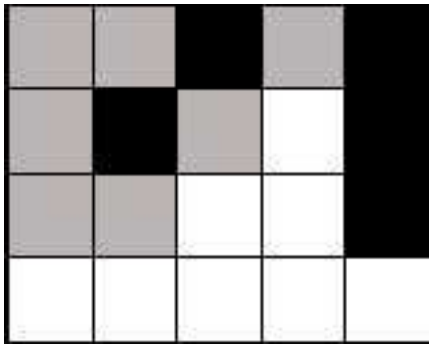
Multimedia-Technik: Übungsblatt 12

Aufgabe 1: Farbanalysen

Gegeben sei folgender vergrößerter Bildausschnitt (1 Kästchen = 1 Pixel).



(a) Geben Sie für ein beliebiges zweidimensionales, digitales Bild der Größe N×M mit 256 Graustufen einen Algorithmus zur Berechnung des zugehörigen Histogramms an.

```
funct calculateHistogram(I : Image) : array [0..255] of integer
    H : array [0..255] of integer
    for i = 0 to 256 − 1 do H[i] = 0 od
    for x = 0 to I.width-1 do
        for y = 0 to I.height-1 do
            H[I(x,y)] = H[I(x,y)] + 1
        od
    od
    return H
end
```

(b) Berechnen Sie das Grauwerthistogramm für oben gegebenen Bildausschnitt.

Gemäß obigen Algorithmus' ergeben sich an drei Positionen des Array Einträge: Position 0 (schwarz) mit 5 Pixeln, Position 128 (grau) mit 7 Pixeln und Position 255 (weiss) mit 8 Pixeln.

(*) Wie kann man ein Histogramm für Farbbilder (RGB) berechnen und effizient abspeichern?

Mit dem obengenannten Ansatz benötigt man für drei Kanäle (R,G,B) mit jeweils 8 bits $2^{24} = 16$ MB Einträge. Eine mögliche Lösung ist die Reduzierung der einzelnen Kanäle auf jeweils 3 bits, d.h. es werden nur die oberen 3 bits jedes Kanals für die Berechnung des Histogramms verwendet.

(c) Berechnen Sie den CCV auf oben gegebenem Bildausschnitt, wobei die 4er-Nachbarschaft zur Bildung von Regionen zugrundegelegt wird. Die 4er-Nachbarschaft bezieht sich auf die zu einem Pixel in gerade Linie benachbarten Pixel, im Gegensatz zur 8er-Nachbarschaft, bei der alle umliegenden Pixel als Nachbarn betrachtet werden.

Anzahl der Regionen: 7

Durchschnittliche Größe der Regionen: $20/7 = 2,86$

CCV: Grauwert 0 (schwarz)

- 1 Region mit 3 Pixeln; $3 > 2,86 \Rightarrow \alpha_0 = 3/20 = 0,15$
- 2 Regionen mit je 1 Pixel; $1 < 2,86 \Rightarrow \beta_0 = 2/20 = 0,1$

usw.

$\Rightarrow \text{CCV} = [(0,15;\ 0,1), (0,25;\ 0.1), (0,4;\ 0,0)]$

Aufgabe 2: Schnitterkennung

(a) Wie können zwei Histogramme H1 und H2 miteinander verglichen werden?

Betrachtet man die beiden Histogramme als Vektoren, d. h. $H_1 =< h_0, h_1, \ldots, h_n >$ und $H_2 =< k_0, k_1, \ldots, k_n >$, so kann man folgende Distanzmaße verwenden:

$$L_1 = \sum_{i=0}^{n} | h_i - k_i | \qquad\qquad L_2 = \sum_{i=0}^{n} (h_i - k_i)^2$$

(b) Entwerfen Sie einen Algorithmus, der auf Basis von Histogrammen Schnitte in digitalem Video automatisch erkennt. Die Bilder seien dabei als char image[height][width] vereinbart.

```
funct cutDetect(I1 : Image, I2 : Image, T : integer)
   H1 = calculateHistogram(I1)
   H2 = calculateHistogram(I2)
   difference = 0
   for i = 0 to numberOfEntries do
       difference += abs(H1[i] - H2[i])
   od
   return difference > t
end
```

(c) Beurteilen Sie die Aussagekraft von Histogrammen in Bezug auf die histogrammbasierte Schnitterkennung in Videosequenzen. Wie kann man die von Ihnen aufgezeigten Probleme umgehen? Welche Probleme entstehen dabei?

Die Verwendung von Histogrammen für die Erkennung von Schnitten in Videosequenzen ist problematisch, da grundverschiedene Bilder das gleiche Histogramm aufweisen können; man betrachte zum Beispiel zwei Bilder: Bild 1 zeigt eine Wiese mit Mohnblumen und Bild 2 stellt einen großen, roten Sonnenschirm dar. In beiden Bildern sind demnach viele rote Bildpunkte vorhanden, ihre Verteilung über die Bildfläche ist jedoch völlig unterschiedlich.

Man kann nun versuchen, Histogramme verschiedener Teilbilder zu erstellen und diese mit nachfolgenden Teilbildern zu vergleichen, um die Verteilung einzubeziehen. Damit verändern aber auch Objektbewegungen die Histogramme grundsätzlich und können zu einem falschen Ergebnis führen.

Aufgabe 3: Gradient

Der Betrag des Gradienten eines Bildes I(x,y) wird folgendermaßen bestimmt:

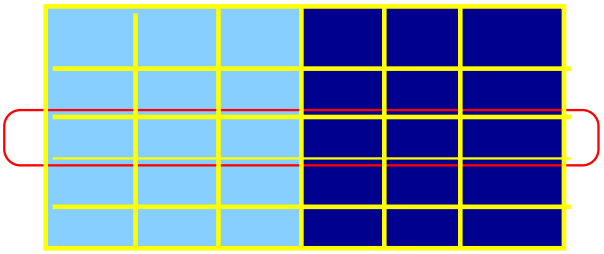
$$|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$$

Die partiellen Ableitungen können dabei folgendermaßen approximiert werden:

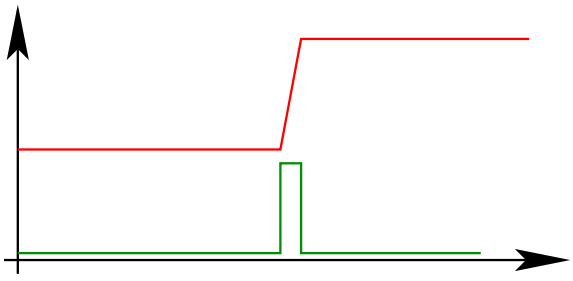
$$\frac{\partial I(x,y)}{\partial x} \approx \frac{I(x+1,y) - I(x-1,y)}{2}$$

$$\frac{\partial I(x,y)}{\partial y} \approx \frac{I(x,y+1) - I(x,y-1)}{2}$$

Im Folgenden ein Beispiel, dass den Zusammenhang zwischen Ableitungen und Bildstrukturen verdeutlichen soll:



100	100	100	200	200	200
100	100	100	200	200	200
100	100	100	200	200	200
100	100	100	200	200	200
100	100	100	200	200	200
100	100	100	200	200	200



Pixelwerte

Steigung

(a) Nachfolgend sind die Grauwerte eines Bildes der Größe 6×6 abgebildet.

100	100	200	100	200	100
100	**100**	200	100	200	100
100	100	200	200	200	100
100	100	200	200	200	100
100	100	200	100	200	100
100	100	200	100	200	100

Am Beispiel des fett markierten Pixels (Wert: 100) ergibt sich folgende Rechnung für den Gradienten:

Horizontale partielle Ableitung (X-Richtung):

$$(200-100)/2 = 50$$

Vertikale partielle Ableitung (Y-Richtung):

$$(100-100)/2 = 0$$

Gradient:

$$\sqrt{50^2+0^2} = 50$$

Für den gesamten Block ergibt sich das folgende Ergebnis.

70	70	100	50	100	111
50	50	0	50	0	100
50	50	50	50	50	100
50	50	50	50	50	100
50	50	0	50	0	100
70	70	100	50	100	111

b) Der Gradient kann auf folgende Weise implementiert werden:

```
funct gradient(int in_buf[][], int width, int height) : int[][]
  for x = 0 to width − 1 do
    for y = 0 to height − 1 do
      grad_x[y][x] = (in_buf[y][x+1] − in_buf[y][x−1])/2;
    od od
  for y = 0 to height − 1 do
    for x = 0 to width − 1 do
      grad_y[y][x] = (in_buf[y+1][x] − in_buf[y−1][x])/2;
    od od
  for y = 0 to height − 1 do
    for x = 0 to width − 1 do
      grad[y][x] = sqrt(grad_x[y][x] + grad_y[y][x])
    od od
  return grad;
```

Es ergibt sich folgendes Kantenbild:

c) Welche visuelle Bedeutung hat der Gradient (welchen visuellen Effekt erzielt man)? Können Sie begründen, warum der Gradient diesen visuellen Effekt hervorruft?

Der Gradient wirkt als Kantendetektor. Der Gradient berechnet die erste Ableitung der Grauwerte. Verändern sich die Grauwerte wenig, so ist die „Steigung“ gering und der Gradient an der entsprechenden Stelle klein. An einer Kante zeigen die Grauwerte große Unterschiede, dementsprechend ist der Gradient groß.

Aufgabe 4: Bildverarbeitung

Gegeben sei ein zweidimensionales Grauwertbild I. Welches Ergebnis liefert untenstehender Bildverarbeitungsoperator *Operator*? Begründen Sie Ihre Antwort mittels einfacher Beispiele.

proc *Operator* (**var** S : *Image*, **var** D : *Image*)
  foreach *pixel* $p \in S$ **do**
    $D(p) \leftarrow \max\{S(q) : q \in N_G(p) \cup p\} - \min\{S(q) : q \in N_G(p) \cup p\}$
  od
end

Hinweise:

(1) Über $I(p)$ erhält man den Grauwert des Pixels p. Je größer der Grauwert, desto heller ist das Pixel.

(2) N_G bezeichnet die Nachbarschaft eines Pixels bezüglich eines gewählten Gitters. $N_G(p)$ liefert die Menge der zu p benachbarten Bildpunkte. Verwenden Sie hier als Gitter die 8er-Nachbarschaft.

Lösungsidee:
Es handelt sich um den sogenannten morphologischen Gradienten. Die Anwendung des Operators ergibt ein Kantenbild. Als Beispielbild eignet sich ein schwarzes Rechteck auf weißem Hintergrund.

Aufgabe 5: Bildverarbeitung

Bildverarbeitungsoperatoren:

Gegeben sei ein zweidimensionales Bild I, welches ausschliesslich schwarze und weisse Pixel enthält (Pixelwert 0 = schwarz, Pixelwert 255 = weiss). Dieses Bild sei durch die Anwendung eines Kantendetektors entstanden, der Kantenpixel als weiss markiert und alle anderen Pixel auf schwarz gesetzt hat.

Desweiteren sei folgender Bildverarbeitungsoperator *Operator(S,D)* gegeben (S bezeichnet dabei das Eingabebild, D das Ergebnisbild):

proc *Operator* (**var** S : *Image*, **var** D : *Image*)
  foreach *pixel* $p \in S$ **do**
    $D(p) = \max\{S(q) : q \in N_G(p) \cup p\}$
  od
end

Hinweise:

- Über $S(p)$ erhält man den Grauwert des Pixels p im Eingabebild S. Analog wird über $D(p)$ der Grauwert im Ergebnisbild D an der Position p gesetzt.
- N_G bezeichnet die Nachbarschaft eines Pixels bezüglich eines gewählten Gitters. $N_G(p)$ liefert die Menge der zu p benachbarten Bildpunkte. Verwenden Sie hier als Gitter die 8er-Nachbarschaft.

Aufgaben:

- Welches Ergebnis liefert der Operator angewendet auf das Bild I? Begründen Sie Ihre Antwort und geben Sie ein einfaches Beispiel (Bild) an.

  Lösungsidee:
  Es handelt sich um den Dilatationsoperator. In obiger Implementierung wird ein quadratisches Strukturelement der Größe 3×3 verwendet. D.h. ein Kantenpixel wird auf die Größe 3×3 vergrößert. Ein einfaches Beispiel ist ein weisses Rechteck mit Linienstärke 1 auf schwarzem Hintergrund. Nach Anwendung des Operators sind die einzelnen Linien auf Stärke 3 verdickt.

- Nennen Sie ein in der Vorlesung behandeltes Verfahren aus der Videoinhaltsanalyse, in dem dieser Operator eingesetzt wird. Erläutern Sie, warum dieser Einsatz erforderlich ist.

  Lösungsidee:
  Der Operator wird im Algorithmus zur Berechnung der Edge-Change-Ratio (ECR)

eingesetzt. Die Verdickung der Kantenpixel dient dazu, leichtes Zittern der Kamera – d.h. leichte Verschiebungen von Kantenpositionen – auszugleichen.

Applikationen:

Im Rahmen einer Teleteaching-Veranstaltung soll ein Modul eingesetzt werden, welches die Kamera automatisch den Bewegungen des Dozenten nachführt. Ihre Aufgabe ist es, ein solches Modul zu entwickeln.

Gehen Sie davon aus, dass zu Beginn einer Veranstaltung die Kamera einmal auf den Kopf des Dozenten ausgerichtet wird. Nehmen Sie desweiteren an, dass der Steuerungsmotor der Kamera Anweisungen wie „10 Pixel nach links“ umsetzen kann.

- Geben Sie ein Verfahren an, mit dem Bewegungen des Dozenten festgestellt und berechnet werden können, und erläutern Sie dieses kurz. Wie ermitteln Sie aus diesen Bewegungen die Nachführung der Kamera?

  Lösungsidee:
  Auf der Basis von zwei aufeinanderfolgenden Bildern werden Bewegungsvektoren berechnet. Da der Hintergrund statisch ist, werden Bewegungen nur durch den Dozenten verursacht. Ein sinnvolles Maß für die Kameranachführung ist der aus den Bewegungsvektoren berechnete Durchschnittsvektor.

- Was versteht man unter dem Blendenproblem? In welcher Art und Weise kann dieses Problem im gegebenen Szenario auftreten? Geben Sie ein Beispiel an und erläutern Sie, wie Sie dieses Problem in Ihrem Ansatz berücksichtigen.

  Lösungsidee:
  *D*a Verfahren zur Berechnung von Bewegungsvektoren auf einer lokalen Umgebung agieren, tritt das sogenannte Blendenproblem auf. D.h. es läßt sich nicht zuordnen, wohin sich eine bestimmte Struktur bewegt hat. In obigen Szenario kann dieser Fall auftreten, wenn der Dozent z.B. ein einfarbiges Hemd trägt. Die Verschiebung von Blöcken innerhalb der einfarbigen Fläche ist nicht eindeutig bzw. kann nicht ermittelt werden.

  Das Problem läßt sich lösen, in dem man nur ausreichend texturierte Bereiche in die Berechnungen einfliessen lässt.