

Übungsblatt 4

Aufgabe 1: Homogene Koordinaten

Implementieren Sie eine Klasse homogene 3D Koordinaten. Nutzen Sie dazu die Matrixoperationen, die Sie in der Vorlesung kennengelernt haben. Dabei soll es folgende Funktionen geben:

- Translation: ein homogener Punkt soll damit verschoben werden können
- RotationX: ein homogener Punkt soll damit um die X-Achse rotiert werden können
- RotationY: ein homogener Punkt soll damit um die Y-Achse rotiert werden können
- RotationZ: ein homogener Punkt soll damit um die Z-Achse rotiert werden können
- Affine: ein homogener Punkt soll damit affin transformiert werden können

Nutzen Sie ihre Klasse um den Punkt $[3, 7, 2]^T$ um den Punkt $[5, 2, 2]^T$ zu verschieben, ihn danach um 90 Grad um die y-Achse zu drehen und dann um den Faktor 3 zu skalieren.

Aufgabe 2: SIFT

Laden Sie sich die Dateien `SIFT_algo.py`, `SIFT_KeyPoint.py`, `SIFT_Params.py` und `SIFT_Visualization.py` aus dem Moodle Kurs herunter. Diese Dateien enthalten eine Implementierung des SIFT Algorithmus. Ihre Aufgabe ist es nun, die ersten 3 Schritte des SIFT-Algorithmus zu implementieren. Nutzen Sie nicht den SIFT-Algorithmus von OpenCV oder die bereits implementierten Methoden aus `SIFT_algo.py`.

Folgende Schritte sollten implementiert werden:

- Erstellen Sie einen Scale Space, die dazugehörigen Delta-Werte und Sigma-Werte
- Berechnen Sie aus dem erstellten Scale Space die DoG-Bilder
- Finden Sie lokale Extrema in den DoG-Bildern

Führen sie alle Schritte des SIFT-Algorithmus in richtiger Reihenfolge aus. Benutzen Sie hierfür ihre eigenen Methoden und die Methoden aus `SIFT_algo.py`.

Das Matching der Keypoints ist nicht Teil dieser Aufgabe. Sie müssen keine Keypoints vergleichen.

SIFT-Implementierung

Bevor Sie mit der Implementierung beginnen, hier noch ein paar Informationen über den Source-Code aus Moodle.

- `SIFT_algo.py` enthält eine minimalisierte Implementierung aller Schritte von SIFT.
- `SIFT_KeyPoint.py` enthält eine Klasse, die einen Keypoint / Extremum repräsentiert.
- `SIFT_Params.py` enthält eine Klasse, die alle Parameter für den SIFT-Algorithmus enthält
- `SIFT_Visualization.py` enthält zwei Methoden, welche die Scale-Space und die dazugehörigen Keypoints / Extrema visualisieren.

Die Klasse `SIFT_Params` enthält alle Parameter, die Sie für den SIFT-Algorithmus benötigen. Die Parameter sind bereits mit Standardwerten initialisiert und müssen eigentlich nicht geändert werden.

Hinweis: Der Scale Space sollte in seiner letzten Oktave ein Bild mit mindestens 12 Pixeln Breite und Länge enthalten. Wählen Sie deshalb ein Bild mit genügend Pixeln aus, aber halten Sie das Bild so klein wie möglich. Das beschleunigt die Berechnung deutlich.

Hinweis 2: Der SIFT Algorithmus erwartet Graustufenbilder mit Werten im Intervall $[0,1]$. Transformieren Sie ihr Bild dementsprechend.

Die Klasse `SIFT_KeyPoint` enthält alle Informationen, die der SIFT-Algorithmus benötigt. Viele dieser Variablen sind jedoch nicht für Sie wichtig. Sie sollten lediglich die Variablen `o`=Oktave, `s`=Scale, `m`=x-Koordinate und `n`=y-Koordinate setzen. Die anderen Variablen werden von der Klasse `SIFT_Algorithm` gesetzt.

Zur Visualisierung können Sie die Methoden `visualize_scale_space` und `visualize_keypoints` aus der Datei `SIFT_Visualization.py` nutzen. Die Methode `visualize_scale_space` zeigt Ihnen den kompletten Scale-Space in einem Plot an. Die Methode `visualize_keypoints` legt Extrema/Keypoints auf diesen Scale-Space.

Ihre Implementierung sollte drei Methoden umfassen, die ebenso in `SIFT_algo.py` zu finden sind: `create_scale_space`, `create_dogs` und `find_discrete_extremas`. Entscheiden Sie selbst, welche Parameter diese Methoden benötigen. Ihre Extrema sollten vom Typ `SIFT_Keypoint` sein.

Hinweis: Weitere Methoden des SIFT-Algorithmus `SIFT_algo.py` benötigen die zusätzlichen Variablen `deltas` und `sigmas`. Diese Variablen sollten während `create_scale_space` erstellt werden.