ÜBUNG 183.651

Video Analysis

1. Übungsaufgabe

SS 2017

Sebastian Zambanini

Computer Vision Lab
Institut für Rechnergestützte Automation
http://www.caa.tuwien.ac.at/cvl/course/video-analysis-ue/

vidana@caa.tuwien.ac.at

1 Allgemeines

Dieser Abschnitt enthält allgemeine Informationen zur Organisation der UE Video Analysis.

1.1 Genereller Ablauf

Innerhalb des Semesters sind von den Studierenden in 2er-Gruppen zwei Aufgaben zu bearbeiten. Für jede Aufgabe gilt es ein Programm zu entwickeln, das auf vorgegebenen Videodaten die jeweilige Aufgabe lösen soll. Die Studierenden sollen dabei selbstständig möglichst leistungsfähige Methoden recherchieren und implementieren. Der gewählte Lösungsweg ist dabei zu dokumentieren und an zwei Terminen innerhalb des Semesters zu präsentieren.

1.2 Programmiersprache

Die Programmiersprache ist prinzipiell frei wählbar, empfohlen wird jedoch MATLAB oder C++ (inkl. OpenCV). Es muss auf alle Fälle eine auf Windows lauffähige Version abgegeben werden, die sich über die in Abschnitt 2 definierte Schnittstelle aufrufen lässt. Eine Abweichung des Inputs oder Outputs des abgegebenen Programms von der Vorgabe führt unweigerlich zu Punkeabzügen.

1.3 Abgabe

Die im Abschnitt 2 beschriebene Aufgabe ist von jeder Übungsgruppe selbstständig zu lösen. Die Abgabe erfolgt per Mail an vidana@caa.tuwien.ac.at. Abzugeben sind:

- Source-Code
- Kompilierte Version des Programms (falls nicht in MATLAB entwickelt)
- PDF-Dokumentation (1-2 Seiten): kurze Beschreibung des abgegebenen Programms und des gewählten Lösungsweges

1.4 Benotung

Für die 1. Aufgabe (Vordergrund-Segmentierung) werden 10, für die 2. Aufgabe 15 Punkte vergeben. Bewertet wird vor allem die Originalität und Leistungsfähigkeit der entwickelten Lösung. Für ein Mindestmaß von 5 Punkten für die 1. Aufgabe muss zumindest ein einfaches adaptives (lernendes) Verfahren wie das in Abschnitt 3.2 beschriebene *Colour Mean and Variance* Verfahren implementiert werden.

1.5 Daten

Den Studierenden werden Videodaten inkl. Ground Truth Annotationen zum Testen des eigenen Programms zur Verfügung gestellt. Diese Daten sind von den Studierenden vertraulich zu behandeln und dürfen nur im Rahmen der Laborübung verwendet werden. Auf keinen Fall dürfen die Videodaten oder Ergebnisse weitergegeben oder in irgendeiner Form öffentlich zugänglich gemacht werden.

2 Aufgabe 1: Vordergrund-Segmentierung in Videos

Bei der 1. Aufgabe soll ein Verfahren entwickelt werden, das Vordergrundobjekte in Videos detektiert. Das Programm soll hierfür zuerst für eine gegebene Videosequenz in einer Initialisierungsphase (kein Vordergrund-Objekt im Video) den Hintergrund lernen. Im Anschluss soll das Programm in der Szene auftretende Vordergrundobjekte erkennen und als binäre Maske zurückliefern.

Das Programm soll dabei entweder als Matlab-Funktion oder als kompiliertes Programm, das sich über die Windows-Kommandozeile aufrufen lässt, abgegeben werden. Die Eingabeparameter sind wie folgt:

`path`: Pfad zum Ordner der Bilddateien, die die Frames der Videosequenz zeigen
`filename`: Präfix der Bilddateien. Das 1. Frame hat den Suffix `'_0000.jpg'`
`frames`: Anzahl der Video-Frames
`initframes`: Anzahl der Frames, die für die Initialisierung verwendet werden können

Als Ausgabe soll das Programm für jedes Eingabeframe (mit Ausnahme der Frames der Initialisierungsphase) ein SW-Bild zurückliefern, in dem Vordergrund (weiß) und Hintergrund (schwarz) gekennzeichnet sind. Diese Bilddateien erhalten den Präfix `'Seg_'` und sollen als PNG-Bilder abgespeichert werden.

Beispiel für Matlab: die Funktion `detectForeground('.\video1\','video1',200,50)` detektiert den Vordergrund in einer Videosequenz, die im Ordner `'.\video1\'` in den Bilddateien `'video1_0000.jpg'` bis `'video1_0200.jpg'` gespeichert ist. Die ersten 50 Frames können zur Initialisierung verwendet werden, das Programm sollte somit im Ordner `'.\video1\'` 150 Bilder (`Seg_video1_0051.png` bis `Seg_video1_0200.png`) speichern, die den detektierten Vordergrund zeigen.

Die Eingabebilder liegen immer als Farbbilder im RGB-Format vor. Weiters kann angenommen werden, dass die Bilder mit einer statischen Kamera aufgenommen wurden (d.h. die Kamera bewegt sich nicht).

Tipp

In Matlab kann der Code zum Laden der Bilder folgendermaßen aussehen:

```
for idx=1:frames
    imgname = sprintf('%s%s_%04d.jpg',path,filename,idx)
    img = imread(imgname)
    % ---------------------------------
    % weiterer Code
    % ---------------------------------
end
```

3 Theorie: Vordergrund-Segmentierung in Videos

Verfahren zur Vordergrund-Segmentierung (Bewegungserkennung) haben eine große Bedeutung in der Bilddatenverarbeitung und Bildcodierung, wie z. B. der Videoüberwachung, der Objekterkennung und -verfolgung oder der Kompression von Videosequenzen. Die mit Abstand gängigsten Verfahren zur Erkennung von Bewegung in von statischen Kameras aufgenommene Bildsequenzen stellen die sogenannten *Background Subtraction* Algorithmen dar. Dabei wird, vereinfacht gesagt, ein (bewegtes) Vordergrundobjekt als der Unterschied zwischen dem gerade betrachteten Bild und dem Bild des statischen Szenenhintergrunds detektiert. Grundsätzlich wirft sich bei dieser Vorgehensweise die Frage auf, wie die automatische Aufnahme des Bildes des statischen Hintergrunds von statten gehen soll. Die theoretische Anforderung an Verfahren zur Bestimmung des Hintergrundbildes ist, dass das Hintergrundbild nicht statisch sein darf, sondern sich anpassen muss an:

- Beleuchtungsänderungen
  - graduelle (veränderte Sonneneinstrahlung im Laufe des Tages)
  - plötzliche (Wolken, Schatten, etc.)
- Bewegungsänderungen (multimodaler Hintergrund)
  - Wackeln der Kamera
  - Bewegung im Hintergrund (Bäume, Wellen, Vorhänge, etc.)
- Änderungen in der Geometrie des Hintergrunds
  - Parkende Autos, geöffnete Türen, verschobene Gegenstände,...

3.1 Adjacent Frame Differencing (AFD)

In der einfachsten Variante wird der Absolutbetrag der Pixeldifferenz zwischen aktuellem und letzten Bild (das als Hintergrundbild fungiert) mit einem benutzerdefinierten Schwellwert verglichen.

$$|Bild_t - Hintergrund_t| > Schwellwert \tag{1}$$

Übersteigt die Differenz den Schwellwert, wird das Pixel als bewegt erkannt. Im praktischen Einsatz wird das resultierende Differenzbild vor der Schwellwertoperation tiefpassgefiltert, um Rauschen zu unterdrücken. Zusätzlich werden Löcher in der detektierten Vordergrundmaske mittels morphologischer Operationen geschlossen. Dieses Verfahren führt zu verbesserten Ergebnissen, erhöht jedoch auch die benötigte Rechenleistung.

3.2 Colour Mean and Variance (CMV)

Das Hintergrundmodell dieses Algorithmus wird durch einen pixelweisen Durchschnittswert μ und Varianz σ^2 repräsentiert. Dieser Ansatz basiert auf der Annahme, dass ein Hintergrundpixel und das zu erwartende Bildrauschen durch eine Normalverteilung $N(\mu, \sigma^2)$ modelliert werden

kann. Eine einfache Implementation benutzt den Standard RGB Farbraum (Kamerafarbraum). Das Hintergrundmodell und ist wie folgt definiert:

$$B = (\mu, \sigma) = (\mu =< \mu_R, \mu_G, \mu_B >, \sigma =< \sigma_R, \sigma_G, \sigma_B >) \tag{2}$$

Bewegung wird durch eine Schwellwertoperation auf den absoluten Differenzen zwischen dem Pixelwert x_c und dem Durchschnitt μ_c der einzelnen Farbkanäle detektiert. Wenn die Differenz größer als die Standardabweichung $K\sigma_c$ ist, wird das Pixel als Vordergrund markiert, anderenfalls als Hintergrund.

$$\Delta_c = |x_c - \mu_c|, \quad \text{für } c \in R, G, B \tag{3}$$

$$\begin{array}{lll} \text{If} & \Delta_R > K\sigma_R \vee \Delta_G > K\sigma_G \vee \Delta_B > K\sigma_B & \text{Pixel :=Vordergrund} \\ \text{Else} & & \text{Pixel :=Hintergrund} \end{array} \tag{4}$$

Im Gegensatz zu dem vorher beschriebenen Algorithmus wird der Schwellwert nicht für das ganze Bild festgelegt, sondern durch die K-fache Standardabweichung $K\sigma_c$ für jedes Pixel (und Farbkanal) separat bestimmt. K wird zumeist auf 3.5 gesetzt, da dieser Wert eine 99.9% Wahrscheinlichkeit repräsentiert, dass der Pixelwert nicht zur Gaußverteilung $N(\mu, \sigma^2)$ gehört.

Das Hintergrundmodell wird aus den ersten N Bildern einer Bildsequenz gebildet:

$$\mu_c = \frac{1}{N} \sum_{t=1}^{N} X_c(t), \qquad \text{mit c} \in R, G, B \tag{5}$$

$$\sigma_c^2 = \frac{1}{N} \sum_{t=1}^{N} (X_c(t) - \mu_c)^2 \tag{6}$$

Für die folgende Bilder wird das Hintergrundmodell aktualisiert, um sich an Beleuchtungsänderungen zu adaptieren.

$$\begin{aligned} \mu_c(t+1) &= (1-\alpha)\mu_c(t) + \alpha x_c(t) \\ \sigma_c(t+1) &= (1-\alpha)\sigma_c(t) + \alpha |x_c(t) - \mu_c(t)| \end{aligned} \tag{7}$$

Dabei werden, abhängig ob es sich um ein Vordergrund- oder Hintergrundpixel handelt, verschiedene Werte für die Lernrate α eingesetzt (typischerweise $\alpha_{HG} >> \alpha_{VG}$). Dieses als Selektivität bekannte Prinzip ermöglicht ein schnellere Adaption an langsame (geringe) Änderungen und eine langsame Adaption an starke Bildänderungen (echte Bewegung, Stillstand, etc.).