Geometrische Modellierung von Balkenstrukturen am Beispiel des Bautzner Dachstuhls

Torsten FRENZEL, Lorent MUCHA und Babett HÜBSCH

Kurzzusammenfassung

In dieser Arbeit wird die geometrische Modellierung einer komplexen Balkenkonstruktion beschrieben. Der dabei gewählte Ansatz zielt auf eine möglichst automatische und generische Bearbeitung des Problems mittels der Point-Cloud-Bibliothek (PCL) ab. Als Ergebnis berechnet der Algorithmus aus der Punktwolke ein geometrisches Datenmodell, welches zur weiteren Auswertung genutzt werden kann.

1 Einleitung

Eine wichtige Teilaufgabe in der automatischen Bildverarbeitung ist die Erkennung und Modellierung geometrischer Strukturen, wie sie an und in Bauwerken vorkommen. Anhand der gewonnen Geometrie lassen sich später weitere baustatische Informationen, wie z.B.: Schwerpunkt, gewinnen, oder die Daten können in dynamischen Simulationen genutzt werden.

Die Gliederung der Arbeit ist wie folgt: Nach der Beschreibung der Ausgangsdaten und des Aufnahmeobjektes, folgt die Darstellung des Workflows, welcher grob in 3 Schritte unterteilt werden kann: Datenaufbereitung, Zerlegung der Konstruktion in Balkenkomplexen und geometrische Rekonstruktion der einzelnen Balken. Nach der Darstellung und Bewertung der Ergebnisse erfolgt eine Beschreibung des Projektmanagements und ein abschließendes Fazit

2 Ausgangsdatensatz und Aufnahmeobjekt

Der zur Verfügung gestellte Datensatz von Laserpunktwolken zeigt den Dachstuhls des Bautzener Doms (Abb. 1). Die Aufnahme erfolgte im Rahmen einer Diplomarbeit von Frau A. Bienert, TUD (2004) mit einem *Riegel LMS-Z420i* Laserscanners. Die Punktwolke besteht aus 12 Aufnahmestandpunkten und ca. 34 Millionen Punkten. Die einzelnen Aufnahmen waren bereits Registriert und die Genauigkeit der einzelnen Standpunkte zueinander beträgt 5 -7 mm. Für die Segmentierung und Modellierung eines Stützbalkens diente die gesamte Punktwolke als Datengrundlage (SCHELLER & SCHNEIDER 2006).

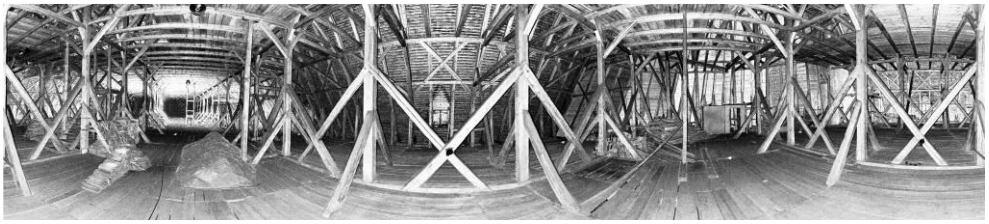

Abb. 1: Laserscanneraufnahme eines Standpunktes im Dachstuhl Bautzener Dom als zweidimensionales Intensitätsbild (Quelle: SCHELLER & SCHNEIDER 2006)

3 Workflow

Der gesamte Workflow gliedert sich in 3 Schritte: Zuerst wird die Punktwolke aufbereitet und in ein geeignetes Datenformat zu Weiterverarbeitung exportiert. Dann wird die Punktwolke verschnitten und unwichtige sowie störende Daten (Boden, Dach, Wände) werden entfernt. Hierbei erfolgt auch eine Zerlegung in einzelne Elemente, den Balkenkomplexen, die mehrere verbundene Balken enthalten. Für diese Balkenkomplexe wird in einem dritten Schritt die Geometrie extrahiert und modelliert.

3.1 Datenaufbereitung

Mittels der scannereigenen Software *RISCAN PRO*[1] erfolgte ausschließlich der Export der Messdaten, da die Registrierung der Aufnahmepositionen bereits vorhanden war. Als Exportformat wurde eine Textdatei gewählt, in der jede Zeile ein Koordinatentripel enthält. Für die Nutzung von *PCL* für das erstellte C++ Programm erfolgte für jede Aufnahmeposition eine Anpassung der Header-Information in ein PCL-Konformes Textformat *PCD* (POINTSCLOUD.ORG 2014). Diese Konvertierung wird für die Unterstützung einiger Erweiterungen der Punktwolkenverarbeitung in *PCL* benötigt.

Der Algorithmus führt die Punktwolken der einzelnen Scanneraufnahmen zu einer Punktwolke als Merge zusammen. Um die Koordinaten aufnehmen zu können, wird eine leere Punktwolke erzeugt, in der die 3D-Punkte hineingeschrieben werden. Der Arbeitsablauf ist dabei als Schleife konzipiert und öffnet die zuvor erstellten PCD-Dateien. Die Punkte des jeweiligen Scans werden mit *loadPCDFile* eingelesen und mit dem Additions-Operator an die zuvor erstellte leere Punktwolke angefügt. Nach dem Iterieren aller Scans wird das Ergebnis des Merges in die Ausgabedatei geschrieben. Dieser Teil erfolgt vollautomatisiert.

Nach Zusammenführen aller Scans in eine Punktwolke wird ein Downsampling durchgeführt, um die Anzahl der zu verarbeitenden Datenmenge zu verringern. Dazu wird der VoxelGrid-Filter genutzt (POINTCLOUDS.ORG 2014,1). Neben dem Input ist die Definition der Blattgröße *setLeafSize* erforderlich. Hier ist die Parameterwahl abhängig von der gegebenen Punktwolke. Der Algorithmus führt ohne interaktive Abfrage ein Downsampling für drei verschiedene Blattgrößen von 5 cm, 10 cm und 20 cm durch. Die entsprechenden Blattgrößen werden dabei in den Dateinamen codiert, um die verschiedenen Ergebnisse für den

[1] Riscan Pro Software: http://www.riegl.com/products/software-packages/riscan-pro/

weiteren Ablauf nutzen zu können. Die Wahl der Blattgröße bestimmt, wie stark die Punktmenge reduziert wird und damit die Lauf- und Rechenzeit sowie Genauigkeit der nachfolgenden Algorithmen. Eine große Blattgröße führt zu einer starken Reduktion der Punktmenge und Reduzierung der Rechenzeit. Für ein exaktes Ergebnis sollte die Punktwolke mit der kleinsten Blattgröße, bzw. die nicht reduzierte Punktwolke verwendet werden.

3.2 Segmentierung

3.2.1 Der Workflow

In der Abbildung 13 (Anhang) ist der Workflow als Gesamtheit aller Inputs und Outputs, mit den durchgeführten Algorithmen abgebildet. Dieser schließt sich an den ersten Schritt der Datenaufbereitung an und kann in die drei Schritte: Filterung → Segmentierung → Merge zusammengefasst werden. In der Abbildung stellen dabei die grünen Felder die Inputs und die rot gefärbten Kästen die angewendeten Algorithmen aus der PCL Library dar. In den folgenden Kapiteln soll auf die drei Schritte näher eingegangen werden. Dabei werden die verwendeten Algorithmen kurz vorgestellt und am Schluss das Ergebnis zusammengefasst.

3.2.2 Filterung des Bodens und des Dachs

Um im zweiten Schritt eine Extraktion der einzelnen Balken durchzuführen, mussten zu Beginn das Dach und der Boden entfernt werden. Für den Boden hat sich der morphologische Filter als geeignetste Algorithmus herausgestellt. Dieser Filter trennt Boden-(Gelände-)Punkte von den ”Nicht-Boden“-Punkten, indem ein Strukturelement erstellt wird, welches die Geländehöhenunterschiede in Abhängigkeit von der Entfernung beschreibt. Je größer die horizontale Entfernung, desto größer kann der Höhenunterschied zwischen den Bodenpunkten sein.

Für die Filterung wird dieses Strukturelement vertikal beginnend unterhalb der Punktmenge nach oben verschoben. Dazu wird es planimetrisch auf jedem Punkt zentriert und nach oben geschoben, bis es an einen Punkt innerhalb des Umkreises angepasst wird und die Verschiebung gestoppt wird. Der untersuchte Punkt ist kein Geländepunkt, wenn er nicht das Zentrum ist, auf den das Strukturelement zentriert wurde. Auf diese Weise wird die gesamte Punktwolke klassifiziert (SITHOLE & VOSSELMAN, 2003). Der Algorithmus ist in PCL als ProgressiveMorphologicalFilter (POINTCLOUDS.ORG, 2017) implementiert.

Da der Algorithmus sehr rechenintensiv ist, kam als Input nur die downgesampelte Punktwolke („merged.pcd“) mit der Blattgröße von 5cm aus dem vorangegangenen Datenaufbereitungsschritt, in Frage. Die Anwendung auf die originale Punktwolke wurde nach drei Tagen Rechenzeit abgebrochen. Um jedoch eine größtmögliche Punktdichte zu erreichen, wurde im Anschluss ein Merge mit der Originalen Punktwolke durchgeführt (Kapitel 3.2.4).

Die Entfernung des Daches aus der Punktwolke wurde mit einem PassThrough-Filter (POINTCLOUDS.ORG, 2013) realisiert, welcher Punkte entfernt die nicht in einem festgelegten Bereich liegen. Die Parameter des PassThrough-Filters konnten mit Hilfe des Pro-

gramms CloudCompare[2] bestimmt werden. Der gefilterte Bereich wurde für die z-Koordinate auf 0 m und 3 m gesetzt. Abbildung 2 dokumentiert das Ergebnis der Filterung von Decke und Boden.

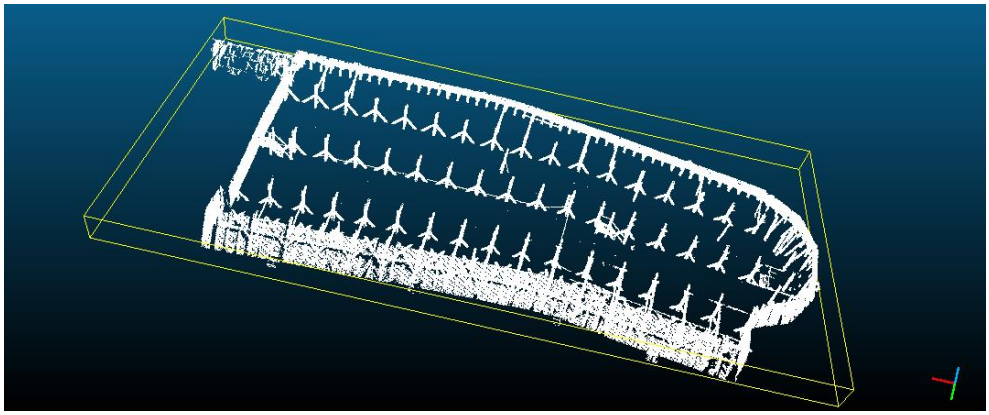

Abb. 2: Balkenwerk nach Filterung von Decke und Boden.

3.2.3 Clusterung und Segmentierung

Nach dem Filterprozess erfolgte die Extraktion der Balken durch eine Clusterung und Segmentierung. Das Ziel der Clusterung bestand darin, möglichst alle Balkengrundrisse zu bekommen, indem die *EuclideanClusterExtraction* (POINTCLOUD.ORG, 2003) auf eine vorher gefilterte „Balkkenstump"-Punktwolke angewandt wurde. Hierfür wurde erneut ein PassThrough Filter verwendet.

Als Ergebnis wurden alle irrelevanten und störenden Elemente bis auf die Seitenwände entfernt, was mit der Parametrisierung der z-Achse zwischen 1.8 m und 2 m erreicht werden konnte. Abbildung 3 stellt den Output des PassThrough Filters dar. Das Ergebnis wurde als PLY-Datei unter dem Namen „balkenstuempfe.ply" gespeichert und diente als Input für die Clusterung.

Für die Clusterung der Balken wurde das euklidische Clusterverfahren verwendet, welches ebenfalls in der PCL als *EuclideanClusterExtraction* (POINTCLOUD.ORG, 2003) zur Verfügung gestellt wird. Die geclusterten Balkenstümpfe, wurden in einer Punktwolke gespeichert. Um eine Segmentierung der Balken anhand der Clusterungsergebnisse durchzuführen, ist es essentiell den Mittelpunkt zu ermitteln, mit welchen im darauffolgenden Schritt der *MinCut* (POINTCLOUD.ORG, 2014,2) Segmentierungsalgorithmus angewandt werden konnte. Aus diesem Grund wurde in jedes Cluster ein Centroid eingepasst, welcher den jeweiligen Mittelpunkt des Clusters darstellt. Die Menge aller Centroid-Punkte, wurde in einer weiteren Punktwolke gespeichert (center.ply). Abbildung 4 zeigt diese Punktwolke mit den Mittelpunkten der Cluster.

[2] http://www.danielgm.net/cc/

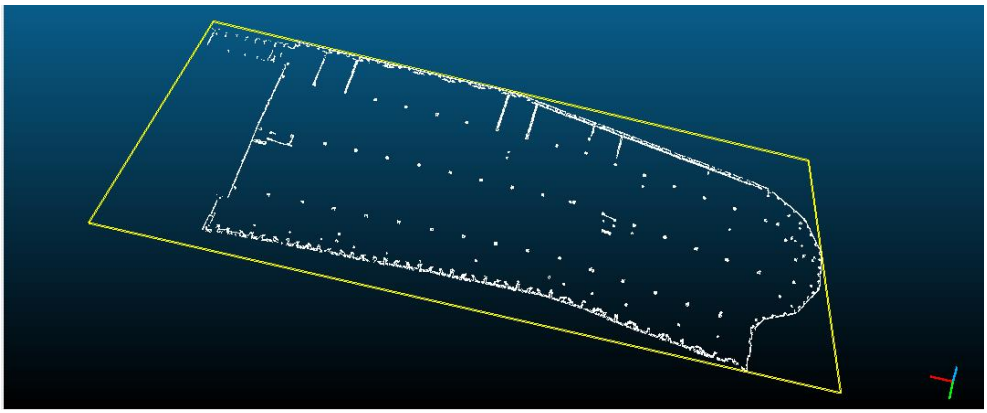

Abb. 3: Gefilterte Balkenstümpfe für das EuclideanClusterung

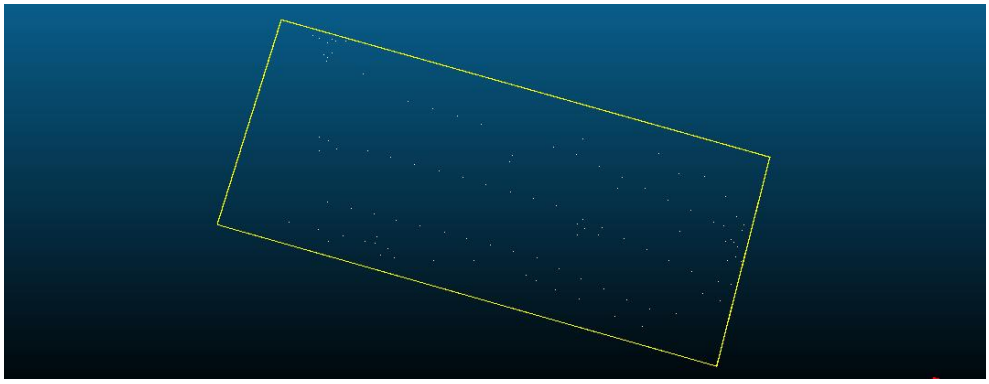

Abb. 4: Punktwolke der Centroide, welche anhand der Cluster gebildet wurden.

Die „Centroide"-Punktwolke enthielt jedoch auch Punkte für die Seitenwände, welche nicht das Balkenwerk widerspiegelten, das extrahiert werden sollte. Deshalb wurde erneut der *PassThrough* -Filter zur Entfernung dieser Punkte angewandt. Der gefilterte Bereich für die y-Koordinate lag zwischen -7 m und 14 m und konnte mit Hilfe von CloudCompare ermittelt werden.

Als nächstes erfolgt die Segmentierung der Balkenkomplexe, aus der downgesampleten Gesamt-Punktwolke mittels der *MinCut*-Segmentierung. Dieser Algorithmus führt eine binäre Zerlegung der Eingangsdaten anhand eines Mittelpunktes und eines Radius, in Vorder- und Hintergrundpunkte (Punkte, die zum Objekt gehören und welche, die nicht dazu gehören), durch.

Für die Segmentierung wurde die Centroid-Punktwolke iteriert und jeder iterierte Punkt als Mittelpunkt gesetzt, der Radius betrug 1,5m. Jeder so gefilterte Balkenkomplex wurde in eine separate Ausgabedatei gespeichert „balken_downsampelt*.ply". Als Erweiterung der Iteration diente ein Array, welches die Anzahl der extrahierten Balkenkomplexe speicherte. Dafür wurde jeder extrahierte Balken gezählt und der Wert im Array gespeichert. Der Maximalwert des Arrays stellte die Anzahl der gefilterten Balken dar.

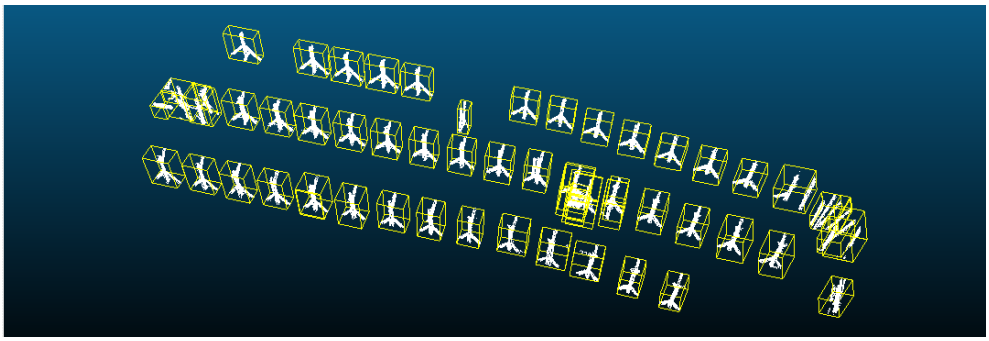

Abb. 5: Ergebnis der MinCut-Segmentierung mit den einzelnen Balkenkomplexe

3.2.4 Zusammenführen der Gedownsampelten Balkenkomplexe mit der Originalen Punktwolke

Die extrahierten Balkenkomplexe wiesen nur eine sehr geringe Punktdichte auf, was auf die Verwendung der downgesampelten Punktwolke zurückzuführen ist. Für ein besseres Ergebnis musste die originale Punktwolke einbezogen werden, welche die höchste Punktdichte besaß. Für diesen Schritt wurde über alle extrahierten Balkenkomplexe iteriert und die Bounding-Box berechnet. Darauffolgend wurde geprüft, welche Punkte aus der originalen Punktwolke innerhalb der Bounding-Box lagen. Alle Punkte innerhalb der Bounding-Box, welche zur orginalen Punktwolke gehörten, wurden in eine neue Punktwolke kopiert und in einer Ausgabedatei gespeichert („balken_orginal*.ply“). Die Anzahl der Punkte je Balkenkomplex konnte dadurch von durchschnittlich 700 auf 50.000 erhöht werden.

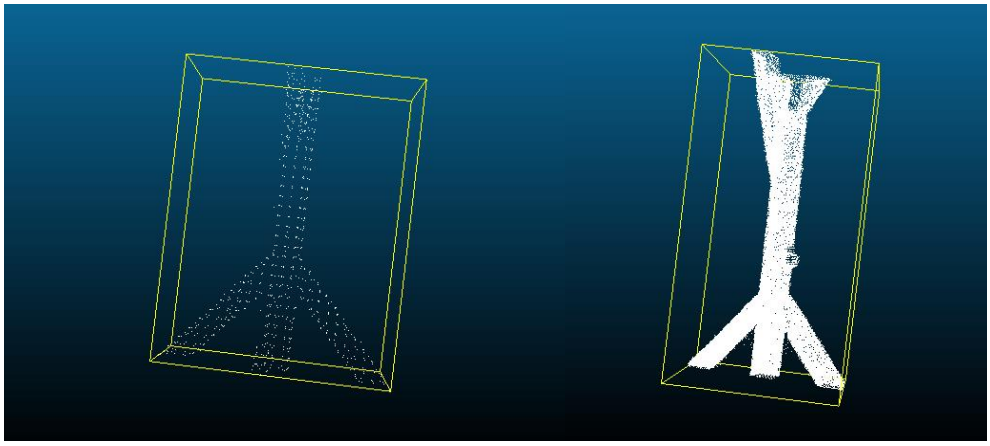

Abb. 6: Vergleich Extraktion eines Balkenkomplexes aus downgesampelten und originalen Punktwolke, mit unterschiedlichen Punktdichten

In PCL ist leider keine automatische Berrechnung der Bounding Box vorhanden, weshalb mittels der Methode pcl::getMinMax3D(POINTCLOUD.ORG, 2014) die Maximalwerte

für X,Y und Z ermittelt werden konnte. Anhand dieser Werte konnte die ZUsammenführung durchgeführt werden.

3.2.5 Ergebnis der Segmentierung

Mit der Segmentierung und Zusammenführung der originalen Punktwolke wurden 56 Balkenkomplexe automatisch extrahiert. Die originale Punktwolke besitzt jedoch nur 51 Balken, womit 5 falsch extrahiert wurden. In Abbildung 7 (links) sind die Problemfälle dokumentiert. Durch eine fehlerhafte Clusterung wurden einige Balkenkomplexe mehrfach Segmentiert (Fall 2 und 3). Hier liegt das Problem an zwei Leitern, welche als Balkenkomplexe geclustert wurden. In den Fällen 4 und 5 wurden keine Balkenkomplexe erkannt, weil diese innerhalb der MinCut -Segmentierung außerhalb der Schwellenwerte lagen und somit nicht segmentiert wurden. In diesem Fall müsste geschaut werden inwieweit es möglich wäre mit erneuten Iterationen über die Ergebnisse der Clusterung die Schwellenwerte automatisch anpassen zu können. Bei den verschiedenen Fällen in Punkt 1 handelt es sich um Wände, welche den Balken vorgelagert sind. Diese konnten durch den PassThrough Filter und der Setzung von Schwellenwerten bei der Euklidischen Clusterung nicht entfernt werden, da ihre Punktmenge ähnlich der Balkenkomplexe war. Diese Problematik bildet Abbildung 7 (rechts) ab. Dabei konnte im Laufe dieser Arbeit das Problem nicht behoben werden. Es wurde versucht die Clusterung durch einen kleineren Radius fehlerfreier zu gestalten, was aber zu falschen Ergebnissen bei den Balken führte.

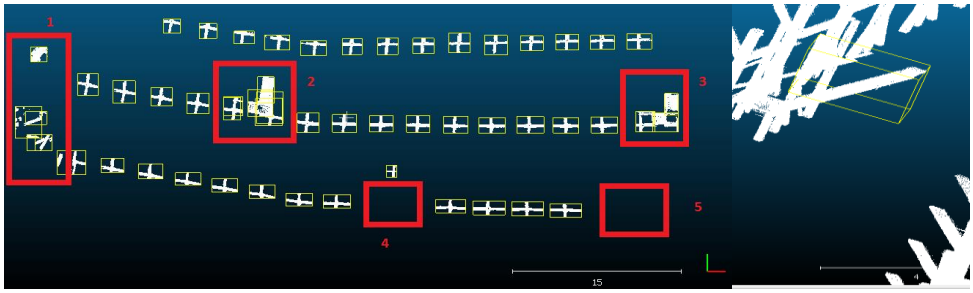

Abb. 7: Falsch extrahierte Balken mit den Fällen 1 bis 5 (links) und Leiter als Grund für falsche Extraktion (rechts).

3.3 Modellierung der Balken

Die geometrische Modellierung der Balken erfolgt in mehreren Phasen, die nacheinander sequentiell ausgeführt werden. Zur Modellierung erfolgt zuerst eine weitere Zerlegung der Punktwolke eines Balkenkomplexes in Cluster und danach eine sukzessive Extraktion der von Quader-Modellen mit 8 Eckpunkten in mehrehren aufeinander aufbauenden Ansätzen.

3.3.1 Vorbereitende Phase

Zuerst wird die Punktwolke in 32 horizontale Slices entlang der z-Koordinate zerschnitten. Für jeden Slice erfolgt ein Clusterung mittels euklidischer Distanzclusterung. Dadurch zerfällt der Slice in seperate Cluster, für welche die Bounding-Box bestimmt wird. In benachbarten Slices erfolgt die Bestimmung sich überlappender Cluster anhand der xy-

Koordinaten der Bounding-Boxen. Ist die Größe der Bounding-Boxen sehr ähnlich, so werden die beiden Cluster einem gemeinsamen Balkenabschnitt zugeordnet. Durch die Transitivität von Überlappungen entstehen größere Balkenschnitte aus einer größeren Anzahl von Clustern

3.3.2 Phase 1: Extraktion der Seiten eines Balkenabschnittes mittels RANSAC

In der ersten Phase wird ein modifizierter RANSAC Ansatz gewählt, der weiteres Ad-hoc Wissen nutzt um die 4 Seiten des Modells zu extrahieren. Mit zusätzlichen Bedingungen, soll sichergestellt werden, dass die extrahierten Ebenen den Begrenzungsseiten des Balkens entsprechen und nicht quer durch die Punktwolke schneiden.

Als erstes werden zufällig 3 Punkte ausgewählt und darauffolgend Zwangsbedingungen geprüft. Haben diese Punkte einen definierten Mindestabstand zueinander, wird die Anzahl der Inlier und linken bzw. rechten Ausreißer vom Ebenenmodell berechnet. Die Anzahl der linken oder rechten Ausreißer muss, dabei einen Schwellwert unterschreiten. Des Weiteren wird die minimale Distanz aller Inlier zum ihrem Mittelwert geprüft, die ebenfalls einen vorher definierten Schwellwert nicht unterschreiten darf.

Nachdem so zwei Ebenen extrahiert wurden, wird der Schnittwinkel der beiden Ebenen auf Rechtwinkligkeit geprüft. Falls der Winkel zu klein ist, wird die zweite Ebene verworfen und eine neue Ziehung beginnt.

Wenn mittels RANSAC erfolgreich zwei rechtwinklig zueinanderstehende Ebenen detektiert wurden, erfolgt die Rekonstruktion der 4 senkrechten Kanten des Quader-Modells. Dazu wird als erstes die Schnittgerade der beiden Ebenen errechnet, welches die erste Kante ist. Die fehlenden 3 Kanten des Quader-Modells ergeben sich durch das Verschieben der ersten Kante entlang der beiden Flächennormalen bis zum entferntesten Rand der Bounding-Box. Für die diagonal gegenüberliegende Kante muss entsprechend eine zweifache Verschiebung durchgeführt werden. Dabei ist zu Beachten, dass die Verschiebung nur entlang der x und y-Koordinate durchgeführt wird und die z-Koordinate unverändert bleibt. Zudem darf das augespannte Volumen des Quader-Modells das Volumen der Bounding-Box nicht um die Hälfte unterschreiten. Der Start- und Endpunkt der Kanten ergibt sich durch zwei weitere Schnitte mit den oberen und unteren horizontalen Begrenzungsflächen der Bounding-Box des Clusters.

Als Ergebnis erhält man die 8 Punkte quaderförmigen Modells, welches die Begrenzung des durch den Cluster verlaufenden Balken approximiert.

3.3.3 Phase 2: Erweiterung eines Balkenabschnittes

Als Ergebnis der vorhergehenden Phase gibt es vereinzelte Quader-Modelle in erfolgreich modellierten Clustern. Dabei sind innerhalb eines Balkenabschnittes noch Cluster vorhanden die noch nicht modelliert sind. Deshalb wird in einem weiteren Schritt eine näherungsweise Rekonstruktion dieser Cluster durchgeführt, indem die Quader-Modelle kopiert und entlang der Hauptrichtung des Balkens in diese unmodellierten Cluster verschoben werden.

3.3.4 Phase 3: Optimierung der Einpassung der Modelle

In einem Zwischenschritt erfolgt die Verbesserung der bisher generierten Quader-Modelle. Dazu erfolgt eine Zerlegung der Punktwolke des Clusters entsprechend des minimalen Abstandes zu den modellierten 4 Seitenflächen des Quaders. Mittels Hauptkomponentenanalyse werden aus der separierten Punktwolken die 3 Hauptrichtungen extrahiert. Die dritte Hauptrichtung bildet die neue Flächennormale der Seite und der Durchschnitt der Punktkoordinaten definiert den Fußpunkt. Die neuen Eckpunkte des Quaders werden durch Verschneidung der 4 Seitenflächen und den oberen und unteren horizontalen Begrenzungsflächen bestimmt.

In Abbildung 8 ist links sind die Bounding-Boxen der extrahierten Cluster in der initialen Phase, in der Mitte die Bounding-Boxen der extrahierten Balkenabschnitte in der initialen Phase und rechts das Ergebnis der extrahierten Modelle nach Ausführung der Phasen 1 bis 3 für den Test-Balken002 zu sehen.

3.3.5 Phase 4: Rekonstruktion unmodellierter Balkenabschnitte

Nach der Phase 2 gibt es Balkenabschnitte die bereits vollständig modelliert sind und Balkenabschnitte die noch unodelliert sind, weil in Phase 1 kein Cluster modelliert werden konnte. Für bisher unmodellierte Balkenabschnitte wird ein mittlerer Richtungsvektor *r* über alle Cluster im Balkenabschnitt berechnet. Dazu werden die Differenzen der Punkte der Bounding-Boxen der benachbarten Cluster summiert und gemittelt. Als Ergebnis ergibt sich ein Richtungsvektor *r*, dessen Richtung näherungsweise der Hauptrichtung des Balkens entspricht. Danach wird in jedem Cluster ein Quader-Modell eingepasst, dessen vertikale Kanten in Richtung des Richtungsvektors *r* verlaufen und dessen horizontale Kanten parallel zur x- und y-Achse verlaufen. Durch Verschneidung mit den Begrenzungsebenen der Bounding-Box lassen sich wiederum die 8 Eckpunkte des Modells extrahieren.

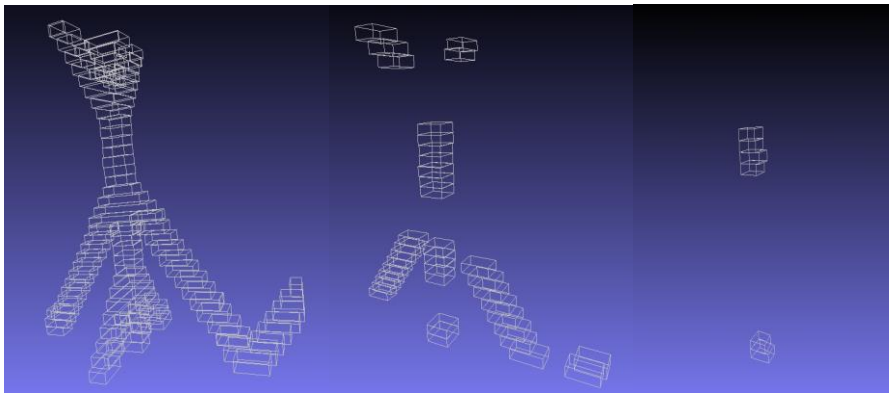

Abb. 8: Ergebnis nach dem Clustering und der Extraktion der Bounding-Boxen in der initialen Phase (links), nach der Extraktion der Balkenabschnitte in der initialen Phase (mitte) und nach der Phase 3 (rechts) eines Balkenkomplexes (Test-Balken002).

3.3.6 Phase 5: Zusammenfügung einzelner Balkenabschnitte

Nach der Phase 3 sind alle vorhandenen Balkenabschnitte vollständig modelliert. In der nächsten Phase werden Balkenabschnitte, die zu einem einzelnen Balken gehören zusammengefügt. Dazu wird ein mittlerer Verschiebevektor v über die einzelnen Quader-Modelle eines Balkenabschnittes berechnet. Dieser Vektor ergibt sich aus dem Durchschnitt der Differenzen zwei gleicher Eckpunkte in benachbarten Modellen. Haben zwei Balkenschnitte sehr ähnliche Verschiebevektoren und gibt es eine lückenlose Kette von überlappenden Cluster zwischen diesen Balkenabschnitten, so werden beide Balkenschnitte zu einem neuen Balkenabschnitt zusammengefügt, wobei die dazwischenliegenden Cluster integriert und durch das Kopieren der bereits existierenden Quader-Modelle modelliert werden.

3.3.7 Phase 6: Verschneidung von Balken

In einem letzten Schritt erfolgt das Einfügen von Quader-Modellen in Cluster in denen ein Balken, (für den ein Ziel-Quader-Modell generiert werden soll,) an einen anderen Balken (für den bereits ein Referenz-Quader-Modell in diesem Cluster existiert) angrenzt. Dazu wird an einem Ende des Ziel-Balkenabschnittes ein überlappendes Cluster ermittelt, zu dem es bereits ein sogenanntes Referenz-Quader-Modell in einem anderen Balkenabschnitt gibt. Zunächst wird ein initiales Ziel-Quader-Modell, durch Kopieren nach in Phase 2 vorgestellten Schema, generiert. Für die Verschneidung wird im Referenz-Quader-Modell zuerst heuristisch die Seite ermittelt, welche unmittelbar an den Ziel-Quader angrenzt. Dies erfolgt anhand der Ausrichtung und der Bestimmung des Abstandes zum Ziel-Quader-Modell. Maßgebend für das Ergebnis der Verschneidung sind die 4 vertikalen Seiten des Quaders. Dabei können prinzipiell 5 Fälle für die Lage der 4 Schnittpunkte unterschieden werden (Abb 9):

1. alle 4 Schnittpunkte liegen oberhalb der oberen horizontalen Begrenzungfläche: Der Ziel-Quader bleibt ungeschnitten
2. zwei Schnittpunkte liegen oberhalb und 2 Schnittpunkt liegen zwischen oberer und unterer horizontaler Begrenzungsfläche: Der Ziel-Quader ist nur teilweise angeschnitten. Zur Modellierung werden kleinere zwei Quader generiert. Der erste Quader ist ungeschnitten und der zweite Quader ist vollständig geschnitten
3. zwei Schnittpunkte liegen oberhalb der oberen und 2 Schnittpunkte liegen unterhalb der unteren Begrenzungsfläche: Der Ziel-Quader ist vollständig angeschnitten
4. zwei liegen zwischen maximaler und minimaler z-Koordinate und 2 Schnittpunkte liegen unterhalb der minimalen z-Koordinate: Der Quader fast vollständig abgeschnitten, es bleibt nur noch ein Stumpf
5. alle 4 Schnittpunkte liegen unterhalb der unteren horizontalen Begrenzungsfläche. Der Ziel-Quader ist vollständig abgeschnitten, d.h. das Ende der Balkenverbdingung ist erreicht

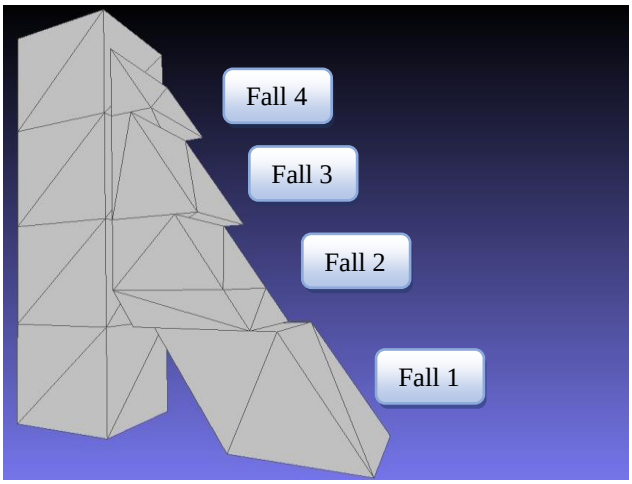

Abb. 9: Darstellung unterschiedlichen Fälle der Verschneidung zweier Balken

Je nach Fall müssen weitere Schnittpunkte horizontaler Seiten mit der Referenzfläche berechnet werden, um die Eckpunkte für den Ziel-Quader auszurechnen. Zusätzlich gibt es stets 4 weitere Unterfälle zu unterscheiden, die sich aus den 4 unterschiedlichen Richtungen ergeben, aus den der Ziel-Quader mit dem Referenz-Quader zusammenlaufen kann. Außerdem ist zu unterscheiden ob das Zusammenlaufen von unten nach oben oder von oben nach unten erfolgt. Der zweite Fall lässt sich durch geeignetes Vertauschen der z-Koordinaten der Eckpunkte auf den ersten Fall abbilden.

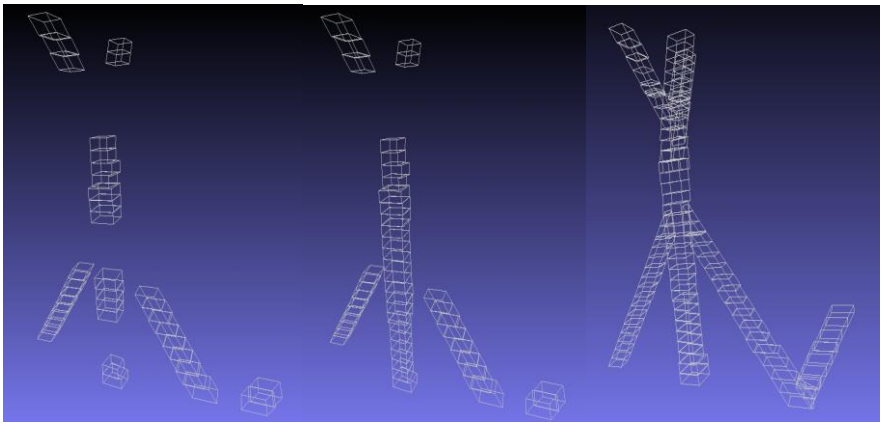

Abb. 10: Ergebnis nach der Phase 4 (links), der Phase 5 und das Endergebnis der geometrischen Rekonstruktion nach der Phase 6 (rechts) eines Balkenkomplexes (Test-Balken002).

In Abbildung 10 ist links das Ergebnis nach der Phase 4, in der Mitte das Ergebnis nach der Phase 5 und rechts das entsprechende Endergebnis nach der Verschneidung zu sehen, welches als PLY-Datei mit den Kanteninformationen exportiert ist.

4 Ergebnisse

Allgemein lässt sich für die Balkenmodellierung feststellen, dass 10 der 55 segmentierten Balkenkomplexe erfolgreich modelliert wurden.

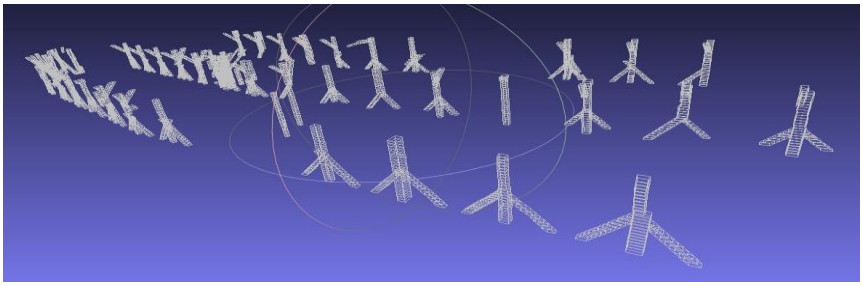

Abb. 11: Endergebnis der geometrischen Modellierung der extrahierten Balkenkomplexe

Abbildung 11 vermittelt einen qualitativen Eindruck von dem Ergebnis nach der Modellierung aller extrahierten Balkenkomplexe Die fehlerhafte Modellierung lässt auf folgende Probleme zurückführen:

1. Auswahl falscher Schnittkanten bei der Verschneidung
2. Störende Elemente (Querstreben, Seile, Leitern)
3. Falsche Flächeneinpassung der Quadermodelle
4. Ungünstiges Clustering
5. Zu kurze Balkenabschnitte

Die detaillierte Auswertung der Balkenmodellierung wird exemplarisch für 5 Balken durchgeführt, die erfolgreich modelliert werden konnten. Wobei die ersten 2 Balkenkomplexe zum Test des Algorithmus benutzt wurden, während die anderen 3 Balkenkomplexe eine Teilmenge der 55 Balkenkomplexe aus der automatischen Segmentierung sind.

Für die einzelnen Phasen der Modellierung lässt sich bestimmen, wie viele Quader-Modelle jeweils extrahiert werden konnten, wie in Tabelle 1 dargestellt.

Tabelle 1: Anzahl der rekonstruierten Quader-Modelle in den einzelnen Phasen der Modellierung

	Phase 1	**Phase 2**	**Phase 4**	**Phase 5**	**Phase 6**	**Cluster**
Test-Balken001	2	5	39	51	92	68
Test-Balken002	4	6	37	47	97	74
Balken 10	0	0	16	22	52	37
Balken 22	4	10	15	15	42	37
Balken 44	3	10	17	22	52	37

Dabei ist zu erkennen, dass in der Phase 1 mittels RANSAC nur sehr wenig Cluster erfolgreich modelliert werden können. Daraus lässt sich zum einen folgern, dass die Daten nicht sehr genau sind, als auch, dass zu wenig Daten vorhanden sind, um die Flächen korrekt zu detektieren. In der Phase zwei erfolgt ein Kopieren der bisher extrahierten Modelle auf die einzelnen Balkenabschnitte. In der Phase 4 werden bisher nicht modellierte Balkenabschnitte näherungsweise anhand der Hauptrichtung modelliert. Dadurch erhöht sich die Anzahl der gewonnenen Modelle signifikant. In der Phase 5 schließlich werden die Balkenabschnitte erweitert und zusammengefasst. In der letzten Phase bei der Verschneidung von Balken erhöht sich die Zahl der Quader-Modelle, wobei zu beachten ist, das einem Cluster mehrere Modelle zugeordnet werden können und somit die Zahl der Modelle die Zahl der Cluster übersteigt.

Für die durchgeführte Extraktion der Geometrie lassen sich auch Genauigkeitsbetrachtungen durchführen. Dadurch lässt sich abschätzen, wie gut die modellierten Flächen sich in die Daten einpassen, und man erhält damit ein quantitatives Maß für die Qualität der Modellierung.

Dazu wird für jedes Modell die zugehörige Cluster-Punktwolke betrachtet. Für jeden Punkt der Punktwolke wird die Distanz zu allen vier senkrechten Flächen des Quaders bestimmt und davon wiederum die minimale Distanz ausgewählt. Der Durchschnittswert der minimalen Distanzen aller Punkte bildet den Fehlerwert für das Modell. Diese Berechnung wird für alle modellierten Cluster durchgeführt, um den Gesamtfehlerwert zu erhalten.

Tabelle 2: Genauigkeiten vor und nach der Ebenen-Einpassung in Phase 3

	Mittlerer Abstand vor Einpasssung	**Mittlerer Abstand nach Einpassung**
Balken 1	1,87 mm	0,926 mm
Balken 2	1,69 mm	0,691 mm
Balken 10	1,64 mm	0,732 mm
Balken 27	1,82 mm	0,651 mm
Balken 44	1,69 mm	0,69 mm

Tabelle 2 zeigt die mittleren Abstände der fünf modellierte Balken vor und nach der Einpassung der Ebenen in Phase 3. Dabei lässt sich erkennen, dass die Fehlerwerte sämtlich im Millimeterbereich liegen. Da die Genauigkeit der Ausgangsdaten aufgrund fehlender Referenzmessungen nicht bekannt ist, lässt sich nicht bestimmen, ob der Fehler nur modellbedingt verursacht ist oder auch signifikante Messungenauigkeiten beinhaltet.

5 Projektmanagement

Aufgrund der unbekannten Qualität und der Größe der Daten war der Aufwand für die Bearbeitung schwer abzuschätzen. Durch die unterschiedlichen technischen Kenntnisse der Projektteilnehmer hat sich eine Zerlegung der Aufgaben in drei relativ unabhängige Pakete als zweckmäßig erwiesen. Dies ermöglicht sich eine seperate Bearbeitung der Algorithmen. Zum besseren Austausch der Ergebnisse wurde das Projekt in ein Repository in GitHub

hinterlegt, Änderungen konnten somit leichter verfolgt und Probleme jederzeit besprochen werden.

Ein Problem war, dass sich nur unzureichend Milestones für Zwischenergebnissen definieren ließen. Dies hat sich insbesondere darin gezeigt, dass zur Rekonstruktion nur bedingt Standardalgorithmen verwendet werden konnten.

6 Fazit (und ggf. Ausblick)

Der Ansatz der Arbeit verfolgte eine vollständig automatisierte Extraktion der Geometrie einer komplexen Balkenkonstruktion. Dabei zeigten sich aufgrund der Datenmenge und der Datenqualität verschiedene Probleme.

Eine erfolgreiche Extraktion der Seitenflächen durch robuste Schätzer ist auf der Grundlage der Daten nur eingeschränkt möglich, da die Kanten nur sehr schwach abgebildet sind. Dies führt dazu, dass die Flächen sehr oft nicht den tatsächlichen Seitenflächen der Balken entsprechen. Deshalb mussten für die Rekonstruktion der Geometrie neue Ansätze entwickelt werden.

Prinzipiell ist wenig zusätzliches Wissen nötig, um eine erfolgreiche Geometrie zu extrahieren. Allerdings gibt es in den verwendeten und implementierten Algorithmen viele Parameter, die konfiguriert werden müssen. Dies beginnt mit der Zerlegungsrichtung der Balken, über Wahl der Parameter für die Slice-Größe, der Parameter für das euklidische Clustering und den angepassten RANSAC-Algorithmus, bis hin zur Bestimmung von Schwellwerten für das Alignment der Balken.

Probleme ergeben sich, bei der Verschneidung zweier Balken an den Verbindungsstellen Dies wird vor allem dadurch sichtbar, dass die Verschneidung oft nicht korrekt durchgeführt wird, oder die modellierte Hauptrichtung eines Balkens von der tatsächlichen Hauptrichtung abweicht.

Die generische Anwendbarkeit der Lösung ist schwer abzuschätzen. Da jedoch außer der vertikalen Richtung keine explizites Ad-hoc Wissen verwendet wurde, sollte es prinzipiell auch möglich sein, durch Anpassung und Tuning der Parameter auch andere Balkenkonstruktionen zu modellieren.

Die drei Hauptkomponenten des Algorithmus sind aufgrund der unabhängigen Entwicklung, als getrennte Programme geschrieben worden. Jedes Programm läuft für sich ohne manuelle Eingriffe, so dass man sagen kann, die ausgehend von der extrahierten Punktwolke bis zu den modellierten Balken eine sehr hohe, wenn nicht sogar vollständige Automatisierung möglich ist.

Literatur

pointcloud.org. (2013,1). *pcl::EuclideanClusterExtraction< PointT > Class Template Reference*. Von Point Cloud Library (PCL): http://docs.pointclouds.org/1.7.1/classpcl_1_1_euclidean_cluster_extraction.html abgerufen

pointclouds.org. (2013,2). *pcl::PassThrough< PointT > Class Template Reference*. Von Point Cloud Library (PCL): http://docs.pointclouds.org/1.7.1/classpcl_1_1_pass_through.html abgerufen

pointscloud.org. (2014). *The PCD (Point Cloud Data) file format*. Von Documentation - Point Cloud Library (PCL): http://pointclouds.org/documentation/tutorials/pcd_file_format.php abgerufen

pointclouds.org. (2014,1). *Downsampling a PointCloud using a VoxelGrid filter*. Von Documentation - Point Cloud Library (PCL): http://pointclouds.org/documentation/tutorials/voxel_grid.php abgerufen

pointcloud.org. (2014,2). *Min-Cut Based Segmentation*. Von Documentation - Point Cloud Library (PCL): http://pointclouds.org/documentation/tutorials/min_cut_segmentation.php abgerufen

pointclouds.org. (2017). *pcl::ProgressiveMorphologicalFilter< PointT > Class Template Reference*. Von Point Cloud Library (PCL): http://docs.pointclouds.org/trunk/classpcl_1_1_progressive_morphological_filter.html abgerufen

Scheller, S., & Schneider, D. (2006). Extraktion von Primitiven aus Laserscannerpunktwolken zur Rekonstruktion von Tragwerken.

Sithole, G., & Vosselman, G. (2003). Comparison of filtering algorithms. *International Archives of Photogrammetry and Remote Sensing*, Vol. XXXIV, 3/W13.

Anhang

Workflow Abbildungen

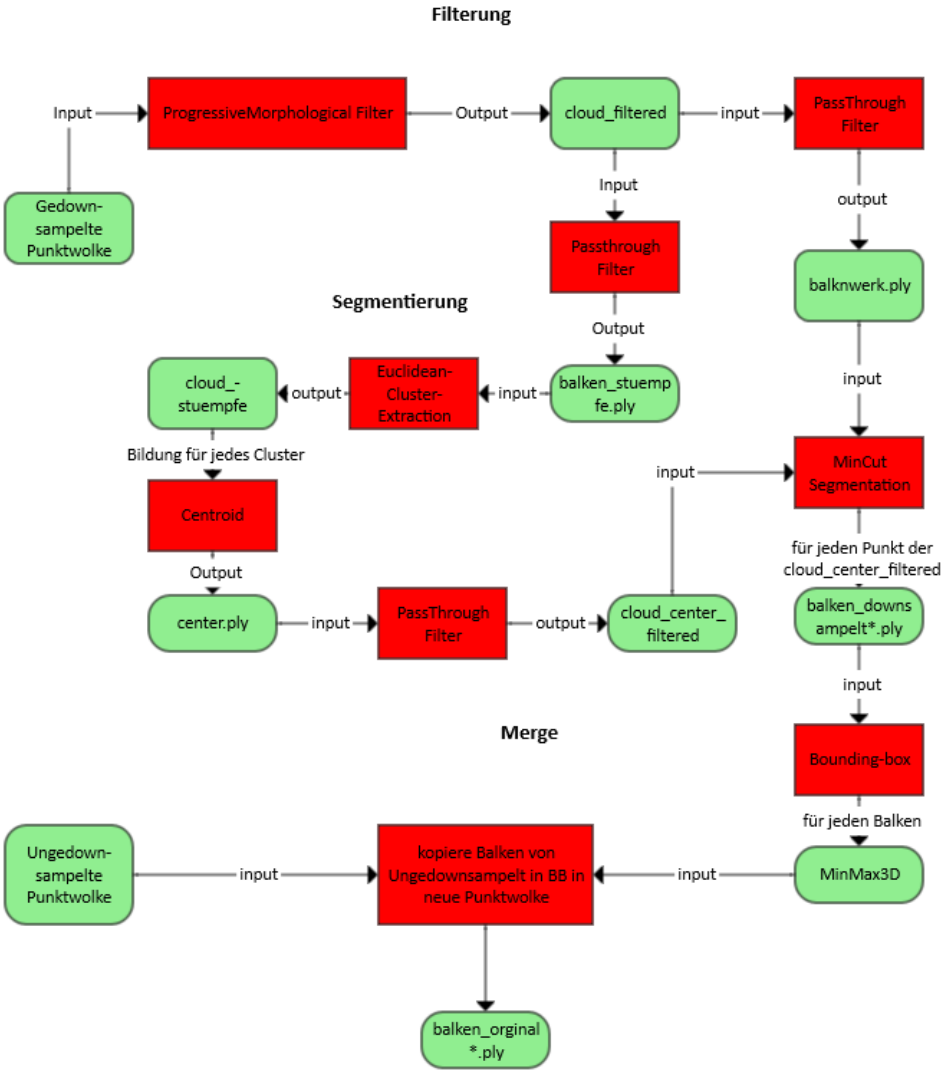

Abb. 12: Filterung, Segmentierung und Merge