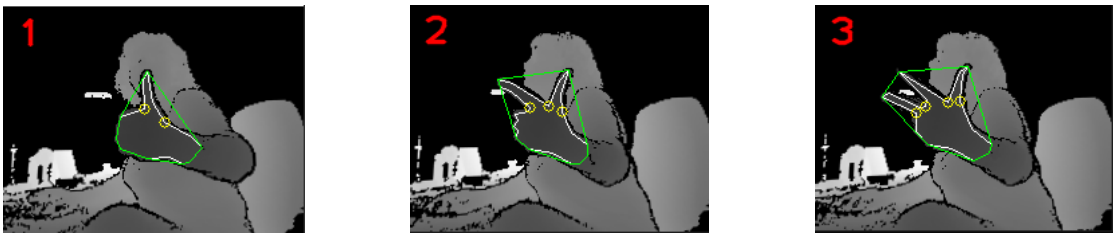
Praktikum 3: Klassifikation der Handgeste

1. Ziel

Aufbauend auf dem zweiten Praktikum werden in diesem Praktikum Merkmale vom Binärbild der Hand abgeleitet, um nun die Entscheidung über die Art der Geste treffen zu können. Die Implementierung der Bildverarbeitungsmethoden erfolgt wiederum mit der C++ Bibliothek OpenCV und dem Kamera SDK royale.



(a) Geste 1 (b) Geste 2 (c) Geste 3

Abbildung 1: Ergebnisse der Gestenauswertung für das Beispielvideo

2. Lernziele

- Schreiben und Verwendung eigener Funktionen
- Einbinden und Verwendung von externen Bibliotheken einschließlich des Gebrauchs der Dokumentation
- Anwendung von Bildverarbeitungsalgorithmen zur Auswertung der Bilddaten

3. Aufgaben

Setzen Sie die in den beiden vorangehenden Praktika begonnenen Verarbeitungen fort. Die Ergänzungen des dritten Praktikums können alle in die Auswertefunktion des zweiten Praktikums integriert werden, so dass sie sowohl bei der Livebild-Auswertung (Übergabeparameter 1) als auch bei der Auswertung des aufgezeichneten Videos (Übergabeparameter 3) aufgerufen werden.

- Um die Auswertung zu vereinfachen, soll zunächst der Ramer–Douglas–Peucker Algorithmus angewendet werden, der eine Kontur zu einer ähnlichen Kurve mit weniger Punkten dezimiert, indem der Abstand zwischen der alten und der approximierten Kurve kleiner oder gleich einer angegebenen Genauigkeit ist. Umgesetzt ist der Algorithmus in der OpenCV-Funktion approxPolyDP(). Nutzen Sie im Praktikum als

Genauigkeit 2 Pixel. Zeichnen Sie diese vereinfachte Kontur anstatt der genauen Kontur in das Tiefenbild ein.

- Bilden Sie anschließend die konvexe Hülle der approximierten Kontur (grüne Linie in Abb. 1). Hierzu können Sie die Funktion convexHull() verwenden. Zeichnen Sie die konvexe Hülle als gefülltes weißes Objekt in ein leeres Bild mit der Funktion drawContours() ein.
- Berechnen Sie anschließend die Konvexitätsfehler der approximierten Kontur durch Anwendung der Funktion convexityDefects(). Zeichnen Sie die Punkte, wie bei den Beispielen in Abb. 1 als gelbe Kreise ein. In der OpenCV-Dokumentation gibt es ein entsprechend erläuterndes Bild, was unter dem Konvexitätsfehler zu verstehen ist. Um kleine Fehler ausschließen zu können, muss anschließend der Abstand jedes Fehlerpunktes zur konvexen Hülle ermittelt werden. Dies könnte man über eine Abstandberechnung des Punktes zu den verschiedenen Konturlinien mit der Punkt-Abstandsgleichnung lösen. Eine vereinfachte Vorgehensweise nutzt die Distanztransformation zur Berechnung. Diese muss zunächst auf dem Binärbild mit der konvexen Hülle ausgeführt werden, indem die Funktion distanceTransform() aufgerufen wird. Der Wert an der Stelle des jeweiligen Fehlerpunktes in diesem Distanzbild charakterisiert die Stärke der Einbuchtung.
- Zum Schluss soll die Klassifikation der Geste in Abhängigkeit von der gefundenen Anzahl von Konvexitätsfehlern erfolgen. Um eine gewisse Skalierungsunabhängigkeit zu erreichen, werden die ermittelten Distanzwerte auf die maximale Distanz bezogen. Ist das ermittelte Verhältnis größer als 1/3 des Radius des größten Kreises, der in die Hülle passt, soll der Konvexitätsfehler mitgezählt werden. Bei zwei Fehlern in der approximierten Kontur handelt es sich um Geste 1, bei drei Konvexitätsfehlern um Geste 2 und bei vier Fehlern um Geste 3. Das Ergebnis der Auswertung soll mit der Funktion putText() in das Bild geschrieben werden (siehe Abb. 1). Bei einer Anzahl kleiner zwei und größer vier soll der Text “keine Geste” ausgegeben werden.

4. Testat

Voraussetzung ist jeweils ein fehlerfreies, korrekt formatiertes Programm. Der korrekte Programmlauf muss nachgewiesen werden. Sie müssen in der Lage sein, Ihr Programm im Detail zu erklären und ggf. auf Anweisung hin zu modifizieren.