5. Übungseinheit

Bildmanipulation – Green-Box

Die Greenbox ist eine Methode Filmszenen zu synthetisieren, dabei wird eine Szene mit handelnden Personen vor einem virtuellen Hintergrund dargestellt. Das Verfahren hat sich aus der Blue-Box, einer analogen Technik die mit Doppelbelichtung arbeitete, entwickelt. Mit der digitalen Filmtechnik hat sich die Farbe Grün als besser geeignet herausgestellt. In dem Verfahren werden grüne Pixel in der primären Szenerie durch ein jeweiliges Hintergrundpixel ersetzt.

Datenmaterial

Es wird eine Serie von Digitalbildern von Personen, die in der Green-Box aufgenommen wurden, als Primärbilder zur Verfügung gestellt. Wählen Sie aus dem Internet beliebige Hintergrundbilder. Die Größe muss in bei der Auswahl nicht übereinstimmen, sie kann in einem weiteren Schritt mit *imresize()* angepasst werden.

imread(), imresize(), imagesc(), axis image, figure

Identifikation der Referenzfarbe

In diesem Schritt werden die Hintergrundpixel bestimmt. Die Pixel sind grün, der exakte Farbwert kann jedoch von Pixel zu Pixel leicht unterschiedlich sein und überdies kann die Helligkeit durch inhomogene Beleuchtung verschieden sein.

Es ist daher sinnvoll nicht in der RGB Repräsentation zu arbeiten, sondern die Helligkeitskomponente auszuschließen und nur die reinen Farbkomponenten zu verwenden. Ein geeignetes Format ist der YUV-Farbraum.

- Transformieren Sie das Bild in den YUV-Raum, verwenden Sie dazu die Funktion *rgb2ntsc().*
- *Hinweis:* Für die weitere Bearbeitung müssen Sie auf die einzelnen Farbkomponenten der Bilder zugreifen, dazu haben sie folgende Möglichkeiten:
  - **a(i,j,2)** adressiert die *zweite* Farbkomponente des Pixels an der Position *(i,j),* d.h. Zeile *i*, Spalte *j*.
  - **a(i,j,:)** spricht alle Komponenten des Pixels an
  - **a(:,:,1)** spricht die *erste* Farbkomponente des *gesamten* Bildes an.
- Der Farbanteil des Bildes besteht aus zwei Werten *(U,V),* was einer vektoriellen Größe entspricht. Für den Vergleich zweier Vektoren **a** und **b** verwenden wir den einfachen Euklidschen Abstand:

$$d(a,b) = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2}$$

- Zur robusten Bestimmung des Hintergrundes muss ein möglichst repräsentativer Vergleichswert verwendet werden. Wir verwenden den

Mittelwert aus einer manuell definierten Region. Benützen Sie dazu die Funktion *MBVrefColor().*

Hintergrundpixel

Durch Vergleich mit dem Referenzwert, werden die Hintergrundpixel bestimmt.

- Bestimmen Sie einen geeigneten Abstand *d*, um den Hintergrund zu identifizieren.
- Ersetzen Sie den Hintergrund durch eine beliebige Szenerie.
- Zum Größenabgleich zwischen Primärbild und Hintergrundszene verwenden Sie die Funktion *imresize().*
- Achten Sie darauf, dass Sie bei der Synthese des Bildes wieder im RGB-Raum arbeiten, denn nur dieses Format kann dargestellt werden.
- Hinweise:
  - Mit folgender Sequenz können Sie ein leeres RGB-Bild in der Größe der Vorlage *A* erzeugen:
    B=uint8(zeros(size(A))
  - Achten Sie darauf, dass RGB-Bilder abhängig von Datentyp verschiedene Maximalwerte verlangen:
    double -> 1; *uint8 -> 255*
- Testen Sie das Verfahren anhand verschiedener Bilder

Verbessertes Verfahren

Mit obigem Verfahren werden eventuell nicht nur Hintergrundpixel identifiziert, sondern auch Objektpixel, die zufällig in diesen Farbbereich fallen. Unter der Annahme, dass der Hintergrund eine zusammenhängende Region darstellt, kann er durch einen *Region-Growing (Flood-Fill)* Algorithmus bestimmt werden.

In der Vorlesung wurde eine Variante des Region-Growings vorgestellt. Diese *rekursive* Implementierung ist jedoch nicht geeignet, da bei der vorliegenden Bildgröße der Stack schnell zu klein wird.

- Entwickeln sie eine *iterative* Methode für das Region-Growing, bei der Sie den Stack selbst verwalten.
- Der Stack ist ein Array in dem Sie die Koordinaten der bearbeiteten Pixel ablegen und bei Bedarf wieder zurückspeichern. Der Stackpointer ist eine Variable, die auf das letzte Element im array zeigt.
- Matlab alloziert Felder dynamisch, d.h. Sie müssen für den Stack keinen Speicher reservieren, sondern Sie können die Zugriffe direkt durchführen. Zwei Beispiele:

```
ptr=ptr+1;                  % Ablage am Stack
stack(ptr)=value

value=stack(ptr);           % Zugriff auf den Stack
ptr=ptr-1;
```

- Während der Entwicklung arbeiten Sie mit kleineren Bildern, da die Laufzeit durchaus lang sein kann.
- Verwenden Sie den entwickelten Region-Growing Algorithmus zur Bestimmung des Hintergrundes.