use?, 2011. <http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-10.html> [last accessed: June 29, 2011].
- [6] HUBERT, M. Statistische modellen & data-analyse, 2011.
- [7] LI, Y., GONG, S., SHERRAH, J., AND LIDDELL, H. M. Support vector machine based multi-view face detection and recognition. *Image and Vision Computing* 22 (2003), 413–427.
- [8] MURPHY-CHUTORIAN, E., AND TRIVEDI, M. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 4 (Apr. 2009), 607–626.
- [9] SHEERMAN-CHASE, T., ONG, E.-J., AND BOWDEN, R. Feature selection of facial displays for detection of non verbal communication in natural conversation. In *IEEE International Workshop on Human-Computer Interaction* (Kyoto, Oct 2009).
- [10] SUYKENS, J. *Least squares support vector machines*. World Scientific, 2002.
- [11] TURK, M. A., AND PENTLAND, A. P. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (1991), IEEE Comput. Soc. Press, pp. 586–591.
- [12] WIKIPEDIA. Support Vector Machines — Wikipedia, the free encyclopedia, 2011. [http://en.wikipedia.org/wiki/Support\\_vector\\_machine](http://en.wikipedia.org/wiki/Support_vector_machine) [last accessed: June 29, 2011].
- [13] ZHANG, X., AND GAO, Y. Face recognition across pose: A review. *Pattern Recognition* 42, 11 (2009), 2876–2896.

## A. Labeling van testdata

Tabel 3 toont onze manuele labeling van de testdata. We hebben dit gedaan om ground truth data te hebben om de verschillende experimenten te kunnen beoordelen en te vergelijken. Merk op dat hier waarschijnlijk wat onnauwkeurigheden tussen zitten.

naam	richting	rotatie
001	P	5°
002	Y	-90°
003	Y	30°
004	P	5°
005	Y	20°
006	Y	30°
007	Y	45°
008	N	
009	P	10°
010	Y	30°
011	Y	45°
012	Y	90°
013	Y	-45°
014	Y	20°
015	P	5°
016	P	10°
017	Y	-45°
018	Y	10°
019	Y	20°
020	Y	30°

**Tabel 3:** Labeling van testdata.