Bewegungstransfer

auf 2D Zeichentrickcharaktere

Diplomarbeit

Ausgeführt zum Zweck der Erlangung des akademischen Grades
Dipl.-Ing. für technisch-wissenschaftliche Berufe

am Masterstudiengang Digitale Medientechnologien an der
Fachhochschule St. Pölten, **Masterklasse Experimentelle Medien**

von:
Lorenz Pritz, BSc
1610262571

Betreuer und Erstbegutachter: FH-Prof. Mag. Markus Wintersberger
Zweitbegutachter: Dipl.-Ing. Thomas Wagensommerer, MA BA BSc

St. Pölten, 14.01.2020

Ehrenwörtliche Erklärung

Ich versichere, dass

- ich diese Arbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfe bedient habe.

- ich dieses Thema bisher weder im Inland noch im Ausland einem Begutachter/einer Begutachterin zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.

Diese Arbeit stimmt mit der vom Begutachter bzw. der Begutachterin beurteilten Arbeit überein.

St. Pölten, 14.01.2020 

Ort, Datum Unterschrift

Kurzfassung

Das Thema Bewegungstransfer auf 2D Zeichentrickcharaktere befindet sich an der Schnittstelle zwischen den Fachbereichen maschinelles Lernen, Computersehen und motion capture. Der Fokus liegt bei klassischem motion capture darauf, dreidimensionale Daten über Gelenkspositionen zu erfassen und wieder auf 3D Charaktere anzuwenden. Bei Bewegungstransfer mittels maschinellen Lernens wird der Zwischenschritt in die dreidimensionale Sphäre übersprungen. So kann im Vergleich zu traditionellen Trickfilmanimationstechniken relativ rasch, jedoch zum aktuellen Zeitpunkt mit weniger Kontrolle, ein synthetisierter Charakter in neuen Zielposen berechnet werden. Die Zielposen werden dabei aus einer Bewegungsquelle extrahiert.

Eine besonders relevante Fragestellung ist die grundsätzliche Möglichkeit und erreichbare Qualität der Erfassung von Posen aus gezeichneten Bildern eines Zielcharakters.

In dieser Diplomarbeit werden im ersten Schritt etablierte Möglichkeiten der Bewegungserfassung erarbeitet. Diese Informationen werden dann als Basis genutzt, um einen auf maschinellem Lernen und Computersehen basierenden State of the art-Softwareansatz zum Bewegungstransfer in der Landschaft der Animationswerkzeuge zu verorten. Die der Software zugrundeliegende Funktionsweise wird dabei ebenfalls theoretisch erarbeitet.

Den Hauptteil der Arbeit stellt ein praktischer, kontrolliert gestalteter Versuch dar. In diesem zeige ich, dass die verwendete Softwarelösung zum Bewegungstransfer auf einen gezielt und begründet gewählten 2D Zeichentrickcharakter in ähnlichem Ausmaß möglich ist wie der Bewegungstransfer auf denselben Charakter, wenn dieser für das Training der Software auf 3D Renderings basiert. Qualitative Beobachtungen und Vergleiche mit z.B. klassischem Videomaterial erlauben darüber hinaus weitere Annahmen über das Ergebnis des vergleichenden Hauptexperiments und die Funktionsweise des Bewegungungstransfers, wie er hier durchgeführt wird.

Abstract

Motion transfer from a person in a source video to another person in a synthesized target video is at the intersection of the disciplines machine learning, computer vision and motion capture.

Traditional motion capture techniques seek to gather and later apply 3D joint coordinates onto a 3D model of a character. Motion transfer based on software techniques skips the intermediate step into the three-dimensional space. In doing so, it is, considering the necessary input by the user, faster than traditional 2D and 3D animation techniques. The tradeoff for the faster synthesis of a character or person in new unseen poses is a currently very limited amount of control over the intermediate steps, as well as a long preparation and training phase. For achieving this 2D transfer, the target poses of the final render are extracted from a source video.

Core questions are whether it is possible to extract poses from drawn source material as well as the achievable quality of the extracted poses.

The first part of this thesis gives an overview on the current landscape of motion capture. On this basis, it will be easier to assess the usability of motion transfer with its current drawbacks in comparison to well-established 2D and 3D animation tools. The state-of-the-art software method to achieve motion transfer will also be analyzed from a theoretical perspective.

A practical experiment designed in a controlled fashion, will present the main part and findings of this thesis. It will demonstrate that the here used software method for motion transfer is equally useful for a specifically chosen 2D character in comparison to the same character based on 3D renderings. Qualitative observations while conducting the experiment a well as further comparisons with, e.g. traditional video footage of a real target person, will allow for additional assumptions about the current possibilities and obstacles in achieving motion transfer in respect to 2D drawn characters.

Inhaltsverzeichnis

1 Einleitung

Abbildung 1: Veranschaulichung des Transfers der Bewegung eines Tänzers (Mars & Lia, 2017) auf den 2D Animecharakter Son Goku

Bewegung als Gestaltungsmittel: Meine Motivation die Aufgabenstellung des Bewegungstransfers auf einen gezeichneten Charakter zu bearbeiten, kommt aus einer persönlichen Leidenschaft für Bewegung. Insbesonders Tanz und Parkour begleiten mich in meiner Auseinandersetzung mit zeitbasierten visuellen Medien. Bewegung ist auch für traditionelle Film- und Videoproduktionen ein höchst relevantes Gestaltungsmittel. Sobald sich ein Charakter oder die Kamera bewegt, erregt dies Aufmerksamkeit. Bei der Fragestellung wie einem fiktiven gezeichneten Charakter Leben eingehaucht werden kann, ist Bewegung ebenfalls von zentraler Bedeutung. Bewegung ist ein Gestaltungsmittel. Gezeichnete Charaktere enthalten zusätzlich den persönlichen Ausdruck der KünstlerInnen, die sie erschaffen. Die schrittweise Animation von sich realistisch anmutend bewegenden 2D Charakteren ist nach wie vor zeitaufwändig. Im Gegensatz dazu liefert klassisches Motion Capture für 3D Animation bereits eine Vereinfachung in vielen Fällen, auch wenn dadurch die händische Keyframeanimation nicht obsolet wird. Die Unterstützung, welche Motion Capture heute bereits für 3D Animation darstellt, wäre auch für die 2D Trickfilmanimation wünschenswert. Vielleicht hilft diese Arbeit hier, einen kleinen Schritt näher an stiltreue Animation von 2D Zeichentrickcharakteren heranzurücken, ohne ein 3D Modell verwenden zu müssen.

Ich gebe in dieser Arbeit einen kurzen Überblick über den aktuellen Stand der Bewegungserfassung mittels etablierten Motion Capture Werkzeugen inklusive

eines kleinen Versuchs. Diesem folgt eine Darstellung von neuen Methoden zur Bewegungserfassung durch Fortschritte aus dem Bereich maschinelles Lernen und Computersehen (engl. Computer Vision). Danach gebe ich einen Überblick über die dem maschinellen Lernen zugrundeliegenden Konzepten, wie sie in der Hauptquelle für den Ablauf des Bewegungstransfer, dem Paper „Everybody Dance Now“ zum Einsatz kommen (Chan et al., 2018). Die darin vorgestellten Abläufe, Methoden und Konzepte, werden im nächsten Abschnitt behandelt.
Der praktische Bewegungstransfer stellt darauffolgend den Kern meiner Arbeit dar. Sowohl die dafür zum Einsatz kommenden Datensätze als auch die Versuchsdurchführung sind entlang der Logik eines wissenschaftlichen Experiments gestaltet. An Aussagekraft gewinnen die Ergebnisse der Versuchsdurchläufe jedoch erst durch eine kontrollierte Betrachtung. Dafür kommen mehrere Metriken zum Einsatz, wie z.B. die Wahrnehmungsdistanzmetrik LPIPS (Zhang et al., 2018).
Ein gut trainiertes neuronales Netzwerk erzielt nach einiger Zeit des Optimierens auf das Trainingsdatenset hochwertige generative Ergebnisse. Idealerweise generalisiert dieses auch in ähnlicher Qualität auf neue Daten der gleichen Art. Dies ist bei gut gewählten Trainingsdaten und erfolgreich trainierten Netzwerken auch dann der Fall, wenn die neuen Testdaten nicht im Trainingsdatenset enthalten sind. Der Unterschied zwischen dem Fehler bei der Analyse der Trainingsdaten und der Analyse neuer Daten ist idealerweise minimal. Das Ziel für das Netzwerk ist es also "fit" für neue Daten zu sein. Erste Voraussetzung dafür ist es, im Training gute Ergebnisse zu erzielen. Wie schnell und ob gute Ergebnisse im Training erzielt werden sind wichtige Datenpunkte bei der Bewertung des Trainingsalgorithmus. Die grunsätzliche Generalisierfähigkeit eines, mittels der in „Everybody Dance Now“ vorgestellten Methode, auf hochwertigen Zieldaten trainiertes neuronales Netzwerk kann als gegeben angenommen werden. Daher konzentriert sich diese Arbeit darauf die Trainingsergebnisse der neuronalen Struktur bei gezeichnetem und im kontrollierten Vergleich dazu stehenden gerenderten Ausgangsdaten des Animecharakters Son Goku gegenüberzustellen, um daran die Machbarkeit von Bewegungstransfer auf 2D Zeichentrickcharaktere zu erarbeiten.
Qualitative Beobachtungen eines vollständig durchgeführten Bewegungstransfer, sowie ein kurzes Fazit und ein Ausblick auf durch die Ergebnisse neu gewonnenen Forschungsperspektiven, schließen die Arbeit ab.
Auf dem dieser Arbeit beigefügten Datenträger sind die frei verfügbaren zum Einsatz kommenden Elemente meines praktischen Teils angefügt. Dazu zählt das Programm „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) sowie die gewonnen Ergebnisse meiner durchgeführten Versuche, z.B. die Videos der Bewegungstransfers.

2 Begriffsdefinition motion capture

Definition: Menschen bewegen ihren Körper unter anderem, um von A nach B zu gelangen, um zu trainieren, oder um einer Stimmungslage, beispielsweise beim Tanzen, Ausdruck zu verleihen. Häufig ist von der Art und dem Stil der Bewegung weiter interpretierbare Information über Gesundheitszustand, Tagesverfassung, Energielevel, Motivation und vieles mehr ableitbar. An dieser Information können je nach Kontext mehrere Parteien interessiert sein – mit verschiedensten Zielen der Auswertung. Ein Physiotherapeut möchte vielleicht den Gesundheitszustand bei der ersten Therapie ermitteln oder ein Gast sich von einer Balletaufführung unterhalten lassen.
Wie ein Großteil zeitbasierter Information ist auch die menschliche Bewegung flüchtiger Natur. Um diese Information überdauern zu lassen oder genau auszuwerten, bedarf es einer Möglichkeit, diese zu speichern. Bewegungserfassung (englisch: motion capture) hat genau dieses Ziel. In welcher Form die Bewegung gespeichert und verwendet wird, ist von Fall zu Fall verschieden und hängt von der weiteren Nutzung der Daten ab.

Einteilung nach Nutzen: Moeslund, Hilton, & Krüger (2006, S. 90) teilen die Nutzbarkeit der Bewegungsinformation in drei Bereiche ein, welche vom Ziel der Anwendung der gesammelten Daten ausgehen. Diese sind Überwachung, Kontrolle, sowie Analyse.

Überwachung ist im öffentlichen Raum nicht wegzudenken. Da hier legitime wie auch moralisch kaum legitimierbare starke Interessen vorliegen, beispielsweise in Fragen der Sicherheit, sind diese Interessen ein wichtiger Treiber der Forschung im Bereich der Bewegungserfassung.

Weiterverarbeitung/Kontrolle: Die Absicht, die Bewegungsdaten nach der Aufzeichnung modifizieren und kontrollieren zu können, ist das klassische Paradebeispiel aus der Unterhaltungsindustrie. In Filmen, Spielen, oder auch Messagingdiensten, wie z.B. dem Facebook Messenger, ist es gewünscht, die Bewegungsinformation auf einen digitalen Charakter (z.B. ein Avatar) oder auf eine digital ‚aufgehübschte' Version seiner selbst zu übertragen. Je mehr kreative Kontrolle man über diese Daten oder das Modell, welches die Daten repräsentiert, hat, desto größer ist der Möglichkeitsrahmen, wie diese angewandt werden können. Der Wunsch, die gewonnenen Bewegungsdaten weiterzuverarbeiten ist

aber nicht auf die Unterhaltungsindustrie beschränkt. In diversen Softwarebereichen, wie z.B. der Entwicklung von Smartphoneanwendungen wünschen sich auch Appentwickler Kontrolle über gewonnene Bewegungsdaten. Weit verbreitet ist beispielsweise das Tracking von sportlicher Aktivität über den Verlauf eines Tages mittels tragbarer Geräte oder auch die Steuerung diverser Umgebungen mittels Gesten. Die untenstehende Werbung verdeutlicht das weite Feld der kommerziellen Nutzbarkeit von Bewegungsdaten (Hillcrest Laboratories, 2015).

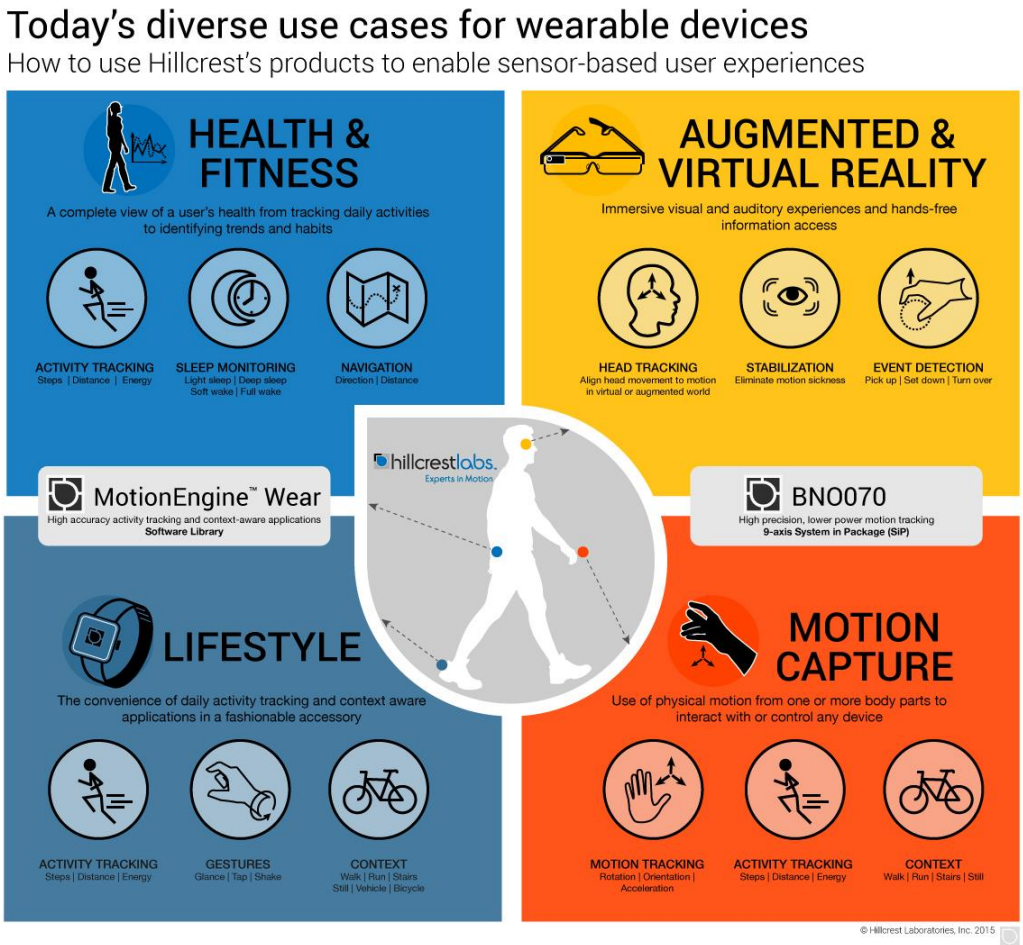


Abbildung 2: Beispiele kommerzieller Nutzbarkeit von Bewegungsdaten

Analyse: Der dritte Nutzungsfall ist die Analyse der Bewegungsinformationen. Besonders medizinische, biomechanische und sportliche Fragestellungen bedienen sich gerne dieser Informationen. So vielfältig wie die menschliche Erscheinungsform und die ausführbaren Bewegungen sind, so vielfältig scheinen auch die Möglichkeiten zur Erfassung dieser Bewegung.

Aktiv/Passiv: Ausgehend von den Motion Capture Kontexten kann in aktive und passive Erfassung unterteilt werden. Aktiv bedeutet, dass während der Bewegung diese mittels z.B. Infrarotlicht direkt mitgeschrieben wird. Ein weiteres Beispiel dafür wäre sensorbasiertes Capturing oder auch ein angeschnalltes Exoskelett. Meist sind die aktiven Methoden für Motioncapture genauer als die passive nachträgliche Auswertung von Bildern. Jedoch sind im Gegenzug meist besondere Anforderungen an die Aufzeichnungsumgebung gegeben, oder die Technologie ist invasiv, sprich sie beeinflusst die Probanden in ihrer Bewegung und verfälscht dadurch das Resultat (Mündermann et al., 2006, S. 3).
Passive Methoden finden seit Fortschritten in der Software zur Auswertung der Aufzeichnungen vermehrt Vebreitung. Das gängigste Beispiel ist optisches Motioncapture mit und ohne Marker und mit einer oder mehreren Kameras. Besonders die markerlose Erfassung mit einer Kamera ist besonders interessant, da Sie die wenigsten Anforderungen an den Aufbau stellt und auch die Bewegungsfreiheit nicht einschränkt. Jedoch sind aktive Methoden zum aktuellen Zeitpunkt immer noch von vergleichsweise hoher Qualität, weshalb diese in der Biomechanik, der Medizin und dem Sport nach wie vor zum Einsatz kommen. Besonders, aber nicht nur im Überwachungsbereich gilt jedoch der Leitsatz: Je geringer der technische Fußabdruck der Bewegungserfassung, desto besser.

Modellbasiert: Wie die Daten einer Erfassung betrachtet und weiterverarbeitet werden ist in modellbasiert oder modellfrei einteilbar (Mündermann et al., 2006, S. 4). Gibt es eine Repräsentation in Form eines digitalen Skeletts oder Gelenks, so stellen sich die Fragen, wie flexibel diese an die Erscheinung der Schauspieler anpassbar ist und wie genau die menschliche Natur abgebildet werden kann. Die Unterhaltungsindustrie bedient sich häufig Methoden der Bewegungserfassung, welche ein solches Modell als Zwischenschritt verwenden. Wenn man sich beispielsweise die Daten eines Kniewinkels von Patienten nach einer Knieoperation ansieht, passiert dies hingegen häufig ohne eine Repräsentation der Person durch ein Modell. In Analysekontexten wird teilweise sogar gänzlich auf eine Veranschaulichung als Modell verzichtet.

Unterhaltungsindustrie: Die Anwendungszwecke von Motioncapture innerhalb der Unterhaltungsindustrie können wiederum in mehrere Kategorien eingeteilt werden. Zu den Nutzungsszenarien zählen unter anderem Film und Kino, (Online)Video, Videospiele, augmentierte Realität (AR), virtuelle Realität, soziale Medien und Videotelefonie.

3 Etablierte motion capture Lösungen

3.1.1 Technische Lösungsansätze

Motion Capture kann auf mehrere Arten eingeteilt werden. Am häufigsten wird nach den verschiedenen Techniken, die als Lösung zum Einsatz kommen, kategorisiert. Vlasic et al. unterscheiden zwischen sieben technischen Kategorien: Optisch, bildbasiert / ComputerVision, mechanisch, magnetisch, akustisch, trägheits- und lagesensorisch, Hybride / Mischformen (Vlasic et al., 2007, S. 35–2). Im Vergleich dazu teilen Szykman & Gois den Prozess in online mit Trackingmarker und offline ohne Trackingmarker ein (Szykman & Gois, 2014, S. 1133). Zu den bereits oben genannten technischen Ansätzen kommt hier des Weiteren noch die günstige Alternative der Infrarottechnik hinzu. Da bei Bewegungserfassung im ersten Schritt meist die Aufzeichnung der Bewegung des Körpers im Fokus steht, kommt Gesichtsmotioncapture häufig erst in einem zweiten Schritt zum Einsatz. Häufig verwendet man hier eine Kombination aus verschiedenen Technologien, wieso es beispielsweise von Menache separat angeführt wird (Menache, 2011, S. 31). Diese Kategorisierungen ergeben kombiniert folgende Einteilung zum Überblick:

Technologie	*online mit Marker*	*offline ohne Marker*	*kamerabasiert*	*niedrige Kosten*
Optisch (mit Marker)	x		x	
Bildbasiert / ComputerVision		x	x	x
Mechanisch		x		
Elektromagnetisch		x		
Akustisch / hochfrequenztracking		x		
Trägheits- und Lagesensorisch		x		(x)

Hybride / Mischformen	x	x	x	
Gesichts motion capture	x	x	x	
Infrarot		x	x	x

Tabelle 1: Mögliche Einteilung von Methoden zur Bewegungserfassung (x = trifft zu, (x) = trifft manchmal zu)

Jene technischen Lösungsansätze, welche Kameras verwenden (optisch, bildbasiert, hybrid, Gesichtstracking und Infrarot), eint das Problem der Verdeckung. Um diese Schwierigkeit zu umgehen, können mehrere Kameras gleichzeitig zum Einsatz kommen. Daher findet sich auch häufig die zusätzliche Einteilung in Ein- oder Mehrkameralösungen. Bei Mehrkameralösungen besteht der Vorteil darin, dass Verdeckungen von Körperteilen durch die Schauspieler selbst kein Problem mehr darstellen, wohingegen bei einer Einkameralösung diese Verdeckungen teils zu erheblichen Problemen und Ungenauigkeiten führen, bis hin zur Unbrauchbarkeit der Daten. Bei Interaktionen mit Objekten ist jedoch auch eine Mehrkameralösung nicht vor Verdeckungen gefeit.

3.1.2 Markt

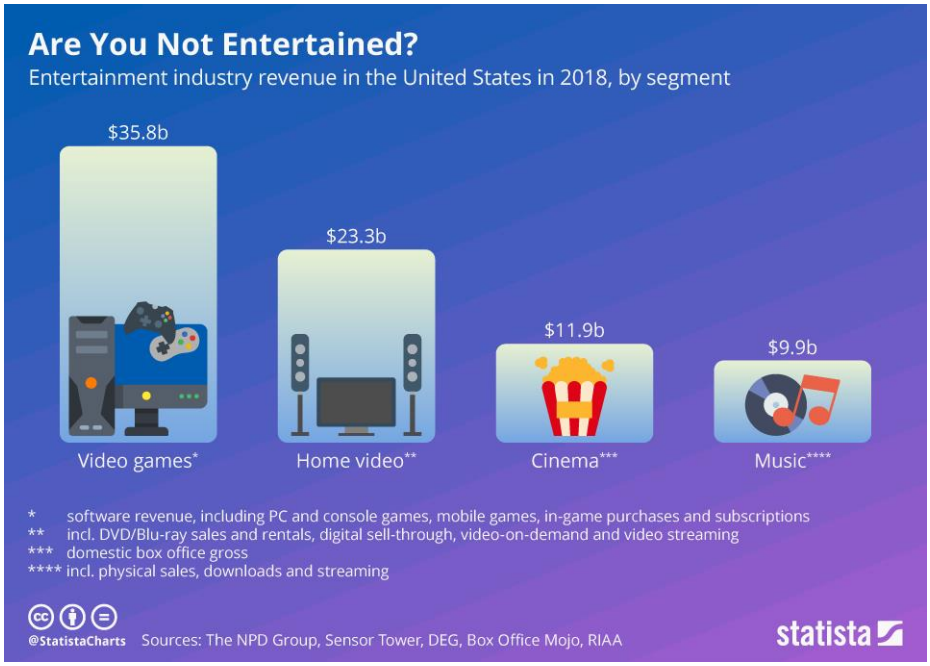


Abbildung 3: Marktanteil nach Sektor - Unterhaltungsindustrie USA 2018 (Richter, 2019)

Motion Capture Markt in traditionellen Unterhaltungsmedien: Der Markt für Motioncapture im Unterhaltungssektor ist abhängig vom Konsumverhalten der Bevölkerung. Werden viele Filme rezipiert, die computer generierte Bilder (CGI) oder auch 3D Charakteranimation einsetzen, führt dies beispielsweise auch zu einer erhöhten Nachfrage der Technologie zur Bewegungserfassung auf der Film- und Fernsehproduktionsseite.
In der vorhergehenden Abbildung ist sichtbar, wie groß in einem der Hauptmärkte, den USA, die Videospielindustrie im Vergleich zu anderen Bereichen der Unterhaltungsindustrie ist. Da Computerspiele häufig sich wiederholende Bewegungen von menschlichen oder menschenähnlichen Charakteren verwenden, ist die Videospielbranche als Hauptsektor für Motion Capture zu betrachten. Diese Bewegungen werden meist von auf genau diesen Markt spezialisierten Motion- bzw. PerformancecapturekünstlerInnen ausgeführt. Seltener werden auch die Bewegungen von Tieren, wie z.B. Hunden, erfasst.

Industriestandard: Die größte Verbreitung in der Produktion von hochqualitativen zeitbasierten visuellen Medien finden optische 3D Motioncapture Systeme mit reflektierenden Markern, wie z.B. von Vicon (Vlasic et al., 2007, S. 2). Durch Triangulation kann mit den Mehrkamerasystemen von Vicon in Echtzeit die Bewegung von mehreren Personen hochqualitativ erfasst werden. Sowohl bei

Animationsfilmen, mit CGI gemischten Realfilmen als auch bei 3D Videospielen ist es die Absicht, dreidimensionale Daten aufzuzeichnen und diese auf ein Modell eines 3D Charakters zu übertragen. Da kommerzielle Videospiele, aber auch Filme häufig ein größeres Budget zur Verfügung haben, können aufbauintensive und komplex zu bedienende Lösungen zum Einsatz kommen. Unabhängige Film- und Videospielproduktionen greifen daher seltener auf Lösungen von Vicon oder ähnlich hochpreisigen Anbietern zurück.

Neue Medien: Aktuell entwickelt sich durch aufkeimende neue Medien wie virtuelle Realität (VR) sowie augmentierte Realität (AR) die Notwendigkeit und dadurch auch der Markt, die Bewegung einer Rezipientin bzw. eines Rezipienten zu erfassen, um z.B. das Sich-umblicken in der virtuellen Realität abbilden zu können. Interaktion und Immersion durch natürliche Bewegung in der virtuellen Welt ist hier die Begrifflichkeit, welche die Notwendigkeit zur Bewegungserfassung bereits impliziert.

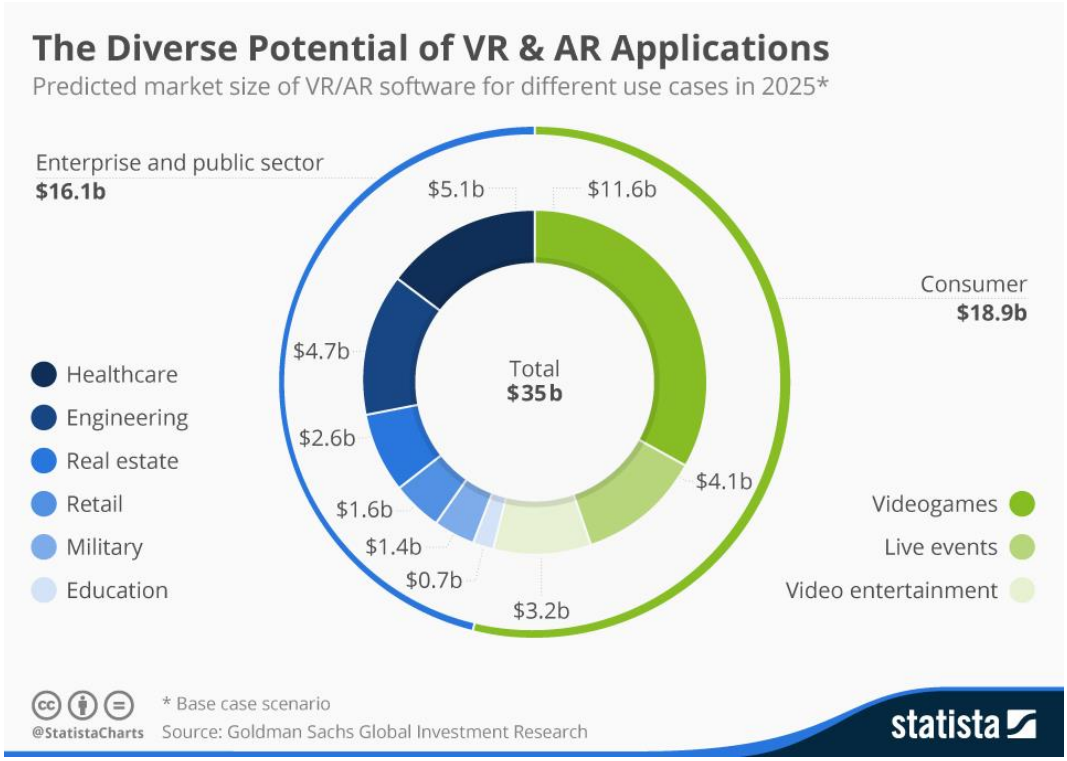


Abbildung 4: Potential von VR und AR Anwendungen (Richter, 2016)

VR/AR Gerätemarkt: Die vorhergehende Abbildung bildet den Softwaremarkt von VR und AR Applikationen ab. Dem Absatzmarkt für die Geräte, mit denen VR und AR Inhalte rezipiert werden und in welchen die Technologie zur Bewegungserfassung direkt integriert ist, wird ebenso eine rosige Zukunft vorhergesagt.

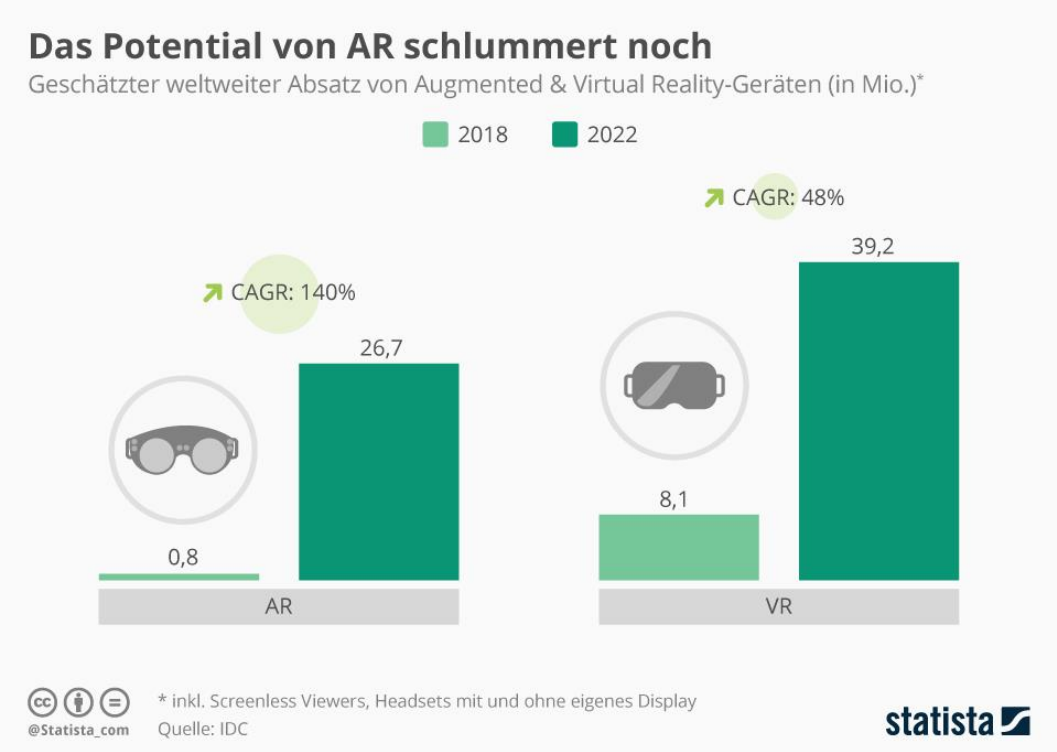


Abbildung 5: Potential von VR und AR Geräten (Brandt, 2018)

Spielkonsolen mit Bewegungserfassung: Auch in traditionellen Videospielen sind Methoden zur Bewegungserfassung direkt bei den RezipientInnen bereits kreativ zum Einsatz gekommen. Eine ganze Konsole, die Nintendo Wii, basiert darauf körperlich aktiv Videospiele zu erleben. Um die Bewegungen zu erfassen, bedient sich die Konsole eines Controllers mit MEMS Trägheitssensor, sowie einer Sensorleiste, welche auf Höhe des Monitors/Fernsehers platziert wird (Nintendo, 2019). Die Sensorleiste erinnert an die Microsoft Kinect Kamera, welche es der Masse and Heimanwendern erstmal ermöglicht hat, ein Tiefenbild zu nutzen. Bei der Microsoft Kinect kommt eine RGB Kamera, ein Tiefensensor und ein Array aus mehreren Mikrofonen zum Einsatz. Die Technologie zur 3D Tiefenrekonstruktion des Körpers verwendet „continuosly projected infrared structured light. This system called Light Coding employs a variant of image-based 3D reconstruction" (Darujati & Hariadi, 2013, S. 61).

Überblick der Anbieter: Der Markt für Bewegungserfassung ist vielfältig und groß. Dadurch wird deutlich, dass in Bewegung viel Information und auch

Potential zur Interaktion steckt, welches noch nicht ausgeschöpft wurde. Klassisches motion capture von Körper und Gesicht für Unterhaltungsmedien ist bereits seit den ersten Ideen, wie dem dem Rotoscope von Max Fleischer 1915, ein Thema. Mit dem Rotoscope wurde ein Film eines gefilmten Clowns Bild für Bild auf einen Leuchttisch projeziert, damit ein Künstler diesen nachzeichnen konnte (Menache, 2011, S. 3). Auch wenn die Technologie bis zur brauchbaren Aufzeichnung einer 3D Repräsentation der Bewegungen eines Schauspielers bzw. einer Schauspielerin und ihrer Etablierung in der Unterhaltungsindustrie noch einige Jahrzehnte schlummern musste, so ist motion capture heute nicht mehr aus der Medienproduktion wegzudenken. Um das aktuelle Angebot an Lösungen zur Bewegungserfassung zu veranschaulichen, ist in folgender Tabelle ein Überblick über die Anbieter verschiedener technischer Lösungen und ihre durchschnittliche Preisklasse, sofern verfügbar, gegeben.

Technologie	*Anbieter*	*durchschnittliche Preisklasse in USD*	*Information bezogen von*
Optisch	Codamotion, Vicon, Phasespace, OptiTrack	15.000 - 250.000,-	(Szykman & Gois, 2014, S. 1133)
Bildbasiert / Computer Vision	OpenPose, Densepose	0,- / OpenSource	(Cao et al., 2018)
Mechanisch / Exoskelett	Meta Motion Gypsy	7.995,-	(Meta Motion, 2019)
Elektromagnetisch	Ascension, Polhemus Fastrak, Premo	5.000 - 150.000,-	(Menache, 2011, S. 27)
Infrarot	Microsoft Kinect, Asus Xtion, Ipi Soft, Nui Capture	199 - 1.195,-	(Szykman & Gois, 2014, S. 1134–1135)

Lage/Trägheitssensoren	Perception Neuron, XSENS MVN	1.799 – 12.000,-	(Noitom, 2019), (Barsegyan, 2017)
Akustisch / Hochfrequenztracking**	GPS, LPS, Bat system, Cricket location system, WearTrack	Trackingfokus; Meist in Kombination mit anderen Sensoren um Drift zu verhindern	(Menache, 2011, S. 22–26) (Vlasic et al., 2007, S. 35–2)
Gesichts motion capture	OpenPose, Vicon, …	0 – 25.000,-	(AdamKK, 2014)
Hybrid	Nintendo Wii*, Constellation, Ascension Hy-BIRD	150 – 150.000,-	(Vlasic et al., 2007, S. 35–2)

Tabelle 2: Preisklassen verschiedener Anbieter von motion capture.
**Auf Bewegungserfassung einzelner Bewegungen/Gesten beschränkt*

Marktführer: In den Hochglanz-Studioproduktionen der Unterhaltungsindustrie, wie sie beispielsweise aus Hollywood bekannt sind, ist trotz des hohen Anschaffungspreis von mehreren hundertausend US-Dollar die optische Technologie mit Markern und mehreren Kameras, wie sie von Vicon angeboten wird, der Standard. Bei dem Marktsegment der unabhängigen Film- und Videospielproduktionen sind Systeme wie das OptiTrack Flex um $ 15.000,- oder der Lage- und Trägheitssensoranzug XSens MVN um ~$ 12.000,- am weitesten verbreitet (Szykman & Gois, 2014, S. 1133). Mit zunehmender Qualität der Softwarelösungen ohne Sensoren und Tiefendaten droht besonders im Niedrig- und Mittelpreissegment in den nächsten Jahren eine Verschiebung des Marktes. Treiber dieser Verschiebung ist die aktuell starke Forschung and Computer Vision Problemen mittels maschinell lernenden Softwaresystemen.

Maschinelles Lernen als Zukunftsmarkt: In einem Interview mit Andy Serkis, einem der bekanntesten motion capture Schauspieler, wie es 2014 in der Süddeutschen Zeitung online erschienen ist, antwortet dieser auf die Frage wohin sich die Technologie des motion bzw. performance capture entwickeln wird: " Wir werden einen Punkt erreichen, an dem die Technik nicht mehr störend sein wird und wir uns völlig frei bewegen können" (Schmieder, 2014, S. 2). Seit diesem

Interview hat sich besonders auf dem Bereich des maschinellen Lernens und Computersehens viel weiterentwickelt. Algorithmen zur Erfassung von Bewegung des Körpers und Gesichts liefern zunehmend hochwertige Daten. Ein Algorithmus zur nachträglichen Bewegungserfassung – OpenPose – ist auch zentraler Bestandteil des praktischen Versuchs dieser Diplomarbeit. Dennoch wird motion capture 2019 weiterhin vorwiegend mit etablierten Technologien, wie markerbasiertem optischen Merhkameratracking, durchgeführt. Ein Blick auf die marktwirschaftliche Größe und Vielfalt der Einsatzgebiete von maschinell lernenden Systemen, wie sie sich hinter dem Modebegriff künstliche Intelligenz (KI) verstecken, zeigt wie stark diese Fortschritte die gesamte Wirtschaft bereits jetzt beeinflussen.

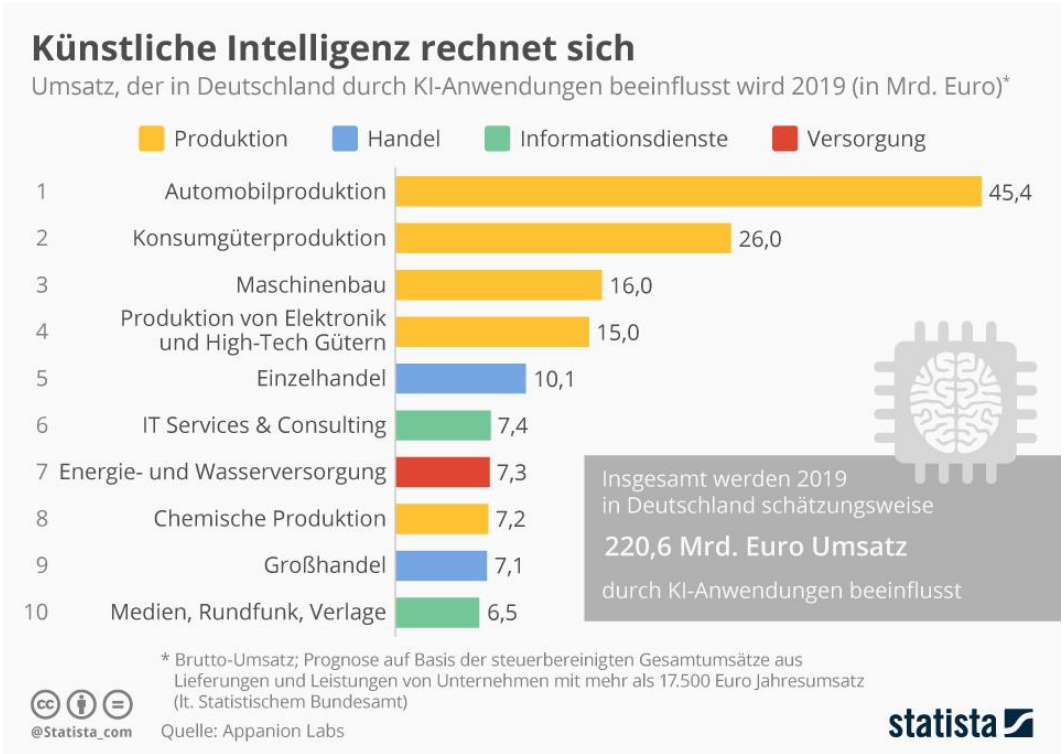


Abbildung 6: Durch KI Anwendungen beeinflusster Umsatz 2019 (Brandt, 2019)

Wie die Forschung in den vielfältigen Bereichen, in denen maschinelles Lernen zum Einsatz kommt, ist auch das Feld der 2D Bewegungserfassung aus geringqualitativen und spärlichen Ausgangsdaten, wie beispielsweise niedrig aufgelösten monokularen Videos, sehr aktiv.

3.1.3 Motion capture im Realfilm mit Computer Generated Imagery (CGI)

CGI und Bewegungserfassung: Realfilme, oder im Englischen auch Live Action Movies genannt, sind heute nicht mehr ohne CGI, i.e. gänzlich oder teilweise digital erstellte Bildinhalte, vorstellbar. Nicht nur Elemente wie Monster oder SuperheldInnen werden digital erstellt oder manipuliert, sondern auch Bereiche wie Set- oder Kostümgestaltung setzen Technologien zum nachträglichen Generieren oder Zusammensetzen von Bildelementen ein. Als die Technologie des motion capture und vor allem zum realisitschen Rendern von digitalen Charakteren noch in den Kinderschuhen steckte, war der Nutzen dieser Technologie ein höchst kontroverses Thema. Animationsberufe, die Keyframe für Keyframe händisch einem Charakter Leben einhauchten, schienen plötzlich ihre Daseinsberechtigung neben den wesentlich schnelleren und dadurch günstigeren Motion Capture Produktionen zu verlieren. Doch nach anfänglicher Euphorie stellte sich schnell heraus, dass Bewegungserfassung nicht für jeden Einsatzzweck mit nachträglich bewegten Charakteren geeignet ist. Anfangs war die Technologie speziell für Stunts und digitale Extras in Menschenmengen nützlich. Besonders die organisatorisch aufwendigen und kostenintensiven Menschenmengen, wie sie aus Monumentalfilmen bekannt sind, konnten mittels digitaler Charaktere für einen Bruchteil der Kosten ins Bild gesetzt werden (Menache, 2011, S. 43). Seit die Qualität digital erstellter Charaktere, nicht zuletzt dank der rapiden Entwicklungen der Computerhardware, immer realistischer gestaltet werden kann, finden sich diese auch vermehrt in Hauptrollen wieder. Die Charaktere werden dann teils von Starschauspielern der performance capture Welt, wie Andy Serkis, gemimt. Einige Beispiele aus den letzten Jahren sind: Der unglaubliche Hulk, Herr der Ringe (insbesonders Gollum), King Kong, Avatar, oder auch die Neuverfilmungen von Planet der Affen (Menache, 2011, S. 44). Wie diese kleine Liste bereits veranschaulicht, findet sich motion- bzw. performance capture heute vorwiegend in Realfilmen, die eine Mischung aus realen und fantastischen Charakteren zeigen.

3.1.4 Motion und performance capture Schauspiel

Menschen und Kreaturen: Fantastischen Charaktere sind nicht darauf beschränkt, menschliche Gestalt zu haben. Der Nutzen von Bewegungserfassung des Körpers nimmt zwar ab, je weiter man sich von der menschlichen Anatomie entfernt, dennoch finden sich die Daten, welche aus Gesichtsaufzeichnungen gewonnen werden, immer häufiger auch in fantastischen Wesen.

"Das ist nichts anderes als Schauspielerei" (Schmieder, 2014). So beschrieb Andy Serkis performance capture in einem Interview anlässlich des Films War of the Planet of the Apes, wo er den Anführer der Affen, Caesar, spielt. Lediglich um das Aussehen an sich, macht sich Andy Serkis bei einer Performance Capture Session weniger Gedanken. Er empfiehlt auch anderen Schauspielern, die „Seele“ des Charakters zu suchen und diese bestmöglich zu verkörpern (Schmieder, 2014, S. 2).

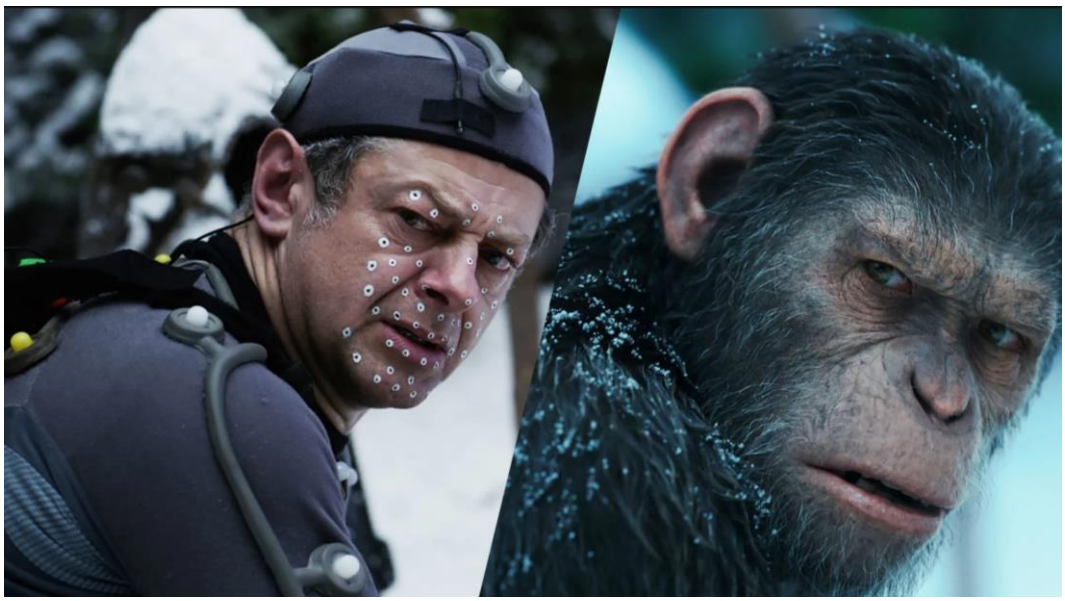


Abbildung 7: Andy Serkis als Ceasar in „War of the Planet of the Apes” (Reeves, 2017)

Glaubwürdigkeit: Alberto Menache beschreibt in seinem Buch über Motion Capture seine eigene Erfahrung einer Bewegungsaufzeichnung eines Menschen und eines Schimpansen im direkten Vergleich, jeweils mit dem Ziel, einen digitalen Affen zu bewegen. Der Mensch wirkte dabei im Vergleich zu der Qualität der echten Bewegungserfassung des Schimpansen immer wie ein Mensch im Affenkostüm. Daraus schließt Menache, dass es höchts komplex ist, ein Tier glaubwürdig durch einen Menschen spielen zu lassen und rät grundsätzlich davon ab (Menache, 2011, S. 78). Erfolgreiche performance capture Produktionen, wie Planet der Affen, zeigen jedoch, dass durch spezialisierte Schauspieler durchaus ein Mehrwert besteht bei der Übertragung menschlicher Bewegung auf Tiere oder Fabelwesen.

Grenzen der Nutzbarkeit: Sogar für den sehr weit abseits menschlicher Physiognomie stehenden Drachen Smaug in dem Film Der Hobbit: Smaugs Einöde wurde bereits Performance Capture angewandt. Starbesetzt durch Benedict Cumberbatch wird hier versucht, dem Drachen durch hochwertiges

Schauspiel Leben einzuhauchen. Benedict Cumberbatch beschreibt seine Herangehensweise in einem Interview wiefolgt: "It was important to bring some anthropomorphic quality to how we created him, [...] I wanted to do some motion capture, and steer a little bit of the animation in his face." (Lamble, 2013)

Abbildung 8: Benedict Cumberbatch als Smaug in „Der Hobbit: Smaugs Einöde“ (Jackson, 2013)

Übliche Praxis: Aus diesen Erfahrungsberichten und auch der Beschreibung von Menache lässt sich ableiten, dass selten die Performance Capture Information der Körperbewegungen der Schauspieler direkt auf einen digitalen Charakter übertragen werden, ohne das Animatoren diese zur Gänze oder teilweise händisch nachbessern müssen. Bei Gesichtsausdrücken ist der Nutzen stärker gegeben, da das Publikum auch in fiktiven Kreaturen wie dem Drachen Smaug menschliche Eigenschaften in Form von Mimik sucht.

3.1.5 Motion capture im Animationsfilm

Im Gegensatz zum Realfilm stellen Animationsfilme visuell keinen Realitätsanspruch. Daher finden sich hier unrealistische und teils auch nach unserem Verständnis der Naturgesetze unmögliche Körperformen, Fortbewegungsarten, Verformungen von Gegenständen und Charakteren etc. Der Kreativität sind hier kaum Grenzen gesetzt. Jeder der selbst bereits versucht hat, einen Charakter Keyframe für Keyframe zu bewegen, weiß, wie viel Arbeit dies ist. Da liegt der Griff zur motion capture Technologie nahe. Doch obwohl es einfacher und weniger kostenintensiv erscheint auf Bewegungserfassung an Stelle händischer Animation zu setzen, finden sich nur wenige Animationsfilme, welche gänzlich mittels Motion Capture Technologie produziert wurden (Menache, 2011, S. 53).

Uncanny Valley: Wenn ein digitaler Mensch oder auch ein Roboter der Erscheinung eines echten Menschen stark ähnelt, ihn aber nicht ganz glaubwürdig abbilden kann, führt das beim Publikum zum Gruseln. Die Charaktere befinden sich dann im "Uncanny Valley" – nicht Mensch, nicht fiktive Figur. Das Publikum ist besonders gut darin, andere Menschen als solche zu erkennen. Wenn ein digitales Tier wie ein Hund oder ein Affe nicht ganz der Realität entspricht, fällt dies weniger auf (Schmieder, 2014). Diese Erscheinung einer Figur ist jedoch nur ein Teilaspekt, welcher das Publikum verstören kann. Auch wie sich ein Mensch real fortbewegt, kann das Publikum schnell erkennen. Daher sticht menschliche Bewegung unreflektiert und ohne Sorgfalt als Bewegung einer fiktiven Figur angewandt schnell ins Auge. Unter anderem aus diesem Grund sollte sich eine Produktion, besonders bei Animationsfilmen, gut überlegen ob, für welche Charaktere, und wie genau motion capture zum Einsatz kommen soll.

Illusion und Realität: Viele Fragestellung, die in der Geschichte und Entwicklung der modernen 2D und 3D Animationswelt beantwortet wurden, beschäftigen sich damit, wie eine ansprechende Illusion erzeugt werden kann. Eines der Standardwerke für Animationsstudenten von Frank Thomas und Ollie Johnston trägt beispielsweise den Titel "The Illusion of Life, Disney Animation". Dieses Buch beschäftigt sich damit, wie durch händisch gezeichnete Bewegung die Illusion einer lebendig wirkenden Figur erreicht werden kann. Auch wie es technisch gelingt durch eine Bildfolge von gezeichneten Charakteren Bewegung zu erzeugen, wird meist als Illlusion beschrieben (Musa et al., 2013). Der Illusion steht, zumindest im allgemeinen Sprachgebrauch, die Realität gegenüber. Dem Publikum ist beim Rezipieren eines 3D Animations- oder 2D Zeichentrickfilms durchaus bewusst, dass die Charaktere und die Welt, in der sie agieren nicht real sind und das was sie sehen gänzlich von KünstlerInnen erschaffen wurde. Überhöhte Bewegungen von Charakteren werden dabei nicht nur akzeptiert, sondern tragen gekonnt eingesetzt sogar zu einem noch intensiveren filmischen Erlebnis bei. Daher stellt sich auch hier die Frage, inwiefern Daten, welche möglichst genau die Bewegung einer Schauspielerin/eines Schauspielers in der realen Welt abbilden, überhaupt geeignet sind etwas zu der Illusion eines Animationsfilmes beizutragen.

3.1.5.1 Animationsprinzipien

Überhöhte Bewegung als Gestaltungsmittel: Die Unterschiede zwischen der Gestaltung von Bewegung durch traditionelle händische Animations- und Zeichentricktechniken und der Erzeugung von Bewegungsdaten durch Aufzeichnung von SchauspielerInnen sind besonders sichtbar im Vergleich mit den zwölf Animationsprinzipien wie sie in "The Illusion of Life: Disney Animation"

erstmals postuliert wurden (Thomas & Johnston, 1981). In der hier folgenden Auflistung beschreibe ich kurz die Bedeutung der Animationsprinzipien im Hinblick auf die Bewegung von Objekten und Charakteren. Die englischen Bezeichnungen sind in der Literatur nicht einheitlich, weshalb ich hier die Bezeichnungen von Menache übernehme (Menache, 2011, S. 79–80).

Squash and Stretch / Komprimieren und Strecken: Eine Bewegung führt dazu, dass sich der Charakter oder das Objekt beim Beschleunigen wie eine zähe Masse in die Länge zieht und beim Abbremsen komprimiert. Das Volumen des Sub- bzw. Objekts bleibt dabei gleich.

Timing / Zeitliche Positionierung: Eine Bewegung sollte im richtigen Moment beginnen, eine glauwürdige Dauer aufweisen und eine entsprechende Reaktion bewirken. Wie diese Momente auf der Zeitleiste platziert sind, ist nicht trivial. Wann reagiert oder agiert ein Charakter? Der richtige Zeitpunkt für einen bestimmten, beispielsweise komischen, Effekt muss erst getroffen werden.

Anticipation / Antizipation: Idealerweise sollte man ab und an dem Publikum die Möglichkeit geben, Bewegung vorauszuahnen. Ein Charakter streckt sich beispielsweise erst in die Gegenrichtung, um Schwung zu holen und läuft dann in die entgegengesetzte Richtung los. Das Publikum kann dabei vorausahnen, wie sich der Charakter bewegen wird.

Staging / Inszenierung: Das sogenannte Staging liefert die Antwort auf die Fragen, wann Bewegungen wo im zeitlichen Ablauf und der räumlichen Welt der Geschichte platziert werden. Dies sollte bewusst gestaltet werden.

Follow through and overlapping action / Übergänge von einer Bewegung in die nächste: Eine Figur, die Bewegungen nicht kombinieren kann oder direkt aus einer in die nächste übergeht, erscheint steif. Ziel ist es, möglichst glaubwürdig Bewegungen zu verbinden und Effekte von vorhergehenden Bewegungen nicht zu ignorieren.

Straight-ahead action and pose-to-pose action / Bild-für-Bild oder Pose-für-Pose: Beschreibt die Frage ob es notwendig ist jedes Einzelbild einer Bewegung zu gestalten oder ob es ausreicht einzelne Keyframes für wichtige Posen zu setzen und den Rest dazwischen zu interpolieren (Früher: Inbetweening).

Ease-in and ease-out / Langsames Ein- und Ausschwenken: Bewegung beginnt selten abrupt, sondern folgt meist einer gleichmäßigen Kurve in Abhängigkeit der zu bewegenden Masse und der zur Verfügung stehenden

Energie. Wenn ein Charakter zum Stopp kommt, trifft dasselbe zu, nur in umgekehrter Form.

Arcs / Bögen: Die meiste Bewegung folgt keinem geradlinigen Weg, sondern einer Kurve. Dies sollte sich auch in der Animation wiederfinden.

Secondary Motion / Sekundärbewegung: Nachdem der Hauptcharakter bewegt wurde, sollte nicht darauf vergessen werden, auch zusätzliche Elemente wie Kleidung zu animieren.

Exaggeration / Übertreibung: Bewegungen können übertrieben oder auch übertrieben subtil gestaltet werden, um beispielsweise Absichten oder Gefühle der Charaktere zu unterstreichen.

Appeal / Attraktivität: Bewegung sollte für das Publikum interessant und ansprechend sein.

Personality / Persönlichkeit: Durch Bewegung kann der individuelle Charakter einer Figur zum Vorschein kommen.

Animation kann und darf mehr als die reale Welt es uns Menschen zulässt. Dies ist nicht zuletzt an den Animationsprinzipien abzulesen. Da aber auch die händische Animation mit realer Physik spielt, gibt es doch Übereinstimmungen zwischen Motion Capture und Keyframe oder auch Bild für Bild Animation. In folgender Tabelle habe ich jene Prinzipien gelistet, welche in einer Bewegungserfassung mit Schauspielern berücksichtigt werden können. Diese stehen den Prinzipien gegenüber, welche durch die zentrale Rolle der Überhöhung der realen Physik für eine Motion Capture Session kaum machbar sind (Menache, 2011, S. 81).

Animationsprinzip	*Geeignet für Motion Capture*	*Ungeeignet für Motion Capture*	*Unabhängig von Animationstechnik*
Komprimieren und Strecken		x	
Zeitliche Positionierung			x
Antizipation	(x)	x	
Inszenierung			x

Übergänge von einer Bewegung in die nächste	(x)	x		
Bild-für-Bild oder Pose-für-Pose	x			
Langsames Ein- und Ausschwenken	x			
Bögen	x			
Sekundärbewegung	x			
Übertreibung		x		
Attraktivität				x
Persönlichkeit				x

Tabelle 3: Machbarkeit der zwölf Animationsprinzipien mittels Motion-Capture
(x) = nicht in überhöhtem Ausmaß machbar

Anreicherung der Daten: Animation oder gar Zeichentrick werden durch das spielerische Gestalten von Bewegung oft erst interessant für die Rezipienten. Dieser Cartoonstil kann nicht uneingeschränkt durch Schauspieler erreicht werden, wie die vorhergehende Tabelle zeigt. Dennoch gibt es sowohl prozedurale als auch händische Ansätze von Motion Capture Daten ausgehend diese Überhöhungen hinzuzufügen. Dies stellt sich jedoch häufig als aufwendiger heraus als eine von vornherein händische Keyframe Animation der Charaktere. Der Animationsfilm Shrek (DreamWorks, 2001) setzt beispielsweise kein Motion Capture ein, wohingegen Happy Feet (Warner Bros., 2006) mittels Bewegungsdaten aus Performance Capture animiert wurde (Menache, 2011, S. 81). Beide Filme haben es geschafft den Oscar für den besten animierten Spielfilm zu gewinnen.

3.2 Demokratisierung durch Sensoranzüge

Faktoren für Vergünstigung: Tabelle zwei zeigt einen Überblick über die Technologien zum Erfassen und Aufzeichnung von Bewegung. Neben dem besonders aktiven Bereich der Softwareentwicklung ist durch Beschleunigungs/Trägheits-, Lage-, sowie Magnetsensortechnik Motion Capture für eine breite Masse and Low Budget Produktionen zugänglich. Sowohl die günstigere Hardware der Sensoren als auch die Entwicklung der dazugehörigen Software um die Daten direkt auf ein brauchbares repräsentatives Skelett/Modell und anschließend in von Animationssoftware lesbare Bewegungsdaten zu übertragen führen dazu, dass Bewegungserfassung demokratisiert wird. Maschinelles Lernen zur Bewegungserfassung ist jedoch keine neue Entwicklung. Das Interesse und die Möglichkeiten sind gestiegen, aber bereits vor 13 Jahren war dies ein sehr aktiver Forschungsbereich (Mündermann et al., 2006, S. 3–4). Die Entwicklung leistbarer IMC Anzüge hat seit der Jahrtausendwende besonders an Fahrt aufgenommen.

Preisentwicklung: Drei relevante Anbieter von kommerziellen Anzügen mit Beschleunigungssensoren sind Rokoko, XSENS, und Noitom. XSENS wurde 2000 gegründet und bietet neben Anzügen für Motion Capture auch einzelne Sensoreinheiten für verschiedene Einsatzzwecke an an. Der XSENS MVN Anzug wird je nach Anfrage bepreist, bewegt sich jedoch um die $ 12.000,- (Barsegyan, 2017). Der Rokoko Smarsuit kostet aktuell $2,495 (Rokoko, 2019). Noitom hat mit einer Kickstarter Kampagne 2014 ihre Lösung - den Perception Neuron Anzug - vorgestellt (Perception Neuron, 2014). Dieser kommt auf einen Preis von $ 1799, was ihn ebenso wie den Rokoko auch für sehr kleine Produktionshäuser attraktiv macht (Noitom, 2019).

3.2.1 Sensortechnik

Kontextunabhängigkeit: Der Unterschied zwischen sensorbasiertem und optischem Bewegungserfassen liegt darin, dass keine Sichtverbindung zum zu erfassenden Subjekt bestehen muss. Verdeckungen sind daher kein Problem. Theoretisch lässt sich mit einem Sensoranzug kontextunabhängig, beispielsweise auf einer Skipiste, Motion Capture umsetzen. In der Praxis zeigen sich aber besonders was die Position im Raum betrifft Herausforderungen, welche teils mit zusätzlichen Sensoren gelöst werden müssen (Brodie et al., 2008).

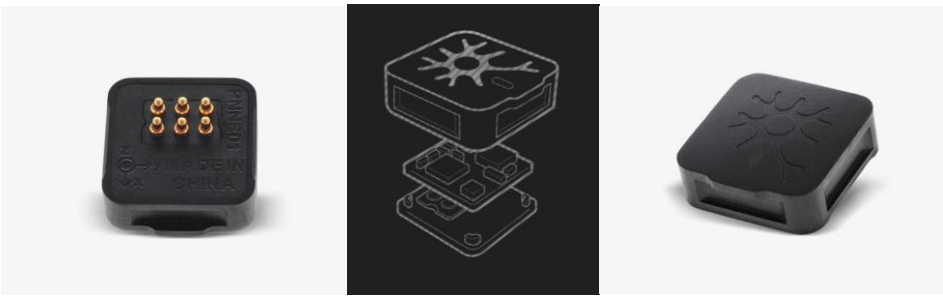


Abbildung 9: Einzelner Perception Neuron Sensor (Noitom, 2019)

Beschleunigung und Trägheit: Wie schnell eine Bewegung beginnt oder langsamer wird, kann durch Beschleunigung beschrieben werden. Dies ist eine der drei zentralen Funktionen einer Messeinheit, oder auch Inertial Measurement Unit (IMU) genannt.

Lage: Der zweite gemessene Wert beschreibt die Lage im dreidimensionalen Raum mittels eines Gyroskops/Lagesensors.

Magnetreferenz: Um die Bewegung durch den Raum und die Erfassung der Ausrichtung der Sensoren zu verbessern kommt ein Magnetometer in der einzelnen IMU zum Einsatz.

Virtuelle Realität: Dieselbe Sensortechnik ist auch in modernen Smartphones zu finden und wird auch hier bereits für einfachere Bewegungserfassung verwendet. Beispielsweise in der Brille Samsung Gear VR wird der Lage- und Beschleunigungssensor des Smartphones genutzt, um sich in der virtuellen Welt umzublicken.

Neuralgische Punkte: Je nach Anzug wird an den wichtigsten Gelenken und Punkten am Körper ein Sensor angebracht. Einem digitalen Skelett zugewiesen, können die Werte dieser Sensoren dann geschickt in der Software kombiniert werden, um eine Aufzeichnung der menschlichen Bewegung zu ermöglichen.

Abbildung 10: Am Körper platzierte IMU Sensoren anhand einer Werbung für die Perception Neuron und Perception Neuron Studio Anzüge (Noitom, 2019)

Magnetfelder: Neben diesen Messeinheiten gibt es jedoch auch weitere Sensoren, welche für Motion Capture einzeln oder als Hybridlösung zum Einsatz kommen können. Magnetsensorlösungen für Bewegungserfassung entstanden aus den überlagerten Zielerfassungsdisplays in den Helmen von Kampfpiloten des Militärs. Sie bestehen aus einer Sendeanlage und den am Subjekt angebrachten einzelnen Sensoren, welche direkt in ein Koordinatensystem übertragbare Daten liefern. Für die erzeugten Magnetfelder kommt entweder Gleich- oder Wechselstrom zum Einsatz. Jede dieser Lösungen hat andere Probleme mit nahen Metallen oder elektrischen Geräten, welche ebenfalls ein Magnetfeld produzieren. Auch die Transportabilität ist ein kritischer Faktor, da die Sensoren nicht mit starken Magnetfeldern in Kontakt kommen dürfen (Menache, 2011, S. 26).

Abbildung 11: Head Mounted Display im BAE Striker II Kampfpilotenhelm (Smith, 2017)

Exoskelette: Sensoren an den wichtigsten Gelenken waren eine bereits in den 1980er Jahren zum Einsatz kommende Lösung für Motion Capture mit Messungen direkt am Körper. In frühen Iterationen lieferten optische Potentiometer die Daten, aus welchen die Gelenkspositionen berechnet werden konnten. Diese waren, obwohl störanfällig, die Basis für den ersten digital erstellten Charakter, welcher mittels Motion Capture im Film "Toys" zum Leben erweckt wurde (Menache, 2011, S. 10).

Abbildung 12: Exoskelett wie es 1988 für den Film "Toys" zum Einsatz kam (Pacific Data Images; Heute DreamWorks)

Akustische Radio- und Hochfrequenzsensoren: Das Globale PositionierungsSystem GPS ermöglicht es durch die Berechnung der Laufzeitunterschiede zwischen den von den Satelliten ausgesandten Signalen beim Empfänger die Position auf der Erde zu berechnen. Für Motion Capture rein auf GPS Basis ist dieses System jedoch zu ungenau. Eine Alternative könnte ein Lokales PositionierungsSystem – kurz LPS sein, wie es von Alberto Menache entwickelt wurde. Hier wird die Technologie umgedreht. In einem klar definierten, aber willkürlich skalierbaren Raum sind Antennen mit bekannter Position verteilt. Die Signale, welche von den am Körper getragenen Sensortags ausgesendet werden, empfangen diese Antennen. Laufzeitunterschiede und Signalcharakteristiken liefern dann die genaue Position im Raum. Besonders für eine akkurate Translation ist dieses System gut geeignet. Im Vergleich dazu stellt Translation für inertial Motion Capture Anzüge eine große Herausfoderung dar. Brodie et al. schaffen es so beispielsweise mit der Kombination aus Lage-/ Trägheitssensoren und einer GPS Ortung eine zumindest für die Sportler und Trainer relevante Bewegungserfassung durchzuführen.

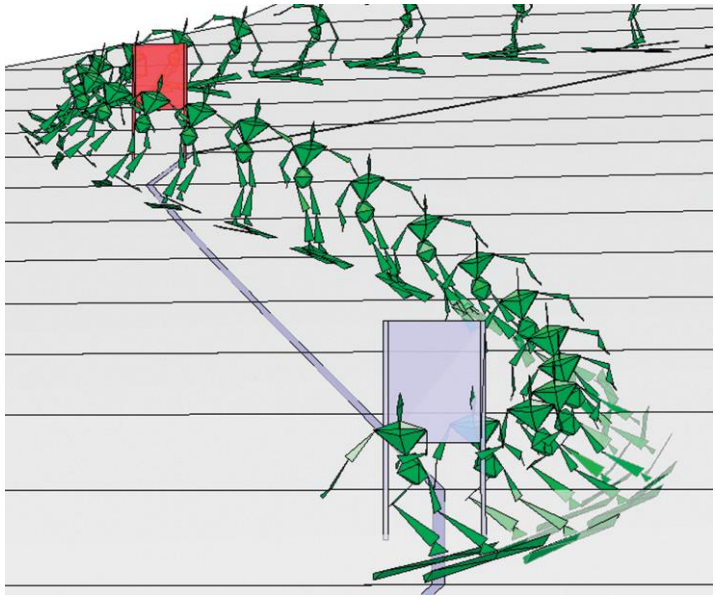


Abbildung 13: Visualisierung der mit GPS kombinierten IMU Bewegungsdaten eines Skirennläufers des neuseeländischen Nationalteams (Brodie et al., 2008, S. 24)

3.2.2 Perception Neuron

Vergleich mit optischer Lösung: Was Perception Neuron auszeichnet, ist in erster Linie der günstige Anschaffungspreis. Er besteht wie Mitbewerber auch aus IMU Sensoren mit Gyroskop, Beschleuingungssensor und Magnetometer. Im direkten Vergleich mit optischen Lösungen können beschleunigungssensorbasierte Anzüge in einigen Szenarien ähnliche Qualität liefern wie die teuren Mitbewerber. Besonders statische Momente, aber auch gewünschte Rutschbewegungen der SchauspielerInnen können jedoch ein Problem darstellen, weil einzelne Sensoren bzw. Gelenke manchmal unerwartet zu driften beginnen (Fleron et al., 2019, S. 374). Meine eigenen Ergebnisse bei einem Test der Kontextunabhängigkeit des Systems decken sich weitgehend mit der Analyse von Fleron et al. Diese Erfahrungen beschreibe ich im folgenden Kapitel.

3.2.3 Test von Perception Neuron

Motivation: Für ein Projekt mit gemischter Realität, wo ein realer Mensch mit digitalen, cartoonähnlichen 3D Charakteren interagieren soll, wurde ich beauftragt, eine Lösung für günstiges Motion Capture zu erarbeiten. Meine Wahl fiel auf den Anzug Perception Neuron 32 von Noitom mit Fingertracking. Dieser bot sowohl preislich als auch von zu erwartenden Ergebnissen ein vielversprechendes Gesamtpaket. Ohne Kameras auszukommen und die

Sensoren am Körper, theoretisch sogar unter Regenkleidung tragen zu können, lediglich von einer handlichen, mobilen Powerbank mit Strom versorgt, scheint einen sehr weiten kontextunabhängigen Einsatzbereich zu eröffnen.

Beurteilungskriterien: Für das von mir angestrebte Projekt waren besonders relevante Aspekte des Tests: die Anforderungen und einzuplanenden Zeiträume an die Aufzeichnung, Aufbereitung, sowie Integration der gewonnenen Daten. Der Workflow vom Auswählen der spielbaren Aktionen über das Anlegen und Kalibrieren der Sensoren bis zur Auswertung der Daten und Anwendung auf einen 3D Charakter sollte getestet werden. Auf dieser Basis, so mein Ziel, sollte sich eine qualitativ hochwertige subjektive Einschätzung des Aufwands und der Praktikabilität des Anzugs Perception Neuron tätigen lassen. Genauigkeit bei dennoch einfacher Handhabung und großem Kontext- sowie Bewegungsspielraum würden das ideale Ergebnis darstellen. Des Weiteren war es mir wichtig, die Interaktionsfähigkeit mit Objekten auf verschiedenen Höhen zu testen. Ein Beispiel für solch ein Hindernis sind Treppen. Da auch Überkopfbewegungen für die Bewegungserfassung in Frage kommen, waren auch diesen Teil meines Tests.

Innentest: Bei meinen vier Testläufen mit dem Anzug trat leider nie der Idealfall ein. Während meines ersten Tests in einem Büroraum stieß ich auf das Problem, dass nicht alle Sensoren erkannt wurden bzw. dass einzelne Sensoren als entmagnetisiert angezeigt wurden. Mittels der Software "Neuron Doctor", war es mir jedoch möglich, diese Sensoren erneut zu kalibrieren, wodurch sie auch wiedererkannt wurden. Die Quelle-Ziel Kalibrierung mittels der von der Software Axis Neuron vorgeschlagenen Posen funktionierte gut. Leider begannen die Sensoren, besonders auf den Händen und Fingern, nach einigen Sekunden oder ausladenden Bewegungen ungenau zu werden. Dies führte zu unbrauchbarem Fingertracking nach etwa 30 Sekunden Aufzeichnungszeit. Ein weiteres Problem stellte die Stromversorgung dar. Die Verbindung über ein USB Kabel direkt zum PC funktionierte problemlos, war jedoch nicht praktikabel für komplexere Bewegungen durch die Kabelbindung. Daher verwendete ich eine kleine Powerbank mit 5000 mah. Diese wird ebenso mit einem mini USB Kabel mit dem Hub des Anzugs verbunden. Da der MiniUSB Port jedoch keinen Verschlussmechanismus aufweist, löst sich die Verbindung bei ausladenden oder abrupten Bewegungen manchmal, was zu einem Ausfall des Systems führt. Je akrobatischer die Bewegungen wurden, desto weniger funktionierte der Anzug. Ein Rutschen über den Boden ist nicht möglich ohne nachträgliches Bearbeiten der Daten. Ebenso sind Bewegungen auf dem Boden nicht gleichwertig aufzuzeichnen wie Bewegungen im Stand. Traditionelle Tanzbewegungen ohne Ebenenwechsel funktionierten jedoch mit Einschränkungen beim Fingertracking gut.

Außentest: Bei meinem zweiten Test wollte ich die Limits der Kontextunabhängigkeit sowie die Brauchbarkeit für Bewegungserfassung von vielfältigen Sportarten wie Parkour und Freerunning testen. Dazu wählte ich als Testort den Freerunningpark St. Pölten in der Bimbo-Binder-Promenade. Der Test sollte zeigen, ob und mit welchem Aufwand verbunden ein Auf- und Abstieg über verschiedene Hindernisse realisierbar ist. Mit den bereits im Innenraumtest erfahrenen Problemen konfrontiert gelang es dennoch eine kurze Sequenz aufzuzeichnen, welche auch brauchbare Daten lieferte. Den Höhenunterschied in die Bewegungsdaten einzuarbeiten gelang jedoch nicht direkt in der Software Axis Neuron. Durch manuelle Animation der x und y Koordinaten eines gerenderten Testcharakters mit den bereits angewandten Bewegungsdaten innerhalb der Software Adobe After Effects konnte ich dennoch den originalen Bewegungspfad über die Hindernisse nachbilden.

Fazit: Ziel dieser Tests war es, die Machbarkeit von Low Budget Motion Capture in Hinblick auch auf kommerzielle Projekte zu erarbeiten. Besonders das unzureichende Fingertracking und die Gefahr von spontan entmagnetisierten Sensoren sind ausschlaggebend, dass ich den Perception Neuron Anzug für den zeitlich möglichst knapp planbaren Produktionsalltag mit komplexen Bewegungen nicht empfehlen kann. Einfache Bewegungen, die nicht in Kombination mit klassischen Schauspielern auf einem Set aufzuzeichnen sind, können jedoch auch mit dem Perception Neuron und genügend Spielraum in der Nachbearbeitung qualitativ ansprechend umgesetzt werden. Für den konkreten Anwendungsfall sollte dennoch unbedingt ein Test vorab erfolgen. Ein weiteres, von mir nicht erarbeitetes Thema in den Tests, ist die simultane Erfassung der Gesichtsbewegungen der motion capture SchauspielerInnen.

Abbildung 14: Perception Neuron Test im AFF Freerunningpark St. Pölten

4 Motion capture von planem Videomaterial

Traditionell wird ein Video zweidimensional rezipiert. Dadurch ist keine Tiefeninformation enthalten, welche die Rezipienten erfahren können. Das tut dem filmischen Erlebnis keinen Abbruch, jedoch ist es ein ständig aktuelles Thema, wie stark physische Immersion durch z.B. stereoskopische Video- und Filmproduktionen das Erlebnis des Publikums bereichert.

Nun besteht bereits eine gigantische zweidimensionale Videosammlung, welche auch für Animatoren von großem Nutzen ist. Referenzen für die händische Keyframe Animation oder auch für andere Aspekte der Bewegtbildproduktion, wie z.B. Charakterdesign können durch den Schatz an digitalen zweidimensionalen Daten unterstützt werden.

4.1 Zunehmende Relevanz

Der Gedanke, die Bewegung von auf Film und digitales Video gebannten SchauspielerInnen zu nutzen, ist also kein neuer. Seit der Erfindung des Motion Capture und der 3D Computeranimation gewinnen 2D Bewegungsdaten jedoch eine neue Dimension der Relevanz. War ein einzelnes Video einer Bewegung früher als Referenz nützlich, so können heute bereits die dreidimensionalen Daten extrahiert und weiterverarbeitet werden (Cao et al., 2018; Güler et al., 2018).

Maschinelles Lernen bietet mit Algorithmen und Erkenntnissen aus dem Feld des Computer Sehens (englisch. Computer Vision) eine Basis, auf welcher rasante Fortschritte in der Extraktion von 3D Bewegungsdaten aus monoskopischen ein-Kamera 2D Ausgangsvideos zu verzeichnen sind. Besonders starke Verbreitung finden die Lösungen OpenPose sowie DensePose. Diese unterscheiden sich auf den ersten Blick darin, dass OpenPose die Bewegungen als buntes Skelett darstellt, wohingegen DensePose den Menschen auch samt seines Volumens und Umrisses als Netz repräsentiert. Hier Grafik einfügen.

5 Bewegungstransfer mittels maschinell lernender Software

5.1 Künstliche Intelligenz und maschinelles Lernen

Buzz: Künstliche Intelligenz (KI) hat sich über die letzten Jahre durch großes Interesse und unscharfen Sprachgebrauch als Modewort etabliert. Produkten und Dienstleistungen aller Art werden durch das Kürzel KI meistens zu Verkaufszwecken Eigenschaften zugesprochen, welche fernab von intelligentem Agieren liegen. Sogar bei Paradebeispielen wie selbstfahrenden Autos kann noch nicht von künstlicher Intelligenz im ursprünglichen Sinn gesprochen werden. Hier ist keine Intelligenz vorhanden, wie sie Menschen zugeschrieben wird. Es handelt sich zum aktuellen Zeitpunkt fast immer um eine Lösung, die in einem klar abgegrenzten Raum der mathematischen Möglichkeiten durch unzählige Rechenschritte mittels großem Dateninput einen Fehler zu minimieren oder eine Zielfunktion zu optimieren sucht. Deshalb wird innerhalb des Fachgebiets vorsichtiger mit dem Begriff "Intelligenz" umgegangen.

Status Quo: So kann beispielsweise unterschieden werden zwischen schmaler/schwacher und breiter/genereller künstlicher Intelligenz. Die zweite Form wäre durch ihre Fähigkeit zu abstahieren und gelernte Information auf neue Problemstellungen anzuwenden zumindest multi-purpose, sprich für mehr als eine Aufgabenstellung nützlich. Der vermeintlich letzte große Sprung hin zur menschlichen Intelligenz wäre die generelle künstliche Intelligenz. Wenn dieses Ziel erreicht wird, kann davon ausgegangen werden, dass es eine Explosion an Anwendungen für KI geben wird, da es nicht mehr notwendig ist, für jede Fragestellung eigens eine Softwarelösung zu entwickeln. Der aktuelle Stand ist jedoch noch einige Meilensteine von genereller KI entfernt. Auch die Lösung zum Bewegungstransfer, welche in dieser Arbeit genauer betrachtet wird, erfüllt keine Kriterien menschlicher Intelligenz.

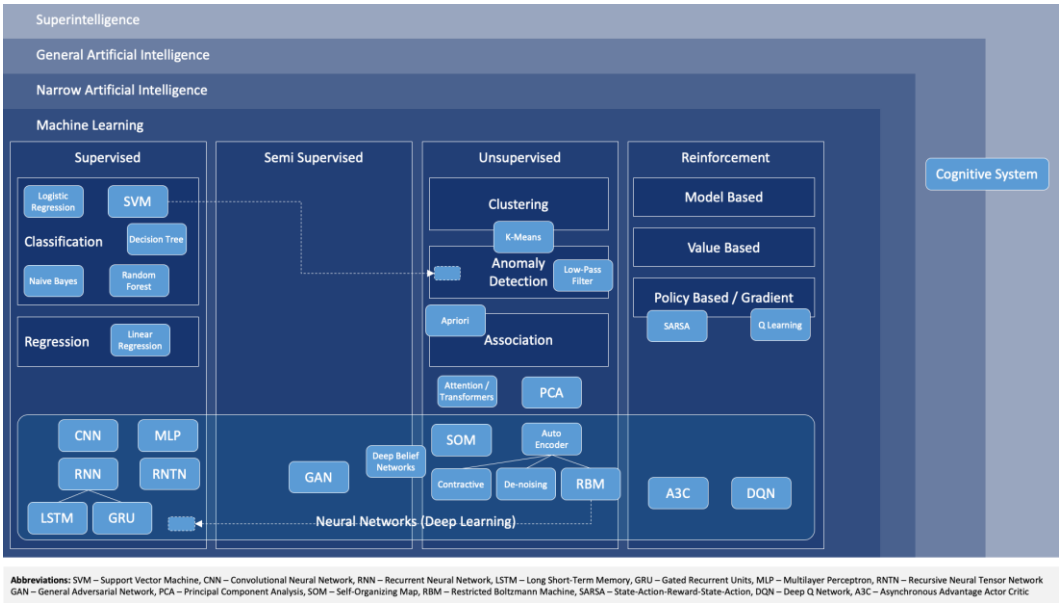


Abbildung 15: Überblick über maschinelles Lernen und übergeordnete Intelligenzebenen (Ackermann, 2018)

Maschinelles Lernen: In der obenstehenden Grafik wird eine Differenzierung der Potenz von künstlichen Intelligenzen und Lösungenskompetenz sichtbar. Wie eine Maschine praktisch lernen kann, ist dabei in beaufsichtigt (Supervised), halb-beaufsichtigt (Semi Supervised), unbeaufsichtigt (Unsupervised) und Verstärkungslernen (Reinforcement) eingeteilt. Die Softwarearchitektur, durch welche es möglich ist, Bewegung von einer Person auf eine andere zu übertragen, das Generative Adversarial Network (GAN), findet sich im Bereich des halb-beaufsichtigten Lernens. Zu dieser Lösung folgt später mehr.

5.2 Neuronale Netze

Definition: Neuronale Netze sind biologisch inspiriert, aber nicht eins zu eins vergleichbar mit dem Nervensystem wie es in biologischen Lebewesen zu finden ist. Sie sind vielmehr mathematische Funktionen mit als Neuronen bezeichenbaren Einheiten. Diese als Neuronen bezeichneten Knotenpunkte speichern Werte über ihre Aktivierung. Durch ihre Verbindung zu weiteren Einheiten können sie als Netzwerk Datensätze und darin enthaltene Muster lernen. Im Idealfall bildet ein neuronales Netz bedeutungsvolle Information ab, welche eine gewisse Allgemeingültigkeit aufweist und dadurch auf neue Problemstellungen derselben Art anwendbar ist. Man spricht dann davon, dass

das Netzwerk gut generalisiert.

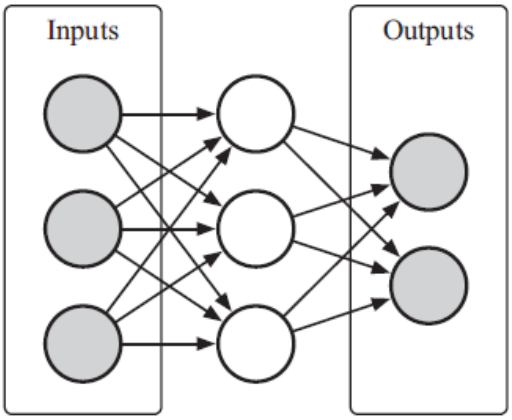


Abbildung 16: Einfaches neuronales Netzwerk mit drei Eingabeneuronen, drei Neuronen in der voll verbundenen ersten Ebene und zwei Ausgabeneuronen (Gerrish, 2018, S. 109)

Strukturelemente: Ein neuronales Netzwerk besteht aus einzelnen Knotenpunkten/Neuronen, welche in Ebenen geordnet sind. Die erste Ebene entspricht dabei den Eingabeneuronen. Bei Bilderkennung kann bspw. jeder Pixelwert eines Bilds ein Eingabeneuron darstellen. Die letze Ebene besteht bei vielen Problemstellungen aus wesentlich weniger Neuronen, welche bei der Aufgabe einer Klassifikation bspw. je eine Kategorie, wie Katze oder Hund, mit einem dazugehörigen Wahrscheinlichkeitswert ausgeben. Jedes Neuron ist bei der häufigen Form des voll verbundenen Netzwerks mit sämtlichen Neuronen der nächsten und der vorherigen Ebene verbunden. Diese Verbindung ist jedoch nicht zu allen Neuronen gleich relevant, sie ist gewichtet. Diese Gewichte bewerten die Verbindung durch ihren Einfluss auf die mathematische Relevanz eines Eingabeneurons für das jeweilige Zielneuron. Jedes Zielneuron ist zusätzlich mit einem Beeinflussungswert ausgestattet, welcher die Aktivierung/Bedeutung des Zielneurons direkt beeinflusst. Diese Verbindungsbewertungsfaktoren und Beeinflussungswerte sind die essenziellen Bestandteile für den Vorgang des maschinellen Lernens, da sie die gesamte Struktur durch die Güte der Verbindungen zwischen den Ebenen und einzelnen Neuronen des Netzwerkes ausbilden. Im Englischen werden hierfür die Begriffe weight (Verbindingsbewertungsfaktor) und bias (Beeinflussungswert) gebraucht, welche sich auch häufig in der Literatur finden. Wie konkret diese Struktur mathematisch ausgebildet wird entspricht den einzelnen maschinellen Lernalgorithmen wie beispielsweise gradientenbasiertem Abstieg (engl. Gradient descent) oder Fehlerrückführung (engl. Backpropagation (of errors)).

Aktivierungsfunktion: Jede gewichtete Verbindung zu den Neuronen der vorangegangenen Ebene in Kombination mit dem Beeinflussungswert des Zielneurons wird in einem finalen Aktivierungswert, welchen das Zielneuron in dieser Trainingsrunde speichert, zusammengefasst. Dafür werden die mit den dazugehörigenn Verbindungsgewichten multiplizierten Eingangsneuronen addiert. Aus dieser Zahl wird mit dem Beeinflussungswert eine weitere Summe gebildet, welche abschließend durch eine Aktivierungsfunktion in einen gewünschten Wertebereich verlagert werden kann. Für diese Aktivierungsfunktion, die meist einen kontinuierlicher Wert zwischen 0 und 1 als Ausgabeziel hat, gibt es verschiedene Ansätze diese mathematisch zu konzeptionalisieren. Zwei Überlegungen, wie diese Aktivierungsfunktion sinnvoll gestaltet werden kann sind, dass auf der einen Seite ein biologisches Neuron eher mit aktiv und inaktiv zu beschreiben ist (0 und 1 binär), wohingegen für gradientenbasiertes Lernen ein kontinuierlicher Fließkommawert zwischen 0 und 1 von Nutzen ist, um die Richtung des schnellsten mathematischen Abstiegs zu errechnen (Sanderson, 2017).

Biologisch inspiriert: Ganz exakt müssten die vorhin bereits beschriebenen neuronalen Netze als künstliche neuronale Netze bezeichnet werden. Obwohl eine lockere Analogie besteht zwischen den biologischen neuronalen Strukturen und den Softwarenetzen hält diese einer genaueren Betrachtung nicht stand. So sind die Abläufe, wie das biologische und das digitale Netzwerk aktualisiert wird, grundverschieden. Des Weiteren werden in komplexe künstliche neuronale Netze statistische Methoden eingearbeitet, welche die Abläufe noch weiter von ihrem natürlichen biologischen Namensgeber entfernen (Wittek, 2014, S. 63).

Netzwerkzoo: So vielfältig wie die gelernten Strukturen durch die Gewichte und Beeinflussungswerte sind, so vielfältig kann auch die neuronale Architektur von den ProgrammierInnen gestaltet werden. Besonders seit dem Boom der tiefen neuronalen Netze, aber auch bereits davor wurde eine Unzahl an Kombinationen von neuronalen Ebenen und ihren Verbindungen und Aktualisierungsabläufen konzipiert. Einige Forscher sprechen hier sogar von einem neuronalen Netzwerkzoo (L. Brunton, 2019; Veen & Leijnen, 2016).

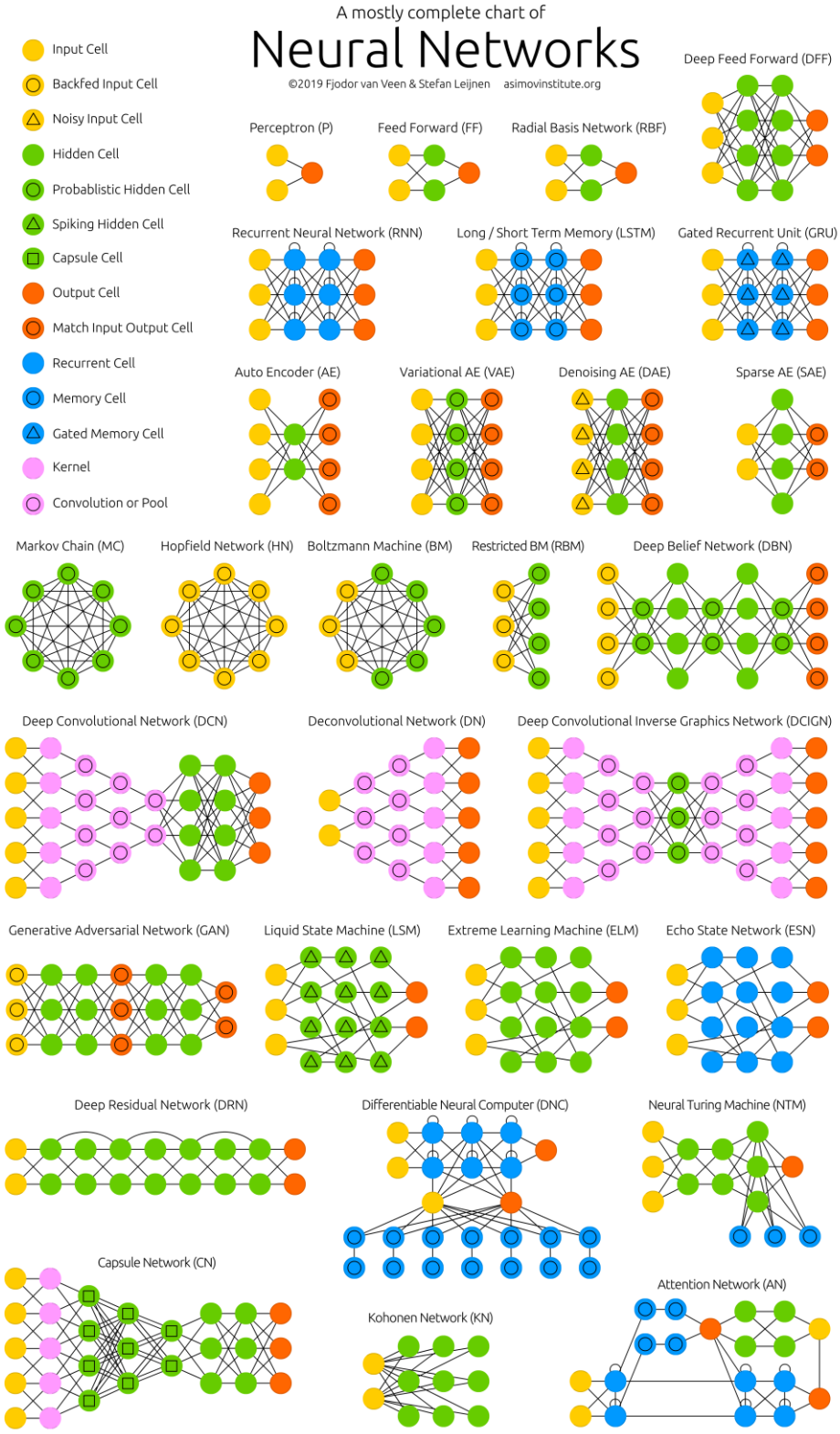


Abbildung 17: Eine vereinfachte Übersicht über die Vielfalt an modernen neuronalen Netzwerkarchitekturen ((Veen & Leijnen, 2016) aktualisiert 2019).

Steigende Komplexität: Die einfachste Struktur, bei der man von einem neuronalen Netzwerk spricht ist das Perceptron. Dieses kann jedoch nicht linear zu trennende Datensätze niemals lernen. Dieses Problem wird auch XOR Problem genannt (Wittek, 2014, S. 63) Erst durch die Kombination mehrerer Perceptrons in einem Hopfiel Netzwerk kann dieses gelöst werden. Im Hopfield Netzwerk können die Daten auch zurückfließend aktualisiert werden, wodurch das XOR Problem gelöst werden kann (Wittek, 2014, S. 65). Ausschließlich vorwärts aktualisierte neuronale Netze, wie auch in Abbildung 14 und 15 dargestellt, werden im Englischen Feedforward Networks genannt.

Tiefe Netze: Wenn nun in den Ebenen zwischen der Ein- und Ausgabeebene viele versteckte Zwischenebenen eingefügt werden, spricht man von einem tiefen neuronalen Netzwerk. Diese tiefen neuronalen Strukturen sind hierarchisch geordnet. Das bedeutet, dass am Ende des Datenflusses bei der Ausgabeebene ein abstrakteres Konzept von den Neuronen beinhaltet wird als von der vorangegangenen Ebene, und der Ebene davor, usw. Die Hoffnung ist, durch tiefes maschinelles Lernen "emerging properties", i.e. Eigenschaften, welche durch die tiefe Struktur bei dem „Lernen“ der Daten von selbst hervortreten, abbilden zu können. Dadurch kann bereits heute ohne viel Aufwand durch die ProgrammiererInnen bedeutungsvolle Information von Softwarealgorithmen gelernt werden. Welche Probleme sich besonders für tiefes maschinelles Lernen eignen, ist jedoch noch nicht gänzlich abgegrenzt. Was ein Netzwerk genau lernt, kann dabei häufig nicht sicher herausgefunden und analysiert werden.

Nutzen und Risiken: Das wissenschaftliche Feld ist in diesem Bereich des Lernens mittels tiefer neuronaler Netze besonders aktiv. Seit der Möglichkeit, die massiv parallelen Strukturen moderner Grafikprozessoren auszureizen, sind nicht nur das tiefe neuronale Netzwerklernen beflügelt, sondern auch andere Bereiche des maschinellen Lernens und der damit zusammenhängenden Fragestellungen. Tiefe neuronale Netzwerke bieten also die Möglichkeit, komplexe Zusammenhänge und Datensätze zu lernen und auf neue komplexe Eingabedatensätze, wie bspw. die Sprachverarbeitung der Stimme einer noch nie zuvor gehörten Person, brauchbar zu generalisieren. Jedoch ist das Risiko des Overfittings – der Überanpassung der gelernten Funktion – auf das Eingabedatenset besonders zu beachten, da auch mehr metaphorische Knöpfe und Drehregler bestehen, mittels welchen die Originaldaten leicht auswendig gelernt werden könnten. Dies gilt es durch intelligente mathematische Gegenmaßnahmen und strukturelle Überlegung zu verhindern Zitat (Wittek, 2014, S. 70).

Grafikkartenjackpot: "NVidia, [...], has been printing them like newspaper and selling them like hotcakes" (Gerrish, 2018, S. 144). Lineare Algebra und konkret Matrixoperationen sind zentral für die mathematischen Berechnungen, die ein lernendes modernes neuronales Netzwerk ausmachen. Bereits vor dem aktuellen Hype rund um maschinelles Lernen und künstliche Intelligenz war der Markt um Videospiele und Computerspiele heiß umkämpft für die Hersteller von Grafikprozessoren. Wer mit mehr Bildern pro Sekunde und in noch höherer Auflösung 3D Grafik darstellen kann, hatte und hat auch heute noch die Nase vorn am Gamingmarkt. Dass dieser Wettbewerb auch für tiefe neuronale Netze von Bedeutung ist, zeigte sich erst später. Ähnliche Operationen wie die rasche Berechnung qualitativer Videospielgrafik sind eben auch zentral für lernende neuronale Netzwerke. Verglichen mit einer modernen CPU (central processing unit) kann ein Grafikprozessor je nach Aufgabe den ohnehin zeitintensiven Lernprozess eines tiefen neuronalen Netzwerks zwischen 10 und 50 mal so schnell durchlaufen (Gerrish, 2018, S. 144)

Strukturerkennende Faltungscodierung: Zur Bilderkennung hat sich ein spezieller Typ des neuronalen Netzwerks als besonders geeignet herausgestellt - das Convolutional Neural Network (CNN). Ein – frei übersetzt – faltungscodiertes neuronales Netzwerk beinhaltet zwischen Eingabe- und Ausgabeebene eine oder mehrere versteckte Faltungsebenen. Diese sind am besten als Sammlungen von Filterneuronen zu beschreiben, welche nicht nur einen einzelnen Wert speichern, z.B. einen gewichteten Pixelwert, sondern das Skalarprodukt von dem z.B. 3x3 Pixel Filterneuron und 3x3 Pixeln aus der Eingabeebene plus Beeinflussungswert. Jedes Filterneuron hat dabei andere Filterwerte, welche in der Tiefe des Netzwerks hierarchisch an Abstraktion zunehmen. So kann am Beginn des Netzwerks die Erkennung von Kanten in einem Quadrat von 3x3 Pixeln stehen und am Ende das Konzept eines Katzenauges wie es in 3x3 Pixeln ebenso gefunden werden kann. Diese Mustererkennung macht CNNs besonders nützlich für die Bilderkennung. Ein zusätzlich optimierbarer und für die Funktion des Netzwerks relevanter Wert ist die Schrittgröße (engl. Stride) zum nächsten Block, den der Filter analysieren

soll. So kann ein z.B. 3x3 Filtergitter mit einer Schrittgröße von einem Pixel weiterbewegt werden, um den nächsten Block zu erfassen.

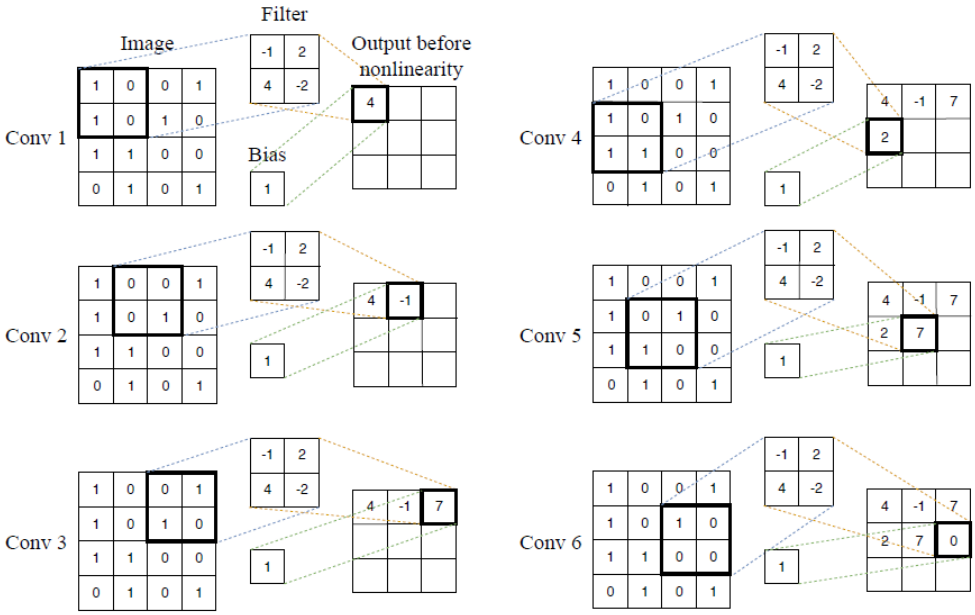


Abbildung 18: Veranschaulichung wie ein Filter der 2x2 Werte aufnimmt mit einer Schrittgröße von 1 von links nach rechts und oben nach unten über eine Tabelle von Eingabewerten eines Bildes bewegt wird (Burkov, 2019, Kapitel 6)

Gefaltene Neuronale Netzwerke zeichnen sich aus durch ihre Fähigkeit, Bilder akkurat zu erkennen und zu kategorisieren. Durch das erhöhte Forschungsinteresse, welches unter anderem auch durch Wettbewerbe wie die ImageNet Large Scale Visual Recognition Challenge – kurz ILSVRC, befeuert werden, und die Möglichkeit, viele Parameter an den Architekturen und den einzelnen Elementen sowie Abläufen der Netzwerke zu optimieren, entstanden in den letzten Jahren auch eine Vielzahl an verschiedenen Ansätzen. Bei der ILSVRC Challenge wird die Leistung der neuronalen Netzwerke u.a. beim Klassifizieren von Bildinhalten gemessen.

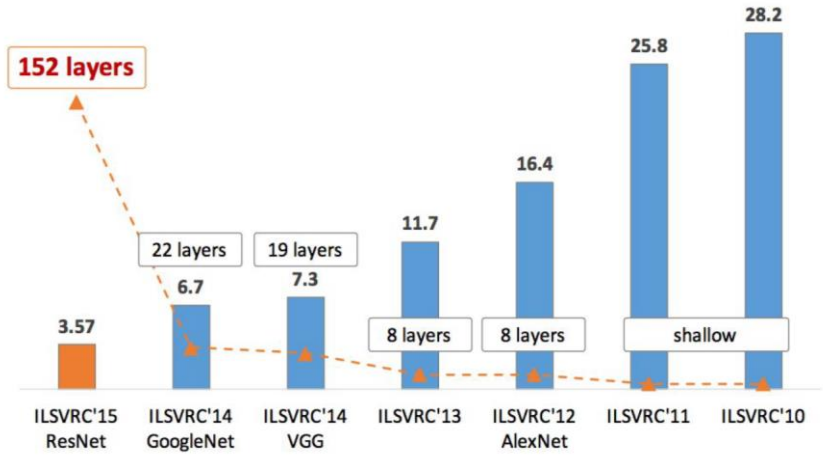


Abbildung 19: In dieser Grafik ist die abnehmende Fehlerrate und die zunehmende Tiefe der neuronalen Netzwerke der Gewinnerteams im Verlauf der ILSVRC von rechts, 2010, nach links, 2015, zu beachten. Das erste tiefe faltungscodierte neuronale Netz ist hier AlexNet, 2012. (Li et al., 2017; Russakovsky et al., 2015)

VGG: Für den in dieser Diplomarbeit im praktischen Teil zitierten Code kommt das VGG Netzwerk, welches 2014 in der ILSVRC erfolgreich war, zum Einsatz.

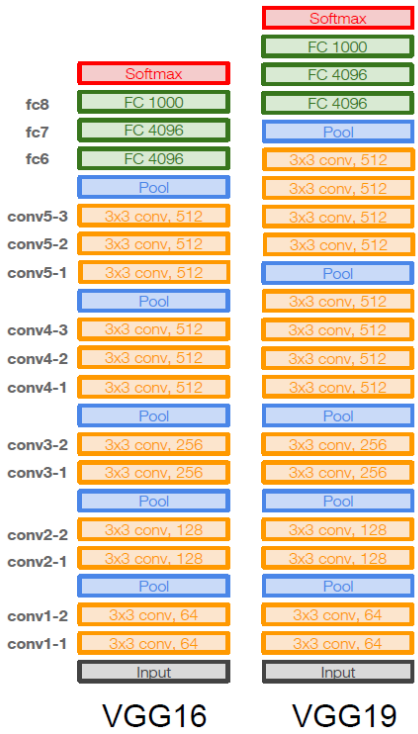


Abbildung 20: Schematisches Flussdiagramm wie die zu analysierenden Daten von unten nach oben durch die Ebenen des VGG16, sowie des VGG19 Netzwerks erfasst, analysiert, und weitergereicht werden. (Li et al., 2017)

5.3 Betriebssystem Ubuntu für Machine Learning

Linux, Windows oder Mac OS? Diese Frage stellt sich nicht nur den Nutzern von maschinell lernender Software. Windows und Mac OS sind die am weitesten verbreiteten Betriebssysteme für Laptops und Standcomputer. Dadurch gibt es auch die meisten Nutzer sowie das größte Potential, Software an diese zu verkaufen.

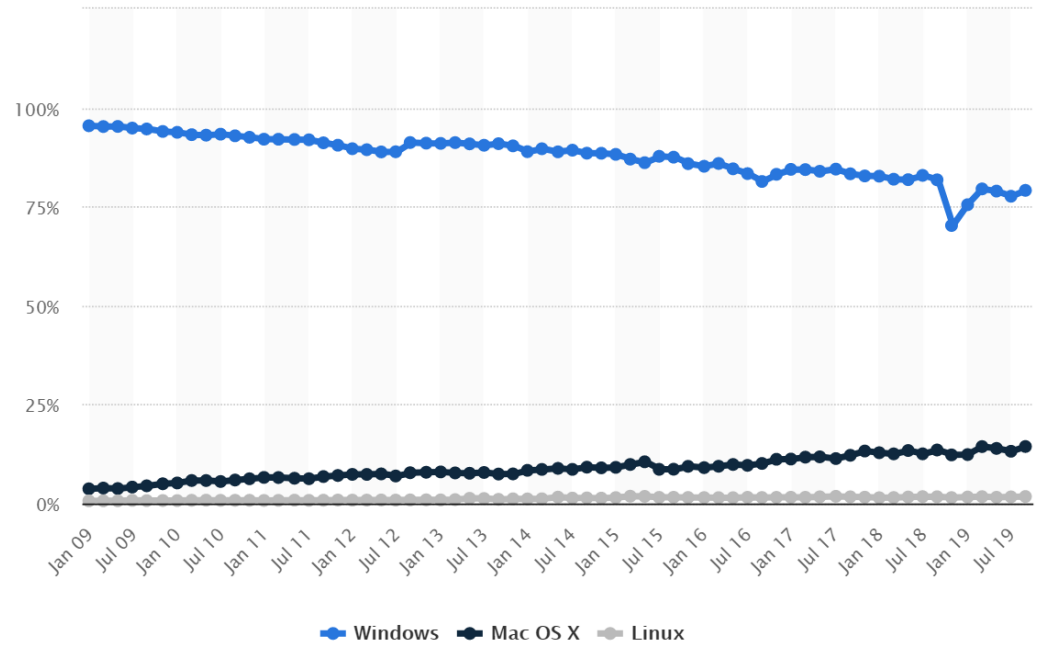


Abbildung 21: Marktanteile der führenden Betriebssysteme weltweit; Oktober 2019: Windows 79,1%, Mac OS X 14,37%, Linux 1,74% (Statista, 2019)

IT Sicherheit und der Markt: Aber auch jene mit üblen Absichten versprechen sich von Windows durch die hohe Marktdurchdringung auch den größten Profit bei der Verbreitung bösartiger Software, wie z.B. Viren. Unter anderem deswegen sind Linux Distributionen für den Betrieb von sicherheitsrelevanten Anwendungen häufig die erste Wahl.

Distribution: Linux Distributionen wie Ubuntu 18.04 LTS bieten eine mächtige Umgebung, um auf grundlegendere Programmfunktionalität, beispielsweise über das Terminal, zuzugreifen. Ganze Programme, einzelne Programmteile oder auch grundlegende Betriebssystemkomponenten können bei auf Debian basierenden Distribution wie Ubuntu 18.04 LTS einfach als Administrator bzw. Mitglied der sudo Benutzergruppe über Programme wie apt-get

(Application Package tool) oder dem Python spezifischen Package Installations Programm (PIP bzw PIP3 install) über das Terminal installiert werden. Viele ForscherInnen im Feld des maschinellen Lernens, aber auch SoftwareentwicklerInnen anderer Bereiche verwenden Linux und dadurch findet sich auch häufig viel Information über Fehlermeldungen spezifisch für eine Linux Distribution. Dies ist besonders hilfreich beim Entwickeln oder auch einfach beim Installieren von Software (Pardeshi, 2019).

Offener Code: Dass es mehr Information über mögliche digitale Stolpersteine gibt, liegt nicht zuletzt daran, dass Linux Open Source ist. Das bedeutet, dass der Code für das Betriebssystem frei einsehbar und zugänglich ist. Außerdem ist Linux gratis, was dazu beiträgt, dass die Entwicklungsmöglichkeiten von Software noch einer größeren Nutzerbasis zugänglich ist. Bekannte und beliebte Distributionen von Linux sind Ubuntu und Debian, wobei Ubuntu, wie viele weitere Distributionen, auf im Rahmen des Debian Projekts entwickelten Code basiert. Eine Distribution kann zusammengefasst beschrieben werden als Betriebssystem-Softwaresammlung, welche auf dem Linux Kernel aufsetzt. Für meine Tests habe ich Ubuntu 18.04 Long Term Support (LTS) gewählt. Es besteht eine kleine Lernkurve, wenn zum ersten Mal auf Linux gearbeitet wird. Jedoch war es mir relativ schnell möglich, mich zurechtzufinden; Nicht zuletzt wegen der von Ubuntu- zu Ubuntuversion immer einfacher gestalteten grafischen Benutzeroberfläche (GUI).

5.3.1 Python

Menschenlesbar: Python ist eine Programmiersprache, die für vielfältige Anwendungsbereiche geeignet ist. Sie ist eine sogenannte "High level" Sprache, die dazu gedacht ist von Menschen geschrieben und verstanden zu werden. Ein Kompilierer (Compiler) übersetzt den Code dann in für die Hardware des Rechners verständlichen Code. Eine vergleichbare ebenso verbreitete und bekannte High level Programmiersprache ist Java. Python ist plattformunabhängig und wird sowohl unter Windows, Linux als auch Mac OS unterstützt. Von Python finden sich zwei relevante Versionen, zwischen denen nicht einfach gewechselt werden kann – Version 2.x und Version 3.x. Daher ist es angebracht, vorab eine Entscheidung zu treffen, welche Abhängigkeiten für die entwickelte Software vermutlich gegeben sein werden und inwiefern diese von der gewählten Python Version Unterstützung finden. Die aktuelle Python Version ist 3.8.0 (Dezember 2019).

Warum Python: Wie Andriy Burkov in seinem Buch "The Hundred-Page Machine Learning Book" behauptet, ist es schwierig, sich einen Data-Scientist vorzustellen, der nicht Python verwendet (Burkov, 2019, S. 6). Ein großer Teil

aktueller Forschung im Bereich des maschinellen Lernens wird in der Programmiersprache Python geschrieben. Der Grund, dass sich diese Programmiersprache häufig durchsetzt, ist nicht eindeutig festzumachen. Jedoch wird die Sprache oftmals als intuitiv, leicht lesbar und und gut dokumentiert bezeichnet (Trask, 2019, S. 10). Die große Bereitschaft der Community, ihr Wissen oft erstaunlich gut aufbereitet zu teilen ist ein weiterer Grund, der dafür spricht, das nächste Projekt in Python zu schreiben.

5.3.2 Pytorch und Tensorflow

Lern-Frameworks: Wie kommt man mit maschinellem Lernen an sein Ziel? Diese Frage führt oftmals zu der Antwort: Mit einem Framework wie Tensorflow (Google), (Py)Torch (Facebook), Azure (Microsoft) oder Keras. Die Anwendung der Frameworks ist vergleichbar mit dem Fahren eines Autos, um von A nach B zu gelangen. Die Theorie wie das Auto, bzw. das machine learning Framework funktioniert, ist hilfreich, aber nicht unbedingt notwendig, um ans Ziel zu gelangen (Trask, 2019, S. 6). Grob umrissen ermöglicht uns ein Framework für maschinelles Lernen, wie z.B. die Open Source Variante Tensorflow, multidimensionale Datenarrays bestehend aus Vektoren (daher **Tensor**flow) auf mehreren central processing units (CPU) oder graphical processing units (GPU) zu verarbeiten (*Foundations of trusted autonomy*, 2017, S. 234).

Unterschiede: Die zwei bekanntesten Frameworks für machine learning sind aktuell Tensorflow und PyTorch. Neben dem grundsätzlichen Unterschied, dass Tensorflow (basierend auf Theano) von Google und PyTorch (basierend auf Torch) von Facebook entwickelt wurden gibt es noch eine wichtige Differenz in der Funktionalität. Tensorflow bedient sich eines vorher komplett zu definierenden Rechengraphs, wohingegen PyTorch auf einen aktiven Rechengraphen setzt. Dadurch lässt sich in PyTorch auch während dem laufenden Programm in den Rechengraph dynamisch eingreifen (Yashwardhan, 2018).

Altersunterschied: Dennoch hat aktuell Tensorflow mehr Nutzer, wodurch auch die online frei verfügbare Wissensbasis der Community etwas größer ist als bei PyTorch. Dies liegt unter anderem auch daran, dass PyTorch ein Jahr jünger ist (2016) als Tensorflow (2015).

6 Everybody Dance Now – UC Berkeley labs

2D Bewegungstransfer: Diese Masterarbeit orientiert sich im praktischen Teil an einer Lösung zum Bewegungstransfer in einer 2D Domäne, wie sie von ForscherInnen an der University of California, Berkeley entwickelt und in dem Paper "Everybody Dance Now" vorgestellt wurde (Chan et al., 2018). Auf den nächsten Seiten beschäftigt sich dieses Kapitel mit einer Analyse dieser Methode. Einleitend geht es darum, womit sich dieses Paper konkret befasst. Gefolgt wird dies von einer Analyse der einzelnen Schritte vom Quellvideo zum Video der tanzenden Zielperson. Abschließend werde ich kurz die Hintergründe und vorangegangenen Arbeiten, auf die sich wiederum das Paper "Everybody Dance Now" stützt, beleuchten. Dadurch soll eine Basis für die im nächsten Kapitel folgende Beschreibung meiner Lösung geschaffen werden. Außerdem soll deutlich zum Ausdruck kommen, welche Teile der in „Everybody Dance Now“ vorgestellten Lösung von mir im praktischen Versuchsaufbau übernommen werden.

Ziel: Wie der Titel des Papers „Everybody Dance Now“ schon durch seine Referenz auf den gleichnamigen Popsong andeutet, ist das Ziel der Arbeit die glaubhafte Nachahmung von Tanzbewegungen eines auf Video festgehaltenen Tanzes in einem synthetisierten Video. In anderen Worten besteht das Ziel darin, die Bewegung einer Quellperson auf eine Zielperson zu übertragen – 2D Bewegungstransfer. Der Titel und auch die Beispiele des Papers beschränken sich dabei auf den Bereich des Tanzes, obwohl grundsätzlich auch andere Anwendungskontexte machbar sind (Chan et al., 2018, S. 2).

Abbildung 22: Videobeispiele der Bewegungstransfermethode von Everybody Dance Now auf Youtube (Klick auf das Bild führt zu: https://youtu.be/PCBTZh41Ris) (Chan, 2018)

Kurzbeschreibung des Systems: Das komplette System, wie es in „Everybody Dance Now“ vorgestellt wird, besteht aus einer Trainings- und einer Transferphase. In der Trainingsphase werden mehrere neuronale Netzwerke trainiert. Dabei ist das neuronale Netzwerk, mit dem die Position der Person extrahiert wird von der Softwarelösung OpenPose bereits fertig trainiert bereitgestellt, ebenso wie das VGG Netzwerk bereits zur Bilderkennung seine Gewichte und Aktivierungen für die neuronalen Netze mitbringt. Selbst trainiert werden von der Lösung von Chan et al. lediglich der Generator, welcher die Zielperson in neuen Posen erzeugt, und der Diskriminator, welcher die echte Zielperson von der synthetisierten zu unterscheiden vermag, sowie Generator und Diskriminator zur Verbesserung des synthetisierten Gesichts. Der ganze Ablauf des Trainings der eigenen neuronalen Netze beginnt mit der Vorbereitung, konkret mit der Extraktion der Gelenkskoordinaten aus dem Zielvideo durch die Posenerfassungssoftware OpenPose. Aus den so gewonnenen Positionsdaten wird anschließend ein Skelett als Repräsentation erstellt und als 2D Bild dem korrespondierenden realen Einzelbild zugeordnet. Das Verhältnis von horizontalen und vertikalen Pixeln, i.e. die Abmessungen des Skelettvideos, gleicht dabei denen des Ausgangsvideos. Daraus ergibt sich ein Bild-Skelettpaar. Dieses Paar ergibt eine Einheit als reale Basis (Grundwahrheit). Der Diskriminator wird trainiert, diese Bildpaare in seinem Netzwerk mathematisch abzubilden, um sie von generierten unterscheiden zu können. Der Generator bekommt die Aufgabe, anhand des Skeletts ein dazugehöriges Bild zu erzeugen. Dieses Bild wird dann mit dem echten, zu dem Skelett gehörenden, Bild verglichen. Dabei machen sich die ForscherInnen sowohl den im Prozess gleichzeitig lernenden Diskriminator als auch das auf allgemeine Fotoobjekterkennung vortrainierte VGG Netzwerk

zunutze, um einen mathematischen Unterschied zwischen generiertem und echtem Bild zu berechnen. Je geringer der Unterschied zwischen echt und synthetisiert, desto besser die Leistung des Generators (Chan et al., 2018, S. 3). Diese Struktur zum maschinellen Lernen mit tiefen neuronalen Netzen nennt sich GAN – Generative Adversarial Network. Hierzu folgt noch eine genauere Erklärung in diesem Kapitel.

Bewegungstransfer =/= Positionstransfer: Der Titel dieser Arbeit lautet jedoch Bewegungstransfer und nicht Positionstransfer. Bewegung als simple Abfolge von verschiedenen visuellen Positionen darzustellen ist naheliegend aber unterschlägt viele implizierte Informationen, wie beispielsweise über Naturgesetze, welche aus der Bewegung ableitbar sind. Vorrangig muss es also darum gehen, die Bewegung einer Person zu isolieren. Diese soll dann auf die Anatomie einer zweiten Person transferiert werden. Da der Möglichkeitsraum von machbaren Bewegungen zu gewissem Grad von der Anatomie einer Person und dem körperliche Fitnesslevel abhängig ist, kann eine vollstände Trennung von Bewegung und Anatomie nicht erfolgen. Bei zwei grundsätzlich ähnlichen Körpern ist jedoch ein zufriedenstellendes Ergebnis zu erreichen. Wo genau der Punkt liegt, an dem die Bewegung nicht mehr glaubwürdig für die Anatomie der Zielperson erscheint, ist schwer festzumachen. Ein Beispiel wäre die Übertragung der Bewegung einer kleinwüchsigen Quellperson auf einen Basketballspieler. Wenn Realismus jedoch nicht das angestrebte Ziel ist, wie auch im praktischen Teil dieser Arbeit beim Transfer auf Zeichentrickcharaktere, wird dadurch der Spielraum an transferierbaren Bewegungen vergrößert. An dieser Stelle sind künstlerisch kreative Auseinandersetzungen und Experimente durchaus möglich.

Ende zu Ende pixelbasiert: Die Lösung zur Generierung einer sich nach der Vorgabe der Quellperson bewegenden Person ist,, wie sie in "Everybody Dance Now" vorgestellt ist Ende zu Ende pixelbasiert. Sowohl das 2D Video der Quellperson besteht ebensowie das repräsentative Skelett des Zwischenschritts und auch das neu generierte Video der Zielperson lediglich aus digitalen Pixeln ohne zusätzliche Datenstrukturen. Viele Motion Capture Prozesse versuchen eine 3D Repräsentation der Bewegung zu erfassen, um auch für 3D Charakter die Bewegung möglichst direkt verwenden zu können. Diese Anforderungen an die 3D Information der Daten fallen bei dem hier beschriebenen Setup weg. Die digitalen Pixeldaten fließen wörtlich durch eine „end to end pixel-based pipeline" (Chan et al., 2018, S. 1).

Transferprozess: Der Transfer der Bewegung setzt ein zufriedenstellend abgeschlossenes Training der neuronalen Strukturen voraus. Dieser folgt dem Ablauf, wie er auch in der nächsten Abbildung ersichtlich ist. Zuerst wird aus dem

Quellvideo die Position der tanzenden Person erfasst, welche anschließend in die Repräsentation als Skelett übertragen wird. Das Skelett wird dann in seinem Verhältnis zur Kamera und mit der Größe der Zielperson verglichen im Schritt der Normalisierung. Dabei wird die Körpergröße und Sprunggelenksposition beider Personen und Skelette in ihren Extrempositionen erfasst um die Skelettpositionen des Quellskeletts anzugleichen. Hier kommt auch eine Einschränkung ins Spiel. Für kreative Kamerawinkel und ungewöhnliche Brennweiten reicht diese Normalisierung nicht aus (Chan et al., 2018, S. 7). Nach diesem Schritt kann die Skelettsequenz des Quellvideos an den Zielvideogenerator übergeben werden, welcher ohne großen Zeitaufwand nun das finale Video als Bildsequenz erstellt. Damit ist der Transferprozess der Bewegung abgeschlossen.

Aufbauende Abfolge: Der Transferprozess der Methode zum Bewegungstransfer von Berkeley Labs kann durch drei aufeinander aufbauende Phasen zusammengefasst werden (Chan et al., 2018, S. 2):

1. **Posenerfassung (Pose Detection):** Hier wird mittels des Bewegungserfassungsalgorithmus OpenPose ein Skelett aus dem 2D Videomaterial extrahiert. Diese Repräsentation wird sowohl für das Quell- als auch das Zielmaterial extrahiert. Die Bildgeneratoren und Diskriminatoren der später noch genauer beschriebenen neuronalen Netze werden jedoch nur für das Zielvideo „trainiert“.

2. **Globale Positionsnormalisierung (global pose normalization):** Im zweiten Schritt wird das Skelett der Tänzerin/des Tänzers, vorbereitet, um möglichst gleichwertig in Position und Größe im Bild das Skelett der Zielperson ersetzen zu können. Dadurch wird versucht, die Qualität der generierten Bilder in Schritt 3 zu verbessern.

3. **Mapping der normalisierten Position auf die Zielperson:** Zum Schluss generiert das auf die Zielperson trainierte Generatornetzwerk mit den ersetzten und normalisierten Skelettlabels des Tanzvideos eine Sequenz, in der sich die Zielperson so bewegt wie die Quellperson.

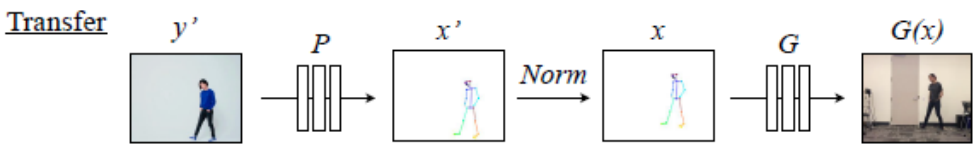


Abbildung 23: Der Transferablauf von Quellvideo zum generierten Zielvideo nach bereits abgeschlossenem Training von Generator G (Chan et al., 2018, S. 3)

6.1 Phase 1 – Posenerfassung

OpenPose: Als Posenerfassungslösung wird von Chan et al. die frei verfügbare (open-source) und dennoch hohen Ansprüchen gerecht werdende Software OpenPose verwendet. Der Begriff Pose beschreibt die anatomische Position, welche eine Person zu einem gegebenen Moment eingenommen hat. Diese gilt es zu erfassen und abzubilden. Posenerfassung (Pose Detection) ist verwandt mit dem Begriff Bewegungserfassung (Motion Capture). Jedoch beschränkt sich die Posenerfassung auf einen Moment wohingegen der Begriff Bewegungserfassung schon eine Abhängigkeit von der zeitlichen Dimension impliziert. Eine Aufgabenstellung ist jedoch für beide Bereiche gleichermaßen relevant – die zur möglichst repräsentativen Darstellung der Position einer Person. Der Körper verändert und verformt sich, um der Vielzahl an anatomisch möglichen Posen nachkommen zu können. Ohne ein genaues physikalisches Modell, das die Naturgesetzte korrekt abbildet, ist somit keine exakte Repräsentation aller Positionen möglich. Bei Säugetieren ist der passive Bewegungsapparat, das Skelett, die genaueste Möglichkeit, eine Position unabhängig von verformbarem Gewebe wie Fett oder Muskel zu beschreiben. Unter anderem deswegen wird in der Bewegungserfassungen das Skelett als Repräsentation gewählt.

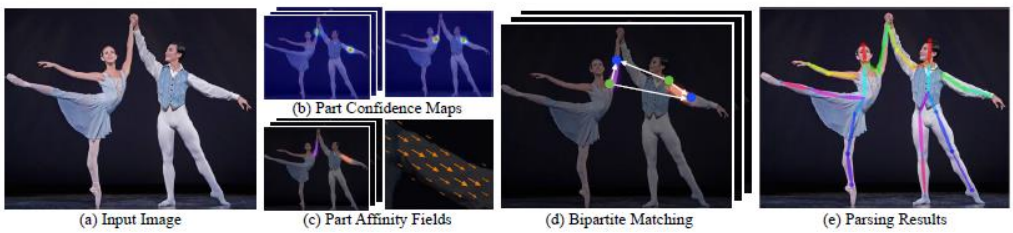


Abbildung 24: Ablauf der Posenerfassung durch OpenPose. Ein Einzelbild (a) stellt die Eingabe dar für ein gefaltenes neuronales Netz (CNN – convolutional neural network). Dieses berechnet anschließend (b) und (c) gemeinsam. Welche Körperteile zusammengehören wird im Schritt (d) erarbeitet. Danach kann das Ergebnis (e) gezeichnet werden.
(Cao et al., 2018, S. 2)

Abhängigkeit von der Zeit: Viele Posen vermag ein Mensch nur flüchtig einzunehmen. Welche Position einer dynamischen Körperhaltung wahrscheinlich vorausgeht und welche ihr folgt ist für den Betrachter relativ einfach abzuschätzen. Dieses Verständnis der realen Welt besteht für eine Software jedoch (noch) nicht. Bei einer Momentaufnahme, wie sie OpenPose betrachtet, entfällt daher die zeitliche Dimension. Klassische Bewegungsaufzeichnungen mit Trackingmarkern und/oder Sensoranzügen, wie Sie zu Beginn dieser Arbeit beschrieben werden, sind aktuell noch der Industriestandard. Durch die hochqualitative Aufzeichnung

von 3D Positionsdaten in hoher Frequenz ist die Qualität der Erfassung in der realen Welt aktuell noch im Vorteil. Im klassischen Motion Capture Kontext werden Skelette, die im Vorhinein definierte Proportionen aufweisen, durch den Input eines Darstellers manipuliert. Dies unterscheidet sich von dem Vorgang der bei jedem Frame neuen Extraktion eines Charakters, der nicht als konstantes vorab definiertes Modell, zumindest im klassischen Sinn der 3D Animationsdomäne, besteht. Die Positionsinformation muss daher nicht erst nachträglich extrahiert werden, wodurch selbst bei simplen Motion Capture Lösungen der Vorteil besteht, temporale Kohärenz von einem Moment zum nächsten, zumindest was die Körperform betrifft, sicherzustellen. Durch die bereits jetzt hohe Qualität moderner Posenerfassungssoftware wie OpenPose und die Forschung zu zeitlicher Kohärenz droht dieser Vorteil jedoch zu verschwinden. Die wesentlich günstigeren und unkomplizierteren Aufzeichnungsanforderungen der Softwarelösungen lassen bereits jetzt viele Produktionen zeitbasierter Medien umdenken. Zeitliche Information wird von OpenPose jedoch noch nicht mitgeliefert. Wie auch im abschließenden Segment des Papers „Everybody Dance Now“ angemerkt, wäre es jedoch wünschenswert, diese zeitlich kohärente und auch noch eindeutigere Repräsentation menschlicher Bewegung zu haben als die Gelenkskoordinaten und die Verbildlichung als Skelett, wie sie OpenPose liefert (Chan et al., 2018, S. 7). Da sich diese Technologie rasant entwickelt, gilt es jedoch festzuhalten, dass zu der Zeit, als das Paper „Everybody Dance Now“ veröffentlicht wurde OpenPose der State-of-the-art Bewegungserfassungsalgorithmus war. Jedoch ist auch bereits jetzt die alternative Verwendung von z.B. DensePose oder einer Kombination aus OpenPose und DensePose denkbar. Ein solcher kombinierter Ansatz zweier Architekturen ist im Paper „Video-to-Video Synthesis“ vorgestellt (T.-C. Wang et al., 2018a, S. 6).

Unabhängig von OpenPose nehmen sich Chan et al. in „Everybody Dance Now“, wie später beschrieben, der Problemstellung der zeitlichen Kohärenz in ihrem Setup zur Raum-Zeit Glättung an (Spatio-Temporal Smoothing).

Repräsentation als buntes Skelett: Wieso wird ein Skelett erstellt und nicht einfach der Positionswert der Gelenke als Koordinate herangezogen?

Abbildung 25: Skelett als Repräsentation der Pose (Chan et al., 2018, S. 2)

Die Softwarearchitektur OpenPose liefert Gelenkskoordinaten auf zwei Achsen (x,y). Diese Koordinaten dienen dann als Grundlage, um ein zweidimensionales buntes Skelett als Repräsentation der Position einer Person als Einzelbild zu erstellen. Wenn mit den Gelenkspositionsdaten nun nachvollziehbar gearbeitet wird, wie es beim Normalisierungsschritt der Fall ist, ist eine Überprüfbarkeit des Ergebnisses wünschenswert. Durch die bildliche Repräsentation in Form eines Strichmännchens/Skeletts, kann beispielsweise besser beurteilt werden, ob die globale Positionsverschiebung hin zu einer besseren Übereinstimmung mit der Position des Skeletts der Trainingspaare des Zielvideos erreicht wurde. Da jedes als Strich repräsentierte Körperteil eine andere Farbe hat, eignet sich dieses Skelett auch in seiner reduzierten Zweidimensionalität für das Training eines neuronalen Netzwerks. Die Repräsentation dient also sowohl dem "Lernen" der neuronalen Netze als auch der Überprüfbarkeit einzelner Zwischenschritte durch die Softwareentwickler.

Bewegungsdatensatz der Zielperson: Die Zielperson, auf welche die Bewegung der Quellperson übertragen werden soll, bekommt ihr eigens trainiertes neuronales Netzwerk, das nur darauf spezialisiert ist, diese Person möglichst realistisch zu generieren. Dies funktioniert dadurch, dass aus einem Video der Zielperson die einzelnen Videobilder extrahiert werden und zu jedem Videobild anschließend ein dazugehöriges Skelett abgespeichert wird. Ein Einzelbild und ein Strichmännchen/Skelett ergibt dann jeweils ein Paar. Um die Qualität der späteren Generierung positiv zu beeinflussen, ist es nützlich, die Zielperson in möglichst vielen verschiedenen Posen zu sehen. Es reicht jedoch bei der im Paper „Everybody Dance Now“ vorgestellten Lösung, zumindest für klassische Tanzbewegungen im Stehen, wenn die Zielperson sich in Ganzkörperansicht für wenige Minuten vor der Kamera bewegt, tanzt und sich auch idealerweise einmal dreht. Dadurch kann auch der Hinterkopf später glaubhaft synthetisiert werden (Chan et al., 2018, S. 1). Eine höhere Framerate bei der Aufzeichnung, wie z.B.

120 anstatt 25 Bilder pro Sekunde vergrößert das Datenset auch signifikant, wodurch ein besseres Endergebnis zu erwarten ist.

6.2 Phase 2 – Normalisierung

Skelettvergleich und Angleichung: Mit der vorgestellten Methode kann erst durch den Prozess der Normalisierung zwischen zwei verschiedenen Personen und Repräsentationsskeletten Bewegung übertragen werden. Bei der Normalisierung wird das Skelett aus dem Video der Bewegungsquelle an das Skelett der Zielperson angepasst. Die Methode von Chan et al. löst unabhängig vom Training, wie bereits beschrieben, drei grundlegende Aufgaben. Zuerst müssen die Positionen der Ziel- und Quellperson extrahiert werden aus den vorhandenen Videos. Diese Skelette müssen dann verglichen werden in dem hier durch Normalisierung betitelten Prozess. Dabei geht es vor allem um die Position im Bild und die Unterschiede in der Körpergröße. Im dritten Schritt wird dann die Position und Bewegung des Skeletts auf die Zielperson übertragen (Chan et al., 2018, S. 2). Damit die Bewegung des Skeletts der Quellperson für das generative Netzwerk, welches auf die Synthetisierung der Zielperson trainiert ist, in einem gut abbildbaren Rahmen ist, sollte sich die Skelettbewegung des Quellskeletts in den Grenzen der Bewegung des Zielskeletts bewegen. Dies wird im Normalisierungsschritt durch aufeinander aufbauende Berechnungen erreicht. Die maximale Bewegung des Skeletts der Zielperson im Raum vor der Kamera gibt den Rahmen vor. Dieser wird berechnet durch die Position des Sprunggelenks im Bild (Chan et al., 2018, S. 9). Ist es nahe an der Kamera und weit unten im Bild? Ist es weiter oben im Bild und dadurch vermutlich weiter von der Kamera entfernt? Anschließend wird derselbe Prozess mit dem Skelett der Quellperson wiederholt. Des Weiteren wird die Körpergröße abgeglichen und angepasst. Durch die Information über die maximale und minimale Position des Sprunggelenks des Zielskeletts im Bild kann die Größe und Position des Quellskeletts auf die max/min Grenze des Zielskeletts beschränkt werden. Werte dazwischen werden dann interpoliert. Wie "Global Pose Normalization" schon andeutet, liegt der Fokus jedoch auf der Angleichung der globalen Position im 2D Raum des Videobildes (Chan et al., 2018, S. 2). Obwohl Körpergröße Beachtung findet, wird auf Unterschiede in Körperform und Länge der Gliedmaßen in dieser Lösung keine Rücksicht genommen (Chan et al., 2018, S. 7). Grundsätzlich wäre es auch denkbar das Skelett der Zielperson nach der Bewegungserfassung, aber vor dem Training des Generators auf die Position und Größe des Quellskeletts anzupassen. Dann würde die Position sowie Größe der Bewegungsquelle sich in der gelernten Struktur des neuronalen Netzwerks wiederfinden. Dadurch wäre

jedoch der Austausch der Bewegungsquelle erschwert. Es macht somit Sinn den häufigen wechselnden Part zu normalisieren, um diese Variable nicht durch die Integration in das Training des neuronalen Netzes zu fixieren, wodurch das ganze System Flexibilität einbüßen würde.

Abgrenzung zu Quelle-Ziel-Kalibrierung: Der Normalisierungsschritt ist nicht mit einer Quelle-Ziel-Kalibrierung zu verwechseln. Dieser Schritt ist häufig bei klassischen Motion Capturelösungen, wie auch bei dem in Kapitel XXX behandelten Perception Neuron Anzug, notwendig. Bei der Quelle-Ziel Kalibrierung wird z.B. ein zu animierender 3D Charakter oder ein repräsentatives Skelett in der Animationssoftware in eine T-Position gebracht und der Schauspieler nimmt mit dem angezogenen Anzug ebenfalls eine T-Position ein. Dies ist notwendig, um Ungenauigkeiten bei der Platzierung der Sensoren oder alternativ der Position der Trackingmarker auszugleichen und eine hohe Übereinstimmung zwischen Schauspieler und Zielskelett, oder bei direkter Übertragung, dem Zielcharakter, zu gewährleisten. Da OpenPose den Schritt der Skelettextrahierung übernimmt, werden die Daten zur Bewegungserfassung durch die neuronale Softwarearchitektur theoretisch (!) stets auf die gleiche Art und Weise gewonnen. Aus diesem Grund fällt eine Quelle-Ziel-Kalibrierung weg. Dass direkt zwischen 2D Skeletten normalisiert werden kann, bringt noch einen weiteren Vorteil mit sich. Der Zwischenschritt in die 3D Sphäre wird von Chan et al. vermieden. Auch in vielen klassischen Motion-Capture Produktionen wird Bewegungstransfer, wenn auch in der 3D Sphäre, angestrebt. Dies ist sogar der meist verbreitete Fall, wenn einem in 3D modellierten Charakter durch einen Schauspieler Leben eingehaucht werden soll. Der Ausdruck Bewegungstransfer kann im wörtlichen Sinn somit auch klassische Bewegungserfassung und anschließende 3D Charakter Animationen beschreiben. Dieser Prozess der Animation unterscheidet sich jedoch grundlegend von der in dieser Arbeit angewandten Lösung (Chan et al., 2018, S. 1).

6.3 Phase 3 – Generierung

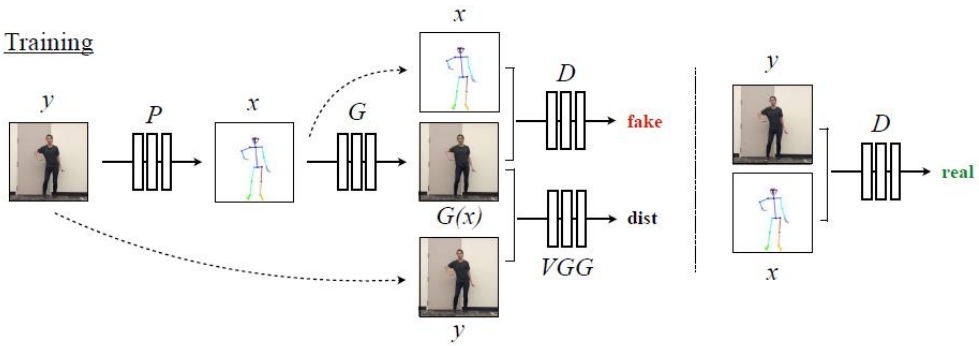
Synthetisierung neuer Posen: Wie gelangt man von den normalisierten Posen der Repräsentation der Quellperson zu der generierten Bildsequenz der Zielperson? Wenn man eine Momentaufnahme, i.e. ein Einzelbild, betrachtet, geht es nicht um Bewegungs-, sondern um Positionstransfer. Die Lösung, wie sie in „Everybody Dance Now" erläutert wird, macht sich maschinell lernende neuronale Netze zunutze, um die Position über den Zwischenschritt der Repräsentation durch das vorhin bereits beschriebene zweidimensionale Skelett von Quell- auf

Zielperson zu übertragen. Hierbei wird das Skelett der Position der Quellperson durch den Positionsermittler OpenPose extrahiert. Das Skelett ist hier das Label, die Anleitung für den Bildgenerator. An dieser Stelle ist festzuhalten, dass der Bildgenerator einzigartig für jede Person ist. Dieser hat somit keine Vorstellung von einer anderen Person neben der Erscheinung der von ihm ‚studierten' Zielperson. Die Anleitung der Bewegungsquelle (i.e. das extrahierte Skelett) ist jedoch bekannt genug um mit der Anleitung, dem Label der Zielperson ausgetauscht werden zu können. Da auch die Position im Bild eine relevante Information im neuronalen Netzwerk des Generators darstellen kann und es auch Verluste an Information durch den Sprung von der 3D in die 2D Sphäre sowie durch Abweichungen zwischen der Anatomie der Quell- und der Anatomie der Zielperson gibt, kommt noch der vorhin beschriebene Zwischenschritt der Normalisation der repräsentativen Skelette zum Einsatz. Der Bildgenerator erstellt nun ein möglichst täuschend echtes Bild der Zielperson nach der Anleitung der Bewegungsquelle.

Bekannte Erscheinung / neue Pose: Die Frage, wie man eine Person in nicht aufgezeichneten Posen synthetisieren kann, ist keine neue. Mehrere Forschergruppen haben bereits Lösungen erarbeitet, wie dies zu bewerkstelligen ist (Chan et al., 2018, S. 2). Jedoch ist häufig die zeitliche Komponente der Generierung nicht mit ausreichender Qualität für die Generierung eines Videos gegeben. Ein vielversprechender Ansatz zukünftige Bewegung vorherzusagen wurde in dem Paper "Learning to Generate Long-term Future via Hierarchical Prediction" vorgestellt (Villegas et al., 2018). Dass die Lösung dieses Teams so gut funktioniert, erklären sie durch die vorhergehende Extraktion von abstrakteren Konzepten wie Erscheinung aus dem Ausgangsmaterial. Wie sich das Erscheinungsbild wahrscheinlich verändern wird, ist dann die nächste Frage. Im Vergleich dazu endet für pixelbewegungsbasierte Lösungen, die sich beispielsweise auf simple Betrachtungen des optischen Flusses beschränken, die Abstraktionskomplexität bei der Frage, wohin sich ein Pixel oder eine Gruppe von Pixeln als nächstes bewegen wird. Bei diesem Prozess besteht keine Repräsentation von identifizierbaren Subjekten und Objekten und ihren möglichen Bewegungsspielräumen in ihrer "optischen Zukunft".

Optischer Fluss (i.e. Optical Flow) ist für Motioncapture Lösungen, die aus 2D Video Bewegungsdaten extrahieren eine nützliche Herangehensweise (Mündermann et al., 2006, S. 4). Grundlegend definiert optischen Fluss die Bewegung eines visuellen Bildinhalts. Dies kann sowohl die Bewegung eines Subjekts oder einer Kamera durch eine statische Umgebung, wie auch die Bewegung eines Sub-/Objekts vor der Kamera oder dem Subjekt sein. Im Fall von Algorithmen, um Bewegung zu erfassen, geht es aber um den optischen Fluss wie

er aus der Positionsveränderung von Pixelwerten zwischen 2D Einzelbildern eines Videos berechnet werden kann. Die Kunst moderner Software besteht darin, die im optischen Fluss enthaltenen Informationen mehr und mehr sinnerfassend zu erkennen und, wie es hier in Phase 3 gelingt, in generativen Verfahren glaubhaft abbilden zu können.



6.4 Trainingsarchitektur

Abbildung 26: Trainingspipeline von „Everybody Dance Now“ (Chan et al., 2018, S. 3)

Trainingsvisualisierung: In der obenstehenden Abbildung ist das komplette Trainingssystem zum Erreichen des Hauptziels – Bewegungstransfer, ersichtlich. Es beginnt mit einem Einzelbild y vom original Zielvideo. Mittels dem Positionserfasser P, OpenPose, wird dann das zu y gehörende Skelett x extrahiert. Die drei vertikalen Blöcke deuten auf die neuronale Architektur von OpenPose. Von x ausgehend erzeugt der Generator G ein neues synhetisiertes Bild G(x), wiederum mit Hilfe seines neuronalen Netzwerks. Bei perfekter Synthetisierung ist G(x) nicht von dem Ausgangsbild y zu unterscheiden. Um schneller zu einer brauchbaren Synthetisierung zu gelangen, wird das erzeugte Bild G(x) mittels dem vortrainierten VGG Netzwerk mit dem Ausgangsbild y verglichen. Daraus ergibt sich ein mathematischer Verlust, welcher dem Generator als Richtungsweiser zur Verbesserung seines Netzwerks hin zu realistischer Bildgenerierung dient. Der Hauptgegenspieler von Generator G ist jedoch der Diskriminator D. Dieser vergleicht das generierte Bild G(x) zusammen mit dem Skelett x mit dem realen Paar (x,y). Dabei versucht der Diskriminator in jeder Runde mehr darüber zu lernen, was den Unterschied zwischen einem realen Paar (x,y) und einem generierten Paar (x,G(x)) ausmacht. Auf ähnliche Weise lernt jedoch auch der Generator G durch das Ergebnis seines Gegenspielers D, wie er den Unterschied

zwischen realem und synthetisiertem Paar durch eine hochwertigere Generierung des Bildes G(x) minimieren kann.

Raum-Zeit Glättung: In dieser Trainingspipeline fehlt jedoch ein wesentlicher Teil der von Chan et al. vorgestellten Methode – ihre Lösung zur zeitlichen Glättung. Um eine zeitliche Glättung der Bewegung in den generierten Bildsequenzen zu ermöglichen, wird in einer zusätzlichen Aufgabe für Generator und Diskriminator auch das jeweils vorangegangene Bild-Skelettpaar gemeinsam mit dem aktuellen Bild-Skelettpaar mit dem vorangegangenen generierten Bild-Skelettpaar und dem aktuellen generierten Bild-Skelettpaar verglichen. Ausreißer und abwegig generierte Einzelbild-Skelettpaare, die auf wenige Bilder beschränkt sind, finden so keinen Weg mehr in die Sequenz. Dadurch erhofft man sich ein verbessertes Endresultat (Chan et al., 2018, S. 4). Der im englischen Paper verwendete Originalbegriff spatio-temporal bedeutet, dass diese Eigenschaft, hier Bewegung, sowohl eine räumliche als auch zeitliche Dimension aufweist, die nicht voneinander isoliert werden können. Der Bindestrich ist hier das ausschlaggebende Element. Bewegung kann nur in beiden Domänen gleichzeitig existieren. Da der Algorithmus jedoch keine Repräsentation der realen Welt mit ihrer Physik enthält, kommt es zu unangenehmen Nebeneffekten, wie z.B. Sprüngen oder Verformungen. Daher ist die Glättung, die sowohl Raum als auch Zeit berücksichtigt hilfreich, um das Ergebnis realer wirken zu lassen.

Temporale Kohärenz: Per-frame image-to-image heißt zu deutsch, dass der ganze Algorithmus alle Fragen zu einem Bild beantwortet, bevor zum nächsten übergegeangen wird. Die Grundfunktionalität von „Everybody Dance Now"'s Softwarelösung kann als Per-frame image-to-image Methode bezeichnet werden. Wenn ein Moment nach dem nächsten in Form von aufeinanderfolgenden Einzelbildern abgearbeitet wird und der Algorithmus dabei jedoch keine Rücksicht auf die inhaltliche Information der vorangegangenen und darauffolgenden Positionen der Person und des Skeletts nimmt, stechen Unterschiede in den einzeln generierten Bildern durch ihre sequentielle Aneinanderreihung schnell ins Auge. Die dadurch entstandenen Fehler wie Sprünge oder erratische Wackler und Zitterer in der Bewegung der Zielperson sind dabei eindeutig Merkmale für den Betrachter, die Sequenz als unechte Generierung einzuordnen. Daher ist es das Ziel, zeitlich kohärentes Videomaterial zu synthetisieren, in dem die Veränderung zum nächsten Einzelbild realistisch und glaubwürdig erscheint.

Zeitliche Kohärenz als zu optimierendes Lernziel: Mit der in „Everybody Dance Now" vorgestellten Methode lassen sich glaubwürdige Tanzsequenzen von Zielpersonen erstellen. Dennoch gibt es einige Abschnitte in der Pipeline, welche von zusätzlicher Forschung profitieren können. Beispielsweise ist die temporale

Kohärenz und wie man diese für ein neuronales Netz in der Verlustfunktion abbildet noch nicht in vollem Umfang optimiert. Jedoch gibt es bereits Ansätze, die gute Ergebnisse liefern (T.-C. Wang et al., 2018a). In dieser Frage bleibt dennoch ein weites Feld für neue Erkenntnisse.

Repräsentation von Bewegung: Ähnlich verhält es sich mit der Repräsentation menschlicher Bewegung. Da OpenPose lediglich einzelne Positionen erfasst und in dieser Repräsentation nicht ganze Bewegungsabläufe beinhaltet, ist auch an dieser Stelle noch zeitliche Information vorhanden, welche in Zukunft verstärkt genutzt werden kann (Chan et al., 2018, S. 7). Dadurch entstandene Ungenauigkeiten von Skelettframe zu Skelettframe versucht das Team der UC Berkeley durch ihr zeitliches Glättungssetup auszugleichen. Ob ein Skelett die beste Repräsentation für menschliche Bewegung ist, bleibt eine aktuelle Frage, wie an Alternativen zu OpenPose wie z.B. DensePose ersichtlich ist.

Fehlerfunktion: Wie lernt ein neuronales Netz? "Neural networks don't really learn data... they minimize the loss function" (Trask, 2019, S. 194). Ein neuronales Netzwerk ist eine Struktur, die Informationen ausbilden, abbilden und wiedergeben kann. Welche Informationen jedoch diese Struktur ausmachen soll, wird erst durch eine Verlustfunktion (engl. Loss function) vorgegeben. Diese Verlustfunktion wird im Lernprozess minimiert. Da diese Funktion so essentiell für die Funktionsweise eines neuronalen Netzes ist, gibt es mehrere gleichwertig verwendete Bezeichnungen. Häufig liest man auch Error Funktion oder Zielvorgabefunktion (engl. Objective function) (Trask, 2019, S. 195). Wenn man zwei ident entworfene neuronale Netze nimmt und ihnen verschiedene Verlustfunktionen zum Minimieren vorgibt, werden sich unterschiedliche Strukturen herausbilden und die Netzwerke werden zu abweichenden Schlüssen kommen. Die Zielvorgabe für die Generatoren und Diskriminatorennetzwerke, wie sie in „Everybody Dance Now“ zum Einsatz kommen, machen sich spezialisierte Fehlerfunktionen aus vorangegangenen Arbeiten wie (kurz) Pix2PixHD (T.-C. Wang et al., 2017) oder VGGNet (Simonyan & Zisserman, 2015) zunutze (Chan et al., 2018, S. 5).

$$\min_G \left(\left(\max_{D_1, D_2, D_3} \sum_{k=1,2,3} \mathcal{L}_{\text{smooth}}(G, D_k) \right) + \lambda_{FM} \sum_{k=1,2,3} \mathcal{L}_{\text{FM}}(G, D_k) \right.$$
$$\left. + \lambda_{VGG} \left(\mathcal{L}_{VGG}(G(x_{t-1}), y_{t-1}) + \mathcal{L}_{VGG}(G(x_t), y_t) \right) \right)$$

Formel 1: Vollständiges GAN Lernziel für Generator und Diskriminator (Chan et al., 2018, S. 5)

Zwischen 0 und 1: Aktivationsfunktionen in neuronalen Netzen können verschieden gestalten werden. Um eine mathematische Wahrscheinlichkeit handhabbar vorherzusagen, ist es häufig hilfreich, den schlussendlich ausgegebenen Wertebereich eines Neurons mittels einer Sigmoid Funktion zwischen 0 und 1 zu quetschen. Je näher der Wert der Verlustfunktion (Loss Function) an 1 ist, desto wahrscheinnlicher trifft die Aussage/Aktivation zu. Jedoch ist die Sigmoid Funktion nicht die einzige, welche zurzeit häufig in neuronalen Netzwerken zum Einsatz kommt (Bettilyon, 2019).

Mean und Mittel: Das arithmetische Mittel beschreibt den Wert, in welchem alle Datenpunkte eines Sets addiert und anschließend durch die Anzahl an Datenpunkten dividiert werden. Dieses Mittel unterscheidet sich vom Median, welcher den genau in der Mitte eines Sets liegenden Wert beschreibt.

Fitness und Overfitting: “a mediocre algorithm with 100 million words of unlabeled training data outperforms the best known algorithm with 1 million words” (Russell et al., 2010, S. 28).

Das Verhältnis von Eingabepixeln, in denen zu lernende Muster enthalten sein können, z.B. ein gezeichneter Charakter, und der Anzahl an Bildern dieses Charakters im Datensatz ist ausschlaggebend für die erreichbare Fitness der neuronalen Struktur. In diesem Beispiel wäre das die Allgemeingültigkeit der Abstraktion des gelernten Charakters, zu welcher das fertig trainierte neuronale Netzwerk am Ende gelangt. Durch zu wenige Beispiele im Datensatz besteht das Risiko, dass das Netzwerk beim Training sehr gute Ergebnisse erzielt, jedoch bei einem Test auf neue Daten ähnlichen Typs schlecht abschneidet. Dieses häufig auftretende Problem nennt sich Überanpassung (engl. Overfitting). Das Risiko des Overfittings kann also durch Verringerung des Möglichkeitsraums, i.e. der Eingabevariablen, sowie durch die Vergrößerung des Trainingsdatensatzes reduziert werden (Russell et al., 2010, S. 705). Die Testgenauigkeit sollte im Idealfall nur wenig schlechter ausfallen als die Trainingsgenauigkeit. Ist dies der Fall spricht man von einem fitten, angepassten neuronalen Netzwerk.

Abbildung 27: Internet Meme zur Veranschaulichung von Overfitting (Oppermann, 2019)

Wie Russell und Norvig im obenstehenden wörtlichen Zitat beschreiben, ist die Größe der Datenbank häufig ausschlaggebender für die Fitness eines Netzwerks als die Wahl des richtigen Algorithmus. Bei meinem praktischen Versuch wird sich diese Erkenntnis als nützlich herausstellen. Die Frage nach dem Einfluss der Datensatzgröße auf das Ergebnis des Bewegungstransfers stellt sich auch in der Bewertung der Leistungsfähigkeit weiterer Lösungen zum Bewegungstransfer neben der in „Everybody Dance Now“ vorgestellten. Ansätze wie die im Kapitel „Verwandte Forschungsarbeiten“ kurz angesproche Lösung „Few-shot Video-to-Video Synthesis“ (T. Wang et al., 2019) verspricht beispielsweise eine Reduktion der notwendigen Zieldaten für einen erfolgreichen Bewegungstransfer.

Beaufsichtigtes maschinelles Lernen: Image-to-image translation von Bewegung eines Strichmännchens zu korrespondierendem Einzelbild einer Zielperson ist ein sogenannter "Supervised machine earning" Prozess. Bei beaufsichtigtem maschinellen Lernen werden Eingabe-Ausgabe Paare gelernt. Anhand der dadurch gewonnenen Struktur kann dann von neuen Eingabedaten gleichen Typs auf eine Ausgabe geschlossen werden (Russell et al., 2010, S. 695). Beaufsichtigtes maschinelles Lernen findet sich insofern im Prozess von „Everybody Dance now“, als dass für jedes Bild sowohl des Bewegungsziels als auch der Bewegungsquelle ein Label, eine strukturierte Zusatzinformation zum Bild erstellt wird, welche die gewünschten Eigenschaften in ausreichend genauem Maß beschreibt. Hier ist das bunte Skelett der Posenerfassung diese Zusatzinformation. Durch das Paar aus Videobild und aus diesem Videobild

extrahierten Skelett existiert stets ein vollständiges Set, um einen Zusammenhang zu erlernen und diesen in einem neuronalen Netz abzuspeichern. Das Label-Bild Paar ist in diesem Fall sogar relativ einfach visuell von den Nutzern der Softwarelösung zu überprüfen.

Zeitliche Kohärenz: Geschickt entworfene mathematische Lernziele und eine Bündelung der Eingabedaten in Bild/Labelsequenzen ermöglichen es auch, mit Methoden des maschinellen Lernens die mögliche Ausprägung von Bewegung in nicht trivialer Manier in neuronalen Strukturen abzubilden (Chan et al., 2018, S. 2). Wie die meisten Lösungen zu Bilderkennung und Bildgenerierung greift auch das Team von „Everybody Dance Now“ auf verschiedene vorangegangene Softwarearchitekturen und Lösungsstrategien zurück. So gleicht die Zieldefinition, die das neuronale Netz des Diskriminators lernen soll, um ein echtes Bild von einem synthetisierten unterscheiden zu können, der Zieldefinition, wie sie in „Image-to-Image Translation with Conditional Adversarial Networks“ (Isola et al., 2016) vorgestellt wurde. Diese Lösung, auf welcher die Arbeit von Chan et al. aufbaut, wird kurz als Pix2Pix bezeichnet. Der erste Teil des Lernziels von „Everybody Dance Now“ besteht darin, ein mapping von Inputskelett zu Ausgabebild einer spezifischen Person zu lernen. Die zweite Hälfte des Lernziels ist durch die zeitlich möglichst kohärente Erstellung einer glaubhaften Sequenz motiviert. Das "temporal smoothing", wie die Lösung in „Everybody Dance Now“ genannt wird, besteht darin, den Diskriminator auch den Unterschied zwischen einer echten Videobildabfolge und einer synthetisierten lernen zu lassen und dies in die Zieldefinition aufzunehmen. Das gleiche Ziel in umgekehrter Optimierung gilt auch für den Generator des GAN (Chan et al., 2018, S. 4).

VGGNet als Wahrnehmungsstellvertreter: 2010 wurde von US Eliteuniversitäten ausgehend eine Challenge gestartet, um Bilder mit dem aus einer gigantischen Anzahl an Fotos und dazugehörigen Kategorien bestehenden Imagenet Datensatz von einem Computerprogramm klassifizieren und im Bild verorten zu lassen (*ImageNet Large Scale Visual Recognition Competition (ILSVRC)*, 2020). Jährlich wurden dann die besten Algorithmen gekürt, auf deren Basis eine Vielzahl der Fähigkeiten des Computersehens, wie es heute in Verwendung ist, erforscht und entwickelt wurde. Der Wettbewerb nennt sich ILSVRC - oder auch ImageNet Large-Scale Visual Recognition Challenge. 2014 haben neben einem Netzwerk von Google zwei tiefe neuronale Netze mit einmal 16 und einmal 19 Ebenen Tiefe und neuen Faltungscodierungen (Engl. Convolution) den ersten Platz in Lokalisation und den zweiten Platz in Klassifikation erreicht (Simonyan & Zisserman, 2015, S. 1). Dieses VGG16 bzw. VGG19 Netzwerk, welches trainiert ist auf höchste Ansprüche in der Erkennung und Verortung von Objekten und Personen in Bildern, kommt auch als

vortrainierter „Stellvertreter“ menschlicher Wahrnehmung in der Lösung von „Everybody Dance Now“ vor. Worauf zu achten ist, damit ein generiertes Einzelbild einem menschlichen Betrachter als realistisch erscheint, muss in den einzelnen Ebenen des trainierten Generatornetzwerkes G gelernt werden. Welche Kombination aus aktivierten Neuronen in der nächsten Ebene des Netzwerkes einen bestimmten Knotenpunkt aufleuchten lässt, muss im Trainingsprozess gelernt werden. Durch eine Verlustfunktion für den Generator in Abhängigkeit von dem auf eine Vielzahl an echten Fotos vortrainierten VGG Netzwerk kann der Realismus der synthetisierten Bilder verbessert werden (Chan et al., 2018, S. 3). Dabei kommen die spezifischen Eigenschaften eines realen Fotos, wie sie in den faltungscodierten "Wahrnehmungsebenen" des VGG Netzwerks mathematischen Ausdruck finden, zum Einsatz. Im Idealfall gleicht dadurch das neue falsche Bild dem echten korrespondierenden Bild, wie es Teil des Zielvideodatensatzes ist, auf eine dem außenstehenden Betrachter realistisch und glaubwürdig erscheinende Weise.

Wahrnehmungsfehlerfunktion: Im englischen Original wird die durch das VGG Netzwerk ermöglichte Wahrnehmungsfehlerfunktion auch als perceptual reconstruction loss bezeichnet. Der Grund weshalb das vortrainierte VGG Netzwerk so nützlich ist, um die Qualität der generierten Bilder zu verbessern, liegt in der Annahme begründet, dass die in den tiefen Ebenen der „neuronalen“ Struktur notwendigen Gewichte und Parameter für ein grob als Wahrnehmung beschreibbares Ziel abgespeichert sind (Johnson et al., 2016, S. 6). Wenn man nun diese semantisch/inhaltlich abstrakten Konzepte in Bezug auf die als realistisch wahrgenommene Einschätzung des generierten Bildes der Zielperson betrachtet, so kann der Generator aus dem mathematischen Unterschied, der Fehlerfunktion (engl. loss function) lernen. Dies wird in der Generatorfunktion, welche mit der Rekonstruktion eines glaubhaften Bildes beauftragt ist, als „perceptual reconstruction loss“ bezeichnet (Chan et al., 2018, S. 3).

Nächster Nachbar: Mit dieser in „Everybody Dance now“ vorgestellten Methode unterscheidet sich der Lernprozess von früheren Algorithmen die z.B. Nächster Nachbar (engl. Nearest Neigbor search) verwenden. Nearest neighbor search ist ein Algorithmus, mit dem der nächste Nachbar in einem Datenset gefunden werden kann. Dies kann beispielsweise bei Empfehlungen zum Einsatz kommen. Angenommen ein Nutzer oder eine Nutzerin einer On Demand Videoplattform überlegt sich, welchen Film er oder sie gerne sehen möchte. Die Plattform kann die Meinung der Nutzer zu Filmen, die er oder sie nicht bewertet hat, nicht wissen. Ein möglicher Ansatz ist es daher, sich Nutzer in der Datenbank anzusehen, die ähnliche Filme sehen und positiv bewertet haben. Darauf basierend kann den neuen Nutzern ein von bestehenden Nutzern ähnlichen Profils

positiv bewerteter Film empfohlen werden. Dafür ist nearest neighbor search eine mögliche Lösungsstrategie (Schutt & O'Neil, 2013, S. 202).

Generative Adversarial Networks (GANs): GANs sind aus der Landschaft des generativen maschinellen Lernens nicht mehr wegzudenken. Die Vielfalt der Anwendbarkeit reicht über die hier beschriebenen Aufgaben von Bild- und Videogenerierung weit hinaus. Grundsätzlich sind alle Fragestellungen optimal für den GAN Ansatz geeignet, welche neue Daten zu generieren suchen wie z.B. realisitsche Bilder, Sätze oder Sprache (Gerrish, 2018, S. 168). Verwandte bereits erforschte Fragestellungen aus dem Bereich der Bildgenerierung schließen von beschreibendem Text, Audiosamples, oder semantischen Karten auf ein zu erzeugendes Bild. Die Funktionsweise von GANs wurde von Ian Goodfellow et. al. in dem Paper "Generative Adversarial Networks" 2014 vorgestellt (Goodfellow et al., 2014). Im Kern steht dabei ein MiniMax Spiel/Algorithmus zwischen einem Diskriminator und einem Generator. Diese zwei „Spieler" sind einander diametral entgegengesetzt. So versucht der Diskriminator beispielsweise immer einen Fehlerwert zu minimieren, welchen der Generator zu maximieren sucht (Goodfellow et al., 2014, S. 2). So könnte beispielsweise ein Diskriminator D die Wahrscheinlichkeit in jedem Rechendurchgang minimieren, ein unechtes Bild fälschlicherweise als echtes zu klassifizieren. Der Generator möchte diesem Unterfangen durch bessere Synthetisierung seines nächsten Bildes gegensteuern. Daraus resultiert ein MiniMax Spiel. Die Wahrscheinlichkeiten für Generator und Diskriminator werden dabei in Runden nacheinander, jedoch in einer einzigen Funktion aktualisiert (Steenbrugge, 2019).

Zusammenhänge lernen mittels Conditional GANs: Der im praktischen Teil dieser Diplomarbeit verwendete Code basiert auf der in "Everybody Dance Now" vorgestellten Methode. Diese bezieht ihr Fundament wiederum aus der Forschung rund um GANs und ihre Anwendbarkeit auf visuelle Fragestellungen. So ist es möglich, mittels dieser Netzwerke scharfe Details und hochqualitative Bilder zu generieren (Chan et al., 2018, S. 2). Besonders relevant ist die Forschung in Bezug auf abstrakte Informationen, die in eine Beziehung zu einem weiteren Datensatz gesetzt werden können. Die Aufgabe, komplexe Zusammenhänge zu lernen, eignet sich besonders für das mathematische GAN Lernziel. Dieser bereits kurz als beaufsichtigter Lernprozess verortete Zugang wird auch häufig als „conditional GAN" bezeichnet. Generator und Diskriminator lernen dabei die Verknüpfung zwischen einem Datensatz mit strukturierten Daten wie semantischen Labels und Zusatzbildern. Das konkret in „Everybody Dance Now" verwendete Beispiel dafür sind die Strichmännchen/Skelette, die mittels OpenPose erstellt und dann dem Video als Abhängigkeit (engl. Condition) beigefügt werden. Weiters ist die Forschung bezüglich der Herausforderung, eine

formale Repräsentation für die temporale Dimension von Video zu finden, auf der das Netzwerk trainiert werden kann, von großem Wert für „Everybody Dance Now“ wie auch diese Arbeit (Chan et al., 2018, S. 8).

6.4.1 Gesichtsoptimierung

Verlorene Identität: Dadurch dass ein Skelett als Zwischenschritt erstellt wird, geht im Idealfall ein Großteil von der Identität der Quellperson der Bewegung verloren (Chan et al., 2018, S. 1). Ziel ist es, die Person so zu abstrahieren, dass lediglich die Bewegung als Merkmal dieser bestehen bleibt. Die übrig gebliebene Bewegung lässt sich als Unterschied zwischen zwei aufeinanderfolgenden Posen beschreiben. Für das synthetisierte Bild gilt jedoch das genaue Gegenteil. Die Identität der Zielperson, wie sie sich in ihrem Erscheinungsbild ausdrückt, sollte möglichst vollständig erhalten bleiben.

Gesichtsoptimierung: Um die Qualität des generierten Gesichts im Ausgabevideo zu verbessern, kommt ein speziell darauf fokussiertes, eigenes neuronales Netzwerk zum Einsatz. Zuerst muss der Bereich im Bild gefunden und eingegrenzt werden, in dem sich das Gesicht befindet. Diese Koordinaten bieten dann die Grundlage, um das Gesicht erneut in höherer Qualität zu synthetisieren. Dazu bietet sich erneut die GAN Struktur an. Im Trainingsschritt generiert ein eigenes Netzwerk ein Bild des Gesichts der Zielperson unter Zuhilfenahme echter Aufnahmen aus dem Video der Zielperson. Der Diskriminator des GAN versucht dann zwischen einem echten Einzelbild und dem vom Generator erzeugten zu unterscheiden. Nach einigen Durchläufen mit sämtlichen Bildern des Datensatzes, sogenannten „Epochen“ (Burkov, 2019, Kapitel 4, S. 5), verbessert sich das Generatornetzwerk idealerweise zu einem für den Menschen glaubwürdigen Generator des Gesichts der Zielperson. Nun kann das generierte Video des Bewegungstransfers herangezogen werden und das Verbesserungsnetzwerk für das Gesicht eine neue, im Kopfbereich noch glaubwürdigere Version des Ausgabevideos berechnen. Gesichtserkennung von verschiedensten Menschen ist eine besonders ausgeprägte Fähigkeit der menschlichen Spezies. Darum fällt eine Ungenauigkeit sowohl in der statischen Form als auch in der zeitlichen Abhängigkeit von Moment eins zu Moment zwei besonders auf. Ist eine realistische Gesichtserzeugung nicht erfolgreich, so gelangt man schnell in den Bereich des „Uncanny Valley“, wo die abweichende, häufig deformierte Anatomie zu Schaudern bei den Betrachtern führt.

Face Image Residual: „The flip side of perception is creativity[…]” wird in einem TED Talk zum Thema generative neuronale Netzwerke proklamiert (Arcas, 2016, Sek. 0:55). Wahrnehmung wird von manchen Forschern im Bereich

Computer Vision als die Kehrseite und zugleich Basis der Kreativität betrachtet. Daher ist ein Netzwerk, das auf die Klassifikation von Objekten in Bildern spezialisiert ist, auch im Umkehrschluss gut geeignet, realistische Bilder zu „halluzinieren“. Den Erfolgen der Forschung zum tiefen VGG Netzwerk nachfolgend hat im Jahr 2015 eine neue Struktur eines tiefen neuronalen Netzes mit 152 Ebenen sämtliche Kategorien des ILSVRC Wettbewerbs für sich entscheiden können (Russakovsky et al., 2015). Die Netzwerkarchitektur nennt sich ResNet was eine Abkürzung für Residual Network darstellt. Die Idee ist es, dass es in manchen Fällen einfacher ist, den Unterschied zu einer Eingabefunktion in der aktuellen Ebene des Netzwerks zu lernen als nur die Ausgabefunktion ohne Unterschied zur „Identitätsfunktion“ der vorherigen Ebene (Sutton & Barto, 2018, S. 184). Identität beschreibt dabei lediglich den Stand des Netzwerkes vor der aktuell durchzurechnenden neuronalen Ebene. Das mathematische Residuum beschreibt den Unterschied dazu. Diese Abweichung, welche im Englischen manchmal auch „Residual Error“ genannt wird, soll minimiert werden (Burkov, 2019, Kapitel 7, S. 11). Das Team von „Everybody Dance Now“ macht sich diese Architektur auch zunutze in ihrer Arbeit. So wird das Gesicht der Zielperson, da es besonders glaubhaft generiert werden soll, nochmals mit dem oben beschriebenen Gesichts-GAN verbessert. Dabei kommt ein „face image residual“ zum Einsatz, der auf der Architektur von Pix2PixHD (T.-C. Wang et al., 2017) basiert und für den Generator eine Auflösung von 128x128 Pixel vorsieht. Der Diskriminator des Gesichtsnetzwerks ist in Form eines 70x70 Patch-GAN gestaltet (Chan et al., 2018, S. 5; Isola et al., 2016).

6.5 Leistungsmetriken

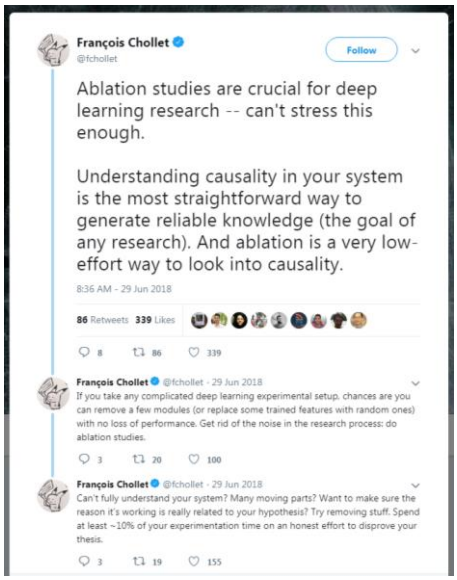


Abbildung 28: Tweet zur Relevanz von Ablationsstudien in der Forschung zu tiefen neuronalen Netzen (Chollet, 2018)

Neuronale Architekturen mit ihren vielen Verbindungen und gefaltenen Ebenen sind häufig nicht nur für Außenstehende, sondern auch für die ForscherInnen und EntwicklerInnen selbst schwer zu analysieren. Was wurde gelernt? Warum funktioniert eine Klassifikation oder eine Generierung? Warum funktioniert sie nicht? Überprüfbare Kausalität ist essentiell für reproduzierbares Wissen, welches Ziel jeder Forschung ist. Eine vergleichsweise simple Methode in der Welt des maschinellen Lernens ein System besser zu verstehen ist es, eine Ablationsstudie durchzuführen. François Chollet, der Entwickler der Bibliothek Keras für maschinelles Lernen in Python, beschreibt in dem hier abgebildeten Tweet diese als essentiell für ein Verständnis kausaler Zusammenhänge (Chollet, 2018). Ablation bedeutet, einzelne Teile eines Systems mit mehreren Elementen strukturiert zu entfernen und dadurch ihren Einfluss auf das System zu ergründen. Häufig deckt sich das erwartete Ergebnis, wie und wie sehr einzelne Elemente zum Erfolg beitragen, nicht mit den Vorstellungen der EntwicklerInnen. Auch in „Everybody Dance Now“ wird eine Ablationsstudie durchgeführt, um den Einfluss ihrer Lösung zur zeitlichen Glättung sowie zur Verbesserung der Gesichtsgenerierung mit Versionen ohne diesen Lösungen zu vergleichen (Chan et al., 2018, S. 7).

Posendistanzmetrik: Die Ergebnisse der von Chan et al. vorgestellten Methode zum Bewegungstransfer sollen in einer geeigneten Metrik bewertbar gemacht werden. Dazu bedient sich das Team von „Everybody Dance Now“ unter anderem einer Positionsdistanzmetrik. Aus den Bewegungsdaten des Quellvideos der Tänzerin/des Tänzers wird ein Skelett erstellt und normalisiert. Nachdem die ganze Pipeline zu dem fertig generierten Video der Zielperson geführt hat, kann wiederum aus diesem finalen Video mittels derselben Posenerfassungslösung OpenPose ein Skelett erfasst werden. Je näher diese Repräsentation an der Repräsentation des Quellvideos ist, desto höher ist die Qualität des Generatornetzwerks. Konkret analysiert wird hierbei die Distanz zwischen jeweils

denselben Gelenken im Quell- und Zielskelett – daher der Name "Pose distance metric" (Chan et al., 2018, S. 5).

Fehlende Erfassungen: Des Weiteren ist es hilfreich zu erfassen, wenn ein Gelenk überhaupt nicht mehr in der Posenerfassung des Zielvideos vorhanden ist, jedoch schon im Quellvideo erscheint. Diese fehlenden Erfassungen sind neben der visuellen Beurteilung durch einen Menschen ein guter Hinweis darauf, dass die Qualität des generierten Bildes Probleme aufweist. Diese fehlenden Gelenkskoordinaten können jedoch auch anderen Schwierigkeiten geschuldet sein oder in der Methode zur Posenerfassung durch OpenPose liegen, ohne dass visuell ein gravierender Unterschied feststellbar ist. Dennoch ist die Annahme naheliegend, dass ein qualitativ gut aufgelöstes Bild einer Person zu einer ebenso hochqualitativen Posenerfassung durch die Softwarelösung OpenPose führt (Chan et al., 2018, S. 7).

Structural Similarity SSIM: Ähnlich zu der Aufgabe eines Diskriminators, die Qualität des generierten aktuellen Bildes zu bestimmen, ist das Ziel, eine Metrik zum objektiven Vergleich der Ergebnisse über verschiedene Architekturen hinweg zu ermöglichen. Ausgehend von der menschlichen Fähigkeit, ein echtes von einem unechten Bild zu unterscheiden, suchen Metriken wie Mean Squared Error (MSE), Structural Similarity (SSIM), oder Learned Perceptual Image Patch Similarity (LPIPS) nach einer möglichst objektiven Art und Weise die visuelle Glaubwürdigkeit, i.e. die nachgebildete Realität, von generierten Bildern zu bewerten. Das Team von „Everybody Dance Now“ macht sich für die Bewertung ihrer Methode die Metriken SSIM und LPIPS zunutze (Chan et al., 2018, S. 7). Ganz simple Metriken wie der Mean Squared Error bildeten den Grundstein für die rasche Bewertung von generierten Bildern. Jedoch kommt konkret diese Metrik schnell an ihre Grenzen bei der Vielzahl an Wegen, auf denen neue Bilder generiert werden können. Um die wahrgenommene Qualität eines Bildes in ein mathematisches Modell zu verpacken, gibt es daher mehrere Ansätze. Eine auf MSE reagierende Hypothese lautet, dass nicht der Durchschnitt der Unterschiede, sondern die Unterschiede in der Struktur der Bildinformation entscheidend sind für die wahrgenommene Qualität. Darauf basiert die Logik der Structural Similarity Metrik SSIM, welche 2014 vorgestellt wurde (Z. Wang et al., 2004).

Learned Perceptual Image Patch Similarity: Die zweite Metrik, welche in „Everybody Dance Now“ Verwendung findet und auch für die Auswertung meiner im Rahmen dieser Arbeit durchgeführten Versuche zum Einsatz kommt, ist Learned Perceptual Image Patch Similarity, kurz LPIPS. Diese wurde 2018 von einer Gruppe and Forschern von der UC-Berkeley, sowie OpenAI und Adobe Research vorgestellt (Zhang et al., 2018). Wenn man Menschen ein Bild zeigt und

dann zwei leicht veränderte, verzerrte Versionen dieses Bildes, ist es meist trivial, als Mensch zu entscheiden, welche Variation dem Ausgangsbild mehr ähnelt. Wieso diese Version dem Ausgangsbild mehr ähnelt, ist jedoch nicht trivial mathematisch zu beschreiben.

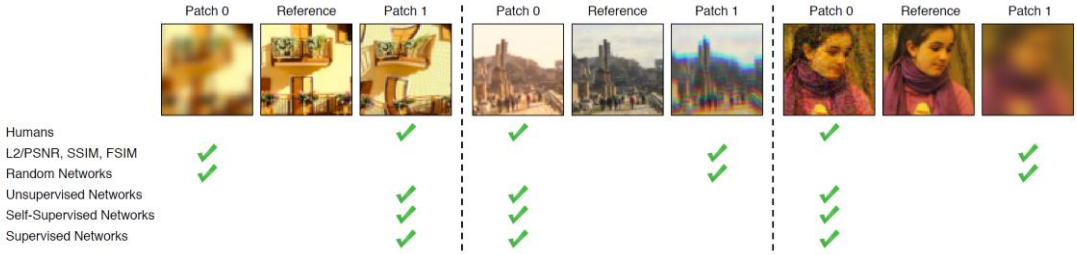


Abbildung 29: Antworten auf die Frage, welches Bild dem mittleren Ausgansbild ähnlicher ist, wie sie Menschen, etablierte Bewertungsmetriken, und tiefe neuronale Netzwerke liefern (Zhang et al., 2018, S. 1).

Wenn man die Aufgabe zur Ermittlung des Wahrnehmungsunterschieds einem neuronalen Netz stellt, welches auf Bilderkennung trainiert wurde, wie z.B. dem VGG Netzwerk, liefert dieses ähnliche Einschätzungen wie ein Mensch. Diese Wahrnehmungsdistanz unterscheidet sich aber meistens von dem vergleichsweise noch einfach nachvollziehbaren Ergebnis der Metrik SSIM. Des Weiteren wird in „The Unreasonable Effectiveness of Deep Features as Perceptual Metric“ dieser Wahrnehmungsunterschied zweier Bilder, ausgedrückt als Distanz, als von selbst entstehende Eigenschaft tiefer neuronaler bildverarbeitender Netzwerke beschrieben (Zhang et al., 2018, S. 1). Die Basis dafür wird in der Tiefe der neuronalen Netze und der Vielzahl an gelernten Eigenschaften und Strukturen, die ein Bild aufweisen kann, verortet. Daher stellt die Metrik LPIPS den Anspruch, eine relativ schnelle Bewertungsmethode eines Bildes zu liefern, welche die für menschliche Wahrnehmung relevanten Qualitätsmerkmale abbildet. Eine größere LPIPS Distanz entspricht dabei geringere Ähnlichkeit.

6.6 Verwandte Forschungsarbeiten

Nachahmung vs. Manipulation: Do-as-I-do retargeting, wie der Bewegungstransfer in "Everybody Dance Now“ noch genannt wird, beschreibt das eins-zu-eins Nachahmen der Quellbewegung durch die generierte Zielperson (Chan et al., 2018, S. 1). Zu klassischen Motion Capture Ansätzen wie auch zu einigen rein digitalen Ansätzen ohne echte Schauspieler für die Bewegungserfassung besteht ein Unterschied. Bei „Everybody Dance Now“ wird ein bestehender Charakter nicht manipuliert, um die Bewegung nachzuahmen, sondern in jedem Einzelbild von Grund auf neu synthetisiert. Manipulation eines

bestehenden Charakters entspricht dem traditionellen Motion Capture Standard, wohingegen die Synthetisierung mittels GAN ohne ein die menschliche Erscheinung definierendes Modell, welches einfach manipulierbar ist, auskommt.

Video Rewrite (Bregler et al., 1997)**:** Die Architektur eines GANs impliziert durch den ersten Begriff des Acronyms – Generative – bereits die Synthetisierung von neuem Material. Dies ist jedoch nicht der einzige Lösungsansatz, der verfolgt werden kann, um Bewegung zwischen Videosubjekten zu transferieren. Beispielsweise bei der Fragestellung, wie eine neue Position und Mimik eines menschlichen Gesichts erzeugt werden kann, um einen so nicht in der realen Welt gesagten Satz auszusprechen, gibt es den Ansatz, in bereits bestehenden, echten Einzelbildern eines Videos der Person den Moment zu suchen, wo Sie ein ähnliches Wort oder einen ähnlichen Satz ausspricht. Durch eine Kombination der Ergebnisse dieser Suche kann jemandem eine Aussage glaubwürdig in den Mund gelegt werden, welche diese Person so nie gesagt hat (Bregler et al., 1997). Da Sprache auf Lauten und Wörtern basiert, die sich wiederholen, ist dieser Ansatz bei genügend Videomaterial, in dem die Zielperson spricht, ausreichend. Bei Bewegungen und der Vielfalt an Positionen, die ein Tänzer in der Lage ist dynamisch einzunehmen, wird es jedoch schwierig, genügend Videomaterial der Zielperson zu produzieren, um jede Position zur Verfügung zu haben. Des Weiteren bietet die generativ-gegenspielige Architektur den Vorteil, dass mit vergleichsweise wenigen Bildern der Zielperson bereits ein plausibler Bewegungstransfer erfolgen kann.

Unbeaufsichtigtes Lernen mit einem GAN: Der Lernprozess, wie er in „Everybody Dance Now“ vorgestellt wird und auch in dem in dieser Arbeit verwendeten Code zum Einsatz kommt, ist beaufsichtigt (engl. supervised), sprich es wird von einem strukturierten Datensatz mit dazugehörigem Label auf einen anderen, neu zu generierenden Datensatz geschlossen. Dies unterscheidet sich grundlegend von Lösungen, die sich keiner Labels bedienen. Im Gegensatz zum beaufsichtigten maschinellen Lernen kommen beim unbeaufsichtigten maschinellen Lernen Datensätze ohne zusätzliche Information aus. Jede Form von unbeaufsichtigtem, maschinellem Lernen ist eine Art der Gruppierung, des Clusterings (Trask, 2019, S. 13). Es wird dabei versucht, die Menge an Eigenschaften in einem Datensatz zu gruppieren, Muster zu erkennen und in der nächsten Ausgabeebene eine auf weniger Eigenschaften verdichtete Information zu gewinnen. Unsupervised adversarial training bei GANs ist auch möglich, beschreibt auch weiterhin das beim generativ-gegenspieligen Prozess typische Katz-und-Maus Spiel zwischen Generator und Diskriminator, jedoch mit unstrukturierten Daten ohne Labels (Gerrish, 2018, S. 168).

MoCoGan (Tulyakov et al., 2017)**:** Wie kann man mit unsupervised adversarial Training die Unterscheidung zwischen Bewegung und Erscheinung lernen? Darauf liefert die Forschungsarbeit „MoCoGan“ eine Antwort. Video ist mehr als die Aneinanderreihung von Einzelbildern. Information wie Bewegung ist in keinem Einzelbild enthalten. Bewegung verrät etwas über die physikalische Welt und ihre Regeln und Gesetze, nach denen sich alle Objekte und Subjekte richten müssen. Dieses physische Modell ist komplex und die Aufgabe, es händisch zu programmieren, alles andere als trivial. Da Menschen rasch auf Bewegung reagieren können und Fehler wie Artefakte besonders auffallen, ist es notwendig, der zeitlichen Kohärenz auch in der Architektur der Softwarelösung zum Bewegungstransfer Aufmerksamkeit zukommen zu lassen. Die Architektur von „MoCoGan“ unterscheidet z.B. zwischen einem „physikalischen“ Bewegungsmodell und der statischen Erscheinung eines Subjekts/Objekts (Tulyakov et al., 2017, S. 1). Dies schafft die Architektur dadurch, dass sie einen ersten Diskriminator darauf trainiert, zwischen einem echten und einem unechten synthetisierten Einzelbild zu unterscheiden. Ein zweiter Diskriminator wird dann auf eine vorab festgesetzte Anzahl an echten Einzelbildern eines Videos und möglichst täuschend echt generierten Einzelbildern derselben Anzahl trainiert, um diese mit hoher Wahrscheinlichkeit zu unterscheiden. Bei „MoCoGan“ ist der vor dem Training des Netzwerks festgesetzte Hyperparameter, welcher die Anzahl der Bilder pro Sequenz beschreibt, standardmäßig 16. Das heißt, es werden 16 Bilder auf einmal betrachtet und mit 16 in gleicher Abfolge generierten Einzelbildern verglichen (Tulyakov et al., 2017, S. 4). Besonders hervorzuheben ist hier, dass dies ohne Beaufsichtigung/Labels (i.e. unsupervised training) funktioniert. Auch bei Bewegungstransfer, wie er in „Everybody Dance Now“ beschrieben wird, ist zeitliche Kohärenz von höchster Bedeutung. Der Vorgang ist hier ähnlich, jedoch kommen Zusatzinformationen (Labels) in Form von einem Strichmännchen zum Einsatz, wodurch der Prozess als beaufsichtigt gild (i.e. supervised training). Des Weiteren werden bei der Lösung von Chan et al. lediglich zwei echte Einzelbilder mit zwei unechten Einzelbildern verglichen (Chan et al., 2018, S. 4).

Dynamics Transfer GAN (Baddar et al., 2017)**:** Idealerweise benötigt man, um die Ziele von Bewegungstransfer zu erreichen, in Zukunft nur noch ein einzelnes Bild einer Zielperson, um diese in neuen Posen synthetisieren zu können. Eine Lösung, die Bewegung mittels eines GANs auf ein statisches Gesicht transferiert, ist das Dynamics Transfer GAN. Aktuell funktioniert die in dieser Forschungsarbeit vorgestellte Lösung lediglich auf sich in klaren Grenzen befindlichen Bewegungen wie Gesichtsausdrücken. Ein speziell auf die Grenzen eines Bereichs, wie z.B. ein Gesicht ihn darstellt, trainierter Diskriminator stellt sicher, dass an den räumlichen Eigenschaften eines Zielbereichs keine

Änderungen vorgenommen werden (Baddar et al., 2017, S. 1). Für Bewegungstransfer, welcher große Translationen im Bild impliziert, einfach anhand eines einzelnen Fotos einer Zielperson ist daher noch einige Forschung nötig. Für Künstler, die Cinemagraphen erstellen, könnte der in „Dynamics Transfer GAN“ vorgestellte Lösungsansatz jedoch bereits jetzt von Nutzen sein. Bei Cinemagraphen werden selektiv und durch kreative Willkür einzelne Elemente eines Bildes mit Bewegung belebt, wohingegen der Rest des Bildes statisch bleibt.

Unbeaufsichtigte Lernfortschritte durch LSGAN (Mao et al., 2017)**:** Bei tiefen neuronalen Netzen können im Training einige, teils noch nicht vollständig erklärbare Schwierigkeiten auftreten. Bei der GAN Netzwerkarchitektur kommt es beim Training des Diskriminators häufig zum Problem der verschwindenden Steigungen (engl. vanishing gradients). Das bedeutet, die Optimierung verlangsamt sich, da die mathematische Steigung in gewissen Wertebereichen der generierten Daten verschwindend gering wird (Mao et al., 2017, S. 2). Um dieses Problem zu behandeln, wird als abschließendes Element für die neuronale Netzwerkstruktur eines GANs spezifisch für den Lernprozess des Diskriminators eine neue Fehlerfunktion als Ersatz für die typischerweise verwendete Sigmoid Kreuzentropie Fehlerfunktion empfohlen. Die Methode der kleinsten Quadrate wird im Englischen „least squares method“ genannt. Daher der Name der Forschungsarbeit: „Least squares generative adversarial networks – LSGAN“. Die Hypothese lautet, dass bei klassischen unbeaufsichtigten GANs der Einsatz der Sigmoid Kreuzentropie die generierten Daten, die weit im positiven, richtigen Bereich liegen, jedoch dennoch weit vom echten Datensatz entfernt sind nicht richtig in der Fehlerfunktion gewichtet (Mao et al., 2017, S. 6). Die in LSGAN verbesserte Zielvorgabe für unbeaufsichtigte Lernaufgaben, setzt einen besseren Anreiz für den Generator Daten zu produzieren, die nahe am echten Datensatz liegen, indem weit im positiven Wertebereich liegende Daten „bestraft“ werden.

Recycle-GAN (Bansal et al., 2018)**:** Im Vergleich zu beaufsichtigten, lernenden Softwarelösungen wie „Everybody Dance Now“ gibt es auch Ansätze, wo eine Transferfunktion zwischen zwei Videos unbeaufsichtigt gelernt wird. Ein weiterer dieser Ansätze ist das Recycle-GAN. Das Recycle GAN kombiniert drei Fehlerfunktionen in ihrer GAN-Zielfunktion, um ohne vom Nutzer erstellte Labels, z.B. die Pose oder den Gesichtsausdruck von einer Person auf eine andere zu übertragen. Dabei werden ein wiederkehrender (engl. recurrent), ein gegenspieliger (engl. adversarial), sowie ein recycle Fehler in die finale Optimierungsfunktion für das Ausbilden der Netzwerkstruktur kombiniert (Bansal et al., 2018, S. 6). Die Recycle Funktion gewichtet besonders die räumliche Veränderung eines Bildinhalts in Abhängigkeit zur Zeit. Die zeitliche Komponente findet sich bei Video als Bildrate definiert. Durch die Zieldefinition, die sowohl

zeitliche wie auch räumliche Position im Bild beachtet, wird eine Anpassung zweier gleichwertiger Bildinhalte ohne Beaufsichtigung ermöglicht (Bansal et al., 2018, S. 5). Erst durch diese Angleichung ist ein Vergleich und Lernen eines Mappings zwischen zwei Videos ohne zusätzliche Labels machbar.

Video-to-Video Synthesis (T.-C. Wang et al., 2018b)**:** Im Vergleich zu dem bereits stärker beforschten Problem der Bildinhalt-zu-Bildinhalt Übertragung gibt es für die einzigartigen Probleme bei der Videoinhalterfassung und Videoinhaltgenerierung, wie zeitliche Kohärenz, noch weniger Forschungsarbeiten, auf denen aufgebaut werden kann. Der von einem Team merherer Forscher von NVidia Research und dem MIT Computer Science & Artificial Intelligence Laboratory gemeinsam entwickelte Prozess zur realistischen Generierung eines Videos anhand eines Quellvideodatensatzes und eines semantischen Labelvideos, wie er in dem Paper Video-to-Video Synthesis vorgestellt wird, liefert erstaunlich hohe generierte Videoquelität. Daher wird auf den Erkenntnissen dieser Arbeit häufig in weiterführender Forschung aufgebaut. Ein ähnlicher Ansatz zweier früheren Arbeiten (auf Einzelbilder beschränkt) bietet ein ebenso fruchtbringendes System, um neue Fragestellungen zu erarbeiten – kurz Pix2Pix (Isola et al., 2016) und Pix2PixHD (T.-C. Wang et al., 2017) genannt. Der Algorithmus von Video-to-Video Synthesis, folgend kurz Vid2Vid genannt, lernt mittels eines bzw. je nach Problem auch mehreren strukturierten Zusatzdatensätzen, wie Segmentationslabels oder Posenerfassungen, ein neues Video zeitlich kohärent zu generieren. Dabei ist das System auch erfolgreich darin, das Erscheinungsbild von der Bewegung der Szene zu trennen, wodurch der Nutzer z.B. den Straßenbelag in einer Stadtsequenz aus der Sicht eines Autos manuell wählen kann, indem er auf relativ einfache Weise die semantische strukturierte Zusatzinformation verändert. Im Paper werden Stadt-zu-Stadt, Pose-zu-Pose, sowie Gesicht-zu-Gesicht Experimente durchgeführt. Bei Pose-zu-Pose, was das Kernproblem von Bewegungstransfer darstellt, wie er in dieser Arbeit versucht wird, kommen mehrere Generatoren und Diskriminatoren zum Einsatz. Es werden semantische strukturierte Zusatzvideos generiert, wie bei Bewegungstransfer z.B. DensePose (Güler et al., 2018), OpenPose (Cao et al., 2018), und eine optische Flow Map Sequenz, welche kombiniert werden, um dem Generator der Ausgabesequenz als Hilfsmaterial zu dienen (T.-C. Wang et al., 2018a, S. 13). Eine Kombination aus mehreren Diskriminatoren für sowohl die Einzelbildqualität als auch die zeitliche Sequenzqualität des Videos ermöglicht es, im letzten Teil der Netzwerkoptimierungsfunktion echtes von halluziniertem Video nach jedem Aktualisierungsdurchgang besser zu unterscheiden (T.-C. Wang et al., 2018a, S. 4).

Few-shot Video-to-Video Synthesis (T. Wang et al., 2019)**:** Few-shot Video-to-Video Synthesis, folgend kurz als few-shot Vid2Vid bezeichnet, verspricht die Funktionalität von Vid2Vid zu erweitern. Wie auch für den in dieser Arbeit durchgeführten Bewegungstransfer ist auch bei der in Vid2Vid zum Einsatz kommenden Methode für jedes Video einer Zielperson, welche in neuen Posen synthetisiert werden soll, ein umfassender neuer Datensatz mit einer beträchtlichen Anzahl an Posen notwendig (T. Wang et al., 2019, S. 2). Der Grund dafür liegt darin, dass jedes neuronale Netzwerk von Neuem trainiert werden muss, um ein Mapping zwischen Posenerfassungen und Zielvideos zu lernen, mit welchem ein neues Video generiert werden kann. Few-shot Vid2Vid beschreibt eine Methode, wie ein auf eine Vielzahl von Trainingvideos trainiertes neuronales Netz mittels eines Aufmerksamkeitsmoduls so aktualisiert werden kann, dass es den aktuellen, aus nur wenigen Bildern bestehenden Zieldatensatz hochwertig generalisiert abbildet. Dabei schafft die Softwarelösung von Few-shot Vid2Vid dieses Unterfangen in mehreren Domänen, wie z.B. Bewegungs-, Gesichtsausdrucks- oder auch Straßenansichtstransfer (T. Wang et al., 2019, S. 1). Bei dem konkreten Beispiel des Bewegungstransfers wird dieses Ergebnis dadurch erreicht, dass mittels eines enormen Datensatzes von rund 1500 Tanzvideos von YouTube ein Basisnetzwerk trainiert wird, welches nur noch mit dem Aufmerksamkeitsmodul an die aktuelle Zielperson angepasst werden muss (T. Wang et al., 2019, S. 6). Obwohl unter anderem durch das Aufmerksamkeitsmodul ein erfolgreicher Bewegungstransfer mittels bereits weniger als zehn Bildern möglich ist, verbessert sich die Qualität des Transferergebnisses bei einer größeren Anzahl an Zielbildern. Auch ein größerer Datensatz für das ursprüngliche Training verbessert das Ergebnis (T. Wang et al., 2019). Wie auch im abschließenden Teil dieser Arbeit thematisiert, bietet der in Few-shot Vid2Vid präsentierte Ansatz auch für 2D Bewegungstransfer eine interessante Perspektive für weitere Forschung, da genau diese Einschränkung der geringen Posenanzahl bei gezeichneten Charakteren ein Hindernis darstellt. Jedoch wird in dem Paper auch darauf hingewiesen, dass z.B. computergenierierte Charaktere aktuell zu keinem guten Ergebnis führen, da sie stark von der Erscheinung der in dem Trainingsdatensatz der 1500 YouTube Tanzvideos vorkommenden Personen abweichen (T. Wang et al., 2019, S. 9).

7 Praktischer Bewegungstransfer

7.1 Methode und Aufbau des vergleichenden Versuchs

Elemente: Experimente und Versuche, die danach streben Ursache-Wirkungs-Beziehungen herzustellen, bestehen aus mehreren Schlüsselelementen. Diese sind vereinfacht zu beschreiben und unterteilen in Experimentalgruppe(n), Kontrollgruppe(n), sowie abhängige Variablen, unabhängige Variablen und kontrollierte Variablen (Koch et al., 2019). Die Unterscheidung zwischen Experimentalgruppe und Kontrollgruppe besteht in der unabhängigen Variablen, welche womöglich durch ihre Manipulation den Grad und die Art ihres Einflusses preisgibt. Die Veränderung, welche am Ergebnis der beiden Durchläufe entsteht, sollte dadurch auf diese unabhängige Variable zurückführbar sein. Die abhängige Variable beschreibt jene Effekte, die gemessen und beobachtet werden können bei und nach dem Durchlauf des Experiments. Die kontrollierte Variable stellt jene Aspekte des Versuchs und der Gestaltung der Gruppen, der Abläufe und der Versuchsumgebung dar, welche über die Dauer des Experiments konstant gehalten werden. Für die Aussagekraft der beobachteten Ergebnisse ist die korrekte Definition, wie auch die Einhaltung der Konstanz dieser kontrollierten Variablen von höchster Bedeutung (Koch et al., 2019, S. 59).

Zeichnung und Rendering: Die Leitschnur anhand derer ich meinen Versuch gestalte orientiert sich an diesen Anforderungen an ein wissenschaftliches Experiment. Ich vergleiche zwischen einem Experimentaldatensatz und einem Kontrolldatensatz. Der Experimentaldatensatz besteht aus einem zweidimensionalen gezeichneten Charakter, zusammengestellt in den planen Einzelbildern eines Videos, wohingegen der Entstehungsprozess der Einzelbilder des Charakters im zweiten Video, i.e. des Kontrolldatensatzes, auf dreidimensionalen Daten basiert. Den Unterschied zwischen den Ergebnissen des Versuchsdurchlaufs messe ich anhand einer Reihe von quantitativen sowie qualitativen Metriken.

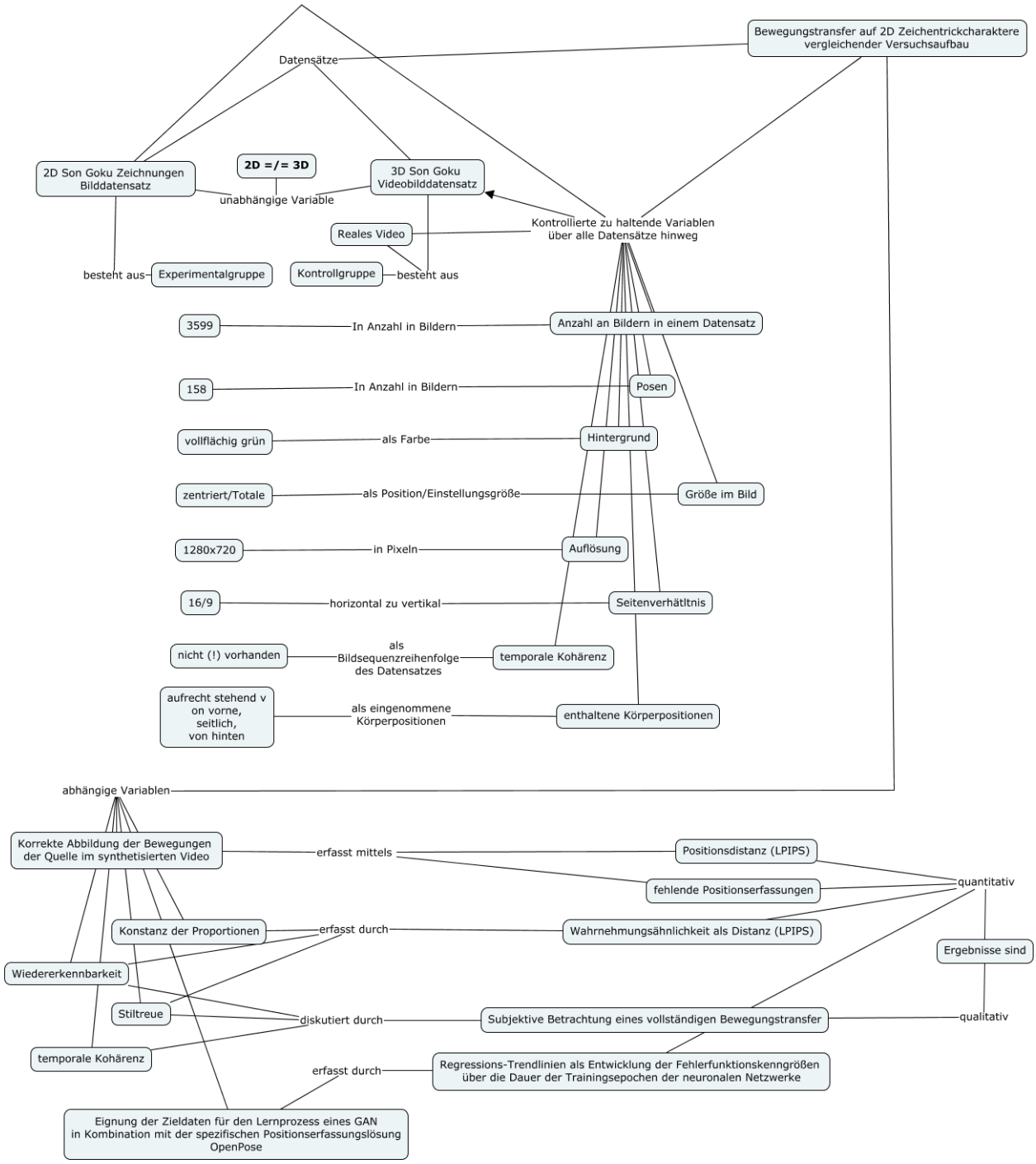


Abbildung 30: Gestaltung und Aufbau des vergleichenden Bewegungstransfers

7.2 Ausgangssituation

Fokus und Ausrichtung: Mein Ziel im praktischen Teil ist es, einen state-of-the Art Softwarealgorithmus zum Bewegungstransfer auszuwählen und auf seinen Nutzen in der zweidimensionalen Zeichentricksphäre zu überprüfen. Aktuelle Programme, wie „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020), schaffen Bewegungstransfer typischerweise zwischen zwei in monoskopischen Videos aufgezeichneten realen Personen. Eine davon dient dabei als Bewegungsquelle und die zweite als Bewegungsziel. Daraus ergibt sich für mich die Frage, ob dieselbe Funktionalität auch bei 2D gezeichnetem Material erhalten bleibt. Im Idealfall entsteht aus meinem Versuch ein erfolgreicher Transfer der Bewegung eines Tänzers oder einer Tänzerin auf einen zweidimensionalen gezeichneten Charakter. In jedem Fall soll dieser experimentelle Versuch jedoch ein besseres Verständnis über den aktuellen Stand des 2D Bewegungstransfer anhand einer konkreten maschinell lernenden Softwarearchitektur bieten. Dies bedeutet, dass ich Bewegungstransfer auf einen von mir zusammengestellten zweidimensionalen Datensatz bestehend aus diversen Zeichnungen eines Charakters anwende. Dabei nehme ich keine Änderungen an der Funktionsweise Softwarelösung vor.

Kontrolle: Um zwischen den generierten Bilddaten zu vergleichbaren Ergebnissen zu gelangen ist der Ablauf von der Auswahl der Daten für den Zieldatensatz, hin zum fertigen Bewegungstransfer so kontrolliert wie möglich gestaltet. Es wird lediglich eine Variable zwischen den zwei Testdurchläufen verändert. Konkret unterscheidet die zwei Datensätze, dass einer aus Zeichnungen und der andere aus computergenerierten Renderings eines Charakters besteht. Dies Konzeptionalisierung ist in der vorangegangenen Abbildung ersichtlich.

Elementkategorien: Um meinen praktischen Teil durchzuführen benötige ich Elemente aus drei Kategorien. Dazu zählen die Daten für den gezeichneten wie auch den computergenerierten **Charakter** des Bewegungsziels, die Daten für die Quelle der **Bewegung**, und die **Programme** zur Vorbereitung der Daten, zur Durchführung des Experiments, sowie zur weiterführenden Handhabung der gewonnenen Ergebnissequenzen. Welche Daten und Programme ich gewählt habe und eine Begründung der Entscheidungen beschreibe in den folgenden Kapiteln.

7.3 Daten Bewegungsziele

Textur und neuronale Hierarchien: Die Auswahl des Bewegungsziels – eines geeigneten zweidimensionalen gezeichneten Charakters – ist bereits entscheidend für den möglichen Erfolg des Bewegungstransfers. Daher gilt es hier, auf einige Punkte zu achten, welche besonders rückblickend nach der Auswertung der Ergebnisse klar werden. Das Softwaresystem „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) verwendet den Posenerfassungsalgorithmus OpenPose. Dieser wird bereits als fertig trainierter Algorithmus zur Erfassung der Pose von Menschen in realen Bildern und Videos in das Softwaresystem integriert. Da im Trainingsprozess des GANs zu jedem Einzelbild des Zieldatensatzes eine korrekte dazugehörige extrahierte Pose als Skelett-label vorhanden sein soll, ist die erfolgreiche Erfassung der Pose des gezeichneten Charakters von zentraler Bedeutung. Der Posenerfassungsalgorithmus OpenPose funktioniert durch die Extraktion und Aufbereitung der Posendaten in mehreren Stufen. Besonders relevant ist dabei die besondere die neuronale Architektur, die im ersten Schritt nach Körperteilen im Bild sucht. Diese basiert, wie viele andere Lösungen aus dem Bereich Computer Vision, auf Faltungscodierung (Cao et al., 2018, S. 3). Wie in dem Kapitel zu neuronalen Strukturen bereits beschrieben, sind faltungscodierte neuronale Netzwerke (CNNs) besonders geeignet, um Muster und Regelmäßigkeiten in einem Bild zu erfassen. Dies führt jedoch dazu, dass eine Hierarchie entsteht zwischen strukturellen Eigenschaften eines Bildes, wie Haut-, Kleidungs- oder Haartexturen, und abstrakteren Konzepten, wie Umrissen und für Menschen klar abgrenzbaren Einheiten wie „eine Person“ (Geirhos et al., 2019). Auf der Erkenntnis aufbauend, dass der Textur eines Subjekts bei faltungscodierten neuronalen Netzwerken eine hohe Relevanz zukommt, stellt sich die Frage, wie robust der Algorithmus OpenPose gegenüber im Originaltrainingsdatensatz vermutlich nicht enthaltenen Strukturen, Texturen und Körperformen ist. Die Vielfalt an Körperformen und die detaillierte Ausgestaltung der Textur von gezeichneten 2D Charakteren und ihrer Haut, Kleidung, oder Haare ist so vielfältig wie die Kreativität ihrer Erschaffer. Auch die bei der Erstellung verwendeten Werkzeuge spielen eine Rolle. Ist ein Charakter rein digital gezeichnet, weist eine Farbfläche im final präsentierten Bild eine andere Struktur auf, als wenn auf Papier gezeichnet wurde.

7.3.1 Evaluierung der Posenerfassung auf gezeichneten Charakteren

OpenPose – Standalone: Konkret bedeutet dies für die Auswahl eines geeigneten Zieldatensatzes, dass die Eignung eines gezeichneten Charakters für Posenerfassung mittels OpenPose einigen Anforderungen unterliegt, welche in ihrem Einfluss nicht genau abzugrenzen sind. Zwei Vermutungen liegen jedoch nahe: Die erste Annahme lautet, dass aufgrund der faltungscodierten Struktur des neuronalen Netzwerks von OpenPose ein Bild mit viel Textur einem Bild mit wenig Textur überlegen ist. Die zweite Annahme ist, dass ein Charakter je näher er an der menschlichen Form konzipiert wurde, einem abstrakteren, von der menschlichen Form weiter abweichenden Charakter überlegen ist. Für die endgültige Auswahl eines geeigneten Charakters bleibt für die Beurteilung im Rahmen dieses praktischen Teils jedoch nur ein Testdurchlauf mit dem separat betrachteten Algorithmus der Standalone Version von OpenPose (Hidalgo et al., 2017/2020). Da die Entwickler der Softwarelösung OpenPose diese nicht nur als Bibliothek für Python zur Verfügung stellen, sondern auch als alleinstehendes Programm habe ich diesen Weg zur Entscheidungsfindung gewählt. Das Programm kann für nicht kommerzielle Zwecke für die Verwendung unter Windows auf www.github.com heruntergeladen werden (Hidalgo et al., 2017/2020). Für den Testdurchlauf ist der im Abschnitt „Quick Start“ der README.md von OpenPose beschriebene Ablauf relevant.

Beliebtheitsranking: Um einen Überblick zu gewinnen, welche zweidimensional gezeichneten Charaktere geeignet sein könnten für Bewegungstransfer, bediene ich mich eines Rankings der 150 beliebtesten Cartooncharaktere, wie es von Nutzern der Website www.ranked.com erstellt wurde (Hahn, 2020). Da in dieser Liste auch animierte 3D Charaktere wie Shrek oder Woody aus Toy Story vorkommen, überspringe ich diese und wähle jeweils den nächsten Charakter. Dadurch ergeben sich aus 115 Einträgen der Liste 100 zu evaluierenden Charaktere. Der Grund, weshalb ich nicht eine zufällige Auswahl an Zeichentrickcharakteren für diese Evaluierung heranziehe, sondern die 115 beliebtesten, liegt darin, dass ich für den Zieldatensatz möglichst viele verschiedene Posen eines Charakters benötige. Dadurch entsteht die Herausforderung, einen Charakter zu wählen, der sowohl vielversprechend für den Bewegungstransfer als auch für die Verfügbarkeit von ausreichend Zeichnungen in Ganzkörperansicht ist. Je beliebter ein Charakter, so die Annahme, desto mehr Originalzeichnungen als auch Fan-Art werden online zur Verfügung stehen, um daraus einen Datensatz zusammestellen zu können. Alternativ würde sich auch die Möglichkeit bieten, selbst einen Charakter in einer Vielzahl von Posen zu zeichnen. Da dies für die Bewertung der grundlegenden Funktionalität von

Bewegungstransfer auf gezeichnete Charaktere jedoch nicht unbedingt notwendig ist, sehe ich davon bei diesem ersten Erarbeiten der Machbarkeit aus Zeitgründen ab.

Sammlung und Aufbereitung der Zeichnungen: Mit dieser Liste an potenziellen Charakteren mache ich pro Charakter eine Suchanfrage auf images.google.com. Idealerweise ist der gezeichnete Charakter ohne verdeckte Körperteile in einer relativ neutralen Pose zu sehen. Diese willkürlich getroffene Auswahl eines Einzelbildes speichere ich dann lokal auf meinem Rechner ab. Nachdem ich auf diese Art 100 Einzelbilder gesammelt habe, füge ich diese zu einem Video mit den Abmessungen 1080x1080 Pixel zusammen. Dieses Video, wo jedes Einzelbild einem Charakter der Liste entspricht, übergebe ich dann an die Software OpenPose. Über den folgenden Befehl starte ich die Posenerfassung und Speicherung der Einzelbilder auf schwarzen Hintergrund sowie überlagert über das Originalbild mittels der Windows Kommandozeile.

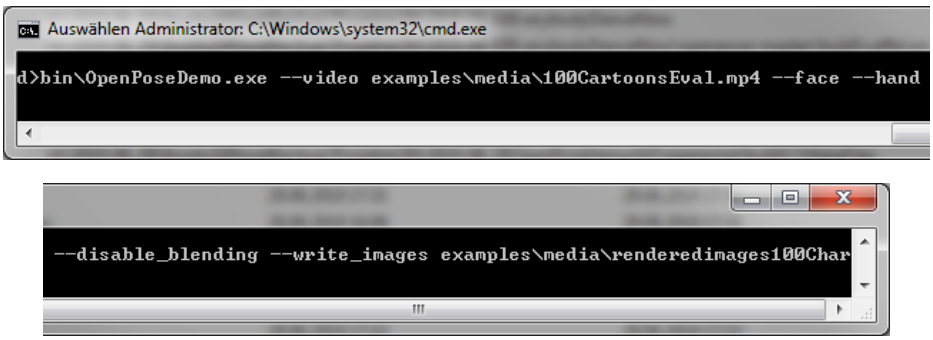
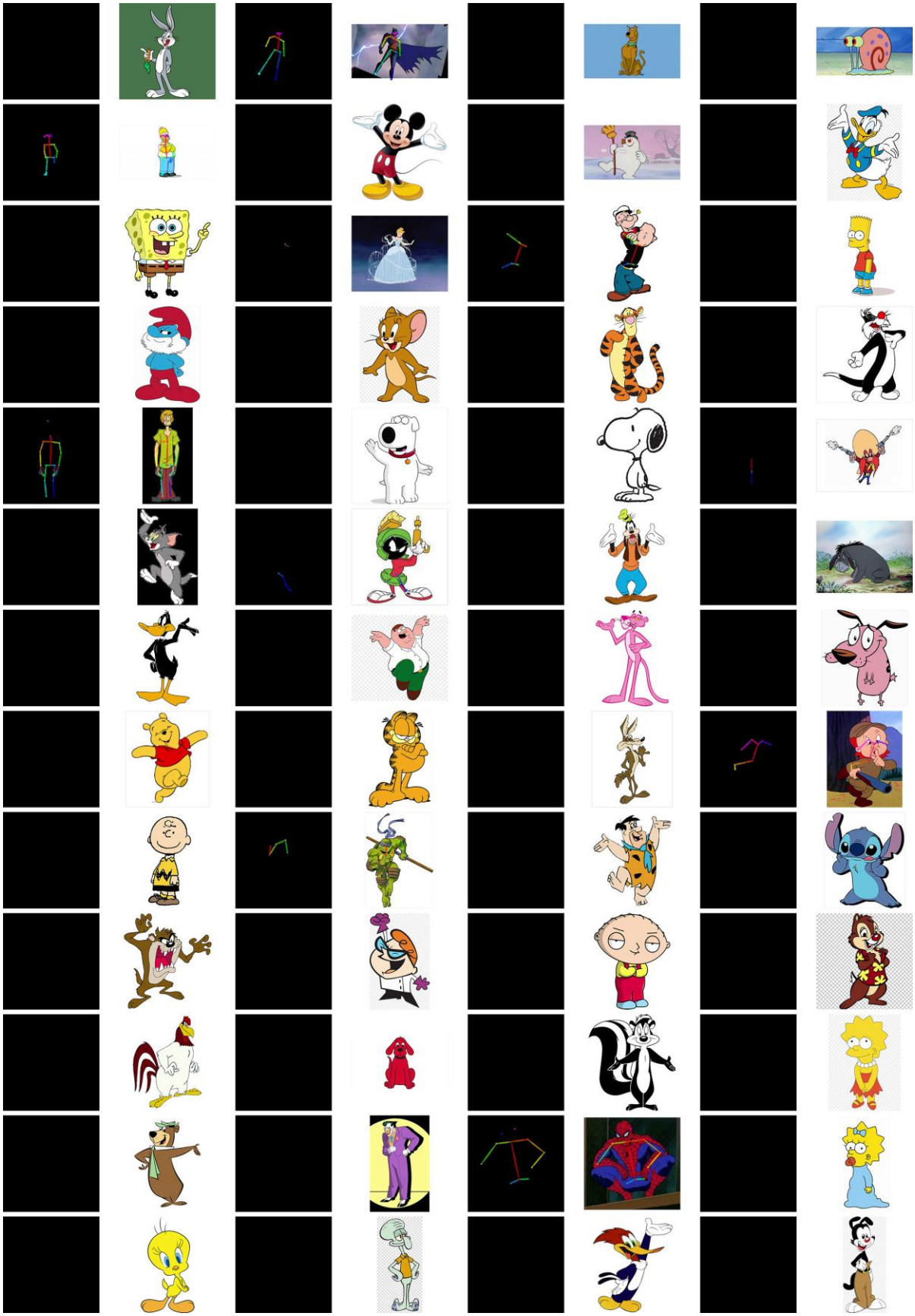


Abbildung 31: Befehl zur Evaluierung der 100 gezeichneten Charaktere mittels OpenPose (Hidalgo et al., 2017/2020) gestartet über die Windows Kommandozeile

Auf Schwarz: Um die erfasste Pose ohne Hintergrund auf schwarz zu erhalten, ist es es notwendig die binäre Statusvariable (engl. flag) „–-disable_blending zu deaktivieren. Auf diese Weise speichert das Programm automatisiert sowohl eine überlagerte als auch die auf schwarz dargestellte Posenerfassung für jeden Charakter ab. Weitere Einflussmöglichkeiten auf das Testprogramm von OpenPose sind in dem Dokument „README.md“ unter "Flags", wie es im Github Verzeichnis zu finden ist, nachzulesen (Hidalgo et al., 2017/2020).

Darstellung: In der folgenden Tabelle habe ich das Ergebnis dieser Posenerfassungen dargestellt. Zuerst ist die Posenerfassung als Skelett auf schwarz zu sehen, gefolgt von der auf das Originalbild überlagerten Pose.



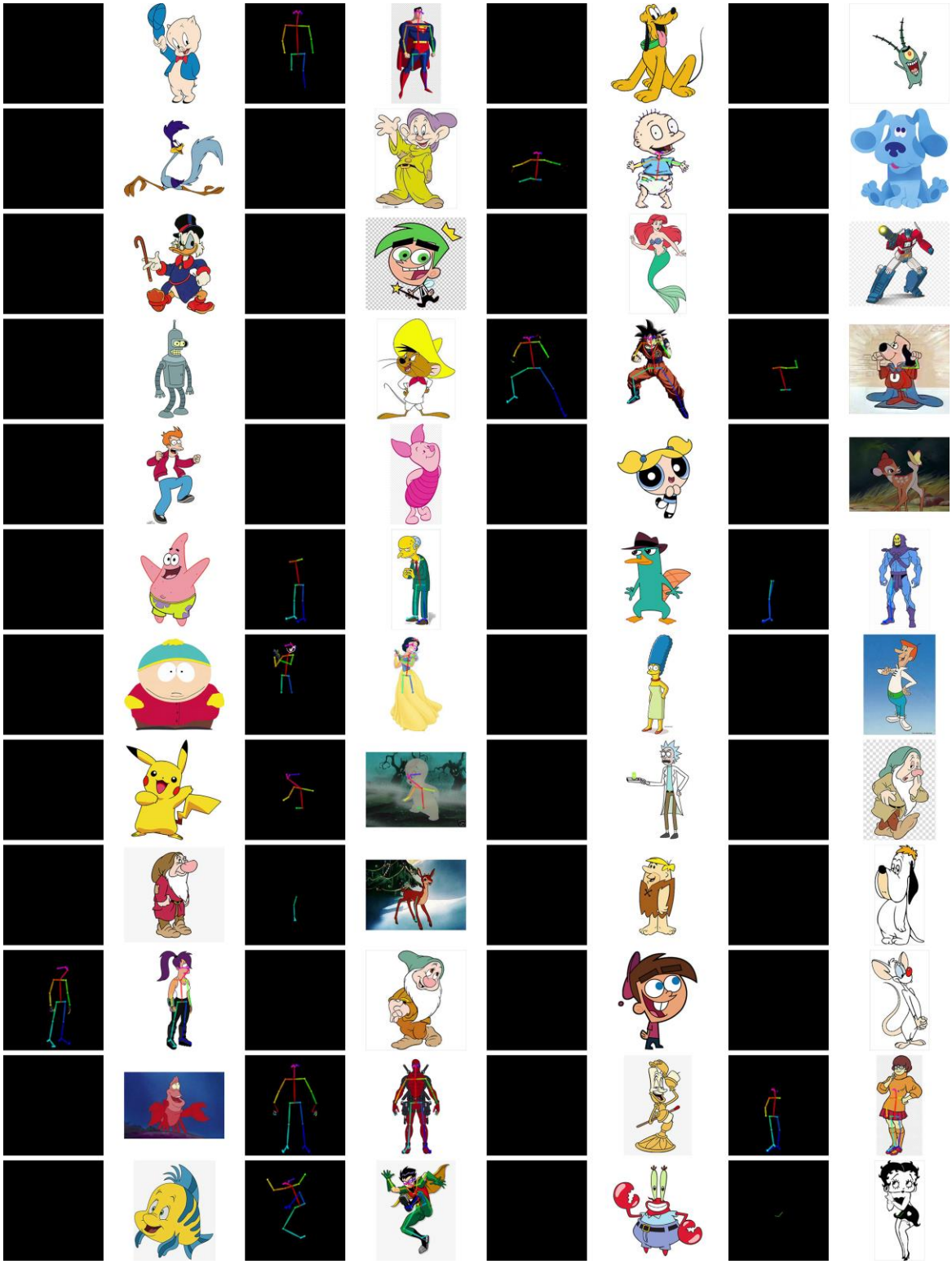


Abbildung 32: Gegenüberstellung von erkannter Pose auf schwarz, sowie überlagert auf die evaluierte Originalzeichnung von 100 der 115 beliebtesten Zeichentrickcharaktere, wie sie auf Ranked.com zu finden sind (Hahn, 2020)

Daran ist bereits augenscheinlich zu erkennen, dass ein humanoider Charakter mit Details und Textur in der Ausgestaltung des Körpers und der Kleidung größeres Potential hat, von dem Algorithmus korrekt erkannt zu werden. Ausgewertet ergibt die hier durchgeführte Evaluierung jedoch eine insgesamt schlechte Eignung von Zeichentrickcharakteren für den Posenerfassungsalgorithmus OpenPose. Die folgenden Ergebnisse sind eingeteilt in Charaktere ohne erfasste Pose, Charaktere mit teilweise erfasster Pose und Charaktere mit vollständig und korrekt erfasster Pose. Als vollständig werte ich auch eine Erfassung, welche die Pose trotz fehlender Detailerfassungen, wie z.B. der Finger, korrekt und gänzlich erkennbar abbildet.

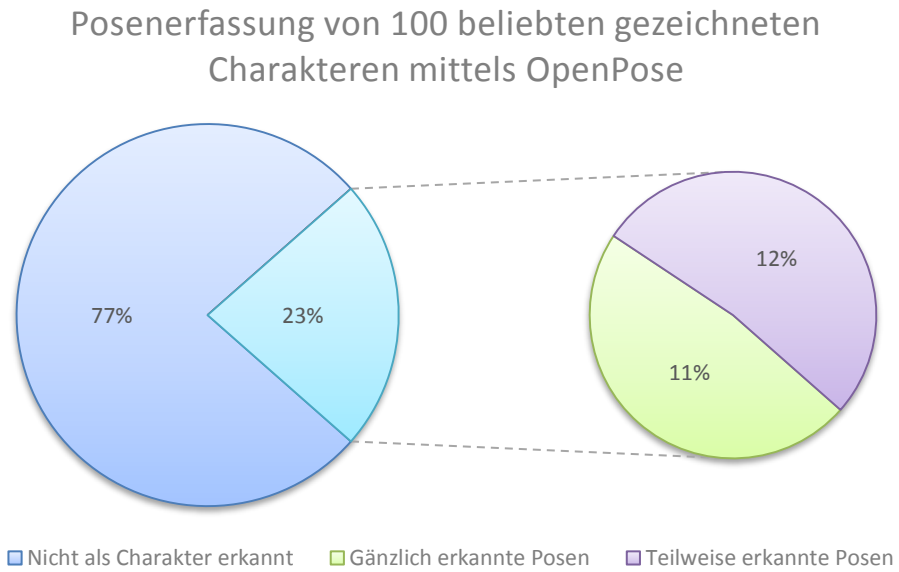


Abbildung 33: Zu vorangegangenen Abbildung gehörende Aufschlüsselung der erkannten und nicht erkannten Posen der evaluierten 100 Zeichentrickcharaktere

Einzelfallbasierte Einschätzung: Dass lediglich in 23 Fällen, knapp einem Viertel aller Beispiele, überhaupt ein Charakter erkannt wurde verdeutlicht, dass OpenPose nur sehr eingeschränkt zur Posenerfassung von gezeichneten Charakteren nützlich ist. Daher bedarf es für jeden Charakter einer neuen Beurteilung, ob dieser für Bewegungstransfer in Frage kommt.

Animeheld: Da Son Goku, Hauptcharakter der Dragon Ball Manga und Anime Serie, ein vielversprechendes Posenerfassungsergebnis liefert und auch bei einer Google Suche auf viele Bildeinträge stößt, wähle ich diesen Charakter für den weiteren praktischen Versuchsdurchlauf.

7.3.2 Zusammenstellung der Zieldatensätze

Animationssequenz und Einzelzeichnung: Für die Zusammenstellung des Experimentaldatensatzes von gezeichneten Ganzkörperansichten des Animecharakters Son Goku kommen für mich drei Quellen von Einzelbildern in zwei Kategorien in Frage. Die erste Kategorie sind einzelne, gezeichnete Posen, wie sie sowohl von dem Produktionshaus der Originalserie (Bird Studio/Shueisha Inc.) als auch häufig von Fans und unabhängigen Künstlern angefertigt und veröffentlicht werden. Diese einzelnen Zeichnungen sind häufig gekennzeichnet durch hohe Qualität und Detailauflösung und legen dabei eher selten Wert auf den Hintergrund. Die zweite Kategorie sind Einzelbilder eines Videos oder Gifs, in denen Son Goku in Totalansicht vorkommt. Diese – obwohl spärlich vorhanden – bieten den Vorteil, dass mehrere Bilder und dadurch Posen in den Sequenzen enthalten sind und diese auch einen in seiner Erscheinung konstant gezeichneten Charakter zeigen, da dieser immer auf die gleiche Art gezeichnet wurde. Jedoch ist hier auch der Nachteil gegeben, dass besonders bei dem .gif Format, die Auflösung und der Detailreichtum keinen hohen Ansprüchen genügen. Außerdem ist der Anime Hauptcharakter häufig in Aktion und Interaktion mit der Umgebung und anderen Charakteren zu sehen, wodurch es zu Verdeckungen kommt.

*Abbildung 34: Ergebnis einer Onlinesuche nach Kurzvideos des Charakters Goku auf der Webseite www.giphy.com (*GIPHY*, 2019)*

Hintergrund von 2D Zeichnungen: Da ich einen wechselnden Hintergrund in der später zum Einsatz kommenden Version der Softwarelösung „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) nicht verwenden kann, setzen sich die erstellten Datensätze zu großem Teil aus

gezeichneten .png (portable network graphics) Einzelzeichnungen zusammen. Bilder im Format .png kann ich dank ihrer häufig vorhandenen Alphaebene, ohne den Charakter manuell freistellen zu müssen, vor einen grünen Hintergrund setzen. Dies geschieht im Compositingprogramm „Adobe After Effects CS6“. Sofern sich der grüne Hintergrund auch im fertigen Bewegungstransfer findet, ist dadurch eine spätere Verwendbarkeit in vielfältigen Szenerien gegeben. Der Prozess des gezielten Löschens eines Hintergrunds oder auch anderer Bildelemente nach Pixelfarbwert nennt sich Chroma Keying.

Zieldatensatzeigenschaften: Für die folgende Auflistung der Datensätze ziehe ich diese Eigenschaften heran

- Gesamtanzahl der Bilder

- Anzahl der verschiedenen Posen

- Gleichmäßigkeit der enthaltenen Erscheinungsformen

- Hintergrund

Drei Datensätze: Im Laufe meines Versuchs verwende ich drei verschiedene Datensätze in der Experimentalkategorie, der eines gezeichneten Son Goku. Der erste Datensatz mit dem Namen „GokuGiphy“ besteht aus Video- und Gifsequenzen, welche den Zusammenhang von Hintergrund und Charakterpose veranschaulichen. Des Weiteren erlaubt dieser Datensatz nach dem Durchlaufen des Bewegungstransferprozesses erste Annahmen zu zeitlicher Kohärenz.

Abbildung 35: Einzelbild des Videodatensatzes „GokuGiphy“

Als zweiter Datensatz dient ein auf den Erkenntnissen der ersten Versuchsdurchläufe basierender neuer, vergrößerter Datensatz, der sich ausschließlich aus Einzelbildern zusammensetzt. Bei dem vergrößerten Datensatz

kommt die Eingangs erwähnte Alphaebene der .png Dateien zum Einsatz. So ist der Charakter im Datensatz „PNGGokuWiggle“ stets ohne Hintergrund vor einer grünen Farbfläche abgebildet. Um diesen Datensatz zu vergrößern, nehme ich bei der Zusammensetzung wenig Rücksicht auf die Gleichmäßigkeit der Erscheinung bzw. der Art der Erstellung als CG Rendering oder als Zeichnung. Des Weiteren werden die enthaltenen 150 Posen durch ein zufälliges Versetzen der wiederholten Bilder im Raum des Videos auf 1628 Gesamtbilder im Datensatz erweitert. Dies wird erreicht durch die Expression „wiggle()“ für die Attribute Rotation, Position, und Skalierung im Compositingprogramm Adobe After Effects CS6. Anhand der Ergebnisse mit diesem Datensatz lassen sich Annahmen über den Einfluss verschiedener Erscheinungen von Goku im Zieldatensatz auf die Gleichmäßigkeit des generierten Bewegungstransfers treffen. Dazu zählt z.B. die wechselnde Haarfarbe. Des Weiteren bietet dieser Datensatz die Basis für die Initialisierung des Hauptexperiments, da das dazugehörige Netzwerk über 220 Epochen trainiert wird.

Abbildung 36: Gegenüberstellung von drei der 150 in „PNGGokuWiggle“ enthaltenen vielfältigen Erscheinungsbilder des Charakters Son Goku

Goku2D158: Der dritte und letzte Datensatz, welcher auch als finaler Experimentaldatensatz für den vergleichenden Versuch dient, baut auf den Erkenntnissen aus den ersten Durchläufen auf. So setzt sich dieser Datensatz zusammen aus Zeichnungen, die in ihrer Erscheinungsform Son Goku relativ konstant zeigen. Außerdem wird auf Bilder, sowohl aus Einzelbild- als auch aus Bildsequenzquellen (Videos) zurückgegriffen. Da in den Videos jedoch ein Hintergrund und manchmal auch weitere Elemente neben dem gesuchten Charakter Son Goku zu sehen sind, entferne ich diese teilautomatisiert mittels des Onlinetools der Webseite www.remove.bg (Kaleido AI GmbH, 2020).

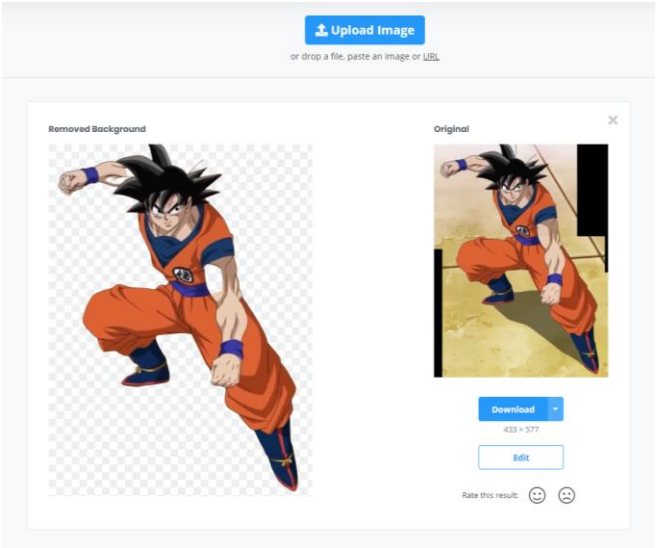


Abbildung 37: Automatisch entfernter Hintergrund mittels des Onlinetools „remove.bg“ (Kaleido AI GmbH, 2020)

Rote Haare: Ein weiterer von mir unternommener Schritt, damit eine Anzahl von 158 Posen eines relativ gleichbleibend erscheinenden Son Gokus erreicht werden kann, besteht darin, die Haarfarbe einiger Zeichnungen, auf denen Son Goku mit roten Haaren zu sehen ist, während der finalen Zusammenstellung des Datensatzes in Adobe After Effects CS6 zu ändern. Dies gelingt mittels Masken und des Effekts „In Farbe ändern“ und ist in der folgenden Abbildung dargestellt.

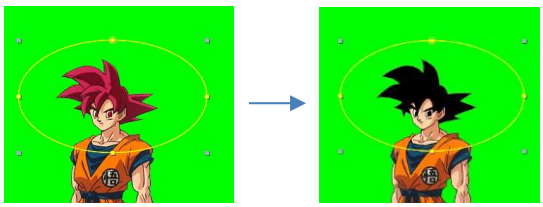


Abbildung 38: Anpassung der Haarfarbe für den Experimentaldatensatz „Goku2D158“

Erste Beobachtungen: Die Probleme und Erkenntnisse aus den Durchläufen mit den jeweiligen Datensätzen werden im Abschnitt Ergebnisse genauer beleuchtet. Wie jedoch bereits kurz angerissen, treten in den verschiedenen Durchläufen mit den gezeichneten Zieldatensätzen besonders Probleme betreffend der Generalisierungsfhäigkeit des Generators auf neue Posen, eines falsch gelernten Hintergrunds, sowie bezüglich eines konstant bleibenden Erscheinungsbilds des Charakters Son Goku auf.

Hauptdatensatz: In der Kontrollgruppe kommen ebenso drei verschiedene Datensätze zum Einsatz. Primär relevant für das Experiment des Vergleichs zwischen den auf 2D Zeichnungen und 3D Renderings basierenden Bildern des

Charakters Son Goku ist der Datensatz Goku3D158. Dieser ähnelt in den von mir kontrollierbaren Eigenschaften dem Datensatz Goku2D158. So besteht er ebenfalls aus 158 zufälligen Posen, welche auf dieselbe Art auf eine Gesamtanzahl von 3599 Bildern erweitert werden wie Goku2D158. Die Quelle für diesen Zieldatensatz bietet ein YouTube-Video, welches eine 3D Animation des Charakters Son Goku vor grünem Hintergrund zeigt (*Free green screen videos, visual effects, dragonball z, goku, chroma key, vfx, 3d animation, 4K, hd*, 2019). Um die Eignung dieses Datensatzes für die Posenerfassung zu überprüfen, habe ich auch diesen mit der Standalone Version von OpenPose betrachtet (Hidalgo et al., 2017/2020).

Abbildung 39: Einzelbild aus dem Datensatz Goku3D158 bei der Evaluierung der grundsätzlichen Eignung für Bewegungstransfer mittels OpenPose (Hidalgo et al., 2017/2020)

Goku3DFull: Um in einem weiterführenden Versuch eine Annahme in Bezug auf den Einfluss der Anzahl an Posen im Zieldatensatz treffen zu können, kommt ein zweiter, in seiner Posenanzahl nicht reduzierter Datensatz mit erhaltener korrekter zeitlicher Abfolge der Posen aus derselben Videoquelle zum Einsatz. Dieser Datensatz enthält bei 3599 Bildern auch dieselbe Anzahl an Posen.

Reales Video: Abschließend kommt in der Kontrollgruppe zur Veranschaulichung der Leistungsfähigkeit des hier verwendeten Softwaresystems ein Datensatz bestehend aus realem Videomaterial einer echten Person zum Einsatz. Dieses Video wurde im Rahmen eines früheren Projekts gemeinsam mit Emil Bauer und Sebastian Enk im Videostudio der FH St. Pölten aufgezeichnet. Auch dieser Datensatz wird auf 158 zufällige Posen reduziert, welche im Anschluss mittels einer „Wiggle()“ Expression in Adobe After Effects CS6 auf 3599 Bilder erweitert werden.

Abbildung 40: Zufällig rotiertes Einzelbild des Datensatzes SebLA158 (kurz für Sebastian Live-Action in 158 Posen)

Die folgende Tabelle schlüsselt alle im praktischen Teil zum Einsatz kommenden Datensätze nach den in diesem Kapitel eingangs erwähnten Eigenschaften auf.

Zieldatensätze	*Goku Giphy*	*PNGGoku Wiggle*	*Goku 2D158*	*Goku 3D158*	*Goku 3Dfull*	*Seb LA*
Bilderanzahl	1015	1628	3599	3599	3599	3599
Posenanzahl	148	150	158	158	3599	158
Gleichmäßigkeit der Erscheinungsformen	Trifft nicht zu	Trifft nicht zu	Trifft großteils zu	Trifft zu	Trifft zu	Trifft zu
Hintergrund	variiert	grün	grün	grün	grün	grün*

*Tabelle 4: Auflistung der Eigenschaften der zum Einsatz kommenden Zieldatensätze; *mit Textur*

7.4 Daten Bewegungsquellen

Eignung als Quelle: An die Bewegungsquellen werden weniger Anforderungen gestellt als an die Bewegungsziele. Dennoch ist die Einhaltung dieser Anforderungen entscheidend für den Erfolg des Bewegungstransfers. Grundsätzlich ist es möglich, die Posen einer jeden in einem Video klar vom Posenerfassungssystem OpenPose erkennbaren Person zu verwenden. Um die Qualität der Bewegungsübertragung besser beurteilen zu können, sind jedoch ausladende und vielfältige Bewegungen von Vorteil. Tanz ist hier nicht nur als Namensgeber für das Paper „Everybody Dance Now“ (Chan et al., 2018) nützlich, sondern erfüllt auch hohe Ansprüche an die Veranschaulichung der Leistungsfähigkeit des Softwaresystems. Des Weiteren liefert Tanz als Bewegungsquelle ein Plus an Unterhaltungswert. Dieser ist zwar nicht zwingend notwendig, dient aber dennoch der Motivation und der einfacheren Kommunikation der Ergebnisse.

Anatomische Ähnlichkeit: Ziel ist es, die Bewegungen der Quelle vollständig als Posen zu erfassen und in einem repräsentativen Skelett abzuspeichern. Die erfasste Bewegung sollte frei von Störungen, wie Wacklern oder Sprüngen, sein. Des Weiteren ist es von Vorteil, wenn die Person der Bewegungsquelle nicht stark von der Anatomie und Position der Person bzw. des gezeichneten oder gerenderten Charakters im Bild des Bewegungsziels abweicht. Dies verringert die möglichen Probleme durch den dadurch nicht mehr zwingend notwendigen Normalisierungsschritt.

Vergleichbarkeit: Der Hauptdatensatz, welcher über alle Versuche hinweg als Bewegungsquelle und somit als Basis für die subjektive und qualitative Bewertung des Bewegungstransfer dient, ist das Musikvideo: "That’s What I Like" von Bruno Mars (Mars & Lia, 2017). Diese Bewegungsquelle wähle ich deshalb, weil sie sowohl bei Versuchen und Veranschaulichungen in z.B. dem zum Paper „Everybody Dance Now“ gehörenden gleichnamigen Youtubevideo (Chan, 2018) als auch bei auf diesem Paper aufbauenden Software- und Forschungsprojekten zum Thema Bewegungstransfer zum Einsatz kommt. Eine Suchanfrage nach Projekten, die auf „Everybody Dance Now“ verweisen, auf der zum Teilen von Codesammlungen und Programmen beliebten Webseite www.github.com ergibt fünf Einträge über öffentlich verfügbare Programmverzeichnisse zu diesem Thema (GitHub, 2020). Dadurch ermöglicht die Häufigkeit der Wahl des Musikvideos von Bruno Mars als Bewegungsquelle eine qualitative Einschätzung, wie sich der 2D Bewegungstransfer mit anderen Zieldatensätzen und Softwarelösungen visuell vergleicht.

Weitere Quellen: Des Weiteren wähle ich als Bewegungsquellen für meine Versuche sowohl Videos von Tänzern wie sie auf der Webseite www.YouTube.com zu finden sind als auch selbst aufgezeichnete Videos. Die verschiedenen verfolgten Fragestellungen werden im Kapitel Ergebnisse kurz besprochen. Folgende Auflistung gibt einen Überblick über die von mir verwendeten Bewegungsquelldatensätze:

Bewegungsquelle *Videotitel*	*Bruno Mars Musik-video*	*100 Walks*	*Mr. Bean - Bombastic*	*Auf-nahme Break-dance*	*Auf-nahme Parkour 1 Präzi*	*Auf-nahme Parkour 2 AFF*
Anzahl Posen/Bilder	5262	5419	750	877	1277	1987
Große Translation	x		x			x
Stark variierende Skalierung im Bild	x				x	x
Überkopf-bewegungen				x	x	x
Verdeckungen					x	x

Tabelle 5: Zum Einsatz kommende Bewegungsquellen und ihre Herausforderungen (x = trifft zu)

Die ersten drei Spalten der Tabelle enthalten Webvideos. Im Vergleich dazu finden sich in den letzten drei Spalten von mir aufgezeichnete Bewegungsquellvideos.

7.5 Codesammlung „Everybody Dance Now – reproduced in pytorch“

Zugang und Verfügbarkeit: Der Originalcode von „Everybody Dance Now“, wie ihn das Team von der UC Berkeley in ihrem gleichnamigen Paper beschrieben hat, ist zwar zu Forschungszwecken verwendbar, jedoch nicht direkt zugänglich (Chan et al., 2019). Die Idee, einem zweidimensionalen Charakter relativ einfach Leben und Ausdruck durch Bewegung einzuhauchen, steht im Zentrum dieser Arbeit. Der Stand der Forschung in den Bereichen Computer Vision und maschinelles Lernen erlaubt den Test vielfältiger Lösungsansätze ohne hohe Kosten und organisatorischen Aufwand. Webseiten wie www.github.com ermöglichen den einfachen Austausch von Programmen und Quellcode sowie ein kollektives Finden von Fehlern und Überwinden von programmspezifischen Herausforderungen. Wie im vorangegangenen Kapitel bereits kurz angesprochen, ergibt eine Suche auf der Website www.github.com fünf frei zugängliche Projekte, die auf den Erkenntnissen der Forschungsarbeit der UC Berkeley aufbauen (GitHub, 2020). Darunter findet sich auch das Projekt "EverybodyDanceNow - reproduced in pytorch" (Wu et al., 2018/2020). Dieses wurde programmiert und geteilt von Forschern und Studierenden an der chinesischen Universität Hongkong/Shenzhen – kurz CUHKSZ.

Funktionsumfang: Der Unterschied der hier verwendeten Reproduktion und der Orignalarbeit von Chan et. al. liegt in dem Umfang der Funktionalität. So fehlt beispielsweise bei der Reproduktion die Möglichkeit der zeitlichen Glättung (Wu et al., 2018/2020). Obwohl dieses Projekt nicht den vollen Funktionsumfang der in der Forschungsarbeit "Everybody Dance Now" vorgestellten Lösung umfasst, ist dieses dennoch geeignet für das in diesem praktischen Teil unternommene Unterfangen der Evaluierung des 2D Bewegungstransfers. Der Grund dafür liegt darin, dass eine temporale Kohärenz im gezeichneten Zieldatensatz kaum vorhanden ist. Dieser besteht aus einer zeitlich großteils nicht zusammenhängenden Bildsequenz. Daher kann eine logische zeitliche Abfolge in diesem Fall auch garnicht von einem neuronalen Netzwerk abgebildet werden. Es gilt jedoch anzumerken, dass für die extrahierten Posen aus der Bewegungsquelle eine zeitliche Abhängigkeit in den Quelldaten besteht, i.e. jene der Bewegung. Diese wird idealerweise in der Sequenz der Posenerfassungenen auch korrekt abgebildet. Auf zeitliche Kohärenz optimierte realistische Posenerfassungen würden nach den Erkenntnissen aus der Forschungsarbeit „Everybody Dance Now“ vermutlich auch das generierte Endergebnis positiv beeinflussen (Chan et al., 2018, S. 7). Wo eine weiterführende zeitliche „Glättung“ von Nutzen sein könnte, wird an späterer Stelle in der Beschreibung der Ergebnisse erörtert.

MIT-Lizenz: Der Code, welcher an der chinesischen Universität HongKong/ Shenzhen erstellt wurde, ist kostenlos auf www.github.com zum Download verfügbar (Wu et al., 2018/2020). Dieser unterliegt der MIT Lizenz, was bedeutet, dass der Code sowohl bearbeitet, verbreitet, sowie privat als auch kommerziell genutzt werden darf. Dies ist gestattet bei der erfüllten Voraussetzung, dass auf das originale Copyright hingewiesen wird und eine Kopie der Lizenzdatei dem Code beigefügt ist. Für die Funktionalität wird dabei von den Inhabern des Copyrights keine Garantie übernommen (*The MIT License | Open Source Initiative*, 2020). Der für diesen Test verwendete Code findet sich mit dieser Lizenzdatei auf dem Datenträger am Ende dieser Diplomarbeit.

7.5.1 Soft- und Hardwareumgebung

Vorbereitungen: Um das Programm in der mir vorliegenden Form verwenden zu können, bedarf es einiger Vorbereitungen. Dazu zählen mitunter die Auswahl des Betriebssystems, des Grafikkartentreibers, sowie der Versionen der Bibliotheken, um den Code optimiert auf dem Grafikprozessor laufen lassen zu können. Eine genaue Liste mit Anforderungen liefert aus dem Dokument "README.md" die folgende Auflistung (Wu et al., 2018/2020):

Minimalanforderungen/im Original verwendete Versionen

- Ubuntu 16.04
- Python 3.6.5
- Pytorch 0.4.1
- OpenCV 3.4.4

Programmbibliotheken: Da der Bewegungstransfer von Bewegungsquelle zu generiertem Video mehrere separate Pythonprogramme umfasst und besonders die Versionen der einzelnen Programmbibliotheken sich teilweise aktuell stark weiterentwickeln, ist eine genaue Beachtung der verwendeten Bibliotheksversionen vonnöten. Einer der bei meinen praktischen Versuchen aufgetretenen Fehler betreffend einer Programmbibliotheksversion ist der folgende: Ich nutze als Programmpaketmanager unter Ubuntu PIP. Dieser Manager ist ein Standard Programmpaket-Installationsprogramm speziell für Python Programmbibliotheken. Bei der Ausführung des GAN Trainingsprogramm „train_pose2vid.py“ trat die Fehlermeldung „cannot import name PILLOW_VERSION“ auf. Nach einer kurzen Suche nach dem Fehler auf www.google.com fand ich eine Lösung – die Version 7.0.0 funktioniert nicht im Zusammenspiel mit einer anderen Programmbibliotheksversion (Carlson, 2020). Nach einem Downgrade auf die „Python Imaging Library“ in der Version 6.1.0

funktionierte das Programm ohne Fehler. Dieses Beispiel steht stellvertretend für einige weitere Versionsinkompatibilitäten, die während meines praktischen Teils auftraten.

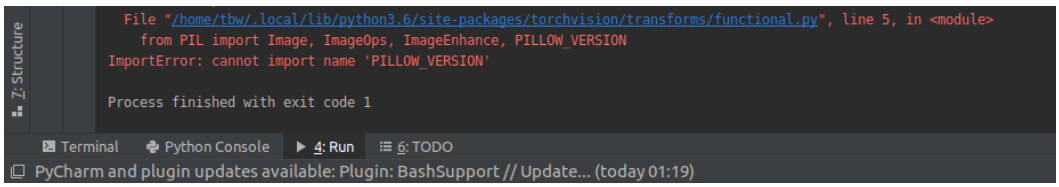


Abbildung 41: Fehlermeldung aufgrund zu aktueller Programmbibliotheksversion

PIP: Die meisten Programmelemente sind jedoch in der aktuellsten Version auch miteinander kompatibel. Die während meines Versuches zum Einsatz kommenden Bibliotheken können der folgenden Auflistung des Packagemanagementsystems PIP auf meinem Rechner unter Ubuntu 18.04LTS als Referenz entnommen werden:

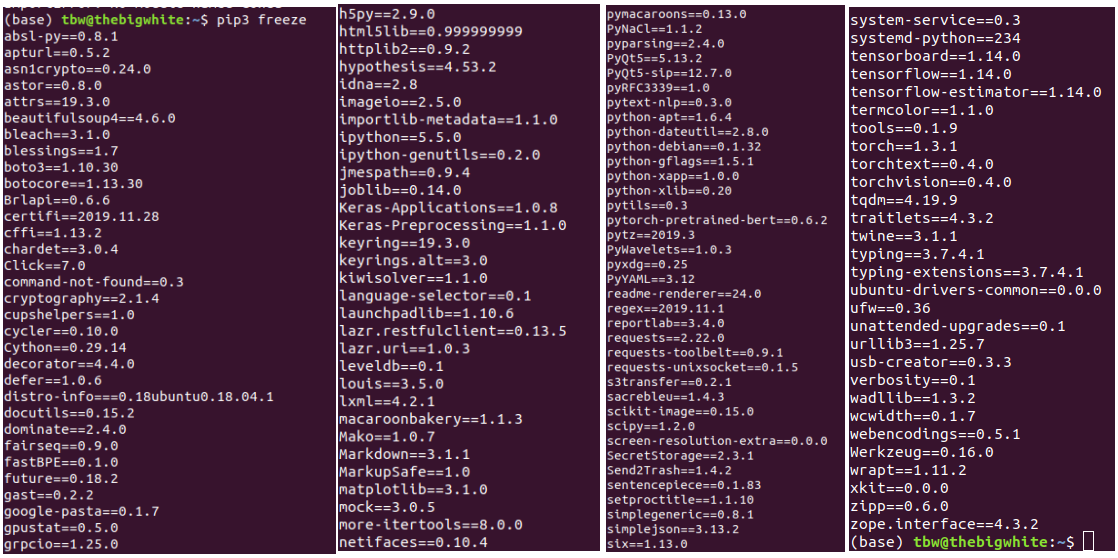


Abbildung 42: Auflistung der während des praktischen Versuches auf meinem System installierten Python Programmbibliotheksversionen

IDE: „PyCharm bietet intelligente Codevervollständigung, Code-Inspektionen, Fehlerhervorhebung in Echtzeit und Quick-Fixes, zusammen mit automatischer Code-Refaktorierung und umfassenden Navigationsmöglichkeiten“ (*Features | PyCharm*, 2020). Als integrierte Entwicklungsumgebung, kurz IDE, kommt PyCharm Community Edition zum Einsatz. Diese Entwicklungsumgebung ist zum Zeitpunkt des Experiments auf der Website des Herstellers JetBrains frei zum Download verfügbar (*Download PyCharm*, 2019). Diese IDE ist nicht zwangsweise notwendig, erleichtert aber das Bearbeiten, Ausführen und Betrachten der einzelnen Programmteile.

Zusatzsoftware: Zusätzlich zu der Hauptsoftware für den Bewegungstransfer „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) kommen bildstapel-, video- und datenpunktverarbeitende Programme zur Auswertung, Aufbereitung und Veranschaulichung der Ergebnisse zum Einsatz.

Bildstapelverarbeitung: Da die einzelnen Programmteile von „EverybodyDanceNow reproduced in pytorch“ Eingabedaten sowie Zwischen- und Endergebnisse als Bildsequenz verarbeiten und ausgeben, ist es häufig notwendig, mehrere Dateien geordnet umzubenennen, um für die Betrachtung als Video geeignete Nummerierungen zu schaffen. Dafür kommen unter Ubuntu 18.04LTS Nautilus, sowie unter Windows der Windows Explorer sowie Adobe Bridge zum Einsatz.

Videoverarbeitung: Bei der Erstellung der Quell- sowie Zieldatensätze ist es notwendig, Videomaterial in Bildsequenzen umzuwandeln. Bei der Verarbeitung der Ergebnisse ist der Schritt dann umgekehrter Natur – Bildsequenz zu Video. Um mehrere Bildsequenzen als Video nebeneinander darzustellen, einen nach Farbwert entfernbaren Hintergrund zu keyen, oder auch Posenerfassungssequenzen in ihrer zeitlichen Abfolge zusammenzufügen, sind Werkzeuge zur Videoverarbeitung notwendig. Dafür kommt unter Ubuntu 18.04LTS das frei verfügbare Tool FFmpeg zum Einsatz (FFmpeg, 2019). Unter Windows, wo die Hauptverarbeitung der Videos und Bildsequenzen stattfindet, verwende ich Adobe Premiere Pro CS6, sowie Adobe After Effects CS6.

Datenpunkterfassung und Analyse: Um zusätzliche Daten und Vergleichswerte zwischen den Gruppen der Versuche und des Hauptexperiments zu gewinnen, kommen neben dem Github Repository „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) auch zwei weitere frei verfügbare Codesammlungen, die über Github geteilt werden, zum Einsatz. Unter Windows verwende ich OpenPose Portable in der Version 1.5.0. als Standalone Programm zur Analyse und Beurteilung der Datensätze, sowie ihrer generierten Pendants in Kombination mit der Windows Kommandozeileschnittstelle (Hidalgo et al., 2017/2020). Des Weiteren kommt unter Ubuntu für den Vergleich der Wahrnehmungsqualität, wie er sich als Distanz eines Bildpaares beschreiben lässt, die Codesammlung der Metrik LPIPS in der Version 0.1 zum Einsatz (Zhang et al., 2018/2020). Für die Auswertung und grafische Darstellung der auf diese Art neu gewonnenen Daten verwende ich neben einem einfachen Texteditor, die Tabellensoftware Microsoft Excel.

Arbeitsgerät: Um für den Versuch eine angemessene Hardwareumgebung zu schaffen, verwende ich den Grafikprozessor Nvidia GTX 1080 Ti. Diesen konnte

ich freundlicherweise für die Dauer meiner Versuche von meinem Arbeitgeber ausborgen. Der Rest des Systems besteht aus privatem Equipment. Die Eckdaten des Rechners sind zum Zeitpunkt des Versuchs wie folgt:

- Prozessor: Intel i7-3930K
- Grafikkarte: NVidia GTX 1080 Ti
- Arbeitsspeicher: 48GB
- Festplatte für Betriebssysteme: Sata-6 SSD
- Festplatten für Versuche: 7200rpm HDD; NVMe m.2 SSD

Softwareabhängigkeiten: Die Anforderungen an die Softwareumgebung und das Betriebssystem habe ich wie folgt in meinem System hergestellt:

- Betriebssystem: Ubuntu 18.04 LTS
- Python Programmversion: Python 3.6
- Python Paketmanagementsystem: PIP3
- Deep-Learning Framework: Pytorch
- CUDA Tookit Version: 10.1
- Grafikkartentreiber Version: 430.50

7.5.2 Programmteile

Aufteilung der Datenverarbeitungsschritte: Die Softwaresammlung „EverybodyDanceNow reproduced in pytorch“ besteht aus mehreren Programmteilen, die nacheinander ausgeführt werden müssen und teilweise nach einer einfachen manuellen Vorbereitung der Daten zwischen den Schritten verlangen. Eine über die hier zusammengefasste Funktion der einzelnen Programmteile hinausgehende Beschreibung dieser und des Ablaufs zum Bewegungstranfer ist der dem Github beigefügten „README.md“ Dokument zu entnehmen (Wu et al., 2018/2020). Alternativ ist diese Datei auch auf dem Datenträger am Ende dieser Diplomarbeit verfügbar.

Datenaufbereitung der Bewegungsquelle: Der erste Schritt zum Bewegungstransfer mittels der hier verwendeten Codesammlung, ist die Datenaufbereitung durch die separaten Python Progamme „make_target.py“ sowie „make_source.py“. Mittels „make_source.py“ wird aus dem Bewegungsquelldatensatz in der Form eines Videos eine Bildsequenz erstellt. Des Weiteren werden Posenerfassungen zu jedem Einzelbild mittels der Programmbibliothek PyOpenPose extrahiert und dem jeweiligen Bild als Zusatzinformation beigefügt. Für eine optionale Gesichtsoptimierung im letzten Schritt wird außerdem die Position des Gesichts in jedem einzelnen Bild in der NumPy Datei „pose_source.npy“ abgespeichert. Zusätzlich dazu wird auch noch

der vermutete Ausschnitt, in dem sich das Gesicht im Bild findet, als direkt zu betrachtendes .jpg Bild erstellt und abgespeichert.

Zieldatensatz: Für die Aufbereitung des Zieldatensatzes kommen die gleichen Schritte zum Einsatz. So wird auch mittels „make_target.py“ eine Bildsequenz mit dazugehörigen Posenerfassungen und den jeweiligen Koordinaten, sowie den als Bild abgespeicherten Ausschnitten des Gesichts erstellt. Die Daten des Zielvideos dienen dabei als Trainingsdaten für das neuronale Netwerk, wohingegen die Daten des Bewegungsquellvideos als Testdaten für das neuronale Netzwerk dienen. Das Ergebnis des Tests ist im letzten Schritt dann der gewünschte Bewegungstransfer.

GAN Training: In dem Programmteil „train_pose2vid.py“ werden die neuronalen Netze für den Diskriminator und Generator mittels der GAN Zielvorgabe speziell auf den aktuellen Zieldatensatz ausgebildet und trainiert. Dies erfolgt über die Dauer einer frei definierbaren Anzahl an Rechendurchgängen. Wie lange trainiert werden soll, kann über das Programmteil zur Konfiguration, „train_opt.py“, eingestellt werden. Die Maßeinheit für die gewünschte Anzahl an Trainingsschritten ist eine sogenannte Epoche. Eine Epoche beschreibt alle Rechendurchgänge, die notwendig sind, um einmal sämtliche Daten eines Datensatzes betrachtet zu haben (Burkov, 2019, S. 5 Kapitel 4). Das aktuelle Modell, welches die zuletzt gewichtete Struktur des neuronalen Netzwerks abspeichert, findet sich für den Diskriminator im PyTorch Programmteil „latest_net_D.pth“ und für den Diskriminator in „latest_net_G.pth". Alle 10 Epochen wird des Weiteren je ein zusätzliches Modell für Generator und Diskriminator abgespeichert. Über die Konfigurationsdatei „train_opt.py“, kann außerdem festgesetzt werden, ob von Grund auf, sprich zufälligem Rauschen, oder von einem bereits bestehenden Modell ausgehend das Training initialisiert werden soll. Zur visuellen Betrachtung wird des Weiteren ein Bildpaar aus echtem und zuletzt generiertem Bild anhand des extrahierten Skeletts der Posenerfassung abgespeichert. Dadurch lässt sich der Fortschritt des GAN Trainings nach jeder Epoche kontrollieren. Denselben Fortschritt, jedoch ausgedrückt in den Ergebnissen der Fehlerfunktionen als Zahlenwert, speichert das den Modellen beiliegende „loss_log.txt“ Textdokument.

Datenabgleichung: Für die Angleichung des Labels, i.e. der Posenerfassung der Bewegungsquelle, ist der Programmteil „normalization.py“ vorgesehen. Mittels diesem kann die Position im Bild und Größe des erfassten Skeletts der Bewegungsquellperson so an die Position und Größe der Zielperson angepasst werden, dass sie etwas besser dem gelernten Datensatz der Posenerfassungen des Zielvideos entspricht.

Bewegungstransfer: Die letzten Stücke des Weges durch die neuronale Netzwerkstuktur sind in dem Programmteil „transfer.py“ enthalten. Hier wird das gelernte neuronale Netz auf das neue aus Posenerfassungen bestehende Testdatenset angewandt. Dies geschieht, um die Einzelbilder der Zielperson in den neuen Posen zu generieren. Diese finalen Ausgabebilder werden in einer Auflösung von 512x512 Pixeln als .jpg in quadratischem Seitenverhältnis abgespeichert. Zum direkten Vergleich werden auch wieder die als Basis der Synthetisierung dienenden, zum Endergebnis korrespondierenden Posenerfassungen als Bilder der gleichen Auflösung und Abmessung abgespeichert.

Datenweiterverarbeitung: Zur Weiterverarbeitung des gewonnenen Bewegungstransfers bieten sich noch zwei zusätzliche Programmteile, welche jedoch ebenso wie die Normalisierung optional sind. Mittels des Gesichtsoptimierungssystems, wiederum bestehend aus drei Python-Programmen, kann ein separates neuronales Netzwerk lernen, wie das Gesicht in einem bereits generierten Bild des Transfers verbessert werden kann. Damit dies gelingt, müssen die Bildsequenzdaten nach bereits erfolgtem Haupttraining erneut für dieses zusätzliche neuronale Training aufbereitet werden. Dies geschieht in dem Unterprogramm „prepare.py“. Nach diesem Schritt kann mittels „main.py“ das Gesichts-GAN ausgebildet werden. Anders als beim Haupttraining für den Bewegungstransfer anhand Posenerfassungen wird beim Training des Gesichts-GAN nach jedem abgeschlossenen Rechendurchlauf ein .pth PyTorch Modell abgespeichert. Außerdem kann durch zwei zu den neuronalen-Modelldateien gehörende visuelle Gegenüberstellungen die Leistung des Systems beurteilt werden. Diese Gegenüberstellungen werden jede Runde um das aktuelle Modell erweitert. In diesen grafischen Gegenüberstellungen wird das Originalbild mit dem generierten, dem Fehler in der Synthetisierung des ausgewählten Ausschnitts, i.e. dem Gesichts-Residuum, und der aktuell möglichen Optimierbarkeit durch das mittels des aktuellen Generators fertig verbesserte synthetisierte Zielbild verglichen. Eine solche Grafik findet sich im Abschnitt „Qualitative Beobachtungen“. Nachdem dieses Gesichtsoptimierungstraining abgeschlossen ist, kann der aktuelle Generator auf den als Testdatensatz zu beschreibenden, vorab erfolgten Bewegungstransfer angewandt werden. Dies geschieht mittels des Unterprogramms „enhance.py“.

.gif: Abschließend bietet „make_gif.py“ noch die Möglichkeit die gewonnenen Bilddaten zu einem bewegten Video in dem .gif Format zusammenzufügen. Da die Weiterverarbeitung in meinen praktischen Versuchen jedoch in einem Videoprogramm mittels Bildsequenzen funktioniert, wird auf diesen Programmteil in den folgenden Kapiteln nicht näher eingegangen.

7.6 Durchführung des Hauptexperiments

Das Hauptexperiment des Vergleichs von Bewegungstransfer auf denselben fiktiven Charakter – einmal basierend auf Bildern von 2D Zeichnungen und einmal auf Bildern eines 3D Renders – erfolgt in sechs Phasen:

1. Vorbereitung der Zieldatensätze für das Training
2. GAN Training der neuronalen Netzwerke für jeden Datensatz
3. Vorbereitung des Quelldatensatzes für die Transfers
4. Bewegungstransfer der Bewegungsquelle auf die Bewegungsziele
5. Sammlung und Aufbereitung der Ergebnisse/Daten des Trainings und des Transfers
6. Auswertung der gewonnenen Daten

1. Vorbereitung der Zieldatensätze für das Training

Goku2D158: Der 2D Zieldatensatz Goku2D158 setzt sich zusammen aus 158 einzelnen Zeichnungen aus einer Mischung von frei verfügbaren Onlinequellen. Dieser Datensatz wird zusammengestellt unter Zuhilfenahme der Suchmaschinen images.google.com für Bilder, sowie www.giphy.com und www.YouTube.com für Videos.

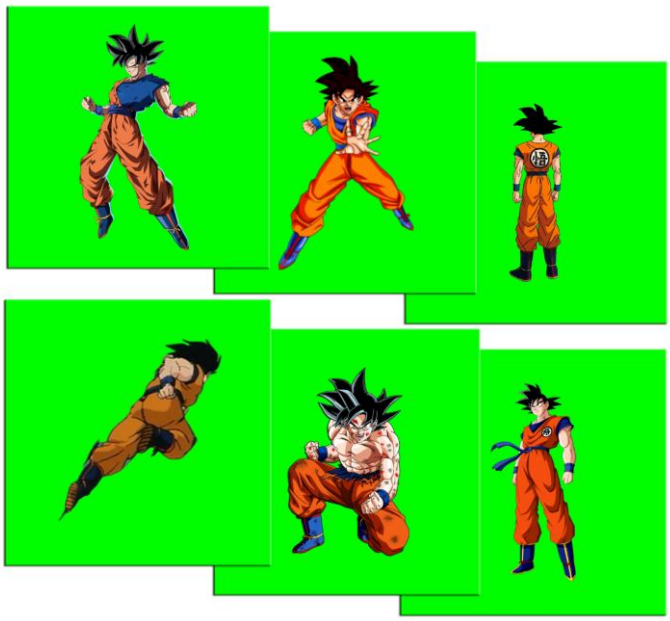


Abbildung 43: Auszug einiger Zeichnungen wie sie im Zieldatensatz der Experimentalgruppe Goku2D158 zum Einsatz kommen

Diese 158 Posen werden dann durch Duplizierung und zufällige Translation, Rotation, sowie Skalierung in einem HD Bild auf 3599 Gesamtbilder im finalen Zieldatensatz Goku2D158 erweitert. Goku2D158 wird mit dem ersten Programmteil von „EverybodyDanceNow reproduced in pytorch“, dem Python Programm „make_target.py“ zur im folgenden Trainingsschritt nutzbaren Ressource.

Goku3D158: Dieser Ablauf wiederholt sich für den Datensatz Goku3D158, welcher auf dieselbe Art wie Goku2D158 erstellt wurde. Auch dieser enthält 158Posen und insgesamt 3599 Bilder, welche mittels Translation, Rotation und Skalierung die Robustheit des gelernten Netzwerkes erhöhen sollen.

Abbildung 44: Auszug einiger Renderings, wie sie im Zieldatensatz der Kontrollgruppe Goku3D158 enthalten sind

2. GAN Training der neuronalen Netzwerke für beide Zieldatensätze

Zeitliche Dauer: Den zeitintensivsten Prozess des gesamten Ablaufs stellt das zentrale Element der Bewegungstransfer Softwarepipeline dar. Mittels "train_pose2vid.py" ist es möglich, dank der Zuhilfenahme von einem vortrainierten VGG19 Netzwerk aus Bildern des Charakters Goku, dessen Erscheinung in Abhängigkeit von einer dazugehörigen Pose zu erlernen. Für das Training über 40 Epochen bei der Datensatzgröße von 3599 Bildern ist auf meiner Maschine mit einer einzelnen Grafikkarte (NVidia GTX1080Ti) ein Zeitaufwand von 14h reiner Rechenzeit notwendig. Im Vergleich dazu dauert die Erstellung des

Zieldatensatzes aus dem Ausgangszielvideo lediglich vier Stunden. Dies liegt auch daran, dass die Suche nach dem Gesicht im Bild im Vergleich zu der Extraktion der Pose eher rechenintensiv ist. Ohne eine spätere Gesichtsoptimierung wäre dieser Prozess also wesentlich schneller durchlaufbar. Mittels der Konfigurationsdatei „train_opt.py“ wird vor dem Start des Trainings die Epochenanzahl auf 40 festgesetzt.

Initialisierung: Ähnlich der Idee von großen Datensätzen in einem vorab-Training zu profitieren, wie sie in „Few-shot Video-to-Video Synthesis“ thematisiert wird (T. Wang et al., 2019, S. 9), verwende ich die Möglichkeit, ein vortrainiertes Netzwerk als Basis für die erste Initialisierung des Trainingsprozesses zu nutzen, wie sie sich in der Codesammlung von „EverybodyDanceNow reproduced in Pytorch“ bietet. Die Basis für diese Initialisierung habe ich über eine Dauer von 220 Epochen lokal auf meinem Computer vortrainiert. Auf diesem Rechner dauerte dieses Training drei Tage (reine Rechenzeit). Dieses Netzwerkmodell wurde mittels dem Datensatz PNGGokuWiggle ebenso auf Bildern von Son Goku vor einem grünen Hintergrund trainiert, jedoch ohne Rücksichtnahme auf Haarfarbe oder Dimensionalität der Originalbilder bei der Zusammenstellung des Datensatzes. Die Hoffnung beim Verwenden dieses Netzwerks besteht darin, dass z.B. Eigenschaften, die über alle zum Einsatz kommenden Zieldatensätze hinweg gleich sind, wie der grüne Hintergrund, bereits entsprechend im Netzwerk abgebildet sind. Das finale neuronale Zielnetzwerk gelangt dadurch theoretisch rascher zu einem guten Lernerfolg. Ein weiterer Mitgrund für dieses Vorgehen liegt in der zeitlichen Einsparung, welche die Durchführbarkeit verschiedener Versuchsdurchläufe ermöglicht. Konkret wird dies erreicht durch die Aktivierung der „load_pretrain“ Einstellung in der Konfigurationsdatei „train_opt.py“ (Wu et al., 2018/2020).

Zwischenstand: Mit dem Programm „train_pose2vid.py“ wird dann der Trainingsprozess gestartet. Da während dem gegenspieligen Training des neuronalen Diskriminators und Generators auch bereits erste Ergebnisse sichtbar werden, lohnt ein Blick in den Ordner „Checkpoints“, wo diese fortlaufend abgespeichert werden.

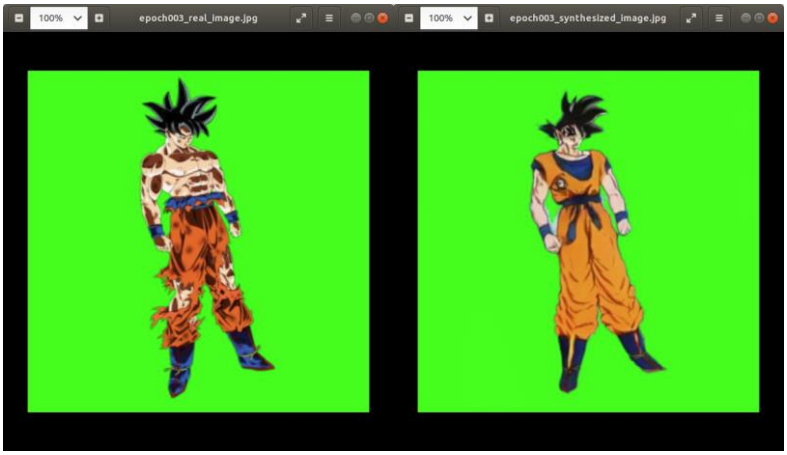


Abbildung 45: Originalbild und generierte Zeichnung nach Epoche 3

Kleidung: Nach Epoche drei ist in einem generierten Testbild zu beobachten, wie der Generator visuelle Eigenschaften abbildet, welche nicht im Originalbild vorhanden sind.

Halluzinierte Körperteile: In der nächsten Gegenüberstellung kann man erkennen, dass im mittleren Bild, welches die Posenerfassung des rechtsstehenden Originalbildes darstellt, das linke Bein nur bis zum Knie erkannt wurde. Der Charakter Goku besitzt jedoch in den meisten Posenerfassungen und in sämtlichen Bildern des Datensatzes zwei Beine. Daher liegt die hypothetische Annahme nahe, dass sich diese anamtomische Eigenschaft des Charakters Son Goku in den neuronalen Ebenen des Generators nach Epoche 26 bereits in irgendeiner abstrakten Form wiederfindet.

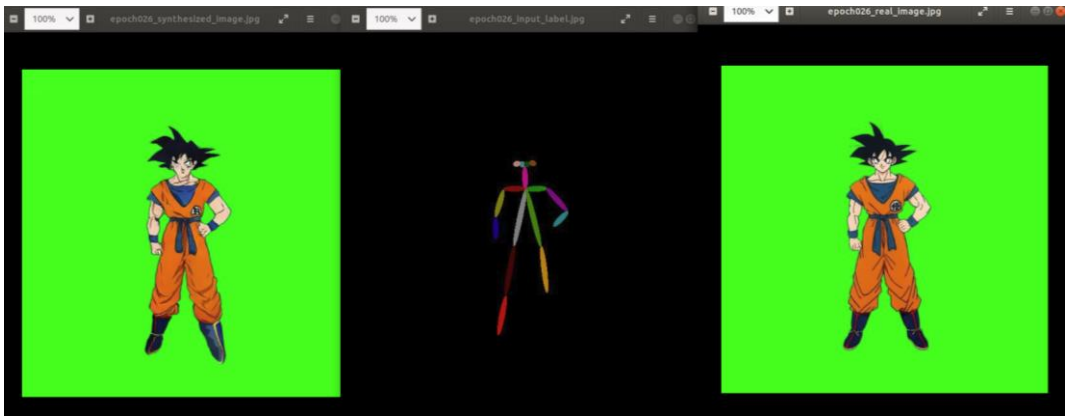


Abbildung 46: Generierung einer vollständigen Gliedmaße trotz unvollständiger Posenerfassung nach Epoche 26 bei dem Datensatz Goku2D158; links syntherisiertes Bild, Mitte Posenerfassung, rechts Originalbild

Nachdem das Training abgeschlossen ist, werden die Gewichte und neuronalen Aktivierungen als "latest_net_D.pth" (Diskriminator) und „latest_net_G.pth" (Generator) PyTorch Modell abgespeichert.

3. Vorbereitung des Quelldatensatzes für die Transfers

Datensätze und verfolgte Ziele: Als Bewegungsquelle für das Hauptexperiment dienen insgesamt drei Datensätze. Tatsächlicher Bewegungstransfer wird jedoch nur mit einem durchgeführt – dem Musikvideo „That's What I Like" von Bruno Mars (Mars & Lia, 2017). Da für ein möglichst kontrolliertes Experiment tatsächlicher Bewegungstransfer mittels nicht im Trainingsdatensatz enthaltener Posen vielen Variablen im Zusammenspiel mit den spezifischen verwendeten Zieldatensätzen unterliegt, dient dieser im Experiment nur zur subjektiven qualitativen Beurteilung des Trainingserfolgs. Die wichtigeren Datensätze für die später folgende Auswertung sind die Zieldatensätze Goku2D158 sowie Goku3D158. Dadurch dass diese Bewegungszieledatensätze jeweils auch als Bewegungsquelldatensatz aufbereitet werden, wird eine Erhebung von quantitativen Daten, wie es die Wahrnehmungsmetrik LPIPS darstellt, möglich. Dadurch können die nach je 40 Epochen ausgebildeten neuronalen Netzwerke von Goku2D158 und Goku3D158 besser miteinander verglichen werden.

Konkrete Vorbereitung: Der Programmteil, mit dem aus einem Musikvideo wie „That's What I like" von Bruno Mars eine Quelle entsteht, trägt den deskriptiven Namen „make_source.py". Je nach Anzahl der enthaltenen Bilder in dem Video, kann die Erstellung der Quelldaten einige Zeit in Anspruch nehmen. Die Erstellung einer Bewegungsquelle aus dem Bewegungszieldatensatz Goku2D158 mit 3599 Bildern dauert auf dem mir zur Verfügung stehenden Rechner in etwa vier Stunden.

4. Bewegungstransfer der Bewegungsquelle auf die Bewegungsziele

Transfer: Nachdem aus der Bewegungsquelle sämtliche Posenerfassungen als Bildsequenz zur Verfügung stehen und für den Zielcharakter ein neuronales Generatorennetzwerk besteht, kann der Transfer durchgeführt werden. Dieser dauert sowohl für Goku2D158 als auch Goku3D158 bei „That's what I like" als Bewegungsquelle nur wenige Minuten. Das ausgeführte Programm für diesen Schritt heißt „transfer.py". Der fertige Bewegungstransfer als Bildsequenz findet sich dann in dem Ordner /results/target/test_latest/images. Für jeden der zwei Datensätze wird einmal ein Transfer mit dem Musikvideo „That's What I like" (Mars & Lia, 2017) als Bewegungsquelle und einmal mit dem eigenen Trainingsdatensatz als Quelle (Goku2D158, Goku2D158) durchgeführt.

5. Sammlung und Aufbereitung der Daten des Trainings und der Ergebnisse des Transfers

Bildsequenzpaare der Netzwerktrainings- und Posenerfassungsleistung: Die Ergebnisse der Transfers ähneln grundsätzlich der Erscheinung der Trainingsbilder. Im konkreten Fall sind dies aus 512x512 Pixeln bestehende Einzelbilder die als Bildsequenzen des Charakters Son Goku vor grünem Hintergrund gespeichert werden. Diese Bildsequenzen der erfolgten Transfers werden gesammelt und sortiert abgespeichert. Zusätzlich wird zum Vergleich der Trainingsleistungen der Netzwerke nach dem erfolgten Training auf den zweidimensionalen Zeichnungen sowie den dreidimensional gerenderten Bildern auch noch jeweils die Bildsequenz des Trainingsdatensatzes abgespeichert. Auch diese Bildsequenzen werden in einer Auflösung von 512x512 Pixel erstellt. Für den direkten Vergleich eines Bildes aus dem Zieldatensatz und eines dazugehörigen generierten Bildes bei Epoche 40 mithilfe der Wahrnehmungsdistanzmetrik LPIPS müssen die korrespondierenden Bildpaare jeweils denselben Namen tragen. Dafür kommt eine Stapelumbenennung mittels dem Ubuntu 18.04LTS eigenen Dateimanager Nautilus zum Einsatz. Dies ist außerdem notwendig, um die Daten des originalen Zieldatensatzes, sowie die des nach Epoche 40 generierten Zieldatensatzes, der Standalone Version des Posenerfassungsalgorithmus OpenPose zu übergeben. Dafür werden die umbenannten Bildsequenzen noch in einem Zwischenschritt mittels Adobe Premiere Pro CS6 zu jeweils einem Video zusammengefügt. Dies erlaubt eine einfachere Dateiverwaltung. Diese Videos werden dann wiederum an OpenPose übergeben, um jeweils zwei neue Posenerfassungsbildsequenzen pro 2DOriginal/2DGeneriert und 3DOriginal/3DGeneriert Bildsequenzpaar zu erhalten. Anhand dieser Bildsequenzpaare der Originalzieldatensätze und den dazugehörigen generierten Bildern, sowie den in diesen originalen Bildern und wiederum generierten Bildsequenzen als Skelett erfassten Posen, wird im nächsten Schritt, i.e. der Auswertung der gewonnenen Daten, die Wahrnehmungsdistanz mittels der Metrik LPIPS berechnet.

Musikvideo-Bewegungstransferbildsequenz: Die gewonnenen Bildsequenzen des Transfers anhand der Bewegungen des Tänzers im Musikvideo werden ebenso als Stapel umbenannt, damit diese von dem Videoschnittprogramm Adobe Premiere Pro CS6 als Bildsequenz korrekt importiert werden können.

Fehlerfunktionsdatenpunkte: Nicht nur die Daten des Transfers sind interessant für eine Auswertung, sondern auch die des Lernfortschritts während des GAN Trainings. Um diese Daten auszuwerten, wird die zum jeweiligen

Generator-Diskriminatornetzwerkpaar gehörende "loss_log.txt" Textdatei herangezogen.

6. Auswertung der gewonnenen Daten

Auswertungsarten: Die in dem Experiment durch Schritte 1-5 gewonnenen Daten und Bildsequenzen werden nun auf verschiedene Arten ausgewertet und im nächsten Abschnitt, den Ergebnissen, miteinander verglichen und interpretiert.

Synthetisierungsqualität: Die erste Auswertung beschäftigt sich mit der Frage der Qualität des trainierten Netzwerks, wie sie sich in der Generierung einer Bildsequenz anhand der Bewegung eines Tänzers zeigt. Die qualitative, subjektive Betrachtung der Ergebnisse dieser Bewegungstransfers erleichtere ich durch eine visuelle Gegenüberstellung zwischen den generierten Bildsequenzen. Diese Gegenüberstellungen wurden jeweils mit den gespeicherten Bewegungstransferbildern erstellt. In der Mitte findet sich die Bewegungsquelle, i.e. der Tänzer des Musikvideos „That's what I like“ (Mars & Lia, 2017). Diese Gegenüberstellung ist dank des grünen Hintergrunds einfach möglich, da dieser mit dem Effekt „Ultra Key“ in Adobe Premiere Pro CS6 einfach entfernbar ist.

Abbildung 47: Visuelle Gegenüberstellung der Bewegungstransfers generiert mittels Goku3D158-GAN und Goku2D158-GAN

Wahrnehmung quantifiziert: Um eine quantitative Maßeinheit für die theoretische visuelle Synthetisierungsleistungsfähigkeit der neuronalen Netzwerke in Bezug auf das Trainingsdatenset zu erhalten, berechne ich mit dem frei auf Github verfügbaren Programm LPIPS (Zhang et al., 2018/2020) die Wahrnehmungsdistanz für alle Bildpaare zwischen Goku2D158 und synthetisiertem Goku2D158 Datensatz nach Trainingsepoche 40. Den gleichen

Prozess führe ich auch für Goku3D158 in Kombination mit dem nach Epoche 40 synthetisierten Goku3D158 Datensatz durch.

Positionsdistanz: Wie gut anhand der generierten Synthetisierungen des Zieldatensatzes nach Epoche 40 erneut die Posen des darin abgebildeten Charakters erfasst werden können, gibt Aufschluss über die korrekte Reproduktion einer Pose durch das synthetisierte Bild des Generators wie auch über Anforderungen an das Ausgangsbildmaterial, welche für eine funktionierende Posenerfassung erfüllt sein sollten. Für die Berechnung dieses Werts kommt wieder das Programm „compute_dists_dirs.py“ zum Einsatz (Zhang et al., 2018/2020). In diesem Distanzwert vergleiche ich die auf schwarzem Hintergrund dargestellten Posenerfassungen der Zieldatensätze Goku2D158 und Goku3D158 mit ihren korrespondierenden extrahierten Posenerfassungen aus den nach Epoche 40 synthetisierten gleichnamigen Ergebnisdatensätzen.

Fehlende Erfassungen: Darüber hinaus werte ich aus den ersten 100 der 158 Posen beider Posenerfassungssequenzen aus den nach Epoche 40 synthetisierten Bildsequenzen die fehlenden und teilweise fehlenden Posenerfassungen händisch aus. Dies erreiche ich dadurch, dass ich die Posenerfassung in der Originalbildsequenz mit der synthetisierten Bildsequenz vergleiche. Ist eine Posenerfassung in beiden Bildern eines Paares vollständig vorhanden, zählt diese als erfolgreich. Ist eine Erfassung vollständig im Original, aber unvollständig im synthetisierten Bild, so werte ich diese als teilweise Erfassung. Wenn im Zielbild keine Erfassung der Pose vorhanden ist, jedoch im Original zumindest eine teilweise, dann zählt diese als fehlende Posenerfassung.

Trendlinien - GAN Lernfortschritt: Um den Lernfortschritt von Epoche 1-40 vergleichen zu können, werte ich mittels dem Tabellenverarbeitungsprogramm Excel die „loss_log.txt“-Datei der zwei neuronalen Netzwerke, Goku2D158-GAN und Goku3D158-GAN, graphisch als Diagramm aufbereitet aus. Dies erfolgt durch die Darstellung der einzelnen Kenngrößen der Fehlerfunktionen für Diskriminator und Generator als Punktwolke, in welcher mittels linearer Regression der Lerntrend eingezeichnet wird. Auch die durch das Programm LPIPS gewonnen Daten bereite ich in Excel grafisch auf.

7.7 Ergebnisse

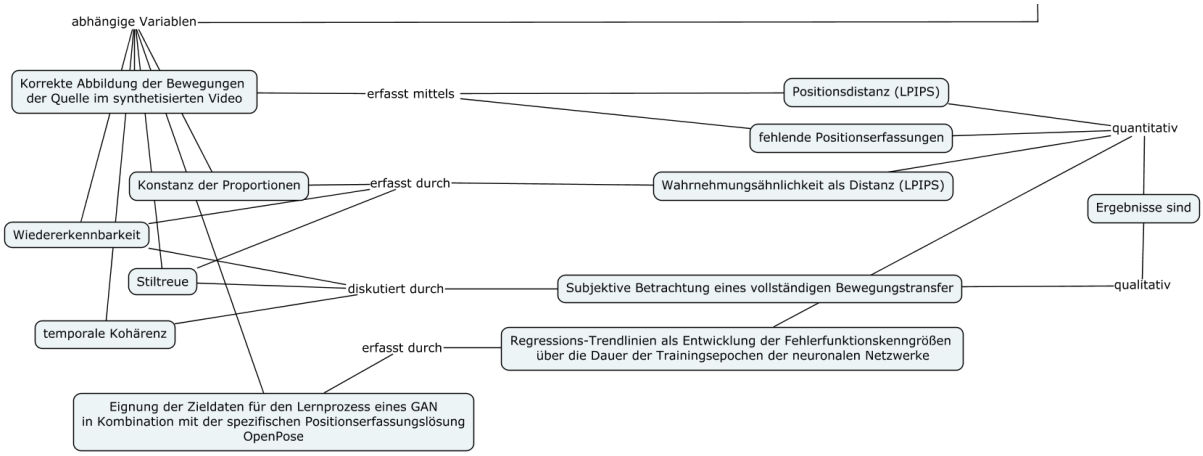


Abbildung 48: Zu untersuchende Effekte (links) nach der Auswertung der gewonnenen Daten und Kenngrößen aus den Versuchsdurchläufen, anhand der vorab definierten Metriken (rechts)

Kernfrage: Hauptforschungsziel ist es, herauszufinden, ob das neuronale maschinell lernende Softwaresystem „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) zum Bewegungstransfer bei gezeichnetem Zielcharakter grundsätzlich geeignet ist. Dies geschieht anhand des Animecharakters Son Goku. Durch einen kontrollierten Vergleich zwischen zwei möglichst ähnlichen Datensätzen des Charakters Son Goku werden Daten über die Leistung und Qualität der einzelnen Programmelemente gesammelt und Beobachtungen dokumentiert. Die wichtigste Unterscheidung zwischen den Datensätzen ist ihre bei der Erstellung der Bilder zum Einsatz kommende Technik und die darin enthaltene Dimensionalität. Das bedeutet, es geht primär darum möglichst gut zu isolieren, ob die Art der Charakterbilderstellung durch 3D Animationswerkzeuge einen nenneswerten Vor- oder Nachteil gegenüber dem Prozess der Zeichnung eines Charakters hat im Hinblick auf die konkret betrachtete maschinell lernende Lösung zum Bewegungstransfer.

7.7.1 Vergleiche der ausgewerteten Datenpunkte

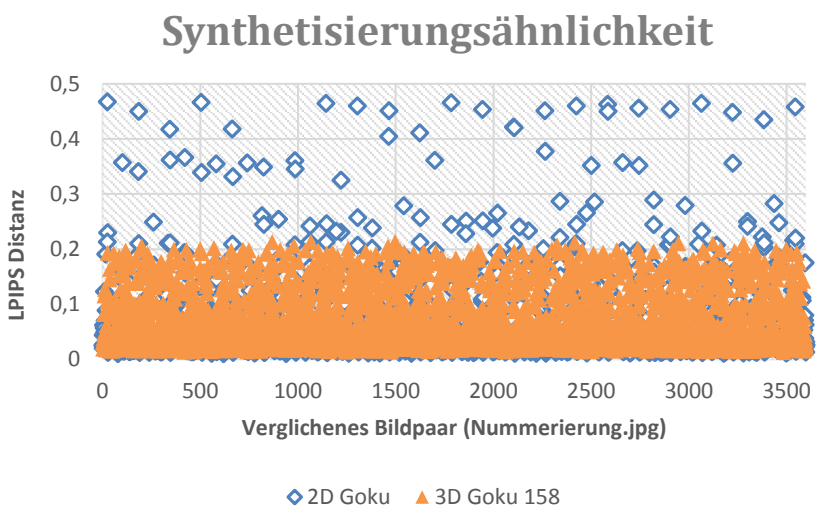


Abbildung 49: LPIPS Distanz zwischen identen Trainings- und Testdatensätzen, jeweils bei gezeichnetem (blau) und gerendertem (orange) Son Goku; Eine niedrigere LPIPS Distanz beschreibt eine bessere visuelle Ähnlichkeit

Streubereich: In dieser Abbildung sind sämtliche Ergebnisse des Vergleichs der 3599 Originalbilder des Zieldatensatzes Goku2D158 mit den nach Trainingsepoche 40 synthetisierten Bildern anhand des Quelldatensatzes Goku2D158 in blau abgebildet. Trainings- und Testdatensatz sind ident, wodurch auch keine Unterschiede in den Zusatzdaten, den Posenerfassungen zu Trainings- und Testzeitpunkt (i.e. Zeitpunkt des Transfers) für das neuronale Generatornetzwerkmodell bestehen. Dieselbe vergleichende LPIPS Distanzinformation für den Ziel- und Quelldatensatz Goku3D158 ist in orange zu sehen. Die blauen Werte befinden sich großteils zwischen 0,0 und 0,2, mit einigen Ausreißern bis 0,5, wohingegen die in orange eingezeichneten Werte über die erreichte Ähnlichkeit beim Vergleich mit den synthetisierten Bildern bei Goku3D158 sich in einem augenscheinlich niedrigeren Bereich verteilen.

Interpretation der Streuung: Ein niedriger LPIPS Wert bedeutet, dass dieses verglichene Bildpaar einander stark ähnelt. Die Werte von Goku3D158 verteilen sich gleichmäßiger als jene von Goku2D158. Dies kann mit mehreren Störfaktoren zusammenhängen. Darunter fallen z.B. die unterschiedliche Gleichmäßigkeit in den Erscheinungsformen von Son Goku bei den zwei Datensätzen, oder auch eine variierende Leistung des Posenerfassungsalgorithmus OpenPose. Die Leistung der Posenerfassung wird an späterer Stelle jedoch noch verglichen. Ich vermute einen Hauptgrund für die scheinbar leicht größere Streuung in den von verschiedenen Menschen

angefertigten Zeichnungen und den darin auftretenden Unterschieden der Erscheinung Son Gokus.

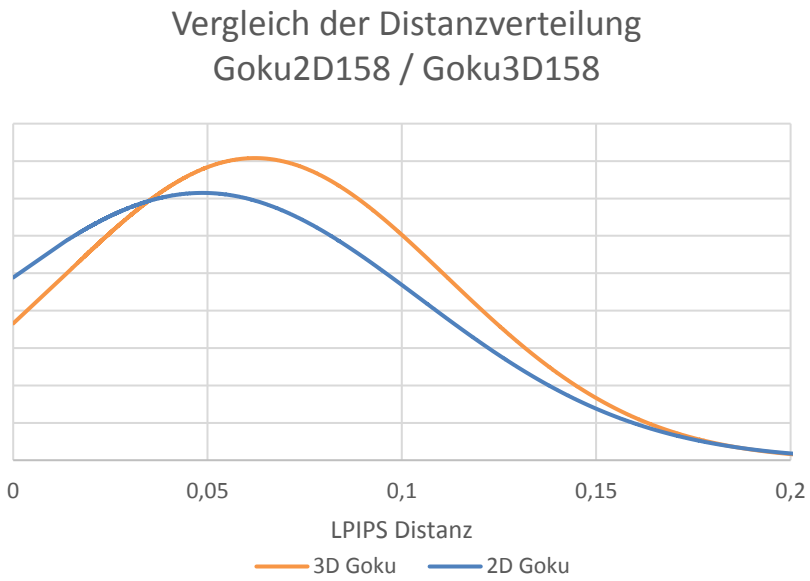


Abbildung 50: Wahrscheinlichkeit des Erzielens eines bestimmten LPIPS Ähnlichkeitswerts nach Epoche 40 für Goku2D158 und Goku3D158; niedriger, i.e. Maximum weiter links, ist besser

Verteilung: Im obenstehenden Vergleich der Verteilungskurven der Häufigkeit konkreter Wahrscheinlichkeitswerte zeigt sich, dass die Distanzwerte für die synthetisierten gezeichneten Bilder sich um ein niedrigeres Mittel streuen als die Werte des auf Renderings des Charakters Son Goku basierenden LPIPS Ähnlichkeitswerte des Goku3D158 Datensatzes. Diese sind dafür wiederum über einen geringeren Bereich verteilt. Das arithmetische Mittel liegt für Goku2D158 bei 0,05 und Goku3D158 bei 0,06.

Interpretation der Verteilung: Der geringe Unterschied des Mittels der zwei Kurven deutet auf eine ähnliche Qualität des Lernerfolgs beider neuronaler Netzwerke hin. Dies könnte konkret bedeuten, dass beim hier betrachteten Charakter Son Goku der Unterschied in der Dimensionalität der Ausgangsdaten, i.e. Zeichnung im Vergleich zu Rendering, kaum einen Einfluss auf den erreichbaren Lernerfolg mit der Software „EverybodyDanceNow reproduced in pytorch“ hat. Dies könnte weiters bedeuten, dass mit der Softwarearchitektur von „EverybodyDanceNow reproduced in pytorch“ auch gezeichnete Zielcharaktere grundsätzlich in einem neuronalen Netzwerk in ähnlicher Qualität abgebildet werden können wie ihr 3D gerendertes Pendant. Der hier durchgeführte Versuch

zeigt dieses Ergebnis trotz verschiedener Gleichmäßigkeit in den Erscheinungsformen des gewählten Charakters wie sie sich in den jeweiligen Zieldatensätzen finden.

Anzahl der reproduzierten Posen: Die Metrik der fehlenden Erfassungen zeigt, ob in einem synthetisierten Video eines bestimmten Charakters auch erneut von der Posenerfassungssoftware OpenPose eine Pose, die der Labelpose des Originals nahekommt, erkannt wird. Im Optimalfall werden nach einem Bewegungstransfer ebenso viele Posen vollständig erkannt, wie vor dem Transfer im Quellmaterial vorhanden sind. Hierfür kommen die ersten 100 Posenpaare der zwei 2D Datensätze, bestehend aus Bildern von Goku2D158 im Original und den jeweils synthetisierten Pendants nach Epoche 40, sowie den 3D Bildern von Goku3D158 und den daraus synthetisierten Pendants, zum Einsatz. In Hinblick auf die wieder als solche erfassbaren Posen zeigt sich eine leicht bessere Leistung des Goku2D158-GAN.

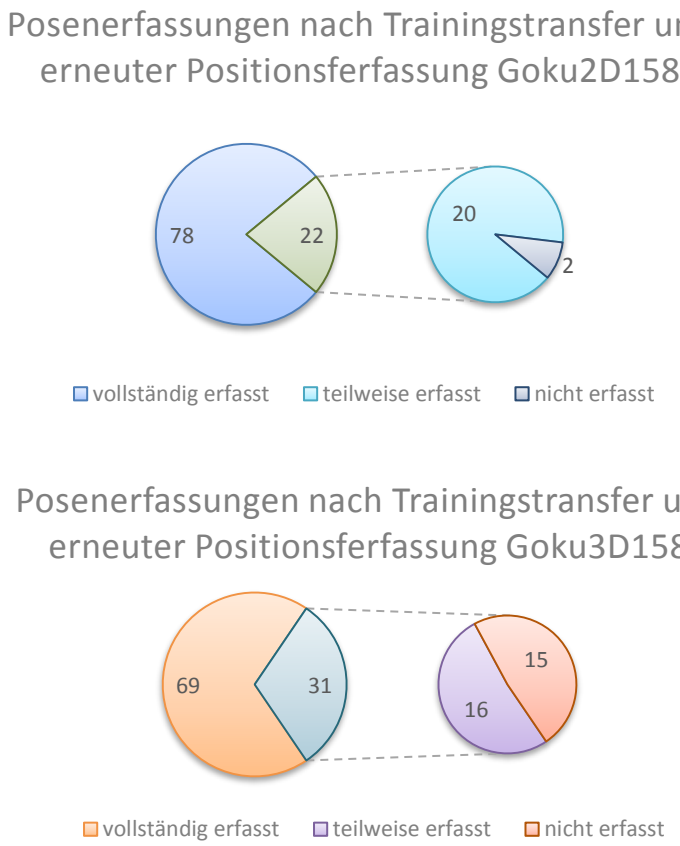


Abbildung 51: Vergleich der vollständigen, teilweisen und gänzlich fehlenden Erfassungen mittels OpenPose (Hidalgo et al., 2017/2020) aus den jeweils nach Trainingsepoche 40 neu synthetisierten Zieldatensätzen

Reproduktionsgenauigkeit der neu erfassten Pose: Die Posendistanzmetrik ist eine weitere Auseinandersetzung mit der Veränderung der Leistung des Posenerfassungssystems OpenPose nach dem erfolgten Bewegungstransfer. Auch in dem Paper „Everybody Dance Now“ wird diese Metrik erhoben, jedoch geschieht dies darin als direkte Distanz zweier korrespondierender Gelenkskoordinaten (Chan et al., 2018, S. 5). Im Gegensatz dazu, verwende ich erneut die Wahrnehmungsdistanz LPIPS, um die 3599 Posenerfassungenspaare aus jeweils den Bildsequenzen der Originalzieldatensätze und der synthetisierten Zieldatensätze zu vergleichen. Um durch bereits im Originalzieldatensatz nicht erfasste Positionen das Ergebnis nicht zu verfälschen, filtere ich in Excel alle Werte der 3599 Bildpaare aus der LPIPS Ergebnistabelle heraus, welche einen Wert von 0 aufweisen. Die Werte ohne Posenerfassungen sind deswegen 0, weil ein schwarzes Bild ohne enthaltene Pose mit einem zweiten schwarzen Bild ohne enthaltene Pose ident ist.

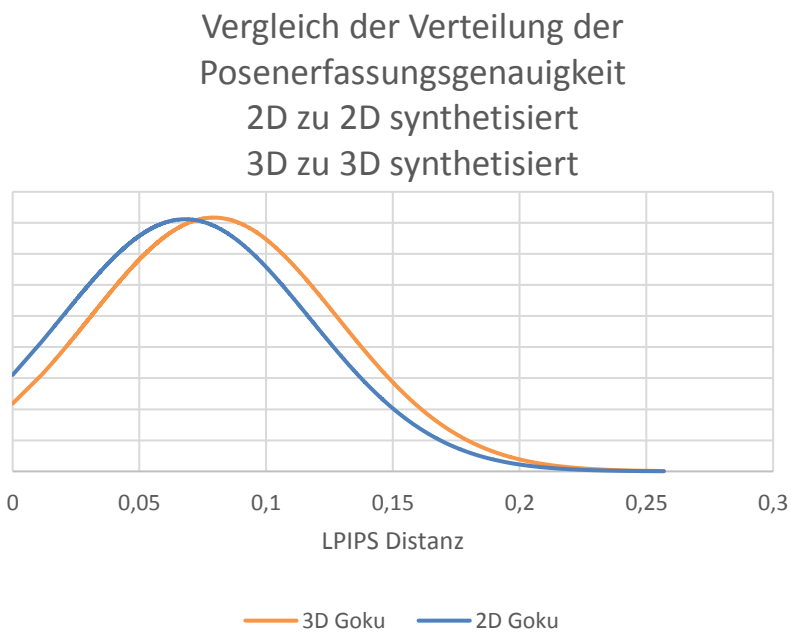


Abbildung 52: Vergleich der Verteilung der Wahrscheinlichkeit eines bestimmten LPIPS Ähnlichkeitswerts beim Betrachten eines Bildpaares zweier OpenPose Posenerfassungen anhand der Datensätze Goku2D158, Goku3D158, sowie ihren jeweiligen synthetisierten Pendants nach GAN Trainingsepoche 40.

Beobachtete Ähnlichkeit: In obenstehender Abbildung ist zu erkennen, dass sich die Verteilungskurven beider Wahrnehmungsdistanzen in Mittelwert und Standardabweichung stark ähneln. Obwohl die Qualität der Reproduktion der erfassten Posen und die damit zusammenhängende Fähigkeit des Generators die Pose aus der Bildquelle korrekt zu reproduzieren stark schwankt und sich teilweise

auch in einem schlechten hohen Wertebereich wiederfindet, ist diese dennoch zwischen den voneinander unabhängigen neuronalen Modellen des Goku3D158-GAN und des Goku2D158-GAN vergleichbar. Dies deutet erneut darauf hin, dass die Dimensionalität bei der Erstellung der Ausgangsdaten im konkret durchgeführten Versuch wenig bis keinen Einfluss ausübt.

GAN Training: Die Betrachtung quantitativer Kenngrößen schließe ich ab mit dem GAN Lernfortschritt der neuronalen Strukturen für die Fähigkeit einen Son Goku in einer im Trainingsdatensatz enthaltenen Pose zu erstellen, der entweder einer Zeichnung oder einem Rendering ähnelt.

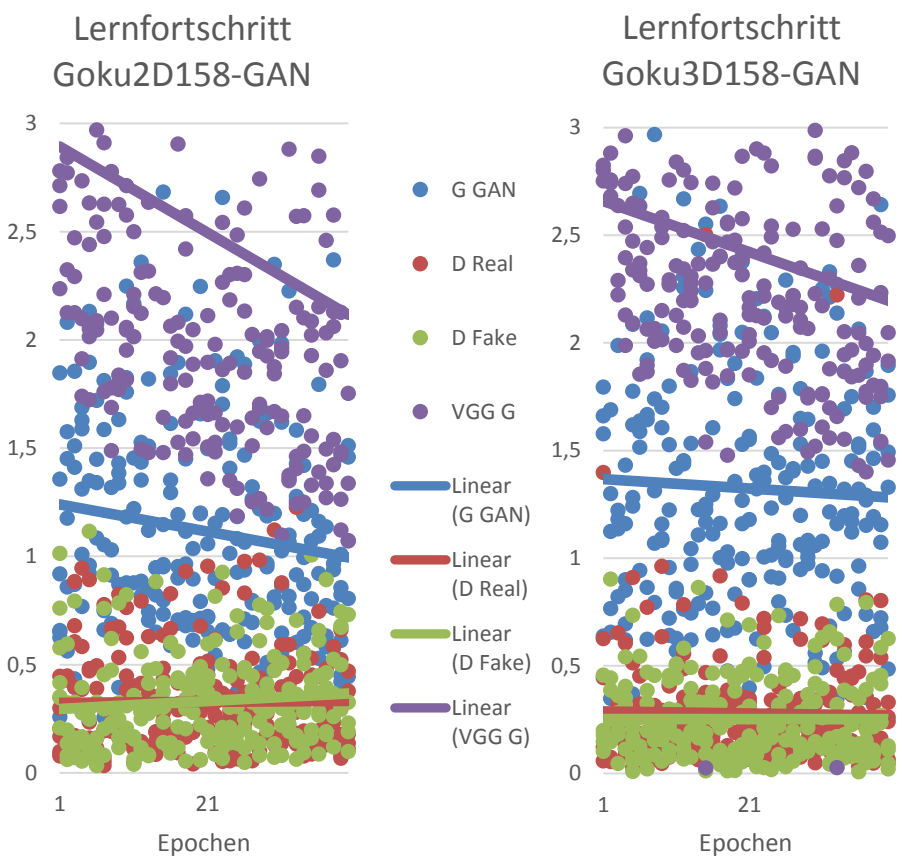


Abbildung 53: Vergleich der Lernfortschritte durch Regressionstrendlinien beim Synthetisieren von gezeichnetem und gerendertem Ausgangsmaterial über 40 Trainingsepochen anhand der Kenngrößen der Fehlerfunktionen für Generator, Diskriminator, sowie einer Wahrnehmungsfehlerfunktion (VGG G)

Stabiles Wechselspiel: Auch bei der Betrachtung der Fehlerfunktionen, wie sie mit den Werkzeugen des Frameworks Pytorch während des Trainings mitgeschrieben werden können, bietet sich ein ähnliches Bild wie bei den vorangegangenen Vergleichen. Das GAN Training ist ein Wechselspiel zwischen

Diskriminator und Generator. So versucht der Generator ein so täuschend echtes Bild des Charakter Son Goku zu erstellen, dass der Diskriminator dieses nur sehr schwer als Fälschung erkennen kann. Diese Leistung des Generators wird als G GAN dargestellt. Wenn der Generator eine gute Fälschung erstellt, ist seine Fehlerfunktion gering, wohingegen die Fehlerfunktion des Diskriminators einen hohen, i.e. schlechten, Wert ausgibt. Aber auch der Diskriminator lernt dazu. Jede Runde lernt er, ausgedrückt im wert D fake, den vom Generator genutzten Plan bei der Erstellung des synthetisierten Bildes besser aufzudecken, wodurch die Fehlerfunktion vom Generator in dieser Runde wiederum einen schlechteren Wert ausgibt. Ein sich so in stabilem Wechselspiel gegenseitig trainierendes neuronales Netzwerk optimiert sich jede Runde, obwohl die relativ konstant erscheinenden Fehlerfunktionen dies nicht auf den ersten Blick verraten (Brownlee, 2019).

Wahrnehmungsfehler: Der VGG G loss ist dagegen ein einfacher zu interpretierender Fehlerwert. Das in „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) verwendete VGG19 Netzwerk ist ein faltungscodiertes neuronales Netzwerk, welches bereits fertig trainierte 19 neuronale Aktivierungsebenen mitbringt, die darauf spezialisiert sind, Bildinhalte zu lokalisieren und zu erkennen (Simonyan & Zisserman, 2015). In der letzten Abbildung beschreibt der VGG G Wert die Einschätzung des VGG19 Netzwerks darüber, wie sehr ein aktuell vom Generator G produziertes Bild einem realen Foto oder Bild ähnelt. Dieser Fehler entspricht also, ähnlich dem Konzept hinter der Metrik LPIPS, in gewisser Weise der menschlichen Wahrnehmung. Deshalb wird die Fehlerfunktion auch als „perceptual loss“, i.e. Wahrnehmungsfehler, bezeichnet. Die Lineare Regression in dem Diagramm zum Trainingsfortschritt der Netzwerke Goku2D158 und Goku3D158 zeigt daher gut, wie beide GAN Systeme die Bilder nach jeder Epoche immer „realistischer“ erstellen. Obwohl beide Netzwerke in ihrer Leistung sich ähnlich entwickeln, verbessert sich der VGG G Fehlerwert bei dem 2DGoku158-GAN Training etwas signifikanter positiv.

Grundsätzliche Eignung: Die Betrachtung der Lernfortschritte führt mich zu der Annahme, dass sowohl zweidimensionale Zeichnungen als auch 3D Renderings grundsätzlich geeignete Repräsentationen eines Charakters sind, um mittels eines GAN Lernziels in einem neuronalen Netzwerk abgebildet zu werden.

7.7.2 Qualitative Beobachtungen:

Abbildung 54: Schwachstellen bei den generierten Son Gokus mittels Goku3D158-GAN (links) und Goku2D158-GAN (rechts) anhand des Musikvideos „That's What I Like" (Mars & Lia, 2017) als Bewegungsquelle

Sichtbare Unterschiede: Bei einer visuellen Betrachtung nach vollständigem Bewegungstransfer der Bewegung des Tänzers aus dem Musikvideo „That's What I Like" (Mars & Lia, 2017), werden mehrere Unterschiede zwischen den Ergebnissen mittels den zwei fertig trainierten neuronalen Netzwerken sichtbar. Ein Unterschied liegt beispielsweise darin, wie häufig ein plausibles Gesicht synthetisiert wird. In der obigen Abbildung erkennt man eine Pose, in der das Netzwerk Goku3D158-GAN mit dieser Aufgabe überfordert ist.

Stiltreue: Auf die Frage, ob der synthetisierte Stil der gezeichneten und gerenderten Darstellung des Charakters dem Originalstil treu bleibt, kann bei den meisten vom Netzwerk halluzinierten Bildern mit ja geantwortet werden. Zumindest im Fall des Charakters Son Goku werden sowohl Eigenschaften spezifisch für eine Zeichnung als auch für ein Rendering des Charakters durch das neuronale Netzwerk gelernt. Dies ist besonders gut zu beobachten an der Darstellung von Kleidung und Farbflächen. Zeichnungen sind dadurch, dass sie bei klassischer Animation oft für jedes Einzelbild neu gezeichnet werden, von Natur aus einer größeren Schwankung unterlegen. Bei dem 3D Modell besteht im Original hingegen eine konstante dreidimensionale Repräsentation. Eventuell fällt durch das Fehlen eines klar definierten Modells bei den durch das Netzwerk halluzinierten Zeichnungen ein kurzzeitig abweichender Stil auch weniger auf, als bei dem 3D Rendering.

Wiedererkennbarkeit: Sowohl der 3D Son Goku als auch der 2D Son Goku entsprechen in generierter Form über weite Strecken ihren Erscheinungsbildern im originalen Zieldatensatz. Häufig sind jedoch das Gesicht oder ganze

Körperpartien verformt oder fehlen komplett, wodurch der Charakter unkenntlich gemacht wird. Dennoch nehme ich an, dass jemand der die fiktive Animefigur kennt, diese auch nur anhand der generierten Bilder wiedererkennen könnte.

Temporale Kohärenz: In Abhängigkeit der Zeit ist zu beobachten, dass die Bewegung zwar korrekt vom Tänzer auf Son Goku übertragen wird, jedoch häufig Sprünge oder unvollständige Synthetisierungen auftreten. Dies ist in beiden synthetisierten Zielsequenzen gleichermaßen der Fall. Dadurch werden ganze Teilsequenzen des Bewegungstransfers in ihrer Qualität stark gemindert. Dies kann an mehreren Problemen liegen. Ich nehme an, dass die im Zieldatensatz enthaltenen Erscheinungsformen, Perspektiven, sowie die Neigung, den Charakter großteils von vorne zu zeigen für die zeitlichen Inkonsistenten im Endergebnis eine Rolle spielen. Ähnliche und gleichmäßige Zieldaten, oder eine auf diese Schwierigkeit rücksichtnehmende GAN Funktionsdefinition, wie sie in „Everybody Dance Now“ von Chan et al. präsentiert wird (Chan et al., 2018, S. 4), könnten dieses Problem lindern. Auch bei der Posenerfassung vermute ich großes Potential durch das Verringern der Anzahl an gänzlich oder teilweise fehlenden Erfassungen das synthetisierte Endergebnis positiv zu beenflussen. Im direkten Vergleich mit dem ebenfalls auf dieselbe Art trainierten Realvideo-Generators (basierend auf SebLA158) ist jedoch kaum ein Unterschied in der zeitlichen Kohärenz des erfolgten Transfers auszumachen. Ein visueller Vergleich der drei Bewegungstransfers mittels Goku2D158, Goku3D158, sowie SebLA158, ist in der nachfolgenden Abbildung zu sehen.

Abbildung 55: Qualitativer visueller Vergleich der Bewegungstransfers basierend auf gerenderten, gezeichneten, sowie einem realen Zieldatensatz (von links nach rechts). In der Mitte ist der Tänzer des Musikvideos (Mars & Lia, 2017) und ganz rechts das zu seiner Pose gehörende Label des Positionserfassers abgebildet.

7.8 Weiterführende Versuche und Ergebnisse

Um die quantitativen Daten aus den vorangegangenen Kapiteln nochmals besser beurteilen zu können, ist ein Vergleich mit dem Ergebnis der Metriken bei dem auf dieselbe Art und Weise zusammengestellten Zieldatensatz SebLA158 nützlich. Der Datensatz besteht ebenfalls aus 158 Posen, 3599 Bildern und einer einzelnen Person vor einem Greenscreen, aufgezeichnet im Videostudio der FH St. Pölten. Das dazugehörige neuronale Netzwerk wurde auch von derselben Initialisierung ausgehend über 40 Epochen hinweg trainiert.

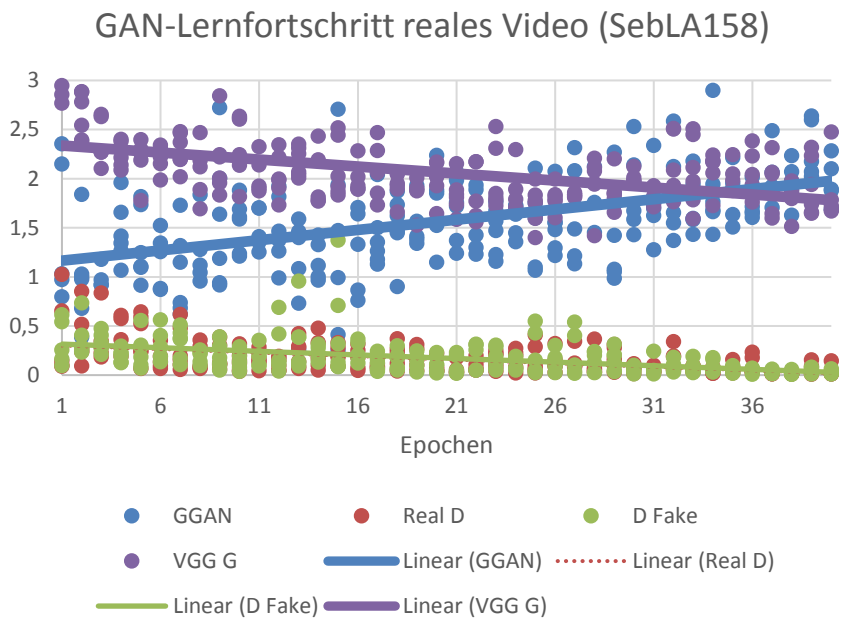


Abbildung 56: Lernfortschritt von SebLA158-GAN über Epoche 1-40

Momentaufnahme: Bei dieser Abbildung ist zu beobachten, dass auch bei realem Video nach 40 Epochen Training die lineare Regressionstrendlinie des Wahrnehmungsfehlers VGG G auf eine gelernte Verbesserung der Qualität deutet. Generator- und Diskriminatorenfehler befinden sich jedoch (noch nicht) in einem stabilen Equilibrium, was ich so deute, dass noch Potential besteht, das GAN in weiteren Trainingsdurchläufen zu verbessern.

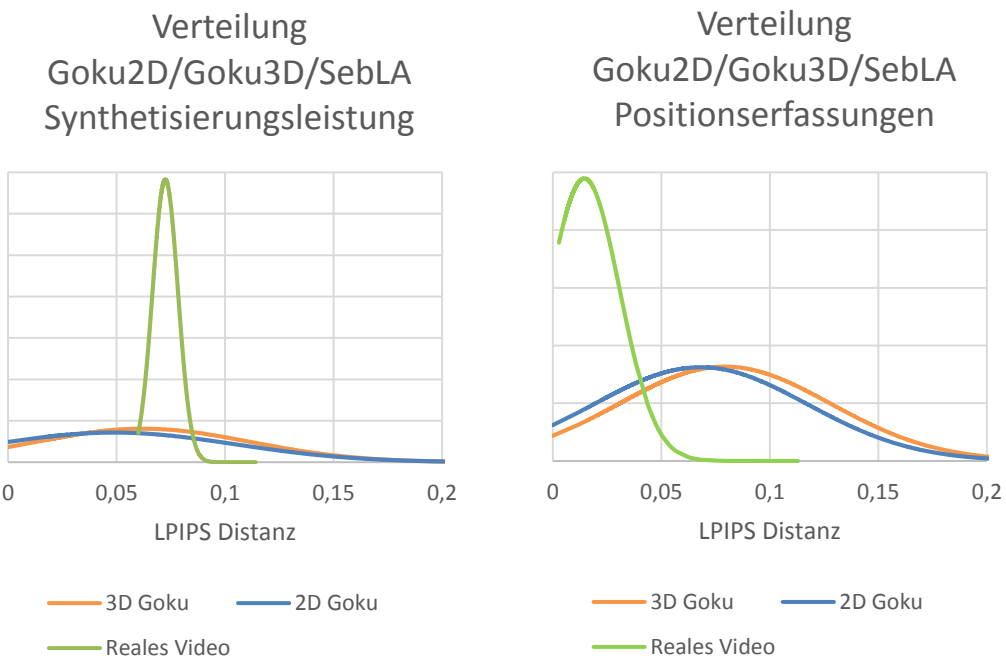


Abbildung 57 und Abbildung 58: Die Verteilung der veglichenen Synthetisierungsleistungen bei identem Trainings- und Testdatensatz (links) sowie die Verteilung der verglichenen Posenerfassungsleistung nach für jedes Netzwerk 40 Trainingsepochen. Maximum weiter links und schmälere Verteilung ist besser. 3D Goku = Goku3D158-GAN, 2D Goku = Goku2D158-GAN, Reales Video = SebLA158-GAN

Synthetisierungsähnlichkeit in schmalem Band: Auch ein Blick auf die Verteilungsfunktionen bei der Ermittlung der Synthetisierungsleistung gibt einen Einblick wie sich diese zwischen dem Charakter Son Goku und einer realen Person vergleicht. Ebenso die Leistung einer erneuten Posenerfassung mittels OpenPose portable (Hidalgo et al., 2017/2020) offenbart einen beträchtlichen Unterschied des GAN Lernfortschritts anhand der zu der geringen Schwankungsbreite der Fehlerfunktionen für Diskriminator und Generator passenden Punktwolke der vorherigen Trendliniengrafik. Hervorzuheben ist dabei, dass die synthetisierten Bilder stärker von den Originalbildern abweichen. Dies ist im linken Diagramm Anhand der Metrik LPIPS abzulesen (Zhang et al., 2018/2020). Dies schlechteren Werte verteilen sich jedoch in einem sehr viel

schmäleren Wertebereich als dies bei Son Goku passiert. Meine Annahme diesbezüglich ist, dass der Hintergrund, auch wenn er ebenso grün gehalten ist wie bei Goku2D158 und Goku3D158, dennoch Textur enthält, welche eine zusätzlich lernbare Ebene für das GAN System darstellt.

Unterschiedliche Eignung für OpenPose: Anders als bei der Betrachtung der Synthetisierungsqualität bietet sich bei der daneben abgebildeten Reproduzierbarkeit der Originallabels aus dem synthetisierten Zieldatensatz SebLA158 ein wesentlich vielversprechenderes Bild als bei beiden neuronalen Generatorennetzen des fiktiven Charakter Son Goku. Der Wahrnehmungsfehler ist niedrig und in einem schmalen Schwankungsbereich verteilt, was darauf deutet, dass OpenPose portable (Hidalgo et al., 2017/2020) die Posen leichter und hochwertiger aus dem nach Epoche 40 synthetisierten Datensatz erneut extrahieren kann. Der Grund dafür ist nicht definitiv festzumachen, jedoch liegen wieder mehrere Annahmen nahe. Der Positionserfasser OpenPose wurde auf einem Datensatz von Menschen trainiert (Cao et al., 2018), welcher auch besonders von Menschen häufig eingenommene Posen gewichtet. Die davon abweichende Erscheinung und auch Darstellung teils außergewöhnlicher Perspektiven und Posen bei den Datensätzen, welche den Charakter Son Goku zeigen, bieten eine schwierigere Ausgangssituation für den Algorithmus zur Posenerfassung, wodurch es womöglich zu dem höheren Fehler und der größeren Schwankung desselben bei Goku2D158 sowie Goku3D158 kommt.

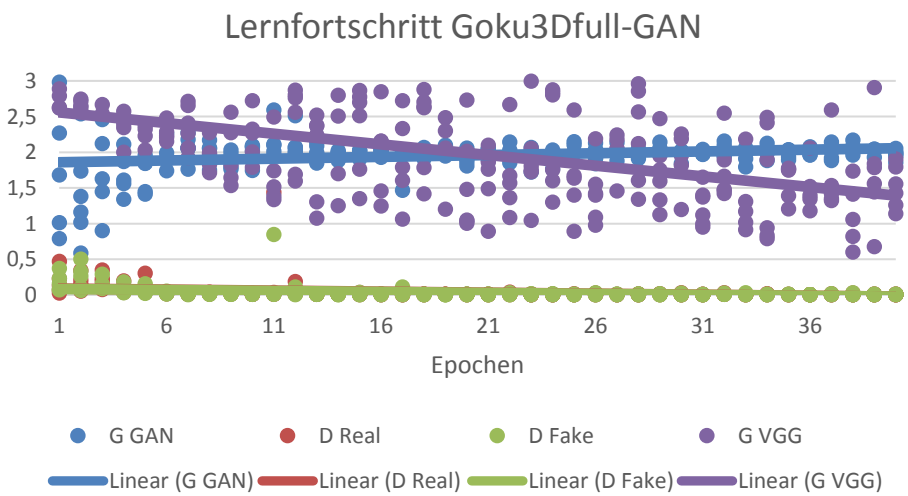


Abbildung 59: GAN Lernfortschritt bei auf 3599 erweiterter Posenanzahl über 40 Epochen mittels dem Datensatz Goku3Dfull

Reduktion der Varianz: Die Frage der Auswirkung einer erhöhten Posenanzahl bei gleichbleibender Bildanzahl versuche ich mit dem Trainings- und

Testdurchlauf eines GAN-Trainings über 40 Epochen anhand des in der Posenanzahl nicht reduzierten gleichen Videos wie für Goku3D158, i.e. mittels des Datensatzes Goku3Dfull, zu erarbeiten. In der vorangegangenen Abbildung ist der Lernerfolg dieses Netzwerks zu sehen. Auf den ersten Blick fällt auf, dass der Wahrnehmungsfehler über die Trainingsdauer fällt, der Diskriminator sich jedoch besser entwickelt als der Generator. Da es dem Generator bei zunehmender Trainingsdauer nicht mehr gelingt, einzelne „fake“ Synthetisierungen dem Diskriminator als echt erscheinen zu lassen, könnte dies darauf hindeuten, dass das Generatorennetzwerk entweder sehr gut gelernt hat, was den 3D gerenderten Charakter Son Goku ausmacht, oder sich in einem falschen mathematischen Bereich bewegt, der ihm aber als geeignet für weitere Synthetisierungen erscheint.

Qualitative Betrachtung: Ein Blick auf eine visuelle Gegenüberstellung der von den Netzwerken Goku3D158-GAN (links) und Goku3Dfull-GAN (rechts) synthetisierten Bilder eines tatsächlichen Bewegungstransfers anhand des Musikvideos „That’s What I Like“ (Mars & Lia, 2017) zeigt augenscheinlich keine Verbesserung, sondern sogar eher eine schlechtere Generalisieriungsleistung des auf alle Posen des Ausgangsvideos über 40 Epochen trainierten Goku3Dfull-GAN.

Abbildung 60: Visuelle Gegenüberstellung von zwei nach identem Label generierten Testbildern der neuronalen Netzwerke Goku3D158-GAN und Goku3Dfull-GAN nach je 40 Trainingsepochen überlagert über die Bewegungsquelle „That’s what I like“ (Mars & Lia, 2017)

Annahme GAN Training: Der Grund für diese nicht der Erwartung an eine erhöhte Posenanzahl entsprechende verbesserte Generatorleistung kann in verschiedenen Eigenschaften der verwendeten Daten sowie der Softwareelemente zum Bewegungstransfer begründet liegen. Ich vermute jedoch, dass es beim Training des Goku3Dfull-GAN zum bereits im Kapitel zu neuronalen Netzen beschriebenen Overfitting, i.e. der Überanpassung an die Trainingsdaten, gekommen ist. Rein visuell ist nach meiner Einschätzung bei diesem konkreten

Datensatz des gerenderten Charakters Son Goku keine Verbesserung durch die Erhöhung der Posenanzahl am Endergebnis eingetreten.

Normalisierungskompromiss: Den Schritt der Normalisierung habe ich anhand der Anpassung mehrerer Bewegungsquellen durch den Programmteil „normalization.py“ getestet (Wu et al., 2018/2020). Obwohl in meinen Versuchen die Position und Skalierung im Bild grundsätzlich erfolgreich an die im Zieldatensatz enthaltenen Posen-Positionen und Posen-Skalierungen des Zielcharakters angepasst werden konnten, ist ein unerwünschter Nebeneffekt aufgetreten. Wenn die Person der Bewegungsquelle kurz den Boden verlässt, z.B. bei einem Sprung, oder andere große Translationen im Raum des Bildes unternimmt, werden durch die Normalisierung diese Bewegungen verfälscht. Dies passiert insofern, als dass z.B. bei einem Sprung die vertikale Skelettposition horizontal auf einer Ebene gehalten wird. Es kommt zu einem unnatürlichen „Am-Boden-kleben-bleiben“. Dies wird in der nächsten Abbildung veranschaulicht.

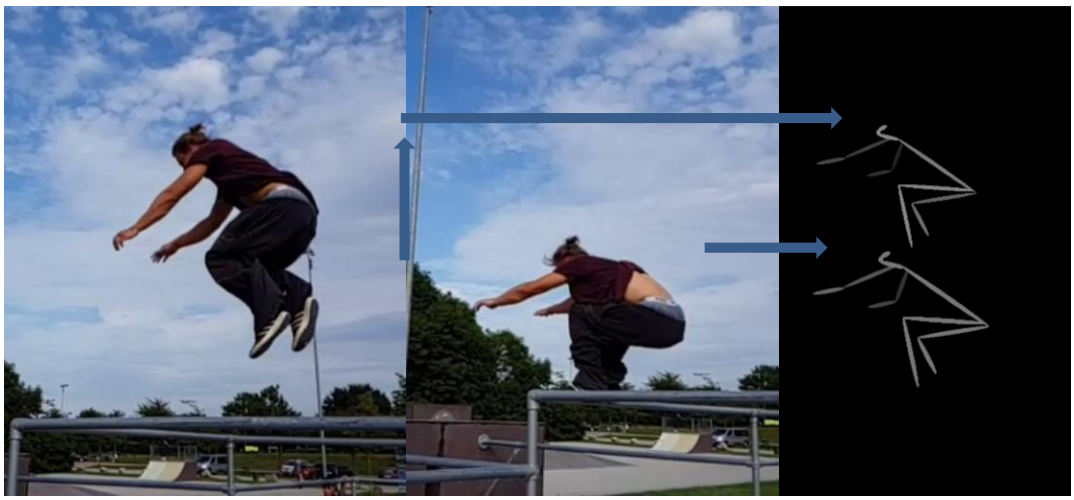


Abbildung 61: Durch einen Sprung (links) entstehender vertikaler Fehler der Posen-Position im Raum des Bildes durch Verschiebung auf der y-Achse beim Normalisieren der Posenerfassung einer Sprungbewegung. Bewegungsquellbild (Mitte) für die Labels (rechts) der Aufnahme „Parkour 1 – Präzi“ und die originale Posenerfassung (rechts unten) im Vergleich zur normalisierten Posenerfassung (rechts oben)

Grenzen der Posenerfassung: Wie die teilweise unvollständigen Posenerfassungen aus dem Musikvideo des Hauptversuchs bereits vermuten ließen, nimmt die Qualität dieser bei komplexeren Bewegungen, wie z.B. Überkopfbewegungen, weiter ab. In der folgenden Abbildung kann dies anhand eines Bewegungstransfer mittels der Bewegungsquelle „Aufnahme Breakdance“ beobachtet werden.

Abbildung 62: Aus einer unvollständigen Posenerfassung der komplexen Bewegungsquelle „Aufnahme Breakdance“ (links) resultierende unvollständige Synthetisierung (rechts)

Kontrastveränderungen: Sowohl für den Datensatz Goku3D158 als auch für Goku2D158 führe ich über je 8534 Aktualisierungen des neuronalen Gesichtsoptimierungsmodells ein solches Training durch. Das Ergebnis ist in der nächsten Abbildung dargestellt. Wie daran zu erkennen ist, führt das Training zu einer Verbesserung der Gesichtsgenerierung, jedoch in der konkreten Umsetzung in der Softwarelösung „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020), wie sie auf meinem Rechner zum Einsatz kommt, auch zu einer unerwünschten Veränderung der Tonwerte und des Kontrasts. Da die genaue Evaluierung der Gesichtsoptimierung nicht Teil der Forschungsziele dieser Diplomarbeit ist, werden auf der Basis der Beobachtung der subjektiv unerwünschten Veränderungen in den das Gesicht enthalteneden Bildausschnitten keine weiteren Testdurchläufe oder Optimierungsschritte des Gesichts unternommen. Ein gut funktionierendes Gesichtsoptimierungssystem ist jedoch für zukünftige Versuche durchaus interessant. Dass dieses grundsätzlich funktioniert, wird in dem Paper „Everybody Dance Now“ bewiesen (Chan et al., 2018, S. 4–7).

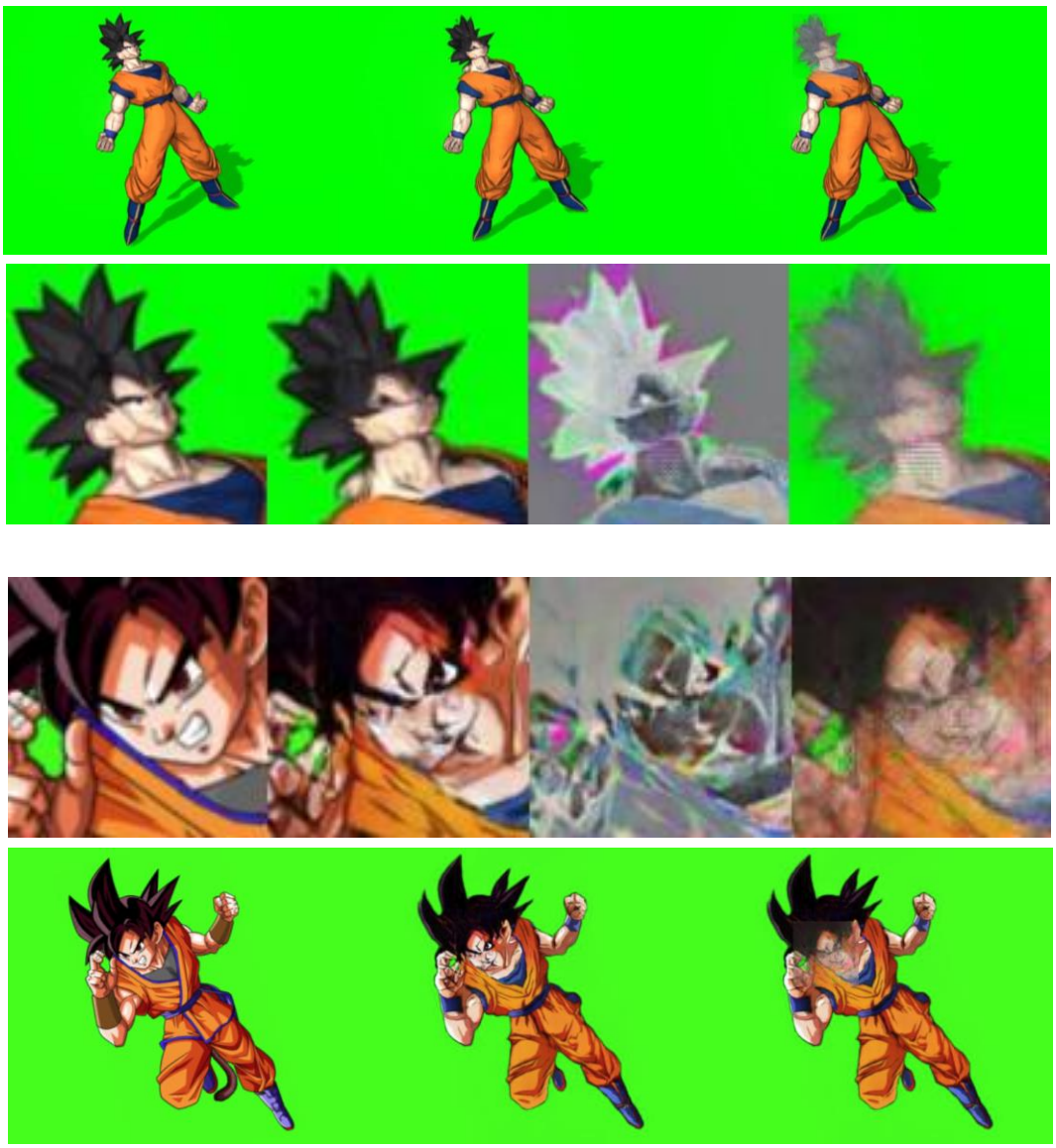


Abbildung 63: Bildveränderungen durch die Gesichtsoptimierungs-GANs, jeweils bei PyTorch Modell #8352, aufbauend auf der Leistung der Generatorennetzwerke von Goku3D158-GAN (oben), sowie Goku2D158-GAN (unten) nach 40 Epochen

8 Fazit und Ausblick

Bewegungstransfer verlangt nach hochwertiger Bewegungserfassung. Motion capture bietet eine etablierte, jedoch teils aufwändige Herangehenesweise an diese Aufgabe. Die Möglichkeiten zur Durchführung von Motion Capture sind vielfältig in ihrer Leistungsfähigkeit, ihrem notwendigen Aufwand bei der Aufzeichnung, sowie der jeweiligen verbundenen Kosten. Des Weiteren liefert klassisches Motion Capture Werkzeuge, die besonders für die 3D Charakteranimation geeignet sind. Klassische Animationsprinzipien von Zeichentrickanimationen sind ebenso noch nicht einfach als Gestaltungsmittel mit Bewegungserfassungen zu vereinen.
Außerdem ist es bei der Anwendung klassischer Motion Capture Daten und Werkzeuge häufig schwierig, die reduzierte Dimensionalität eines gezeichneten fiktiven Charakters zu erhalten. Häufig wird sich lediglich einer dem 2D gezeichneten Charakter ähnelnder 3D Repräsentation bedient.
Auf die Frage, wie dem Stil der TrickfilmzeichnerInnen treu geblieben werden kann, ohne den Schritt in die 3D Sphäre unternehmen zu müssen und dennoch auf menschliche Bewegungsdaten zurückgreifen zu können, liefern Forschungsarbeiten wie „Everybody Dance Now“ (Chan et al., 2018) an der Schnittstelle des Computersehens und des maschinellen Lernens eine vielversprechende Grundlage. Fortschritte in der Erfassung menschlicher Bewegung aus einfachen monoskopischen Videos, befeuert durch die zunehmende Relevanz der Sammlung und Auswertung von Videodaten in Bereichen, die über die Unterhaltungsindustrie hinausgehen, bieten in Kombination mit Fortschritten im Bereich maschinelles Lernen mit tiefen faltungscodierten neuronalen Netzwerkarchitekturen die Basis für den hier durchgeführten Bewegungstransfer auf den gezeichneten Charakter Son Goku.
Die Ergebnisse meiner Versuche über 2D Bewegungstransfer sind zusammenzufassen anhand der für den spezifischen Animecharakter Son Goku beobachteten Einflussgrößen.
Mittels der Softwarelösung „EverybodyDanceNow reproduced in pytorch” (Wu et al., 2018/2020) lässt sich durch die integrierte Lösung zur Bewegungserfassung, i.e. OpenPose (Cao et al., 2018), die Pose einer Person erfassen. Diese extrahierte Pose kann für sich genommen als Basis für den Bewegungstransfer dienen, oder in Kombination mit dem Video, aus dem diese Pose extrahiert wurde, für das maschinelle Lernen des Zusammenhangs zwischen der Pose und der durch diese bedingte konkrete Erscheinung der Person die GAN Lerngrundlage bilden. Eine Evaluierung auf 100 beliebte Zeichentrickcharaktere hat jedoch eine schlechte Eignung des Posenerfassungsalgorithmus OpenPose für 2D Zeichentrickcharaktere ergeben. Hier ist weitere Forschung angebracht, wie diese Posenerfassung bei einem größeren Spektrum von Zeichentrickcharakteren in

Zukunft funktionieren kann. Auch zeitliche Inkohärenz in der Abfolge der Posenerfassungen der Bewegungsquelle beeinflussen die Ergebnisse meines praktischen Versuchs negativ. Nach der Betrachtung der Kenngrößen des neuronalen Trainings messe ich dem Einfluss der zeitlich und momentan korrekten Posenerfassung das größte Potential für zukünftige Verbesserungen am 2D Bewegungstransferprozess auf Trickfilmcharaktere bei.
Der Zwischenschritt der Normalisierung wird durch die Erweiterung der Positionen des Zielcharakters im Bild mittels einer zufälligen Verschiebung nach jedem Einzelbild bei meinem Versuch umgangen. Dadurch kommt es zu keinen unerwünschten Nebeneffekten, wie einem Am-Boden-kleben-bleiben des generierten Charakters z.B. bei einem Sprung. Dennoch wäre für die Übertragung der Bewegung zwischen sehr verschiedener Anatomie zweier Personen oder Charaktere, auch im Hinblick auf die Animationsprinzipien (z.B. Squash-and-Stretch), weiterführende Forschung reizvoll.
Der Hauptversuch zeigt, dass bei dem Animecharakter Son Goku von der maschinell Lernenden Softwarelösung „EverybodyDanceNow reproduced in pytorch“ (Wu et al., 2018/2020) sowohl gerenderte als auch gezeichnete Eigenschaften in vergleichbarer Qualität gelernt werden können. Gezeichnete Charaktere können also zu ähnlichen Lernergebnissen führen wie gerenderte. Eine weiterführende Untersuchung mit anderen 2D Charakteren, sowie eine eigene Untersuchung, ob die Generalisierungsfähigkeit des Generatornetzwerks auf neue Posen zwischen den aus Zeichentrick-Charakteren und gerenderten Charakteren bestehenden Zieldatensätzen in verschiedener Qualität besteht, wäre außerdem reizvoll. Besonders die Frage der Generalisierungsfähigkeit betreffend ist jedoch mit einem erheblichen Aufwand bei der Konzeption und Erstellung vergleichbarer Validationsdatensätze zu rechnen. Dass die Gestaltung des maschinell lernenden Trainingsprogramms anhand der GAN Optimierungsvorgabe grundsätzlich bestehende Zusammenhänge zwischen strukturierten Zusatzdaten und ihren visuellen Zielpendants als Netzwerkmodell abbilden kann, von welchem auch auf neue ungesehene Daten generalisiert werden kann, ist bereits von Chan et al. glaubhaft dargestellt und deckt sich auch mit meinen Beobachtungen in den Ergebnissen der Transfers (Chan et al., 2018).
Die Dauer der Durchführung eines Bewegungstransfers ist aktuell noch sehr lange. Das Durchlaufen des Trainings für einen Charakter über 40 Epochen dauert auf meinem Rechner trotz eines leistungsfähigen Grafikprozessor über 12 Stunden. Dabei ist bei dem Training von neuronalen GANs mit Lernfortschritten bis Epoche 300 zu rechnen (Brownlee, 2019). Auch die Erstellung einer Bewegungsquelle, die Normalisierung zwischen Quelle und Ziel, sowie das Training eines Gesichtsoptimierungsnetzwerks dauern auf meinem System jeweils mehrere Stunden. Die Notwendigkeit der Erstellung eines geeigneten Datensatzes des Zielcharakters in ausreichend vielfältigen Posen vor einem einzigen oder keinem Hintergrund stellt einen großen Aufwand dar. Ansätze, wie sie in „Few-shot Video-to-Video Synthesis“ (T. Wang et al., 2019) vorgestellt werden, sind daher spannend für weiterführende Forschung und Versuche. Die Möglichkeit, von

nur wenigen Posen eines gezeichneten Charakters auf eine Vielzahl von möglichen darstellbaren Zielposen schließen zu können, ohne für jeden Charakter ein eigenes Netzwerk von Grund auf trainieren zu müssen, würde die Machbarkeit von Bewegungstransfer auf eine Vielzahl von Zeichentrickfiguren erweitern.
Der aktuelle Stand der Möglichkeit zum Bewegungstransfer mittels „EverybodyDanceNow reproduced in pytorch“ ist grundsätzlich geeignet für 2D Charaktere. An vielen Stellen der Software müssen für eine einfache Nutzbarkeit jedoch noch Optimierungen vorgenommen werden, damit am Ende auch, zeitlich, strukturell und qualitativ einfacher Bewegungstransfer auf Trickfilmcharaktere möglich ist. Die von mir zum jetzigen Zeitpunkt als am relevantesten eingeschätzten optimierbaren Prozesse sind die Posenerfassung für 2D Charaktere, der Umgang mit einer geringen Anzahl an Zielcharakterbildern, sowie die Miteinbeziehung von zeitlicher Glättung in das GAN Lernziel.

Literaturverzeichnis

Ackermann, N. (2018, Dezember 13). *Artificial Intelligence Framework: A Visual Introduction to Machine Learning and AI*. Medium. https://towardsdatascience.com/artificial-intelligence-framework-a-visual-introduction-to-machine-learning-and-ai-d7e36b304f87

AdamKK. (2014, September). *Facial motion capture hardware and software—What are you experiences?* polycount. https://polycount.com/discussion/139369/facial-motion-capture-hardware-and-software-what-are-you-experiences

Arcas, B. A. y. (2016, Juni). *How computers are learning to be creative*. https://www.ted.com/talks/blaise_aguera_y_arcas_how_computers_are_learning_to_be_creative

Baddar, W. J., Gu, G., Lee, S., & Ro, Y. M. (2017). Dynamics Transfer GAN: Generating Video by Transferring Arbitrary Temporal Dynamics from a Source Video to a Single Target Image. *ArXiv:1712.03534 [Cs]*. http://arxiv.org/abs/1712.03534

Bansal, A., Ma, S., Ramanan, D., & Sheikh, Y. (2018). Recycle-GAN: Unsupervised Video Retargeting. *ArXiv:1808.05174 [Cs]*. http://arxiv.org/abs/1808.05174

Barsegyan, A. (2017, Februar 16). *Getting ready for the first production MoCap session and some motion capture examples—CGI coffee*. http://cgicoffee.com/blog/2017/02/getting-ready-for-the-first-production-mocap-motion-capture-session

Bettilyon, T. E. (2019, Juni 12). *Deep Neural Networks As Computational Graphs*. Medium. https://medium.com/tebs-lab/deep-neural-networks-as-computational-graphs-867fcaa56c9

Brandt, M. (2018). *Das Potential von AR schlummert noch*. https://ezproxy.fhstp.ac.at:2081/infografik/15313/geschaetzter-weltweiter-absatz-von-augmented-und-virtual-reality-geraeten/

Brandt, M. (2019, Februar 13). *Infografik: Künstliche Intelligenz rechnet sich*. Statista Infografiken. https://ezproxy.fhstp.ac.at:2081/infografik/16992/umsatz-der-in-deutschland-durch-ki-anwendungen-beeinflusst-wird/

Bregler, C., Covell, M., & Slaney, M. (1997). Video Rewrite: Driving visual speech with audio. *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '97*, 353–360. https://doi.org/10.1145/258734.258880

Brodie, M., Walmsley, A., & Page, W. (2008). *Fusion motion capture: A prototype system using inertial measurement units and GPS for the biomechanical analysis of ski racing*. *1*, 13.

Brownlee, J. (2019, Juli 7). How to Identify and Diagnose GAN Failure Modes. *Machine Learning Mastery*. https://machinelearningmastery.com/practical-guide-to-gan-failure-modes/

Burkov, A. (2019). *The hundred-page machine learning book*. Andriy Burkov.

Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., & Sheikh, Y. (2018). OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *ArXiv:1812.08008 [Cs]*. http://arxiv.org/abs/1812.08008

Carlson, N. (2020, Januar 2). *New Pillow version (7.0.0) breaks torchvision (ImportError: Cannot import name „PILLOW_VERSION" from 'PIL') · Issue #1712 · pytorch/vision*. GitHub. https://github.com/pytorch/vision/issues/1712

Chan, C. (2018, August 22). *Everybody Dance Now*. https://www.youtube.com/watch?v=PCBTZh41Ris

Chan, C., Ginosar, S., Zhou, T., & Efros, A. A. (2018). Everybody Dance Now. *ArXiv:1808.07371 [Cs]*. http://arxiv.org/abs/1808.07371

Chan, C., Zhou, T., Efros, A. A., & Ginosar, S. (2019, Dezember). *Everybody Dance Now*. https://carolineec.github.io/everybody_dance_now/

Chollet, F. (2018). *Ablation Studies*. https://twitter.com/fchollet/status/1012721582148550662?lang=en

Darujati, C., & Hariadi, M. (2013). Facial motion capture with 3D active appearance models. *2013 3rd International Conference on Instrumentation, Communications, Information Technology and Biomedical Engineering (ICICI-BME)*, 59–64. https://doi.org/10.1109/ICICI-BME.2013.6698465

Download PyCharm: Python-IDE für professionelle Entwickler, von JetBrains. (2019, Juli). JetBrains. https://www.jetbrains.com/pycharm/download/

DreamWorks. (2001, Mai 18). *Shrek*. DreamWorks. http://www.imdb.com/title/tt0126029/

Features | PyCharm. (2020, Januar 11). JetBrains. https://www.jetbrains.com/pycharm/features/

FFmpeg. (2019). *Download FFmpeg*. https://www.ffmpeg.org/download.html

Fleron, M. K., Ubbesen, N. C. H., Battistella, F., Dejtiar, D. L., & Oliveira, A. S. (2019). Accuracy between optical and inertial motion capture systems for assessing trunk speed during preferred gait and transition periods. *Sports Biomechanics*, *18*(4), 366–377. https://doi.org/10.1080/14763141.2017.1409259

Foundations of trusted autonomy. (2017). Springer Berlin Heidelberg.

Free green screen videos, visual effects, dragonball z, goku, chroma key, vfx, 3d animation, 4K, hd. (2019, März 22). https://www.youtube.com/watch?v=l0j7V6ha46Q&t=16s

Geirhos, R., Michaelis, C., Wichmann, F. A., Rubisch, P., Bethge, M., & Brendel, W. (2019). *IMAGENET-TRAINED CNNS ARE BIASED TOWARDS TEXTURE; INCREASING SHAPE BIAS IMPROVES ACCURACY AND ROBUSTNESS*. 22.

Gerrish, S. (2018). *How smart machines think*. The MIT Press.

GIPHY. (2019). https://giphy.com/

GitHub. (2020, Januar). *Build software better, together*. GitHub. https://github.com

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. *ArXiv:1406.2661 [Cs, Stat]*. http://arxiv.org/abs/1406.2661

Güler, R. A., Neverova, N., & Kokkinos, I. (2018). DensePose: Dense Human Pose Estimation In The Wild. *ArXiv:1802.00434 [Cs]*. http://arxiv.org/abs/1802.00434

Hahn, G. (2020, Januar 9). *The Greatest Cartoon Characters in TV History*. Ranker. //www.ranker.com/list/best-cartoon-characters-on-tv/greg-hahn

Hidalgo, G., Cao, Z., Wei, S.-E., Simon, T., Joo, H., Sheikh, Y., & Raaj, Y. (2020). *CMU-Perceptual-Computing-Lab/openpose* [Python; C++]. CMU-Perceptual-Computing-Lab. https://github.com/CMU-Perceptual-Computing-Lab/openpose (Original work published 2017)

Hillcrest Laboratories. (2015). *Hillcrest-Wearable-Infographic-Square-medium-1200x1080.jpg*.

ImageNet Large Scale Visual Recognition Competition (ILSVRC). (2020, Januar 12). http://www.image-net.org/challenges/LSVRC/

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2016). Image-to-Image Translation with Conditional Adversarial Networks. *ArXiv:1611.07004 [Cs]*. http://arxiv.org/abs/1611.07004

Jackson, P. (2013, Dezember 13). *The Hobbit: The Desolation of Smaug*. http://www.imdb.com/title/tt1170358/

Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual Losses for Real-Time Style Transfer and Super-Resolution. *ArXiv:1603.08155 [Cs]*. http://arxiv.org/abs/1603.08155

Kaleido AI GmbH. (2020). *Remove Background from Image*. remove.bg. https://www.remove.bg/

Koch, T., Peter, C., & Müller, P. (2019). *Das Experiment in der Kommunikations- und Medienwissenschaft: Grundlagen, Durchführung und Auswertung experimenteller Forschung*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-19754-4

L. Brunton, S. (2019, Juni 5). *Neural Network Architectures*. https://www.youtube.com/watch?v=oJNHXPs0XDk

Lamble, R. (2013, Dezember 13). *Benedict Cumberbatch interview: The Hobbit, Holmes and Smaug*. Den of Geek. https://www.denofgeek.com/movies/the-hobbit/28590/benedict-cumberbatch-interview-the-hobbit-holmes-and-smaug

Li, F.-F., Johnson, J., & Yeung, S. (2017, Mai 2). *Lecture 9: CNN Architectures*.

Mao, X., Li, Q., Xie, H., Lau, R. Y. K., Wang, Z., & Smolley, S. P. (2017). Least Squares Generative Adversarial Networks. *ArXiv:1611.04076 [Cs]*. http://arxiv.org/abs/1611.04076

Mars, B., & Lia, J. (2017, März 2). *Bruno Mars—That's What I Like (Official Video)*. https://www.youtube.com/watch?v=PMivT7MJ41M

Menache, A. (2011). *Understanding motion capture for computer animation* (2nd ed). Morgan Kaufmann.

Meta Motion. (2019, Dezember). https://metamotion.com/gypsy/gypsy-motion-capture-system.htm

Moeslund, T. B., Hilton, A., & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, *104*(2–3), 90–126. https://doi.org/10.1016/j.cviu.2006.08.002

Mündermann, L., Corazza, S., & Andriacchi, T. P. (2006). The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications. *Journal of NeuroEngineering and Rehabilitation*, 11.

Musa, S., Ziatdinov, R., & Griffiths, C. (2013). *INTRODUCTION TO COMPUTER ANIMATION AND ITS POSSIBLE EDUCATIONAL APPLICATIONS*. 25.

Nintendo. (2019). *Bedienungsanleitung Nintendo WII*.

Noitom. (2019, Dezember 2). *Perception Neuron* [Text]. Perception Neuron by Noitom. https://neuronmocap.com/products/perception_neuron

Oppermann, A. (2019, Oktober 3). *Overfitting and Underfitting in Deep Learning*. https://www.deeplearning-academy.com/p/ai-wiki-overfitting-underfitting

Pardeshi, M. (2019, August 2). *Linux or Windows? Which one is better for AI or Deep learning projects.* Medium. https://medium.com/@maheshpardeshi002/linux-or-windows-for-ai-or-deep-learning-projects-a065d18ddf2c

Perception Neuron. (2014). *Project PERCEPTION NEURON: Motion Capture, VR and VFX*. Kickstarter. https://www.kickstarter.com/projects/1663270989/project-perception-neuron

Reeves, M. (2017, Juli 14). *War for the Planet of the Apes*. http://www.imdb.com/title/tt3450958/

Richter, F. (2016). *Infographic: The Diverse Potential of VR & AR Applications*. https://www.statista.com/chart/4602/virtual-and-augmented-reality-software-revenue/

Richter, F. (2019). *Infographic: Entertainment industry revenue*. https://www.statista.com/chart/17394/entertainment-industry-revenue/

Rokoko. (2019, Dezember 5). *Rokoko | Shop*. https://www.rokoko.com/en/shop

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, *115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

Russell, S. J., Norvig, P., & Davis, E. (2010). *Artificial intelligence: A modern approach* (3rd ed). Prentice Hall.

Sanderson, G. (2017, Oktober 16). *Gradient descent, how neural networks learn | Deep learning, chapter 2*. https://www.youtube.com/watch?v=IHZwWFHWa-w&t=4s

Schmieder, J. (2014, November 28). *Andy Serkis im Interview über Performance Capture*. Süddeutsche.de. https://www.sueddeutsche.de/kultur/schauspieler-andy-serkis-rollenbesetzung-nach-typen-gehoert-der-vergangenheit-an-1.2237477

Schutt, R., & O'Neil, C. (2013). *Doing data science* (First edition). O'Reilly Media.

Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv:1409.1556 [Cs]*. http://arxiv.org/abs/1409.1556

Smith, C. (2017, September 30). *Inside BAE's Striker II – The most advanced fighter pilot helmet ever made*. BT.Com. http://home.bt.com/tech-gadgets/future-tech/inside-baes-striker-ii-the-most-advanced-fighter-pilot-helmet-ever-made-11364174261792

Statista. (2019, Oktober). *Betriebssysteme—Marktanteile weltweit bis September 2019*. Statista. https://de.statista.com/statistik/daten/studie/157902/umfrage/marktanteil-der-genutzten-betriebssysteme-weltweit-seit-2009/

Steenbrugge, X. (2019, September 13). *Face editing with Generative Adversarial Networks—YouTube*. https://www.youtube.com/watch?v=dCKbRCUyop8&list=LLvuEGS_9iylpk4meLXnYFig&index=3&t=0s

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (Second edition). The MIT Press.

Szykman, A. G., & Gois, J. P. (2014). *Evaluating the Feasibility of Low Cost Motion Capture Systems*. 4.

The MIT License | Open Source Initiative. (2020, Januar 11). https://opensource.org/licenses/MIT

Thomas, F., & Johnston, O. (1981). *Disney animation: The illusion of life* (1st ed). Abbeville Press.

Trask, A. W. (2019). *Grokking deep learning*. Manning.

Tulyakov, S., Liu, M.-Y., Yang, X., & Kautz, J. (2017). MoCoGAN: Decomposing Motion and Content for Video Generation. *ArXiv:1707.04993 [Cs]*. http://arxiv.org/abs/1707.04993

Veen, F. van, & Leijnen, S. (2016, September 14). *The Neural Network Zoo*. The Asimov Institute. https://www.asimovinstitute.org/neural-network-zoo/

Villegas, R., Yang, J., Zou, Y., Sohn, S., Lin, X., & Lee, H. (2018). Learning to Generate Long-term Future via Hierarchical Prediction. *ArXiv:1704.05831 [Cs]*. http://arxiv.org/abs/1704.05831

Vlasic, D., Adelsberger, R., Vannucci, G., Barnwell, J., Gross, M., Matusik, W., & Popović, J. (2007). Practical motion capture in everyday surroundings. *ACM Transactions on Graphics*, *26*(99), 35. https://doi.org/10.1145/1239451.1239486

Wang, T., Liu, M., Tao, A., Liu, G., Kautz, J., & Catanzaro, B. (2019). *Few-shot Video-to-Video Synthesis*. 14.

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Liu, G., Tao, A., Kautz, J., & Catanzaro, B. (2018a). Video-to-Video Synthesis. *ArXiv:1808.06601 [Cs]*. http://arxiv.org/abs/1808.06601

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Liu, G., Tao, A., Kautz, J., & Catanzaro, B. (2018b). Video-to-Video Synthesis. *ArXiv:1808.06601 [Cs]*. http://arxiv.org/abs/1808.06601

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., & Catanzaro, B. (2017). High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *ArXiv:1711.11585 [Cs]*. http://arxiv.org/abs/1711.11585

Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612. https://doi.org/10.1109/TIP.2003.819861

Warner Bros. (2006, November 17). *Happy Feet*. Warner Bros. http://www.imdb.com/title/tt0366548/

Wittek, P. (2014). *Quantum machine learning: What quantum computing means to data mining* (1. ed). Elsevier [u.a.].

Wu, P., Lin, J., Liao, Y., Qing, W., & Xu, Y. (2020). *CUHKSZ-TQL/EverybodyDanceNow_reproduce_pytorch* [Python]. CUHKSZ_TQL. https://github.com/CUHKSZ-TQL/EverybodyDanceNow_reproduce_pytorch (Original work published 2018)

Yashwardhan, J. (2018, November 14). *Tensorflow or PyTorch: The force is strong with which one?* Medium. https://medium.com/@UdacityINDIA/tensorflow-or-pytorch-the-force-is-strong-with-which-one-68226bb7dab4

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. *ArXiv:1801.03924 [Cs]*. http://arxiv.org/abs/1801.03924

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2020). *Richzhang/PerceptualSimilarity* [Python]. https://github.com/richzhang/PerceptualSimilarity (Original work published 2018)

Abbildungsverzeichnis

Tabellenverzeichnis

Formelverzeichnis