Aufgaben zu Bildmerkmalen

Lösen Sie die nachfolgenden Aufgaben und bereiten Sie diese bis zum nächsten Lehrveranstaltungstermin vor. Unterstrichene Aufgaben sind nach Möglichkeit während der Lehrveranstaltung zu lösen.

LB-BM 01.

a) Schreiben Sie ein Programm, das als einziges Argument den Dateipfad eines Bildes erwartet und die SIFT-Schlüsselpunkte dieses Bildes ermittelt, darin einzeichnet und in einem Fenster anzeigt. Verwenden Sie zum Ermitteln die Klasse `cv::SIFT` mit deren Funktion `detect` und zum Einzeichnen die Funktion `cv::drawKeypoints`.

Benötigte Header: `opencv2/features2d.hpp`, `opencv2/highgui.hpp`

b) Erweitern Sie Ihr Programm aus a) derart, dass das angegebene Bild zusätzlich verzerrt wird und anschließend die Schlüsselpunkte neu berechnet, eingezeichnet und in einem weiteren Fenster **klar zuordenbar** angezeigt werden. Die Verzerrung soll dabei aus einer 30-Grad-Drehung um den Bildmittelpunkt und einer insgesamten Vergrößerung des Bildes um einen Faktor 3 (unter Beibehaltung seiner Dimensionen) bestehen und mittels der Funktionen `cv::getRotationMatrix2D` und `cv::warpAffine` realisiert werden. Das verzerrte dargestellte Bild soll dabei dieselben Dimensionen (Breite und Höhe) wie das Originalbild haben.

Zusätzlich benötigte Header: `opencv2/imgproc.hpp`

LB-BM 02.

a) Erweitern Sie Ihr Programm aus LB-BM 01. b) derart, dass übereinstimmende Merkmale zwischen den beiden Bildern (original und verzerrt) gefunden und eingezeichnet werden. Verwenden Sie zum Finden der Übereinstimmungen die Funktion `cv::BFMatcher::knnMatch` mit $k = 1$ und zum Anzeigen der Übereinstimmungen die Funktion `cv::drawMatches`. Die benötigten Deskriptoren für die Schlüsselpunkte können mit der Funktion `cv::SIFT::compute` erzeugt werden.

b) Erweitern Sie Ihr Programm aus a) um einen Filter für Übereinstimmungen nach Lowe (2004). Verwenden Sie dazu $k = 2$ für die Übereinstimmungskandidatensuche und entfernen Sie jene (scheinbaren) Übereinstimmungen, bei denen die Distanz (Eigenschaft `cv::DMatch::distance`) des besten Kandidaten mehr als 80% der Distanz des zweitbesten Kandidaten beträgt.

c) Modifizieren Sie Ihr Programm aus b) derart, dass ein anderer Merkmalerkennungs- und -extraktionsalgorithmus (z.B. ORB) verwendet wird.

LB-BM 03.

a) Modifizieren Sie Ihr Programm aus LB-BM 02. b) derart, dass anstatt des verzerrten Originalbildes ein zweites Bild eingelesen und verwendet wird. Passen Sie das Programm zudem so an, dass es exakt zwei Übergabeparameter (die Dateipfade der beiden Bilder) erwartet.

b) Modifizieren Sie Ihr Programm aus a) derart, dass die Übereinstimmungen zwischen den Bildmerkmalen nicht mehr eingezeichnet werden und erweitern Sie Ihr Programm zudem derart, dass die beiden Bilder in guter, d.h. perspektivisch quasi-korrekter, Näherung zusammengefügt werden. Bestimmen Sie dazu zuerst eine Homographie-Matrix mittels der Funktion `cv::findHomography` nach der RANSAC-Methode aus den Schlüsselpunktkoordinaten (Eigenschaft `cv::KeyPoint::pt`) beider Bilder. Verformen Sie anschließend das zweite Bild derart mittels der Funktion `cv::warpPerspective` und der berechneten Homographie-Matrix, dass beide Bilder in ein gemeinsames, zusammengefügtes Bild eingezeichnet werden können. Zeigen Sie abschließend das zusammengefügte Bild in einem Fenster an.
Hinweis: Es empfiehlt sich, nur die Koordinaten der gefilterten Schlüsselpunkte zur Berechnung der Homographie-Matrix zu verwenden. In den gefundenen Übereinstimmungen können dazu die Eigenschaften ***`cv::DMatch::queryIdx`*** *(im ersten Bild) bzw.* ***`cv::DMatch::trainIdx`*** *(im zweiten Bild) verwendet werden, um auf die entsprechenden Indizes in den* ***`cv::KeyPoint`****-Vektoren zuzugreifen.*

Zusätzlich benötigte Header: `opencv2/calib3d.hpp`

LB-BM 04.

a) Schreiben Sie ein Programm analog zu LB-BM 03. b), das die Dateipfade zweier Bilder erwartet und daraus ein Panoramabild erzeugt. Verwenden Sie dazu an Stelle des homographiebasierten Verfahrens die Klasse `cv::Stitcher` und deren Methode `stitch`. Vergleichen Sie die Panoramabilder mit jenen aus LB-BM 03. b) erzeugten hinsichtlich ihrer Qualität unter Verwendung der beiden Beispielbildpaare. Vermerken Sie Ihre Beobachtungen in Kommentarform im Quellcode.

Benötigte Header: `opencv2/stitching.hpp`

b) Erweitern Sie Ihr Programm aus a) derart, dass beliebig viele Bilder entgegengenommen werden können. Verifizieren Sie die Funktionalität Ihres Programmes mit mehr als zwei Bildern, die zu einem Panoramabild zusammengefügt werden können.