

Morphing

EDBV WS 2014/2015: AG_D5

Gecal Mahmut Salih (1125153)

Schefcik Jürgen (1025543)

Tucek Tom (1325775)

Ulreich Maximilian (1027769)

Valdellon Patrick (0925869)

06. Jänner 2015

1. Gewählte Problemstellung

1.1. Ziel

Das Ziel unseres Projektes ist es, mit Hilfe der Software MatLab zwei Portraitfotos einzulesen und als Ausgabe ein gemorphed Bild der beiden zu liefern. Mittels Userinput werden auf diesen Bildern Punkte gesetzt, welche im Ergebnisbild enthalten sind und dessen Inhalt beeinflussen sollen.

1.2. Eingabe

Als Input dienen zwei Bilder vom Typ .jpg/.png. Nachdem die Bilder eingelesen wurden, ist es dem Benutzer über ein einfaches GUI möglich durch das Festlegen von einzelnen Kontrollpunkten auf beiden Bildern das Aussehen des Ausgabebildes zu beeinflussen.

1.3. Ausgabe

Nachdem man die Feature Points der beiden Gesichter angegeben hat, öffnet sich ein neuer Dialog, welcher das Endresultat anzeigt. In diesem Fenster besteht die Möglichkeit, direkt mit einem Schieberegler die prozentuelle Gewichtung der beiden Bilder auf das Endresultat anzuwenden und anzuzeigen. Außerdem kann man das erzeugte Ergebnis als .PNG Datei abspeichern. Die GUI bietet dazu noch die Möglichkeit, eine Animation anzuzeigen, welche bei 100% des ersten und 0% des zweiten Bildes startet, und in 5% Schritten zu 0% des ersten und 100% des zweiten Bildes läuft (sowie danach wieder zurück). Der Morphing-Algorithmus wird also in 5%-Schritten angewandt, und die Bilder werden im Abstand von 0.05 Sekunden angezeigt.

Gespeicherte Bilder können im Ordner output gefunden werden.

1.4. Voraussetzungen und Bedingungen

Die Inputbilder müssen über die gleichen Dimensionen verfügen und gut belichtet sein, da dies den Abgleich erleichtert. Des Weiteren wird angenommen, dass außer der Person keine weiteren Objekte im Vordergrund des Bildes sichtbar sind und das Gesicht der Person zur Gänze sichtbar ist, also die Person frontal zur Kamera abgebildet wurde und keine Objekte – wie zum Beispiel Haare - Teile des Gesichts verbergen.

1.5. Methodik

- 1. Einlesen von 2 Bildern**
- 2. Setzen der Feature Points**
- 3. Warping der Bilder durch interpolierte vertikale und horizontale Skalierung zwischen den Punkten**
- 4. Farbwertinterpolation**

1.6. Evaluierungsfragen

- Wie verhält sich die Schärfe des Ausgabebildes zu der der Eingabebilder?
- Wie beeinflusst die Anzahl der Kontrollpunkte das Ergebnis?
- Wie gut funktioniert die Featureerkennung?
- Entstehen Geister-Bilder?
- Ist unser Programm in der Lage ein fröhliches und ein trauriges Smiley bei einem 50:50-Morph zu einem neutralen Smiley zu vereinen?

2. Arbeitsteilung

(0,5 Seiten)

Wer hat welche Aufgaben übernommen (MATLAB-Funktionen, Abschnitt im Bericht, Evaluierung, Datenerfassung, etc.)?

Name	Tätigkeiten
Gecal Mahmut Salih	Konzepterstellung myginput.m Matlab GUI Präsentationsfolien Evaluation
Schefcik Jürgen	Konzepterstellung interpolativeScale.m (Geometrische Transformation) ownImresize.m Präsentation Eye-Recognition Methodik und Schlusswort im Bericht
Tucek Tom	Konzepterstellung Matlab GUI Farbwert-Interpolation Animation Implementierung im Bericht
Ulreich Maximilian	Konzepterstellung Datensatz Diverse Arbeiten am Bericht
Valdellon Patrick	Konzepterstellung Diverse Arbeiten am Bericht Eye-Recognition

3. Methodik

Auswahl der Feature-Points durch den Anwender

Die wesentlichen Feature-Points in Gesichtern sind automatisch nur schwer zuverlässig detektierbar. Deshalb haben wir in unserer Lösung auf eine manuelle Auswahl durch den Anwender zurückgegriffen. Dadurch können zuverlässig die wichtigsten Feature-Points in beiden Bildern ausgewählt werden.

Behandlung der Feature-Points

Bevor die ausgewählten Punkte für die interpolative Skalierung verwendet werden können müssen sie noch in eine verwendbare Form gebracht werden.

Diese Form sieht wie folgt aus:

- Der erste Punkt ist die erste Koordinate auf der Achse (z.B. $x = 1$)
- Der letzte Punkt ist die letzte Koordinate auf der Achse (z.B. $x = 512$)
- Die Punkte des ersten Bildes sind aufsteigend sortiert (die Punkte des zweiten Bildes werden entsprechend der Reihenfolge im ersten Bild sortiert, damit der Feature-Point-Zusammenhang erhalten bleibt)
- Die Punkte des zweiten Bildes müssen nach der Sortierung monoton wachsen

Um auch das monotone Wachstum im zweiten Bild zu garantieren kann es vorkommen, dass zwei Feature-Points entlang einer Achse zu einem neuen Feature-Point zusammengeführt werden müssen. Das passiert, wenn zwei Feature-Points in den beiden Bildern in umgekehrter Reihenfolge auftreten.

Um den neuen Feature-Point zu bestimmen wird ein neuer Durchschnittswert in beiden Bildern bestimmt, so dass die Werte in der Liste wieder monoton wachsend sind. Da Feature-Points bei Gesichtern meistens proportional sehr ähnlich sind (Augen und Mund sind zum Beispiel nie vertauscht) funktioniert die Behandlung der Punkte auf diese Art sehr gut für Gesichter.

Interpolative horizontale und vertikale Skalierung (geometrische Transformation)

Um die Punkte der beiden Bilder zu überlappen verwenden wir eine interpolative Skalierung (horizontal und vertikal). Diese Form des Warping wurde uns von Ines Janusch empfohlen (als schnelle Lösung des Warping).

Die Skalierung basiert auf dem Interpolationsfaktor. Dieser gibt an, wieviel Prozent des ersten Bildes erhalten bleiben. Ist dieser Faktor beispielsweise $k=0.9$, so werden die Feature-Points

des ersten Bildes um 10% in Richtung der Feature-Points im zweiten Bild - und die Feature-Points des zweiten Bildes um 90% in Richtung der Feature-Points im ersten Bild verschoben.

Um die Feature-Points zu verschieben wird der Bildausschnitt zwischen dem vorherigen und dem nachfolgenden Feature-Point entweder vergrößert bzw. verkleinert.

Durch diese Skalierung entlang der horizontalen und der vertikalen Achse liegen schlussendlich die Feature-Points übereinander.

Die interpolative Skalierung ist der erste Schritt von "Local Warp, then Cross-Dissolve"[1]. Cross-Dissolve bezieht sich hier auf die Farbinterpolation.

Farbinterpolation

Nachdem die Bilder skaliert wurden, werden sie mit Hilfe von Farbinterpolation in ein einzelnes Bild zusammengefügt. Dabei wird wieder der Interpolationsfaktor berücksichtigt, welcher angibt, wieviel Prozent der Farbwerte des ersten Bildes zu den restlichen Prozent der Farbwerte des zweiten Bildes addiert werden, um ein neues Bild zu generieren.

4. Implementierung

Wie habt ihr die im vorhergehenden Abschnitt vorgestellte Methodik praktisch umgesetzt? Wie werden die einzelnen Methoden kombiniert (zB. Implementierungspipeline)?

Die Implementierung und Kombination der einzelnen Methoden findet größtenteils entsprechend der Methodik statt. Anfangs wählt der Benutzer zwei Bilder aus. Anschließend kann er Feature-Punkte in den Bildern auswählen.

Der Standardparameter für die Anzahl an gewählten Punkten wurde nach dem Testen auf den Wert 13 festgelegt, jedoch sollten nicht mehr als 21 gesetzt werden (siehe 5.2). Dieser Parameter ist vom Benutzer jedoch auch durch Bearbeiten von main.m veränderbar.

Nachdem die Feature-Points gewählt wurden, öffnet sich das GUI, welches das erste Bild, sowie einen Slider, ein Textfeld und drei Buttons anzeigt. Durch Adjustieren des Sliders, oder Editieren des Textfeldes, kann der Interpolationsfaktor bestimmt werden. Bei einem Faktor von 0.5 werden beispielsweise beide Bilder gleich gewichtet verwendet, um das neue Bild zu erstellen.

Durch einen Klick auf den "Ok"-Button wird die Skalierungsfunktion mit dem momentan ausgewählten Interpolationsfaktor aufgerufen. Die Resultate - also die skalierten Bilder-,

werden daraufhin durch die Farbwertinterpolationsfunktion zu einem Bild kombiniert und auf dem Bildschirm angezeigt.

Der Animate-Button verwendet im Grunde die selben Methoden, allerdings ruft er diese 41mal in Folge, mit unterschiedlichen Interpolationsfaktoren im Intervall 0 -> 1 -> 0 (Schrittgröße 0.5) auf.

Der Save-Button speichert das aktuell im GUI sichtbare Bild im Ordner output als .PNG-Datei.

Wie startet der User das Programm? Welche Parameter hat der User zu setzen?

Der Benutzer startet das Programm über einen Aufruf von main.m. Er wird im Folgenden dazu aufgefordert zwei Bilder aus dem Datensatz zu wählen, welche dann für den Morphvorgang verwendet werden.

Von diesen beiden Bildern werden automatisch einige Merkmale - die Position der Augen und des Mundes- extrahiert, weil damit auch ohne Nutzereingabe ein rudimentärer Morph möglich ist.

Darufhin können vom User auf den Bildern einige Featurepoints gesetzt werden, deren Anzahl für beide Bilder identisch ist, und in main.m geändert werden kann.

Damit ist die direkte Interaktion des Benutzers mit den Bildern zu Ende. Die restliche Eingaben, also die Auswahl des Morphverhältnisses, das Speichern der Ergebnisbilder und das Starten des Morphs, werden über das GUI gesteuert.

5. Evaluierung

Der im Projekt enthaltene Datensatz umfasst 11 Bilder. Alle Bilder haben eine Auflösung von 646 x 848 Pixel. Die Fotos stammen von einem professionellem Fotografen, der sie für das Morphprojekt zur Verfügung gestellt hat. Im Original hatten die Bilder 2000x2662 Pixel und eine Auflösung von 300dpi. Solch eine hohe Qualität und Auflösung ist in unserem Fall nicht nötig und verlangsamt den Morphprozess um einen großen Zeitfaktor. Die originalen Fotos wurden mit einer Phase One IQ 180 Mittelwertkamera mit folgenden Kameraeinstellungen aufgenommen:

- Blendenzahl: F/22
- Belichtungszeit: 1/180 Sek.
- ISO-Wert: ISO-50
- Brennweite: 120 mm

Die Fotos konnten grundsätzlich folgende Eigenschaften vorweisen:

- Alle Fotos wurden vom Haaransatz bis zur Schulter aufgenommen (Portrait)
- Sie zeigen Personen im Alter zwischen 20 und 40 Jahren
- Die Hautfarbe ist mitteleuropäisch
- Keine der Personen kann markante Gesichtsmerkmale vorweisen
- Die Augenhöhe befindet sich immer knapp über der Hälfte der Mitte des Bildes
- Die Länge der Gesichter ist ungefähr gleich
- Die Gesichter sind stark belichtet damit Konturen gut zu erkennen sind
- Alle Bilder haben einen Schatten nach rechts

Evaluierungsfragen: (Alle Beispielbilder sind im Verhältnis 50:50 gemorphht)

1. Wie verhält sich die Schärfe des Ausgabebildes zu der der Eingabebilder?

Das Endresultat hängt sehr stark von den gesetzten Feature Points ab, da die Punkte von den beiden Bilder aufeinander skaliert werden. Danach erfolgt die Farbwertinterpolation, die im Vergleich zu den Eingabebildern natürlich nicht so scharf ist, aber dennoch subjektiv scharf wirkt. Je mehr und je genauer die Punkte gesetzt sind, desto schärfer wird das Ergebnis.

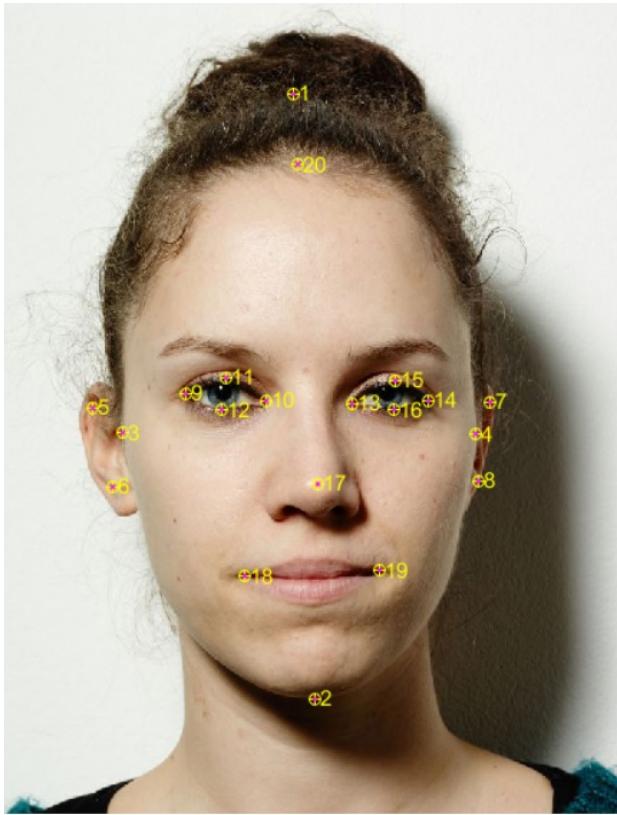
2. Wie beeinflusst die Anzahl der Kontrollpunkte das Ergebnis?

Wie oben erwähnt spielt die Anzahl der Kontrollpunkte eine sehr große Rolle. Dennoch muss man darauf achten, nicht zu viele Punkte zu setzen, da eine sehr große Anzahl an gesetzten Punkten zu Überschneidungen und Unschärfe zwischen den Punkten führt. Tests haben ergeben, dass man bei den markanten Punkten im Gesicht zumindest einen höchstens aber drei Kontrollpunkte setzen sollte. Anhand dieses Beispieles wird das gut ersichtlich:



Bei diesem Beispiel sind 5 Kontrollpunkte ausgewählt worden. Man sieht an der Nase, dass die Bilder hier sehr gut gemorpt wurden. Betrachtet man hingegen die Ohren, erkennt man, wie sehr verschoben sie voneinander sind, weil an den Ohren keine Kontrollpunkte gesetzt wurden. In diesem Beispiel wurden zu wenig Kontrollpunkte gesetzt. Im nächsten Beispiel betrachten wir die gleichen Personen mit 20 Kontrollpunkten.





In diesem Beispiel wurden 20 Kontrollpunkte pro Bild ausgewählt und man erkennt sofort einige Unterschiede im Vergleich zu dem vorigem Beispiel:

- Die Ohren sind fast überein
- Die Augen sind auch fast überein
- Das Kinn ist überein

Dennoch gibt es auch negative Aspekte:

- Die Nase ist nicht mehr übereinander
- Kiefer passt nicht mehr

Man sieht, dass mehr Kontrollpunkte das Bild nicht immer besser machen, sondern im schlimmsten Fall das Resultat verschlechtern. Die Ursache dafür ist die horizontale und vertikale Verschiebung der Kontrollpunkte im Algorithmus, da durch zu viele Kontrollpunkte zu viel verschoben werden kann.

Folgende Feature Points sind beim Morphen wichtig: (KP = Kontrollpunkt)

- Kopf:
 - Kinn (1 KP)
 - oberster Punkt des Kopfes (1 KP)
- Gesicht:
 - Augen (2 - 6 KP)
 - Nase (1 - 3 KP)
 - Mund (1 - 3 KP)
 - Haaransatz über der Stirn (0 oder 1 KP)
 - Schläfe (0 oder 2 KP)
 - Ohren (0 - 4 KP)

3. Wie gut funktioniert die Featureerkennung?

Als Featureerkennung wurde die Augendetektion gewählt. Die Implementierung bereitete Probleme, da sie nicht in jedem der Input-Bilder Augen finden konnte.

Wegen mangelnder Zeit konnte die Augendetektion leider nicht vollständig implementiert werden.

(Ist im Matlab-Ordner dennoch enthalten)

4. Entstehen Geister-Bilder?

Ja, da die Kontrollpunkte zueinander (ein Punkt im ersten Bild auf den selben Punkt im zweiten Bild) verschoben und skaliert werden. Somit sind Geisterbilder nicht zu vermeiden, weil jedes Gesicht einzigartig ist und nie genau in die Gesichtsform einer anderen Person passen wird.



Anhand des oben angeführten Beispiels erkennt man zum Beispiel an den Ohren, dass Geisterbilder entstehen wenn nicht genügend Kontrollpunkte gesetzt sind. Hier dasselbe Beispiel mit 4 zusätzlichen Kontrollpunkten an den Ohren:



An diesem Beispiel ist klar zu erkennen, dass das Geisterbild an den Ohren viel schwächer geworden ist. Würde man jetzt noch 2 weitere Kontrollpunkte pro Ohr (linkes und rechtes Ende) hinzufügen, wäre das Ergebnis noch viel besser oder das Geisterbild würde komplett verschwinden.

5. Ist unser Programm in der Lage ein fröhliches und ein trauriges Smiley bei einem 50:50-Morph zu einem neutralen Smiley zu vereinen?



Nein.

6. Schlusswort

Dieses Projekt hat sich als schwerer herausgestellt als anfangs angenommen. Vor allem das Warping der Bilder hat uns vor Schwierigkeiten gestellt.

Unsere Warping-Lösung funktioniert meist sehr gut, liefert allerdings bei größeren Unterschieden in den Proportionen (durch die Durchschnittsbildung des neuen Feature-Points) manchmal unerwünschte Ergebnisse.

Auch das Morphing zweier Smileys führt auf Grund dieser Lösung nicht zum gewünschten Effekt.

Um ein konsistenteres Ergebnis zu erhalten wäre das automatische Detektieren aller Feature-Points in den beiden Bildern eine gute Möglichkeit.

Um bei dem Morphing flexibler zu sein könnte ein Meshwarp an Stelle der interpolativen Skalierung implementiert werden. Dadurch wäre man auch von den Proportionen her flexibler und nicht nur auf Gesichter beschränkt (wenn man den ersten Verbesserungsvorschlag ignoriert).

Literatur

[1] University of Wisconsin-Madison (Author unbekannt). Image Morphing
<http://pages.cs.wisc.edu/~dyer/cs534/slides/13-morphing.pdf>

H. B. Kekre, T. K. (2011). A novel pixel based color transition method for 2D image morphing. ICWET '11 Proceedings of the International Conference & Workshop on Emerging Trends in Technology (S. 357-362). New York: ACM.

Peisheng Gao, T. W. (1. 12 1998). A work minimization approach to image morphing. The Visual Computer, S. 390-400.

Sasikumar Gurumurthy, B. (July 2012). Design and Implementation of Face Recognition System in Matlab Using the Features of Lips. International Journal of Intelligent Systems and Applications, S. 30-36.

Shoujue Wang, X. L. (2011). A novel math method of image morphing. Computer Research and Development (ICCRD), 2011 3rd International Conference on (Volume:4) (S. 125 - 128). Shanghai: IEEE.

Watkins, C. (1993). Modern Image processing: Warping, Morphing, and Classical Techniques. Academic Press.

Yuan-Hui Yu, C.-C. C. (1. October 2006). A new edge detection approach based on image context analysis. Image and Vision Computing, S. 1090-1102.