

Analyse von Pflanzenwachstum auf Basis von 3D-Punktwolken

Jakob Görner

HSNR - Master Arbeit - Vortrag

jakob.goerner@stud.hn.de

November 30, 2021

Inhalt

Ziele

Generierung von Punktwolken

Segmentierung

Registrierung

Server

Demo

Quellen

Ziele

- ▶ Analyse von Pflanzen mittels 3D-Punktwolken
 - ▶ Größe
 - ▶ Anzahl Blätter
 - ▶ Biomasse
 - ▶ ...
- ▶ Generierung von Punktwolken auf Basis von Bildern
- ▶ Kernprobleme
 - ▶ Entfernung des Hintergrundes
 - ▶ Segmentierung der Pflanze
 - ▶ Skalierung der Punktwolken ist unbekannt.
- ▶ REST-Interface zum einspielen von Datensätzen und ansteuern der Funktionalitäten.
- ▶ Datenübertragung zum Server sollte gering gehalten werden.

Generierung von Punktwolken

- ▶ Einsatz von spezieller Hardware sollte nicht nötig sein.
 - ▶ Hohe Anschaffungskosten
 - ▶ Bedienung ist nicht trivial
 - ▶ Daher Structure from Motion (SfM)
 - ▶ SfM ermöglicht die Generierung von Punktwolken aus einer Menge an Bildern.
- ▶ Da es viele bestehende Lösungen für SfM existieren, wird auf eine existierende Implementationen zurück gegriffen.
- ▶ Voraussetzungen an die Implementation
 - ▶ Möglichst wenig Bilder sollten reichen für gute Ergebnisse.
 - ▶ Performance der Implementation sollte möglichst gut sein.
 - ▶ Keine Information über Kamera Position und Ausrichtung.
 - ▶ Gegebenenfalls keine Information über die Reihenfolge der Bilder.

Generierung von Punktwolken

- ▶ Evaluation mehrerer Implementationen
 - ▶ Open Drone Map [1]
 - ▶ Colmap [2]
 - ▶ AliceVision (Meshroom) [4]
 - ▶ OpenMVG [6]
 - ▶ OpenCV SFM Pipeline
- ▶ Open Drone Map(ODM) und Colmap liefern gute Ergebnisse.
- ▶ Colmap liefert die besser Auflösung.
- ▶ ODM ist wesentlich performanter als Colmap (schnellere Berechnung bei weniger Ressourcen-Verbrauch).
- ▶ ODM ermittelt eine besser Abdeckung der Oberfläche.
- ▶ ODM liefert zusätzlich eine Schätzung der Normalen für jeden Punkt.

Generierung von Punktwolken



Figure: ODM



Figure: Colmap

Generierung von Punktwolken



Figure: Meshroom



Figure: OpenMVG

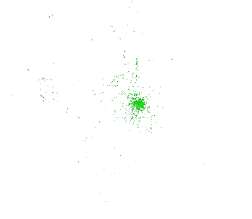


Figure: OpenCV

Segmentierung - Pflanze

- ▶ Ansatz 1: Entscheidung auf Basis der Krümmung eines Punktes
- ▶ Je höher die Krümmung eines Punktes ist desto wahrscheinlicher gehört dieser zu einem Stiel.
- ▶ Problem: Es muss eine gute Parametrisierung für alle Pflanzen-Arten gefunden werden.
- ▶ Problem: Blätter haben teilweise ähnlich Krümmung wie Stiele.
- ▶ Nachteil: Entfernung des Hintergrundes bleibt offen.
- ▶ Nachteil: Es liegt lediglich ein binärer Classifier vor. Es können nur die Stiele von dem Rest der Pflanze differenziert werden.
- ▶ Nachteil: Es müssen Normalen bekannt sein.

Segmentierung - Pflanze

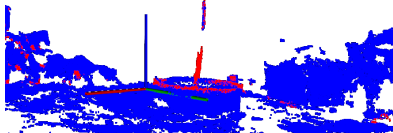


Figure: Avocado Ansatz 1

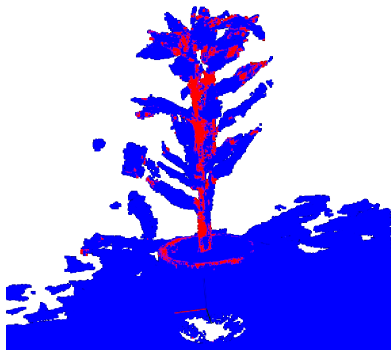


Figure: Zimmerpflanze Ansatz 1

Segmentierung - Pflanze

- ▶ Ansatz 2: Nutzung von Neuronalen Netzen (PointNet++)
- ▶ Erstellen eines Trainings-Datensatz aus 144 individuelle Punktwolken, mit bis zu 20 Subsamples je Punktwolke.
- ▶ Nutzung des Datensatzes in verschiedenen Trainings-Szenarien.
- ▶ Vorteil: Neben Blättern und Stielen können weitere Klassen segmentiert werden.
- ▶ Nachteil: Erstellen der Trainings-Daten sehr Zeit aufwendig + Daten müssen verfügbar sein.

Segmentierung - Pflanze

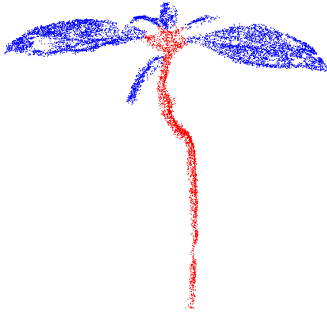


Figure: Avocado Ansatz 2

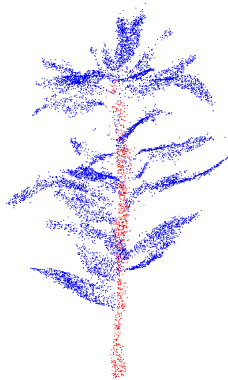


Figure: Zimmerpflanze Ansatz 2

Segmentierung - Hintergrund

- ▶ Auch hier wurde PointNet++ verwendet.
- ▶ Ergebnisse der Hintergrundsegmentierung sind durch den großen Anteil der Hintergrund-Punkte teilweise fehlerhaft.
- ▶ Pflanzen die ungewollt mit in der Szene enthalten sind, werden mit in die Analyse aufgenommen.
- ▶ Hier besser eine Szenen-Analyse durchführen und auf Bereichen die als Pflanze erkannt wurde Hintergrund-Segmentierung anwenden um Pflanze freizustellen.

Segmentierung - Hintergrund

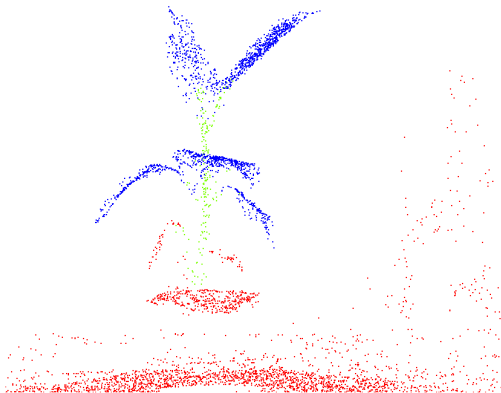


Figure: Ergebnis Hintergrundsegmentierung

Registrierung

- ▶ Problem: Da beim Erstellen der Punktwolke mit SfM der Maßstab nicht ermittelt werden kann, liegen verschiedene Punktwolken derselben Szene in unterschiedlichen Maßstäben vor.

$$\operatorname{argmin}_{R,t} \left(\sum_{i=1}^N \|Rp_{s_i} + t - p_{t_i}\|^2 \right) \quad (1)$$

- ▶ Lösung: Daher müssen Punktwolken zu unterschiedlichen Zeitpunkten miteinander registriert werden.
- ▶ Problem: Die meisten Registrierungsverfahren berücksichtigen nicht die Skalierung.
- ▶ Es wurden mehrere Ansätze untersucht dieses Problem zu lösen.
- ▶ Grundgedanke: Punktwolken mit einer Hintergrund-Punktwolke registrieren.

Registrierung - ICP mit Schätzung der Skalierung

- ▶ In der Bibliothek Point Cloud Library (PCL) wird eine Implementation die auch einen Wert für die Skalierung liefert bereit gestellt.
- ▶ Problem: ICP benötigt gute Initialisierung.
- ▶ Initialisierung finden:
 - ▶ Punktwolken an der XY-Ebene ausrichten.
 - ▶ Punktwolken auf die selbe Größe bringen.
 - ▶ Bereich um Zentrum entnehmen.
 - ▶ Punktwolken auf die selbe Größe bringen.
 - ▶ Störung herausfiltern
 - ▶ Registrierung mit SIFT-3D
- ▶ Nach Schätzung der Skalierung Nachverarbeitung.
- ▶ Problem: Ansatz funktioniert nur bedingt für einige Punktwolken.

Registrierung - DCP anpassen

- ▶ DCP ist ein Neuronales Netz, welches das Registrierungsproblem löst, aber keine Skalierung liefert.
- ▶ SVD-Head anpassen und Eingabe mit Einsen erweitern.
- ▶ Resultat des SVD-Head ist nun eine 4×4 Matrix.
- ▶ Annahme das diese Matrix als Transformations-Matrix interpretiert werden kann hat sich nicht bestätigt.
- ▶ Besser: Berechnung der Skalierung auf Basis der Rotation

Registrierung - Iterative Schätzung der Skalierung

- ▶ Iteratives durchlaufen verschiedener Skalierungen mit anschließender Registrierung.
- ▶ Für jede Iteration den Abstand der Punktwolke messen und die Iteration wählen, welche den kleinsten Abstand ergab.
- ▶ Einsatz von verschiedenen Registrierungsverfahren möglich. RPM-Net und ICP haben sich hier als robust erwiesen.
- ▶ Relativ gute Ergebnisse, aber auch hier kommt es immer wieder zu Ausreißern.

Registrierung - Iterative Schätzung der Skalierung

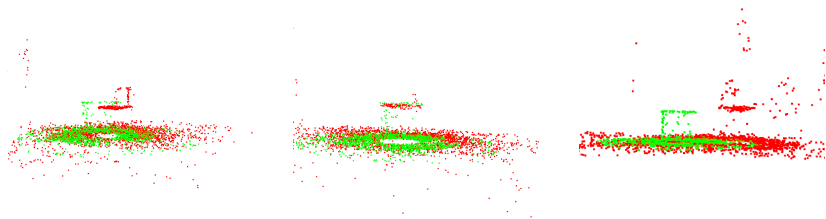


Figure: Registrierungsergebnisse für drei Zeitpunkte einer Pflanze

- ▶ Fünf Schnittstellen um Anwendung zu nutzen.
 - ▶ POST /detail/{Messreihe}/{Zeitstempel}
 - ▶ PUT /detail/{Messreihe}/{Zeitstempel}
 - ▶ GET /detail/{Messreihe}/{Zeitstempel}
 - ▶ GET /listing/{Messreihe}
 - ▶ GET /result/{Messreihe}/{Zeitstempel}
- ▶ Bearbeitung einzelner Jobs im Hintergrund.
- ▶ Zugriffe auf geteilte Ressourcen werden über Mutexe geschützt.
 - ▶ Job-Queue
 - ▶ Status
 - ▶ Result

Server - Pipelines und Jobs

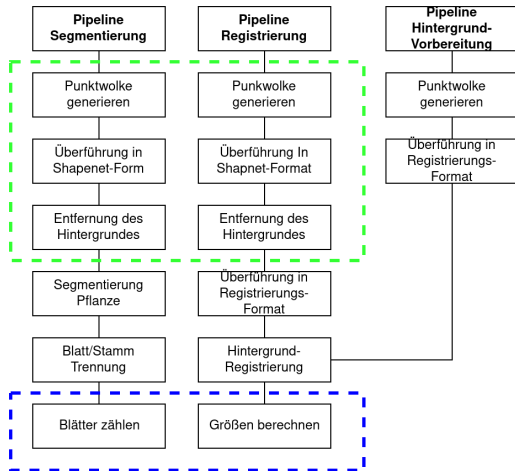


Figure: Übersicht über die einzelnen Pipelines und die darin enthaltenen Jobs.

Demo

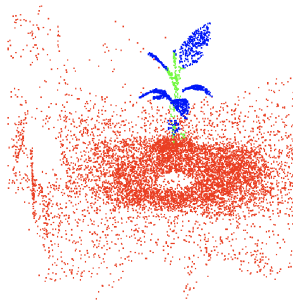


Figure: Hintergrund-Segmentierung

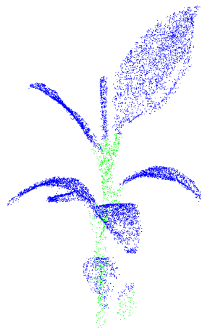


Figure: Pflanzen-Segmentierung






Figure: Blatt-Segmentierung

Referenzen I

-  pierotofy, “Open drone map - a command line toolkit to generate maps, point clouds, 3d models and dems from drone, balloon or kite images.,” 2020.
-  J. L. Schönberger and J.-M. Frahm, “Structure-from-motion revisited,” 2016.
-  J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, “Pixelwise View Selection for Unstructured Multi-View Stereo,” in *European Conference on Computer Vision (ECCV)*, 2016.
-  P. Moulon, P. Monasse, and R. Marlet, “Adaptive structure from motion with a contrario model estimation,” in *Proceedings of the Asian Computer Vision Conference (ACCV 2012)*, pp. 257–270, Springer Berlin Heidelberg, 2012.

Referenzen II

-  M. Jancosek and T. Pajdla, “Multi-view reconstruction preserving weakly-supported surfaces,” in *CVPR 2011*, IEEE, jun 2011.
-  P. Moulon, P. Monasse, R. Perrot, and R. Marlet, “Openmvg: Open multiple view geometry,” in *International Workshop on Reproducible Research in Pattern Recognition*, pp. 60–74, Springer, 2016.
-  alalek, “Structure from motion module,” 2016.