

# Computergrafik.Online

## Drehbuch Bilddatenreduktion

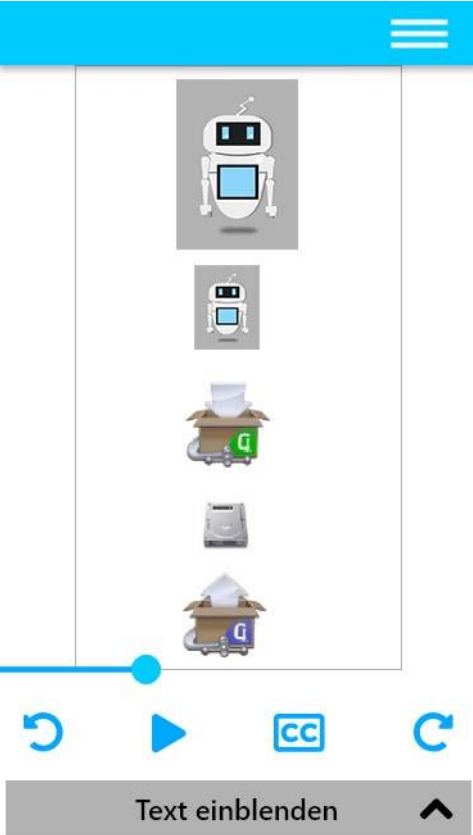
Hochschule Furtwangen University  
Fakultät Digitale Medien  
Betreut von:  
Prof. Jirka Dell'Oro-Friedl

Version: 2.5  
Letzte Änderung: 09.12.2018  
Autor: Steven Romanek

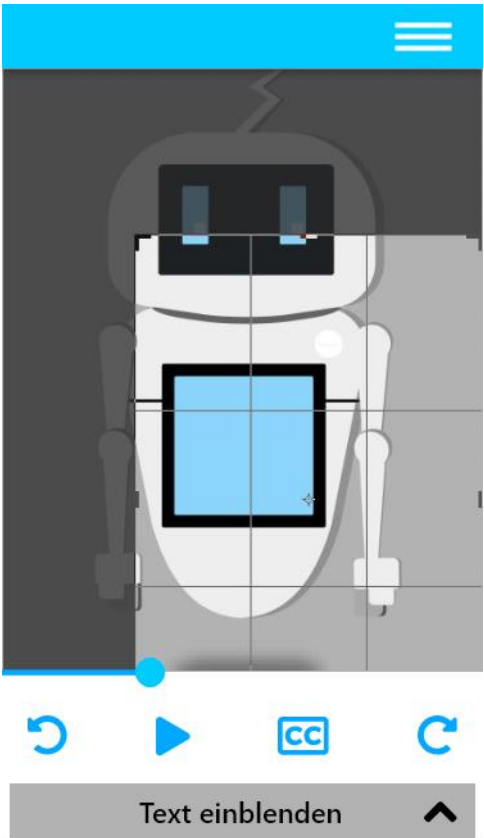
# Inhalt

6.1 (A) Einleitung.....	2
6.2 (A) Skalieren und Beschneiden .....	3
6.3 (A) Farbtiefenreduktion .....	4
6.4 (I) Farbtiefenreduktion.....	5
6.5 (A) RLE .....	6
6.6 (A) LZW .....	7
6.7 (A) Huffman-Kodierung.....	8
6.8 (A) JPG .....	9
6.9 (I) JPG .....	12
6.10 (A) GIF.....	13
6.11 (A) PNG.....	15
6.12 (A) Tipps & Tricks.....	18

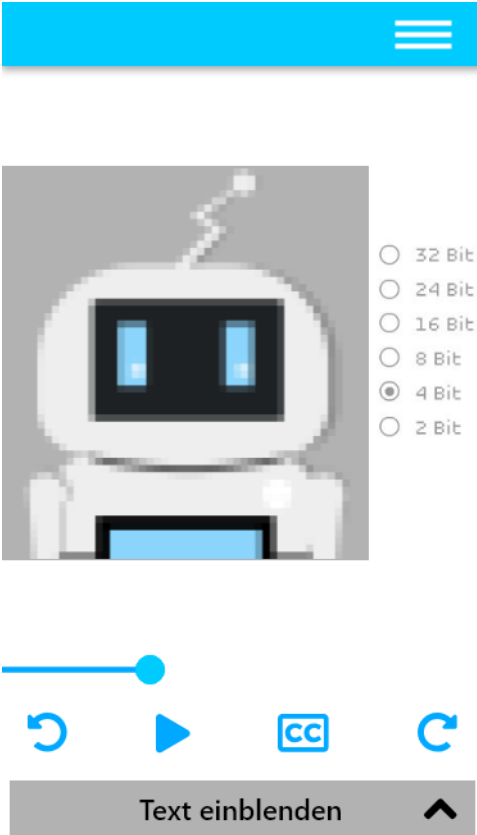
## 6.1 (A) Einleitung

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>060101</b> Beim Arbeiten mit digitalen Bildern können sehr schnell große Datenmengen entstehen. Um dennoch eine kompakte Speicherung und schnelle Übertragung zu gewährleisten, ist es häufig sinnvoll, die Bilddaten zu reduzieren.</p> <p><b>060102</b> Zunächst kann das Bild auf einen relevanten Bereich zugeschnitten, skaliert und in der Farbtiefe reduziert werden. Anschließend wird es komprimiert, also platzsparend kodiert, wobei zwischen verlustfreier und verlustbehafteter Kompression gewählt werden kann.</p> <p><b>060103</b> Nun ist das Bild bereit um abgespeichert oder verschickt zu werden. Will man es nun wiederverwenden, so wird es vorher dekomprimiert. Bei der verlustfreien Kompression kann das Bild dabei vollständig wiederhergestellt werden, während dies bei der verlustbehafteten nicht möglich ist.</p>	<p><b>060102</b> -Zuschneiden, skalieren oder Farbtiefe reduzieren</p> <p><b>060103</b> - Verlustfreie oder verlustbehaftete Kompression</p>	<p><b>060101-03</b> Die einzelnen Kompressionsschritte werden nacheinander eingeblendet.</p>

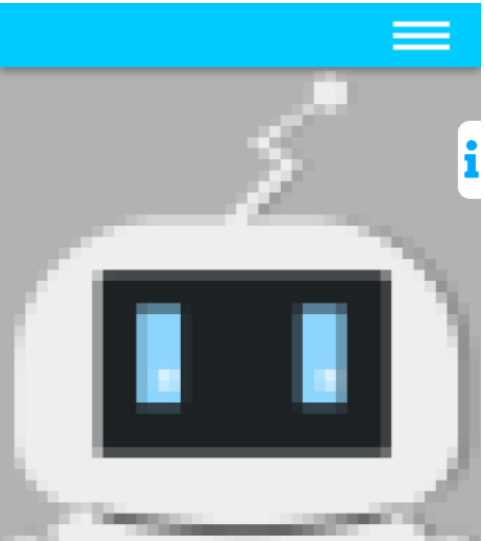
## 6.2 (A) Skalieren und Beschneiden

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>060201</b>          Bilddatenreduktion muss nicht automatisch Kompression bedeuten. Bilddaten können vorab durch einfache Beschränkungen ebenfalls reduziert werden. Zum Beispiel indem die Pixelanzahl verringert wird. Beschneidet man das Bild, und skaliert es anschließend auf eine geringere Kantenlänge, so wird die Bilddatenmenge vorab erheblich reduziert. Dabei muss aber drauf geachtet werden, welche Skalierungsmethode gewählt ist.</p> <p><b>060202</b>          Pixelwiederholung ist eine Methode, die Pixel beim Verkleinern weglässt, was unschöne Effekte auf das Bild haben kann.</p> <p><b>060203</b>          Interpolation hat in der Regel ein schöneres optisches Ergebnis zur Folge, erzeugt aber neue Farbzwischenstufen. Im Gegensatz zur Pixelwiederholung, hat das Bild nun mehr Farben als vor der Skalierung.</p>	<p><b>060201</b>          -Skalieren und bescheiden verringert die Datenmenge stark</p> <p><b>060202-03</b>          -2 Arten der Skalierung: Pixelwiederholung und Interpolation</p>	<p><b>060201-03</b>          Das Skalieren und Beschneiden wird visuell dargestellt.</p>

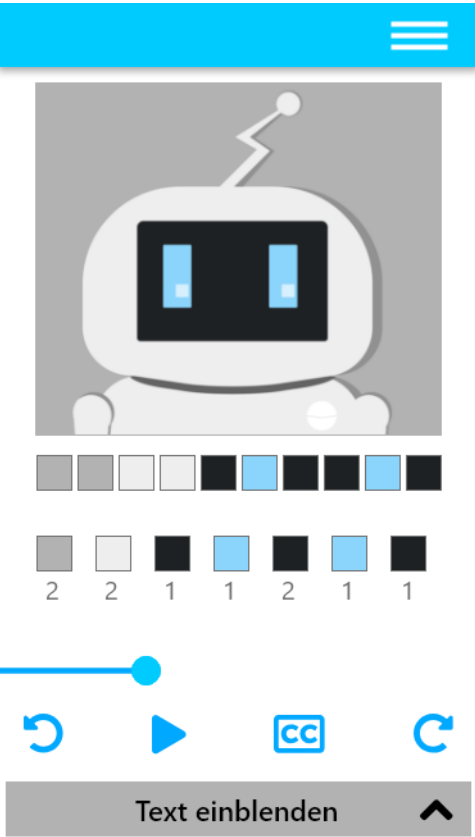
## 6.3 (A) Farbtiefenreduktion

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>060301</b> Die Farbtiefe ist die Anzahl der Bits die pro Pixel zur Speicherung der Farbinformation zu Verfügung stehen. Üblich sind Farbtiefen wie 8,16, 24 oder 32 Bit.</p> <p><b>060302</b> Manchmal überschreitet die gewählte Farbtiefe allerdings die vom Bild benötigten Farben. Bei 32 Bit Farbtiefe ist das vierte Byte in der Regel für einen unsichtbaren Transparenzkanal, denn sogenannten Alpha Kanal, vorgesehen. Diese Farbtiefe ist nur für bestimmte Aufgaben notwendig. Eine Reduktion auf 24 Bit spart in einem solchen Fall Speicherplatz.</p> <p><b>060303</b> Reduziert man die Farbtiefe weiter, so spart man noch mehr Speicherplatz. Es treten aber oft unschöne Farbverschiebungen auf.</p>	<p><b>060301</b> - Gibt Bits pro Pixel an</p> <p><b>060302</b> - Alpha Kanal für Transparenz möglich</p> <p><b>060303</b> - Zu geringe Farbtiefe -&gt; Farbverschiebungen</p>	<p><b>060301-03</b> Das Beispielbild wird in verschiedenen Farbtiefen angezeigt.</p>

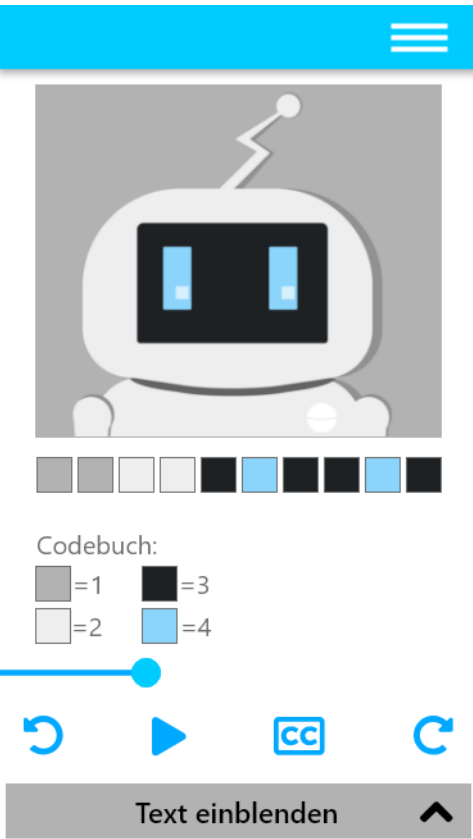
## 6.4 (I) Farbtiefenreduktion

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
 <p data-bbox="170 935 412 959">Datenmenge: 2 kByte</p> <div data-bbox="170 1007 517 1158"> <div>2 Bit <input type="radio"/></div> <div>4 Bit <input checked="" type="radio"/></div> <div>8 Bit <input type="radio"/></div> <div>16 Bit <input type="radio"/></div> <div>32 Bit <input type="radio"/></div> <div>64 Bit <input type="radio"/></div> </div>	<p><b>060401</b>          Probiere nun selbst die Farbtiefe einzustellen und beobachte dabei die jeweilige Datenmenge.</p>		<p><b>060401</b>          Man kann die verschiedenen Farbtiefen einstellen und somit sehen, welchen Einfluss die Bit-Angabe auf das Endergebnis hat.          Außerdem wird auch der verbrauchte Speicherplatz angezeigt.</p>

## 6.5 (A) RLE

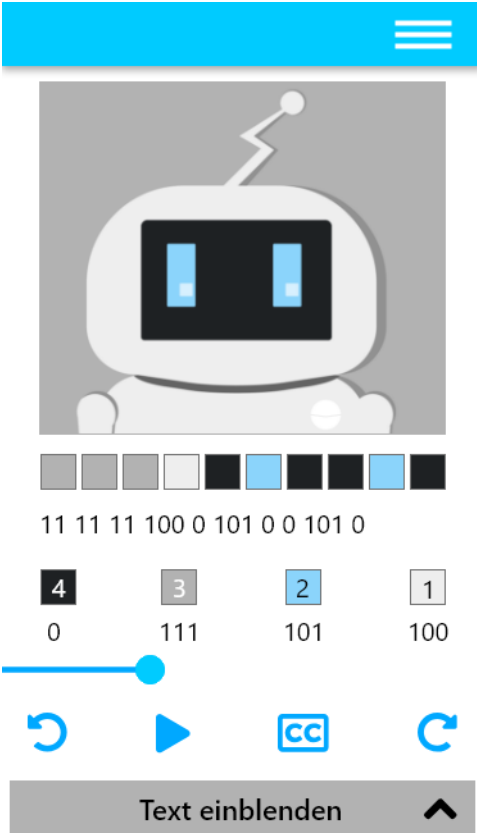
Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
 <p>The screenshot shows a video player interface. At the top is a blue header with a hamburger menu icon. Below it is a video frame showing a white robot with blue eyes. Under the video is a color palette with two rows of colored squares. The first row has 8 squares: grey, grey, light grey, light grey, black, blue, black, black. The second row has 7 squares: grey, light grey, black, blue, black, blue, black. Below the palette are two rows of numbers: the first row has '2 2 1 1 2 1 1' and the second row has '2 2 1 1 2 1 1'. At the bottom of the player are playback controls: a progress bar, a play button, a CC icon, and a refresh icon. A grey button labeled 'Text einblenden' with an upward arrow is at the very bottom.</p>	<p><b>060501</b>  RLE steht für run-length encoding, also Lauflängenkodierung und ist der einfachste der verlustfreien Kompressionsalgorithmen. Er untersucht die Daten des Bildes und fasst aufeinander folgende gleichfarbige Pixel zusammen. Dabei wird die Farbe einmal abgespeichert sowie ein Zahlenwert angegeben, der beschreibt in wie vielen Pixeln die Farbe hintereinander auftaucht. RLE ist eine sehr einfache Form der Kompression, kann aber bei Bildern im Bitmap-Format äußerst effektiv sein.</p>	<p><b>060501</b>  - run-length encoding  - Fasst Pixel zusammen</p>	<p><b>060501</b>  Das Zählen der Pixel wird visuell dargestellt.</p>

## 6.6 (A) LZW

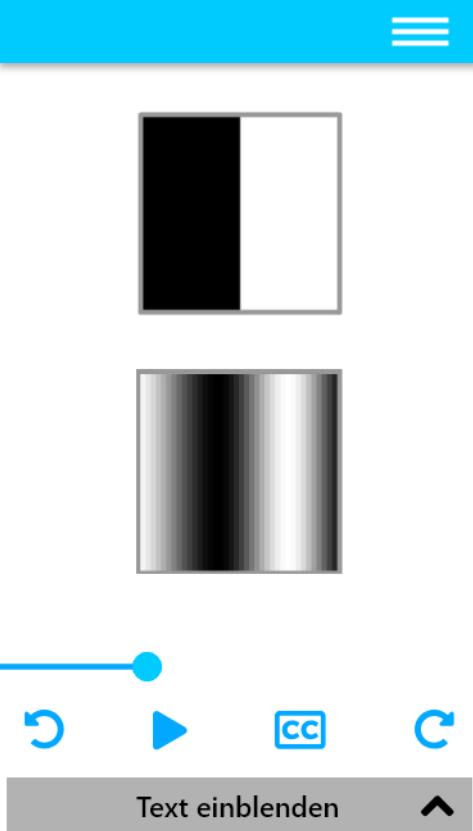
Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>060601</b> LZW ist ein weiterer verlustfreier Kompressionsalgorithmus, der nach seinen Erfindern Lempel, Ziv und Welch benannt ist. Dieser Algorithmus untersucht die Pixel eines Bildes auf sich wiederholende Muster. Ausgehend von den im Bild vorhandenen Farben, wird ein sogenanntes Codebuch angelegt. Dieses wird erweitert, wenn der Algorithmus das Bild auf Pixelkombinationen untersucht. Findet er beim Codieren eine unbekannte Pixelkette, so speichert er den zuletzt gefunden bekannten Wert. Die unbekannte Pixelkombination wird anschließend im Codebuch abgelegt.</p> <p><b>060602</b> Anschließend, wenn die gleiche Pixelkette noch einmal gefunden wird, verweist LZW nur noch auf den Eintrag im Codebuch, was eine Einsparung an Speicherplatz bedeutet.</p> <p><b>060603</b> Diese Form der Kompression gibt nur bei bestimmten Bildinhalten ein optimales Ergebnis, schafft aber in der Regel eine bessere Datenreduktion als RLE.</p>	<p><b>060601</b> - Verlustfrei - Pixelkombinationen werden im Codebuch vermerkt</p> <p><b>060603</b> - Besser als RLE</p>	<p><b>060601-03</b> Die Einträge ins Codebuch werden visuell dargestellt.</p>



## 6.7 (A) Huffman-Kodierung

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>060701</b> Die Huffman-Kodierung ist ebenfalls ein verlustfreier Kompressionsalgorithmus. Bei dieser Form der Kodierung nutzt man die Häufigkeit einzelner im Bild auftretender Farben aus.</p> <p><b>060702</b> Der Algorithmus untersucht zuerst das ganze Bild und ermittelt die Häufigkeit jeder einzelnen Farbe. Dann wird jeder Farbe ein Bit-Wert zugewiesen. Häufig auftretende Farben werden mit möglichst wenigen Bits dargestellt, während seltenere Farben mit mehr Bits repräsentiert werden. Die häufigste Farbe kann somit nur mit einem einzigen Bit kodiert werden.</p> <p><b>060703</b> Die hier dargestellten farbigen Pixel erzeugen einen Code von nur 19 Bit Länge.</p>	<p><b>060701</b> - Verlustfrei</p> <p><b>060702</b> - Farben bekommen Bit-Wert - Häufigste Farbe hat 1 Bit</p>	<p><b>060701</b> Das Zählen der Farben wird visuell dargestellt. Zum Schluss werden die Ergebnisse in Bits eingeblendet.</p>

## 6.8 (A) JPG

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>060801</b> JPG ist ein verlustbehaftetes Grafikformat, bei dem mehrere Kompressionsvorgänge durchgeführt werden. Verlustbehaftete Kompression bedeutet, dass bei der Kompression Bildteile zusammengefasst oder weggelassen werden. Dies geschieht hinsichtlich der optischen Ansprüche eines Menschen. Dadurch kann eine sehr kleine Datenmenge entstehen, die nach der Dekompression dem Original nur noch ähnlich ist.</p> <p><b>060802</b> Bei JPG im Speziellen läuft es folgendermaßen ab. Zuerst wird das Bild vom RGB in den YUV Farbraum umgerechnet, hierbei findet die erste Reduktion statt, da die Farbinformation U und V mit geringer Auflösung gespeichert werden.</p> <p><b>060803</b> Als Nächstes kommt es zur diskreten Kosinustransformation, oder kurz DCT genannt, die anhand des Y-Kanals beispielhaft gezeigt wird. Zunächst wird das Bild in 8x8 Pixel große Blöcke eingeteilt. Nun wird versucht, die Helligkeits- und Farbstrukturen in jedem Block mit Hilfe von Verläufen mathematisch anzunähern, diese</p>	<p><b>060801</b> - Verlustbehaftet</p> <p><b>060802</b> - Transformation von RGB zu YUV</p> <p><b>060803</b> - Diskrete Kosinustransformation  - Annäherung durch mathematische Verläufe</p>	<p><b>060801</b> Das Beispielfoto wird eingblendet.</p> <p><b>060802</b> Die Konversion eines Beispielfotos von RGB in den YUV Farbraum wird gezeigt.</p> <p><b>060803</b> Dann wird gezeigt wie das Bild in 8x8 Pixel große Blöcke eingeteilt wird. Danach wird oben der originale Block gezeigt und unten ein Block, der sich nach und dem Original annähert, indem mehrere</p>


	<p>Verläufe basieren auf einfachen vordefinierten Kosinuskurven. Verschiedene solcher Verläufe werden in unterschiedlicher Gewichtung überlagert. Dabei kann die Annäherung an das Originalbild durch die Anzahl der Überlagerungen beeinflusst werden. Zuletzt werden diese nun Zickzack ausgelesen und mit einer Huffman-Kodierung nochmals verlustfrei nachkomprimiert. Als Endergebnis erhält man eine um zehn bis hundertfach verkleinerte Datenmenge, aus der das ursprüngliche Bild wieder betrachtungsfähig konstruiert werden kann.</p>		<p>Blöcke übereinander gelegt werden.</p>
--	--	--	---

The screenshot shows a 3D software interface. At the top, there is a blue header bar with a white hamburger menu icon on the right. Below the header, the text 'Y-Kanal' is displayed. The main area is a large 3D viewport showing a wireframe model of a dog's head. Below this, there are two smaller viewports: 'U-Kanal' on the left and 'V-Kanal' on the right. The 'U-Kanal' viewport shows a yellow and purple wireframe projection, while the 'V-Kanal' viewport shows a red and green wireframe projection. At the bottom of the interface, there is a blue slider control and four navigation icons: a circular arrow, a play button, a Creative Commons license icon, and a refresh icon. The bottom of the screen features a dark gray bar with the text 'Text einblenden' and a white upward-pointing arrow icon.

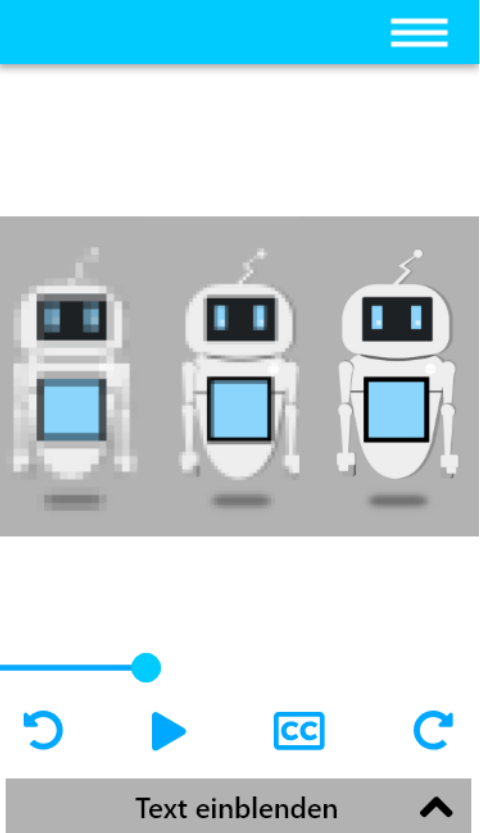
A screenshot of a video player interface. At the top, there is a blue header bar with a white hamburger menu icon on the right. Below the header, the video frame displays a horizontal gradient bar transitioning from black on the left to white on the right. Underneath the video frame is a blue progress bar with a white circular playhead marker positioned at approximately one-third of the way across. At the bottom, there is a dark grey control bar containing four white icons from left to right: a circular arrow for rewind, a right-pointing triangle for play/pause, a 'CC' icon for closed captions, and a circular arrow for full screen. Below the control bar, a white text label 'Text einblenden' (Show text) is visible, along with a white upward-pointing chevron icon on the far right.

A screenshot of a video player interface. At the top is a blue header bar with a white hamburger menu icon on the right. Below the header is a video player area. The video content is a square with a horizontal gradient from black on the left to white on the right. Below the video is a blue progress bar with a white playhead marker. At the bottom is a grey control bar containing four icons from left to right: a blue circular refresh icon, a blue triangular play icon, a blue square icon with 'CC' (Creative Commons), and a blue circular refresh icon. Below the control bar is a grey bar with the text 'Text einblenden' and a black upward-pointing chevron icon.

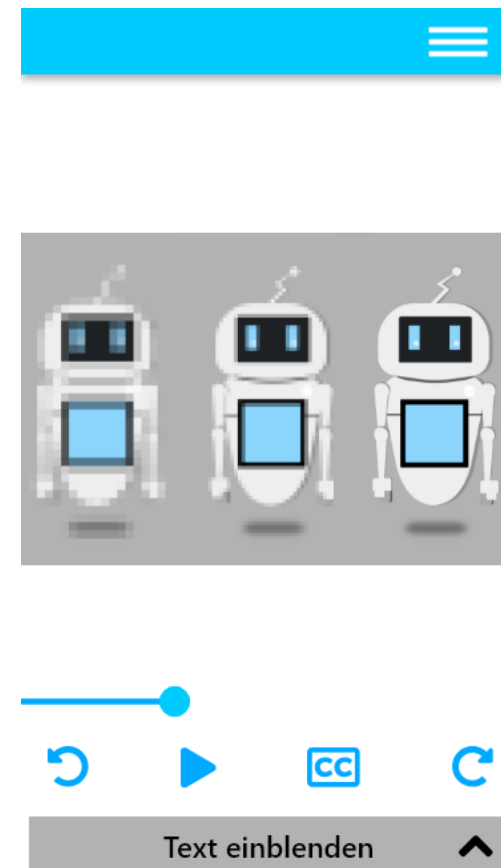
## 6.9 (I) JPG

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
 <p>Dateigröße: 80,4 KB</p> <p>Stärke der Kompression:</p> <p>gering stark</p>	<p><b>060901</b></p> <p>Probiere nun die JPG Kompression mithilfe des Reglers durchzuführen, beobachte dabei die Bildung von Blockartefakten im Bild.</p>		<p><b>060901</b></p> <p>Durch das Verschieben des Reglers wird die Stärke der Kompression erhöht und man kann die Bildung von Blockartefakten beobachten. Außerdem wird die aktuelle Dateigröße angezeigt.</p>

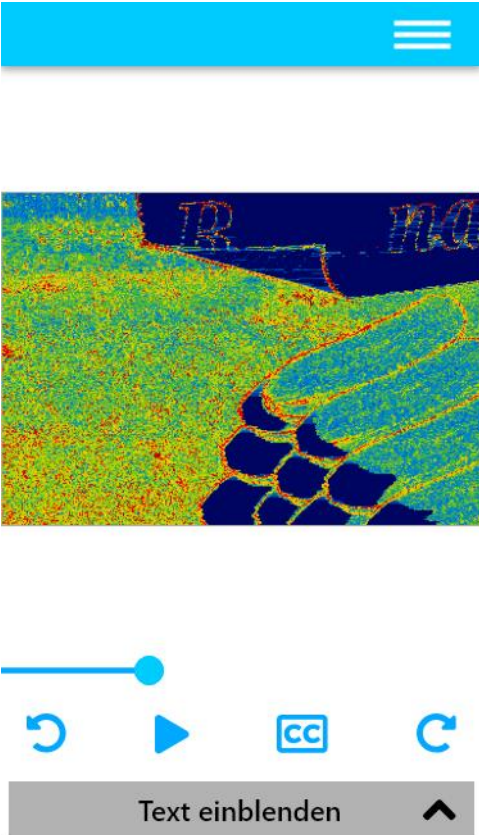
## 6.10 (A) GIF

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>061001</b> Das Graphics Interchange Format, kurz GIF, ist ein Grafikformat mit einer verlustfreien Kompression. Das besondere an GIF ist die Möglichkeit der Speicherung von mehreren, auch übereinanderliegenden, Einzelbildern. Dadurch wird die Darstellung als Animation ermöglicht, was auch der Grund ist, warum GIF eine hohe Popularität besitzt.</p> <p><b>061002</b> GIF unterstützt, inklusive Transparenz, nur 256 indizierte Farben oder Graustufen, weshalb komplexe Bilder in ihrer Farbdarstellung reduziert werden müssen.</p> <p><b>061003</b> Außerdem unterstützt das Format auch so genanntes Interlacing, wodurch beim Laden eines GIFs die Auflösung Schritt für Schritt erhöht werden kann. Das war vor allem früher von Vorteil, da trotz langsamer Internetverbindung schon etwas grob angezeigt wurde.</p>	<p><b>061001</b> - Verlustfrei - Animationen möglich</p> <p><b>061002</b> - Unterstützt nur 256 Farben inklusive Transparenz</p> <p><b>061003</b> - Interlacing</p>	<p><b>061001</b> Als erstes wird eine GIF-Animation gezeigt.</p> <p><b>061002</b> Dann wird gezeigt, wie die Farben eines Beispielbildes auf 256 reduziert werden.</p> <p><b>061003</b> Zum Schluss sieht man wie Interlacing aussieht.</p>

	Zur Kompression wird LZW verwendet, welchen wir schon in einem anderen Kapitel kennengelernt haben.		
--	---	--	--

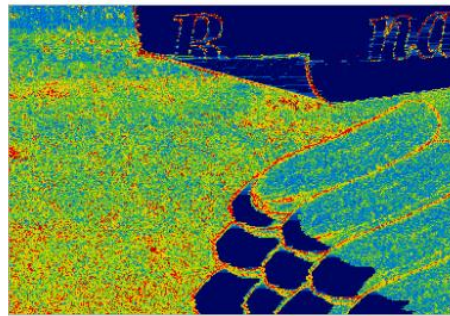
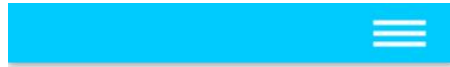
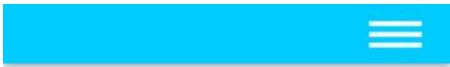


## 6.11 (A) PNG

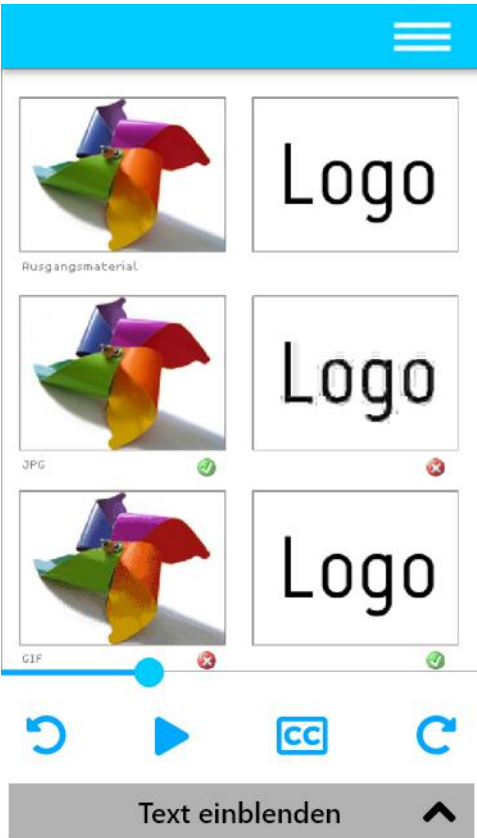
Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>061101</b> PNG steht für Portable Network Graphics und ist heute das meistverwendete verlustfreie Grafikformat im Internet. Die Entwicklung dieses Grafikformates begann Ende 1994 mit dem Ziel das Grafikformat GIF zu ersetzen.</p> <p><b>061102</b> Das PNG Format unterstützt verschiedene Farbtiefen, üblicherweise 8, 24 und 32 Bit, wobei die 32Bit Variante einen zusätzlichen 8-Bit-Kanal für Transparenz-Informationen enthält.</p> <p><b>061103</b> Die Kompression eines PNG verläuft in drei Schritten. Zuerst kommt es zum Vorfiltern, wo sehr ähnliche Farben auf einen Farbwert gesetzt werden.</p> <p><b>061104</b> Danach kommt es zur Wörterbuch-basierten Kodierung per LZ77 Algorithmus, welcher ein Vorgänger des im LZW verwendeten LZ78 ist.</p> <p>Das Verlustfreie LZ77 sucht zu Beginn nach sich wiederholende Sequenzen von Daten. Wenn der</p>	<p><b>061102</b> - Unterstützt 8, 24 und 32 Bit - Transparenz möglich</p> <p><b>061103</b> - Ähnlich Farbwerte werden vereinheitlicht</p> <p><b>061104</b> - Wiederholende Sequenzen erzeugen nur Verweis</p>	<p><b>061101-03</b> Zu Beginn wird ein Beispielbild eingeblendet.</p> <p><b>061104</b> Dann wird gezeigt wie PNG sich wiederholende Sequenzen auslöst. Zur Veranschaulichung wird gezeigt welche Sequenzen im Beispielbild sich wiederholen.</p>



	<p>Algorithmus auf eine Sequenz trifft, welche es schon einmal gab, gibt es nur einen Verweis auf die entsprechende Sequenz, was bei manchen Bildern viel Speicherplatz spart. Enthält ein Bild zum Beispiel zwei identische schwarze Kreise, so verbraucht nur der erste Kreis Speicherplatz.</p> <p><b>061105</b></p> <p>Im Letzten Schritt werden die bis dahin erzeugten Daten noch mithilfe der Huffman-Kodierung komprimiert.</p>		
--	---	--	--



## 6.12 (A) Tipps & Tricks

Screen	Sprechertexte	Stichwörter / Notizen	Regieanweisungen
	<p><b>061201</b> Um richtig zu komprimieren, sollten im Grafikbereich ein paar Tipps und Tricks beachtet werden. Nicht alle Kompressionsverfahren lassen sich gut auf jede Art von Bildern anwenden. Kompression wird häufig bei Bildern verwendet um diese im Internet zu publizieren, per Mail zu verschicken oder zum Download anzubieten.</p> <p><b>061202</b> Ein Bild, das viele Farbverläufe besitzt, also z.B. ein klassisches Foto, sollte mit JPG komprimiert werden. Dieses Verfahren wurde entwickelt, um Bilder mit Farbverläufen besonders gut zu komprimieren, da diese optisch leicht mit mathematischen Verläufen angenähert werden können.</p> <p><b>061203</b> Ein Bild das harte Farbkanten besitzt, wie zum Beispiel Logos oder Schriftzüge auf einfarbigem Grund, sollte mit GIF oder PNG komprimiert werden, da bei diesen Verfahren die Reduktion nicht durch Zusammenfassung von Bildinhalten erfolgt. Unscharfe Kanten, wie bei JPG üblich, gibt es bei PNG und GIF nicht. Dafür ist speziell GIF auf</p>	<p><b>061202</b> - JPG für Bilder mit Farbverläufen</p> <p><b>061203</b> - PNG &amp; GIF für harte Farbkanten</p>	<p><b>061201-03</b> Die Tipps werden nach und nach anhand eines Beispielfotos visuell dargestellt.</p>

	256 Farben beschränkt, weshalb man heute in der Regel PNG verwendet.		
--	--	--	--