FRAGE 2

Motion Path

Frage 2)
Ein Motion Path soll im UHD Bildschirm 3840x2160 Pixel @ 16:9 ein Stern Polygon mit den Punkten `P1-4-7-2-5-8-3-6-1` periodisch durchlaufen. Der UHD Nullpunkt (`"canvas"`) ist links oben, die x Achse nach rechts und die y Achsen nach unten. Auf jeder UHD Seite wird der Bereich von 1/3 bis 2/3 rot markiert. Mit 4 weiteren Schrägen werden diese 4 Abschnitte zu einem `"8-Eck_rot"` verbunden und begrenzen den Bereich der Animation. Die Mittelpunte der 8 Polygon Seiten bilden dann einen 8-Eck Stern beginnend bei P1(1920, 0) im Uhrzeigensinn. P3 ist in der Mitte rechts, P5 in der Mitte unten, und P7 in der Mitte links. Tragen Sie im Polygon `"8-Stern_gruen"` die 8 `points x,y` in der Reihenfolge der Animation `P1-4-7-2-5-8-3-6-1` ein. Der Kreis `"Flying_Spot"` erfolgt mit `animateTransform` als relative Translation beginnend bei P1(1920, 0) mit den `values 0,0`. Die Animation endet ebenso mit `values 0,0` nach einer vollen Periode. Tragen Sie die anderen 7 aufeinander erfolgenden Verschiebungen x,y dazwischen ein. (12 Punkte)

Teilaufgaben:

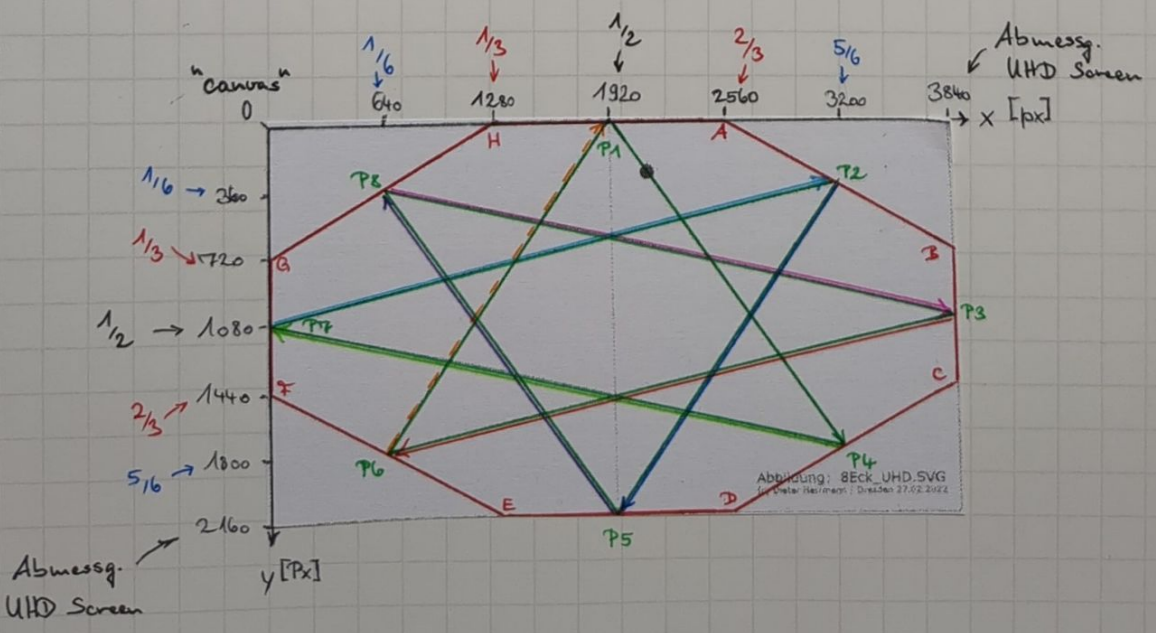
a) Koordinatensystem einzeichnen

b) optional: "8-Eck_rot" nachzeichnen, Eckpunktkoordinaten im Code

c) optional: "8-Stern-gruen" nachzeichnen
P1 bis P8 einzeichnen: P1 oben mittig, dann im Uhrzeigersinn

d) Statische Koordinaten der Punkte ablesen bzw. berechnen und in den entsprechenden Lücken im Codeblock eintragen. Reihenfolge beachten!

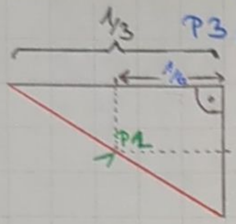
e) Verschiebungsvektoren von Punkt zu Punkt in die unteren Lücken im Codeblock eintragen.



d) – hab mir erstmal die Reihenfolge der Punkte eingetragen

– P1 ist gegeben

– P3, P5 und P7 können abgelesen werden



– P2, P4, P6 und P8 liegen mittig auf den Schrägen des 8-Ecks. Diese Schrägen bilden wiederum mit den Bildschirmecken jeweils ein rechtwinkliges Dreieck

– da die Punkte mittig auf der Hypothenuse sitzen, haben sie auch genau die halbe Abmessung der Katheten

– die Katheten haben jeweils 1/3 der Gesamtabmessung des Bildschirms, also sind die Punkte jeweils auf 1/6 bzw. 5/6 der Achsen.

```
<svg id="8-Eck_UHD"  width="100%" height="100%"
                     viewBox="-20 -20  3880 2200"       . . .    >
<title> Motion Path 8-Eck Stern in UHD 3840x2160px </title>
<rect id="canvas"  x="0" y="0" width="3840" height="2160" ... />
<line id="horiz_x" x1="840" y1="1080" x2="3000" y2="1080" ... />
<line id="verti_y"  x1="1920"  y1="0" x2="1920" y2="2160" ... />

<polygon id="8-Eck_rot"
   points=" A 2560,0    B 3840,720  C 3840,1440 D 2560,2160
            E 1280,2160    F 0,1440    G 0,720   H 1280,0   "   ... />
<polygon id="8-Stern_gruen"   style="stroke:rgb(0,200,0); "

   points=" P1 1920,0   P4 3200,1800  P7 0,1080  P2 3200,360

            P5 1920,2160  P8 640,360  P3 3840,1080  P6 640,1800 "  ... />

<circle id="Flying_Spot"  cx="1920" cy="0"  r="40"         ... /> (durchgestrichen)
<animateTransform  id="Clockwise_P1-4-7-2-5-8-3-6-1"
type="translate"  begin="1s" dur="9s"  repeatCount="indefinite"
                             P1→P4           P4→P7          P7→P2
values="    0,0       ;    1280,1800 ;    -3200,-720 ;    3200,-720   ;
         P2→P5            P5→P8          P8→P3          P3→P6
         -1280,1800 ;     -1280,-1800 ;   3200,720  ;    -3200,720   ;
*P6→P1
1280,-1800;   0,0   " />  </circle>                               </svg>
```

Bsp.:

P2: $x = 5/6 * 3840$
$= \underline{3200}$

$y = 1/6 * 2160$
$= \underline{360}$

e) – der gegebene erste Vektor beträgt 0,0 da im Startpunkt P1 keine Bewegung stattfindet

– dann bewegt sich der "Flying-Spot" nach P4, also nach rechts unten. Die entsprechenden Verschiebungswerte sind unser x und y

Bsp.: P1 → P4: $x = 3200 - 1920 = \underline{1280}$
$y = 1800 - 0 = \underline{1800}$

– entsprechend mit den restlichen Verschiebungen verfahren, geht die Richtung entlang der Achse nach 0 (oben bzw. links), dann ist der entsprechende Wert negativ

(von Herrmann) * FEHLER: • der <circle>-Tag wird erst nach <animat./> geschlossen
• die Verschiebung zurück nach P1 fehlt.

Frage 3

Pattern Matching

Frage 3)

Allg. Bei Bildstabilisatoren, Objekt-Suche und Verfolgung, Gesichts-Erkennung oder Personen-Identifikation, in VR/AR bis zum Predictive Modeling wird Pattern-Matching (Block-Matching) eingesetzt. In der MPEG Video-Kompression mit Motion-Compensation MC oder MPEG-7 zur visuellen Beschreibung (Visual Description) von Merkmalen wie Farben, Texturen, Kanten, Formen, Bewegungen und dergleichen wird ebenso Block-Matching genutzt.

Ein Pixel-Raster aus 4 Bildpunkten wie ein Symbol Y, Linker und Rechter Arm oben, Sternpunkt mittig und darunter Mittelfuß, mit Objekt [1 1 1 1] bewegt sich relativ vom I-Frame 1 zu P-Frame 2. Im Block-Matching Y im Bereich [3*3] Pixel sind jeweils die 4 Einzel-Differenzen di=f(Frame 2 - MC Frame 1) in allen 9 Richtungen um die 9 Fadenkreuze einzutragen. Aus den jeweils 4 Einzel-Differenzen ist unten die Summe der Differenz-Quadrate zu bilden. Abschließend sind die Abweichungen in den 9 Richtungen nach einem Ranking zu sortieren von 1 für Minimal bis 9 für Maximal. Eventuelle Gleichrangigkeit wird entsprechend Abarbeitung ausgeschlossen. Die minimale Abweichung 1 entspricht damit dem gefundenen und wahrscheinlichsten Bewegungs-Vektor. (12 Punkte)

Erster Absatz: Allgemeine Begriffserklärung

Teilaufgaben:

a) Beschreibung des Objekts in Form eines Y-Symbols

b) je 4 Einzel-Differenzen in 9 Fadenkreuze eintragen

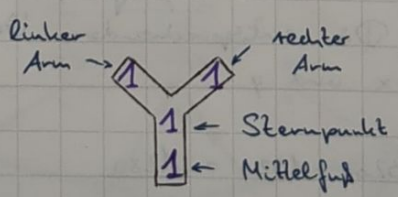
c) Quadratische Mittelwertbildung

d) Ergebnisse aus c der Größe nach aufsteigend nach 1-9 sortieren. Rang 1 entspricht dem wahrscheinlichsten Bewegungsfaktor

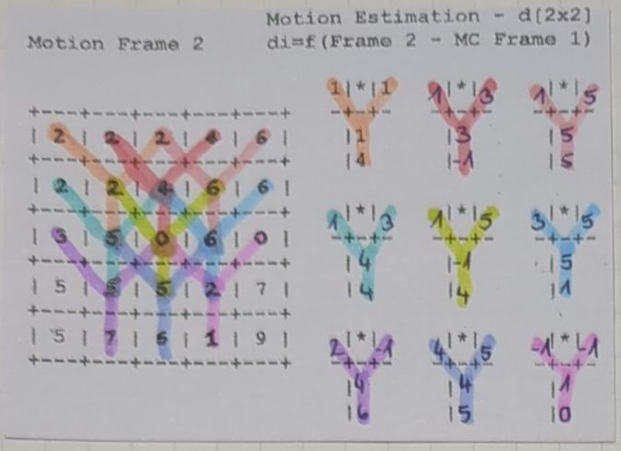
I-Frame 1

1	2	3	4	5
2	**1**	4	**1**	6
3	4	**1**	6	7
4	5	**1**	7	8
5	6	7	8	9

a) Die unterstrichenen Ziffern bilden gemeinsam das Symbol Y



Dieses Objekt bewegt sich nun vom I-Frame 1 in den P-Frame 2 (Motion Frame 2) mit 5*5 Pixel. Der Algorithmus sucht daraufhin das 3*3 Objekt in dem verrauschten Frame an allen 9 möglichen Stellen.



b) - mir hat es geholfen, die möglichen Stellen des Y-Objekts farblich im Raster zu markieren und die entsprechenden Fadenkreuze in derselben Farbe.

- anhand des vorgegebenen Fadenkreuzes erkennt man direkt die simple Vorgehensweise: von den markierten Bildpunkten einfach die 1 des Y abziehen

c) - aus den Ziffern in den 9 Fadenkreuzen bildet man nun jeweils Quadratsummen

Bsp.: $1^2 + 3^2 + 4^2 + 4^2 = \underline{42}$

- durch das Quadrieren verschwinden negative Werte

- außerdem werden Abweichungen potenziert, denn wir wollen die kleinste Abweichung finden

Block-Summe (Diff)²

19	20	76
42	43	60
57	82	3

Ranking Min-Max 1...9

2	3	8
4	5	7
6	9	1

- haben mehrere Summen den gleichen Wert, erfolgt die Reihenfolge anhand der Leserichtung ⟹

d) Die Quadratsummen werden nach ihrer Größe in ein Ranking gebracht.
Rang 1 ist dann unser gesuchter Bewegungsvektor, also die Stelle an die sich das Objekt bewegt hat.

„Der Algorithmus findet das Objekt an der Stelle, wo es am wenigsten verrauscht ist!"

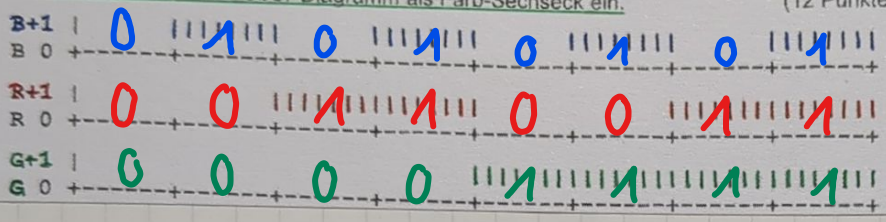
FRAGE 4

Farb-Differenzen

Frage 4)

Allg.

In visuellen Systemen (Grafik, Foto, Video, HDTV, UHD) werden Testbilder, Farb-Differenzen YCbCr(RGB) oder Farb-Grauwert Transformationen Y(RGB) eingesetzt. Gegeben sind die 8 Farben additiv RGBW und subtraktiv CMYK in einer tri-chromatischen Test-Impulsfolge mit 3 bit (blue, red, green) zur additiven Erzeugung von 8 Farb-Balken (RGB) und 8 Luminanz-Stufen Y. Berechnen Sie die positiven wie negativen Farb-Differenzen B-Y und R-Y für UHD TV (Fast und vereinfacht). Skalieren Sie die Werte wie in der HD Fernsehtechnik auf Cb und Cr jeweils im Bereich von -0,5 bis +0,5. Tragen Sie die 6 Farb-Differenz-Koordinaten Cb und Cr von (B R M G C Y) in das CbCr-Diagramm als Farb-Sechseck ein. (12 Punkte)



* das kann man hier ablesen

||||||||| ⇒ 1
⇒ 0

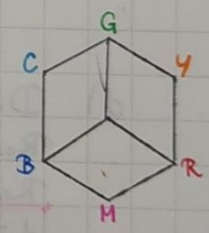
Erster Absatz: Allgemeine Beschreibung

Teilaufgaben:

a) Luminanz-Stufen Y und Farb-Differenzen B-Y und R-Y berechnen

b) Skalieren der Werte auf Cb und Cr im Bereich -0,5 bis 0,5

c) Eintragen der 6 Farb-Differenz-Koordinaten in das CbCr-Diagramm → Ergebnis: Farb-Sechseck



---

a) Berechnung der Luminanz Y:
- für RGB einfach die Faktoren aus der Formel eintragen
- die anderen Farben entsprechend aus RGB addieren

Berechnung der Farbdifferenzen:
- für B bzw. R eine 1 oder 0 einsetzen, je nachdem ob sie in der jeweiligen Farbe enthalten ist. *

Luminanz Y(Fast) = f(R,G,B) = 0,25 * R + 0,50 * G + 0,25 * B

	K	B	R	M	G	C	Y	W
Y	0.0000	0.25	0.25	0.50	0.50	0.75	0.75	1.0000
			R+B		B+G	G+R		
B-Y	0.0000	0.75	-0.25	0.50	-0.50	0.25	-0.75	0.0000
R-Y	0.0000	-0.25	0.75	0.50	-0.50	-0.75	0.25	0.0000

Bsp.:

$B\text{-}Y_C = 1 - 0{,}75 = 0{,}25$

$R\text{-}Y_C \quad 0 - 0{,}75 = -0{,}75$

↘ Cyan = Blue + Green
↳ enthält also B ↘ 1
↳ enthält nicht R ↘ 0

Die Maximal- bzw. Minimalwerte liegen bei 0,75 und -0,75

Normieren auf -0,5 bis 0,5: Cb=0,66667 (B-Y); Cr=0,66667 (R-Y)

Cb	0.0000	0.500	-0.167	0.333	-0.333	0.167	-0.500	0.0000
	K	B	R	M	G	C	Y	W
Cr	0.0000	-0.167	0.500	0.333	-0.333	-0.500	0.167	0.0000

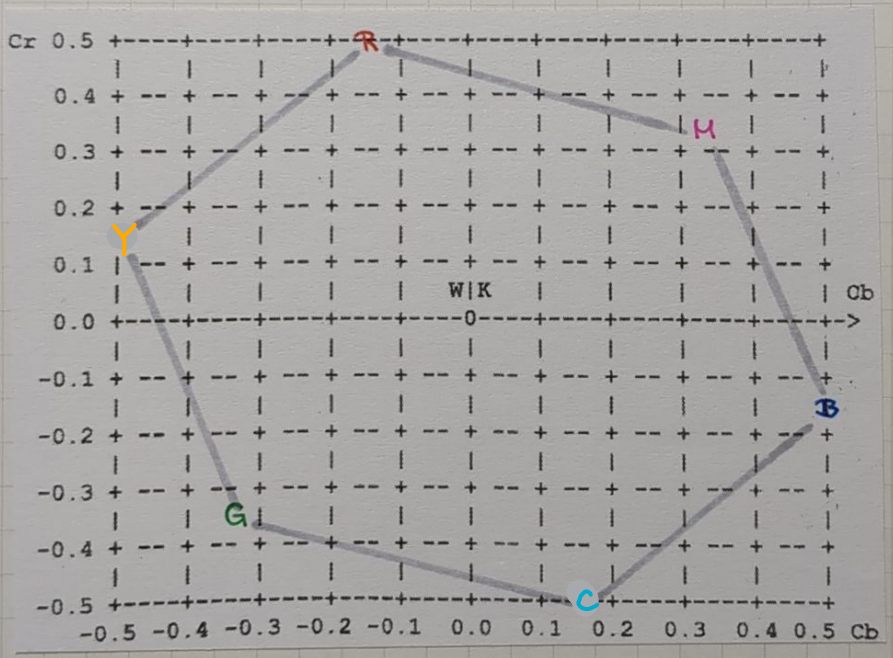
b) Die berechneten Werte müssen nun in den Bereich -0,5 bis 0,5 skaliert werden.

Umrechnungsfaktor: $\frac{0,5 - (-0,5)}{0,75 - (-0,75)} = \frac{1}{1,5} = 0,\overline{6}$

↑ Werte entsprechend multiplizieren

Bsp.: $Cb_M = 0,66667\,(B - Y_M)$
$= 0,66667 \cdot 0,5$
$= 0,333$

$Cr_Y = 0,66667\,(R - Y_Y)$
$= 0,66667 \cdot 0,25$
$= 0,167$



Red
Green
Blue
White
} RGBW additives Farbsystem

Cyan
Magenta
Yellow
Karbon ↳ Key
} CMYK subtraktives Farbmodell

c) Anhand der berechneten Werte bzw. Farb-Differenz-Koordinaten die Farben in das CbCr-Diagramm eintragen
→ Farb-Sechseck

Cb .. Chrominanz bzw. Farbigkeit zu Blau
Cr .. Chrominanz bzw. Farbigkeit zu Rot

FRAGE 6

Transfer Funktion

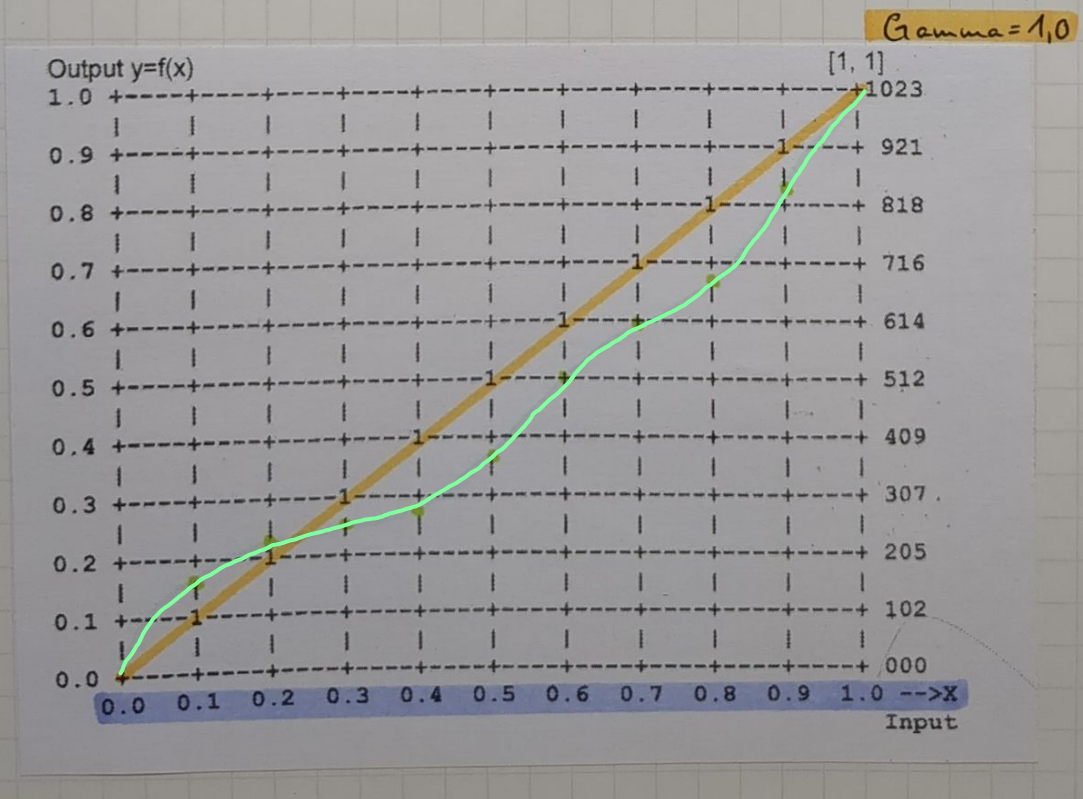
Frage 6)
Eine alltägliche Anwendung in allen visuellen Systemen (CRT, TFT, DSC, DVC, TV, Video, Disc, Print, UHD, HDR-TV, VR, AR) ist die Anpassung der Signal-Übertragungskennlinie durch Gamma-Korrektur oder eine adaptive Transfer-Funktion.
(Applying gamma correction means that each of the three R, G and B must be converted to $R'=R^{gamma}$, $G'=G^{gamma}$, $B'=B^{gamma}$, before handing to the Operating System. This may rapidly be done by building a 10 bit HDR 1024-element Look-Up Table - LUT.)
Gamma=1,0 entspricht einer linearen Übertragung ohne Korrektur, Signal-Ausgang gleich Signal-Eingang. Die Look-Up Table LUT in 10 bit HDR mit 1024 Elementen soll adaptiv vom kumulativen Histogramm abgeleitet werden. Die diskrete Histogramm-Funktion ist mit 11 absoluten Häufigkeiten von 0 bis 10 mit Hi gegeben. Berechnen Sie das kumulative Histogramm cH durch fortlaufendes Aufsummieren von Hi. Mit dem Maximal-Wert von 100 % werden alle anderen kumulierten Summen normiert zu relativen Histogramm-Werten rH zwischen 0.0 und 1.0. Schließlich erfolgt dann die Rücknormierung auf 10 bit Ausgabe mit Y2 von 0 bis 1023. Berechnen Sie die absoluten und relativen Koordinaten der Transfer-Curve und skizzieren Sie diese im Diagramm mit 10 bit Skala.
(12 Punkte)

a) cH durch Aufsummieren von Hi berechnen

b) anhand des Maximal-Werts an rH zwischen 0,0 und 1,0 skalieren

c) Rücknormierung auf 10 bit mit Y2 von 0 bis 1023

d) Skizzieren der Transfer-Kurve



a) - cH berechnen

0 + 80 = 80

80 + 30 = 110

110 + 10 = 120 usw.

b) - cH-Werte auf rH skalieren

- 500 entspricht 1,0

↑ kumulative Histogramme sind immer monoton wachsend

Bsp.:

$$rH(0.1) = \frac{80}{500} = \underline{0,16} \text{ bzw. } 16\%$$

Histogramm - Hi , kumuliertes Histogramm - cH :

Hi	0	80	30	10	20	50	60	50	40	70	90
cH	0	80	110	120	140	190	250	300	340	410	500
--	---	---	---	---	---	---	---	---	---	---	---

Gamma = 1.0 : [Y1 = f(X1), Kurve 1]

X1	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
G1	0.0	.10	.20	.30	.40	.50	.60	.70	.80	.90	1.0
Y1	000	102	205	307	409	512	614	716	818	921	1023
--	---	---	---	---	---	---	---	---	---	---	---

HDR adaptive Transfer-Curve, LUT - Look-Up Table [10 bit von dezimal 000 bis 1023]

X1	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
rH	0.0	.0,16	.0,22	.0,24	.0,28	.0,38	.0,5	.0,6	.0,68	.0,82	1.0
Y2	000	164	225	246	286	389	512	614	696	839	1023
--	---	---	---	---	---	---	---	---	---	---	---

c) - anhand von rH die Y2-Werte berechnen

Bsp.:

$$Y2 = rH * 1023$$

$$Y2(0.1) = 0,16 * 1023$$

$$= \underline{164}$$

d) - Transfer-Kurve einzeichnen