Deep Learning mit Vektordaten

Basis-Repo für das entwickelte Python-Skript.

Wissenschaftliche Hilfskraft:
B.Sc. Annika L. Walter

Ziel:

Erstellung einer Anwendung für die Qualitätsüberprüfung selektierter Gebäude hinsichtlich einer geeigneten Verwendbarkeit als Trainingsdaten im Rahmen der Dissertation von M.Sc. Herrn Martin Knura.

Funktionalität:

Um im Rahmen einer Qualitätsüberprüfung feststellen zu können, ob die selektierten Gebäude bezüglich Ihrer Erscheinung die geforderten Qualitätsmerkmale erfüllen und somit als Trainingsdaten für den entwickelten Deep-Learning Algorithmus verwendet werden können, muss das Python-Skript eine Vielzahl an Funktionen erfüllen können.

In diesem Zusammenhang wurde sich darauf verständigt, dass die Qualität eines jenen selektierten Gebäudes in Hinblick auf das entsprechende Referenzgebäude anhand einer numerischen Angabe zwischen 0 und 1 wiedergegeben werden soll. Während eine Übereinstimmung von 0 bedeutet, dass das selektierte Gebäude hinsichtlich seiner Form nicht denen im Referenzgebäude wiedergespiegelten Geometrien und somit auch nicht den Anforderungen einer ausreichenden Qualität entspricht, nimmt die Eignung des selektierten Gebäudes als Trainingsinstrument mit ansteigenden Werten zu. Infolgedessen spricht ein Ergebnis von 1 für eine vollkommende Übereinstimmung. Dies bedeutet, dass das selektierte Gebäude den geometrischen Anforderungen des Operators ohne Einschränkungen nachkommt.

Um die Qualität des selektierten Gebäudes unter zu Hilfenahme einer entsprechenden numerischen Angabe zwischen 0 und 1 ausdrücken zu können, wurden die im Folgenden aufgeführten Operationen implementiert.

1.
Die Koordinaten des Referenzgebäudes werden in den Ursprung (0/0) des jeweiligen Koordinatensystems verschoben.

```
perfect_char_array = Funktionen.perfekter_buchstabe_koordinatenliste_ursprung_0 (perfect_char_liste)
```

2.
Die Fläche des Referenzgebäudes wird berechnet.

```
perfect_char_flaeche = Funktionen.perfekter_buchstabe_flaeche (perfect_char_x, perfect_char_y)
```

3.
Der Datenrahmen, in dem die selektierten Gebäude unter anderem mit Ihren jeweiligen Geometrien verankert sind, wird um eine Spalte, in der die berechnete Qualität des selektierten Gebäudes vermerkt werden soll, erweitert. Die Anzahl der hinzuzufügenden Spalten richtet sich nach der Anzahl der berücksichtigten Referenzformen.

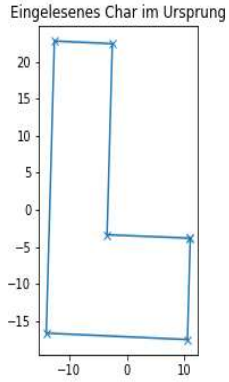
```
df["IOU_MAX_L"] = ""
```

4.
Die berechnete Qualität des selektierten Gebäudes wird in einer Liste abgespeichert. Die Anzahl der anzulegenden Listen richtet sich nach der Anzahl der berücksichtigten Referenzformen.

```
liste_hoechste_Uebereinstimmung_L = []
```

5.
Um die erforderlichen Operationen für alle selektierten Gebäude in Abhängigkeit von Ihrer Geometrie durchführen zu können, muss eine mit for eingeleitete Zählschleife implementiert werden.

5.1.
Nachdem eine Koordinatenliste des jeweiligen selektierten Gebäudes erstellt wurde, muss das entsprechende Gebäude in den Ursprung verschoben werden. Hierfür wird der Mittelwert der die Geometrie des selektierten Gebäudes definierenden Koordinaten berechnet und von jenen abgezogen. Für den Index I = 19, ist das erzielte Ergebnis im Nachfolgenden aufgeführt.



5.2.
Um das selektierte Gebäude so zu skalieren, dass die Fläche mit der Fläche des Referenzgebäudes übereinstimmt, muss die Fläche des in den Ursprung verschobenen selektierten Gebäudes berechnet werden.

```
pol2_rotiertes_char_0_flaeche_nicht_skaliert = Funktionen.perfekter_buchstabe_flaeche (pol2_rotiertes_char_0_x, pol2_rotiertes_char_0_y)
```

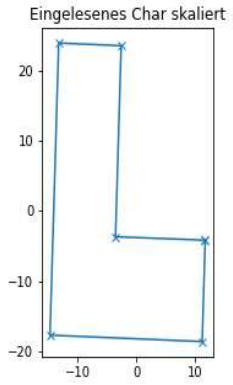
In Anschluss kann der anzubringende Skalierungsfaktor durch die nachstehende Formel berechnet werden:

$Skalierungsfaktor = \left({\frac{{Fläche Referenzgebäude}}{Fläche selektiertes Gebäude}} \right)$

Der berechnete Skalierungsfaktor wird gemäß

https://math.stackexchange.com/questions/1889423/calculating-the-scale-factor-to-resize-a-polygon-to-a-specific-size

dann sowohl in x-, als auch in y-Richtung angebracht. Für den Index I = 19, ist das erzielte Ergebnis im Nachfolgenden aufgeführt.



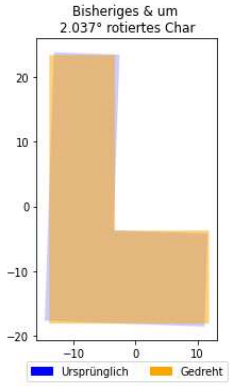
5.3
Um die höchstmögliche Überscheidung angeben zu können, muss sichergestellt werden, dass das Referenzgebäude und das selektierte Gebäude der gleichen Ausrichtung unterliegen. Hierfür wird gemäß

https://python.tutorialink.com/calculate-distance-between-points-in-polygon/

die längste Kante des selektierten Gebäudes detektiert und dieses entsprechend

https://stackoverflow.com/questions/61439009/rotate-alignment-an-image-by-line-along-the-y-axis

so rotiert, dass die längste Kante parallel zur y-Achse ausgerichtet wird. Unter zu Hilfenahme des berechneten Rotationswinkels wird dann eine Affintransformation durchgeführt. Um sicherzustellen, dass das selektierte Gebäude in die richtige Richtung rotiert wird, wird von dem rotierten selektierten Gebäude erneut die längste Kante und der dazugehörige Rotationswinkel bestimmt. Entspricht dieser Rotationswinkel 0°, wurde das selektierte Gebäude richtig rotiert und die in Zeile 143 des Python-Skriptes implementierte if-Abfrage wird ausgeführt. Für den Fall, dass der Rotationswinkel nicht 0° entspricht, sondern sich verdoppelt, wurde die Rotation in die falsche Richtung durchgeführt. In diesem Fall wird in den darauffolgenden Schritten nicht mit dem berechneten Rotationswinkel, sondern dem Winkel, der sich ergibt, wenn man den berechneten Rotationswinkel mit -1 multipliziert, verwendet. Auch wenn die nachfolgende Vorgehensweise identisch ist, muss dieser Fall durch eine else-Abfrage, die im Pyhon-Skript in Zeile 305 implementiert wurde, aufgefangen werden. Für den Index I = 19, für den der berechnete Rotationswinkel nicht mit -1 multipliziert werden musste, ist das erzielte Ergebnis im Nachfolgenden aufgeführt.

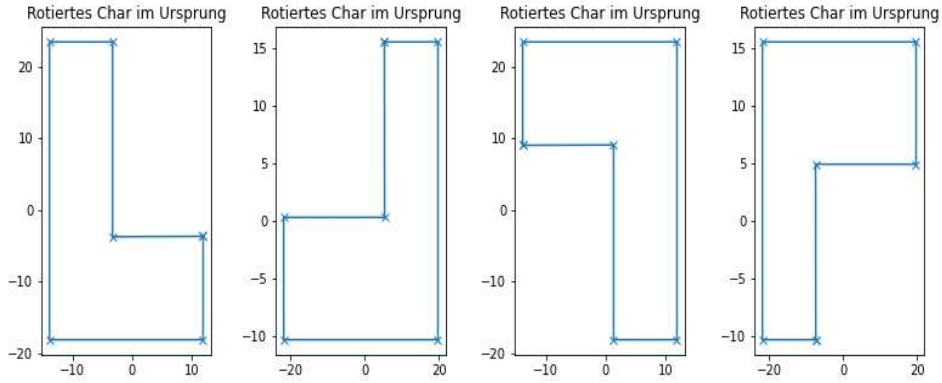


5.4

Neben der Ausrichtung der längsten Kante muss zusätzlich sichergestellt werden, dass das Referenzgebäude und das selektierte Gebäude auch hinsichtlich Ihrer totalen Ausrichtung identisch liegen. In diesem Zusammenhang kann es sein, dass die höchste Übereinstimmung zum Beispiel erst nach einer Rotation von 90°, 180° oder 270° erlangt wird. Folglich wird eine weitere Affintransformation, in der die genannten Winkel übergeben werden, durchgeführt.

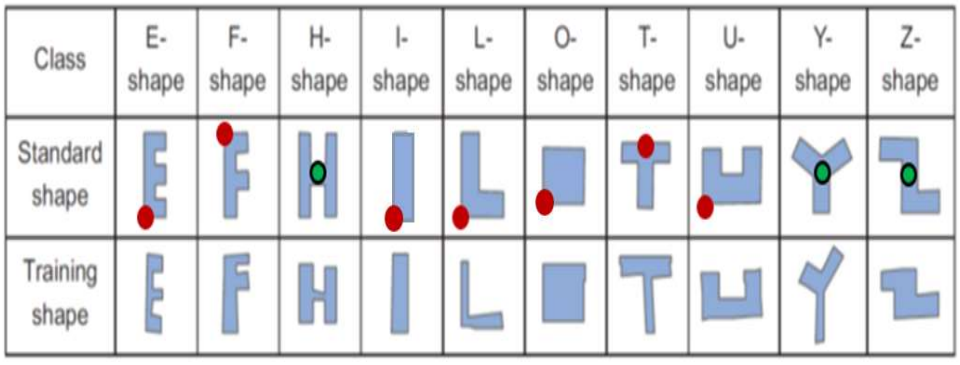
```
pol2_rotiertes_char_0_koordinaten_x = Funktionen.rotationen_0_90_180_270 (affintransformation_final_0, l)[0]
pol2_rotiertes_char_0_koordinaten_y = Funktionen.rotationen_0_90_180_270 (affintransformation_final_0, l)[1]
```

Für den Index I = 19, sind die erzielten Ergebnisse im Nachfolgenden aufgeführt.



5.5

Da das Referenzgebäude - im Gegensatz zum selektierten Gebäude - im Ursprung (0/0) verortet ist, muss eine weitere Translation vorgenommen werden. Hierfür werden, je nach Geometrie, verschiedene Ankerpunkte (siehe nachstehende Abbildung) selektiert.



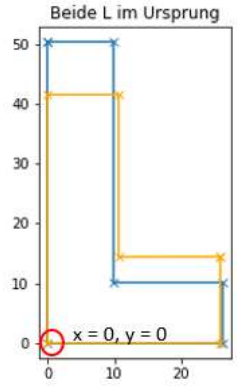
5.5.1

Wie man der Abbildung entnehmen kann, werden für die in diesem Python-Skript berücksichtigten Geometrien L, I, U und E jeweils dieselben Ankerpunkte verwendet. Folglich sind die entsprechenden Ankerpunkte in der unteren linken Ecke verortet. Um das selektierte Gebäude an dieser ausrichten zu können, wird auf die folgende Funktion zurückgegriffen.

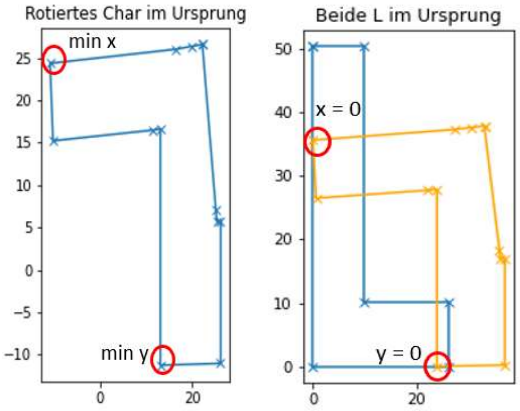
```
pol2_rotiertes_char_0_skaliert_ursprung_array = Funktionen.perfekter_buchstabe_koordinatenliste_ursprung_0
(pol2_rotiertes_char_0_koordinaten)
```

Folglich werden der Funktion die Koordinaten der jeweiligen vier Rotationsergebnisse übergeben. Von den entsprechenden Koordinaten wird dann der minimale x- und der minimale y-Wert extrahiert. Der minimale x-Wert wird von allen x-Koordinaten abgezogen und der minimale y-Wert wird von allen y-Koordinaten abgezogen. Folglich weißt

das daraus resultierende Polygon einen minimalen x-Wert von 0 und einen minimalen y-Wert von 0 auf. Eine Veranschaulichung für den bisher verwendeten Index = 19 ist im Nachfolgenden aufgeführt.

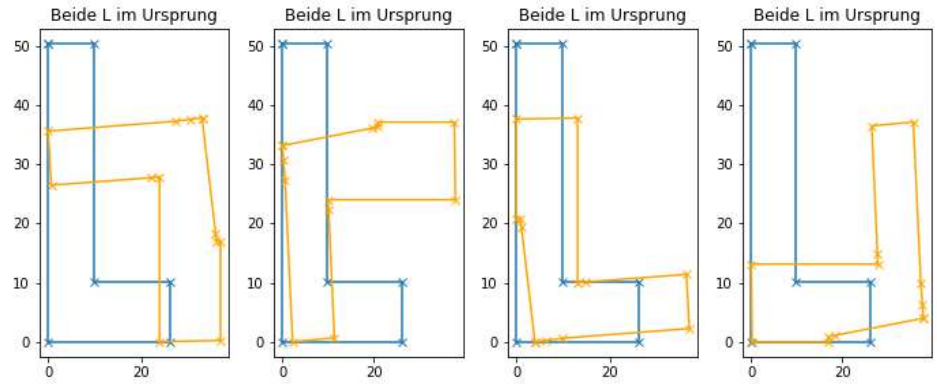


Das x = 0 und y = 0 nicht zwingend dem gleichen Punkt angehören müssen, wird in den nachstehenden Abbildungen für den Index = 17 veranschaulicht.



Wie man den beiden Abbildungen entnehmen kann, ist in diesem Beispiel die linke untere Ecke des selektierten Gebäudes nicht direkt im Ursprung verortet. Nichtsdestotrotz ist der selektierte Buchstabe immer so ausgerichtet, dass es einen Punkt mit x = 0 und y = 0 gibt.

Die auf dieser Translation für den Index = 17 dazugehörigen Rotationsergebnisse sind im Nachfolgenden aufgeführt.



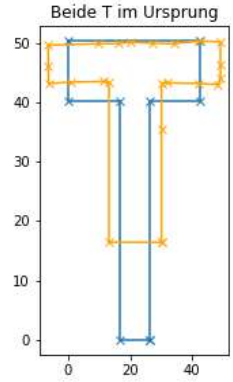
5.5.2
Für die selektierte Gebäude, die die Geometrie eines T aufweisen, liefert eine Überlagerung ausgehend von dem Mittelpunkt der oberen Geometriekante das beste Ergebnis.

```
pol2_rotiertes_char_0_skaliert_ursprung_array          =          Funktionen.perfekter_buchstabe_koordinatenliste_ursprung_oben_mitte
(perfect_chars['T']['geom'][:,0], perfect_chars['T']['geom'][:,1], pol2_rotiertes_char_0_koordinaten)
```

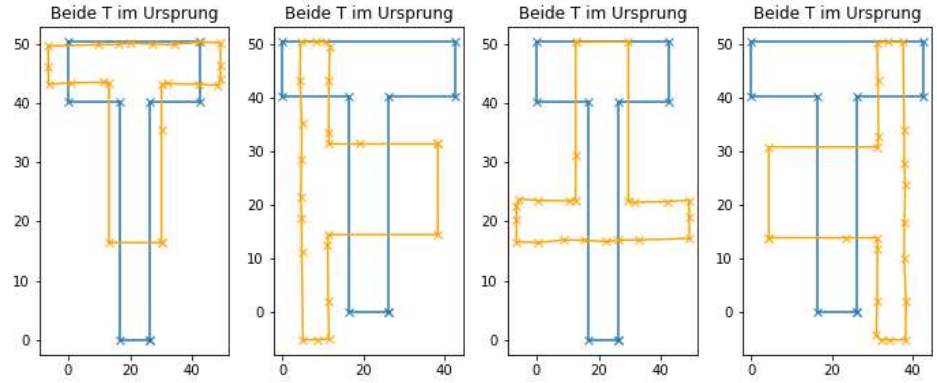
Um dieses Ergebnis zu realisieren, wird zunächst der gleiche Ansatz wie in 5.5.1 verfolgt. Hiervon ausgehend muss das selektierte Gebäude um eine Distanz x nach oben verschoben werden. Um diese Distanz berechnen zu können, wird die maximale y-Koordinate des selektierten Gebäudes von der maximalen Koordinate des Referenzgebäudes abgezogen. Daran anknüpfend wird das selektierte Gebäude so nach links verschoben, dass die Mittelpunkte der oberen Geometriekanten übereinanderliegen. Um dem nachzukommen wurden die Mittelpunkte der Geometriekanten des Referenzgebäudes und des selektierten Gebäudes berechnet. Für den Index = 197 beträgt dieser 21.30, bzw. 27.76. Demzufolge muss das selektierte Gebäude um

27.76 - 21.30 = 6.46

nach links verschoben werden. Das Ergebnis kann der nachstehenden Abbildung entnommen werden.



Die daraus resultierenden vier Rotationsergebnisse können den nachstehenden vier Abbildungen entnommen werden.

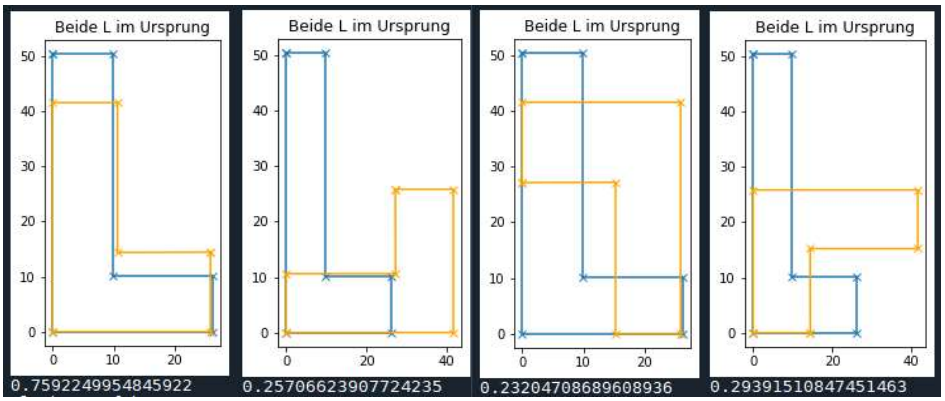


5.5.3

Für die selektierten Gebäude, die die Form eines F aufweisen, liefert eine Überlagerung ausgehend von der linken oberen Ecke das beste Ergebnis. Folglich wird das selektierte Gebäude nicht wie unter 5.5.2 beschrieben an den Mittelpunkten, sondern an den linken oberen Ecken, überlagert.

5.6

Die pro Geometrieeinheit erzielten vier Rotationsergebnisse werden nun verwendet um den jeweiligen - bereits erwähnten - numerischen Qualitätsfaktor zwischen 0 und 1 berechnen zu können. Hierfür wird sowohl die Kreuzung (Intersection) als auch die Vereinigung (Union) der beiden Flächen berechnet. Anhand einer Division (Kreuzung / Vereinigung), kann die sogenannte "Intersection over Union", die als Qualitätsmerkmal festgelegt wurde, berechnet werden. Wie hoch diese für die exemplarischen Ergebnisse des Index = 19 ausfallen, ist im Nachstehenden aufgeführt.



6.

Nachdem für jede der vier ausgeführten Rotationen die dazugehörige "Intersection over Union" berechnet wurde, muss das selektierte Gebäude, das so verschoben, skaliert und rotiert wurde, dass es die höchst mögliche Übereinstimmung zu dem Referenzgebäude erzielt hat, ausgewählt werden.

```
IOU_max_key =  max(IOU_0_dict.keys(), key=lambda x: IOU_0_dict[x]['IOU'])
IOU_max_value = (IOU_0_dict[IOU_max_key]['IOU'])
```

Der entsprechende Wert wird, je nach Referenzgeometrie, in die unter SChritt 3. beschriebene erweiterte Spalte eingefügt.

7.

Das nächste selektierte Gebäude wird automatisch durch die in Zeile 57 des Python-Skriptes implementierte und mit for eingeleitete Zählschleife ausgewählt. Im Folgenden werden die beschriebenen Schritte 5. bis 6. so lange durchgeführt, bis das Ende der Zählschleife erreicht ist.

Aktueller Stand:

Das Skript ist für eine Qualitätsüberprüfung selektierter Gebäude welche die Formen L, I, U, E, F und T aufweisen, geeignet. Nachdem die Referenzgeometrien in AutoCAD erzeugt werden, kann der die selektierten Gebäude enthaltene Datensatz im .geojson Format, welcher zwingendermaßen die Geometrieinformation mit dem Typ "Multipolygon" und den dazugehörigen Koordinaten enthalten muss, eingelesen werden. Als Beispieldatensatz wurde der zur Verfügung gestellte Datensatz "test_5000.geojson" importiert.

Voraussetzungen:

Um die Anwendung mit vollem Funktionsumfang aufrufen zu können, müssen die im Folgenden beschriebenen Voraussetzungen gegeben sein.

- Python 3 (getestet mit Python 3.8)
- Anaconda Prompt zur Installation der Python-Module (getestet mit anaconda3)
- AutoCAD oder vergleichbare Software zur Erstellung der Referenzgeometrien (getestet in AutoCAD 2022)
- Installation der Dependencies:

Dazu kann für das Projekt eine eigene virtuelle Umgebung angelegt werden. Es kann beispielsweise 'my-env' kreiert und aktiviert werden.

```
conda create -n my-env
conda activate my-env
```

Anschließend können die Dependencies installiert werden:

```
conda install numpy
conda install pandas
conda install geopandas
conda install -c conda-forge shapely
conda install -c conda-forge matplotlib
conda install -c conda-forge python-markdown-math
```

Ausführung:

Um die Anwendung auszuführen muss zunächst sichergestellt werden, dass die benötigten Referenzgeometrien und die Datei mit den selektierten Gebäuden vorhanden sind und, wenn möglich, im gleichen Ordner wie das Skript "220716_Skript_L_I_U_E_F_T_short.py" und das dazu benötigte Skript "Funktionen.py" liegen.

Über die "Run file (F5)" Applikation kann das Python Skript ausgeführt werden.