Computer Vision Challenge

Gruppe 21

Nguyen, Noah Binh
03683505

Smajli, Fatbardh
03689615

Ruano Martinez, Roberto
03733955

Butsch, Florian
03663488

Caneda Portela, Marta
03721184

12/07/2020

Inhaltsverzeichnis

1 Problemstellung und Motivation

Aufgrund der Corona-Krise finden heutzutage mehr Video-Calls und Konferenzen innerhalb der eigenen vier Wände statt. Um möglichst viele unangebrachte Szenen im Hintergrund zu vermeiden, soll mit Hilfe des Programms mehrere verschiedene Hintergrundarten angezeigt werden. In unserem Projekt, geht es um die Lösung dieses Problems durch die Implementierung eines Programs, das Hintergrund und Vordergrund unterscheiden kann. Dabei bestimmt die Auswahl der Modi (foreground, background, overlay, substitute) welche Hintergrundart angezeigt werden soll.

2 Methodik - Background Subtraction

Als Methodik haben wir uns für die Background Subtraction entschieden, indem wir für das Background Modeling den *Mean* von allen Bildern vom Tensor genommen haben, bis auf ein Bild. Unter anderem haben wir auch Modelle wie Gaussian Mixture Model (GMM) oder Markov Random Field (RMF) ausprobiert, nur leider waren wir mit den Ergebnissen nicht zufrieden. Die Idee der Background Subtraction ist ganz einfach: Es werden N+1 Bilder in den Tensoren geladen und es soll für das letzte Bild im Tensor die Maske berechnet werden. Sei N=4. Dann wird später der Mean aus den ersten 4 Bildern genommen und dann vom letzten Bild im Tensor abgezogen. Beim nächsten Aufruf von *next()* werden die nächsten Bilder in den Tensor geladen. Die Funktionsweise von *next()* ist in Abbildung 1 zu sehen. Die Berechnung des Backgrounds [5, 3] kann allgemein wie folgt dargestellt werden:

$$B(x, y, t) = I(x, y, t - 1) \tag{1}$$

Hier ist *B(x,y, t)* der Background zum Zeitpunkt *t* und *I(x, y, t)* das normale Bild zum Zeitpunkt *t*. Erweitert man den Ansatz mit dem Meanfilter folgt daraus:

$$B_{mean}(x, y, t) = \frac{1}{n} \sum_{i=0}^{n-1} I(x, y, t - i) \tag{2}$$

Somit folgt für die Background-Subtraction in unserem Projekt:

$$I_{difference} = |I(x, y, t) - B_{mean}| \tag{3}$$

Damit nun eine Foreground Mask erstellt werden kann, muss vor dem Post-Processing ein Schwellenwert ϕ (threshold) eingestellt werden. Somit folgt endgültig für die Background-Subtraction:

$$I_{difference} > \phi \tag{4}$$

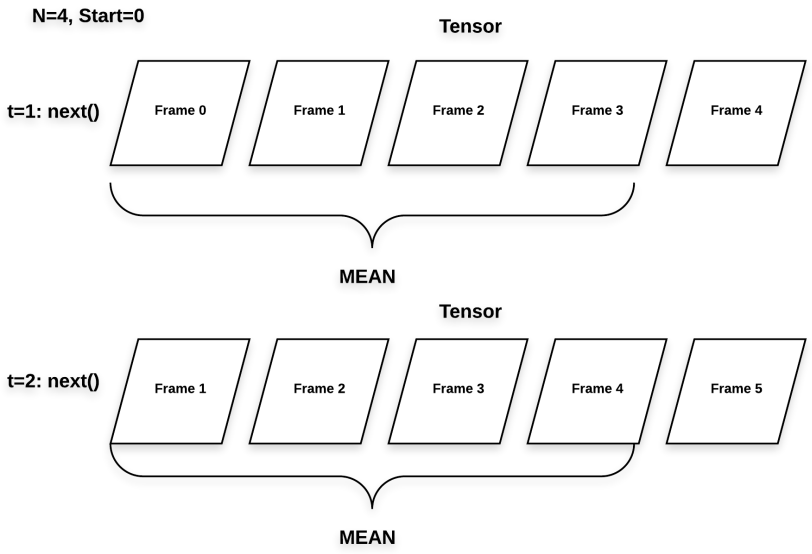

Abbildung 1: Ablauf der next() Funktion

3 Post-Processing

In diesem Abschnitt sollen die Verarbeitungsschritte erläutert werden, die wir bei der Erstellung unserer Maske angewendet haben. Aufgrund der Aufgabenstellung konnten wir nur begrenzt unsere Ideen für die Background-Subtraction umsetzen, weshalb wir leider den *Adaptive Background* [4] Ansatz nicht umsetzen konnten. Dadurch haben wir nur einen konstanten Schwellwert ϕ für alle Bilder genommen, der hauptsächlich durch Trial and Error"bestimmt wurde. Wir haben es auch versucht, die Otsu Schwellmethode anzuwenden. Leider passt es nicht für unsere Anwendung, weil unsere Bilder keine gute Farbsegmentierung haben.
Mit Hilfe der Matlab Funktion *bwareafilt* haben wir dem Algorithmus mitgeteilt, dass er im binären Bild den größten "Blop"(Fleck) im Bild nehmen sollte, um einzelne weiße Stellen im Bild auszublenden. Somit wurde immer der Körper der Person erkannt. Danach wird die Funktion *imfill* [2] genutzt, um Löcher innerhalb des extrahierten Flecks zu füllen. So haben wir einige Löcher in der Brust etc. stopfen können, die durch den Schwellenwert verloren gegangen sind. Die letzte Funktion, die unsere binäre Maske geholfen hat war *imclose*[1], die ähnlich wie *imfill* agiert. Dabei wird anhand einer auswgählten Form und einem Radius versucht, die Lücken zu füllen.

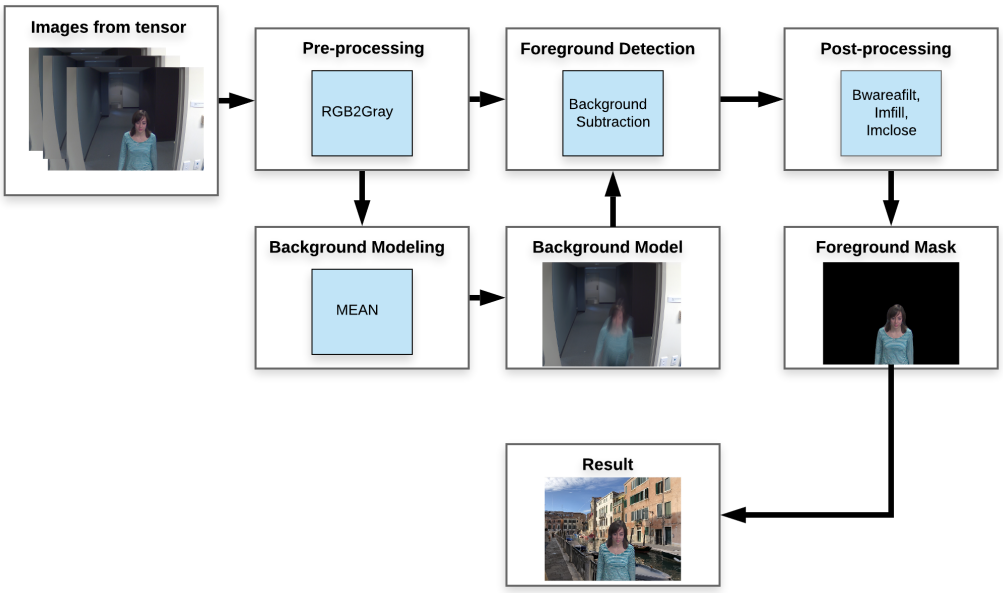

Abbildung 2: Ablauf des Pogramms

4 Graphical User Interface (GUI)

Das Design der GUI wurde mit Hilfe von Äpp Designerërstellt. Visuelle Komponenten und Bedienelemente können per Drag Drop aus der Komponentenbibliothek ausgewählt werden, um ein Layout für die Gestaltung der Benutzeroberfläche zu erstellen. Das Verhalten und die Interaktion mit dem betreffenden Funktionen kann mit dem integrierten Editor programmiert werden.

Die GUI lässt sich durch den Befehl *start_gui* starten. Der Benutzer hat dann die Möglichkeit, mehrere Eingaben auszuwählen, wie z.B. den Ordnerpfad einer Szene oder einen bestimmten Startpunkt in der Framesequenz. Die Parameter *L* und *R* für die Auswahl der Kamera-Ordner können ebenfalls geändert werden. Den Rendering-Modus kann man im Dropdown-Menü angeben (Auswahl zwischen *background, foreground, overlay, substitute* und *bonus*). Für die Modi *substitute* und *bonus* sollte ein virtueller Hintergrund in Form eines Bildes oder eines Videos geladen werden. Der Pfad zu diesen Dateien kann ebenfalls vom Benutzer festgelegt werden. Für die Speicherung des gerenderten Films ist ebenfalls die Angabe eines beliebigen Dateipfades erforderlich. Außerdem verfügt die GUI über vier Steuerelemente. Wenn man auf *Start* drückt, werden die Berechnungen durchgeführt. Durch Markieren des Kästchens *Show Rendering* werden sowohl die Stereoinputs als auch der maskierte Outputstream angezeigt. Der Knopf *Stop* hält die Visualisierung an. Wenn *Loop Video* aktiviert ist, wird nach Ende des Ordners neu angefangen am Anfang des Ordners.

Der aktuelle Stand und Fortschritt des Programms wird ebenfalls überwacht. Die Protokollmeldungen werden in einem Fenster auf der rechten Seite angezeigt. Auch die Anzahl der verarbeiteten Frames wird während der Berechnungen ausgegeben. Schließlich wird auch die *elapsed time* an-

gezeigt, entweder am Ende der Berechnungen oder für den Fall, dass der Benutzer den Ablauf des Programms durch *Stop* unterbricht.

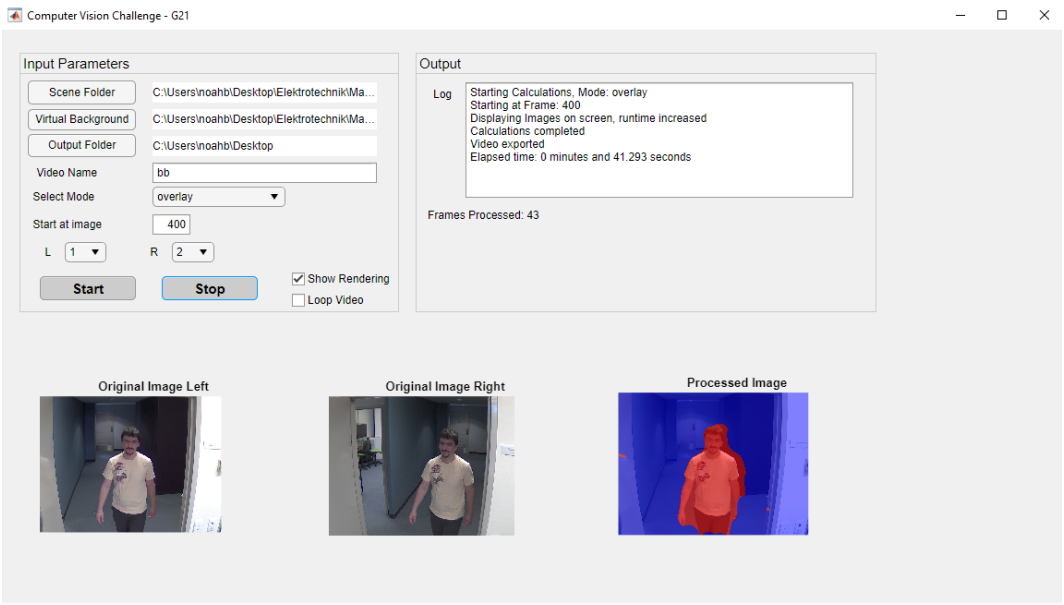

Abbildung 3: Abbildung der GUI

5 Ergebnisse/Abbildungen

Die Segmentierung unseres Programmes liefert für die meisten Szenen klare und durchaus gute Ergebnisse. Die größten Probleme hat das Programm, wenn die Szenen mehrere Leuten und verschiedenen Lichtverhältnisse beinhalten. Bei der Implementierung dieses Algorithmus ist uns aufgefallen das man nicht das perfekte Ergebnis bekommen kann für die Segmentierung und eine Abwegung treffen muss, ob eine komplette Erkennung der Person wichtiger ist oder ein genaueres Ausschneiden der Person. Wir erachten die korrekte Erkennung aller Personen als wichtiger als das genaue Ausschneiden.

Der Wert von N, welcher für die Hintergrundberechnung verwendet wird hat großen Einfluss auf das Endergebnis. Durch Tests mit verschiedenen Szenen haben wir entschieden, diesen Wert auf 7 zu setzen. Das bedeuted, das jeweils 7 Bilder für die Berechnung der Maske des 8ten Bildes verwendet werden. Dies hat zur Folge, dass beim Start erst das 8te Bild maskiert wird. Nachfolgende Bilder für die Maske zu nehmen liefert sehr schlechte Ergebnisse mit unserer Methode, weshalb akzeptieren, die ersten 7 Bilder nicht zu analysieren.

Im Schnitt braucht unser Programm circa 28-29 Minuten für einen gesamten Ordner, abhängig jedoch von der Leistung des Rechners. Eine bessere Laufzeit kann erreicht werden, indem die Stereobilder und das Ergebnis nicht angezeigt werden, sondern nur am Ende das generierte Video. Im Folgenden werden verschiene Bilder aus verschiedenen Szeneorder im Modus foregroundängezeigt.

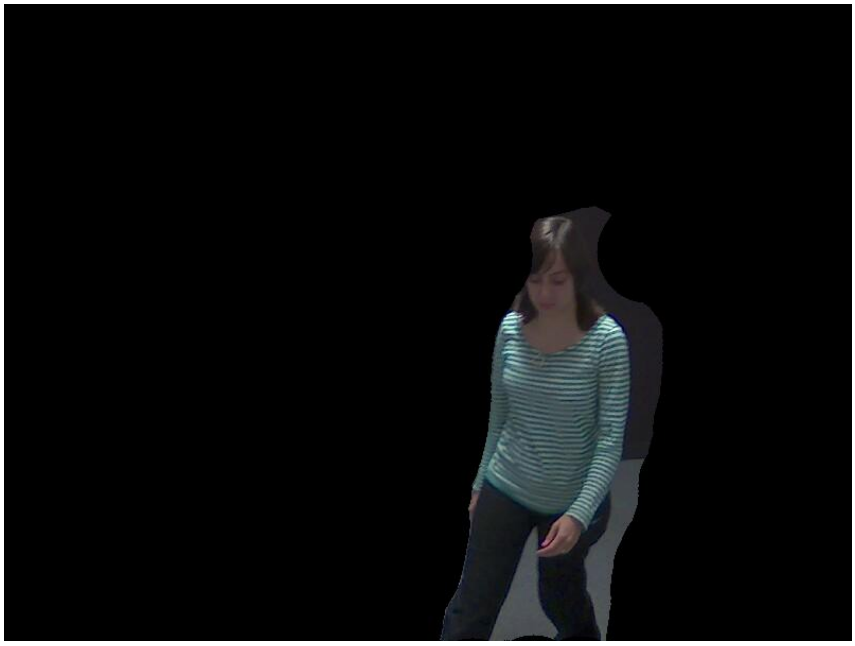

Abbildung 4: P1E_S1. Modi: foreground

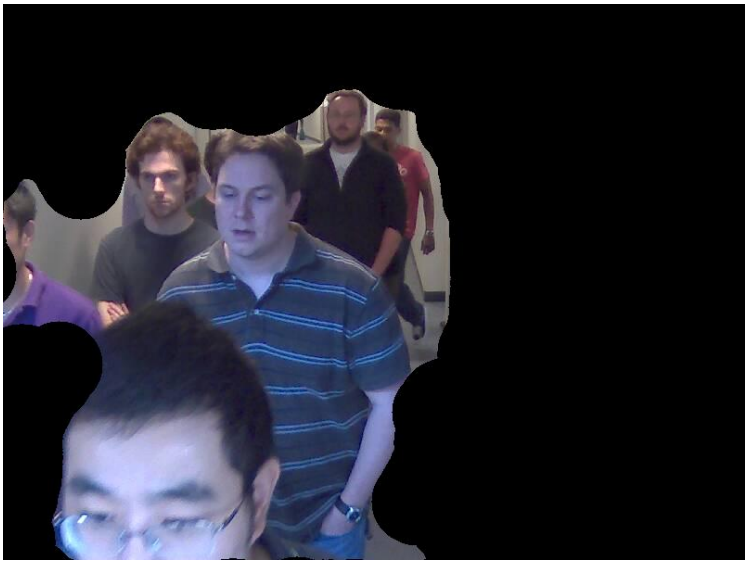

Abbildung 5: P2L_S5. Modi: foreground

Abbildung 6: P1L_S3. Modi: foreground

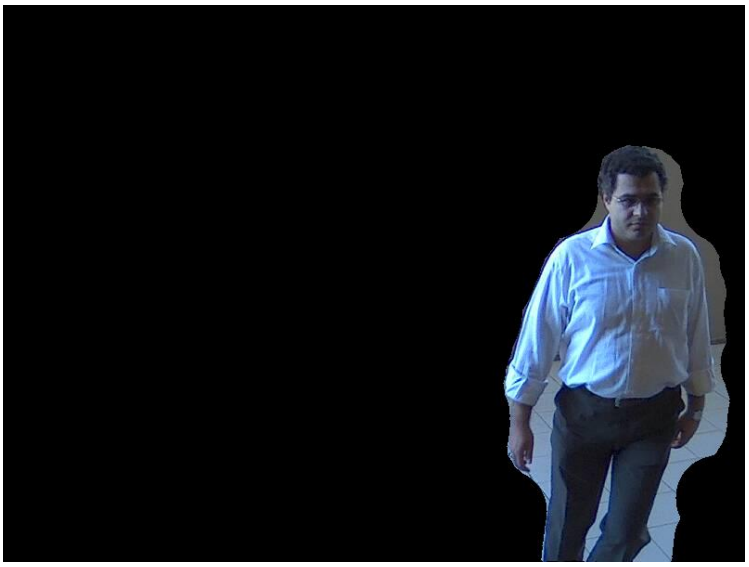

Abbildung 7: P2E_S2. Modi: foreground

Abbildung 8: P2E_S4. Modi: foreground

Abbildung 9: P2E_S5. Modi: foreground

Literaturverzeichnis

[1] https://de.mathworks.com/help/images/ref/imclose.html.

[2] https://de.mathworks.com/help/images/ref/imfill.html.

[3] Boufares, O. and Aloui, N. and Cherif A. Adaptive threshold for background subtraction in moving object detection using stationary wavelet transforms 2d. *(IJACSA) International Journal of Advanced Computer Science and Applications*, 7(8):29–36, 2016.

[4] McHugh,J. M. and Janusz, K. and Venkatesh S. Foreground-adaptive background subtraction. *(IEEE SIGNAL PROCESSING LETTERS*, 16(5):390–393, 2009.

[5] B. Tamersoy. *Background Subtraction*. The University of Texas, 2009.