mundialis GmbH & Co. KG

Sitz u. Registergericht: Bonn
Amtsgericht Bonn HRA 8528

Steuer-Nr.: 205/5823/1574
Ust.-ID-Nr.: DE300200756

K o m p l e m e n t ä r i n :
mundialis Verwaltungsgesellschaft mbH

vertreten durch:
M. Eichhorn

Dokumentation

vom 06.10.2023

Gebäude- und Dachbegrünungsdetektion

Extraktion von Gebäuden und begrünten Dächern auf Basis Digitaler Orthophotos (DOPs), Punktwolken und einer Flächennutzungskartierung (FNK)

3.0.2

erarbeitet für:

Regionalverband Ruhr
Referat Geoinformation und Raumbeobachtung
Team Geodaten, Stadtplanwerk, Luftbilder

Kronprinzenstraße 35
45128 Essen

E-Mail: geodaten@rvr.ruhr

erarbeitet von:

m u n d i a l i s GmbH & Co. KG
Kölnstraße 99
53111 Bonn

Telefon: +49 228 - 387 580 80
Telefax: +49 228 - 962 899 57

E-Mail: info@mundialis.de
Internet: www.mundialis.de

Ansprechpersonen:
G. Riembauer riembauer@mundialis.de
J. Haas haas@mundialis.de

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Änderungen:

Datum	Autor:in	Beschreibung
16.12.2022	mundialis	Hinzufügen der Dokumentation zur Erkennung von Dachbegrünung und zur Verwendung eines nutzerdefinierten DGMs, Anpassung zur Nutzung von GRASS GIS 8
03.01.2023	mundialis	Hinzufügen der Dokumentation zur Berechnung von Qualitätsmaßen
07.02.2023	mundialis	Überarbeitung der Dokumentation zu Version 3.0.0
10.02.2023	mundialis	Ergänzung der Kapitel *Gültigkeitsbereich der Analysetools* und *Umgang mit GRASS Warnungen und Fehlern*
14.02.2023	mundialis	Finales Review
19.07.2023	mundialis	Einarbeitung Feedback
06.10.2023	RVR	Kleinere Ergänzungen

Tabelle 1: Änderungshistorie des Dokumentes

Diese Dokumentation erläutert die notwendigen Arbeitsschritte zur Extraktion von Gebäuden auf Basis Digitaler Orthophotos (DOPs), Punktwolken, eines digitalen Geländemodells (DGM) und einer Flächennutzungskartierung (FNK) sowie die Detektion von Dachbegrünung auf Basis ebenjener Daten, eines Gebäudedatensatzes und optional eines Baumdatensatzes.

1 Vorbereitung

1.1 Testumgebung

Die Software zur vorliegenden Dokumentation wurde auf einem System mit 16 GB RAM, 16 CPUs und dem GRASS 8.3 Alpine Docker Image getestet.

1.2 Skripte und GRASS-Addons herunterladen

Die notwendigen Skripte und GRASS-Addons können aus diesem GitHub Repository heruntergeladen werden: https://github.com/mundialis/rvr_interface.

Durch Klick auf den grünen „Code“-Button oben rechts kann das Repository entweder für die Nutzung mit der Git-Software geklont werden:

```
# Im Zielverzeichnis ausführen:
$ git clone https://github.com/mundialis/rvr_interface.git
# bzw. wenn ein SSH-Key auf GitHub hinterlegt wurde:
$ git clone git@github.com:mundialis/rvr_interface.git
```

oder als ZIP-Datei heruntergeladen und auf dem Rechner entpackt werden. Das Klonen des Repositorys hat den Vorteil, dass es einfach aktualisiert werden kann, während beim Weg über die ZIP-Datei immer das ganze Repository neu herunterzuladen ist.

Die Version 1.0.0 beinhaltet Skripte und GRASS-Addons zur Gebäudeextraktion, während Version 2.0.0 zusätzlich ein GRASS-Addon zur Erkennung von Dachbegrünung enthält. Ab Version 2.0.0 ist auch die Möglichkeit implementiert, ein selbstdefiniertes DGM zur nDOM-Erstellung zu verwenden.

Ab der Version 3.0.0 ist das neue Modul *m.import.rvr* für Import und Vorprozessierung der jeweils benötigten Daten enthalten. Die bisherigen Addons *r.extract.buildings*, *v.cd.areas* und *r.extract.greenroofs* wurden zu einem Multi-Addon *m.analyse.buildings* zusammengefasst. Dies erleichtert die Installation und bündelt die zusammengehörigen Module. In den genannten Addons wurde ebenfalls ein Tiling eingebaut, sodass ein Großteil der Berechnung nun (mittels der Worker-Addons) über mehrere Kacheln erfolgt.

1.3 Benötigte Software

Gebäudeextraktion und Detektion der Dachbegrünung erfolgen größtenteils in **GRASS GIS**.

GRASS GIS, oder kurz GRASS (Geographic Resources Analysis Support System), ist ein kostenloses Open-Source Geographisches Informationssystem (GIS). Mit seinem großen Funktionsumfang wird es für die Verwaltung und Analyse von Geodaten, die Bildverarbeitung, die Erstellung von Grafiken und Karten, die räumliche Modellierung und Visualisierung verwendet. GRASS wird derzeit

in akademischen und kommerziellen Einrichtungen auf der ganzen Welt sowie von vielen Regierungsbehörden und Umweltberatungsunternehmen eingesetzt.

1.3.1 GRASS GIS-Basics

Zur Nutzung von GRASS sind einige grundlegende Kenntnisse über die Software nötig.

GRASS unterscheidet sich von anderen GIS-Anwendungen wie ESRI ArcGIS oder QGIS, in denen die Nutzenden sofort verschiedene Daten aus verschiedenen Datenquellen in verschiedenen Projektionen laden und mit der Arbeit an einem Projekt beginnen können. Beim Starten von GRASS muss zunächst das Arbeitsprojekt definiert werden, in dem die GRASS-Sitzung arbeiten soll.

Beim Start von GRASS GIS sind drei Voraussetzungen erforderlich (über den Startbildschirm oder die Kommandozeile einrichtbar):

- **Datenbankverzeichnis:** Ein Verzeichnis auf der lokalen oder Netzwerkplatte, das alle Daten enthält, auf die GRASS zugreift. Dies ist üblicherweise ein Verzeichnis namens „*grassdata*“, das sich im individuellen Home-Verzeichnis befindet.
- ***Location*:** Spielt die Rolle eines "Projekts". Alle Geodaten, die innerhalb einer *Location* gespeichert sind, müssen das gleiche räumliche Koordinatensystem haben (GRASS unterstützt aus verschiedenen Gründen keine *on-the-fly*-Projektion, kann aber beim Import reprojizieren).
- ***Mapset*:** Enthält aufgabenbezogene Daten innerhalb eines Projekts. Dies unterstützt, die Daten in logischen Gruppen zu organisieren oder eine parallele Arbeit mehrerer Nutzenden am selben Projekt zu trennen.

Ein weiterer Unterschied zu anderen GIS-Anwendungen ist das Konzept der ***Computational Region*.** Diese legt für Berechnungen und Analysen von Rasterdaten den räumlichen Bereich fest, für den die Berechnung durchgeführt wird. Das heißt, mit der *Region* wird die Ausdehnung und Auflösung des Ergebnisrasters bestimmt. Die aktive *Region* kann mit dem GRASS-Befehl *g.region* festgelegt werden.

Zum weiteren Verständnis der GRASS Basics kann zum Beispiel dieses Intro oder diese GRASS Manual-Seite konsultiert werden.

Weitere hilfreiche Links zum Umgang mit GRASS sind:

- GRASS GIS Homepage
- GRASS GIS 8.3 Reference Manual
- GRASS User Wiki Installation Guide
- GRASS GIS Wiki
- GRASS GIS Tutorial

1.3.2 Einrichten und Starten von GRASS in Docker und Linux

Im ersten Schritt muss zunächst GRASS GIS eingerichtet werden. Alle Daten, die in GRASS verwendet werden, liegen in einer Datenbank (*grassdb*) in GRASS-spezifischen Formaten. Diese Datenbank kann als normaler Ordner neu erstellt werden. Im gewünschten Verzeichnis, in dem die GRASS-Datenbank angelegt werden soll (Achtung: diese kann im Laufe der Analyse sehr groß werden!), soll ein neuer Ordner erstellt werden (üblicherweise als *„grassdata“* im Home-Verzeichnis):

```
$ mkdir grassdata
```

Die Erstellung der Datenbank ist nur einmalig zu Beginn notwendig. Besteht bereits eine *grassdb*, kann auch diese verwendet werden.

Zur Nutzung der Gebäudemodule wird GRASS in Docker verwendet. Hierfür muss Docker auf dem System installiert sein. Es wird das offizielle und öffentlich zur Verfügung stehende Docker Image von GRASS 8.3 verwendet. Es enthält alle für die Gebäudemodule notwendigen Abhängigkeiten wie Python3, GDAL und PDAL. Zunächst muss ein lokales Docker Image gebaut werden. Dies geschieht im Wurzelverzeichnis des RVR-Interfaces, in dem auch ein Dockerfile liegt. Dieser Schritt muss nur einmal ausgeführt werden:

```
# Navigieren zum Wurzelverzeichnis des RVR-Interfaces:
$ cd /pfad/zum/interface

# Aufbauen des Lokalen Docker Image
# Der Punkt am Ende bedeutet, dass Docker nach dem Dockerfile im aktuellen
# Verzeichnis suchen soll.
$ docker build -t rvr_interface:latest .
```

Statt *latest* kann auch das Datum zum Zeitpunkt der Erstellung (*230922* o. ä.) angegeben werden.

Auf Basis dieses Images kann ein Docker-Container gestartet werden. Die Verzeichnisse, z. B. die GRASS-Datenbank oder die Input-Daten, werden dabei mit *-v* in den Container gemountet:

```
$ docker run -it -v /pfad/zu/grassdata:/ -v /pfad/zu/Eingangsdaten:/mnt/data
rvr_interface:latest sh

# Oder alternativ:
$ docker run -it -v /pfad/zu/grassdata:/ -v /pfad/zu/Eingangsdaten:/mnt/data
rvr_interface:230922 sh
```

Es ist nicht notwendig, das Docker Image als root (sudo docker run …) zu starten. Alternativ können die Nutzenden der Gruppe „docker“ hinzugefügt werden:

```
# aktuell Nutzende der Gruppe „docker“ hinzufügen
$ sudo usermod -a -G docker <username>

# überprüfen mit
$ id
```

Solange GRASS innerhalb des Docker-Containers genutzt wird, werden die Pfade im Docker anstelle der lokalen Pfade verwendet. Nach dem obigen Einmounten sind das die folgenden Stammpfade:

- Pfad zur grassdb: */grassdb/…*
- Pfad zu den Input-Daten: */mnt/data/…*

Diese müssen also beim Durchgehen der Schritte aus der Dokumentation ggf. angepasst werden.

Zunächst wird eine neue ***Location*** mit dem EPSG-Code 25832 (Projektion: ETRS89 / UTM Zone 32N) angelegt. Alle darin importierten Datensätze werden dann über dieselbe Projektion verfügen:

```
# Allgemeines Schema zum Erstellen einer neuen GRASS Location
$ grass -c epsg:epsg-code /grassdb/neuer_location_name

# Erstellen einer neuen GRASS Location mit dem Namen „location_25832“
$ grass -c epsg:25832 /grassdb/location_25832
```

Im selben Terminal wird nun GRASS GIS gestartet, hier erfolgen die weiteren Schritte. Auch in einer geöffneten GRASS-Sitzung können weiterhin alle Terminal-Befehle ausgeführt werden (z. B. Verzeichnisse wechseln, *bash*-Skripte starten, etc.).

Innerhalb der GRASS-*Locations* gibt es *Mapsets*, die die Sammlung von thematisch zusammenhängenden Datensätzen ermöglichen. Bei Erstellung einer neuen *Location* wird automatisch das PERMANENT-***Mapset*** erstellt. Es ist jedoch sinnvoll, ein selbst benanntes **neues *Mapset*** zu erstellen:

```
# Neues Mapset namens “bottrop_2017” erstellen und dort hinein wechseln
$ g.mapset -c bottrop_2017

# TIPP: Für alle GRASS-Befehle können Sie <Befehl> --help ausführen, um
# mehr über die Benutzung des Befehls zu erfahren, z.B.:
$ g.mapset --help
```

Alternativ kann auch direkt beim Starten von GRASS ein neues *Mapset* erstellt werden:

```
# Allgemeines Schema zum Erstellen eines neuen Mapsets in einer bestehenden
# Location
$ grass -c /grassdb/location_name/neuer_mapset_name

# Erstellen eines neuen Mapsets „bottrop_2017“ in der Location „location_25832“
$ grass -c /grassdb/location_25832/bottrop_2017
```

Nun befinden sich die Nutzenden in GRASS innerhalb der (neuen) *Location* (im Beispiel: “location_25832”) und des neu angelegten *Mapsets* (im Beispiel: „bottrop_2017“). Von hier aus können die im weiteren Verlauf in GRASS stattfindenden Schritte ausgeführt werden.

Der folgende Befehl öffnet GRASS zu einem späteren Zeitpunkt in einer bestimmten *Location* und *Mapset* erneut:

```
# Allgemeines Schema zum Öffnen von GRASS in bestimmten Mapsets
$ grass /pfad/zu/grassdata/yourlocation/yourmapset

# Wiederaufnahme einer GRASS-Sitzung in der Location “location_25832” und dem
# Mapset “bottrop_2017”
$ grass /grassdb/location_25832/bottrop_2017
```

Zum Umbenennen oder Löschen eines bestehenden *Mapsets* kann direkt der Ordner des *Mapsets* in der *grassdb* umbenannt bzw. gelöscht werden, z. B.:

```
# Allgemeines Schema zum Umbenennen/Löschen von GRASS Mapsets
$ mv /pfad/zu/grassdata/yourlocation/yourmapset /pfad/zu/grassdata/yourloca-
tion/new_name
$ rm -rf /pfad/zu/grassdata/yourlocation/yourmapset
```

1.3.3 Ändern der Eigentumsverhältnisse der erstellten GRASS Daten

Der GRASS GIS Docker-Container läuft mit Root-Rechten. Daher werden die im Docker erstellten Daten als root erstellt, weshalb das Zugreifen auf diese Daten nicht wie gewohnt von allen Nutzenden möglich ist. Um z. B. eine im Docker erstellte GRASS GIS *Location/Mapset* von außerhalb des Dockers öffnen zu können, müssen vorher ggf. die Nutzenden und die Gruppe außerhalb des Dockers angepasst werden. Dies kann z. B. über die folgenden Befehle erfolgen:

```
# Abfrage der Rechte auf die gesamte GRASS Datenbank
$ ls -lah /pfad/zu/grassdata
drwxr-xr-x 14 root   root   4,0K Jan 31 13:43 location_erstellt_im_docker
drwxrwxr-x 5  USER   GRUPPE 4,0K Jan 24 15:36 location_erstellt_außerhalb_docker

# Anpassen der Nutzenden und der Gruppe (beide können aus der Antwort
# zu dem obigen ‚ls‘ an der anderen Location entnommen werden und
# dementsprechend gesetzt werden
$ sudo chown USER:GROUP -R /pfad/zu/grassdata/location_erstellt_im_docker
```

1.4 Benötigte Datensätze

Für die Gebäudeextraktion werden die folgenden Datensätze benötigt:

- Gebiet:
  - Es wird ein Gebiet im Vektordatenformat (z. B. Shapefile, Geopackage) benötigt, für das die Prozessierung ausgeführt werden soll.
  - Alle anderen Datensätze müssen dieses Gebiet vollständig abdecken.
- DOPs:
  - Die benötigten Digitalen Orthophotos (DOPs) mit vier Farbkanälen (RGBI) im .tif-Format sollten innerhalb eines gemeinsamen Verzeichnisses vorliegen (in diesem Verzeichnis dürfen keine weiteren Dateien enthalten sein).
- Punktwolken:
  - Die benötigten Punktwolken im .laz-Format sollten in einem gemeinsamen Ordner vorliegen (in diesem Verzeichnis dürfen keine weiteren Dateien enthalten sein).
- FNK:
  - Die Flächennutzungskartierung des zu untersuchenden Jahres kann in einem beliebigen Vektordatenformat (z. B. Shapefile, Geopackage) vorliegen. Die Spalte mit den entsprechenden Codes muss im Integer-Format vorliegen.

- ggf. Referenzdaten:
  - Soll ein Vergleich mit externen Gebäudedaten durchgeführt werden, werden entsprechende Referenzdaten benötigt, welche ebenfalls in einem beliebigen Vektordatenformat (z. B. Shapefile, Geopackage,) vorliegen können.
  - Alternativ können mittels Import-Addon die Gebäudedaten von OpenGeodata.NRW heruntergeladen werden. *Achtung*: Das Tool bezieht immer nur die aktuellsten Daten und keine historischen. Historische Daten müssen vorab selbst über OpenGeoddata.NRW heruntergeladen werden.
- ggf. Digitales Geländemodell (DGM; engl. digital terrain model (DTM)):
  - Falls nicht das NRW-DGM1 zur nDOM-Erstellung genutzt werden soll, kann auch ein selbstdefiniertes DGM mit einer idealen Auflösung von 0,5 - 1 m verwendet werden. Dieses muss in einem GDAL-konformen Format vorliegen (z. B. GeoTIFF, ASCII XYZ).

Für die Detektion von begrünten Dachflächen werden zusätzlich folgende Datensätze benötigt:

- Hausumringe (Vektor-Polygone):
  - Die Dachbegrünung wird auf die Flächen der Hausumringe begrenzt. Die Daten können selbst bereitgestellt (z. B. selbst erstellte Gebäudeumringe, ALKIS-Gebäudedatensatz) oder mit Hilfe des Import-Addons die Gebäudedaten von OpenGeodata.NRW heruntergeladen werden. *Achtung*: Das Tool bezieht immer nur die aktuellsten Daten und keine historischen. Historische Daten müssen vorab selbst über OpenGeodata.NRW heruntergeladen werden.
- ggf. Bäume (Vektor-Polygone):
  - Ein Vektordatensatz, der die Ausdehnung der Baumkronen enthält, hilft, Fehldetektionen durch Baumüberhänge über Dächern zu vermeiden. Dieser kann vorab z. B. mit Hilfe des Addons *m.analyse.trees* erzeugt werden (s. Dokumentation „Entwicklung eines Verfahrens zur automatischen Detektion von Baumstandorten“).

1.4.1 Import und Aufbereitung der Datensätze in GRASS GIS

Für den Import und die Aufbereitung der Eingangsdaten wurde das neue GRASS-Addon *m.import.rvr* entwickelt. Dieses fasst mehrere Arbeitsschritte zum Datenimport und zur Aufbereitung zusammen, wobei auch die Möglichkeit besteht, Gebäudedaten von OpenNRW bei Bedarf automatisch herunterzuladen. Das Import-Tool kann ebenso für das Baum-Tool (s. Dokumentation „Entwicklung eines Verfahrens zur automatischen Detektion von Baumstandorten“) genutzt werden. Nachfolgend findet sich eine Übersicht der Aufgaben des Import-Addons:

- Importieren der benötigten Datensätze
  - Gebiet, DOPs, 2,5D-Punktwolken, FNK sowie ggf. Referenzgebäude, Hausumringe, Bäume, DGM

- Aufbereitung der DOPs
  - Importieren der einzelnen DOP .tif-Dateien, die in dem angegebenen Gebiet liegen, und Resampling auf 0,5 m Auflösung
  - Zusammenfassen der einzelnen DOP-Tiles zu einem virtuellen Raster (VRT). Jeder Kanal liegt als eigene Rasterkarte (*dop_red_05*, *dop_green_05*, *dop_blue_05* und *dop_nir_05*) im Mapset. Diese VRT-Karten setzen sich wiederum aus den einzelnen DOP-Tiles zusammen, z. B *dop_red_05* aus verschiedenen anderen *dop_*_05*.
- Aufbereitung der 2,5D-Punktwolken
  - Importieren der 2,5D-Punktwolken in GRASS GIS und Verarbeitung zu einem Digitalen Oberflächenmodell (DOM; engl. *digital surface model* (DSM)) im Rasterformat mit 0,5 m Auflösung
  - Importieren aller .laz-Tiles als DOM-Raster und anschließend Zusammensetzung zu einem Gesamtraster
- Berechnung des NDVI
  - Berechnung des Normalised Difference Vegetation Index (NDVI) Rasters nach dem DOP-Import
  - Der NDVI benötigt den roten und den nahinfraroten Kanal des DOP-Mosaiks, die zuvor in dem Import-Addon importiert und resampled wurden
  - Der Wertebereich wird auf 8-Bit Integer Werte (0-255) skaliert, um das Raster performant zu halten
- Berechnung des normalisierten Digitalen Oberflächenmodells (nDOM; engl. *normalized digital surface model* (nDSM)) mit 0,5 m Auflösung
  - Zur Berechnung des nDOM (der Differenz aus DOM und DGM) wird das Addon *r.import.ndsm_nrw* im Import-Addon aufgerufen. Dieses interpoliert eventuelle NoData-Lücken im importierten DOM, die durch zu geringe Dichte der Punktwolken auftreten können, berechnet aus vorhandenem DOM und einem DGM das nDOM und resampled es auf die aktuelle *Region*.
  - Zur Berechnung des nDOM wird entweder ein selbstdefiniertes DGM oder das NRW-DGM verwendet. Zur Nutzung eines eigenen DGMs kann der Parameter *dsm_dir* im Import-Addon angegeben werden. Wird kein DGM selbst definiert, bezieht das Addon automatisch die benötigten Tiles des NRW-DGM.
  - Das DGM wird sowohl vom RVR als auch vom Land NRW gekachelt im XYZ-Format zur Verfügung gestellt. Dabei ist zu beachten, dass sich die XY-Koordinaten hier nicht wie allgemein üblich auf die Pixelmitte beziehen, sondern auf die linke untere Ecke eines Pixels. Wenn dies nicht beachtet wird, kann es zu einem Versatz um einen halben Pixel kommen, außerdem passen die Grenzen dann nicht mehr zu den Grenzen der DOP-Kacheln.
  - Bei der Nutzung eines eigenen DGMs wird dessen Versatz wie folgt gehandhabt:

- Bei der Nutzung von XYZ-Tiles wird der Versatz vom Import-Addon korrigiert (Auflösung des DGM muss angegeben werden).
- Bei der Nutzung eines GeoTiffs wird von einer vorher erfolgten Korrektur ausgegangen und das GeoTiff „*as is*“ eingeladen. Das Addon geht also davon aus, dass sich die Koordinaten im GeoTiff auf die Pixelmitte beziehen.

Im Folgenden werden die Parameter des Import-Addons genauer aufgeschlüsselt, die für die Gebäudeanalyse relevant sind:

Eingangsparameter:

- ***area***: Pfad zur Vektordatei, die das zu berechnende Gebiet enthält
- ***fnk_file***: Pfad zur Vektordatei der Flächennutzungskartierung (benötigt für Gebäudedetektion, optional für Dachbegrünung)
- ***fnk_column***: Spaltenname des Attributs mit den Klassencodes der Flächennutzungskartierung (benötigt für Gebäudedetektion, optional für Dachbegrünung)
- ***reference_buildings_file***: Pfad zur Vektordatei, die Gebäude-Referenzdaten für die Gebäudedetektion enthält (optional)
- ***building_outlines_file***: Pfad zur Vektordatei, die Gebäudeumrisse für die Dachbegrünungsdetektion enthält (benötigt für Dachbegrünung)
- ***tree_file***: Pfad zur Vektordatei, die Baumpolygone enthält, die bei der Dachbegrünungsdetektion zur Ergebnisoptimierung verwendet werden (optional)
- ***dop_dir***: Pfad zum Ordner, in dem die DOP-GeoTiffs liegen
- ***dop_tindex***: Pfad zu einem / für einen Tileindex für die DOPs, die im *dop_dir* liegen. Der Tileindex muss ein Attribut *location* haben, in dem der absolute Pfad zu den DOPs (innerhalb des Dockers) steht. Dieser wird entweder verwendet, sofern die Datei existiert, damit er nicht bei jedem Lauf des Addons erstellt werden muss oder gespeichert, wenn die Datei nicht existiert, damit er für zukünftige Läufe wiederverwendet werden kann. Ändert sich hingegen die Auflösung der Eingangsdaten (z. B. zwischen zwei Bildflugjahren von 10 cm auf 7,5 cm), muss dieser neu berechnet werden (optional).
- ***dsm_dir***: Pfad zum Ordner, in dem die DOM-LAZ liegen
- ***dsm_tindex*:** Pfad zu / für einen Tileindex für die DOM LAZ-Dateien, die im *dsm_dir* liegen. Der Tileindex muss ein Attribut *location* haben, in dem der absolute Pfad zu den .laz-Daten (innerhalb des Dockers) steht. Dieser wird entweder verwendet, sofern die Datei existiert, damit er nicht bei jedem Lauf des Addons erstellt werden muss oder gespeichert, wenn die Datei nicht existiert, damit er für zukünftige Läufe wiederverwendet werden kann. Das erste

Erstellen des Indexes kann lange dauern, weshalb es sinnvoll ist, den Index wiederzuverwenden. Ändert sich hingegen die Auflösung der Eingangsdaten (z. B. zwischen zwei Bildflugjahren von 10 cm auf 7,5 cm), muss dieser neu berechnet werden (optional).

- ***dtm_file***: Pfad zur DGM-Datei im Format GeoTiff (Import und Resampling ohne Korrektur) oder XYZ (Import und Shift, um die Angabe der unteren linken Ecke statt des Pixelzentrums zu korrigieren, hierfür wird die *dtm_resolution* benötigt); auch ein Pfad zu einem Ordner mit mehreren XYZ-Dateien kann angegeben werden (optional)
- ***dtm_resolution***: Original-Auflösung des DGM, damit bei Verwendung einer DGM-XYZ-Datei eine Korrektur des Versatzes durchgeführt werden kann (notwendig, wenn *dtm_file* als XYZ-Datei gegeben)
- ***dtm_tindex*:** Pfad zu einem / für einen Tileindex für die DGM XYZ-Dateien, die im *dtm_file* Ordner liegen. Der Tileindex muss ein Attribut *location* haben, in dem der absolute Pfad zu den .xyz-Daten (innerhalb des Dockers) steht. Dieser wird entweder verwendet, sofern die Datei existiert, damit er nicht bei jedem Lauf des Addons erstellt werden muss oder gespeichert, wenn die Datei nicht existiert, damit er für zukünftige Läufe wiederverwendet werden kann. Das erste Erstellen des Indexes kann lange dauern, weshalb es sinnvoll ist, den Index wiederzuverwenden. Ändert sich hingegen die Auflösung der Eingangsdaten (z. B. zwischen zwei Bildflugjahren von 10 cm auf 7,5 cm), muss dieser neu berechnet werden (optional).
- ***type***: Analysetyp, für den die benötigten Daten importiert werden sollen: „gebaeudedetektion“ oder „dachbegruenung“
- ***-c***: Flag zum Prüfen, ob alle für den Analysetyp erforderlichen Input-Daten importiert werden können, ohne dabei den eigentlichen Import zu starten
- ***-b***: Flag für den Download der aktuellen Referenzgebäudedaten von OpenNRW, sofern die entsprechenden Parameter *reference_buildings_file* oder *building_outlines_file* nicht gesetzt sind (Achtung: Es steht immer nur der aktuellste Datensatz zur Verfügung).

Ein Teil der Berechnung im Import-Modul kann parallel erfolgen. Für die Konfiguration der Parallelisierung können folgende Parameter gesetzt werden:

- ***memory***: Arbeitsspeicher in MB, der dem Addon zur Verfügung stehen soll (optional, Defaultwert ist 300 MB)
- ***nprocs:*** Anzahl der Kerne für Parallelprozessierung (optional, Defaultwert ist -2 (Anzahl der verfügbaren Kerne minus 1), für serielle Prozessierung auf 1 setzen)

```
# m.import.rvr bspw. ausführen mit:
$ DATAFOLDER=/mnt/data
$ m.import.rvr memory=300 type=gebaeudedetektion \
   area=${DATAFOLDER}/area_Dinslaken.gpkg \
   fnk_file=${DATAFOLDER}/FNK_Dinslaken/fnk_dinslaken.shp fnk_column=code_2020 \
   dsm_dir=${DATAFOLDER}/2020_Sommer/Punktwolke_2_5D_RGBI/ \
   dop_dir=${DATAFOLDER}/2020_Sommer/DOP/ \
   reference_buildings_file=${DATAFOLDER}/hu/hu_shp.shp
```

Hinweis:
Dieser Schritt kann aufgrund der Größe der Eingangsdateien und gemounteten Datenverzeichnisse mehrere Stunden in Anspruch nehmen.
Zur Vermeidung von Fehlern empfehlen wir für den Arbeitsspeicher den Defaultwert von 300 MB. Für *memory* erfolgt dann keine Angabe. Darüber hinaus ist es sinnvoll, die maximale Anzahl der verfügbaren Kerne minus 1 anzugeben (Default für *nprocs*), um die Rechenzeit gering zu halten.

Das Import-Addon importiert dann die folgenden Daten für die Gebäudeanalyse:

- ***study_area***: eine Vektorkarte des Gebietes
- ***reference_buildings***: die Hausumringe als Vektorkarte, die entweder angegeben wurden oder von OpenNRW heruntergeladen werden (bei *type=gebaeudedetektion*)
- **building_outlines**: die Hausumringe als Vektorkarte, die entweder angegeben wurden oder von OpenNRW heruntergeladen werden (bei *type=dachbegruenung*)
- **fnk**: Vektorkarte der Flächennutzungskartierung
- ***dop_red_05, dop_green_05, dop_blue_05, dop_nir_05***: die Rasterkarten der einzelnen Kanäle des DOPs
- ***dop_ndvi_05***: die Rasterkarte des berechneten und skalierten NDVI
- ***dsm_05***: die Rasterkarte des importierten DOMs
- ***ndsm***: die Rasterkarte des berechneten nDOMs
- **trees**: Bäume als Vektorkarte (bei Angabe eines tree-Layers und *type=dachbegruenung*)

2 Analyse

Das neue Multi-Modul *m.analyse.buildings* umfasst die bisherigen Addons *r.extract.buildings*, *v.cd.areas* (vorher *v.cd.buildings*) und *r.extract.greenroofs.*

Die Installation des Multi-Addons installiert automatisch die eben genannten Gebäudeanalyse-Module sowie ihre zugehörigen Worker-Module. Die Worker-Module werden indirekt durch die Hauptmodule aufgerufen und dienen der Prozessierung über ein Gitter. Durch das sogenannte Tiling erfolgt ein Großteil der Berechnung über mehrere Kacheln.

2.1 Tiling

Beim Tiling wird das Untersuchungsgebiet in mehrere Kacheln zerschnitten und der Analyseprozess Kachel für Kachel gerechnet. Dies gilt für einen Großteil der Berechnungen. Manche Schritte, z. B. das Festlegen von Schwellenwerten anhand eines Perzentils, werden weiterhin für das Gesamtgebiet durchgeführt, damit sich der Wert auf das Gesamtgebiet bezieht. Die Kantenlänge des Gitters und damit der einzelnen Kacheln kann selbst gewählt werden, standardmäßig liegt sie bei 1000 m.

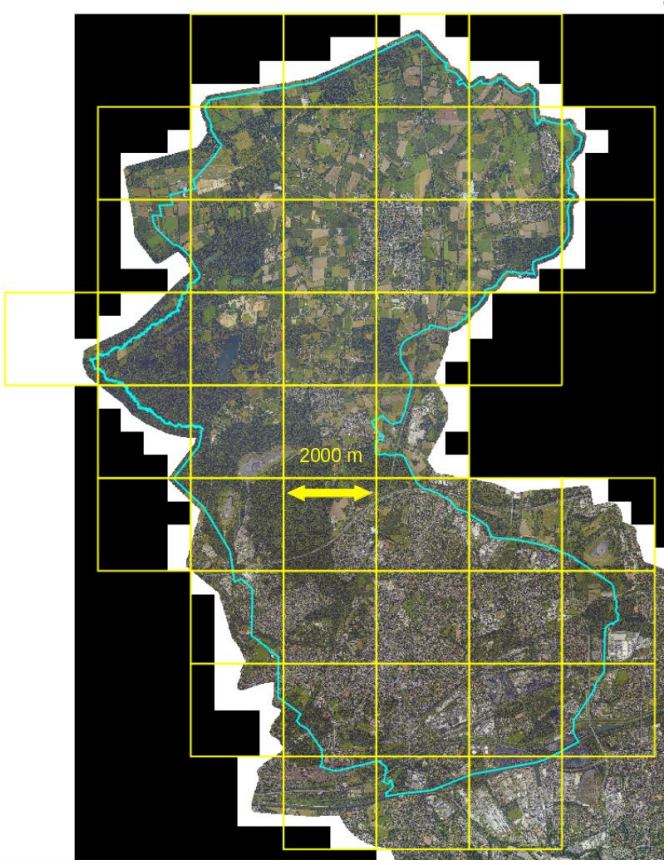

Abbildung 1: Gitter mit 2000 m Kantenlänge für gekachelte Prozessierung

Zusätzlich kann der Parameter ***nprocs***, also die Anzahl der parallelen Prozesse, festgelegt werden. Hiernach richtet sich, wie viele Kacheln gleichzeitig berechnet werden. Für eine serielle Prozessierung ohne Parallelisierung sollte der *nprocs*-Paramter auf 1 gesetzt werden. Standardmäßig ist nprocs auf -2 gesetzt. Das bedeutet, es wird die Anzahl der verfügbaren Kerne ermittelt und von dieser Anzahl 1 abgezogen, sodass wenig rechenintensive Operationen, wie z. B. das Monitoring des Analyseprozesses, weiterhin durchgeführt werden können. Wenn mehrere Prozesse gleichzeitig auf der Maschine laufen, sollte der *nprocs*-Parameter deutlich herabgesetzt werden, sodass weniger Kerne pro Prozess genutzt werden und die Rechenlast pro Prozess geringer ist.

Gleiches gilt auch für den *memory*-Parameter. Dieser sollte beim Starten mehrerer Prozesse gleichzeitig nicht auf dem maximalen Wert der Maschine liegen, sondern herabgesetzt werden.

2.2 Gebäudedetektion

2.2.1 Gebäudeextraktion

Die Identifikation von Gebäuden erfolgt mit dem Addon *r.extract.buildings*. Dieses Tool extrahiert potenzielle Gebäude im Vektorformat anhand ihrer Höhe im nDOM sowie des NDVI-Werts. Gebäude werden von Vegetation anhand eines NDVI-Schwellenwerts unterschieden. Es bestehen zwei Möglichkeiten, den NDVI-Schwellenwert zu definieren. Er kann einerseits fest über den **ndvi_thresh-**Parameter angegeben werden. Gute Ergebnisse für das Testgebiet können hier mit Schwellenwerten zwischen 145 und 150 erzielt werden (da der Wertebereich des NDVI-Rasters zwischen 0 und 255 liegt, entspricht dieser Bereich auf der klassischen NDVI-Skala von -1 bis +1 etwa 0,14 bis 0,18). Andererseits kann der Schwellenwert halbautomatisch aus dem **ndvi_perc-**Parameter ermittelt werden. In diesem Fall werden NDVI-Statistiken aus den mit Vegetation bedeckten Flächen berechnet (diese werden durch die FNK-Codes 400, 410, 420, 431, 432, 441, 472 maskiert). Das gewählte Perzentil dient dann als NDVI-Schwellenwert. Die Grundannahme ist, dass ein niedrig gewähltes Perzentil (z. B. 5 %) eine gute Trennung zwischen Vegetation und Nicht-Vegetation ermöglicht, da es den niedrigsten Werten innerhalb von eigentlich mit Vegetation bedeckten Flächen entspricht. Dieses Verfahren ist zu empfehlen, wenn bisher keine Erfahrungen über geeignete NDVI-Schwellenwerte (als **ndvi_thresh**) vorliegen. Die bisherigen Ergebnisse zeigen jedoch, dass ein **ndvi_thresh** von 145 -150 für die Testgebiete Bottrop und Erkenschwick geeignet ist. Daher wird empfohlen, den **ndvi_thresh-**Parameter zu verwenden.

Bei der Gebäudeerkennung werden zur Verminderung von Prozessierungszeit und von Fehldetektionen Flächen ohne potentielle Gebäude anhand ihrer Landnutzungsklasse aus der Landnutzungskartierung von der Analyse ausgeschlossen. Diese Landnutzungsklassen umfassen mögliche Lagerflächen, Reserveflächen, Lager für Rohstoffe, Bahnanlagen, Flug- und Landeplätze, Freiflächen, Abgrabungsflächen, Friedhöfe, Begleitgrün, Wasserflächen, Wiesen & Weiden, Ackerflächen, Berghalden und Straßen.

Wird die **-s**-Flag angegeben, erfolgt die Analyse objektbasiert (durch Segmentierung) statt pixelbasiert. Dies kann homogenere Ergebnisobjekte erzeugen, erhöht aber die Prozessierungsdauer. Es wird daher empfohlen, in mehreren Durchläufen ohne die **-s**-Flag einen geeigneten **ndvi_thresh-**Wert zu identifizieren und ggf. in einem späteren Durchlauf die **-s**-Flag zu aktivieren.

Der Ergebnis-Vektordatensatz enthält Spalten zu nDOM-Statistiken (z. B. „*ndom_minimum“*, „*ndom_median*“) sowie Spalten mit der Flächengröße („*area_sqm*“), der Fractal Dimension („*fractal_d*“) und der geschätzten Etagenzahl („*floors*“) ausgehend von einer durchschnittlichen Etagenhöhe von 3 m. Im Folgenden werden die **Parameter** dieses Addons genauer aufgeschlüsselt:

- ***ndsm***: Name des nDOM-Rasterdatensatzes
- ***ndvi_raster***: Name des NDVI-Rasterdatensatzes
- ***fnk_vector***: Name des FNK-Vektordatensatzes

- ***fnk_column***: Spalte des FNK-Vektordatensatzes, die die FNK-Codes enthält
- ***min_size***: Minimalgröße der zu detektierenden Objekte in m² (Defaultwert ist 20 m²)
- ***max_fd***: Maximalwert der *fractal dimension* (fd) der detektierten Objekte. Die *fractal dimension* berechnet sich als fd=2*(log(Umfang)/log(Fläche)) und ist ein Kompaktheitsmaß. Ein Maximalwert hilft, sehr lange dünne Objekte, die z. B. fälschlicherweise an Waldrändern detektiert werden können, auszuschließen (Defaultwert ist 2.1, dieser wurde empirisch ermittelt).
- ***ndvi_perc***: Perzentil zur automatischen Ableitung des NDVI-Schwellenwerts (z. B. 5 für 5 %-Perzentil)
- ***ndvi_thresh***: Selbst gewählter NDVI-Schwellenwert (z. B. 145)
- ***output***: Name des Ergebnis-Vektordatensatzes
- **-s**: Aktivieren der Bildsegmentierung auf Grundlage von nDOM und NDVI vor der Gebäudeextraktion für objektbasierte anstatt pixelbasierter Analyse (erhöht die Prozessierungsdauer erheblich)

Ein Teil der Berechnung kann parallel erfolgen. Für die Konfiguration der Parallelisierung können folgende Parameter gesetzt werden:

- ***memory***: Arbeitsspeicher in MB, der dem Addon zur Verfügung stehen soll (optional, Defaultwert ist 300 MB)
- **nprocs**: Anzahl der Kerne für Parallelprozessierung (optional, Defaultwert ist -2 (Anzahl der verfügbaren Kerne minus 1), für serielle Prozessierung auf 1 setzen)
- **tile_size**: Kantenlänge der Gitterkacheln für das Tiling in m (optional, Defaultwert ist 1000 m)

Besonders wichtig bei Raster-Berechnung ist es, die ***Region*** richtig zu setzen, da diese die Ausdehnung und Auflösung des Ergebnisrasters bestimmt. Für die gesamte Analyse ist es sinnvoll, die *Region* auf einen der importierten DOP-Mosaik-Kanäle anzupassen, daher wird dies im Beispiel vor dem eigentlichen Aufruf des GRASS-Addons gemacht:

```
# Computational Region auf ein DOP-Raster setzen
$ g.region rast=dop_red_05 -p


# r.extract.buildings mit NDVI-Schwellenwert, Defaultwerten für Mindestgröße und
# Maximum der Fractal Dimension mit einer Gittergröße von 2000 m ausführen:
$ r.extract.buildings ndsm=ndsm ndvi_raster=dop_ndvi_05 fnk_vector=fnk fnk_col-
umn=code_2017 ndvi_thresh=145 output=buildings_Bottrop_2017 memory=6000
tile_size=2000
```

2.2.2 Vergleich mit Referenzdaten

Die durch *r.extract.buildings* identifizierten Gebäude können anschließend mit Hilfe des Addons *v.cd.areas* (ehemals *v.cd.buildings*) mit einem beliebigen Referenzdatensatz im Vektorformat verglichen werden. Das Ergebnis von *v.cd.areas* ist ein Vektordatensatz, der Flächen enthält, die entweder nur im Ergebnis von *r.extract.buildings* oder nur im Referenzdatensatz vorkommen. Dies wird in der Ergebnisspalte „*source*“ angegeben. Kleinere Artefakte, die bei der Erstellung des Differenzlayers entstehen, werden größtenteils entfernt. Zusätzlich lässt sich über den **min_size**-Parameter noch eine Mindestgröße angeben, die die Veränderungsflächen aufweisen müssen.

Zusätzlich können mit dem Addon Qualitätsmaße für die extrahierten Gebäude berechnet werden. Für binäre Klassifikationen, z. B. Gebäude/Nicht-Gebäude, sind klassische Fernerkundungsvalidierungsmaße wie die Produzent:innen- und Nutzer:innen-Genauigkeit (*Producer‘s* und *User‘s Accuracy*) nicht geeignet; stattdessen werden die folgenden beiden Maße genutzt:

Vollständigkeit als Maß für korrekt identifizierte Gebäudefläche geteilt durch die Gesamtfläche der Gebäude in der Referenz und *Korrektheit* als korrekt identifizierte Gebäudefläche geteilt durch die Gesamtfläche der Gebäude in der Klassifikation.

Im Folgenden werden die **Parameter** des Addons genauer aufgeschlüsselt:

- ***input***: Name des mit *r.extract.buildings* erstellten Gebäude-Vektordatensatzes
- ***reference***: Name des Referenz-Vektordatensatzes
- ***min_size***: Mindestgröße der zu erfassenden Differenzen in m² (Defaultwert ist 5 m²)
- ***max_fd***: Maximalwert der *fractal dimension* (fd) der Differenzen. Da hier auch eher „unkompakte“ Objekte korrekte Differenzen darstellen können, empfiehlt sich ein etwas höherer Schwellenwert als bei *r.extract.buildings* (Defaultwert ist 2.5).
- ***output***: Name des Ergebnis-Vektordatensatzes
- **-q**: Aktivieren der Berechnung der Qualitätsmaße

Ein Teil der Berechnung kann parallel erfolgen. Für die Konfiguration der Parallelisierung können folgende Parameter gesetzt werden:

- **nprocs**: Anzahl der Kerne für Parallelprozessierung (optional, Defaultwert ist -2 (Anzahl der verfügbaren Kerne minus 1), für serielle Prozessierung auf 1 setzen)
- **tile_size**: Kantenlänge der Gitterkacheln für das Tiling in m (optional, Defaultwert ist 1000 m)

```
# Computational Region auf ein DOP-Raster setzen
$ g.region rast=dop_red_05 -p

# v.cd.areas mit einer Gittergröße von 2000 m ausführen:
$ v.cd.areas input=buildings_Bottrop_2017 reference=reference_buildings
min_size=10 output=buildings_alkis_diff_Bottrop_2017 tile_size=2000

# v.cd.areas mit Qualitätsmaßen (-q) ausführen
$ v.cd.areas -q input=buildings_Bottrop_2017 reference=reference_buildings
min_size=10 output=buildings_alkis_diff_Bottrop_2017
```

2.3 Dachbegrünungsdetektion

Die Identifikation von Dachbegrünung auf Gebäuden erfolgt mit dem Addon *r.extract.greenroofs.* Dieses Tool extrahiert begrünte Dachflächen und die zugehörigen Gebäude im Vektorformat anhand ihrer Höhe im nDOM, eines normalisierten Grün-Blau-Ratio-Werts (GBR) und ihres Helligkeitwerts.

Für die Extraktion von Gebäuden mit Dachbegrünung wird das Untersuchungsgebiet auf bestehende Gebäudeumringe eingegrenzt. Begrünte Flächen werden von nicht-begrünten Flächen anhand eines Schwellenwerts des normalisierten GBR sowie der Helligkeit unterschieden. Die Helligkeit wird aus dem roten und grünen Kanal berechnet (*Rot + Grün)/2)*. Der blaue Kanal wird bei der Berechnung des Graustufenrasters nicht einbezogen, da er aufgrund der stärkeren Reflektion blauen Lichtes von der Atmosphäre den Helligkeitswert leicht verfälschen kann. Es bestehen zwei Möglichkeiten, den GBR-Schwellenwert zu definieren. Er kann einerseits fest über den **gb_thresh-**Parameter angegeben werden. Gute Ergebnisse konnten für die Untersuchungsgebiete Dinslaken (2020) und Bottrop (2017) mit Schwellenwerten um 145 erzielt werden (da der Wertebereich des GBR-Rasters zwischen 0 und 255 liegt, entspricht dieser Bereich auf der klassischen normalisierten Ratio-Skala von -1 bis +1 etwa 0,14). Andererseits kann der Schwellenwert halbautomatisch aus dem **gb_perc-**Parameter ermittelt werden. In diesem Fall werden GBR-Statistiken aus den mit Vegetation bedeckten Flächen berechnet (diese werden durch die FNK-Codes 400, 410, 420, 431, 432, 441, 472 maskiert). Das gewählte Perzentil dient dann als GBR-Schwellenwert. Beispielsweise bedeutet ein Wert von gb_perc=25, dass der GBR-Schwellenwert vom 25%-Perzentil aller über die FNK als Vegetation gekennzeichneten Pixel verwendet wird. Dieses Verfahren ist zu empfehlen, wenn bisher keine Erfahrung zu geeigneten GBR-Schwellenwerten (als **gb_thresh**) vorhanden ist. Zusätzlich wird eine Rot-Grün-Ratio, welche nicht selbstdefinierbar ist, zur besseren Differenzierung in verschatteten Bereichen und bei sehr roten Dächern eingesetzt.

Das Modul erzeugt die besten Ergebnisse, wenn ein Baum-Polygonlayer zur Verfügung steht, der über den **trees**-Parameter übergeben werden kann. Wenn dieser nicht zur Verfügung steht, versucht das Modul, Baumüberhänge selbst durch den nDOM-Unterschied zum restlichen Gebäude zu eliminieren. Dies funktioniert jedoch nur bei entsprechenden Höhenunterschieden.

Wird die **-s**-Flag angegeben, erfolgt die Analyse objektbasiert (durch Segmentierung) statt pixelbasiert. Dies erzeugt homogenere Ergebnisobjekte und grundsätzlich ein genaueres Ergebnis, erhöht jedoch die Prozessierungsdauer. Es wird daher empfohlen, in mehreren Durchläufen ohne die **-s**-

Flag einen geeigneten **gb_thresh-**Wert zu identifizieren und im finalen Durchlauf die **-s**-Flag zu aktivieren.

r.extract.greenroofs erfordert die folgenden **Input-Parameter**:

- ***ndsm***: Name des nDOM-Rasterdatensatzes
- ***ndvi***: Name des NDVI-Rasterdatensatzes
- ***red***: Name des roten DOP-Rasterdatensatzes
- ***green***: Name des grünen DOP-Rasterdatensatzes
- ***blue***: Name des blauen DOP-Rasterdatensatzes
- ***fnk***: Name des FNK-Vektordatensatzes (optional, nur erforderlich, wenn ***gb_perc*** angegeben wird)
- ***fnk_column***: Spalte des FNK-Vektordatensatzes, die die FNK-Codes enthält (optional, nur erforderlich wenn ***gb_perc*** angegeben wird)
- ***buildings***: Name des Hausumringe-Vektordatensatzes
- ***trees***: Name des Bäume-Vektordatensatzes (optional)
- ***gb_thresh***: Selbst gewählter GBR-Schwellenwert (z. B. 145, optional, entweder *gb_thresh* oder *gb_perc* muss gewählt werden)
- ***gb_perc***: Perzentil zur automatischen Ableitung des GBR-Schwellenwerts (z. B. 25 für 25%-Perzentil, optional, entweder *gb_thresh* oder *gb_perc* muss gewählt werden)
- ***min_veg_size***: Minimalgröße der zu detektierenden Vegetationsobjekte in m² (default: 5 m²)
- ***min_veg_proportion***: Mindestanteil der Dachbegrünung am Gesamtdach in % (Defaultwert: 10 %)
- **-s**: Objektbasierte statt pixelbasierte Analyse aktivieren (erhöht die Prozessierungsdauer erheblich)

Ein Teil der Berechnung kann parallel erfolgen. Für die Konfiguration der Parallelisierung können folgende Parameter gesetzt werden:

- ***memory***: Arbeitsspeicher in MB, der für die Prozessierung zur Verfügung stehen soll (optional, Defaultwert ist 300 MB)
- **nprocs**: Anzahl der Kerne für Parallelprozessierung (optional, Defaultwert ist -2 (Anzahl der verfügbaren Kerne minus 1), für serielle Prozessierung auf 1 setzen)
- **tile_size**: Kantenlänge der Gitterkacheln für das Tiling in m (optional, Defaultwert ist 1000 m)

Zusätzlich müssen die folgenden **Output-Parameter** definiert werden:

- **output_buildings** enthält das Subset des Input-Datensatzes **buildings** (Hausumringe), das über Dachbegrünung verfügt. Es wird die Attributspalte *vegetation_proportion* angefügt, in der der prozentuale Anteil der begrünten Fläche im Verhältnis zur Gesamtdachfläche angegeben ist.
- **output_vegetation** enthält die Dachbegrünungsflächen als Vektorobjekte.

Besonders wichtig bei Raster-Berechnungen ist es, die ***Region*** richtig zu setzen, da diese die Ausdehnung und Auflösung des Ergebnisrasters bestimmt. Für die gesamte Analyse ist es sinnvoll, die *Region* auf einen der importierten DOP-Mosaik-Kanäle anzupassen, daher wird dies im Beispiel vor dem eigentlichen Aufruf des GRASS-Addons gemacht:

```
# Computational Region auf ein DOP-Raster setzen
$ g.region rast=dop_red_05 -p


# r.extract.greenroofs mit Segmentierung (-s), einem Baumlayer und einer
# Gittergröße von 2000 m ausführen:
$ r.extract.greenroofs -s ndsm=ndsm ndvi=dop_ndvi_05 red=dop_red_05
green=dop_green_05 blue=dop_blue_05 gb_thresh=145 buildings=building_outlines
trees=trees output_buildings=begruente_gebaeude output_vegetation=dachgruen
memory=8000 tile_size=2000
```

Hinweis:
Wenn die Dachbegrünungsdetektion im Anschluss an die Gebäudedetektion durchgeführt werden soll, ist es notwendig den Import-Prozess erneut durchzuführen und hierbei den Typ *dachbegruenung* zu wählen mit den entsprechenden erforderlichen Parametern.

2.4 Gültigkeitsbereich der Analysetools

Wie jedes Klassifizierungstool weisen auch die Gebäude-Analysetools Grenzen auf. Grundsätzlich gilt, dass eine Klassifikation auf die Qualität ihrer Eingangsdaten angewiesen ist. Sind dort Fehler oder Ungenauigkeiten enthalten, wirkt sich dies auf das Ergebnis der Analyse aus. Darüber hinaus sind im Folgenden der mögliche Anwendungsbereich, aber auch die Limitierungen der Tools aufgeführt.

Zu den typischen Problemquellen bei der **Gebäudeerkennung** zählen zum einen falsch positive Ergebnisse, also das Erkennen von gebäudeartigen Objekten, z. B. Container oder stehende Züge, als Gebäude. Hier kann ggf. über die Parameter der Mindestgröße (*min_size*) und der Fractal Dimension (*max_fd*) etwas gegengesteuert werden. Ebenso können beispielsweise Brücken als Gebäude fehlinterpretiert werden, da diese im NDVI, der Höhe im nDOM und ihrer Größe den Kriterien für die Gebäudeselektion entsprechen können. Falls Bäume als Gebäude erkannt werden, kann dies am ehesten über den NDVI-Schwellenwert reguliert werden.
Zum anderen ist es umgekehrt möglich, dass Gebäude z. B. aufgrund ihrer Dachfarbe nicht erkannt werden. Dies betrifft insbesondere grüne/bläuliche Dächer, die einen hohen NDVI-Wert aufweisen

können. Auch Gebäude unter Bäumen können aufgrund der Differenzierung anhand des NDVI nicht erkannt werden. Um auch kleine Gebäude zu extrahieren, muss ggf. der Parameter der Mindestgröße (*min_size*) angepasst werden, standardmäßig liegt er bei 20 m².
Darüber hinaus kann der NDVI-Schwellenwert, der zur Extraktion potentieller Gebäude genutzt wird, je nach genutzten DOPs variieren. Hier kommt es z. B. auf den Befliegungszeitraum für die Luftbildaufnahmen an, ggf. kann es sinnvoll sein, mit verschiedenen NDVI-Schwellenwerten zu experimentieren und/oder große Untersuchungsgebiete in kleinere Einheiten zu unterteilen.

Das Ergebnis der **Veränderungsdetektion** zwischen zwei Gebäudelayern kann neben ganzen Gebäuden, die nur in einem der beiden Layer vorkommen, auch „Randstücke“ oder „-(teil)ringe“ von Gebäuden aufweisen. Diese werden durch die Defaulteinstellungen für die Mindestgröße (*min_size*) und den Maximalwert der Fractal Dimension (*max_fd*) bereits zum Teil entfernt. Es ist ggf. sinnvoll, hier mit anderen Werten zu experimentieren, um mehr dieser Objekte zu entfernen (Achtung: Dadurch werden möglicherweise auch Polygone entfernt, die im Ergebnis gewünscht sind).

Herausforderungen bei der **Detektion von Dachbegrünung** treten z. B. durch Schatten, durch Baumkronenüberhang und durch hohe Vegetation auf Untersuchungsflächen auf.
Die Analyse der Dachvegetationserkennung wird auf Gebäudeflächen begrenzt, die durch den Parameter *buildings* angegeben werden. Da das Tool von der Richtigkeit der Eingangsdaten ausgeht, können auf Flächen, die durch diesen Layer als Gebäude gekennzeichnet sind, auf denen aber kein Gebäude steht sondern Bäume oder Hecken wachsen, Fehler auftreten. Diese können dann aufgrund ihrer Eigenschaften als Dachvegetation interpretiert werden. Um Schwierigkeiten in verschatteten Bereichen auf Dächern oder bei sehr roten Dächern zu mindern, werden ein Helligkeitsraster und eine Rot-Grün-Ratio eingesetzt. Baumkronenüberhänge versucht das Modul durch den nDOM-Unterschied zwischen den Vegetationsflächen und dem restlichen Gebäude zu eliminieren. Dies funktioniert jedoch nur bei entsprechenden Höhenunterschieden. Bessere Ergebnisse werden daher in der Regel durch Nutzung eines zusätzlichen (optionalen) Baumlayers erzielt. Befindet sich ein Dach komplett unterhalb von Bäumen, kann es nicht korrekt identifiziert werden. Zudem sei darauf hingewiesen, dass das Tool generell Vegetation auf Dächern erkennt, aber nicht zwischen Dachbegrünung und Spontanbewuchs unterscheiden kann.
Aufgrund oben beschriebener Problematiken ist zu berücksichtigen, dass mögliche Fehlklassifikationen folglich auch die Berechnung von Statistiken, wie des prozentualen Anteils der Vegetation am Dach, beeinflussen können.
Wie der NDVI-Schwellenwert bei der Gebäudeextraktion kann auch der GBR-Schwellenwert, der bei der Dachbegrünungsdetektion zur Unterscheidung von begrünten und nicht-begrünten Flächen genutzt wird, je nach genutzten DOPs variieren. Ggf. kann es sinnvoll sein, mit verschiedenen GBR-Schwellenwerten zu experimentieren und/oder große Untersuchungsgebiete in kleinere Einheiten zu unterteilen.

3 Visualisierung und Export

3.1 Visualisierung in GRASS

Die Ergebnisse von *r.extract.buildings, v.cd.areas* und *r.extract.greenroofs* lassen sich in GRASS visualisieren. In dem GRASS GIS-Docker ist die GUI leider nicht verfügbar, jedoch kann die GRASS-GUI außerhalb des Dockers zur Visualisierung genutzt werden. Hierzu muss GRASS GIS lokal installiert sein und gegebenenfalls noch der Besitz der GRASS GIS Location angepasst werden (s. Kapitel 1.3.3). Dann kann die GRASS-Session gestartet werden:

```
$ grass /pfad/zu/grassdb/location_name/mapset_name
```

Falls die GRASS-GUI sich nicht automatisch öffnet, kann sie mit

```
$ g.gui
```

gestartet werden. Zur einfacheren Interpretation ist es hilfreich, sich im Hintergrund das DOP-Mosaik als Echtfarbenkomposit anzeigen zu lassen. Zunächst sollte der Kontrast des DOP-Mosaiks zur Visualisierung angepasst werden:

```
$ i.colors.enhance red=dop_red_05 green=dop_green_05 blue=dop_blue_05
```

Durch Klick auf die Schaltfläche „V*erschiedene Rasterkartenlayer hinzufügen*“ - „*RGB-Karte hinzufügen*“ können die Kanäle für die Anzeigefarben <red>, <green> und <blue> gewählt werden:

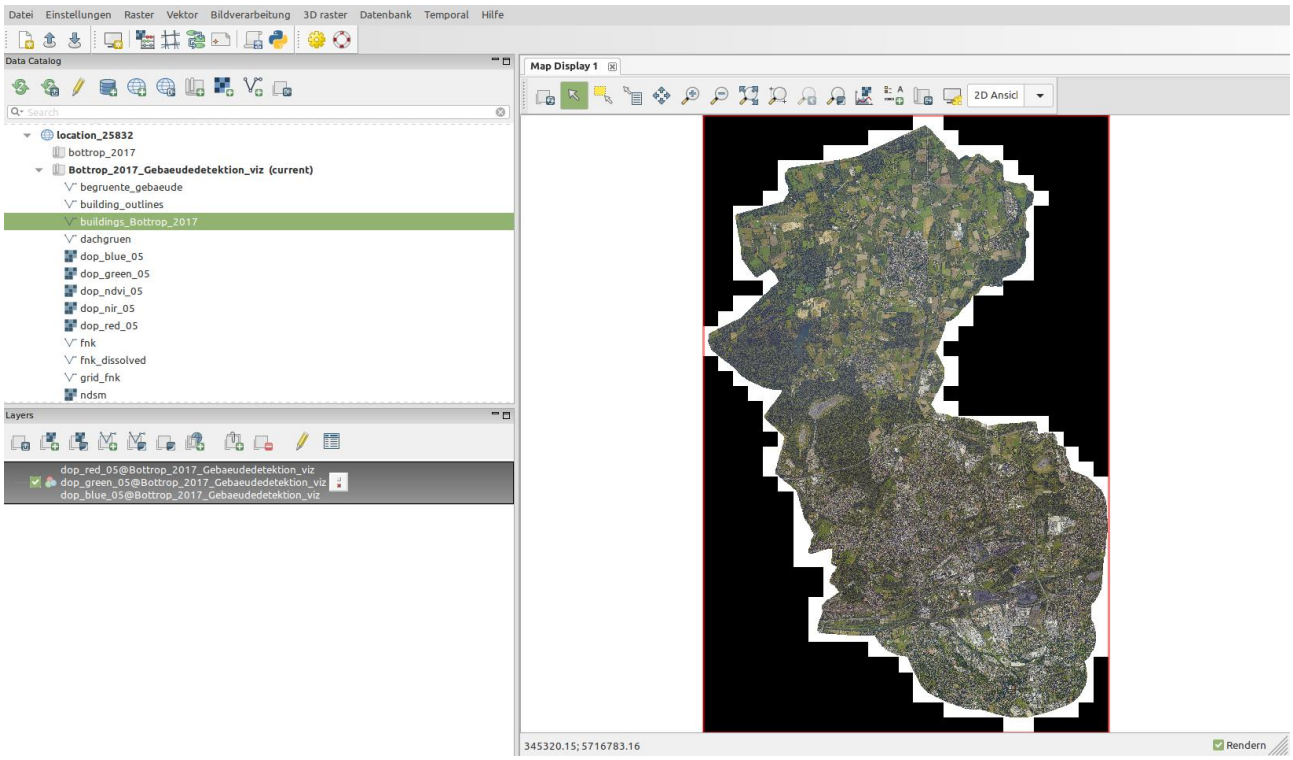

Abbildung 2: RGB-Darstellung des DOP-Mosaiks

Nun können auch die Vektor-Ergebniskarten aus *r.extract.buildings* und *v.cd.areas* per Klick auf die Schaltfläche „*Vektorkarten hinzufügen*“ oder per Doppelklick auf die entsprechende Karte im *Data Catalog*-Fenster in die Ansicht geladen werden:

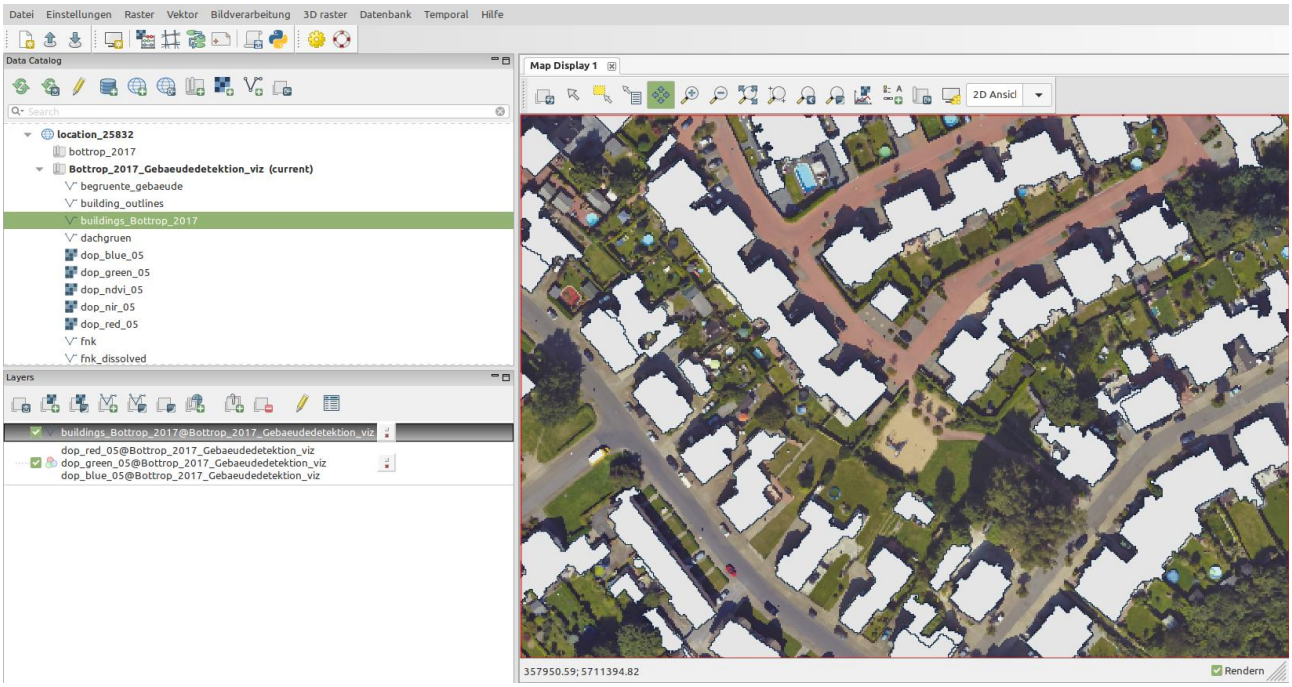

Abbildung 3: Darstellung des Gebäude-Ergebnislayers

Ebenso lassen sich die Ergebnisse von *r.extract.greenroofs* hinzufügen. Per Doppelklick auf den hinzugefügten Layer kann in den Reitern „*Farben*“ und „*Linien*“ die Darstellung geändert werden:

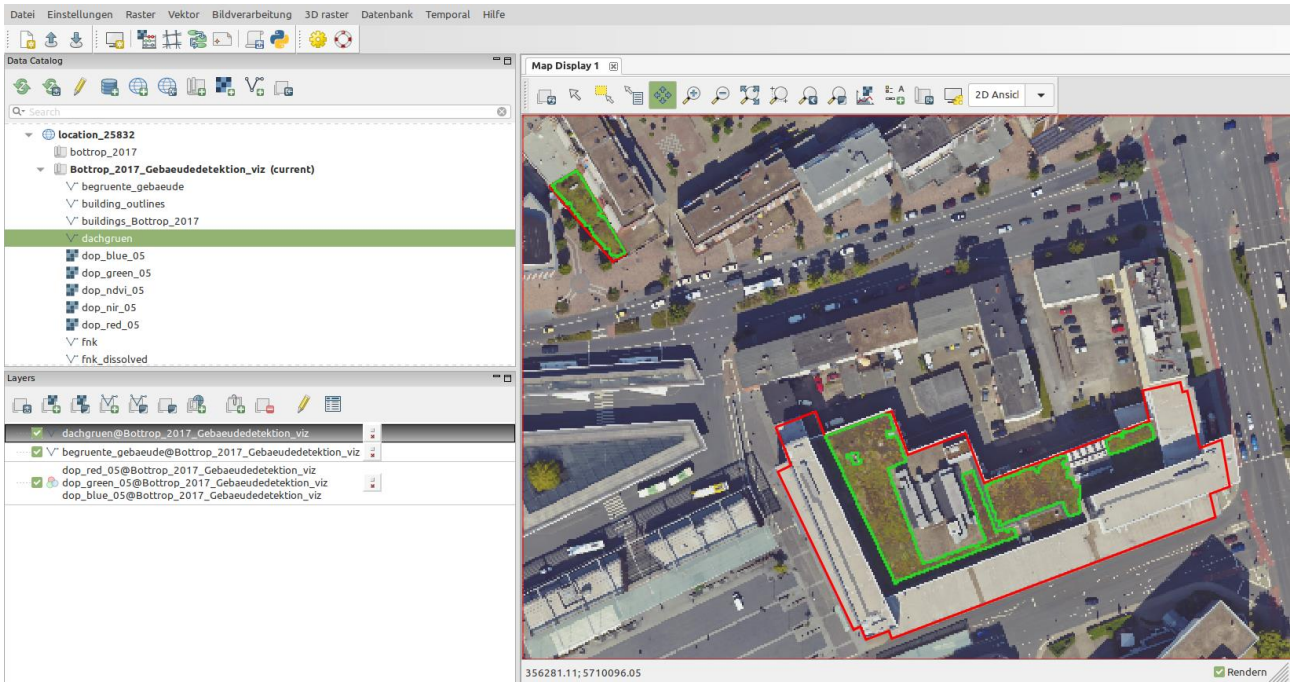

Abbildung 4: Darstellung der Dachbegrünung-Ergebnislayer: Gebäude mit Dachbegrünung (rot) und Dachbegrünungsobjekte (grün)

3.2 Export der Daten

Für einen ersten Eindruck genügt die Visualisierung in GRASS, die schnelle Anpassung der Darstellung ist jedoch in anderen Geoinformationssystemen wie QGIS oder ArcGIS etwas leichter. Zu diesem Zweck sowie zur weiteren Verarbeitung können die Ergebnisse daher zuletzt als gängige Vektordatenformate exportiert werden. Dazu kann das Tool *v.out.ogr* verwendet werden. Dabei muss beachtet werden, dass der Pfad zu den gemounteten Verzeichnis (z. B. „*/mnt/data*“) angegeben werden muss, wenn im GRASS-Docker gearbeitet wird, damit die Daten nach Schließen des Dockers nicht verloren gehen. Über den ***format***-Parameter kann das gewünschte Dateiformat der ***output***-Datei gewählt werden:

```
# Ergebnis der Gebäudedetektion als Shapefile exportieren:
$ v.out.ogr input=buildings_Bottrop_2017 output=/pfad/zum/output.shp for-
mat=ESRI_Shapefile

# Ergebnis des Referenzvergleichs als Geopackage exportieren:
$ v.out.ogr input=buildings_alkis_diff_Bottrop_2017 output=/pfad/zum/output.gpkg
format=GPKG

# Ergebnisse der Dachbegrünungsdetektion als Geopackage exportieren:
$ v.out.ogr input=begruente_gebaeude output=/pfad/zum/output.gpkg format=GPKG
$ v.out.ogr input=dachgruen output=/pfad/zum/output.gpkg format=GPKG

# Export der Rasterkarten als GeoTiff, z. B. ndsm:
$ r.out.gdal input=ndsm_map output=/pfad/zum/output.tif format=GTiff
```

4 Umgang mit GRASS Warnungen und Fehlern

Die GRASS-Module geben prinzipiell eher häufig Warnungen aus, um eine eventuelle Fehlersuche zu erleichtern. Diese dienen der Vollständigkeit und können daher normalerweise ignoriert werden. Sollte jedoch ein Modul nicht das gewünschte Ergebnis produzieren, lohnt es sich, die GRASS-Ausgabe auf eventuelle Warnungen hin zu überprüfen. Eine gute Anlaufstelle für mehr Informationen sind außerdem die Handbuch-Seiten der jeweiligen Module. Die Modul-Übersichten, geordnet nach Gruppen (*vector*, *raster*, *general*, etc.), können z. B. über diese Manual-Seite erreicht werden.

4.1 Häufige Fehler und Warnungen

Im Folgenden werden häufige Fehler und Warnungen beim Prozessieren von Daten mit GRASS GIS sowie der Umgang mit ihnen beschrieben. Darüber hinaus lohnt es sich, bei Fehlern, insbesondere beim Im- und Export, auch in den Handbuch-Seiten der Module nach weiteren Optionen zu schauen, z. B. via Aktivieren von Flags, oder Besonderheiten in den *NOTES* oder *EXAMPLES*.

4.1.1 Warnungen zu Packages

Beim Starten von GRASS GIS kann die folgende oder eine ähnliche Warnung auftreten:

```
$ DeprecationWarning: The distutils package is deprecated and slated for removal
in Python 3.12. Use setuptools or check PEP 632 for potential alternatives
from distutils.dir_util import copy_tree
```

Warnungen dieser Art können ignoriert werden. Sie beziehen sich auf Systemkomponenten (z. B. Python Packages), die zukünftig Änderungen erhalten werden. Dies hat keinen Einfluss auf die aktuelle Funktionstüchtigkeit von GRASS.

Speziell obige Warnung sollte bei der Nutzung von GRASS 8 (im Docker) nicht mehr auftreten, da die Kompatibilität bereits gewährleistet wurde. Generell laufen aktuelle GRASS-Versionen mit aktuellen Python-Versionen.

4.1.2 Warnungen beim Installieren von GRASS-Addons

Bei der Installation von GRASS-Addons können solche oder ähnliche Warnungen auftreten:

```
$ g.extension m.analyse.buildings url=/src/grass-gis-addons/m.analyse.buildings
-s
...
WARNING: Unable to create '/usr/local/grass83/docs/html/grassdocs.css':
     '/usr/local/grass83/docs/html/grassdocs.css' and
     '/usr/local/grass83/docs/html/grassdocs.css' are the same file. Is
     the GRASS GIS documentation package installed? Installation
     continues, but documentation may not look right.
Compiling...
Installing...
Updating extensions metadata file...
Updating extension modules metadata file...
WARNING: No metadata available for module 'm.analyse.buildings': Unable to
     fetch interface description for command '<m.analyse.buildings>'.

     Details: <[Errno 2] No such file or directory:
     'm.analyse.buildings'>
Installation of <m.analyse.buildings> successfully finished
```

Die Warnungen zur GRASS GIS-Dokumentation während der Installation können ignoriert werden. Hier geht es um Metadaten und Dokumentation, die Funktion der Addons ist dadurch nicht eingeschränkt. Ob die Installation eines Addons erfolgreich verlaufen ist, kann an der letzten Zeile abgelesen werden.

4.1.3 Warnungen beim Nutzen der GUI

Beim Nutzen der Graphischen Oberfläche (GUI) treten mitunter folgende oder ähnliche Fehler auf:

```
$ (wxgui.py:6500): Gtk-CRITICAL * *: 13:46:27.691: gtk_box_gadget_distribute:
assertion 'size >= 0' failed in GtkScrollbar
$ (wxgui.py:6500): Gtk-CRITICAL * *: 14:38:27.866: gtk_box_gadget_distribute:
assertion 'size >= 0' failed in GtkCheckButton
$ (wxgui.py:6851): Gtk-CRITICAL * *: 11:49:20.118: gtk_box_gadget_distribute:
assertion 'size >= 0' failed in GtkSpinButton
```

Warnungen dieser Art können ignoriert werden. Die GUI funktioniert dennoch normal.

4.1.4 Warnungen zu inkorrekten Grenzen, *collapsed areas*, Flächenzentroiden, etc.

```
$ WARNUNG: Anzahl inkorrekter Grenzen: 8
WARNUNG: Vect_get_point_in_poly_isl(): collapsed area
WARNUNG: Kann den Flächenzentroiden nicht berechnen.
WARNUNG: Vect_get_point_in_poly_isl(): collapsed area
WARNUNG: Kann keinen Zentroid für die Fläche 101268 berechnen.
WARNUNG: Vect_get_point_in_poly_isl(): collapsed area
WARNUNG: Kann keinen Zentroid für die Fläche 201928 berechnen.
WARNUNG: Flächen mit Größe 0.0 ignoriert.
WARNUNG: Anzahl inkorrekter Grenzen: 7
WARNUNG: Die Datenbankverbindung für den Layer<1> ist nicht definiert
```

Treten Warnungen dieser Art beim Import von Daten auf, sollten sie **nicht** ignoriert werden. Bei der anschließenden Analyse hingegen können die Warnungen ignoriert werden, da die genutzten

GRASS-Module, wie. z. B. *v.patch,* in der Regel auch eine topologische Säuberung beinhalten. Das heißt, die Warnungen werden der Vollständigkeit halber ausgegeben und eventuelle Probleme dann direkt behoben. Eine Datenbankverbindung ist für GRASS-Vektorlayer nicht zwingend erforderlich, deswegen kann diese Warnung meist ignoriert werden. Für den Fall, dass Vektor-Attribute fehlen, könnte hier jedoch ein Grund dafür liegen.

4.1.5 Fehler beim Exportieren als Shapefile

Wenn beim Export als Shapefile der folgende Fehler o. ä. auftritt:

```
$ v.out.ogr in=result_buildings out=/pfad/zum/output.shp format=ESRI_Shapefile
ERROR 6: Failed to add field named 'vegetation_proportion'
ERROR: Unable to create column <vegetation_proportion>
```

dann ist der Name des Attributs zu lang und das Attribut muss umbenannt werden, sodass der Spaltenname maximal 10 Zeichen beinhaltet. Diese Vorgabe kommt von den ESRI Shapefile DBF- Tabellen-Spezifikationen. Anschließend kann der Export z. B. mit folgendem Befehl wiederholt werden:

```
# Umbenennen der Spalte zu new_name
$ v.db.renamecolumn map=result_buildings column="vegetation_proportion,new_name"

# Wiederholen des Exports
$ v.out.ogr input=result_buildings output=/pfad/zum/output.shp format=ESRI_Sha-
pefile --o
```

4.1.6 Warnung beim Export: features without category were skipped

```
$ v.out.ogr in=result_buildings out=/pfad/zum/output.gpkg
...
WARNUNG: 14 features without category were skipped. Features without
     category are written only when -c flag is given.
```

Die Warnung kann ignoriert werden. Es ist empfehlenswert, die *-c*-Flag beim Vektor-Export mit *v.out.ogr* nicht zu setzen. Bei Nutzung der *-c*-Flag wird das Ergebnis nicht das Erwartete sein, da in der *Simple Feature Definition* damit Polygone erstellt werden, die es eigentlich gar nicht gibt. Die C*ategory,* als GRASS-Feature ID, ist nicht zu verwechseln mit den Attributen von Features (Polygonen). „*Features without category*“ bezieht sich auf Löcher in Polygonen. Diese haben weder eine eigene ID noch Attribute.

4.1.7 Fehler beim Export der Farbtabelle von Rasterdaten (z. B. DOM, nDOM) als GeoTiff

```
$ r.out.gdal input=ndsm_map output=/pfad/zum/output.tif format=GTiff
Using GDAL data type <Float32>
Die Eingabe-Rasterkarte enthält Zellen mit dem NULL-Wert (no-data). Der
Wert -nan wird verwendet, um NoData-Werte in der Eingabekarte zu
kennzeichnen. Sie können NoData-Wert mit dem Parameter nodata bestimmen.
ERROR 6: /pfad/zu/tiff/nDOM.tif, band 1: SetColorTable() only supported for Byte
or UInt16 bands in TIFF format.
```

Beim Export als GTiff ist es für Raster, die nicht als Byte oder UInt16 Datentyp vorliegen, empfehlenswert, die Farbregeln nicht mit zu exportieren. Dafür kann die *-c*-Flag gesetzt werden. Für maximale Kompatibilität mit anderen GIS-Programmen ist auch die *-m*-Flag empfehlenswert. Die Farbregeln können alternativ mit der *-t*-Flag in eine separate Raster-Attributtabelle geschrieben werden. Informationen zu den Flags finden sich auf der Handbuchseite von r.out.gdal.

Der Export kann z. B. mit folgendem Befehl wiederholt werden:

```
$ r.out.gdal -cmt input=ndsm_map output=/pfad/zum/output.tif format=GTiff --o
```

5 Referenzen

Hier befindet sich eine Übersicht der wichtigsten Referenzen in dieser Dokumentation. Kontaktieren Sie uns gerne bei Fragen oder Problemen.

Kontakt mundialis

Ansprechpersonen:
G. Riembauer riembauer@mundialis.de
J. Haas haas@mundialis.de

GitHub Repository mit Skripten und GRASS-Addons

https://github.com/mundialis/rvr_interface

GRASS GIS

GRASS Website

- GRASS GIS Homepage
- GRASS GIS Download

GRASS GIS Manual

- GRASS GIS Manual
- GRASS Basics im Manual

GRASS GIS Wiki

- GRASS GIS Wiki
- GRASS User Wiki Installation Guide

Tutorials

- GRASS GIS Workshop
- Analysing environmental data with GRASS GIS