

Ansiktsigenkänning och klassificering



Sammanfattning

I denna rapport beskrivs en metod för att klassificera ansikten med hjälp av ansiktsigenkänning. De enskilda teknikerna som tillsammans utgör ansiktsigenkänningsalgoritmen är i grunden baserade på morfologiska operationer och eigenfaces. Det valda verktyget för att utveckla och skapa dessa metoder är det mycket kompetenta programmet Matlab.

För att klassificera ett ansikte krävs en pålitlig metod för att jämföra ett okänt ansikte med kända ansikten lagrade i en databas. I vår ansiktsigenkänningsalgoritm lokaliserar vi ögonen i den okända bilden med hjälp av verktyg vi skapat för att sedan klippa ut ansiktet ur bilden, skala ansiktsbilden och korrigera eventuella rotationer av ansiktet. Med hjälp av en igenkänningsmetod baserad på eigenfaces analyseras den okända ansiktsbilden och jämförs samt klassificeras mot innehållet i databasen.

INNEHÅLLSFÖRTECKNING

1	INLEDNING	1
1.1	SYFTE	1
1.2	BAKGRUND	1
1.3	METOD OCH KÄLLOR	1
2	ARBETSGÅNG	2
3	ALGORITMER OCH VERKTYG	4
3.1	EIGENFACEMETODEN	4
3.2	ANSIKTSKONTURER	5
3.3	ÖGONKARTA	6
4	RESULTAT	7
5	DISKUSSION	8
6	KÄLLOR	9
7	REFERENSER	9

1 Inledning

1.1 Syfte

I dagens samhälle används ofta kameror för att övervaka publika platser och efterarbetet med att manuellt analysera innehållet i bilderna är ett arbetsmoment som kräver enorma resurser. Att automatiskt analysera innehållet av materialet med hjälp av digitala analyser och algoritmer är i jämförelse mycket effektivt. Denna process är dock mycket avancerad då det finns många olika tillvägagångssätt och metoder för att skapa dessa verktyg och få av dessa hanterat alla möjliga variationer av innehållet i materialet.

Syftet med detta projekt är att skapa en ansiktsigenkänningsalgoritm för att klassificera ansikten. Denna algoritm ska till en viss gräns kunna klassificera ansiktsbilder som är roterade, skalade och med varierande ljusförhållanden.

1.2 Bakgrund

Projektet utgör examinationen i kursen TNM034 - Avancerad Bildbehandling och Bildanalys och förutsätter att vi har tidigare erfarenhet av bildbehandling och bildanalys samt grundläggande förståelse för programmeringsverktyget Matlab.

1.3 Metod och Källor

I den första projektfasen så studerade vi olika rapporter och relevant material som vi dels blev tilldelade av kursansvariga men även artiklar som vi fann på Internet. Vi insåg tidigt att det finns ett stort antal olika tillvägagångssätt för att skapa ansiktsigenkänningsmetoder och den andra projektfasen bestod till stor del av att utvärdera de olika källorna och deras respektive metoder för att kunna bestämma utgångspunkten för vårt projekt. En ansiktsigenkänningsalgoritm baserad på eigenfaces valdes slutligen som bas i vårt projekt.

Vidare fortsatte arbetet med regelbundna möten i projektgruppen där merparten av arbetet utfördes. Arbetet som gjorts enskilt av gruppmedlemmar presenterades och sammanställdes vid dessa tillfällen.

2 Arbetsgång

Enligt den programspecifikation som vi blivit tilldelade av vår examinator så ska det slutgiltiga programmet läsa in en bild och validera denna mot de godkända bilder som utgör databasen över kända ansikten. Om personen i fråga återfinns i databasen så ska programmet returnera en etta samt personens id annars en nolla.

Eftersom vi valt att bygga vår ansiktsgenkänningsalgoritm på en metod baserad på egenfaces så startade vi arbetet med att skapa det ramverk där vi läser in samtliga bilder till programmet, beräknar de egenskapsvektorer som beskriver innehållet i bilderna och slutligen sparar dessa vektorer till det som blir vår databas, mer om egenface metoden i ett senare skede.

För att validera en bild återanvänder vi koden för att beräkna en bilds egenskapsvektorer ifrån det ramverk där vi skapar databasen. Själva valideringen är sedan baserad på avståndsbedömning där vi beräknar det euklidiska avståndet ifrån de egenskapsvektorer som beskriver den bild vi önskar validera till samtliga egenskapsvektorer som utgör databasen. Det set av egenskapsvektorer som har kortast avstånd till testbilden representerar den bild i databasen som är mest lik den bild vi vill validera. För ett visst, tillräckligt lågt, tröskelvärde för detta avstånd så anser vi att vi har en matchning och ansiktet är validerat mot databasen.

Eftersom kvalitén och säkerheten av utgången av klassificeringen av en okänd bild mot databasen är starkt kopplad till kvalitén av de bilder som utgör databasen samt den bild vi önskar validera så skapade vi sedan verktyg för att korrigera, skala samt rotera bilder. Detta verktyg använder vi sedan på de bilder vi önskar bygga vår databas utifrån samt på den bild vi vill klassificera mot databasen.

I ett första steg i detta verktyg lokaliserar vi personens ansiktskontur och ögon i bilden för att kunna klippa ut ansiktet ur bilden och eventuellt korrigera för rotation av ögonen om dessa inte redan ligger på en horisontell linje. Denna ansiktsbild skalas sedan till en kvadrat av storleken 150x150 pixlar för att uppfylla de krav som egenfaces metoden kräver.

För att hitta en ansiktskontur i en ansiktsbild trösklas C_r - och C_b -kanalerna utifrån givna värden då vi känner till ansiktets färguppbyggnad. Morfologiska operationer så som öppning och stängning samt funktioner som etikettering används för att skapa en solid ansiktskarta.

När vi ska lokalisera ögonen i ansiktsbilden utnyttjar vi vår kunskap om att ett öga innehåller fler blå toner i jämförelse med röda toner och skapar en ögonkarta genom att beräkna kvoten av C_b -kanalen och C_r -kanalen. Denna ögonkarta trösklas sedan med ett specifikt värde och kombineras med ansiktskonturen för att bilda ögonkandidater. Den bild som innehåller ögonkandidaterna kombineras sedan i sin tur med originalbilden för att avlägsna allt som inte är ögonkandidater. Med hjälp av en morfologisk operation som krymper öppna områden i bilden till endast en pixel så får vi slutligen två punkter i ansiktsbilden som representerar ögonen.

I slutfasen av projektet bestod största delen av arbetet av att utvärdera och optimera de verktyg som vi skapat för att ansiktsgenkänningsalgoritmen och klassificeringen ska fungera så bra som möjligt. Beslutsgränser och tröskelvärden testades noggrant i ett försök att hitta optimala värden. Slutligen utformades ett testprogram för att enkelt kunna validera och klassificera en valfri bild mot databasen.

3 Algoritmer och verktyg

3.1 Eigenfacemetoden

I metoden använder vi samtliga bilder som tillhör db1, den bilddatabas som vi blev tilldelade av vår handledare. Vi har dessutom lagt till ytterligare bilder i databasen. Dessa bilder är bilderna från db1 fast roterade och skalade. Detta resulterar i att vi får 5 versioner av varje bild vilket ger oss mer övningsdata och en mindre rotations- och skalnings- känslig algoritm. Vi låter ansikt bilderna vara tvådimensionella $N * N$ vektorer. Dessa vektorer linjäriserar vi till en vektor med storleken $2 * N^2$ där bildens färgkanaler C_b och C_r ligger efter varandra.

Om databasen innehåller M st. bilder låter vi dessa vara $I_1, I_2, I_3, \dots, I_M$ och med dessa beräknar vi medelansiktet, ψ :

$$\psi = \frac{1}{M} \sum_{n=1}^M I_n$$

Varje ansikte skiljer sig med vektorn $\Phi_M = I_M - \psi$, från medelansiktet. Med hjälp av dessa vektorer skapar vi en matris, A , som innehåller skillnaderna.

$$A = [\Phi_1, \Phi_2, \dots, \Phi_M].$$

Utifrån A skapas sedan kovariansmatrisen, $C = AA^T$. Vi önskar sedan beräkna egenvärdena och egenvektorerna för C men matrisen har dimensionen $N^2 * N^2$ vilket ställer till problem. Att göra dessa räkneoperationer på C skulle inte gå, men det finns andra sätt. Vi kan med hjälp av PCA (Principal Component Analysis) minska dimensionen från $N^2 * N^2$ till $M * M$.

Vi använder PCA analys och skapar $M * M$ matrisen $L = A^T A$ [1] och beräknar dess egenvektorer v_l . Med hjälp av dessa skapas s.k. eigenfaces, u_l :

$$u_l = \sum_{k=1}^M v_{lk} \Phi_k.$$

När vi får en bild som skall klassificeras mot databasen transformerar vi bilden till eigenface komponenter genom följande operation. $\omega_k = u_k^T (I - \psi)$ för $k = 1, 2, \dots, M$. Dessa komponenter läggs i en vektor $\Omega^T = [\omega_1, \omega_2, \dots, \omega_M]$. Denna vektor beskriver hur mycket varje eigenface bidrar till att representera testbilden.

Tidigare har vi skapat dessa eigenfaces komponenter för samtliga bilder i databasen och för att avgöra vilket ansikte i databasen som motsvarar testbilden beräknas det euklidiska avståndet mellan eigenface komponenterna. Ett godkänt svar avgörs sedan med hjälp av en tröskel.

3.2 Ansiktskonturer

För att lokalisera ansiktskonturen av ett ansikte i en bild så utnyttjar vi de kunskaper vi har rörande det generella ansiktets färgsammansättning. Alltså, efter att vi läst in bilden och ljuskompenserat den, konverteras bilden till YC_bC_r där krominanskanalerna trösklas enligt:

$$C_{b1} = C_b < 0.49 \quad C_{b2} = C_b < 0.33$$

$$C_{r1} = C_r < 0.66 \quad C_{r2} = C_r < 0.55$$

Dessa tröskelvärden motsvarar de intervall som vi konstaterat att ansiktets pixlar ligger inom. Vi multiplicerar därefter C_{b1} med C_{b2} samt C_{r1} med C_{r2} och erhåller nya C_r och C_b som vi genom multiplikation med varandra skapar en bildkarta. Vi applicerar sedan en kombination av morfologiska operationer med lämpligt strukturelement och etikettering för att slutligen modifiera vår bildkarta så enbart ansiktskonturen återstår som ett solitt objekt.

3.3 Ögonkarta

När vi lokaliserar ögonen i ansiktsbilden utnyttjar vi vår kunskap om att ett öga innehåller fler blå toner jämfört med röda toner och skapar en ögonkarta E genom att beräkna kvoten av C_b kanalen och C_r kanalen.

Nästa steg är att multiplicera vår ögonkarta med en modifiering av vår ansiktsskarta F_{mod} , detta för att utesluta onödiga delar av ansiktsbilden.

$$E = E * F_{\text{mod}}$$

Modifieringen av ansiktsskartan består av en krympning samt en uteslutning av pixlar som ligger för långt ner i bilden.

Därefter trösklas ögonkartan med ett varierande värde som minskar till vi hittar minst två st. ögonkandidater. Vid detta stadié vidgas sedan ögonkartan med hjälp av morfologiska operationer i form av en öppning där ett lämpligt strukturelement används för att få bort små håligheter i ögonkartan.

$$E = E \oplus SE \text{ med ett diskstrukturelement av storleken } 10.$$

De slutgiltiga ögonpositionerna tas sedan fram med hjälp av en krympning som pågår till endast en pixel återstår.

Med hjälp av ögonen kan vi positionera alla ansikten i databasen på samma ställe. Vinkeln mellan ögonen beräknas för att kunna lägga ögonen på en horisontell linje. Därefter skalas ansiktena till ett fixerat ögonavstånd på 50 pixlar och beskärs till en kvadrat på 150x150 pixlar.

4 Resultat

För att se hur bra programmet fungerar har vi gjort några tester. Dessa tester genomförs genom att vi jämför bilderna i databasen med samma bilder modifierade enligt tabellerna nedan.

Rotation	2 grader	5 grader	7 grader	10 grader
Korrekta identifieringar.	100 %	100 %	85 %	90 %

Skalning – förminskning.	5 %	10 %	20 %	50 %
Korrekta identifieringar.	90 %	100 %	85 %	0 %
Skalning – förstoring.	5 %	10 %	20 %	50 %
Korrekta identifieringar.	100 %	100 %	90 %	90 %

Translation – x-led.	10 pixlar	20 pixlar	40 pixlar	60 pixlar
Korrekta identifieringar.	95 %	95 %	85 %	70 %
Translation – y-led.	10 pixlar	20 pixlar	40 pixlar	60 pixlar
Korrekta identifieringar.	100 %	80 %	60 %	35 %

Testet visar bra resultat och vi är mycket nöjda med utgången. På vissa ställen visar testet upp lite motstridiga resultat. Till exempel ser vi att programmet klarar 10procentig skalning bättre än 5procentigt. Detta beror på att en 10procentig skalning av originalbilderna redan ligger i databasen.

5 Diskussion

Eigenfacemetoden är relativt enkel att implementera men dock väldigt känslig för förändringar i testbilden. Exempelvis om man gör en translation av bilden fungerar eigenfacemetoden genast mycket tveksamt. Av denna anledning är det mycket viktigt att vi bygger databasen av kvalitativa ansiktsbilder samt att de bilder vi önskar klassificera till så stor grad som möjlig liknar de i databasen.

De verktyg vi skapat för att kompensera bilder för rotation och skalning samt för att slutligen beskära ansiktet ur bilden ger oss bra förutsättningar för att klassificeringen ska fungera. Algoritmen skalar ansiktet så att det slutliga avståndet mellan ögonen blir 50 pixlar och roterar ansiktet så att ögonen ligger på en horisontell linje.

Klassificeringsproblem kan uppstå när testbilder beskärs, skalas eller transformeras men de verktyg vi skapat för att korrigera dessa bilder hanterar just skalning och translationer bra. Om en testbild beskärs av någon anledning så att delar av ansiktet försvinner så blir klassificeringen mycket svårare, om ens möjlig. Eigenface metoden hanterar translationer mycket dåligt och därför är kvalitén på verktyget vi skapat för att korrigera bilder mycket viktig.

Ljusförändringar som uppstår av reflektioner eller olika ljusförhållanden vid skapandet av testbilden är svåra att kompensera och klassificering kan misslyckas. Vi har implementerat en ljuskompenstringsalgoritm som sätter den mörkaste pixeln i bilden till svartpunkt och den ljusaste pixeln i bilden till vitpunkt och skalar därefter intensitetsvärdena i respektive färgkanal. Denna metod möjliggör att vi kan klassificera bilder med varierande ljusförhållanden.

De verktyg och algoritmer som vi skapat klarar inom rimliga gränser de krav som projektuppgiften kräver trots att vi gjort vissa förenklingar och vi anser oss nöjda med resultatet.

6 Källor

Rein-Lien Hsu, Mohamed Abdel-Mottaleb and Anil K. Jain, "Face Detection in Color Images," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, May 2002.

Matthew A. Turk and Alex P. Pentland, "Face Recognition Using Eigenfaces," *Vision and Modeling Group, The Media Laboratory Massachusetts Institute of Technology*, 1991.

7 Referenser

[1] M. Turk, A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.