Hochschule München University of Applied Sciences

LANDESAMT FÜR DIGITALISIERUNG, BREITBAND UND VERMESSUNG

Bachelorarbeit

Konzeption und Implementierung eines Geodatenkonverters in Python für Open Data

Fakultät für Geoinformation

Studiengang Geoinformatik und Navigation

vorgelegt von

Paul Bürger

Betreuer: Prof. Dr. Ludwig Hoegner

In Kooperation mit: Landesamt für Digitalisierung, Breitband und Vermessung Bayern

Betreuer: Dipl. Ing. (FH) Thomas Meier (LDBV, Referat 85)

Wintersemester 2024/2025

Matrikelnr.: xxxxxxxxx

München, den 07.03.2025

Zusammenfassung

Diese Arbeit befasst sich mit der Konzeption und Implementierung eines Geodatenkonverters in der Programmiersprache Python, der speziell für Open Data entwickelt wurde. Ziel ist es, eine Software zu schaffen, die gängige Geodatenformate (Raster- und Vektordaten) sicher erkennt und mit verschiedenen Aktionen prozessiert, um eine nahtlose Weiterverarbeitung zu ermöglichen. Hierbei kommen verschiedene Bibliotheken wie Rasterio und GDAL zum Einsatz, die zwar umfangreiche Funktionen bieten, deren Integration in eine lauffähige Anwendung jedoch mitunter problematisch ist.

Das Ergebnis ist eine zweigeteilte Anwendung, die sowohl im „Manuellen Vorgang“ als auch im „Automatischen Vorgang“ agiert. Der manuelle Vorgang bietet Nutzerinnen und Nutzern maximale Flexibilität in der Dateiauswahl und Aktionserstellung (z. B. Konvertierung, Merging, Koordinatentransformation). Der automatische Vorgang zielt auf weniger erfahrene Anwenderinnen und Anwender ab, indem häufig genutzte Abläufe (etwa Konvertieren von digitalen Geländemodellen oder das Zusammenführen von digitalen Orthophotos) automatisiert durchlaufen werden.

Neben klassischen Formaten wie GeoTIFF, Esri-ASCII, sowie GeoJSON und KML werden auch 3D-Formate (CityGML, OBJ, DXF) unterstützt, um unter anderem die Weiterverarbeitung in Modellierungsprogrammen wie Blender zu vereinfachen. Besonders in diesem Bereich wurden allerdings noch einige Einschränkungen festgestellt, beispielsweise bei der Übernahme von Texturen oder bei komplexen Koordinatentransformationen. Ebenso sind Stand-Alone Lösungen unter Windows noch nicht vollständig zuverlässig, vor allem bedingt durch die schwierige Einbindung der GDAL-Bibliothek.

Die in diversen Testumgebungen gemessenen Performancewerte zeigen, dass die meisten Schritte rasch und mit verhältnismäßig geringem Arbeitsspeicherbedarf durchgeführt werden können. Nur das Zuschneiden großer Raster (ab 20.000 x 20.000 Pixel) erfordert längere Laufzeiten und etwas mehr Arbeitsspeicher. Insgesamt lassen die Ergebnisse darauf schließen, dass der entwickelte Geodatenkonverter eine schnelle und benutzerfreundliche Lösung für alltägliche Aufgaben in der Geodatenverarbeitung darstellt. Um das Tool noch breiter einsatzfähig zu machen, sollten allerdings vor allem die Bereiche GDAL-Integration und 3D-Texturierung weiter optimiert werden.

Schlagwörter: Geodaten Bayern, Python, Konvertierung, Transformation, Software, Open Source, Open Data

Abstract

This thesis focuses on the conception and implementation of a Python-based geodata converter, specifically designed for open data applications. The main goal was to develop a software tool that reliably detects and process geodata. Key libraries like Rasterio and GDAL were utilized, offering extensive features but posing certain challenges when integrated into a fully functional application.

The final tool is split into two operation modes: “Manueller Vorgang” and “Automatischer Vorgang”. The manual mode offers maximum flexibility by allowing users to select specific files and choose from various actions (e.g., converting, merging or coordinate transformation). On the other hand, the automatic mode is tailored to less experienced users by running frequently used workflows (such as converting digital elevating models or merging digital orthophotos) without requiring extensive configuration.

In addition to classic formats like GeoTIFF and Esri-ASCII, the converter also supports 3D formats (CityGML, DXF, OBJ), aiming to simplify further processing in modeling programs such as Blender. However, some limitations were identified in this area, particularly regarding texture handling and more complex coordinate transformations. Similarly, creating a stable stand-alone version for Windows remains a challenge, primarily due to difficulties in integrating the GDAL library.

Performance measurements in various test environments indicate that most operations run quickly and with relatively low requirements. Only cropping a very large raster (at 20.000 x 20.000 pixel) leads to slightly higher run times and increased RAM usage. Overall, the results suggest that the developed geodata converter represents a fast and user-friendly solution for day-to-day geodata tasks. Still, improvements in GDAL integration and 3D texture handling are crucial to make the tool even more robust and versatile in a broader range of applications.

Inhaltsverzeichnis

1 Einleitung

Die Motivation für diese Arbeit liegt darin, die Hürden bei der Verarbeitung von Geodaten zu reduzieren und so einen Beitrag zur breiteren Anwendung zu leisten. Durch die Nutzung von Open Data, insbesondere der Daten des Landesamtes für Digitalisierung, Breitband und Vermessung Bayern (LDBV), wird gezeigt, wie öffentlich verfügbare Datenquellen effektiv transformiert und weiterverwendet werden können (LDBV, 2021). Dies unterstützt nicht nur Fachanwenderinnen und Fachanwender in Wirtschaft und Forschung, sondern fördert auch die Entwicklung innovativer Lösungen in verschiedenen Sektoren.

In der heutigen digitalen Zeit spielen Geodaten in zahlreichen Anwendungsbereichen eine große Rolle. Mit der fortschreitenden Digitalisierung und der Verfügbarkeit von Open Data Portalen stehen riesige Mengen an Geodaten zur Verfügung, die weit über die traditionellen Anwendungen in Karten hinausgehen. Geodaten bilden die Grundlage für fortschrittliche Technologien und Dienstleistungen in einer Vielzahl von Sektoren. Sie ermöglichen es, präzise und detaillierte Visualisierungen von Landschaften, Städten und Umgebungen zu erstellen, die für Benutzerinnen und Benutzer von großem Wert sind. Sie dienen als Basis für räumliche Analysen um Muster und Trends zu erkennen, die in herkömmlichen Datenformaten unsichtbar wären. Diese vielseitigen Einsatzmöglichkeiten eröffnen innovative Lösungen in Bereichen wie Umweltüberwachung, Stadtplanung und Navigationssystemen (Goodchild, 2007, S.214-217; Longley u. a., 2015, S.2).

Um das volle Potenzial dieser Daten auszuschöpfen, ist eine genaue Vorverarbeitung notwendig. Die Verarbeitung und Aufbereitung von Geodaten stellt eine zentrale Herausforderung dar, insbesondere wenn es um die Umwandlung zwischen verschiedenen Datenformaten und Koordinatenbezugssystemen geht (Steiniger und Hunter, 2013, S.137-142). Hierfür werden flexible und leistungsfähige Werkzeuge benötigt, die den unterschiedlichen Anforderungen gerecht werden.

An diesem Punkt setzt die vorliegende Arbeit an. Ziel ist es, ein einfaches und kostenfreies Tool zu entwickeln, das die sinnvolle Konvertierung von Geodaten in andere Formate und Koordinatenbezugssysteme ermöglicht. Auf Basis von Open Source Bibliotheken und der Programmiersprache Python wurde ein Programm erstellt, das die Vorverarbeitung von Geodaten erleichtert und sie für die genannten Anwendungsbereiche nutzbar macht. Durch dieses Werkzeug wird der Prozess der Datenaufbereitung vereinfacht, was wiederum die Nutzung der Daten in verschiedenen Projekten fördert.

Das zweite Kapitel gibt einen Überblick über die theoretischen Grundlagen. Zuerst wird erläutert, welche Geodatenformate existieren und warum sie sich zum Teil stark unterscheiden. Anschließend wird beleuchtet, wie Python und relevante Bibliotheken (z. B. Geopandas oder Rasterio) den Kern für die Anwendungen bilden.

Kapitel 3 widmet sich der eigentlichen Methodik und Implementierung. Dabei wird der grundlegende Programmaufbau skizziert, die grafische Benutzeroberfläche (GUI) vorgestellt und erklärt, wie die Verarbeitungsschritte – vom Konvertieren über das Zusammenführen bis hin zum Zuschneiden von digitalen Ortskarten/ digitalen topographischen Karten (DOK/DTK-Dateien) – intern ablaufen.

In Kapitel 4 werden die Ergebnisse dieser Umsetzung beschrieben. Neben einer Performance Analyse, die zeigt wie schnell und ressourcenschonend der Konverter verschiedene Aufgaben erledigt, enthält es auch eine Fehlerbehandlung und zeigt Anwendungsbeispiele aus der Praxis. Abschließend werden im Vergleich mit anderen Tools (z. B. FME, QGIS FZKViewer) die Stärken und Grenzen der eigens entwickelten Lösung aufzeigt.

Aufbauend auf diesen Erkenntnissen folgt in Kapitel 5 die Erweiterung der Forschung und Diskussion. Dort werden die wichtigsten Ergebnisse reflektiert und zentrale Verbesserungsmöglichkeiten erläutert, bevor die Arbeit mit einem zusammenfassenden Fazit und einem Ausblick auf Weiterentwicklungsmöglichkeiten endet.

2 Theoretische Grundlagen

2.1 Stand der Technik

Nachdem die zentrale Bedeutung von Geodaten in verschiedenen Anwendungsbereichen hervorgehoben wurde, stellt sich die Frage, wie diese Daten effizient gespeichert und verarbeitet werden können.

Geodaten werden in verschiedenen Formaten gespeichert, darunter Rasterformate wie GeoTIFF, Esri-ASCII oder Vektorformate wie Shapefile, GeoJSON oder KML. Diese Formate haben spezifische Anwendungsbereiche: Während Rasterdaten kontinuierliche Phänomene wie Höhenmodelle oder Satellitenbilder repräsentieren, eignen sich Vektordaten für diskrete Objekte wie Straßen oder Grundstücke (Wilson, 2008, S. 1, 19). Die Konvertierung zwischen diesen Formaten ist notwendig, da unterschiedliche GIS-Systeme oft nur ein bestimmtes Format unterstützen oder Projekte verschiedene Datensätze aus heterogenen Quellen zusammenführen müssen. Die Harmonisierung der Formate ist nötig für eine reibungslose Verarbeitung und Analyse.

Eine der meistgenutzten Technologien für Geodatenkonvertierungen ist GDAL (Geospatial Data Abstraction Library) mit ihrem Vektorpendant OGR. GDAL unterstützt zahlreiche Raster- und Vektorformate und bietet Funktionen wie Zuschnitt, Resampling und Transformation. OGR ermöglicht zusätzlich die Konvertierung zwischen Vektorformaten wie Shapefile und GeoJSON (GDAL Developers, o. J.). Dank der Python-Implementierung können Entwicklerinnen und Entwickler auf GDAL/OGR zugreifen, um Konvertierungsaufgaben in automatisierten Workflows zu integrieren. Die Python-Bibliothek Rasterio, die auf GDAL basiert und Fiona, die OGR-Funktionen bereitstellt, sind wesentliche Werkzeuge um Geodaten einfach und effizient zu verarbeiten (GDAL Developers, o. J.).

Datums-Transformation

Neben der Konvertierung der Datenformate spielt auch die Transformation eine entscheidende Rolle für die Genauigkeit und Interoperabilität von Geodaten.

Koordinatensysteme sind essenziell für die präzise Beschreibung und Lokalisierung von Positionen auf der Erdoberfläche. Sie ermöglichen nicht nur die konsistente Erfassung und Verarbeitung geografischer Informationen, sondern dienen auch als Basis für die mathematische Analyse von Beziehungen zwischen Punkten und die Lösung geometrischer Probleme. Durch einheitliche Referenzsysteme wird die genaue Navigation, Kartenerstellung und Vermessung ermöglicht, während die Vielfalt an Projektionen und geometrischen Ansätzen dazu beiträgt, unterschiedliche Anforderungen und Herausforderungen in der Kartografie zu bewältigen (Maling, 1992, S. 27-28).

Das Universale Transversale Mercator-System (UTM) ist ein kartesisches Koordinatensystem, das die Erde in 60 Zonen unterteilt. Jede Zone wird separat projiziert, wodurch Verzerrungen minimiert und eine hohe Genauigkeit in regionalen Anwendungen erreicht wird. UTM ist besonders nützlich für topografische Karten und regionale Vermessungsprojekte, da es eine einfache Berechnung von Entfernungen und Flächen ermöglicht (Sneyder 1987, S. 57-60).

Das World Geodetic System 1984 (WGS84) ist das globale Referenzsystem, das als Grundlage für GPS verwendet wird. Da WGS84 ein geozentrisches System ist, während lokale Systeme wie UTM oft nicht geozentrisch ausgerichtet sind, sind Transformationen zwischen diesen Systemen unverzichtbar, insbesondere für Anwendungen in Europa. Solche Transformationen ermöglichen die Umrechnung globaler geodätischer Daten in regionale Koordinatensysteme, die für Navigation, Kartografie und Vermessung wichtig ist. Für Europa wird betont, dass die Genauigkeit dieser Transformationen durch den Einsatz moderner Methoden und Modelle gewährleistet wird, um präzise Projektionen und Karten zu erstellen (Hofmann-Wellenhof, Lichtenegger, und Collins 1997, S. 233-237, S. 247-249).

Daten-Merging

Nach dem die Datenformate harmonisiert und die Koordinatensysteme transformiert wurden, ist das Zusammenführen dieser Datenquellen der nächste Schritt, um Analysen zu ermöglichen.

Das Zusammenführen (Merging) von Geodaten ist ein grundlegender Schritt in der Verarbeitung und Analyse geografischer Daten, insbesondere im Kontext von Anwendungen wie der Navigation oder der Geovisualisierung. Durch Merging können Rasterdaten verschiedener Quellen kombiniert werden, um eine einheitliche Datenbasis zu schaffen, die für umfassende räumliche Analysen erforderlich ist. Die zunehmende Verfügbarkeit von geografischen Daten, insbesondere durch Fortschritte in der Fernerkundung und Sensortechnik, führt dazu, dass effiziente Werkzeuge zur Datenverarbeitung immer wichtiger werden (Silva-Coira u. a. 2020, S. 1-2).

Das in diesem Projekt entwickelte Merging-Tool nutzt die Python-Bibliothek Rasterio, die wichtige Funktionen für die Verarbeitung von Rasterdaten bereitstellt. Wissenschaftliche Arbeiten zeigen, dass durch den Einsatz moderner Algorithmen, wie beispielsweise des k2-Rasters, nicht nur der Speicherbedarf reduziert, sondern auch die Verarbeitungszeit signifikant verbessert werden kann (Silva-Coira u. a. 2020, S. 3-4). Besonders dabei ist, dass der k2-Raster eine komprimierte Darstellung von Rasterdaten ermöglicht, die Abfragen direkt auf komprimierten Daten ausführt und so eine hohe Speicher- und Rechenzeitnutzung gewährleistet (Silva-Coira u. a. 2020, S. 9). Solche Technologien sind entscheidend, um große Datenmengen zusammenzuführen und die Grundlage für detaillierte Analysen zu schaffen.

Die problemlose Integration von Open Data ist nur möglich, wenn die zuvor diskutierten Aspekte der Datenkonvertierung und -transformation berücksichtigt und gleichzeitig qualitativ hochwertige Datenquellen bereitgestellt werden, die als Grundlage für leistungsfähige Geodatenkonverter dienen.

Grundlagen der Open Data Integration

Die Grundlage für Open Data stellt die öffentliche Verwaltung vor die Herausforderung, Datenqualität und Interoperabilität sicherzustellen. Ein zentraler Aspekt ist die Bereitstellung der Daten in maschinenlesbaren Formaten und über offene Lizenzen, um eine reibungslose Weiterverwendung zu ermöglichen. Metadaten spielen dabei eine entscheidende Rolle, da sie die Auffindbarkeit und die Nachvollziehbarkeit der Datensätze gewährleisten. Das Datenportal *GovData.de* ist ein Beispiel für eine zentrale Plattform, die Metadaten verschiedener Behörden bündelt und den Zugang zu Open Data erleichtert. Einheitliche Standards und klare technische Vorgaben sind eine Voraussetzung, um die Integration offener Daten in bestehende Prozesse zu erleichtern und eine einheitliche Datenqualität zu gewährleisten (Bundesverwaltungsamt 2024, S. 11-20).

Beispiele für Open Data Integration in anderen Ländern

Internationale Beispiele verdeutlichen, wie erfolgreich Open Data genutzt werden kann. Frankreich hat eine umfassende Open Data Strategie implementiert, die durch die Webseite *data.gouv.fr* unterstützt wird. Diese Plattform bietet Funktionen zur Interaktion zwischen Datenbereitstellern und Nutzern, sowie Tools zur Datenqualitätssicherung. Irland fokussiert sich mit *data.gov.ie* auf die Bereitstellung von offenen Daten in hoher Qualität und stellt technische Rahmenwerke zur Verfügung, die Standards für Metadaten, Lizenzen und Formate definieren. Spanien verfolgt mit seiner Initiative „Aporta“ einen ganzheitlichen Ansatz, der sowohl nationale als auch lokale Datenportale integriert, um die Interoperabilität zwischen verschiedenen Verwaltungsebenen zu verbessern (Publications Office of the European Union., Capgemini Invent., und European Data Portal. 2020, S. 7-15).

Auch in Deutschland gibt es zentrale Portale wie zum Beispiel das *open.bydata* vom Bayrischen Staatsministerium für Digitales, welches mithilfe der Bayrischen Agentur für Digitales entwickelt wurde. Sie setzen dabei auf das Open Source Datenmanagementsystem „piveau“ von Fraunhofer FOKUS. Das Portal ermöglicht es jeder Verwaltungsstelle in Bayern, eine eigene Open Data Präsenz innerhalb der Plattform zu nutzen. Ein Beispiel ist die Stadt Augsburg. Das Portal wurde Anfang 2023 als Minimum Viable Product (MVP/minimal brauchbares Programm) gestartet und kontinuierlich weiterentwickelt. Im Mai 2023 ging eine überarbeitete Version online. Die Programmierung erfolgt agil, wobei Nutzerfeedback in die neuen Versionen mit einfließt („open bydata (byte)“, o. J.). Mit *open.bydata* sollen Informationen von Staat und Behörden, die zuvor dezentral verteilt waren, gebündelt und für die Allgemeinheit zugänglich gemacht werden („Open Data Bayern: Eigene Präsenz für Kommunen“, o. J.).

Kontextvisualisierung für den Geodatenkonverter

Die in Deutschland und anderen Länder gesammelten Erfahrungen mit der Open Data Integration liefern wertvolle Erkenntnisse für die Entwicklung eines Geodatenkonverters. Ähnlich wie bei allgemeinen Open Data Portalen ist es entscheidend, dass Geodaten in interoperablen Formaten bereitgestellt werden und Metadaten gemäß etablierten Standards gepflegt werden. Nationale Portale wie *GovData.de* oder länderspezifische Portale wie *Open Data Bayern* könnten als Vorbilder dienen, um die Integration und Weiterverarbeitung von Geodaten zu erleichtern. Internationale Ansätze, wie sie in Frankreich, Spanien und Irland praktiziert werden, bieten weitere Anregungen, insbesondere im Hinblick auf Datenqualität, Benutzerfreundlichkeit und die Zusammenarbeit zwischen verschiedenen Verwaltungsapparaten (Publications Office of the European Union., Capgemini Invent., und European Data Portal. 2020; Bundesverwaltungsamt 2024; „open bydata (byte)“, o. J.).

Verarbeiten großer Rasterdaten

Das Zuschneiden großer Daten in kleinere, handhabbare Ausschnitte ist ein wichtiger Prozess in der Geoinformatik, der als Tile Processing oder Raster Tiling bezeichnet wird. Dieser Ansatz ermöglicht es, umfangreiche Rasterdatensätze in kleinere Kacheln zu unterteilen, wodurch die Daten effizienter gespeichert, abgerufen und verarbeitet werden können (Olasz, Nguyen Thai, und Kristóf, 2016, S. 111).

Ein Beispiel für die Anwendung dieses Verfahrens ist die Studie von Olasz et al. (2016). Die Autoren präsentieren auf dem IQLib-Konzept basierendes System, das im Rahmen des IQmulus EU FP7 – einem vierjährigen Forschungsprojekt des siebten Forschungsrahmenprogramms der europäischen Union – entwickelt wurde. Dieses System ermöglicht die verteilte Verarbeitung großer Datenmengen, indem es Daten in kleinere Kacheln unterteilt und diese parallel verarbeitet. Die Studie demonstriert die Effektivität dieses Ansatzes anhand einer Fallstudie zur landesweiten Verarbeitung von Rasterbildern (Olasz, Nguyen Thai, und Kristóf, 2016, S. 116-117).

Für die Implementierung solcher Prozesse in Python stehen Bibliotheken wie Rasterio zur Verfügung, die auf der GDAL-Bibliothek basieren und Funktionen für das Lesen, Schreiben und Verarbeiten von Rasterdateien bieten. Mit Rasterio können Entwicklerinnen und Entwickler Rasterdaten in kleinere Kacheln unterteilen und diese für verschiedene Anwendungen vorbereiten (Rasterio, o. J.).

2.2 Grundlagen zu Geodaten und Formaten

Geodaten sind Daten mit räumlichem Bezug, die in Vektor- und Rasterformaten vorliegen können. Vektordaten repräsentieren geografische Merkmale als separate Einheiten, während Rasterdaten kontinuierliche Phänomene darstellen (Weidmann und Gleditsch 2020, S. 3). Die Wahl zwischen Vektor- und Rasterdaten hat unterschiedliche Auswirkungen auf die Analyse und Verarbeitung in geografischen Informationssystemen (GIS) (Renz 2014, S. 3). Vektorformate eignen sich besonders für die Darstellung diskreter Objekte, wie Straßen oder Gebäude, während Rasterformate kontinuierliche Phänomene, wie Höhenmodelle oder Satellitenbilder, repräsentieren (Weidmann und Gleditsch 2020, S. 5). Diese grundlegende Unterscheidung zwischen diskreten Objekten und kontinuierlichen Feldern bildet das zentrale Konzept der GIScience. Goodchild (2010, S. 11) beschreibt, dass diese Ansätze sowohl auf Raster- als auch auf Vektorstrukturen abgebildet werden können, jedoch unterschiedliche Anforderungen und Herausforderungen mit sich bringen. So erfordert die Modellierung und Speicherung von Geodaten eine sorgfältige Auswahl des passenden Datenformats, um spezifische Anwendungsanforderungen, wie Interoperabilität und Effizienz, zu gewährleisten.

Im Bereich detaillierter 3D-Stadt- und Landschaftsmodelle hat sich das CityGML-Format etabliert, das vom Open Geospatial Consortium (OGC) spezifiziert wurde und insbesondere für die modellhafte Darstellung von Gebäuden, Straßen und sonstiger Infrastruktur genutzt wird (Gröger u. a. 2012,S. xiv/9). Für Rasterdaten, beispielsweise digitale Orthophotos (DOP), ist das GeoTIFF-Format weit verbreitet, da es zusätzlich zur Bildinformation auch Metadaten zum Koordinatensystem speichert (GDAL o. J.).

GeoJSON und KML sind besonders für webbasierte Anwendungen bedeutsam, da sie spezifische Anforderungen an die Darstellung und den Austausch geografischer Daten erfüllen (Wilson 2008, S. 3). GeoJSON basiert auf der JSON-Struktur und ermöglicht eine nahtlose Integration in Webframeworks, was es ideal für die Visualisierung und Verarbeitung geografischer Daten in RESTful APIs macht. Tests mit GeoJSON im Rahmen eines urbanen Informationssystems haben gezeigt, dass GeoJSON-Daten effizient verarbeitet werden können, obwohl sie im Vergleich zu anderen Formaten wie SIKEL (das sind Bevölkerungsdaten innerhalb der REST APIs) eine höhere Ladezeit aufweisen (38,4 Sekunden im Vergleich zu 14,7 Sekunden bei 100-500 Anfragen) (Rahmatulloh u. a. 2022, S. 6). KML hingegen, das ursprünglich von Google entwickelt wurde, ist für die Nutzung in Anwendungen wie Google Earth oder ähnlichen Plattformen optimiert (Wilson 2008, S. 3). Es basiert auf XML und bietet erweiterte Funktionen zur geografischen Visualisierung, einschließlich der Steuerung von Benutzeransichten und der Darstellung komplexer Geometrien. Aufgrund seiner breiten Unterstützung durch Plattformen und die Möglichkeit der Komprimierung der Dateien eignet sich KML hervorragend für die Darstellung umfangreicher Datensätze (Wilson 2008, S. 19).

Beide Formate haben spezifische Stärken: GeoJSON überzeugt durch seine einfache Integration in webbasierte Workflows und APIs, während KML mit seinen erweiterten Visualisierungs- und Steuerungsfunktionen in 3D-Anwendungen punktet (Wilson 2008, S. 19; Rahmatulloh u. a. 2022, S. 6).

2.3 Technische Basis: Python und relevante Bibliotheken

Python hat sich als eine der zentralen Programmiersprachen für GIS-Anwendungen etabliert, da es eine klare Syntax, eine Vielzahl spezialisierter Bibliotheken sowie eine große Community besitzt. Im Folgenden werden zentrale Bibliotheken vorgestellt, die in diesem Kontext eine wichtige Rolle spielen. Alle genannten Bibliotheken sind Open Source und werden aktiv von der GIS-Community weiterentwickelt. Dies ermöglicht eine kontinuierliche Verbesserung und Anpassung an neue Anforderungen in der Geodatenverarbeitung. Der offene Quellcode erlaubt es zudem, bestehende Funktionen zu erweitern und auf spezifische Bedürfnisse zuzuschneiden.

Geodatenverarbeitung und räumliche Analysen

Eine der bedeutendsten Bibliotheken für die Verwaltung und Verarbeitung von räumlichen Datenformaten ist GDAL (Geospatial Data Abstraction Library). Sie gilt als Standard für das Lesen, Schreiben und Konvertieren von Raster- und Vektordaten und wird in vielen GIS-Anwendungen verwendet (GDAL Developers, o. J.). Aufbauend auf dieser Basis bietet Rasterio eine intuitive API zur Verarbeitung von Rasterdaten, insbesondere GeoTIFFs. Da Rasterio auf GDAL basiert, ermöglicht es effizientes Zuschneiden, Maskieren und Transformieren von Geländemodellen oder Satellitenbildern. Dadurch lässt es sich nahtlos in größere GIS-Workflows integrieren (Rasterio, o. J.).

Für die Verarbeitung und Analyse von Vektordaten ist Geopandas von zentraler Bedeutung. Diese Bibliothek erweitert die Pandas-Datenstruktur um geometrische Datentypen, sodass GIS-Basisfunktionen direkt in Python verfügbar sind. Geopandas nutzt Shapely zur Manipulation und Analyse von Geometrien. Dadurch können geometrische Operationen wie Schnittmengen, Vereinigungen oder Pufferzonen einfach durchgeführt werden. Das macht Geopandas in Kombination mit Shapely zu einem wichtigen Werkzeug für Vektordaten (Geopandas, o. J.; Shapely, o. J.).

Koordinatentransformation und Projektionen

Ein zentraler Bestandteil der GIS-Entwicklung ist die Transformation geografischer Koordinaten. Pyproj bietet Werkzeuge zur präzisen Umrechnung zwischen verschiedenen Koordinatensystemen, basierend auf der PROJ-Bibliothek. Besonders bei der Verarbeitung heterogener Geodatenquellen ist die einheitliche Transformation nicht zu vernachlässigen, um genauen Analysen und Darstellungen zu gewährleisten (Pyproj, o. J.).

Numerische Berechnungen und Datenverarbeitungen

Für numerische Berechnungen in GIS-Anwendungen ist NumPy ideal. Die Bibliothek ermöglicht schnelle Berechnungen großer Datensätze und wird besonders für die Rasterdatenverarbeitung sowie mathematische Operationen auf Geodatenarrays genutzt. Sie harmoniert gut mit anderen wissenschaftlichen Bibliotheken wie Rasterio und GDAL (NumPy, o. J.). Ergänzend dazu dient Pandas als Bibliothek für tabellarische Datenstrukturen. In GIS-Projekten wird sie vor allem zur Verwaltung von Attributdaten verwendet, die oft mit geometrischen Informationen kombiniert werden. Die nahtlose Integration mit Geopandas erleichtert die Analyse und Filterung großer Datensätze deutlich (Pandas, o. J.).

Grafische Benutzeroberfläche

Für die Entwicklung benutzerfreundlicher grafischer Oberflächen in Python wird häufig PyQT eingesetzt. Diese Bibliothek schafft es, leistungsfähige und flexible GUI-Anwendungen zu erstellen. Durch das Signal-Slot-Konzept lassen sich asynchrone Prozesse realisieren, sodass das Hauptfenster nicht blockiert wird, während Berechnungen oder Datenverarbeitungen im Hintergrund laufen (PyQT, o. J.).

2.4 Fazit

Geodaten liegen in verschiedenen Formaten vor, die spezifische Anwendungsfälle bedienen. Rasterformate wie GeoTIFF repräsentieren kontinuierliche Phänomene, während Vektordaten wie GeoJSON diskrete Objekte darstellen. Die Konvertierung zwischen diesen Formaten und die Transformation von Koordinatensystemen, insbesondere zwischen WGS84 und UTM32N, sind essenziell für GIS-Anwendungen.

Technologien wie GDAL, Rasterio und Python-Bibliotheken wie Geopandas und PyQt5 bilden die technische Grundlage für die Datenverarbeitung und die Entwicklung einer benutzerfreundlichen Oberfläche. Moderne Ansätze wie komprimierte Datenstrukturen optimieren zudem Speicher- und Rechenzeit. Insgesamt schafft diese eine solide Grundlage für die Entwicklung eines leistungsfähigen Geodatenkonverters.

3 Methodik und Implementierung

Der Geodatenkonverter wurde in der Sprache Python implementiert, da zahlreiche GIS-Bibliotheken wie Geopandas, Rasterio und PyQT auf ihr basieren. Diese Bibliotheken bieten Funktionen, die die Verarbeitung und den Umgang mit Geodaten erleichtern. Im Folgenden wird auf die Struktur des Programms sowie die Methodik der wichtigsten Funktionen eingegangen.

3.1 Programmlogik und Aufbau

Das Programm gliedert sich in drei Hauptkomponenten: die grafische Benutzeroberfläche (GUI), die Verarbeitungslogik und die Hilfsfunktionen. Die GUI wurde mit PyQT erstellt und ermöglicht die Auswahl von Dateien oder ganzen Ordnern, die Definition von Aktionen und die Speicherung der Ergebnisse. Die Verarbeitungslogik enthält Module für Konvertierungen, Transformationen, das Zusammenführen (Merging) und den Zusammenschnitt von Daten. Die Hilfsfunktionen unterstützen bei der Erkennung von Dateiformaten und Koordinatensystemen.

Die Kommunikation zwischen den Modulen erfolgt über klare Schnittstellen. So werden beispielweise nach der Dateiauswahl das Format sowie das Koordinatensystem der Dateien erkannt. Abhängig davon werden kompatible Aktionen angezeigt, welche durchgeführt werden können. Die Ergebnisse der Auswahl der Aktionen, werden schließlich in der GUI dargestellt und können mit einem vorgeschlagenen oder selbstgewählten Namen in einer neuen Datei gespeichert werden. Diese Struktur sorgt für Modularität und erleichtert die Erweiterung des Programms.

In Abbildung 1 wird ein kurzer Überblick über den Datenverarbeitungsprozess mithilfe eines Flussdiagramms veranschaulicht.

Überblick über den Datenverarbeitungsprozess:

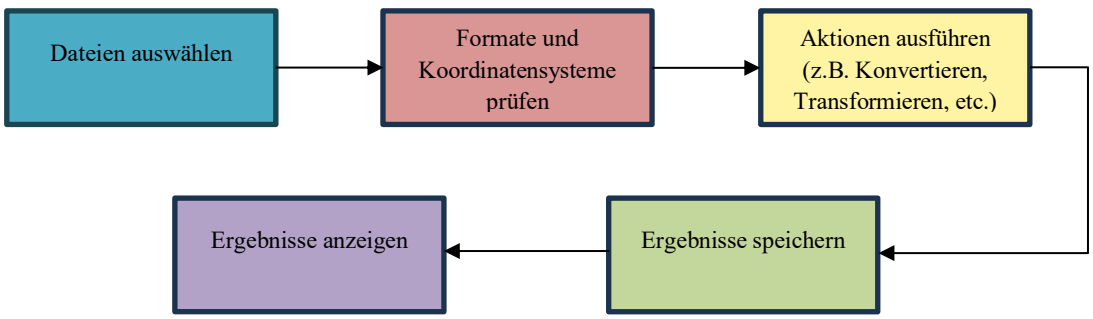

Abbildung 1: Flussdiagramm des Durchgangsprozesses

3.2 Grafische Benutzeroberfläche (GUI)

Die grafische Benutzeroberfläche ermöglicht es der Nutzerin oder dem Nutzer, den gesamten Verarbeitungsprozess interaktiv zu steuern. Sie enthält Buttons für die Dateiauswahl, Dropdown-Menüs zur Auswahl der gewünschten Aktionen und eine Fortschrittsanzeige. Die Steuerung der Aktionen erfolgt über Signal-Slot-Mechanismen. Diese erlauben es längere Prozesse wie das Zusammenführen (Merging) oder die Transformation auszuführen, ohne die Benutzeroberfläche zu blockieren.

Ein Beispiel für den Workflow: Nach dem Auswählen und Hochladen der Dateien erkennt das Programm automatisch das Format und das Koordinatensystem der eingespeisten Informationsquelle. Anschließend wählt die Nutzerin oder der Nutzer im Programm eine Aktion aus einer Auswahl, die in der grafischen Benutzeroberfläche angezeigt wird. Beispielsweise die Konvertierung von GeoTIFF nach ASCII. Die Nutzerin oder der Nutzer kann die von ihm gewünschte Aktion ausführen lassen und die Ergebnisse an einen von ihm gewählten Zielort abspeichern. Dieser Ansatz ist benutzerfreundlich und macht die komplexen Prozesse zugänglicher.

3.3 Verarbeitungslogik

Die Kernfunktionen des Programms umfassen Konvertierungen, Transformationen von Koordinatensystemen, das Zusammenführen von Raster- und Vektordaten sowie den Zuschnitt von DOK-/DTK-Dateien. Jede dieser Funktionen ist modular aufgebaut, sodass sie unabhängig voneinander genutzt werden können. Nachfolgend werden diese Funktionen beschrieben:

3.3.1 Konvertierungen

Die Konvertierungsoption ermöglicht es, Formate wie beispielsweise GeoTIFF, KML oder CityGML in andere Formate wie Esri-ASCII, GeoJSON oder OBJ zu überführen. Ziel dabei ist es, die Daten für spezifische Software oder Workflows kompatibel zu machen. Beispielsweise können GeoTIFF-Daten in das ASCII-Format umgewandelt werden, um Höheninformation in Programmen wie Blender für Geovisualisierungen zu nutzen. Die Konvertierung wird durch die Bibliothek Rasterio unterstützt. Diese liest, schreibt und verarbeitet effizient Daten.

Ein übliches Anwendungsbeispiel ist die Umwandlung eines digitalen Geländemodells im GeoTIFF-Format in ein Esri-ASCII Format. Dabei werden Rasterdaten aus der Startdatei gelesen, in ein textbasiertes Format überführt und als Ausgabedatei gespeichert. Die Konvertierung in dieses Format wird für den weiteren Workflow zwingend benötigt. Der Prozess läuft automatisiert, sodass die Benutzerin oder der Benutzer lediglich Eingabedatei und Ausgabeort angeben muss.

Der Pseudocode für eine Konvertierung könnte so aussehen:

```
FUNKTION geotiff_to_ascii(input_datei, output_datei):
        Öffne GeoTIFF-Datei
        Lese Dimensionen und Rasterdaten
        Konvertiere Rasterdaten in Esri-ASCII-Format
        Schreibe Ergebnisse in die Ausgabedatei
ENDE
```

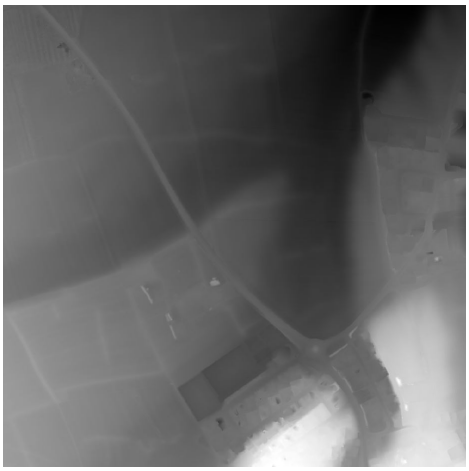

Abbildung 2: DGM als GeoTIFF

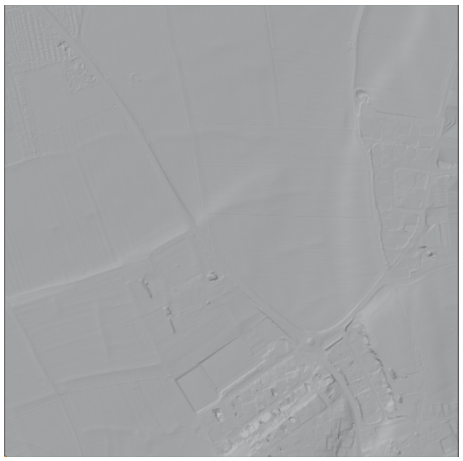

Abbildung 3: DGM als Esri-ASCII

3.3.2 Koordinatentransformation

Die Transformation von Koordinatensystemen (CRS) ist eine der zentralen Herausforderungen bei der Geodatenverarbeitung. Daten unterschiedlicher Quellen liegen häufig in verschiedenen Koordinatensystemen vor. Dies erschwert die gebündelte Nutzung oder die Analyse der Daten. Der Geodatenkonverter unterstützt die Umwandlung zwischen den Koordinatensystemen EPSG:25832 (UTM32N) und EPSG:4326 (WGS84).

Bei einer Transformation werden die vorhandenen Rasterdaten auf das gewünschte Koordinatensystem angepasst. Dies beinhaltet die Veränderung der Geometrie sowie die Anpassung von Bounds und Auflösung. Ein typisches Szenario ist die Transformation einer Karte im UTM-System in WGS84, um diese in einer globalen Web-GIS-Anwendung einzusetzen. Die Umsetzung erfolgt über Rasterio. Diese Bibliothek führt die notwendigen mathematischen Operationen präzise durch.

Der Pseudocode für eine Transformation wie in obigen Beispiel sieht so aus:

```
FUNKTION transform_geotiff(input_datei, output_datei, ziel_crs):
        Öffne Eingabedatei
        Lese Metadaten (Koordinatensystem, Begrenzungen, Dimensionen)
        Berechne Transformationsparameter für das Ziel Koordinatensystem
        Wende Transformation auf Rasterbänder an
        Schreibe transformierte Daten in die Ausgabedatei
ENDE
```

Abbildung 4: GeoTIFF in UTM32N

Abbildung 5: GeoTIFF in WGS84

Abbildungen 4 und 5 zeigen eine Transformation von UTM32N zu WGS84, zu erkennen an der markanten Ausrichtung der Datei.

3.3.3 Zusammenführung (Merging)

Das Zusammenführen von Rasterdaten ist wichtig, um große geografische Bereiche abzudecken sowie diese zu einer einheitlichen Datei zusammenzuführen. Besonders bei regionalen und nationalen Projekten ist es notwendig, mehrere Rasterdateien zu einem nahtlosen Datensatz zu kombinieren. Der Geodatenkonverter nutzt hier ebenfalls Rasterio, um Rasterdaten effizient zusammenzuführen.

Die Funktion überprüft vor dem Merging, ob die Eingabedateien dieselbe räumliche Auflösung und Bounds besitzen. Falls erforderlich, werden die Daten harmonisiert, bevor sie in einen einzigen Datensatz zusammengeführt werden. Dieser Prozess ist wichtig, um Karten oder Modelle zu erstellen, die größere Gebiete darstellen sollen. Ein Beispiel hierfür ist die Zusammenführung von digitalen Orthophotos (DOP), um eine komplette Gemeinde oder Stadt für die Geovisualisierung abzudecken.

Der Pseudocode könnte so aussehen:

```
FUNKTION merge_tiffs(input_dateien, output_datei):
                Öffne Eingabedateien
                Führe Rasterdaten zusammen
                Speichere das gemergte Raster in der Ausgabedatei
ENDE
```

Abbildung 6: DOP noch nicht zusammengeführt

Abbildung 7: DOP zusammengeführt

3.3.4 DOK/DTK Zuschnitt

Das Zuschneiden von DOK/DTK-Daten (digitale Ortskarten, digitale topographische Karten) reduziert die Datenmenge auf den für ein Projekt relevanten Bereich. Diese Funktion ist besonders nützlich, wenn nur ein kleiner Ausschnitt einer Karte benötigt wird und die Kachel vor dem Zuschnitt um ein Vielfaches zu groß war.

Der Prozess beginnt mit dem Merging der DOK/DTK-Daten, um sicherzustellen, dass ein vollständiger Datensatz vorhanden ist. Basierend auf einer Bounding Box, welche aus der Referenzdatei stammt (DOP/DGM), wird der gewünschte Bereich anschließend extrahiert. Die zugeschnittenen Daten werden in einer neuen Datei gespeichert. Diese Methode ist besonders für Projekte geeignet, bei denen Karten für spezifische Anwendungen, wie z. B. 3D-Visualisierungen, vorbereitet werden müssen.

Der Pseudocode hierfür könnte wie folgt aussehen:

```
FUNKTION crop_dok_files(input_dateien, bounding_box, output_datei):
                Schneide die Dateien auf die Bounding Box zu
                Merge die DOK/DTK-Dateien
                Schreibe zugeschnittene Daten in die Ausgabedatei
ENDE
```

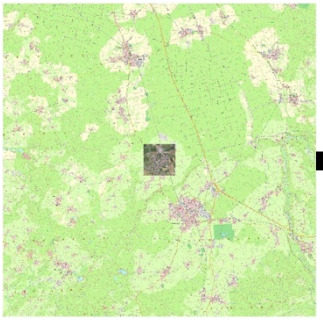

Abbildung 8: DOK in Originalgröße

Abbildung 9: DOK geschnitten, noch nicht zusammengeführt

Abbildung 10: DOK geschnitten und zusammengeführt

In Abbildung 8 wird die digitale Ortskarte (DOK) in Originalgröße gezeigt und ein digitales Orthophoto (DOP), welches den Ausschnitt der DOK zeigt, auf welchen zugeschnitten werden muss. Abbildung 9 zeigt den vorher definierten und den schon zugeschnittenen Ortsabschnitt, welcher aber noch nicht zusammengeführt wurde. Während Abbildung 10 das Endergebnis zeigt: ein zugeschnittenes und zusammengeführtes Bild mit angepasster Auflösung.

3.4 Spezielle Tools und Funktionen

Neben den bereits vorgestellten Kernfunktionen (Konvertieren, Transformieren, Zusammenführen und Zuschneiden) bietet der Geodatenkonverter einige spezielle Funktionen und Erweiterungen, die den Arbeitsablauf vereinfachen und für eine nutzerfreundliche Anwendung sorgen. Um die zusätzlichen Funktionen zu ermöglichen, wird das Programm um zusätzliche Bibliotheken ergänzt wie PyQT5, os, logging und glob und fügen sich in die bestehende Programmlogik ein. Im Folgenden werden einige dieser Funktionen beschrieben.

Automatische Namensgebung

Ein häufiges Problem bei der Verarbeitung mehrerer Dateien ist das einheitliche und konsistente Benennen der Zieldateien. Um dies zu erleichtern, wurde die Funktion *determine_suggested_name* entwickelt. Sie erkennt anhand vordefinierter Muster im Dateinamen, ob es sich beispielsweise um eine DOP- oder DGM-Datei handelt und schlägt automatisch einen passenden Namen für die Zieldatei vor. Dabei werden bekannte Kürzel wie DOP, DGM, DOK10 oder DTK25 ausgegeben. Die Nutzerin oder der Nutzer kann diesen Vorschlag übernehmen oder manuell anpassen. So lassen sich gerade bei Batch-Prozessen Fehlbenennungen vermeiden und Arbeitszeit einsparen.

Dynamisches Aktions-Fenster

Sobald Dateien ausgewählt wurden, öffnet sich ein zusätzliches Dialogfenster (*ActionDialog*), das mithilfe von PyQT5 erstellt wurde. Darin werden alle kompatiblen Aktionen angezeigt, die auf die erkannten Formate angewendet werden können. Nutzerinnen und Nutzer können dort gezielt auswählen, ob beispielsweise eine Koordinatentransformation oder eine Konvertierung gewünscht ist. Diese Übersicht reduziert Fehlbedienungen und zeigt unmittelbar, welche Schritte für die ausgewählten Datentypen zur Verfügung stehen.

Mehrstufiger Workflow mit Signal-Slot-Mechanismus

Für längere Verarbeitungsprozesse wird die Klasse *Worker* verwendet, die auf *QThread* basiert. Dies geschieht, um die grafische Oberfläche während zeitintensiver Aktionen weiterhin reaktionsfähig zu halten. Über den in PyQT implementierten Signal-Slot-Mechanismus werden Fortschritte und Statusmeldungen (z. B. „Datei wird verarbeitet…“) an die GUI übermittelt, ohne diese zu blockieren. In der Oberfläche erscheint dabei eine *QProgressBar*, die den Bearbeitungsfortschritt anzeigt.

Das folgende Beispiel verdeutlicht den Ablauf:

- Die Nutzeraktion (z. B. „Starten“) stößt den *Worker*-Thread an
- Dieser führt die ausgewählten Schritte im Hintergrund aus
- Über Signale teilt der *Worker* der GUI mit, wann ein Teilschritt abgeschlossen ist, woraufhin der Fortschrittsbalken aktualisiert wird.

Dieses Konzept sorgt für eine flüssige Anzeige, selbst wenn komplexe Rechenoperationen oder umfangreiche Datensätze verarbeitet werden.

Automatischer Workflow

Zusätzlich zum manuellen Vorgehen wurde ein halbautomatischer Ablauf für spezielle Anwendungsfälle (beispielsweise das Kombinieren von DOP- und DGM-Daten für BlenderGIS) integriert. Nach dem Auswählen der entsprechenden Dateien erkennt das Programm automatisch die Dateitypen und führt die notwendigen Bearbeitungsschritte, wie Konvertierungen und das Zusammenführen, durch. Diese Funktion ist vor allem für wiederkehrende Routinen hilfreich, bei denen stets dasselbe Ergebnisformat (z. B. Esri-ASCII und ein einzelnes GeoTIFF für Geovisualisierungen) benötigt wird.

3.5 Zusammenfassung

In Kapitel 3 wurde zunächst der grundlegende Aufbau des Geodatenkonverters beschrieben, der sich in eine grafische Benutzeroberfläche, eine Verarbeitungslogik und verschiedene Hilfsfunktionen unterteilt (siehe Kapitel 3.1). Durch diese modulare Struktur lässt sich das Programm leicht erweitern, etwa um neue Formate oder Workflows. Anschließend erfolgt eine detaillierte Vorstellung der GUI (Kapitel 3.2), die mittels PyQT5 entwickelt wurde und dank Signal-Slot-Mechanismen längere Verarbeitungsprozesse ohne Blockierung der Oberfläche ermöglicht. In der Verarbeitungslogik (Kapitel 3.3) wurden die wesentlichen Prozesse – Konvertierung, Koordinatentransformation, Zusammenführen von Geodaten sowie das Zuschneiden von DOK-/DTK-Dateien – ausführlich erläutert. Diese Prozesse bilden die Kernaufgabe des Konverters und greifen auf Bibliotheken wie Rasterio, Geopandas und PyProj zurück, um effiziente und zuverlässige Ergebnisse zu liefern.

Darauf aufbauend wurden in Kapitel 3.4 einige erweiternde Funktionen beschrieben. Es wurden die Funktionen, wie automatische Namensgebung, ein zusätzliches Aktions-Fenster zur Übersicht der möglichen Bearbeitungsschritte sowie ein halbautomatischer Workflow für wiederkehrende Routinen behandelt. Insgesamt führt dieses Zusammenspiel aus klarer Programmstruktur, leistungsfähigen Bibliotheken und nutzerorientierten Erweiterungen zu einer flexiblen, erweiterbaren und zugleich anwenderfreundlichen Softwarelösung für die Geodatenverarbeitung.

4 Ergebnisse und Tests

Nachdem die Methodik, sowie die Implementierung des Geodatenkonverters erklärt wurden, werden die Ergebnisse, die Performance, Fehlerbehandlungen, Anwendungsbeispiele sowie auf die Evaluierung des Programms vorgestellt.

4.1 Ergebnisse

Im Rahmen der durchgeführten Arbeit lässt sich feststellen, dass die Umwandlung von GeoTIFF-Dateien in das Esri-ASCII-Format zuverlässig und ohne große Probleme funktioniert. Auch das anschließende Zusammenführen mehrerer ASCII-Dateien zeigt sich als verlässlich und führt zu korrekten Ergebnissen. Analog dazu kann auch das Merging von GeoTIFF-Dateien selbst erfolgreich durchgeführt werden, ohne dass auffällige Fehler oder Qualitätseinbußen auftreten (siehe Abbildungen 11-13).

Bei der Umwandlung von GeoTIFF-Dateien zwischen unterschiedlichen Koordinatensystemen, zum Beispiel von UTM32N nach WGS84, fällt eine leichte Drehung der Rasterdaten auf. Dieses Verhalten ist jedoch in der Dokumentation der verwendeten Bibliothek bereits beschrieben und ist daher nicht als Fehler, sondern als erwartete Besonderheit zu werten („Rasterio Reprojection“, o. J.). Im 3D-Bereich ist es wichtig, dass ASCII-Daten problemlos in Blender importiert werden können, solange genügend Nachkommastellen bei der Erzeugung der ASCII-Daten berechnet werden. Dadurch lässt sich das dazugehörige GeoTIFF anschließend auch korrekt als Textur auf das importierte Mesh legen.

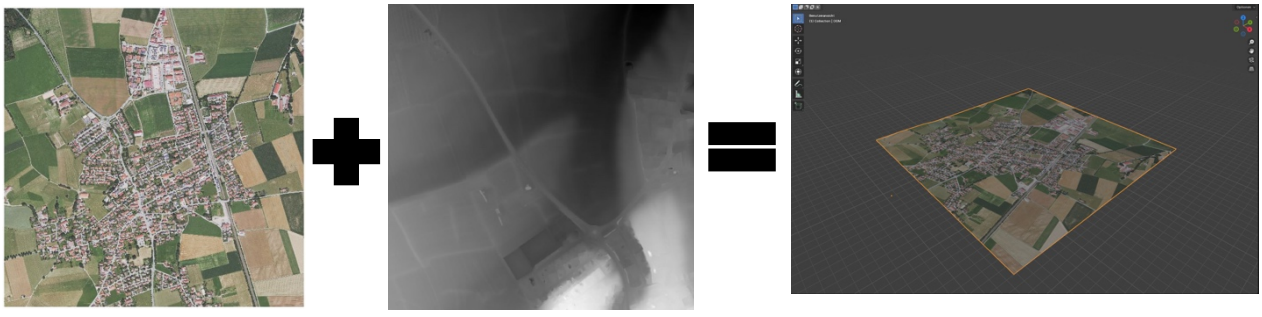

Abbildung 11: DOP als GeoTIFF

Abbildung 12: DGM als Esri-ASCII

Abbildung 13: DOP als Textur auf DGM als Mesh in Blender

Ein weiterer Punkt betrifft das Zuschneiden von DOK- oder DTK-Daten. Diese Quelldaten liegen in der Regel als sehr große Raster (zum Beispiel 20.000 x 20.000 Pixel) vor und werden auf ein kleineres Gebiet (z. B. 2.000 x 2.000 Pixel) reduziert. Dabei zeigt sich, dass Schwarztöne nach diesem Arbeitsschritt häufig in Weiß umgewandelt werden (siehe Abbildung 14). Erst nach näherer Betrachtung stellt sich heraus, dass im Hintergrund eine Farbkonvertierung (Index zu RGB) stattfindet, was diese Abweichung erklärt. Insgesamt kann das Ergebnis aber immer noch genutzt werden, da lediglich bestimmte Markierungen oder Schriften in Weiß statt Schwarz erscheinen.

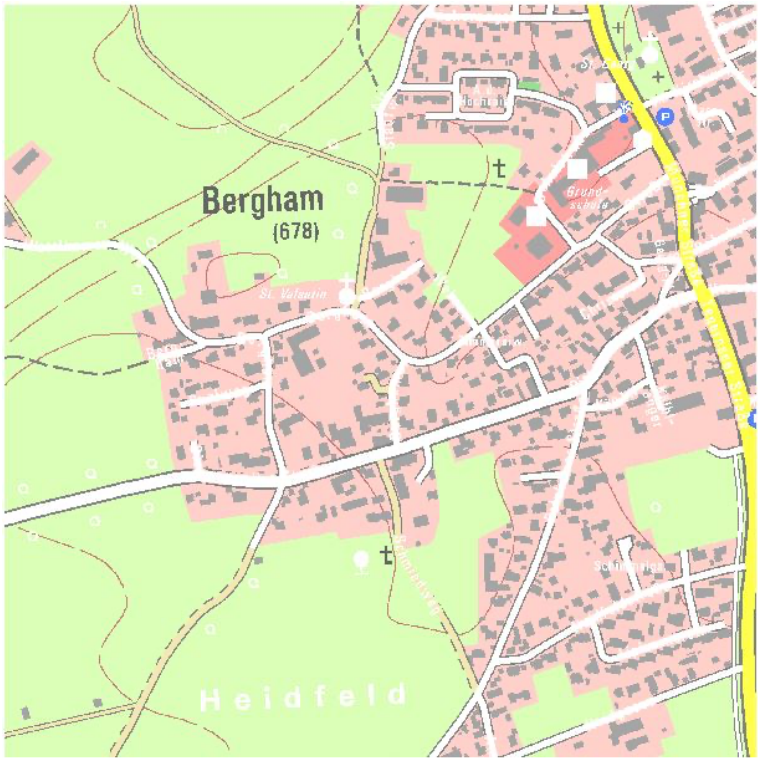

Abbildung 14: DOK in weißer Schrift

Bei der Arbeit mit Vektordaten, zum Beispiel KML- und GeoJSON-Dateien, wird deutlich, dass in der eigentlichen Anwendung (EXE) die Umwandlung wegen fehlender oder fehlerhafter GDAL-Anbindung nicht wie geplant funktioniert. In der Entwicklungsumgebung (IDE) hingegen ist das Konvertieren problemlos möglich, für das Transformieren allerdings sieht es anders aus. Für diesen Fall wird daher aktuell ein Fallback (vorläufige Lösung/ Workaround) eingesetzt. Tritt bei der Transformation ins KML-Zielformat ein Fehler aufgrund der räumlichen Referenzsysteme innerhalb der GDAL-Bibliothek auf, greift eine Schleife automatisch auf das stabilere Format GeoJSON zurück.

Eine ähnliche Einschränkung wird bei CityGML-Daten beobachtet: Während die Konvertierung zu Shape in der IDE reibungslos funktioniert, scheitert sie in der EXE an denselben GDAL-Problemen. Dafür können CityGML-Dateien erfolgreich in OBJ oder DXF umgewandelt werden (siehe Abbildungen 15 und 16). Hier aber mit folgender Einschränkung: es werden nicht alle Texturen mit in das Ziel-Format übernommen. Eine weitere Koordinatentransformation für CityGML oder andere 3D-Formate wurde bisher nicht implementiert und ist somit nicht Bestandteil der Tests.

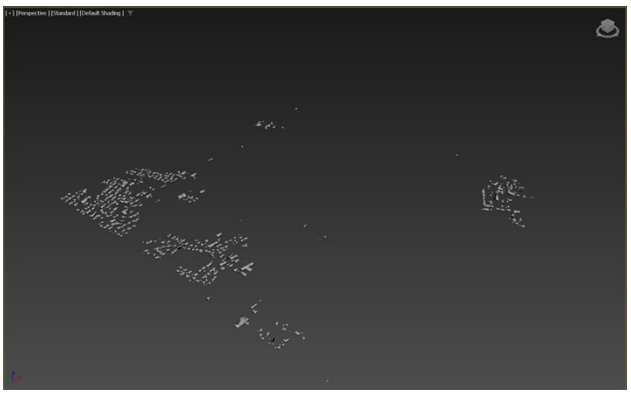

Abbildung 15: OBJ in 3D Studio Max

Abbildung 16: CityGML in FZKViewer

Zusammenfassend ist festzuhalten, dass sich viele Konvertierungen und Verarbeitungsschritte wie geplant durchführen lassen und nur kleinere Komplikationen auftreten, die in den meisten Fällen bereits bekannt oder durch Workarounds lösbar sind. In einigen Fällen, zum Beispiel bei der Integration von GDAL in die finale Anwendung. Dabei gibt es noch Verbesserungsbedarf, damit alle Formate und Transformationen gleichermaßen verlässlich genutzt und weiterverarbeitet werden können.

4.2 Performance Analyse

Im Rahmen der Tests wurde eine Performance Analyse durchgeführt, um herauszufinden wie schnell und ressourcenschonend der Geodatenkonverter verschiedene Aufgaben bewältigt. Dafür wurde ein Apple Mac Mini mit M4 genutzt, der mit einer 10-Kern-CPU (vier Performance- und sechs Effizienz-Kerne), einer 10-Kern-GPU sowie 16 GB RAM ausgestattet ist. Als Testdaten kamen jeweils vier DOP- und vier DGM-Dateien zum Einsatz, sodass bei jedem Verarbeitungsschritt immer vier Eingabedateien verarbeitet wurden. Eine Ausnahme bildet das Zuschneiden (DOK/DTK-Zuschnitt), weil hier zusätzlich zur DOP-Datei gleich vier DOK-Dateien zugeschnitten werden, also insgesamt fünf Dateien.

In Abbildung 17: Performance Analyse (Messwerte: Abbildung 1 im Anhang) sieht man die durchschnittlichen Laufzeiten (Mittelwert ± Standardabweichung) für fünf verschiedene Abläufe: Konvertierung, Merging, Koordinatentransformation, DOK/DTK-Zuschnitt und einen automatischen Durchlauf, der alle Schritte in Folge (welche für die weitere Verarbeitung in dem Programm Blender gebraucht wird) ausführt. Bei der Konvertierung von vier GeoTIFF-Dateien in Esri-ASCII wurden im Schnitt etwa 0,8 Sekunden gemessen, während das Merging von denselben vier Startdateien bei um die 0,6 Sekunden liegt. Die Koordinatentransformation von UTM32N nach WGS84 geht mit circa 0,5 Sekunden noch etwas schneller. Beim Zuschneiden von DOK-Dateien hingegen steigen die Zeiten auf rund 1,2 Sekunden an, was sich vor allem auf die Ausgangsgröße dieser Dateien (20.000 x 20.000 Pixeln) zurückführen lässt. Das Automatisieren der beiden Schritte, Konvertieren der DGM-Daten und das Mergen der DOP-Daten (jeweils zwei Dateien, um auch auf die vier Dateien zu kommen), schlägt im Schnitt mit etwa 0,8 Sekunden zu Buche.

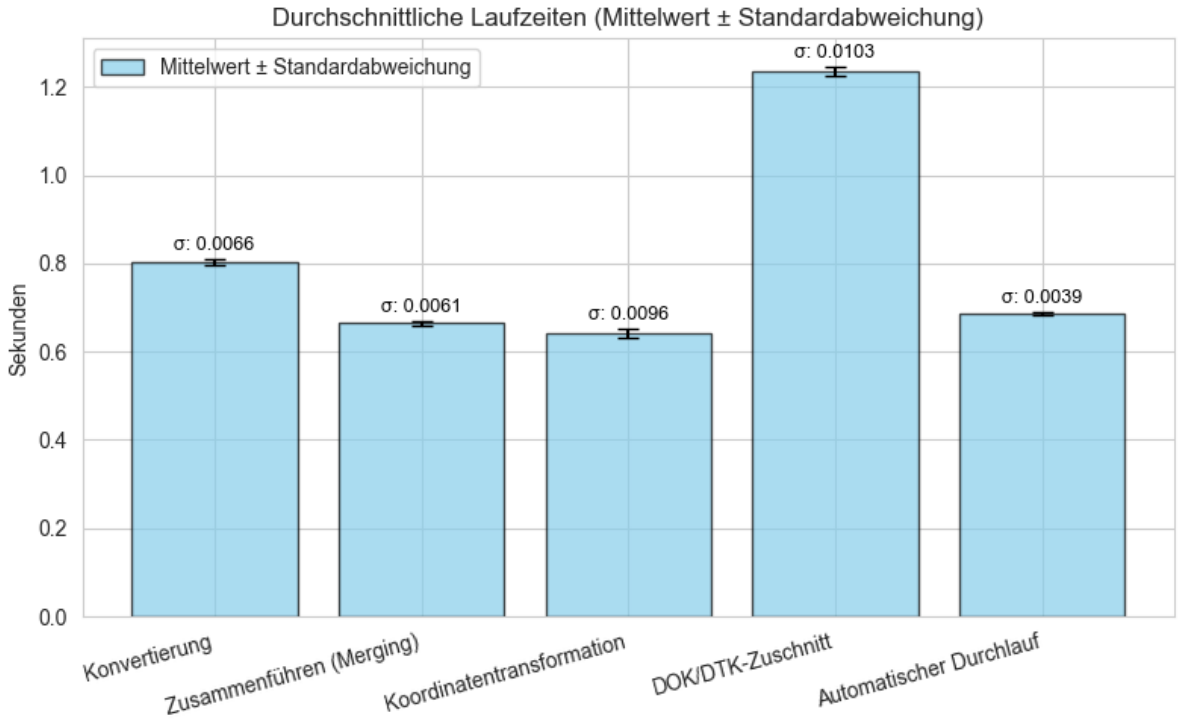

Abbildung 17: Performance Analyse

Zur Berechnung des Mittelwerts und der Standardabweichung werden folgende Formeln verwendet:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad\qquad \sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Formel 1 + 2: Mittelwert und Standardabweichung

Dabei steht $\bar{x}$ für den arithmetischen Mittelwert aller gemessenen Laufzeiten (in Sekunden) und n ist die Gesamtzahl der Messdurchläufe. Die Standardabweichung σ gibt die Streuung der Ergebnisse um den Mittelwert an. Um die Messwerte bequem zu ermitteln, werden auf die Funktionen mean() und std() aus der Bibliothek NumPy in Python zurückgegriffen, die standardmäßig die Stichproben-Standard-Abweichung berechnen.

Parallel wird beobachtet, wie sich der Speicherverbrauch entwickelt, analog dazu ebenfalls die Ergebnisse der Verarbeitungsgeschwindigkeiten. Direkt nach dem Start liegt die RAM-Auslastung bei ungefähr 133 MB. Beim Laden der vier GeoTIFFs bewegt sie sich um 143 MB. Bei der Konvertierung wächst sie leicht auf 147 MB, während das Transformieren der Dateien 151 MB benötigt. Deutlich mehr kommt erst beim DOK/DTK-Zuschnitt hinzu (rund 201 MB). Das höchste Niveau wird beim Merging der vier Dateien erreicht, das kurzzeitig bis zu 269 MB verbraucht. Auch wenn das ein spürbarer Sprung ist, bleibt das immer noch weit unter dem verfügbaren Gesamtspeicher.

Interessant ist außerdem der Blick auf die Dateigrößen. Bei der Umwandlung von GeoTIFF-Dateien (im Schnitt 2,9 MB) in ASCII-Dateien, landet die Ausgabe bei 17 MB pro Datei. Bei vier DGM-Dateien summiert sich das merklich. Noch interessanter ist, dass das anschließende Zusammenführen dieser vier Dateien statt auf 68 MB auf rund 34 MB kommt. Ähnlich verhält es sich beim DOK/DTK-Zuschnitt: Dort sind die Ursprungsdateien mit jeweils rund 17 MB bis 20 MB und einer DOP mit 75 MB nach dem Zuschneiden von Ursprünglichen 20.000 x 20.000 Pixeln auf 2.000 x 2.000 Pixeln auf insgesamt 48 MB zusammengefasst. Auch beim Umwandeln von CityGML in OBJ zeigt sich eine deutliche Reduzierung der Dateigröße von 21,6 MB auf 1,4 MB, was insbesondere für spätere 3D-Anwendungen relevant werden kann.

Nachfolgend eine Tabelle, welche das gesamte Verhalten der Dateigrößen beschreibt:

Arbeitsschritt	Eingabe Format / Größe	Ausgabe Format / Größe	Bemerkung
GeoTIFF => ASCII	1x DGM.tif ca. 2,9 MB	DGM.asc ca. 17 MB	Deutliche Vergrößerung durch unkomprimierte Rasterwerte
UTM32N => WGS84	1x DOP.tif ca. 17 MB	DOP_WGS84.tif ca. 19,1 MB	Leichte Zunahme der Dateigröße
Merging (4x GeoTIFF)	4x DOP.tif => ca. 64,4 MB	DOP.tif (Zusammengeführt) ca. 75 MB	Ergebnis durch internes Handling größer als Summe der Einzeldaten
DOK-Zuschnitt (4x DOK + 1x DOP)	4x DOP.tif + 1x DOK10.tif => ca. 148,7 MB	DOK10.tif (Zugeschnitten) Ca. 48 MB 2.000 x 2.000 Pixel	Deutliche Reduzierung durch kleinere Pixelanzahl
CityGML => OBJ	1x CityGML.gml ca. 21,6 MB	CityGML.obj ca. 1,4 MB	Deutliche Reduzierung durch Datenverlust und Farbverlust
Merging (4x Esri-ASCII)	4x DGM.asc => ca. 68 MB	DGM.asc (Zusammengeführt) ca. 34 MB	Halbierung der Gesamtgröße beim Zusammenführen

Tabelle 1: Vergleich der Dateigrößen

Insgesamt zeigen die Ergebnisse der Performance Analyse, dass alle wesentlichen Arbeitsschritte schnell durchlaufen werden und sich sinnvoll miteinander kombinieren lassen. Die marginal erhöhte Zeit beim Zuschneiden größerer Rasterdaten ist nachvollziehbar, da hier besonders große Bildbereiche verarbeitet werden müssen. Obwohl während des Merge-Vorgangs kurzfristig etwas mehr Arbeitsspeicher benötigt wird, hat sich der Mac Mini als leistungsstark erwiesen. Wichtig ist, die etwas wachsenden Dateigrößen im Auge zu behalten, wenn häufig in Esri-ASCII-Formate konvertiert wird. Dafür lassen sich die erzeugten Daten problemlos weiterverarbeiten. Zum Beispiel indem man die .asc-Dateien in Blender importiert und das passende Ortophoto (DOP) als Textur auf das Mesh legt.

4.3 Fehlerbehandlung

Während der Entwicklung des Geodatenkonverters traten Schwierigkeiten auf, die verschiedene technische und konzeptionelle Anpassungen erforderlich machten. Einige Probleme konnten durch gezielte Änderungen behoben werden, während andere Einschränkungen im Rahmen der bestehenden Möglichkeiten akzeptiert werden mussten. Eine der ersten Schwierigkeiten betraf die Darstellung der DOK-Daten. Unter Windows wurden einige Schriften unerwartet weiß angezeigt, während sie unter macOS korrekt dargestellt wurden. Die genaue Ursache konnte nicht festgestellt werden, weshalb das Problem nicht gelöst werden konnte. Auch bei den Maßstäben der DOK-Dateien ergaben sich Herausforderungen. Die Versionen im Maßstab 1:50.000 und 1:100.000 haben identische Dateinamen, was zu Konflikten bei der automatischen Benennung führte. Es wurde versucht, diese Konflikte durch Präfixe oder Suffixe zu umgehen, jedoch ohne Erfolg. Zusätzlich ließ sich für den Maßstab 1:100.000 kein gemeinsames Raster finden, während dies bei 1:50.000 problemlos möglich war. Der Grund dafür blieb unklar.

Auch die Erstellung der Benutzeroberfläche erwies sich als komplexer als erwartet. Die zunächst gewählte Tkinter-Bibliothek bot nicht die gewünschte Flexibilität, insbesondere bei der Farbgestaltung und dem Layout der Buttons. Erst der Wechsel zu PyQT brachte deutliche Verbesserungen, da damit mehr Individualisierungsmöglichkeiten zur Verfügung standen. Dadurch wurde die Oberfläche benutzerfreundlicher und optisch ansprechender. Neben der UI traten auch in der Datenverarbeitung unerwartete Schwierigkeiten auf. Bei Esri-ASCII-Dateien enthielten die Messwerte anfangs nur zwei Nachkommastellen, die zu Fehlern bei der Texturübertragung führte. Durch eine Erhöhung auf acht Nachkommastellen konnte dieses Problem behoben werden, sodass die Daten präziser dargestellt wurden. Außerdem stellte sich heraus, dass der ursprünglich verwendete „No Data Value“ mit dem Wert von 10000 Probleme verursachte. Erst nach der Umstellung auf -9999 funktionierte die Auswertung der Daten korrekt.

Ein weiteres unerwartetes Verhalten zeigte sich bei der Transformation von GeoTIFF-Dateien in das WGS84-Koordinatensystem (EPSG:4326). Nach der Umwandlung waren die Bilder gedreht, was anfänglich zu Missverständnissen führte. Da dieses Verhalten jedoch auch bei anderen Entwicklern, sowie in den verwendeten Dokumentationen der Bibliotheken wie Rasterio auftrat, deutet vieles daraufhin, dass die Darstellung technisch korrekt, aber ungewohnt ist. Noch größere Herausforderungen ergaben sich durch die fehlgeschlagene Integration der GDAL-Bibliothek. Trotz mehrfacher Installationsversuche ließ sich GDAL nicht stabil einbinden, was zu Abhängigkeiten mit Fiona und verschiedenen Kompatibilitätsproblemen führte. Besonders betroffen war die Erstellung einer ausführbaren Datei (.exe), auch „OneFile“ genannt mit dem Modul pyInstaller, da Bibliotheksprobleme und Anzeigefehler bei den verwendeten Logos die Umsetzung einer funktionierenden Stand-Alone-Version erheblich erschwerten. Dieses Problem konnte umgangen werden mit der Bibliothek cx_Freeze, welche dafür einen Ordner mit allen Abhängigkeiten sowie auch einer ausführbaren Datei (.exe) erstellte.

Des Weiteren gab es Schwierigkeiten bei der Konvertierung von CityGML in das OBJ-Format, in das SHAPE-Format und bei den Konvertierungen von KML zu GeoJSON und auch in die andere Richtung von GeoJSON zu KML. Bei CityGML zu dem SHAPE-Format, sowie bei den KML und GeoJSON Konvertierungen ist wiederholt das „GDAL-Bibliotheksproblem“ zu verorten. Bei der Umwandlung ins OBJ-Format verschlechterte sich die Textur deutlich, sodass Farben und Geometrien von der Originaldatei abweichen (bei dem Dach fehlt die rote Farbe). Während die konvertierte OBJ-Datei in dem Programm 3D Studio Max noch weitgehend korrekt angezeigt wurde, traten in Blender Fehler auf, die eine Weiterverarbeitung kompliziert machten.

Trotz der genannten Herausforderungen konnte der Geodatenkonverter erfolgreich entwickelt werden. In der Entwicklungsumgebung funktionierten die Kernfunktionen und viele technische Probleme können durch Workarounds entschärft werden. Allerdings bleiben auch in dieser Umgebung einige Einschränkungen bestehen. So werden beispielsweise die DOK-Daten unter Windows fehlerhaft dargestellt. Es besteht keine Möglichkeit, eine vollständigen Stand-Alone-Version zu erstellen. Zudem verhindert ein Problem mit der GDAL-Bibliothek das Konvertieren einiger Formate.

4.4 Anwendungsbeispiele

Die Verarbeitung und Konvertierung von Geodaten spielen in vielen verschiedenen Workflows eine zentrale Rolle. Unterschiedliche Geovisualisierungen oder Behörden arbeiten mit verschiedenen Geodatenformaten, die oft nicht direkt kompatibel sind. Hier setzt der entwickelte Geodatenkonverter an: Er ermöglicht eine effiziente und flexible Verarbeitung georeferenzierter Daten und vereinfacht die Umwandlung zwischen Formaten und Koordinatensystemen.

Im Folgenden werden verschiedene mögliche Anwendungsbereiche vorgestellt, in denen der Konverter praktisch genutzt werden kann. Anschließend wird ein beispielhafter Arbeitsablauf aus der Praxis beschrieben.

4.4.1 Mögliche Anwendungsbereiche

Verarbeitung von Open Data aus behördlichen Quellen

Viele Behörden und Open Data Portale wie *GeoDaten Bayern* oder *open.byData* stellen Geodaten in unterschiedlichen Formaten zur Verfügung. Häufig müssen diese Daten erst in das passende Format umgewandelt werden, bevor sie in einer bestimmten Software genutzt werden können. Beispielsweise sind Luftbilder und digitale Höhenmodelle oft als GeoTIFF verfügbar, während Web-Anwendungen meist in Vektorformaten wie GeoJSON arbeiten. Der Geodatenkonverter vereinfacht diesen Prozess, indem er die Koordinatensysteme automatisch erkennt und die Daten in das gewünschte Format konvertiert. Somit können Nutzerinnen und Nutzer schnell und ohne zusätzliche Software mit den vorhandenen Daten arbeiten.

Nutzung in Stadtplanung und 3D-Modellierung

In der Stadtplanung und bei 3D-Modellen werden oft verschiedene Geodaten kombiniert. Digitale Geländemodelle (DGM) oder Orthophotos (DOP) sind wichtige Grundlagen für Planungen, müssen aber für unterschiedliche Softwarelösungen aufbereitet werden. Beispielsweise verarbeiten viele GIS-Systeme ein anderes Format als Modellierungssoftware als Blender. Der Konverter hilft, diese Daten zeitsparend ins gewünschte Format zu bringen, sodass sie direkt in den jeweiligen Programmen weiterverarbeitet werden können. Dies führt zu einer Zeitersparnis und macht zusätzliche Konvertierungsschritte überflüssig.

KML/GeoJSON-Konvertierung für Geovisualisierungen

In einigen Anwendungen ist es wichtig, zwischen den Formaten KML und GeoJSON zu wechseln, da verschiedene Systeme unterschiedliche Datenformate bevorzugen. Zum Beispiel nutzen viele Anwendungen im Web GeoJSON, während KML häufiger in GIS-Programmen oder Google Earth verwendet wird. In einem gleichzeitig stattfindenden Modul (Geodatenfusion) ist es deshalb eventuell nötig, Daten von KML zu GeoJSON umzuwandeln, um sie weiterverwenden zu können. Der Konverter übernimmt diese Aufgabe automatisch und erleichtert den Wechsel zwischen den Formaten, sodass die Daten direkt weiterverwendet werden können.

Integration von Geodaten in Navigations- und Umweltanwendungen

Geodaten spielen auch eine wichtige Rolle in Umwelt- und Navigationsanwendungen. Behörden und Unternehmen nutzen sie zum Beispiel für Hochwasseranalysen, zur Beobachtung der Flächenversiegelung oder für die Planung neuer Verkehrs und Fahrradwege. Häufig sind die benötigten Daten jedoch nicht direkt im richtigen Format vorhanden oder müssen erst auf ein einheitliches Koordinatensystem gebracht werden. Mit dem Konverter lassen sich diese Daten schnell und unkompliziert anpassen, sodass sie in verschiedenen Programmen und Kartenanwendungen einfach genutzt werden können.

4.4.2 Allgemein typische Arbeitsabläufe mit Geodaten

Die Arbeit mit den Daten umfasst verschiedene Schritte, die sich je nach Anwendungsfall unterscheiden können. Dennoch gibt es einige grundlegende Abläufe, die fast immer erforderlich sind, um Geodaten effizient nutzen zu können. Diese reichen von der Beschaffung über die Verarbeitung bis hin zur Nutzung in unterschiedlichen Anwendungen.

Der erste Schritt ist die Datenbeschaffung. Geodaten stammen von unterschiedlichen Quellen, beispielsweise aus Open Data Portalen oder Landesvermessungsämtern. Oft müssen sie manuell heruntergeladen werden und in geeigneten Verzeichnissen organisiert werden, um eine strukturierte Weiterverarbeitung zu ermöglichen. Dabei kann es sich um Rasterdaten wie DGM oder DOP handeln, aber auch um Vektordaten wie Straßenverläufe oder Gebäudeumrisse im Shape-Format.

Nach dem Download folgen die Überprüfung und die Vorverarbeitung. Hierbei wird kontrolliert, ob die Daten im gewünschten Format vorliegen und ob sie bereits das entsprechende Koordinatensystem verwenden. Besonders bei der Arbeit mit mehreren Datensätzen ist es wichtig, dass alle Daten in einem einheitlichen Koordinatensystem vorliegen, damit sie korrekt überlagert und analysiert werden können. Falls nötig, müssen Dateiformate oder Koordinatensysteme angepasst werden, bevor die Verarbeitung beginnt.

Ein zentraler Schritt der Vorverarbeitung, ist die Koordinatentransformation und die Formatkonvertierung. Unterschiedliche Softwarelösungen arbeiten oft mit spezifischen Formaten und Koordinatensystemen. Beispielsweise benötigen Web-Anwendungen häufig das Koordinatensystem WGS84 (EPSG:4326), während in der Stadtplanung oft UTM-Koordinaten (z. B. EPSG:25832) verwendet werden. Ebenso erfordern einige Programme bestimmte Datenformate wie zum Beispiel das Programm Blender. Hier wird das Format Esri-ASCII benötigt, während viele GIS-Anwendungen mit GeoTIFF arbeiten. Falls die Daten nicht im richtigen Format vorliegen, müssen sie konvertiert oder transformiert werden, bevor sie weiterverwendet werden können.

Nach der Umwandlung erfolgen die Weiterverarbeitung und Nutzung der Geodaten. In Geoinformationssystemen wie QGIS (Open Source) oder ArcGIS (Closed Source) können die Daten für Analysen genutzt werden. In 3D-Modellierungsprogrammen wie Blender oder 3D Studio Max lassen sich Geodaten in digitale Stadtmodelle integrieren. Ebenso können die aufbereiteten Daten für Umweltanalysen, Navigationsanwendungen oder auch Web-Kartendienste verwendet werden. Hierfür ist es essenziell, dass die Daten im passenden Format vorliegen, damit sie ohne aufwändige manuelle Anpassungen in die jeweilige Software eingebunden werden können.

Im nachfolgenden Abschnitt werden die genauen Schritte, innerhalb des Programms beschrieben, um Daten für die Weiterverarbeitung für eine Geovisualisierung mit dem Programm Blender zu erhalten.

4.4.3 Beispielhafter Workflow für eine anschließende Geovisualisierung

Ein gewöhnlicher Workflow des Geodatenkonverters der Modi „Manueller Vorgang“ und „Automatischer Vorlauf (BlenderGIS)“ beginnt bei der „Startseite der Anwendung“ (Abbildung 18: Startseite der Anwendung). Dort finden sich zunächst Informationen zu den verschiedenen Datenformaten, die das Programm verarbeiten kann, sowie Links zu den jeweiligen Downloadseiten (z. B. GeoDaten-Bayern). Außerdem sind hier zwei grundlegende Modi sichtbar: Zum einen der automatische Vorgang, welcher speziell für DOP- und DGM-Dateien konzipiert ist und alle Schritte ohne weitere Benutzereingaben durchführt. Zum anderen der „Manuelle Vorgang“, der sich für weitere Formate (u. a. DOK, DTK, DOM, 3D-Gebäudemodelle oder KML/GeoJSON) eignet und mehr Flexibilität erlaubt.

Auf der Startseite befinden sich über der Schaltfläche „Tutorial“ (in orangener Farbe, rechts unten) Erläuterungen, weiterführende Hinweise, Optionen zur Bearbeitung. Sozusagen ein Tutorial, dass den Umgang mit den unterschiedlichen Formaten und Anwendungsfällen detailliert beschreibt.

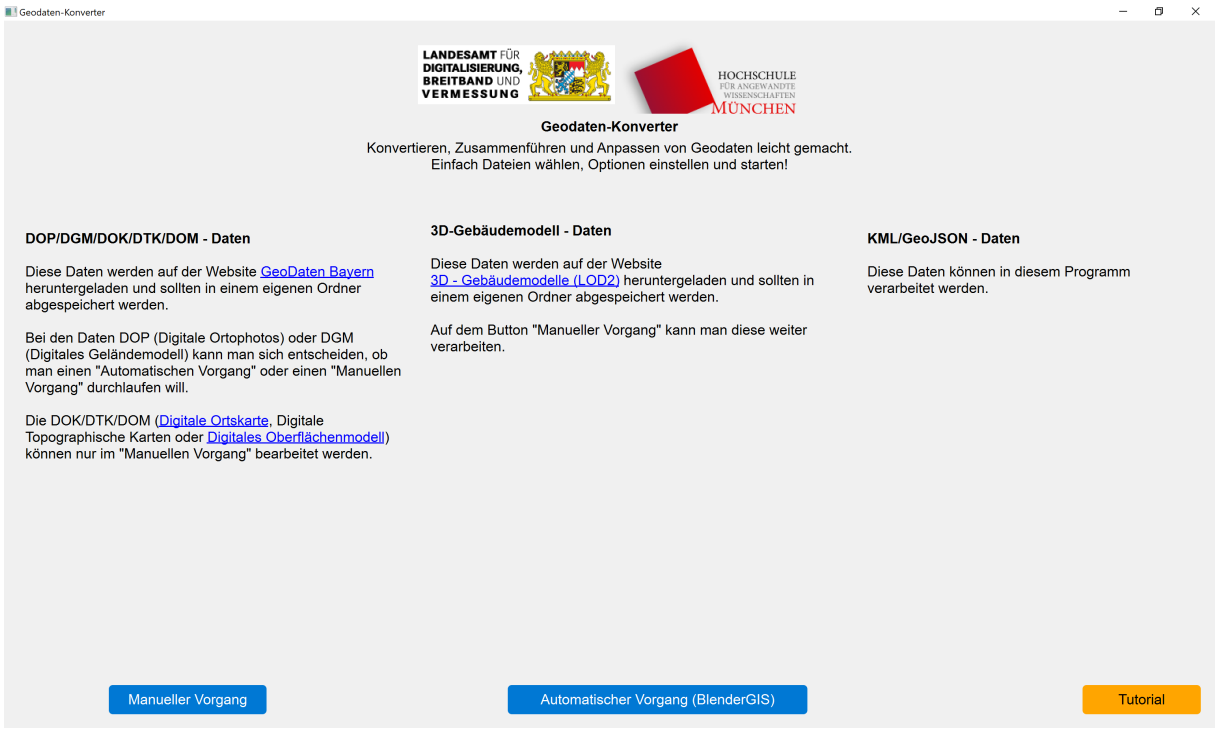

Abbildung 18: Startseite der Anwendung

Nach der Auswahl des „Manuellen Vorgangs“ öffnet sich eine neue Ansicht (siehe Abbildung 19). Im neuen Fenster stehen nun Schaltflächen zur Datei- oder Ordnerauswahl bereit. Über „Dateien auswählen“ lassen sich einzelne oder mehrere Geodatenquellen (z. B. mehrere DGM-Dateien) aufrufen. Beim „Ordner auswählen“ hingegen bezieht sich die Auswahl der Daten auf ein gesamtes Verzeichnis, mit allen in ihm befindlichen Dateien. Hier ist darauf zu achten, dass nur Daten mit den gleichen Formaten vom Geodatenkonverter verarbeitet werden können – außer bei einem DOK/DTK-Zuschnitt. Direkt nach dem Laden der Daten (siehe Abbildung 20: Dateiauswahl) erscheint ein Dialogfenster (siehe Abbildung 21: Aktionsauswahl), indem sich die gewünschten Aktionen festlegen lassen. In diesem Beispiel geht es um die Konvertierung von GeoTIFF-Dateien in ein Esri-ASCII Format, weshalb die Optionen „Konvertieren“ sowie „Zusammenführen (Merging)“ aktiviert werden müssen. Die aktuelle Zusammenfassung der gewählten Aktionen wird stets im unteren Bereich angezeigt und kann über „Aktionen bearbeiten“ nachträglich angepasst werden.

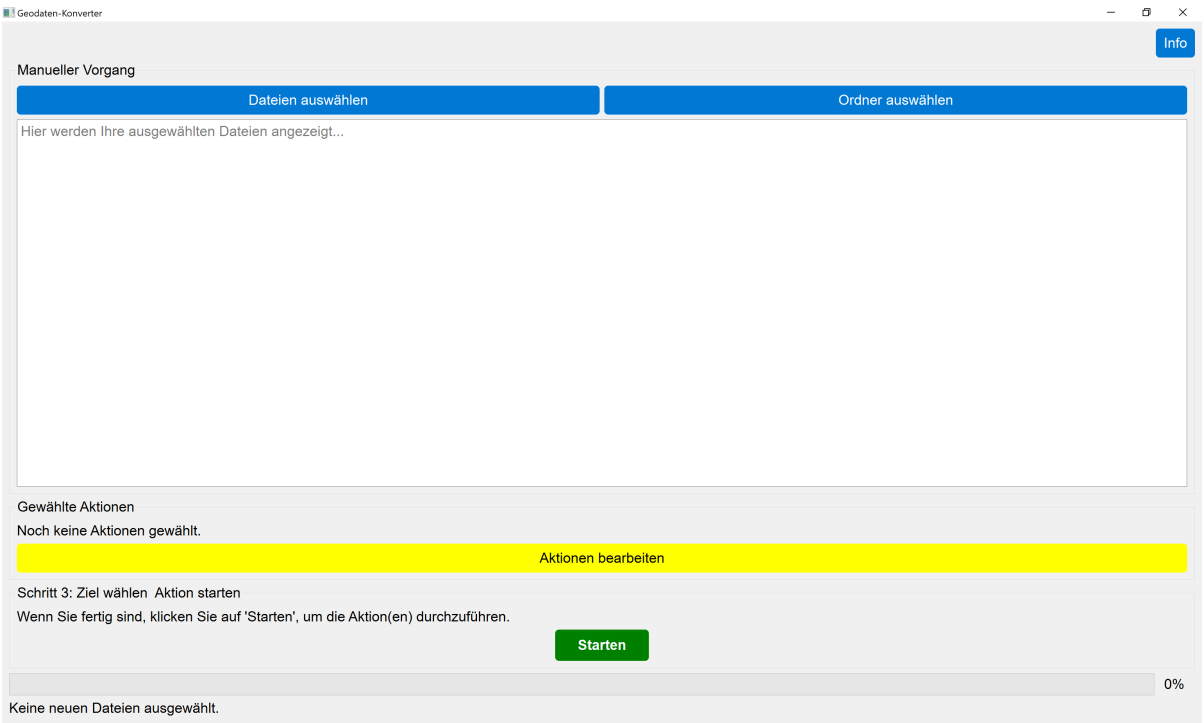

Abbildung 19: Manueller Vorgang

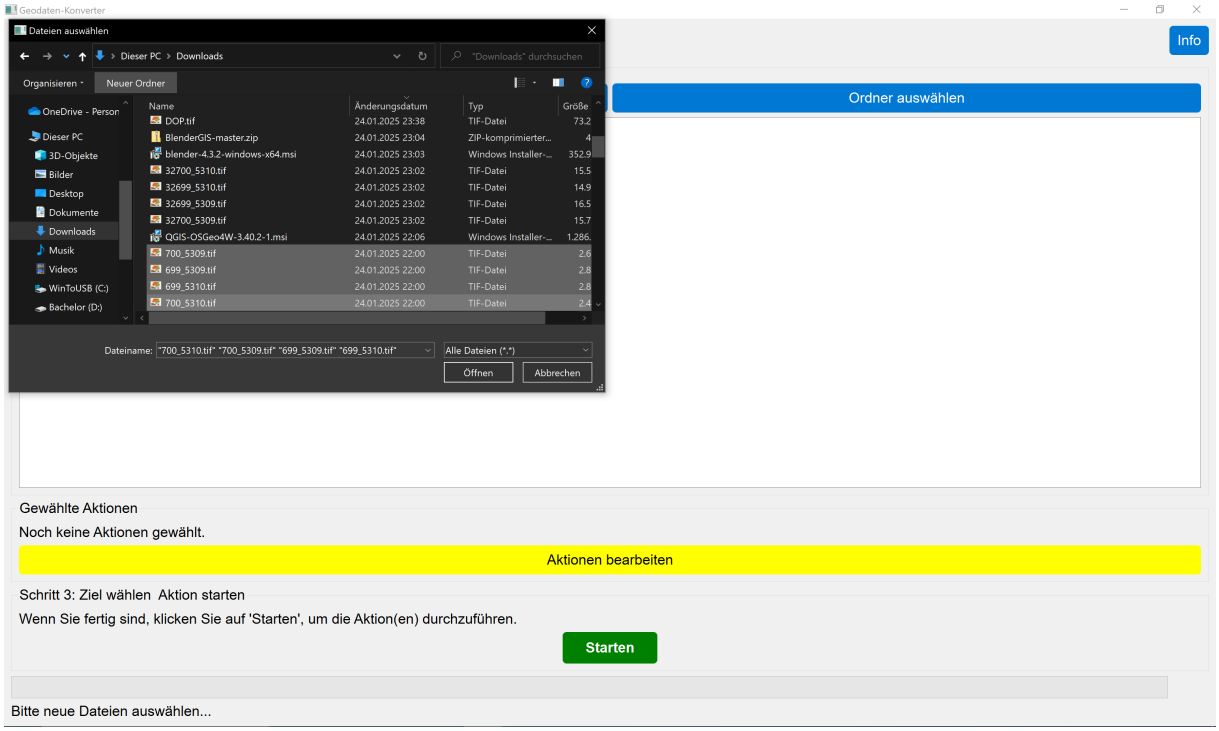

Abbildung 20: Dateiauswahl

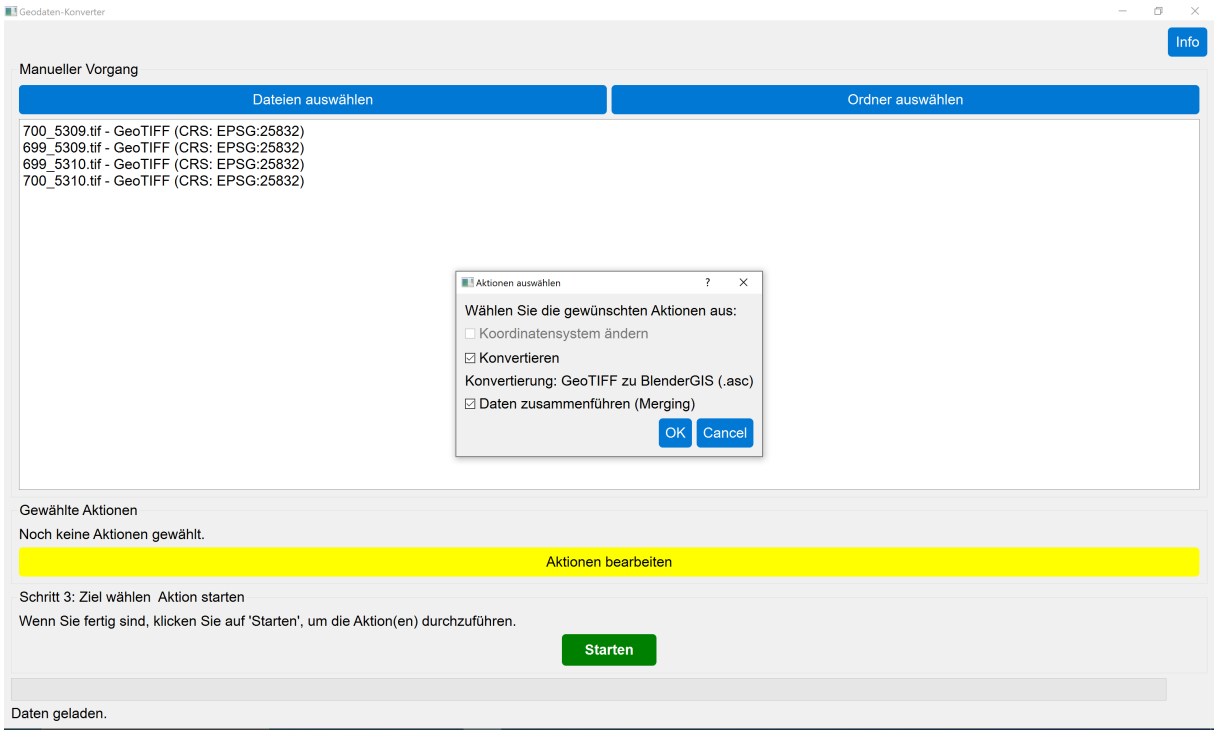

Abbildung 21: Aktionsauswahl

Sobald die Konfiguration abgeschlossen ist, wird über die Schaltfläche „Starten“ der Zielordner festgelegt. Der Geodatenkonverter schlägt von sich aus einen Dateinamen vor, der aus dem Quell- und Zielformat abgeleitet wird. Dieser kann durch eine eigene Namensbezeichnung ersetzt werden (siehe Abbildung 22: Zielauswahl). Nach der Eingabe-Bestätigung beginnt der Konvertierungs- und Merge-Prozess, dessen Fortschritt sich anhand eines Ladebalkens am unteren Fensterrand ablesen lässt.

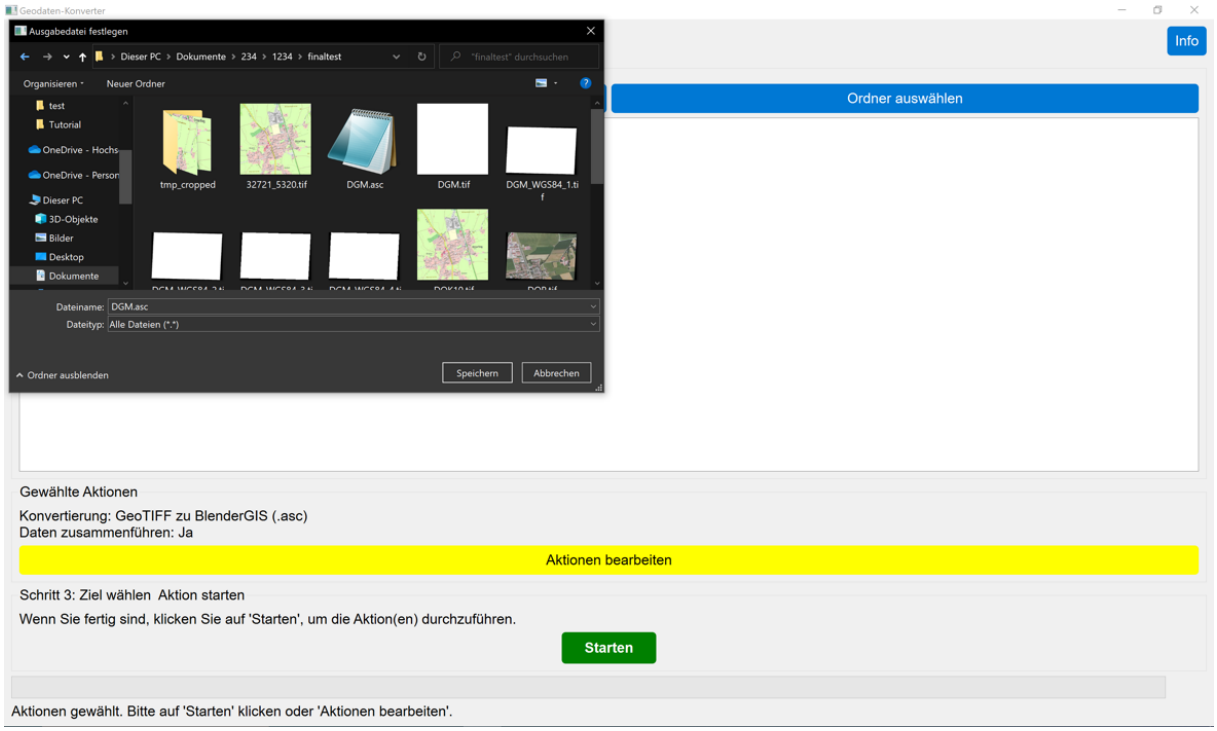

Abbildung 22: Zielauswahl

Ist die Verarbeitung erfolgreich abgeschlossen, blendet das Programm eine Meldung ein, die Auskunft über den Speicherort und den Dateinamen der erzeugten Datei gibt (siehe Abbildung 23: Zielordner und Zieldateiname).

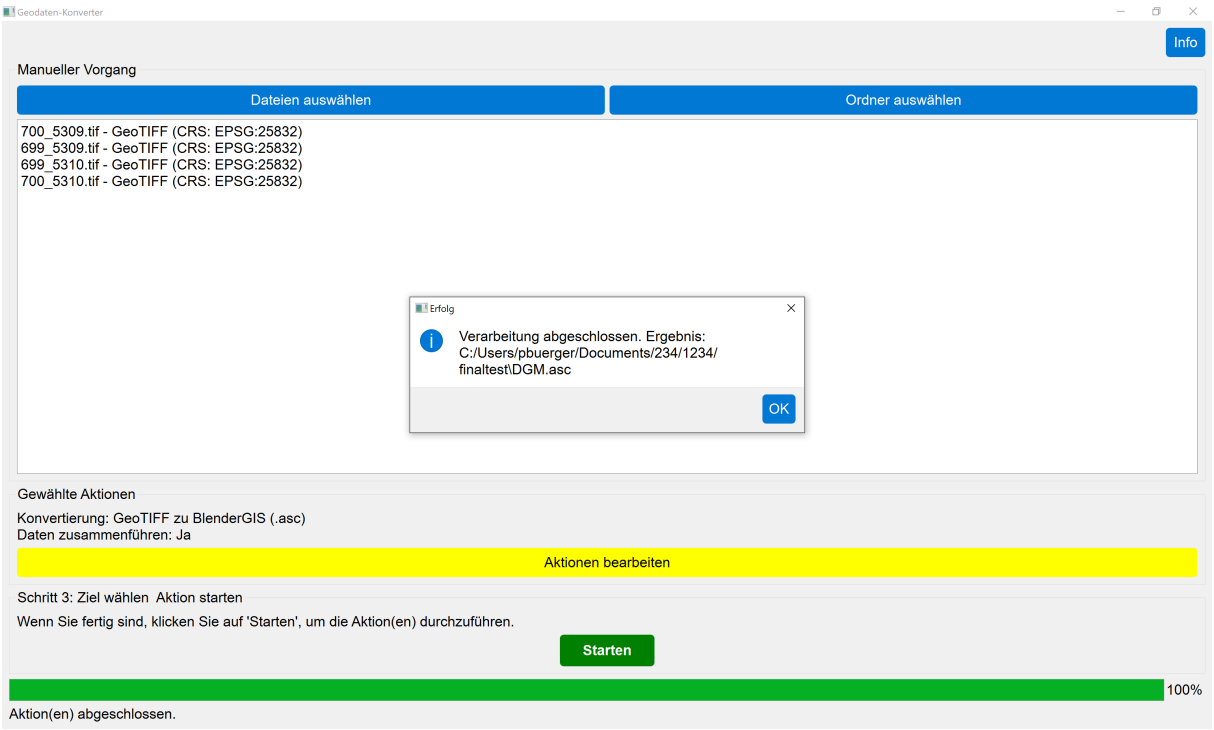

Abbildung 23: Zielordner und Zieldateiname

In diesem Abschnitt erläuterten Beispiel wurden vier GeoTIFF-basierte DGM-Dateien in eine einzelne ASCII-Datei überführt und gibt eine kurze Erklärung eines typischen Workflows wieder.

Bei der Auswahl des Modus „Automatischer Vorgang“ öffnet sich ein Fenster für die Eingabe der Startdateien und anschließend ein weiteres für den Speicherort der Zieldateien. Das Programm wandelt automatisch alle DGM-Daten von GeoTIFF zu ASCII und führt diese zusammen. Alle DOP-Daten werden automatisch gemerged und anschließend werden diese beiden Dateien nach dem voreingestellten Muster benannt und am ausgewählten Speicherort abgespeichert.

4.5 Vergleich mit anderen Tools

Im Rahmen mehrerer Tests wurde das aus dieser Arbeit stammende Programm mit den Programmen FME (Feature Manipulation Engine), FZKViewer (vom KIT) und QGIS (Quantum GIS) verglichen. Nachfolgend werden kurz die jeweiligen zu vergleichenden Tools vorgestellt und anschließend auf die Performance eingegangen. Die Prozesse der verglichenen Software wird in der Farbe Rot dargestellt, während der Geodatenkonverter in der Farbe Blau dargestellt wird. Zu beachten ist, dass nicht jede Software die gleichen Aktionen durchführen kann.

Für die Bestimmung der Laufzeiten kamen unterschiedliche Ansätze zum Einsatz. Bei QGIS und FME wurde auf interne Funktionen zurückgegriffen. QGIS zeigt die Dauer eines Prozesses direkt im jeweiligen Fenster an, während FME die Laufzeit pro Durchlauf im Programmeigenen Terminal ausgibt. Beim Geodatenkonverter wurde der Code entsprechend erweitert, um die Zeit einer einzelnen Aktion zu messen. Da der FZKViewer über keine interne Funktion zur Laufzeitmessung verfügt, wurde hier der gesamte Prozess per Bildschirmaufnahme erfasst. Durch Setzen von Zeitstempeln zu Beginn und am Ende eines Prozesses konnte die benötigte Zeit ermittelt werden. Um aussagekräftige Ergebnisse zu erzielen, wurde jede Aktion zehnmal wiederholt.

FME (Feature Manipulation Engine)

FME von Safe Software ist eine umfassende Plattform für die Datenintegration sowie Datentransformation, die speziell für Geodaten entwickelt wurde. Die Software ermöglicht die Automatisierung komplexer ETL-Prozesse (Extract, Transform, Load) durch eine intuitive, visuelle Arbeitsumgebung. Benutzerinnen und Benutzer erstellen Workflows durch Drag-and-Drop der vorhandenen Komponenten, wodurch tiefgehende Programmierkenntnisse überflüssig werden. FME bietet eine umfangreiche Bibliothek von Transformern, die vielfältige Datenmanipulationen unterstützen, wie das Transformieren von Koordinatensystemen, das Zusammenführen von Datensätzen und das Filtern von Informationen („Platform“, o. J.).

FME unterstützt eine breite Palette von 450 verschiedenen Geodatenformaten, darunter GeoJSON, KML, OBJ, CityGML, DXF, ASCII und GeoTIFF. Diese Vielseitigkeit macht FME zu einem unverzichtbaren Werkzeug für die Konvertierung und Transformation verschiedenster Geodaten („Platform“, o. J.).

Dank der anwenderfreundlichen Benutzeroberfläche und der umfangreichen Dokumentation ist FME sowohl für Einsteiger als auch für erfahrene GIS-Profis zugänglich. Die visuelle Gestaltung der Workflows erleichtert das Verständnis und die Wartung komplexer Datenprozesse erheblich („FME by Safe Software Community | Community“, o. J.).

Bei diesem Vergleich kamen aus Testgründen verschiedene Rechner zum Einsatz, da die zu vergleichenden Programme (FME und Geodatenkonverter (in der IDE)) nicht auf demselben Endgerät funktionieren.

FME wurde auf einem Windows 10 Desktop PC (Intel Core i7 6700 @3,4GHz, 32GB RAM) ausgeführt. Dabei liegt die RAM-Auslastung im Leerlauf zunächst bei rund 500 MB und erhöht sich bei geladenen DOP-Dateien auf etwa 515 MB. Nach dem Laden von DGM-Dateien steigt sie auf 524 MB. Während der Koordinatentransformation (mit DOP-Dateien) erreicht FME einen Spitzenwert von 658 MB, wohingegen die Konvertierung von DGM-Dateien maximal 638 MB benötigt.

Der Geodatenkonverter kam auf einem Windows 10 Laptop (Intel Core i9 @2,3GHz, 32GB RAM) zum Einsatz. Hier beträgt die RAM-Auslastung im Leerlauf etwa 97 MB, beim Laden der DOP-Dateien 110M B und bei DGM-Dateien 116 MB. Während der Konvertierung steigt der Wert auf rund 126 MB und bei der Transformation auf 153 MB. Die höchsten Auslastungen treten beim Zusammenführen mit 310 MB, sowie beim DOK-Zuschnitt mit 324 MB auf. Im automatischen Durchlauf, der mehrere Schritte nacheinander ausführt, liegt der Speicherbedarf bei etwa 156 MB.

Die dazugehörigen Laufzeiten der einzelnen Arbeitsschritte – Konvertierung, Merging, Koordinatentransformation, DOK-Zuschnitt und automatischer Durchlauf – sind in einem Balkendiagramm dargestellt (Abbildung 24: Balkendiagramm Vergleich mit FME). Dabei zeigen sich deutliche Unterschiede zwischen den beiden Programmen. FME benötigt für einzelne Schritte teils höhere Zeiten (im niedrigen bis mittleren zweistelligen Sekundenbereich). Der Geodatenkonverter erreicht bei denselben Prozessen kürzere Laufzeiten (oft im einstelligen Sekundenbereich).

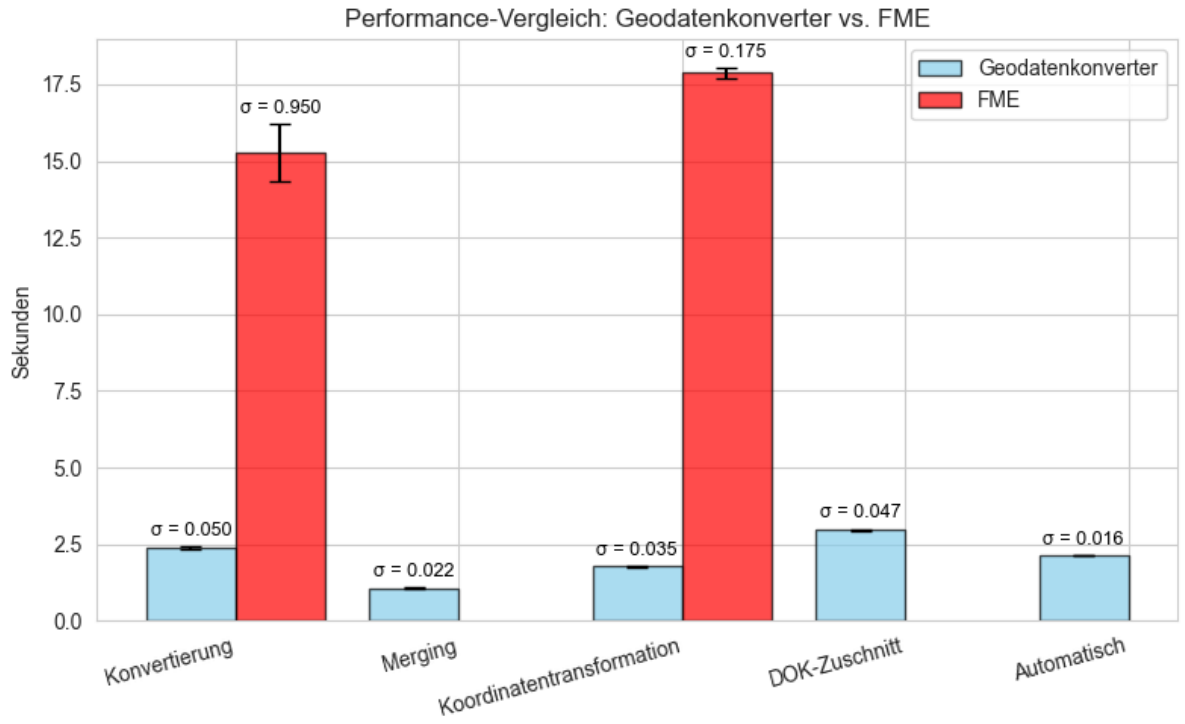

Abbildung 24: Balkendiagramm Vergleich mit FME

Abbildung 24 zeigt einen Performance-Vergleich zwischen dem Geodatenkonverter (Messwerte: Abbildung 2 im Anhang) und der Software FME (Messwerte: Abbildung 3 im Anhang) für die verschiedenen genannten Arbeitsschritte. Die Balken geben die benötigte Zeit in Sekunden an, wobei auch die jeweilige Standardabweichung (σ) berücksichtigt wurde. Dabei ist zu beachten, dass nicht nur die Software, sondern auch die Hardware – beispielsweise Prozessorarchitektur und Taktfrequenz – die Leistungsfähigkeit beeinflusst. Aus diesem Grund sind die dargestellten Ergebnisse als Momentaufnahme unter den jeweils gegebenen Testbedingungen zu verstehen und bedürften weiteren Untersuchungen.

QGIS (Quantum GIS)

QGIS ist eine weit verbreitete Open Source GIS-Software, die eine umfassende Palette an Funktionen für die Erstellung, Bearbeitung, Analyse und Visualisierung von Geodaten bietet. Durch die Unterstützung zahlreicher Plugins lässt sich die Funktionalität von QGIS flexibel erweitern und an spezifische Anforderungen anpassen. Die Software ermöglicht die einfache Konvertierung zwischen verschiedenen Geodatenformaten sowie die Transformation von Koordinatensystemen durch integrierte Werkzeuge und Skripte („QGIS User Guide — QGIS Documentation documentation“, o. J.).

Mit einer intuitiven Benutzeroberfläche und einer aktiven Community ist QGIS sowohl für Anfänger als auch für erfahrene Benutzerinnen und Benutzer geeignet. Die umfangreiche Dokumentation sowie zahlreiche Tutorials erleichtern den Einstieg und die effektive Nutzung der Software erheblich („QGIS Community“, 2022).

QGIS unterstützt nativ Raster- sowie Vektordatenformate wie GeoJSON, KML, DXF, GeoTIFF. 3D-Modelle werden nativ nicht unterstützt, weswegen diese in dieser Arbeit nicht beschrieben werden.

In diesem Vergleich wurden weitere Performance- und Analysetests mit dem Programm QGIS untersucht. Diese wurden beide auf demselben PC wie in Kapitel 4.2 durchgeführt. Abbildung 25 zeigt die durchschnittlichen Laufzeiten (Mittelwert ± Standardabweichung) für die bekannten fünf Arbeitsschritte. Zusätzlich wurden bei beiden Programmen die RAM-Auslastung zu unterschiedlichen Zeitpunkten protokolliert sowie die Dateigrößen der Ausgangs- und Ergebnisdateien erfasst.

Beim Geodatenkonverter beginnt die Arbeitsspeicherauslastung im Leerlauf bei rund 218 MB. Sobald eine DOP-Datei für die Änderung des Koordinatensystems geladen wird, steigt sie leicht auf 137 MB und erreicht kurzzeitig während der eigentlichen Transformation 155 MB. Das Merging von vier Dateien hat eine maximale Auslastung von bis zu 232 MB, bevor sie nach Abschluss wieder sinkt. Für die Konvertierung eines DGM (aus GeoTIFF zu ASCII) notiert der Geodatenkonverter einen Höchstwert von 147 MB. Im DOK-/DTK-Zuschnitt und dem anschließenden automatischen Durchlauf ähneln die Werte jenen aus den Tests aus Kapitel 4.2, da hier dieselben Routinen zum Einsatz kommen. Die erzeugte Zieldatei umfasst bei der Koordinatentransformation einer DOP-Datei beispielsweise rund 19,1 MB, während das

Konvertieren einer DGM-Datei zu Esri-ASCII in diesem Fall 17 MB benötigt. Das Zusammenführen von vier Orthophotos liefert eine gebündelte Datei mit ca. 75 MB Größe.

QGIS hingegen zeigt im Leerlauf eine RAM-Auslastung von etwa 330 MB. Das Erstellen des Mosaiks (Merging) aus vier DOP-Dateien erfordert kurzzeitig 541 MB, wobei QGIS die Bilddateien meist als virtuelles Raster einbindet. Das Laden von Dateien zur anschließenden Konvertierung (z. B. nach Esri-ASCII) führt zu einer Auslastung von 606 MB, die bei anspruchsvolleren Prozessen wie Schummerung und kompletter Umwandlung in Esri-ASCII sogar 804 MB erreichen kann. Bei der Koordinatentransformation einer DOP-Datei bewegt sich der Arbeitsspeicher zwischen 620 MB und 628 MB. Die erzeugten Zieldateien weisen Größen auf, die mit den Ergebnissen des Geodatenkonverters grundsätzlich vergleichbar sind, darunter ebenfalls 75 MB beim Zusammenführen und rund 19,1 MB nach einer Transformation. Für die Konvertierung eines DGM nach ASCII meldet QGIS allerdings eine Endgröße von etwa 4 MB, was auf eine abweichende interne Handhabung oder Kompressionsmethodik schließen lässt.

Das Balkendiagramm in Abbildung 25 verdeutlicht die Unterschiede in den reinen Laufzeiten. Während der Geodatenkonverter (Messergebnisse: Abbildung 4 im Anhang) zum Beispiel die Konvertierung in kurzer Zeit durchführt (0,21 Sekunden ± 0 Sekunden) und das Merging moderate Werte aufweist (0,67 Sekunden ± 0,01 Sekunden), zeigt QGIS (Messergebnisse: Abbildung 5 im Anhang) bei einzelnen Schritten geringfügig andere Verarbeitungsdauern (z. B. Konvertierung: 0,67 Sekunden ± 0,01 Sekunden).

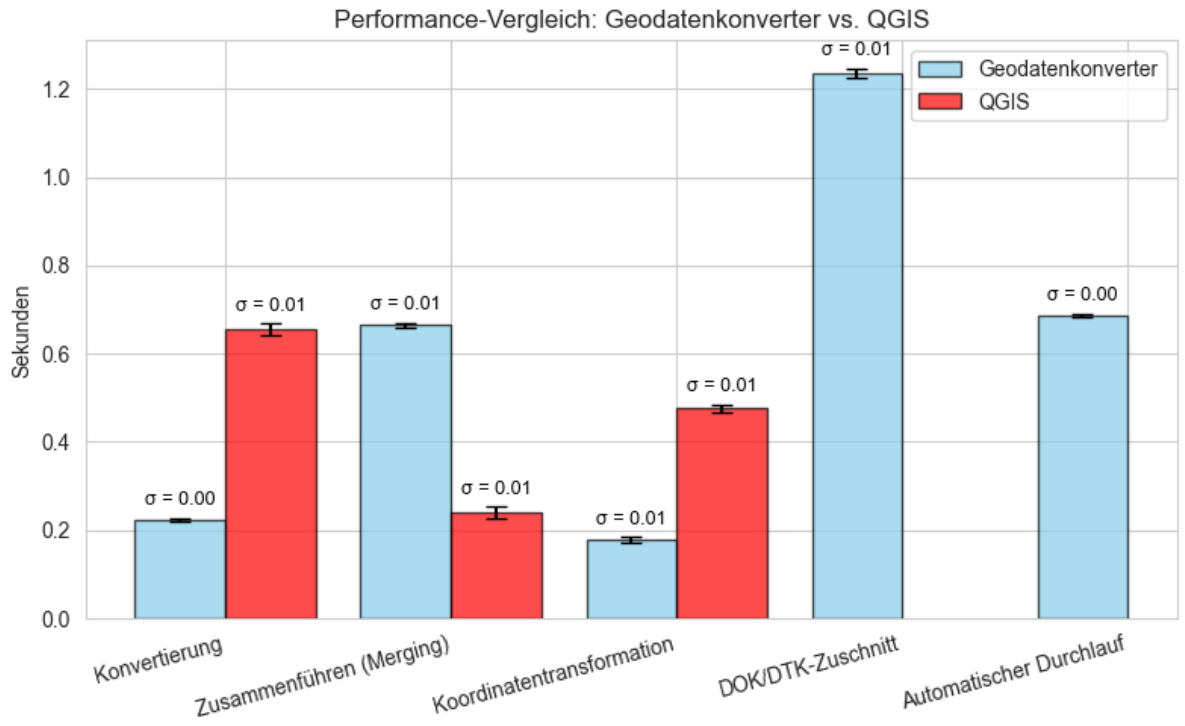

Abbildung 25: Balkendiagramm Vergleich mit QGIS

FZKViewer KIT (Karlsruher Institute of Technology)

Der FZKViewer, entwickelt am Institut für Angewandte Informatik (IAI) des Karlsruher Instituts für Technologie (KIT), ermöglicht die interaktive Darstellung, Analyse und Weiterverarbeitung von 3D-Stadt- und Gebäudemodellen. Gemäß den Informationen auf der Projektseite des IAI unterstützt das Programm verschiedene Formate, darunter CityGML, IFC sowie diverse Exportformate wie DXF („FZKViewer“ 2025). Dadurch können zum Beispiel Gebäudedaten oder Stadtmodelle, die ursprünglich in CityGML oder IFC vorliegen, in DXF konvertiert werden.

Das Programm FZKViewer wurde mit dem gleichen Versuchsaufbau wie bei FME und QGIS untersucht. Am Windows 10 Laptop wurden dabei eine CityGML-Datei (21,2 MB) sowie eine DOP-Datei (15,5 MB) eingespeist, um einerseits die Konvertierung nach DXF durchzuführen und andererseits das Orthophoto (DOP) in das WGS84-Koordinatenssystem zu transformieren.

Beim Geodatenkonverter beträgt die RAM-Auslastung im Leerlauf 102 MB. Nach dem Laden der CityGML-Datei steigt sie leicht auf 104 MB, während das Einlesen des Orthophotos (DOP) zu einer Auslastung von 115 MB führt. Die anschließende Koordinatentransformation beansprucht 121 MB, wohingegen die Konvertierung nach DXF kurzzeitig 329 MB erfordert. Die resultierende Zieldatei umfassen ein DXF mit einer Größe von 16,7 MB sowie eine in WGS84 transformierte DOP-Datei von 18,6 MB.
Der FZKViewer zeigt dagegen im Leerlauf eine Arbeitsspeicherauslastung von 21 MB, die nach dem Laden derselben CityGML-Datei auf 690 MB ansteigt. Die eigentliche Konvertierung nach DXF führt zu einer kurzfristigen Erhöhung auf 723 MB. Aus der CityGML-Datei generiert der FZKViewer eine DXF mit einer Dateigröße von 40,9 MB.

Im Balkendiagramm (Abbildung 26: Balkendiagramm Vergleich mit FZKViewer) werden die Laufzeiten für die Konvertierung einer CityGML-Datei in das DXF-Format, sowie eine Zeit einer Transformation der DOP von UTM32N nach WGS84 dargestellt. Hier zeigt sich, dass der Geodatenkonverter (Messergebnisse: Abbildung 6 im Anhang) im Durchschnitt etwa 7 Sekunden (±0,16 s) benötigt, während der FZKViewer (Messergebnisse: Abbildung 7 im Anhang) denselben Schritt in etwa 3 Sekunden (±0,04 s) ausführt.

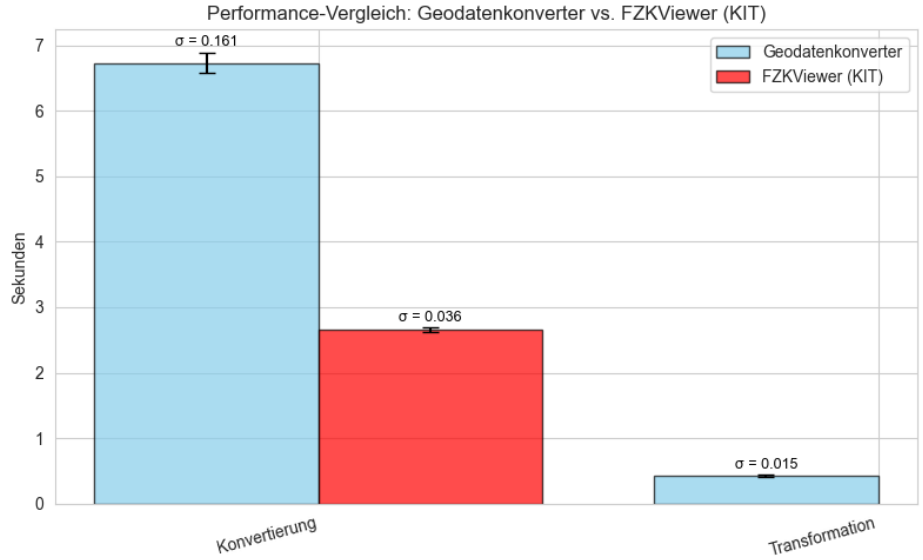

Abbildung 26: Balkendiagramm Vergleich mit FZKViewer

5 Erweiterung der Forschung und Diskussion

5.1 Diskussion

Die in Kapitel 4 vorgestellten Ergebnisse zeigen, dass der entwickelte Geodatenkonverter die wichtigsten Arbeitsschritte – von der Konvertierung und Koordinatentransformation bis zum Zusammenführen (Merging) – insgesamt verlässlich ausführt. Besonders die Umwandlung von GeoTIFF-Dateien ins Esri-ASCII-Format und das Zusammenführen mehrerer ASCII-Dateien haben sich als sehr zuverlässig erwiesen und liefern konsistente Resultate.

Ebenso funktioniert das Merging von GeoTIFF-Dateien einwandfrei, ohne dass es zu auffälligen Fehlermeldungen oder Qualitätseinbußen kommt. Zu erwähnen ist allerdings die leichte Drehung von Rasterdaten bei der Transformation, zum Beispiel von UTM32N nach WGS84. Dieses Verhalten ist bereits in der Dokumentation, der verwendeten Bibliothek, beschrieben worden und gilt daher eher als bekannte Besonderheit statt als Programmfehler.

Im 3D-Bereich funktioniert die Darstellung der konvertierten ASCII-Dateien in Blender, solange ausreichend Nachkommastellen bei der Erzeugung der ASCII-Daten berücksichtigt werden. Auf diese Weise lässt sich das passende zusammengeführte GeoTIFF anschließend als Textur auf ein Mesh legen, was eine wichtige Grundlage bei der 3D-Modellierung, zum Beispiel in der Stadtplanung oder in Visualisierungen, bietet. Allerdings ist hier zu beachten, dass nicht immer alle Texturen in DXF- oder OBJ-Formaten übernommen werden, was den Einsatz für umfangreichere 3D-Projekte derzeit einschränkt.

Probleme mit Vektordaten und GDAL-Integration

Im Bereich der Vektordaten (KML, GeoJSON, CityGML) sind dagegen ein paar Probleme zu beheben. In der Entwicklungsumgebung (IDE) lässt sich vieles problemlos konvertieren, doch in der .exe Anwendung macht vor allem die fehlende oder fehlerhafte GDAL-Anbindung Schwierigkeiten. Das führt dazu, dass KML-Daten nicht wie geplant ins Zielsystem überführt werden können oder dass CityGML-Dateien zwar in OBJ oder DXF umgewandelt, aber ohne vollständige Textur abgespeichert werden. Durch den Fallback über GeoJSON wurde zwar ein Workaround gefunden, jedoch ist das nur eine Übergangslösung, bis man eine Lösung mit den fehlerhaften Referenzsystem gefunden hat.

Diese Einschränkungen hängen eng mit dem bekannten „GDAL-Bibliotheksproblem“ zusammen, das beim Erstellen einer lauffähigen Stand-Alone-Version immer aufgetreten ist. Trotz mehrfacher Versuche ließ sich das Ganze nicht stabil integrieren, sodass einige Formate nur teilweise oder gar nicht konvertiert werden können. Hier besteht daher noch deutlicher Verbesserungsbedarf.

Farbkonvertierung und andere Besonderheiten

Ein weiteres Thema ist die Veränderung von Schwarztönen in Weiß beim Zuschneiden von DOK- oder DTK-Daten. Auch wenn diese Abweichung auf eine Farbkonvertierung (Index→RGB) zurückzuführen ist und die eigentlichen Daten meist weiterverwendbar bleiben, kann das zu Verwirrungen führen. Interessanterweise tritt dieses Problem nur mit bestimmten Rechnern auf. Dies deutet daraufhin, dass hier ein system- oder treiberspezifisches Problem vorliegt, das man bei einer tiefergehenden Fehlersuche genauer untersuchen müsste.

Auch im Zusammenhang mit Rasterdaten sind unterschiedliche No-Data-Werte aufgefallen. Durch eine manuelle Umstellung des Wertes, ließen sich die Daten korrekt auswerten, was zeigt, wie wichtig eine sorgfältige Wahl des No-Data-Werts für eine fehlerfreie Verarbeitung ist.

Performance und Ressourcenbedarf

Die durchgeführten Performance Tests zeigen, dass der Konverter die meisten Vorgänge auch mit wenig Arbeitsspeicherbedarf sehr schnell abschließt. Das Zuschneiden großer Raster (z. B. 20.000 x 20.000 Pixel) dauert naturgemäß etwas länger, ist aber immer noch im vertretbaren Rahmen. Beim Merging mehrerer Dateien gab es zwar kurzzeitig einen erhöhten RAM-Bedarf, dennoch blieb alles deutlich unter dem verfügbaren Gesamtspeicher der Testsysteme.

Interessant sind außerdem die teils stark schwankenden Dateigrößen: Während die Konvertierung von GeoTIFF nach Esri-ASCII die Dateien zum Teil deutlich vergrößert, halbiert sich beim anschließenden Merging die Gesamtgröße teils wieder. Diese Ergebnisse verdeutlichen, wie unterschiedlich die einzelnen Formate intern aufgebaut sind und dass unkomprimierte Formate, wie nach der Umwandlung von dem GeoTIFF-Format und den anfänglichen 2,9 MB zu dem ASCII-Format und den letztendlich 17 MB, oft mehr Platz beanspruchen.

Vergleich mit bestehenden Lösungen

Im Vergleich mit den etablierten Tools wie FME, QGIS und FZKViewer wird deutlich, dass der erstellte Konverter bei vielen Standardprozessen (Konvertierung, Merging, einfache Transformation) durchaus mithalten oder sogar schneller arbeiten kann, wie es der Vergleich mit den verschiedenen Tools gezeigt hat. Bei den Programmen FME und QGIS gab es beim Konvertieren sowie bei der Koordinatentransformation, im Vergleich mit dem Geodatenkonverter, eine deutlich langsamere Laufzeit. Der durchschnittliche Wert einer Konvertierung lag in diesem Fall bei 8 Sekunden, wobei der Geodatenkonverter mit 0,23 Sekunden abschnitt. Die Messwerte sind allerdings nur bedingt eins zu eins vergleichbar, da die Tests einerseits auf unterschiedlichen Rechnern liefen und andererseits jedes Programm seine eigenen Stärken und Schwächen hat.

FME zum Beispiel unterstützt sehr viele Formate, bietet dafür aber auch eine sehr umfangreiche (und komplexe) Benutzeroberfläche („Platform (FME)“, o. J.). QGIS ist als Open Source-GIS äußerst flexibel, hat aber andere Schwerpunkte und nutzt häufig interne Plugins für Konvertierungen („QGIS User Guide — QGIS Documentation“, o. J.). Der FZKViewer ist stärker auf 3D-Stadtmodelle zugeschnitten und zeigt daher bei CityGML-konformen Aufgaben seine Stärken („FZKViewer“, 2025). Gerade beim Handling der CityGML-Datensätze und bei der Übernahme von Texturen, sind die Unterschiede zu den etablierten Lösungen noch am stärksten zu erkennen. Auch die problembehaftete GDAL-Integration zeigt, dass hier noch Verbesserungspotenzial besteht, um umfassend alle Dateiformate in einer zuverlässigen EXE-Version abzudecken.

Zusammenfassung der wichtigsten Aspekte

Insgesamt lässt sich festhalten, dass die zentrale Idee des Geodatenkonverters – also das schnelle und flexible Umwandeln, Mergen und Transformieren von Open Data Geodaten – durchweg funktioniert und in vielen Fällen eine gute Alternative zu großen GIS-Lösungen darstellt. Vor allem Nutzerinnen und Nutzer, die häufig bestimmte Rasterdaten konvertieren und mehrere Dateien zusammenführen, können von den schnellen Abläufen profitieren. Auch ist hervorzuheben, dass kein anderes Programm einen Zuschnitt von Daten, wie in diesem Fall mit DOK/DTK Dateien, bewältigen kann. Gleichzeitig bleibt aber klar, dass noch einige Details verbessert werden müssen, um das Tool in vollem Umfang universell einsetzen zu können.

5.2 Fazit und Ausblick

In dieser Arbeit wurde ein Geodatenkonverter in Python entwickelt, der speziell auf Open Data zugeschnitten ist. Das Hauptziel war, unterschiedliche Raster- und Vektordaten zuverlässig in gängige Formate zu überführen und ihnen bei Bedarf ein passendes Koordinatensystem zuzuweisen. Dabei zeigt sich, dass die grundlegenden Funktionen – wie das Konvertieren von GeoTIFF nach ASCII oder das Zusammenführen mehrerer Dateien – sehr zuverlässig ausgeführt werden und in vielen Fällen sehr kurze Bearbeitungszeiten aufweisen.

Besonders hilfreich ist, dass durch die gewählte Architektur auch größere Datenmengen (z. B. hochauflösende Orthophotos) effizient verarbeitet werden können. Die Messungen haben belegt, dass die meisten Vorgänge im einstelligen Sekundenbereich erledigt werden, wobei erwartungsgemäß, mehr Speicher und Zeit bei sehr großen Rasterdaten benötigt wird. Gleichzeitig sorgt der „Automatische Vorgang“ dafür, dass selbst Nutzerinnen und Nutzer ohne tiefes GIS-Hintergrundwissen den Großteil des Programms nutzen können. Für speziellere Anforderungen (z. B. 3D-Daten, CityGML-Umwandlungen) bietet der „Manuelle Vorgang“ mehr Flexibilität, kommt aber noch nicht in allen Szenarien komplett ohne Workarounds aus.

Den größten Verbesserungsbedarf gibt es aktuell bei der Integration der GDAL-Bibliothek, insbesondere wenn eine lauffähige EXE für Windows erstellt werden soll. Auch das Handling von Texturen bei 3D-Modellen ist noch nicht perfekt gelöst, da bestimmte Formate (z.B. CityGML) nur unvollständig in OBJ oder DXF übertragen werden. Insgesamt hat sich jedoch gezeigt, dass viele dieser Schwierigkeiten durch alternative Ansätze (z. B. Fallback über GeoJSON) zumindest vorläufig umgangen werden können.

Damit der Geodatenkonverter langfristig eine vollwertige Lösung für verschiedenste Open Data Formate wird, sind weitere Tests unter unterschiedlichen Betriebssystemen und mit diversen Bibliotheksversionen sinnvoll. Zudem könnte die Unterstützung für 3D-Formate und ihre Texturen noch ausgebaut werden, um Anwenderinnen und Anwendern aus Bereichen wie Stadtplanung, Architektur und Geovisualisierung einen durchgängigeren Workflow zu ermöglichen. Denkbar wären auch zusätzliche Schnittstellen zu Web-Services oder Cloud-Umgebungen, sodass Geodaten direkt online bezogen und weiterverarbeitet werden können. Durch diese Schritte kann das hier vorgestellte Programm zu einem vielseitigen und robusten Werkzeug reifen, das den offenen Zugang zu Geodaten weiter vereinfacht.

Danksagung

An dieser Stelle bedanke ich mich herzlich bei all jenen, die durch ihre vielfältige Unterstützung zum Gelingen dieser Bachelorarbeit beigetragen haben.

Mein herzlicher Dank geht an Herrn Prof. Dr. Ludwig Hoegner, der meine Bachelorarbeit betreut und begutachtet hat, sowie für die hilfreichen Anregungen bei der Erstellung der Arbeit.

Besonders bedanken möchte ich mich bei Herrn Thomas Meier, der mir mit seiner umfassenden Unterstützung und Fachkompetenz zur Seite stand und mir mit seiner konstruktiven Kritik stets weitergeholfen hat.

Anhang

Der Anhang befindet sich auf dem USB-Stick.

Folgende Inhalte befinden sich auf diesem:

- Programm im zugehörigen Ordner als .exe
- PDF mit den Messergebnissen
- Programm für eine IDE mit den einzelnen Skripten
- Präsentation
- Testdateien können über die Hyperlinks im Programm installiert werden

Literaturverzeichnis, Abbildungen, Formeln, Tabellen

Bundesverwaltungsamt. 2024. „Open Data Handbuch“. Kompetenzzentrum Open Data (CCOD). https://www.bva.bund.de/SharedDocs/Downloads/DE/Behoerden/Beratung/Methoden/open_data_handbuch.pdf?__blob=publicationFile&v=2.

„FME by Safe Software Community | Community“. o. J. Zugegriffen 22. Oktober 2024. https://community.safe.com/. Zuletzt besucht am 27.02.2025.

„FZKViewer“. 2025. https://www.iai.kit.edu/1648.php. Zuletzt besucht am 27.02.2025.

„GDAL Developers“. o. J. https://gdal.org/en/stable/. Zuletzt besucht am 27.02.2025.

„Geopandas“. o. J. https://geopandas.org/en/stable/. Zuletzt besucht am 27.02.2025.

Goodchild, Michael F. 2007. „Citizens as Sensors: The World of Volunteered Geography“. *GeoJournal* 69 (4): 211–21. https://doi.org/10.1007/s10708-007-9111-y.

Goodchild, Michael F. 2010. „Twenty Years of Progress: GIScience in 2010“. *Journal of Spatial Information Science*, Nr. 1 (Juli), 3–20. https://doi.org/10.5311/JOSIS.2010.1.2.

Gröger, Gerhard, Thomas H. Kolbe, Claus Nagel, und Karl-Heinz Haefele. o. J. „OGC City Geography Markup Language (CityGML) Encoding Standard“. Open Geospatial Consortium, Inc. Zugegriffen 21. Dezember 2024. https://doi.org/10.62973/12-019.

Hofmann-Wellenhof, Bernhard, Herbert Lichtenegger, und James Collins. 1997. *Global Positioning System: Theory and Practice*. Vienna: Springer Vienna. https://doi.org/10.1007/978-3-7091-3297-5.

LDBV. 2021. „OpenData Bayern“. 2021. https://geodaten.bayern.de/opengeodata/index.html.

Maling, D.H. 1992. *Coordinate Systems and Map Projections (2nd ed.)*. https://www.google.de/books/edition/Coordinate_Systems_and_Map_Projections/tcL-BAAAQBAJ?hl=de&kptab=editions&gbpv=1.

„Numpy“. o. J. https://numpy.org/doc/1.26/. Zuletzt besucht am 27.02.2025.

Olasz, A., B. Nguyen Thai, und D. Kristóf. 2016. „A NEW INITIATIVE FOR TILING, STITCHING AND PROCESSING GEOSPATIAL BIG DATA IN DISTRIBUTED COMPUTING ENVIRONMENTS“. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* III–4 (Juni):111–18. https://doi.org/10.5194/isprsannals-III-4-111-2016.

„open bydata (byte)“. o. J. Zugegriffen 30. Januar 2025. https://www.byte.bayern/was-wir-machen/open-data-bayern. Zuletzt besucht am 27.02.2025.

„Open Data Bayern: Eigene Präsenz für Kommunen“. o. J. Zugegriffen 30. Januar 2025. https://www.egovernment.de/open-data-bayern-eigene-praesenzen-fuer-kommunen-a-f10f8ed731010a352376d4242b8e8562/. Zuletzt besucht am 27.02.2025.

„Pandas“. o. J. https://pandas.pydata.org/docs/. Zuletzt besucht am 27.02.2025.

„Platform“. o. J. FME by Safe Software. Zugegriffen 22. Oktober 2024. https://fme.safe.com/platform/. Zuletzt besucht am 27.02.2025.

Publications Office of the European Union., Capgemini Invent., und European Data Portal. 2020. *Open Data Best Practices in Europe’s Top Performers: Ireland, Spain and France.* LU: Publications Office. https://data.europa.eu/doi/10.2830/05271.

„Pyproj“. o. J. https://pyproj4.github.io/pyproj/stable/. Zuletzt besucht am 27.02.2025.

„PyQT“. o. J. https://www.riverbankcomputing.com/static/Docs/PyQt5/introduction.html. Zuletzt besucht am 27.02.2025.

„QGIS Community“. 2022. QGIS.Org Blog. 16. Juni 2022. https://blog.qgis.org/tag/qgis-community/. Zuletzt besucht am 27.02.2025.

„QGIS User Guide — QGIS Documentation documentation“. o. J. Zugegriffen 22. Oktober 2024. https://docs.qgis.org/3.34/en/docs/user_manual/index.html. Zuletzt besucht am 27.02.2025.

Rahmatulloh, Alam, Bambang Tri Handoko, Rahmi Nur Shofa, und Irfan Darmawan. 2022. „GeoJSON Implementation for Demographic and Geographic Data Integration Using RESTful Web Services“. In *2022 Seventh International Conference on Informatics and Computing (ICIC)*, 1–6. Denpasar, Bali, Indonesia: IEEE. https://doi.org/10.1109/ICIC56845.2022.10006893.

„Rasterio“. o. J. https://rasterio.readthedocs.io/en/stable/. Zuletzt besucht am 27.02.2025.

„Rasterio Reprojection“. o. J. https://rasterio.readthedocs.io/en/stable/topics/reproject.html. Zuletzt besucht am 27.02.2025.

Renz, Matthias. 2014. *Kombination von Vektor- und Rasterdaten*.

„Shapely“. o. J. https://shapely.readthedocs.io/en/stable/. Zuletzt besucht am 27.02.2025.

Silva-Coira, Fernando, José R. Paramá, Susana Ladra, Juan R. López, und Gilberto Gutiérrez. 2020. „Efficient Processing of Raster and Vector Data“. Herausgegeben von Praveen Rao. *PLOS ONE* 15 (1): e0226943. https://doi.org/10.1371/journal.pone.0226943.

Sneyder, John P. 1987. *Map Projections - A Working Manual*.

Steiniger, Stefan, und Andrew J.S. Hunter. 2013. „The 2012 Free and Open Source GIS Software Map – A Guide to Facilitate Research, Development, and Adoption“. *Computers, Environment and Urban Systems* 39 (Mai):136–50. https://doi.org/10.1016/j.compenvurbsys.2012.10.003.

Weidmann, Nils B., und Kristian Skrede Gleditsch. 2020. „Geodaten und deren Analyse in der Politikwissenschaft“. In *Handbuch Methoden der Politikwissenschaft*, herausgegeben von Claudius Wagemann, Achim Goerres, und Markus B. Siewert, 419–38. Wiesbaden: Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-16936-7_23.

Wilson, Tim. 2008. „OGC Keyhole Markup Language (KML) Encoding Standard“, Nr. 2.2.0.

Hochschule München University of Applied Sciences

Name:	Bürger
Vorname:	Paul
Studiengang :	Bachelor Geoinformatik und Navigation
Matrikel-Nr.:	xxxxxxxxxxxx
Betreuer:in:	Prof. Dr. Ludwig Hoegner
Betreuer:in Extern:	Dipl. Ing. (FH) Thomas Meier (LDBV)

Hiermit erkläre ich, dass ich die Abschlussarbeit selbstständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel benutzt, sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

München, den 07.03.2025 — Ort, Datum

Paul Bürger — Unterschrift