

# Analyse von Pflanzenwachstum auf Basis von 3D-Punktwolken

Jakob Görner

HSNR - Master Arbeit - Vortrag

*[jakob.goerner@stud.hn.de](mailto:jakob.goerner@stud.hn.de)*

December 5, 2021

# Inhalt

Ziele

Generierung von Punktwolken

Segmentierung

Registrierung

Server

Demo

Quellen

# Ziele

- ▶ Analyse von Pflanzen mittels 3D-Punktwolken
  - ▶ Größe
  - ▶ Volumen
  - ▶ Wachstum
  - ▶ Anzahl Blätter
  - ▶ ...
- ▶ Kernprobleme
  - ▶ Generierung von Punktwolken auf Basis von Bildern
  - ▶ Es muss eine Registrierung durchgeführt werden, da der Maßstab der Punktwolken unbekannt ist.
  - ▶ Segmentierung der Pflanze
  - ▶ Entfernung des Hintergrundes
- ▶ 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.
  - ▶ Möglichst wenig Rechenkapazitäten zur Berechnung der Punktwolken.
  - ▶ Keine Information über Kameraposition und Ausrichtung.
  - ▶ Gegebenenfalls keine Information über die Reihenfolge der Bilder.

# Generierung von Punktwolken

- ▶ Evaluation mehrerer Implementationen
  - ▶ Open Drone Map(ODM) [1]
  - ▶ Colmap [2]
  - ▶ AliceVision (Meshroom) [3]
  - ▶ OpenMVG [4]
  - ▶ OpenCV SfM Pipeline [5]
- ▶ ODM und Colmap liefern gute Ergebnisse.
- ▶ Colmap liefert die bessere Auflösung.
- ▶ ODM ist wesentlich performanter als Colmap (schnellere Berechnung bei weniger Ressourcen-Verbrauch).
- ▶ ODM ermittelt eine bessere Abdeckung der Oberfläche.
- ▶ ODM liefert zusätzlich eine Schätzung der Normalen für jeden Punkt.
- ▶ Colmap enthält manchmal Rauschen.

# Generierung von Punktwolken



Figure: ODM

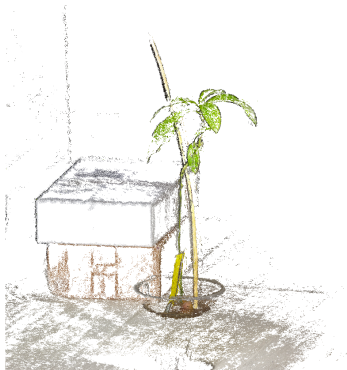


Figure: Colmap

# Generierung von Punktwolken



Figure: Meshroom



Figure: OpenMVG

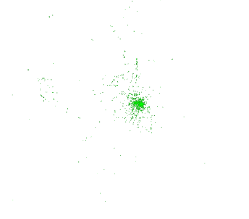


Figure: OpenCV

# 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.

$$f(p_i) = \begin{cases} 1 & k(p_i) \geq T_k \\ 0 & \text{sonst} \end{cases} \quad (1)$$

- ▶ Problem: Es muss eine gute Parametrisierung für alle Pflanzen-Arten gefunden werden.
- ▶ Problem: Blätter haben teilweise ähnliche 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

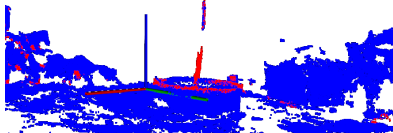


Figure: Avocado Ansatz 1

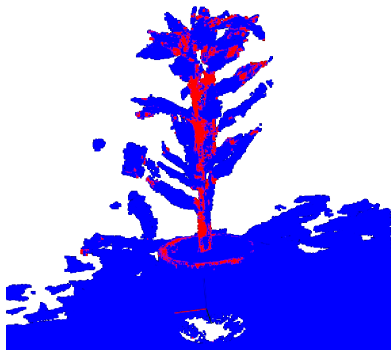


Figure: Zimmerpflanze Ansatz 1

# Segmentierung - Pflanze

- ▶ Ansatz 2: Nutzung von Neuronalen Netzen (PointNet++ [6])
- ▶ Erstellen eines Trainings-Datensatzes aus 144 individuellen Punktwolken, mit bis zu 20 Subsamples je Punktwolke.
- ▶ Nutzung des Datensatzes in verschiedenen Trainings-Szenarien.
  - ▶ Mit und ohne Hintergrund
  - ▶ Hintergrund mit und ohne Zentrum
  - ▶ Mit und ohne Normalen
  - ▶ Mit und ohne Normalisierung
  - ▶ Mit und ohne zufällige Rotationen
- ▶ Vorteil: Neben Stielen können weitere Klassen segmentiert werden.
- ▶ Nachteil: Erstellen der Trainings-Daten sehr Zeit aufwendig + Daten müssen verfügbar sein.

# Segmentierung - Pflanze

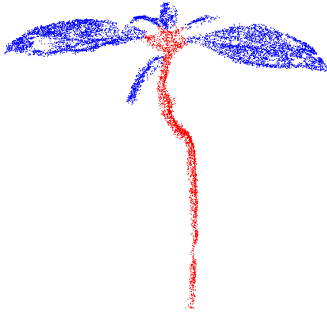


Figure: Avocado Ansatz 2

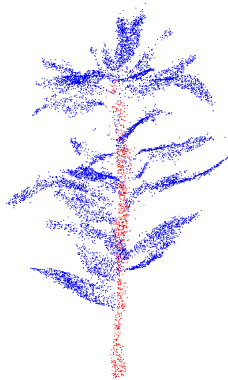
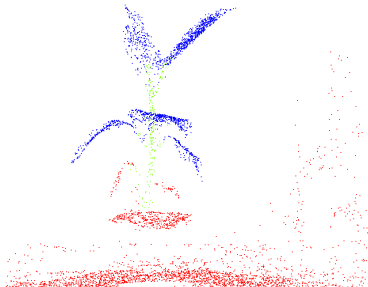


Figure: Zimmerpflanze Ansatz 2

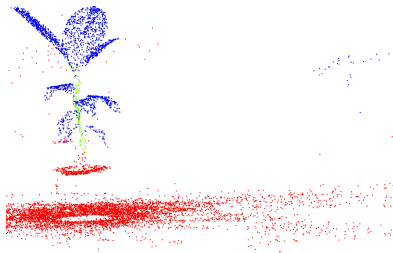
# Segmentierung - Hintergrund

- ▶ Auch hier wurde PointNet++ verwendet.
- ▶ Problem: Ergebnisse der Hintergrundsegmentierung sind durch den großen Anteil der Hintergrund-Punkte teilweise fehlerhaft.
- ▶ Lösung: Nur Bereich um das Zentrum der Punktwolke betrachten und Nachbearbeitung des Segmentierungs-Ergebnisses.
- ▶ Problem: Pflanzen, die ungewollt mit in der Szene enthalten sind, werden mit in die Analyse aufgenommen.
- ▶ Potentielle Lösung: Szenen-Analyse
  - ▶ Einzelne Pflanzen sollen in der Szene erkannt werden.
  - ▶ Isolieren der einzelnen Pflanzen
  - ▶ Wenn nötig Hintergrund-Segmentierung
  - ▶ Pflanze klassifizieren um Unkräuter auszublenden
  - ▶ Segmentierung einzelner Pflanzen

# Segmentierung - Hintergrund



**Figure:** Hintergrundsegmentierung  
Bananenpflanze



**Figure:** Avocado-Pflanze in Szene

# Segmentierung - Training 1

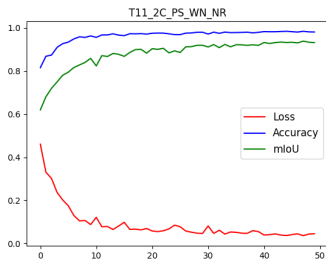


Figure: Ohne Hintergrund

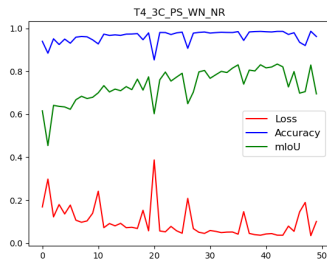


Figure: Mit Hintergrund

# Segmentierung - Training 2

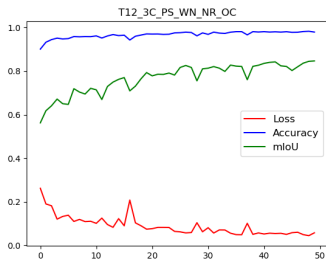


Figure: Hintergrund nur Zentrum

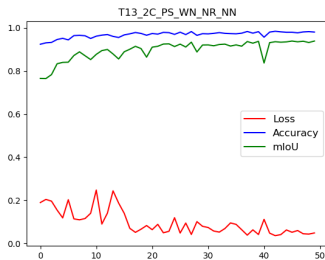


Figure: Ohne Normalen

# Segmentierung - Training 3

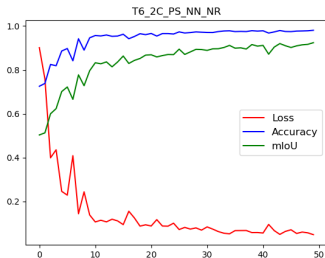


Figure: Ohne Normalisierung ohne Hintergrund

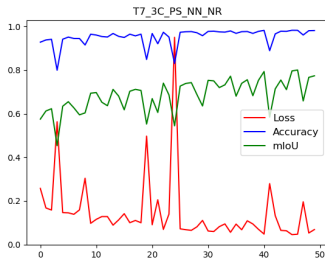


Figure: Ohne Normalisierung mit Hintergrund



# Segmentierung - Training 4

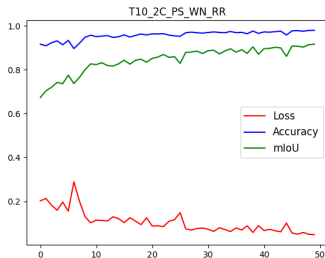


Figure: Mit zufälliger Rotation ohne Hintergrund

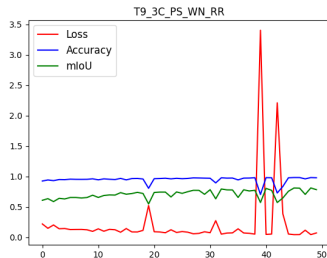


Figure: Mit zufälliger Rotation mit Hintergrund

# Segmentierung - Evaluation

	t11	t6	t10	t13	t4	t7	t12	t9
Loss	0,044	0,056	0,079	0,127	0,038	0,054	0,051	0,05
Genauigkeit	0,98	0,976	0,969	0,95	0,986	0,981	0,98	0,982
mIoU	0,925	0,908	0,893	0,827	0,813	0,788	0,841	0,811

**Table:** Evaluations-Ergebnisse der verschiedenen Modelle. Links ohne Hintergrund (t11, t6, t10, t13). Rechts mit Hintergrund (t9, t4, t7, t12).

# Registrierung

- ▶ Problem: Da beim Erstellen der Punktwolke mit SfM der Maßstab nicht ermittelt werden kann, liegen verschiedene Punktwolken derselben Szene in unterschiedlichen Maßstäben vor.
- ▶ Lösung: Punktwolken mit einer Hintergrund-Punktwolke registrieren, um alle Punktwolken einer Messreihe im selben Maßstab vorliegen zu haben.

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

- ▶ Problem: Die meisten Registrierungsverfahren berücksichtigen nicht die Skalierung.

$$\operatorname{argmin}_{R,t,s} \left( \sum_{i=1}^N \|R(sp_{s_i}) + t - p_{t_i}\|^2 \right) \quad (3)$$

- ▶ Es wurden mehrere Ansätze untersucht dieses Problem zu lösen.

# 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.
  - ▶ Bereich um Zentrum entnehmen.
  - ▶ Punktwolken auf dieselbe Größe bringen.
  - ▶ Störung herausfiltern.
  - ▶ Initiale Registrierung mit ausgewählten Punkten.
- ▶ Nach Schätzung der Skalierung folgt abschließende Registrierung.
- ▶ Problem: Ansatz funktioniert nur bedingt für einige Punktwolken.

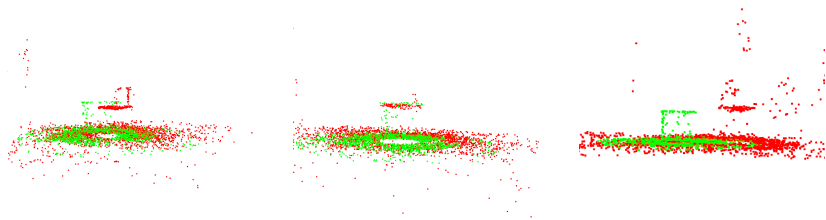
# Registrierung - DCP anpassen

- ▶ Deep Closest Points (DCP)[7] ist ein Neuronales Netz, welches das Registrierungsproblem löst, aber keine Schätzung der Skalierung liefert.
- ▶ SVD-Head anpassen und Eingabe mit Einsen erweitern.
- ▶ Resultat der SVD ist nun eine  $4 \times 4$  Matrix.
- ▶ Die Annahme, dass diese Matrix als Transformations-Matrix interpretiert werden kann, hat sich nicht bestätigt.
- ▶ Besser: Berechnung der Skalierung auf Basis der Rotation wie in [8].

# Registrierung - Iterative Schätzung der Skalierung

- ▶ Iteratives durchlaufen verschiedener Skalierungen mit anschließender Registrierung.
- ▶ Wahl der besten Iteration durch Messen des Abstands zwischen den Punktwolken oder Nutzung des Fehlermaßes der einzelnen Implementationen.
- ▶ Einsatz von verschiedenen Registrierungsverfahren möglich.
  - ▶ PointNetLK [9]
  - ▶ DCP [7]
  - ▶ RPM-Net [10]
  - ▶ ICP (open3d) [11]
  - ▶ RICP [12]
- ▶ 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

# Registrierung - Iterative Schätzung der Skalierung

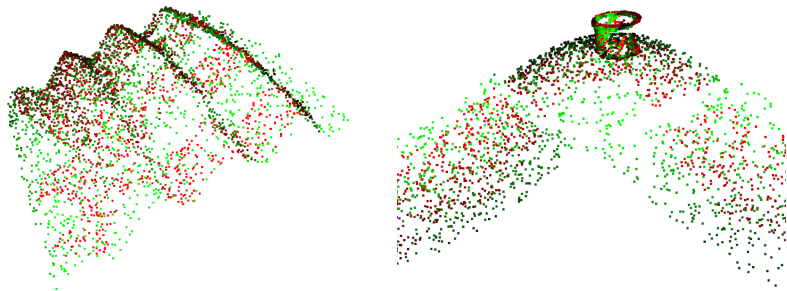


Figure: Registrierungsergebnisse für generierte Oberflächen



# Segmentierung - Evaluation

	DCP	RPM-Net	PointNetLK	ICP	RICP
Banane	0,0585	0,0129	0,1673	0,0117	0,0178
Avocado	0,0721	0,012	0,2235	0,0122	0,0245

**Table:** Evaluations-Ergebnisse der verschiedenen Registrierungsverfahren

# Server - Pipelines und Jobs

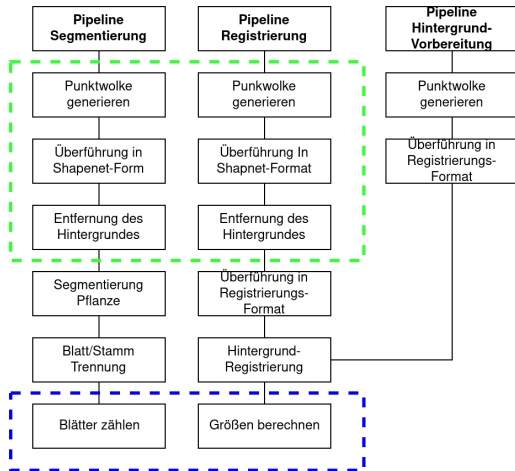


Figure: Übersicht über die einzelnen Pipelines und die darin enthaltenen Jobs.

# Server - Schnittstelle

- ▶ 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

# Demo

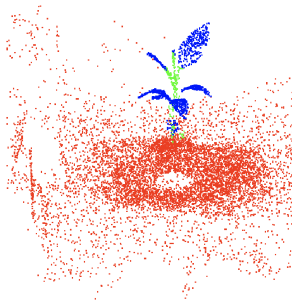


Figure: Hintergrund-Segmentierung

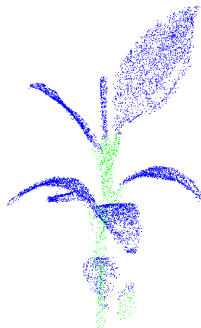







Figure: Pflanzen-Segmentierung







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.
-  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.
-  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.

# Referenzen II

-  C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *arXiv preprint arXiv:1706.02413*, 2017.
-  Y. Wang and J. M. Solomon, “Deep closest point: Learning representations for point cloud registration,” in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
-  T. Zinßer, J. Schmidt, and H. Niemann, “Point set registration with integrated scale estimation,” in *Point Set Registration with Integrated Scale Estimation*, 2005.
-  Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, “Pointnetlk: Robust and efficient point cloud registration using pointnet,” 2019.

# Referenzen III

-  Z. J. Yew and G. H. Lee, “Rpm-net: Robust point matching using learned features,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020.
-  Q.-Y. Zhou, J. Park, and V. Koltun, “Open3d: A modern library for 3d data processing,” *arXiv preprint arXiv:1801.09847*, 2018.
-  J. Zhang, Y. Yao, and B. Deng, “Fast and robust iterative closest point,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.