University of Zurich[UZH]

Photogrammetrical UAV-based Investigation of Torrents

GEO 510 Master's Thesis

Author
Gregor Rafael Schmucki
16-716-722

Supervised by
Dr. Perry Bartelt (bartelt@slf.ch)
Dr. Yves Bühler (buehler@slf.ch)
Christoph Graf (christoph.graf@wsl.ch)
Prof. Dr. Christian Huggel

Faculty representative
Prof. Dr. Christian Huggel

30.04.2022
Department of Geography, University of Zurich

Supervisors:

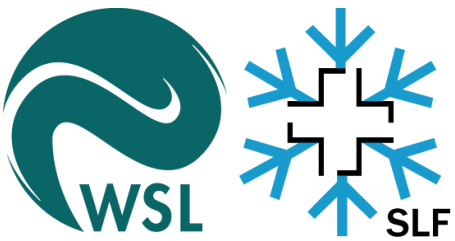
Dr. Perry Bartelt
Swiss Federal Institute for Forest, Snow and Landscape Research WSL
Institute for Snow and Avalanche Research SLF
Research Unit: Alpine Environment and Natural Hazards
Research Group: RAMMS Rapid Mass Movements

Dr. Yves Bühler
Swiss Federal Institute for Forest, Snow and Landscape Research WSL
Institute for Snow and Avalanche Research SLF
Research Unit: Alpine Environment and Natural Hazards
Research Group: Alpine Remote Sensing

Christoph Graf
Swiss Federal Institute for Forest, Snow and Landscape Research WSL
Research Unit: Mountain Hydrology and Mass Movements
Research Group: Torrents and Mass Movements

Prof. Dr. Christian Huggel
University of Zurich
Department of Geography
Research Group on Environment and Climate: Impacts, Risks and Adaptation (Eclim)

The Master's Thesis at hand was conducted in close collaboration with the WSL Institute for Snow and Avalanche Research SLF in Davos.



Swiss Federal Institute for Forest, Snow and Landscape Research WSL
Institute for Snow and Avalanche Research SLF
RU: Alpine Environment and Natural Hazards
RG: RAMMS Rapid Mass Movements

Acknowledgements

First and foremost, I would like to thank all my supervisors. Dr. Perry Bartelt from the Institute for Snow and Avalanche Research (SLF) for his interest and support in writing a dissertation in the research field of drones and torrent investigations. I would also like to thank Christoph Graf from the Swiss Federal Institute for Forest, Snow and Landscape Research WSL. He introduced me to the subject of assessing torrents exposed to debris flow hazards. It was always very interesting to discuss different geomorphological phenomena together in the field. He also offered me excellent support and constructive discussions during the development of the new algorithms and the writing processes. Prof. Dr. Christian Huggel supported me as a member of the Department of Geography. He has greatly contributed to narrowing down my work and embedding it in the context of hazard assessment. I am grateful for the support of Dr. Yves Bühler. He supported me with UAV flights and subsequent data processing. He also gave me an introduction to the OBIA software eCogniton. Furthermore, I received great support and encouragement from Andreas Stoffel. I had many discussions with him to optimise my GIS workflows. He was always very open and interested in optimising the accuracy of the structure-from-motion processing and point cloud comparisons. I would also like to thank Mauro Marty from the Swiss Federal Institute for Forest, Snow and Landscape Research WSL. He introduced me to the work with LAStools and supported me a lot in the development of the LAStools-based classification routine of photogrammetric point cloud data. He also organised and conducted the LiDAR-based UAV flight in the Arelen catchment and provided me with advice and support to assess the quality of the classified point clouds. Further, I would like to thank Dr. Andrin Caviezel for his support in optimizing my figures and Sven Buchmann for his assistance with the R-coding.

Prof. Dr. Stuart Lane from the University of Lausanne has been very supportive in dealing with systematic errors and change detection in photogrammetrically acquired data.

Furthermore, I received support in various discussions about my thesis with Prof. Dr. Reynald Delaloye (University of Fribourg), Dr. Mauro Fischer (University of Bern), Dr. Roland Kaitna and Philipp Aigner (BOKU Vienna).

Finally, I would also like to thank my family and all my friends who supported me during this challenging time.

Abstract

Debris flows are a major hazard in mountainous regions. Cost-effective, long-term studies of debris flow torrents, however, are rare, leading to considerable uncertainties in hazard mitigation methods. Here, we address the question if cost effective remote sensing techniques can be applied along the hazard assessment of mountain torrents and further to gather accurate, long-term information on the evolution of the catchment. Active torrents prone to debris flows are often devoid of vegetation and hence can be well captured with photogrammetrically derived methods based on uncrewed aerial vehicle (UAV) surveys. The possibility of automatic extraction of torrent parameters from high resolution terrain models, such as cross-section area, gradient, etc., is investigated. The presented methodology yields continuous and automatically derived parameters along the torrent, which is a major advantage compared to discretized field surveys. Cross-validation thereof with field measurements show strong congruence. Those parameters are highly accurate along sharply incised sections with strong limitations along sections with steep adjacent slopes and/or dense vegetation. A further important result of this work is that multiple UAV acquisitions enable a highly accurate mass-balance characterization of watersheds. After a successful co-registration, the M3C2 algorithm provides valuable results, from which debris flow events can be back-calculated and thus support the determination of site-specific model input parameters. A complete debris flow hazard assessment procedure - missing in current guidelines - is presented. We show that such assessments strongly benefit from UAV data and a subsequent automated parameter extraction: The original point clouds allow to obtain a first overview of the torrent. Derived parameters provide further insights such that key sections and weak spots can be identified and precisely evaluated in the field. The study highlights that the extraction of the true location of the ground poses the key challenge. We show that photogrammetric routines are severely limited by dense vegetation. We find that UAV data can contribute to a comprehensive, reproducible, and objective assessment of torrent processes and predispositions. However, ground-truthing fieldwork remains essential and further research on remote sensing supported hazard assessment of debris flow prone torrents is highly indispensable.

Zusammenfassung

Murgänge in Gebirgsregionen weisen ein hohes Gefahrenpotenzial auf. Kosteneffiziente Langzeitstudien über Murgänge sind selten. Das führt beim Gefahrenmanagement zu erheblichen Unsicherheiten. Wir gehen der Frage nach, und ob und wie kostengünstige Fernerkundungstechniken bei der Gefahrenbeurteilung von Wildbächen eingesetzt werden können, ob damit genaue, langfristige Informationen über die Entwicklung des Einzugsgebiets hergeleitet werden können. Aktiv von Murgängen geprägte Wildbäche sind oft vegetationsfrei und lassen sich daher gut mit photogrammetrischen unbemannten Luftfahrzeugen vermessen (Englisch: uncrewed aerial vehicle (UAV)). Es wird die Möglichkeit der automatischen Extraktion von beurteilungsrelevanten Parametern auf der Basis von hochauflösten Geländemodellen, wie z.B. Querschnittsfläche, Gerinneneigung etc., untersucht. Die vorgestellte Methodik liefert kontinuierliche und automatisch abgeleitete Parameter entlang des Wildbachs, was einen grossen Vorteil gegenüber diskretisierten Feldvermessungen darstellt. Der Quervergleich mit Feldmessungen zeigt eine hohe Kongruenz. Die erhobenen Parameter sind entlang stark eingeschnittener Gerinneabschnitte sehr genau; entlang von Abschnitten mit steilen Hängen und/oder dichter Vegetation jedoch deutlich einschränkt. Ein weiteres wichtiges Ergebnis dieser Arbeit zeigt, dass mehrere UAV Befliegungen eine exakte Charakterisierung der Massenbilanz von Einzugsgebieten ermöglichen. Nach erfolgreicher Co-Registrierung liefert der M3C2-Algorithmus wertvolle Ergebnisse, die es ermöglichen Murgangereignisse nachzurechnen und standortspezifische Modellinputparameter zu bestimmen. Es wird ein umfassendes Verfahren zur Gefahrenbeurteilung von Murgängen vorgestellt, das in derzeitigen Leitlinien fehlt. Wir zeigen, dass Gefahrenbeurteilungen stark von UAV Daten und einer damit einhergehenden automatischen Parameterextraktion profitieren: Die Visualisierung der Punktwolken ermöglicht es, einen ersten Überblick über den Wildbach zu erhalten. Die abgeleiteten Parameter liefern weitere Erkenntnisse, so dass Schlüsselabschnitte und Schwachstellen identifiziert und vor Ort präzise bewertet werden können. Die Arbeit zeigt, dass die Extraktion der exakten Höhenlage des Bodens eine grosse Herausforderung darstellt. Wir zeigen, dass photogrammetrische Erhebungen durch dichte Vegetation stark eingeschränkt werden. Wir stellen fest, dass UAV-Daten zu einer umfassenden, reproduzierbaren und objektiven Bewertung von Wildbachprozessen und -veranlagungen beitragen können. Feldbasierte Beurteilungen bleiben jedoch unentbehrlich. Weitere Forschung zur fernerkundungsgestützten Gefahrenbeurteilung von Wildbächen ist unerlässlich.

Abbreviations:

CP: *Control Point*
DoD: *Digital Terrain Model of Difference*
DTM: *Digital Terrain Model*
DSM: *Digital Surface Model*
GCP: *Ground Control Point*
GNSS: *Global Navigation Satellite System*
GSD: *Ground Sampling Distance*
ICP: *Iterative Closest Point Algorithm*
IMU: *Inertial Measurement Unit*
LiDAR: *Light Detection and Ranging*
M3C2: *Multiscale Model to Model Cloud Comparison*
PAEK: *Polynomial Approximation with Exponential Kernel*
PPK: *Post-processed Kinematic*
RTK: *Real-Time Kinematic*
TIN: *Triangulated Irregular Network*
VTOL: *Vertical Take-off and Landing*
UAV: *Uncrewed Aerial Vehicle*

Contents

List of Figures

List of Tables

Introduction

Debris flows are gravitationally driven mass movements containing a mixture of rocky debris and water. Depending on the mixture properties, they can reach high velocities ($>$ 10 m/s), propagate long runout distances up to several kilometeres and exert extreme forces ($>$ 30 kPa) on buildings and other constructions such as bridge abutments (Hürlimann et al., 2019). Debris flows are typically triggered by intense rainfall events and associated runoff processes. They are considered as one of the most unpredictable and therefore destructive landslide processes known in mountain regions (Hungr et al., 2001; Iverson, 1997; Jakob and Hungr, 2005; Takahashi, 2007). The damage caused tends to increase with the amount of sediment transported to the cone during an event, especially when water and sediment leave the channel (Rickenmann, 2014). This can lead to large inundated areas and the possibility of widespread destruction. The assessment and planning of protective measures in torrent catchments against debris flow actions is therefore an important part of an integral risk management of natural hazards in mountain regions.

There are several challenges in debris flow science. One of the fundamental problems is the large return period of events. For example, compared to snow avalanches, where event activity can be monitored effectively on a yearly interval, debris flows are comparatively rare. Often, several years can pass between events. Seldom is the disposition of sediments in the torrent known in detail before the event. This hinders a quantitative understanding and assessment of debris flow formation and propagation. Moreover, little information is available concerning the precipitation rates leading to an event. As a consequence, research results are often site specific (Graf et al., 2019) and cannot be generalized. In addition, there are hardly any reviews summarising the current state of knowledge.

Repeated photogrammetric uncrewed aerial vehicle (UAV) surveys enable to characterise the development of landforms over time (Cucchiaro et al., 2018; Niculia et al., 2020; Passalacqua et al., 2015). Based on available spatial data, numerical simulation models of debris flows can be further developed and their parameters determined with greater accuracy (Graf et al., 2019). Hazard-relevant parameters such as runoff area, discharge, velocities and probability of occurrence can be estimated more easily (Hürlimann et al., 2019). To improve the simulations and to understand the sensitivity of the models, systematic case studies are fundamental (Frank et al., 2017). In this work we investigate how repeated photogrammetric UAV flights can be used to improve debris flow hazard mitigation. However, debris flow catchments often extend between non-vegetated high-altitude starting zones to densely vegetated runout zones. Because debris flow prone catchments need to be assessed in terms of their overall hazard, both zones must be analyzed with equal accuracy, especially for numerical modelling applications. To properly capture the morphology of the landscape and the associated change over time, the point cloud data obtained with the UAVs must be classified into ground and non-ground (vegetated) points. In this work, we have developed and tested two classification algorithms for photogrammetric data (see section 4.3).

We use the classified point clouds derived from UAVs to quantify volume changes along the proposed M3C2-based workflow of Bernard et al. (2021). To accurately calculate differences, systematic errors in the point cloud data of both acquisitions may need to be corrected (see section 4.5). From classified point cloud derivatives, a high resolution DTM can be generated. Based on this high resolution DTM, the study aims to systemically characterize debris flow prone torrents as well as to extract torrential properties, which can be used along the hazard assessment workflow. The focus is mainly on geometrical parameters, which are conventionally extracted in the field during geomorphological assessment of the torrent (see section 4.4). The automatically derived parameters can be used to plan field campaigns, especially to assess regions of the torrent which demand special attention. UAVs are intended to support ground-based assessment procedures, not to replace them.

There are decisive differences along the hazard analyses of different countries (Jakob, 2005). For example in Switzerland a few geomorphological assessment approaches have been developed to estimate the magnitude and erosive capacity of debris flows (Frick et al., 2008; Gertsch, 2009; Spreafico et al., 1996). None of these methods has become a standard in engineering practice mainly due to the absence of clear instructions for the hazard assessment procedure in the current Swiss federal guideline (BAFU, 1997). In chapter 6, we discuss where and how the newly developed approaches can be implemented into the pre-existing hazard assessment workflows (eg. Hunziker et al., 2021).

UAVs are likely to bring along many advantages and introduce a major step towards a more reproducible and objective hazard assessment. However, fieldwork will remain a mandatory contribution within the assessment procedure. The available sediment, hill slope processes and state of constructions can only be assessed properly based on accurate fieldwork. As a conclusion, we overview the possibilities and limitations in the context of overall management of natural hazards in mountainous regions.

Torrential Processes, Assessment Methodologies and UAVs: Open Research Questions

This chapter first summarizes torrential processes and existing assessment methodologies currently used for debris hazard mitigation in Switzerland. We then provide an overview of the present state of UAV technology and show how it can be introduced into torrent assessment procedures. As we aim to further develop current assessment methods using newly available UAV remote sensing data, we present the current state of the art of UAVs and their processing procedures to produce high-resolution digital terrain models (DTMs). DTMs generated from UAV data have a high spatial resolution and allow to acquire data with a high temporal resolution, for example shortly after a debris flow event. The use of point clouds for debris flow torrent assessments requires in-depth knowledge of classification schemes for photogrammetric point cloud data. In a final step, we elaborate the central research questions arising from the application of UAVs in debris flow torrent assessment.

There are only a few published studies (eg. Adams et al., 2016; de Haas et al., 2020; Keilig et al., 2019; Tobler et al., 2014) dealing with volume calculations of debris flow events. Adams et al. (2016) performed volume calculations, but only within the area where erosion and deposition occurred. By omitting vegetated areas, they did not encounter the problem of classifying vegetated areas in steep alpine terrain. Since we want to address this problem, we investigated the state of the art in classifying photogrammetric point cloud data. Working with point cloud data enables new methods for assessing change along landforms. We present the state of the art in geomorphological change detection. We then outline research gaps and formulate our research questions.

2.1. Torrential Processes

Torrents are natural, permanently or intermittently flowing channels with rapidly and strongly changing discharge conditions. Torrent geometries and length profiles are irregular and gradients are steep (5 - 10%). The bed morphology contains large boulders and rocks mixed with finer sediments. A large proportion of the sediment is contributed from the adjacent slopes. Floods can rise rapidly and erode large amounts of these solids (Bergmeister et al., 2009). Depending on discharge rates during a given event, more or less solids are mobilised. A torrent system, as shown in the Figure 2.1 a, is based on spatial and functional geomorphic elements. Spatially, the three main geomorphic elements are: catchment (slopes, channels), torrent, and debris flow cone. The corresponding natural hazard classification of these elements are the starting zone, the transition zone and the runout zone. Functional components include sediment processing, sediment delivery to the channel, sediment mobilisation or transport in the channel, and subsequently deposition and redistribution on flat sections in the channel and outside the channel (Rimböck et al., 2013).

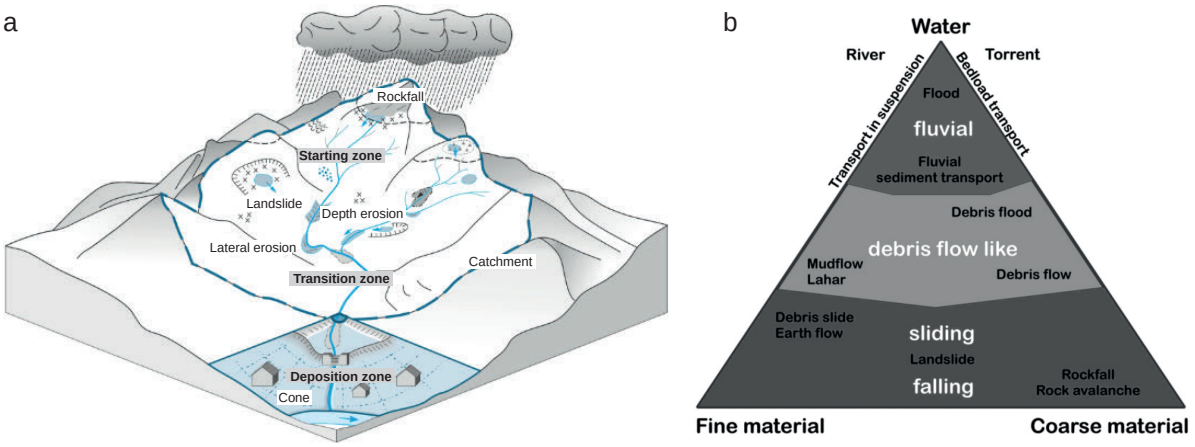


Figure 2.1. a) A torrential catchment is characterized through different system components and processes (HADES, 1992, modified) b) Three phase diagram by Rickenmann (2001) after Phillips and Davies (1991).

There is a wide spectrum of different transport processes. Generally it can be distinguished between normal hydrological discharge, bedload transport, hyperconcentrated flow and debris flow. The diversity of processes can be illustrated in a three-component phase diagram by Phillips and Davies (1991) (see Figure 2.1 b). A debris flow has similarities in terms of the components and flow characteristics of a flood, landslide and rockfall event. Due to the complex interaction of the different processes, the flow and deposition behaviours of debris flows are not yet fully understood (Rickenmann, 2001). From a modelling standpoint, it is important to emphasize that each transport process is characterized by different discharge parameters, leading to considerable uncertainties in the modelling approach.

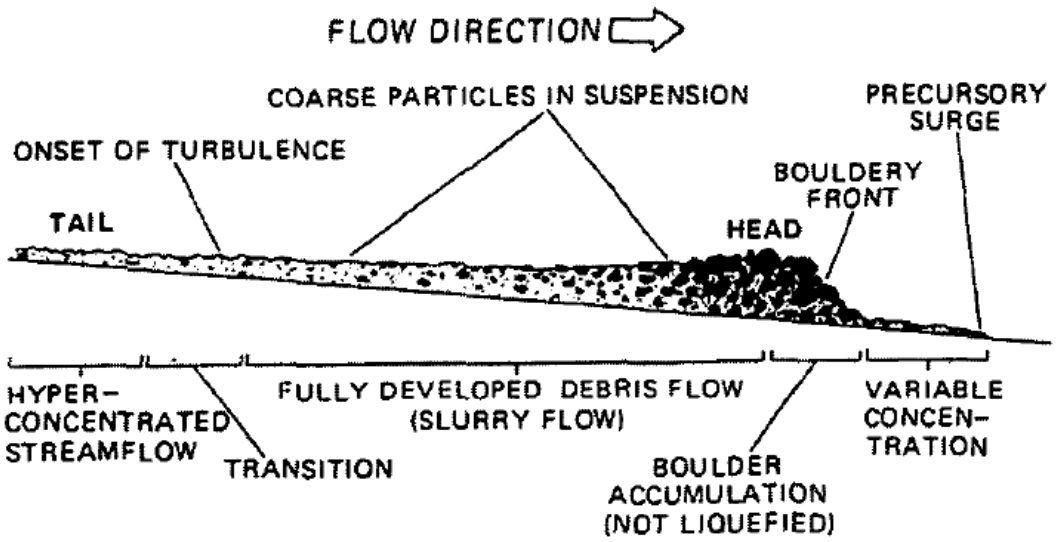


Figure 2.2. Typical structure of a debris flow surge with large boulders in the front (Pierson, 1986).

Varnes (1978) describes a debris flow as a rapid mass movement consisting of a mixture of granular solids, water and air, whose flow characteristics change according to water, clay content, sediment size and sorting. The typical bouldery front of the debris flow results in large impact forces (Pierson, 1986)

(see Figure 2.2). Johnson and Rahn (1970) reports it to be an intermediate form between landslide and waterflood, but with distinct mechanical properties from these two processes. Rickenmann (2014) notes that there are many different definitions to characterize debris flows. A common distinction is to separate hillslope debris flows from torrent-based debris flows. Hillslope debris flows, similar to landslides, are not confined to a channel and occur in open terrain. In this thesis, we focus on channelised debris flow processes. Hungr et al. (2001) mentioned that the distinction of mud flows and debris flows is not of primary importance, as both show similar hyperconcentrated flow properties. Costa (1984) suggested a minimal solid concentration of 80% to define the process as an actual debris flow. Pierson (2005) describes a hyperconcentrated flow as a water flood acquiring fine-grained suspended sediments through erosion and entrainment.

During a debris flow event, often parts of a debris flow dilute and are better characterized as a debris flood. Hungr et al. (2001) suggested that the naming of the process should describe the behaviour of the process in the central part of the deposition zone. Debris flows are characterized to produce thicker, lobate and more hummocky deposits compared to debris floods. A characteristic phenomenon of debris flows is the deposition of levees along the channel margins. Levees are not observed in debris floods as they are not characterized through the longitudinal sorting and bouldery front.

2.2. Disposition and Triggers of Debris Flows

The reasons for the formation of a debris flow are manifold. Water plays an important role in triggering a debris flow (Rickenmann, 2001). Debris flows are often caused by the accumulation of surface water, the liquefaction of loose material or the breaking up of debris jams in the stream bed as a result of a previous landslide. If there has been a long period of precipitation and the soil saturation is strikingly high, even a small rainfall event can trigger a debris flow (Zimmermann, 1997). Furthermore, the slope of the terrain limits the triggering of an event. To trigger a debris flow, the slope must be about 25 to 30%. If there are favoring factors such as narrows, woods or landslides that lead to a jam, debris flows between 15 and 25 % may also be triggered (Rickenmann, 1995).

Zimmermann (1997) introduced a probability-based concept to characterise the disposition of a torrential system to release debris flows (see Figure 2.3). The triggering probability is defined by three factors. The first factor is the *basic disposition* which can be understood as the overall properties of the catchment, including relief, slope angle, climate, vegetation and, most importantly, geomorphological setting. The basic deposition is therefore constant over a long period of time. The so-called *variable disposition* changes over time. For example, the variable disposition is higher during snowmelt in spring or after long periods of rain. The increased amounts of loose sediments can be understood as an increase in variable disposition. Finally, *stress events* such as precipitation events, GLOFS and spontaneous landslides put additional stress on the system, acting as potential triggers. If the stress and disposition are large enough at the same time, an event occurs.

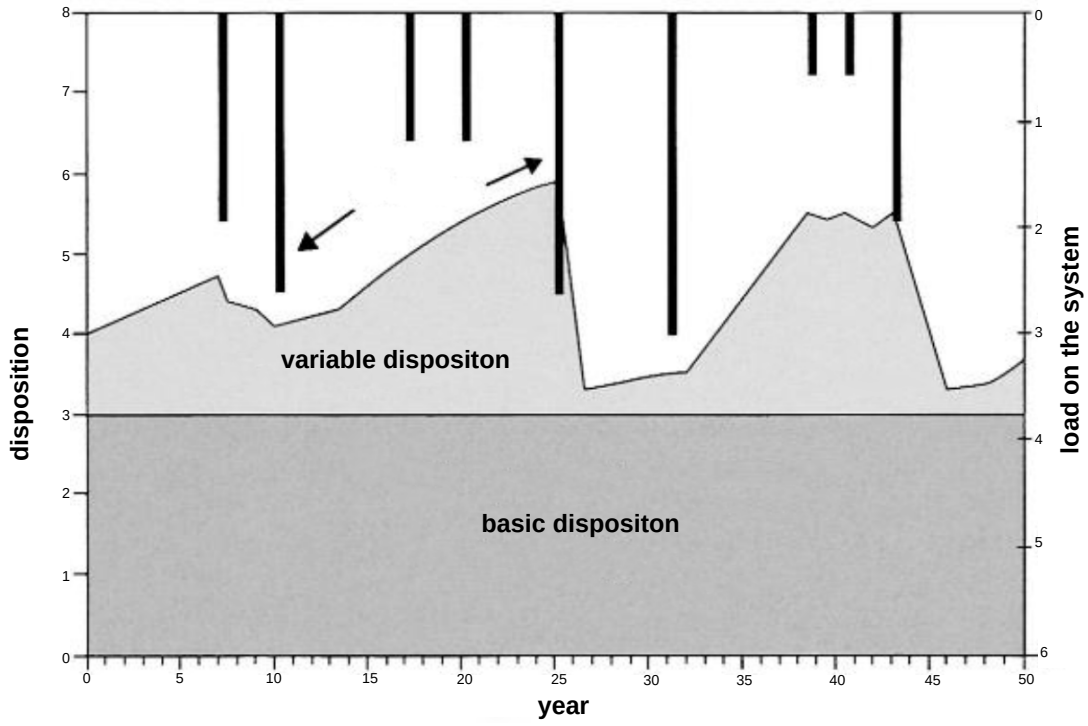


Figure 2.3. Disposition of a torrential system differentiating between basic and variable disposition by Zimmermann (1997, modified).

2.3. Assessment of Mountain Torrents

An integral hazard assessment of torrents is based on three elements: (1) Fieldwork to analyse the state of the torrent, (2) evaluation of past historical events, if there are any, and (3) scenario-based calculations based on empirical and/or numerical models (Petracheck and Kienholz, 2003). The key problem in each stage of the assessment process is to estimate possible debris flow volumes and discharge rates. Numerical simulation models have become an increasingly important tool in the assessment processes (Meyrat et al., 2022). Although it is crucial to recognise that model results are only a representation of the real world, the model results, whenever possible, should be calibrated with results from previous events. There is a close interplay between all three elements of the assessment process. If no event data is available, model results must be interpreted very carefully. In this thesis, we intend to answer the question whether UAVs can be effectively used in the assessment process.

2.3.1. Estimating Debris Flow Volume and Discharge Rates

In recent decades, many different attempts have been made to estimate debris flow volumes for a given catchment. Empirical equations based on some important morphometric characteristics of the catchment have been established to estimate the magnitude of a possible event. Rickenmann (1999) states that these equations may overestimate the debris flow volume by a factor of 100. He therefore recommends a geomorphological assessment of the sediment potential in the field.

The main objective is always to determine the volume of bedload material (magnitude) of a given event scenario and the associated frequencies. Other relevant parameters are peak discharge, velocity,

runout distance, erosion and deposition (Hungr et al., 1984). For the volume estimation, parameters such as the possible peak flood [m^3/s] (water and solids) and the amount of solids [m^3] must be known. Discharge data or descriptions of previous debris flow events are often only sparsely available. In such cases, experts must rely on qualitative and semi-quantitative estimates. For this purpose, hydrological models such as rainfall-runoff models provide more or less reliable runoff estimates. However, estimating bedload transport rates and the total amount of bedload in solids remains extremely difficult (Kienholz et al., 2010). Available information on past events often forms the most reliable indication of bedload transport rates (Rickenmann, 1999). Each torrent catchment has its own character (disposition, history, conditions for future development). In order to meet the uniqueness of each torrent, sufficient knowledge must be acquired. According to Kienholz et al. (2010), this knowledge can only be achieved through fieldwork and combining different diagnostic methods. Remote sensing and GIS-based methods can help to improve the understanding of local processes. However, in most cases, on-site assessment remains indispensable. It is therefore essential that fieldwork is carried out by experienced experts who have a great understanding of the prevailing processes.

2.3.2. Structured Geomorphologic Assessment Methods

Based on the analysis of the large debris flow events in the summer of 1987, it was shown that two thirds of the debris flow volume resulted from bed and lateral erosion (Rickenmann, 1995). Further large volumes of debris are to be expected behind large residual boulders, wood and steep neighbouring slopes prone to landslide processes. Based on this analysis, the main components which need to be determined (estimated) during a geomorphological assessment are debris flow volumes in relation to possible erosion levels.

A few structured, geomorphological assessment approaches have been developed in Switzerland to estimate the magnitude and erosive capacity of debris flows (Frick et al., 2008; Gertsch, 2009; Gertsch et al., 2012; Spreafico et al., 1996). None of these methods has become a standard in engineering practice. Each company has its own customised approach, probably combining different elements of the different approaches, which complicates the comparison of assessment results.

Both Gertsch (2009) and Frick et al. (2008) divide the torrent into homogeneous channel sections or segments. Homogeneous sections are formed taking into account criteria such as the characteristics of the torrent bed and the adjacent slopes. An important parameter is the torrent functionality (erosion, transit, bedload exchange, deposition) and the average gradient of the slope. In both methods, the two sediment sources (bedload and adjacent slopes) are separated. A distinction is made primarily between material-limited torrent beds, such as shallow, soil-covered bedrock, and material-unlimited (e.g. huge fluvio-glacial deposits). Both methods require a high degree of experience of the processes occurring in torrents.

2.3.2.1. Method Gertsch

The Gertsch method focuses on extreme events with recurrence intervals of more than 100 years. The method is based on the analysis of 58 events in various mountain streams that occurred in Switzerland after 1987. The method is validated by 20 events and 23 existing technical reports.

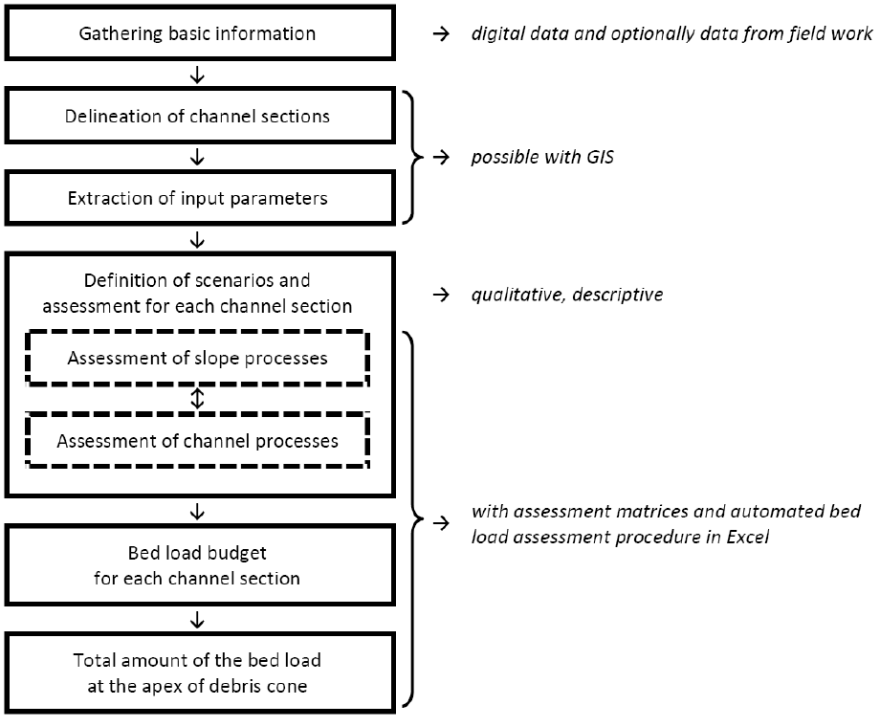


Figure 2.4. Proposed workflow of Gertsch method. Channel sections are delineated based on basic information. Channel and slope processes are assessed individually considering predefined matrices. Finally, the bed load budgets of all channel sections are added up (Kienholz et al., 2010).

The input data of each channel section are: Area of the catchment contributing to the process, channel geometry and input of solids from the channel and adjacent slopes. The special feature of the Gertsch method is the use of gradient relationships. The change in gradient is described by the local and accumulated energy index. This index describes a highly simplified measure of the build-up of kinetic energy. In steep sections with rock or limited material supply, high velocities are likely to build up and the local energy index becomes positive. In sections with a lot of loose sediment, the kinetic energy is dissipated into heat due to friction, so the energy index becomes negative. The energy index $E - I_{GA}$ is based on the availability of loose sediments and the inclination of the torrent bed. The summation of those local energy indices $E - I_{GA}$ of the sections above is crucial for assessing the potential mobilisation of solids at a given location.

In addition, negative factors (such as thawing permafrost, confluences of major channels or breakthrough of logjam) are introduced by the Gertsch method. These factors describe process combinations that hold the possibility of triggering particularly large destructive debris flows. Such scenarios are likely to cause extreme erosion rates. These threshold processes can significantly multiply the mobilised sediment loads. For each torrent section, an estimate of the bedload is made. The rear-

rangement of material from the slope into the channel and the displacement along the channel are assessed separately. Figure 2.4 shows the proposed workflow of the Gertsch method. The information from all torrent sections is merged in an Excel spreadsheet. The Gertsch method always considers a rare to very rare reference scenario. During the assessment process, the scenario is adapted to the catchment-specific requirements with the selection of risk factors (Gertsch, 2009).

2.3.2.2. SEDEX

SEDEX, the second method, stands for SEDiments and EXperts and was developed as a comprehensible and well-documented method for deriving solid scenarios of torrents by Frick et al. (2008).

Checklists and predefined forms guide the user systematically through the survey and evaluation process in the field. The clearly structured evaluation processes guarantee reproducibility. The Sedex method has no fixed scenarios, these can be freely defined by the user. Usually, three scenarios are defined in the hazard assessment, with a rare (100 years) scenario being the reference scenario in many cases. Kienholz et al. (2010) describe three simple example scenarios:

- Debris flow triggered by heavy rainfall in combination with the activation of lateral gullies and shallow-seated landslides (e.g. reference scenario, assumed recurrence interval 100 years).
- Debris flow triggered by continuous rainfall in conjunction with additional reactivation of a dormant, deep-seated landslide in the catchment area (e.g. assumed recurrence interval $>$ 100 years).
- Fluvial transport triggered by heavy rainfall with activation of some bedload in the channel bed and only some unstable banks (e.g. assumed recurrence interval $<$ 30 years).

For a clear and accurate assessment using the SEDEX method, scenario building is a very essential and important step.

With the SEDEX method, the assessment of processes such as bedload transport up to debris flows is possible (the Gertsch method is only designed for extreme events with a return period of 100 years or more.)

Similar to the Gertsch method, the torrent is divided into similar sections. The sediment quantity for each scenario is calculated using 12 different modules. A clear subdivision is made into slope, torrent bed, channels and landslide areas. Simplified geometric abstractions are defined for each of these modules in order to calculate a sediment volume. A clear distinction is made between material-limited and material-unlimited sediment supply. In limited conditions, all available sediments will be transported by the debris flow event. The amount of solids is limited by the amount of available material. After an event, it will take time until the solids have accumulated again. In contrast to the unlimited sediment supply, where the delivery rate is only limited by the transport capacity of the process. High erosion rates in successive events are possible. For each module and scenario, the question must be answered whether erosion will occur and, if so, how much material can be mobilised. Included checklists help to answer these questions and guide the user step by step through the necessary considerations.

Simplifications are necessary because the entire calculation is based on an average value for each section. A length reduction factor is used to reduce the total volume to be expected by a certain percentage. This is recommended if the availability of sediment is more limited in a certain part of the section. Uncertainties can be taken into account in any estimate. The total volume is finally determined by summing all calculated solids. The assessment for the other scenarios is mostly done by reducing or adding to the reference scenario.

In Kienholz et al. (2010), the two methods are compared in detail and Stocker (2013) provides a example calculation at the torrents Dorfbach and Wildibach. More detailed descriptions are found in Frick et al. (2011) and Gertsch (2009).

2.3.3. Empirical Relationships for Debris Flow Assessment

There are many empirical formulas that can provide rough estimates of the total amount of debris flow solids in an alpine catchment. These formulas were developed for specific regions and are therefore difficult to apply under different climatic and geological conditions (Kienholz et al., 2010).

Nevertheless, we present below a few useful empirical relationships. The estimation of debris flow volume strongly depends on the erosion rate during a debris flow event. An erosion depth must be estimated in the field. As a check, the erosion depth E $[m]$ can be roughly estimated by

$$E = 13S \tag{2.1}$$

where S is the inclination of the torrent bed $[m/m]$. This equation tends to overestimate the erosion depth. In general, it can be said that the ratio of erosion width to erosion depth is at most between 2:1 and 5:1 (Rickenmann, 1995).

Based on a presumed debris flow volume, many other parameters can be derived empirically. To assess whether the capacity of the torrent is sufficient for a given event, the peak discharge and the associated flow velocities are important parameters. Rickenmann (2014) describes the relationship between peak discharge Q_P $[m^3/s]$ and debris flow volume M $[m^3]$,

$$Q_P = 0.135M^{0.78}. \tag{2.2}$$

He found that peak discharge correlates with the volume of the largest surge rather than with the total volume of a debris flow event.

The flow velocity of debris flows depends on the material composition (grain size, grain distribution, water content). It is often difficult to determine a specific flow velocity for an event because the material composition changes for different surges. Rickenmann (2014) introduces an equation for determining flow velocity v $[m/s]$ based on discharge Q $[m^3/s]$ and the slope of the torrent bed S $[\%]$,

$$v = 2.1Q^{0.33}S^{0.33}. \tag{2.3}$$

Further empirical relationships are found in Rickenmann (2016).

2.3.4. Assessment Guidelines

The current federal guidelines (BAFU, 1997) on torrent processes do not contain a clear procedure for hazard assessment. The guideline BAFU (1997) is very old and a new guideline is in progress. For the management of landslides, hillslope debris flows and rockfall processes, a new guide has recently been published (BAFU, 2016). It provides clearer guidelines than the previous version, but still provides a wide space for interpretation. A more precise guideline is provided by Hunziker et al. (2021).

The hazard assessment procedure can be divided into two parts. The torrent analysis, which involves the description of the general situation and framing of the hazard relevant scenarios for debris, hydrology and wood. In a second step during the runout analysis, weak points are assessed in detail. Based on expert knowledge, runout distances and thickness of the deposition are estimated. A hazard map results from the combination of frequency and magnitude. (Allen et al., 2022).

New methods which are based on remote sensing or UAV techniques are not mentioned in guidelines and recommendations such as BAFU (1997), BAFU (2016) or Hunziker et al. (2021). Below, we introduce the current state of the art UAVs.

2.4. UAV Surveying

The number of UAV-based research has increased enormously in the last decades. About a decade ago, the application of UAVs was unique in many scientific laboratories and in early operation stage in military and civil domains. They have become very accessible to the public domain. UAVs are carrying cameras, temperature sensors, LiDARs and many other measurement tools. Applications are ranging from monitoring to mapping, search and rescue and many more (Giordan et al., 2020, 2018; Nex and Remondino, 2014).

2.4.1. UAV Types

At current stage, three different UAV categories can be differentiated: Vertical take off and landing (VTOL), rotary-wing systems, such as helicopters and multirotor platforms, and thirdly fixed-wing systems (Ducard and Allenspach, 2021). The most common UAVs are multi-rotor UAVs with four or more propellers. They have the capability to hover and are able to carry relative heavy payloads. Fixed wing UAVs have the advantage to fly over large distances and capture therefore large aerial extents. Hybrid flying machines such as the fixed-wing Hybrid VTOL UAVs (e.g. Wingtra) combine the advantage of long flight duration time of fixed-wing UAV with vertical takeoff and landing capabilities coming from multi-rotor UAVs. This is done by tiling of the wings, rotors or even the entire airframe (Ducard and Allenspach, 2021). Larger multi-rotor UAVs such as the DJI M600 are capable to carry payloads of several kilograms. The possibility to mount LiDAR scanners on UAVs opens a whole field of new research possibilities. In chapter 4, we present an overview over the different types of UAVs we have used.

2.4.2. Georeferencing

Conventional georeferencing using GCPs leads to reliable positioning if the spacing of the GCPs is uniform. Evenly spaced placements are time costly and can even be dangerous to install in steep terrain. Direct differential global navigation satellite system (GNSS) georeferencing of the collected images may overcome these limitations. In the last couple of years, the advancement of GNSS technology, high-quality inertial measurement unit (IMU) and furthermore dedicated RTK (real-time kinematic) and PPK (post-processing kinematic) solutions enable accurate measurement of camera position and orientation. Errors of atmospheric propagation delay and receiver clock errors can be eliminated by double referencing the phase uncertainty between two GNSS-GPS receivers. In comparison to PPK, RTK needs a stable connection to receive the GNSS signal. PPK processes the information after the flight, which reduces the risk of data loss. However, the use of RTK-PPK approaches brings up the advantage that GCPs are not indispensably needed (Zhang et al., 2019).

The accuracy and precision of photogrammetric dense point clouds does not only depend on accurate camera positions. Many other factors such as meteorological conditions during the flight, image quality, flight plan settings and camera calibration influence the accuracy and precision of photgrammetric data. Also the type of the used structure from motion algorithm has an influence on the processing results (Zhang et al., 2019).

The image quality strongly depends on the mounted camera system. Low cost action cameras tend to have a shorter focal length and provide a larger field of view, which subsequently is characterized by radial lens distortion. However, with the use of RTK-PPK the "doming" effect can be greatly mitigated due to precise and dense control of the camera position Zhang et al. (2019). Nevertheless, the tie point density of DSLR camera images is by a multiple denser compared to low cost cameras. The detailed images help to improve the finding and matching of tie points. At the same time, Zhang et al. (2019) conclude that the DSM reconstruction is reproducible for expensive DSLR or action cameras. The main difference can be found by the level of detail of the surface representation.

2.5. Point Cloud Classification

Accurate DTM models are a prerequisite to be able to analyse landscapes and landscape changes. The separation of a point cloud into ground and non-ground points is an essential step in the analysis. In this section we provide an overview of common approaches to classify point cloud data. Many of these approaches require calibrated parameters to achieve highly accurate results. The calibration process is often complicated. Most of the algorithms are developed for LiDAR data. Such classical filtering algorithm, which rely on first and last returns of the LiDAR signal often do not provide usable classification results for photogrammetric point cloud data (Rusnák et al., 2018).

Some point cloud applications such as Pix4D, Terrasolid-Terrascan and Agisoft provide algorithms to classify photogrammetrical point clouds using different algorithms. Anders et al. (2019) have compared some of those algorithms. We present a selection of common algorithms:

- Zhang et al. (2016): This is a filtering method, which only needs a few easy to define parameters. The main idea behind this algorithm is to invert the point cloud and then use a rigid cloth (manifold) to fill up the inverted surface. The interaction between the cloth nodes and the corresponding points from the point cloud is analysed. After an iterative process, the final position of the nodes can be used to determine an approximate ground surface. An advantage of this approach is, that it can be easily used without much experience in calibration of parameters for point cloud classification routines.
- Brodu and Lague (2012): The CANUPO application classifies point cloud data automatically based on training data sets. The user is able to separate the point data into two subsets, based on the fitness of the training data. This algorithm was developed for terrestrial laser scanning data and works based on distance and orientation of the points. RGB-values are not recognized by the algorithm. As it is a binary classification algorithm it is strongly limited in the classification of different vegetation types. And as the point color is not incorporated in the algorithm, the deviation of boulders and dense shrubs is often not evident.
- LAStools (2022): The software packages from rapidlasso GmbH is a software for fast LiDAR point cloud data processing. It is a collection of various highly efficient, batch-scriptable, multi-core command line tools. Different tools can be applied to classify, tile, filter, rasterise, convert, clip and polygonize LiDAR data. Their ground classification routine is based on Triangulated Irregular Networks (TIN). In a first step a sparse TIN is created on minimum number of points. On iterative steps the TIN becomes more densified. The potential ground points have to meet certain criteria of the triangle, such as a maximal distance to the TIN facets, the acceptable offset from current ground and how much the terrain is allowed to bulge. This standard classification routine can be combined with other operations to optimize the classification workflow.
- Terrasolid (2021): Terrasolid-TerraScan is a software extension, which can be used in CAD programs such as Microstation or Spatix. It provides a large selection of tools to classify, process and analyse photogrammetric and LiDAR point cloud data. The workflows can be manually adapted for specific needs and properties of the point cloud. The ground classification routine is also based on TIN densification, as the Lastool algorithm. In a first step low points are selected, which are assumed to be well-defined ground points. The initial ground point selection is controlled by the maximum building size parameter, which describes the maximal assumed area of no ground point occurrence. Initially a surface model is build on the basis of the initial ground points. In the subsequent iterations more and more points are added to the ground points and approaches closer to the real ground. Additional parameters allow an accurate calibrated rule set, which determines the selection of ground points: The parameter iteration angle defines the maximum angle between a point and its projection on the triangle plane and the closest triangle vertex. And the iteration distance avoids big jumps upwards if triangles are large. With this routine ground points that are to high can be avoided. Additionally, the color of the photogrammetric point cloud can be used as weighting factor, which leads to a higher likelihood for green points to be classified as vegetation.

- Becker et al. (2018): In the software package Pix4D, a classification method has been implemented which relies on geometric properties and color information of the point data. The classifiers are trained with Gradient Boosted Trees (Friedman et al., 2001) and can be adjusted manually within the project. Unfortunately this algorithm is only available within the software Pix4D and not applicable on already processed point cloud data.
- Agisoft (2021): The software Agisoft is another all-in-one package for processing aerial imagery into point clouds, ortho-rectified images and digital surface models. It includes a native algorithm for ground point cloud classification. This filtering approach uses three parameters. Firstly, the maximal angle between the terrain and the point which may be considered as ground point. In steep terrain, this value has to be chosen considerably high. Secondly, the maximum distance between the terrain model and potential ground point. And finally the cell size of the intermediate terrain, which describes the maximal expected size of no ground data points due to buildings or dense tree coverage.
- Kraus and Pfeifer (1998): Iterative surface lowering is widely applied in remote sensing. The algorithm works iteratively and is based on linear prediction with an individual accuracy estimate for each measurement. In the beginning a surface is calculated with equal weights for all z-measurements from the point cloud data. The residuals give insights on whatever a point tends to be: vegetation or ground. Terrain points are likely to have negative residuals, whereas vegetation points are likely to have positive residuals. In subsequent iterations only assumed ground points are considered in the computation. As the amount of assumed ground points is not changing anymore, the remaining ground points are considered as surface.

In chapter 4 we present two approaches which we developed in the scope of this work. This approaches are adapted and tested in steep alpine environments with dense vegetation coverage.

2.6. Geomorphological Change Detection

To be able to compare two point cloud data sets, the two datasets have to use the exact same reference coordinate system. This co-registration processes can be done using different procedures. Cook and Dietze (2019) have introduced a simple method to combine two UAV-derived datasets, which aligns all images of both acquisitions together, through this routine the spare point clouds exist of common tie-points. This method has the advantage that no GCPs are needed. If there is a systematic error in between the two flight missions the error must to be removed (James et al., 2020). Such systematic errors often origin from optical lens distortion. The systematic error than has to be spatially modelled and subsequently subtracted of one of the datasets. A further possibility is the application of the iterative closest point (ICP) algorithm. This algorithm aims to minimize the difference between two point clouds, in a way that it iteratively revises the transformation until the error metric is minimized to a specified accuracy (Besl and McKay, 1992). ICP is one of the widely used algorithms in aligning three dimensional models, however this algorithm is limited to correct pronounced topographic systematic errors such as doming. Subsequently, we introduce five

approaches which can be used to measure distances between two different co-registered point clouds within the scope of geomorphologic applications.

2.6.1. DTM of Difference (DoD)

The subtraction of two digital terrain model is far the most common method for change analysis. The point clouds are gridded to produce a digital terrain model. Main application are the analysis of river beds (e.g. Lane et al., 2003; Schürch et al., 2011) and in the analysis of cliffs and river banks as the point clouds are rotated before gridding (e.g. O'Neal and Pizzuto, 2011; Rosser et al., 2005). DoD is a fast method, which measures the difference on a pixel-by-pixel basis measuring the vertical difference. Uncertainties regarding point cloud roughness, data quality and point cloud registration can be addressed. The main disadvantages of DoD are: A DTM cannot handle overhanging landforms such as cliffs, large blocks and bank failures. The information density decreases proportionally with surface steepness. To measure surface difference accurate, they should always be measured along the normal direction of the observed surface (Lague et al., 2013).

Rough surfaces such as talus slopes or mountain torrents have always missing data due to occlusion. In raster datasets the empty cells are interpolated. This leads to an uncertainty on the grid elevation. (The M3C2 algorithm can be applied on 2D scale while imposing a vertical normal calculation. See below.)

2.6.2. Cloud2Cloud (C2C)

This is the fastest and simplest 3D comparison method of point clouds. The method does not depend on gridding or meshing of the data or calculation of surface normals. Cloud to Cloud comparison is often used in cloud matching techniques such as ICP (Besl and McKay, 1992). The calculated distance is sensitive to roughness, outliers and point spacing. This techniques was mainly developed for rapid change detection on very dense point clouds (Lague et al., 2013). Compared to the digital terrain model of difference it overcomes the limitations of overhanging landscape structures.

2.6.3. Cloud2Mesh (C2M)

Cloud2Mesh calculates the distance between a point cloud and a reference 3D mesh (Cignoni et al., 1998; Monserrat and Crosetto, 2008; Olsen et al., 2010). This method works well on flat homogeneous terrain with low roughness, where the reference point can be constructed accurately on a mesh surface. Rough terrain with missing data due to occlusion cannot be represented accurately within a mesh grid.

2.6.4. Multiscale Model to Model Cloud Comparison (M3C2)

The M3C2 calculation is a fully 3D point-cloud comparison state of the art method for geomorphic change detection (James et al., 2017b). The method is especially valuable in steep complex terrain. The calculation of the M3C2 is based on core points. Those core points are a sub-sampled version

of the reference cloud. To speed up the calculation a minimum point spacing is selected to build up the sub-sampled version of the reference cloud (Lague et al., 2013). For any sub-sampled core point a normal vector is calculated. This normal vector derives from a plane which is fitted through all neighboring points within a given radius $D/2$ (normal scale)(see Figure 2.5).

The normal vector is used to project the core point i onto the reference cloud at scale d. This is done by defining a cylinder of radius $d/2$ whose axis goes trough i and has the same orientation as the normal vector. The interception of both clouds with the cylinder creates two sub-clouds. The subtraction of the average of both clouds represents the detected change. An important parameter is the diameter of the cylinder (projection scale). A large projection scale minimizes the influence of surface roughness and therefore reduces the spatial resolution of the output. For change detection in alpine terrain a diameter of 0.5 m adapts well in most cases. Normally, the distance is calculated using the normal estimated on the reference point cloud. This can be explained by the fact that geomorphic processes tend to depend on surface geometry. However it is also possible to impose a vertical normal calculation (Lague et al., 2013).

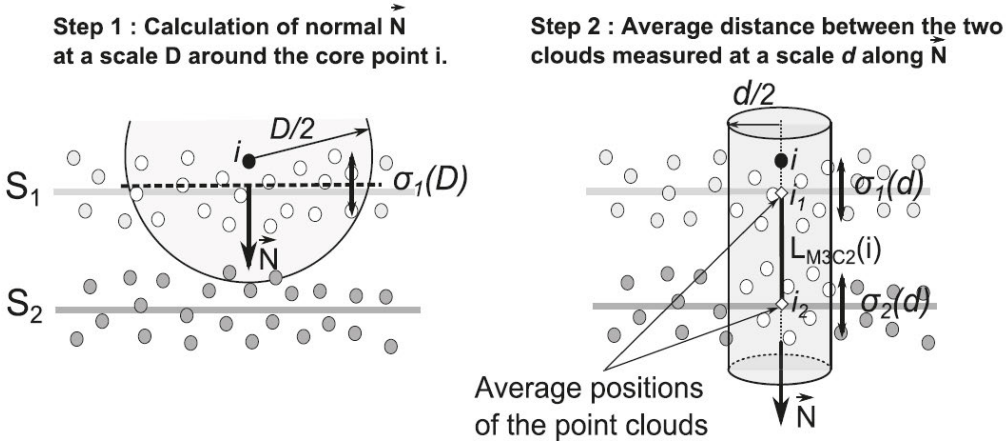


Figure 2.5. Normals (N) are computed for cloud S_1. The orientation is defined by plane, which is fitted along all neighbouring points of i within a given radius $D/2$. Along these normals a cylinder is orientated with a given radius (d/2). The intersection of both point clouds creates two subclouds. Subsequently the average of both subclouds is calculated and subtracted from each other (Lague et al., 2013).

2.6.5. Feature to Feature Supervoxel based Spatial Smoothing (F2S3)

Gojcic et al. (2021) propose a new fully automated deformation analysis routine, which is able to estimate real 3D displacement vectors from point clouds. Compared to the methods presented previously, which are based on distance calculation in the Euclidean space, the current method is based on dense 3D displacement vector fields. The corresponding points are searched on the basis of 3D local feature descriptions.

Those local feature descriptors analyze the geometric peculiarities in the neighborhood of the respective point. The resulting correspondences are noisy and incorporate many outliers. The developed smoothing algorithm by Gojcic et al. (2020) is capable to classify the putative point wise correspondences into inliers and outliers. This is done in a local neighborhood extracted from the super-voxels.

Case studies of landslides show that the traditional methods often underestimate the displacements compared to the F2S3 approach, which is representing the real displacement more accurate Gojcic et al. (2021). This new approach is highly important for landslide monitoring. For the calculation of mass balances after debris flow events, this approach is strongly limited, as the local feature descriptions are not comparable before and after the event. We have now introduced all relevant research fields. In the following we elaborate the research gaps and formulate corresponding research questions.

2.7. Research Questions and Motivation

There is no standardised assessment method for torrents. The methods presented, such as Gertsch (2009) and Frick et al. (2011), rely strongly on expert knowledge and are therefore not always fully reproducible. In practice, time and costs associated with the assessment phase are important factors when formulating mitigation schemes. For a good assessment fieldwork is essential, but often there are not enough financial and personnel resources for proper field investigations and subsequently little fieldwork can be performed. The use of UAVs may open new opportunities for a more objective and time-efficient hazard assessment. The whole catchment can be surveyed and fieldwork can be addressed to specific locations.

New UAV-based remote sensing approaches should be considered as a cost and time saving method to monitor debris flow prone catchments. Technology and research in this field is developing fast. Our primary goal is to explore the advantages and possibilities of remote sensing-based methods. We do not aim to replace fieldwork with automated remote sensing-based workflows. At the accuracy level of hazard zone, methods should rely to a certain degree on field investigations. Our aim is not to develop a new method. We want to show what information can be extracted from photogrammetric UAV data and how this additional information can be used for a more objective assessment of torrents. An important question is how to exploit remote sensing data to improve numerical, scenario-based predictions. One important advantage is that the whole catchment area can be easily monitored. In practice often only the affected area at the valley floor is evaluated in detail. We therefore formulate the following main research question: **How do photogrammetric derived UAV point clouds of debris flow prone catchments improve the investigation of torrential processes?**
To answer this question we analysed various debris flow catchments in the canton Grison. In this thesis we present the data of the two debris flow prone catchments. Additional, and more detailed, research questions are:

- Is it possible to filter bushes and trees based on photogrammetric UAV point cloud data in steep alpine debris flow terrain?
- Is it possible to extract torrential characteristics based on object based image analysis? And can those properties be further used to segment the torrent into homogeneous sections?
- Where and how much fieldwork will still be needed with the new approach?
- Where are the new methodological approaches applicable? How much effort is involved?

To answer these questions we can define three work packages. In a first step we need to build the basis for geomorphologic analysis based on UAV data. This implies the removal of vegetation, which is a challenging task in steep terrain. Secondly we aim to capture torrential properties automatically. In this manner we peruse the aim to derive parameters, which are normally determined during a field campaign, in an automated procedure. And in a third step we investigate the possibilities associated with multiple UAV flights.

In a broader perspective, we aim to identify along which processing steps of the hazard assessment UAV derived data may provide new opportunities for a more objective investigation. Overall we expect an increase in replicability of the assessment. Nowadays hazard assessment is often based on a large part on expert opinion. We are aware that assessment of mountain torrents is a highly complex field of practice and that a lot of professional experience is needed for a sophisticated hazard assessment. Finally, the UAV-based approaches developed in this thesis may find application along various different processing steps in the hazard assessment. We aim to discuss the possibilities where and how we can make use of UAV derivatives. As this methodology probably finds application in practice, questions regarding the effectiveness and costs should be estimated and discussed.

Two Primary Study Sites: Arelen and Fraschmardin

To investigate how UAVs can be applied for torrent assessment we selected two specific test sites for detailed study: Arelen (Davos, GR) and Fraschmardin (Klosters, GR). The sites represent a high geomorphological diversity and were selected from a total of five different candidate sites that we investigated in the course of this project. The three other test sites are Carrerabach (Valendas, GR), Val Mela (Bravuogn, GR) and Val Greva (Madulain, GR) (see Figure 3.1).

Both the Arelen and Fraschmardin catchments have a similar size. Both torrents extend beyond the tree line and therefore have a vegetation free upper catchment and a vegetated transit and runout zone (see Table 3.1). Moreover, the application of UAVs in vegetated and non-vegetated terrain can be analysed at two sites with strongly differing lithologies.

Table 3.1. Overview of the main characteristics of the Arelen and Fraschmardin catchment.

	Arelen	**Fraschmardin**
Catchment Area	0.8 km^2	1.06 km^2
Altitude	1638 - 2530 m a.s.l.	1294 - 2657 m a.s.l.
Length Torrent (vertical)	1.87 km	1.96 km
Length Torrent (terrain)	2.0 km	2.18 km
Mean Slope	31%	43%
Exposition	East	South
Melton (1965)	0.997	1.325
Geology	Ophiolite	Mönchalp Gneiss
Surface Coverage	45% weathred rock 35% shrub forest 15% forest 5% grassland	30% grassland 25 % rock 20 % shrub forest 20 % forest 5% weathred rock
Other Processes	avalanche, rockfall	avalanche, rockfall
Constructions	retention dam, check dams (partly damaged)	embankment solification, lateral dams

3.1. Arelen

The Arelen torrent is located near Davos Wolfgang (see Figure 3.1). This torrent is relatively active and a debris flow retention dam has been constructed in 2018 to protect the buildings located at the Wolfgang pass. The topography around the Wolfgang pass is shaped by three major rockslides from the Totalphorn (about 400 mio m^3) (Abele, 1974; Maisch, 1981; Signer et al., 2018). The geology of Davos Arelen is dominated by strongly tectonized and weathered ophiolites. During heavy rainfall, water cannot infiltrate because the underlying rock is less permeable than the loose surface. It is estimated that saturated unconsolidated sediments can be easily mobilized by debris flows of several 1'000 m^3. The numerous channels build a complex network. In the case of a debris flow event, we expect several surges characterized by a high water content within short succession. Regarding the

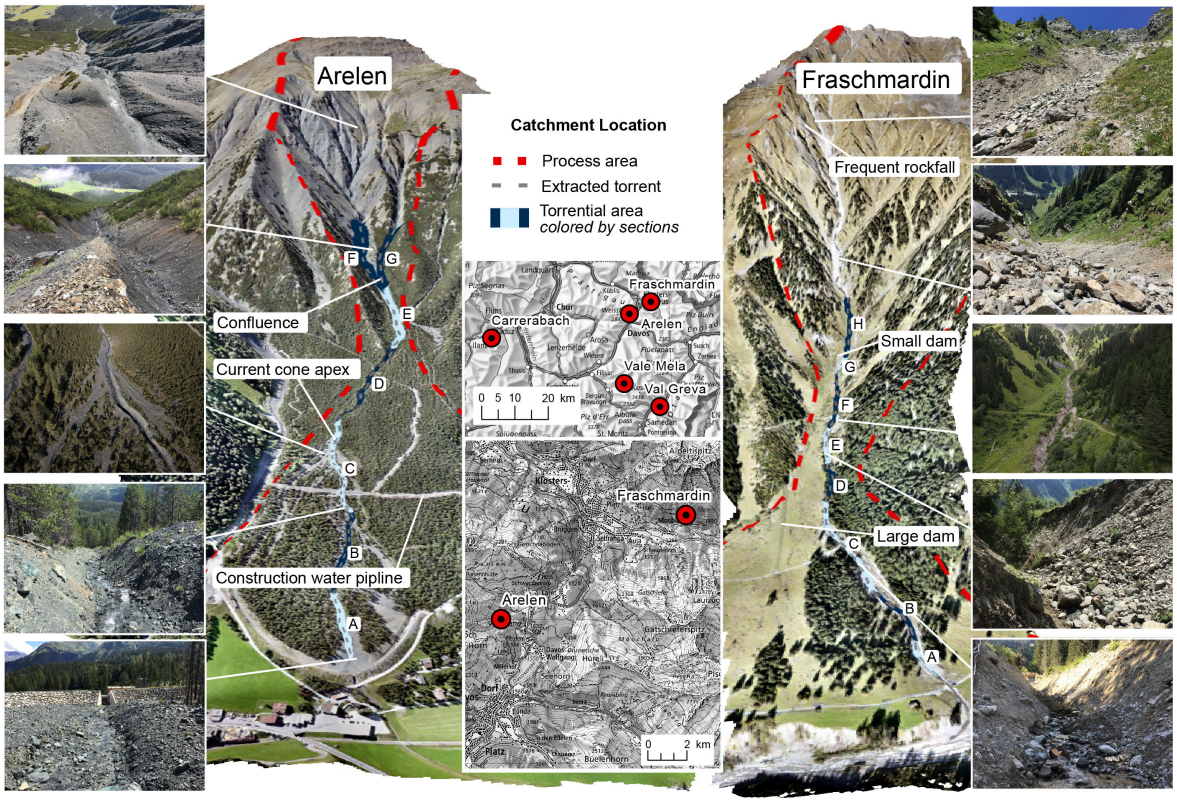


Figure 3.1. Overview over the two main study sites Arelen (left) and Fraschmardin (right). The torrential area is visualised in blue, the sections are colored in light and dark blue. The red dashed line indicates the process area, representing the upper watershed and the entire cone apex. The various insets highlight specific catchment locations.

morphology of the torrent cone, we find homogeneous patterns. These self-similar patterns indicate the occurrence of many smaller debris flow events. The catchment therefore provides an unlimited supply of sediments due to the highly weathered ophiolite, which can retrain a lot of precipitation. Pre-moisture of the soil has a major influence on the likelihood and erodibility of a debris flow event. The size of the event is mainly limited by the intensity and duration of the precipitation event. Since the catchment size is relatively small at 0.8 km^2, the expected runoff is low.

We can observe that the current position of the fan apex is in the middle of the debris cone. Current events tend to be erosive in the upper part of the cone and deposit material only in the lower part. A dam has been constructed in the bottom of the valley to hold back the debris flows. However, the total retention capacity is too large for the expected magnitude. The difficulty is that the events tend to leave the actual torrent bed and deposit material on the debris flow cone. As can be seen clearly in the Figure 3.1, the dam was built assuming that events leave the torrent only in the middle of the debris flow cone, where the actual elevation of the torrent is higher than the level of the debris flow cone.

Based on field inspections we have assessed the specific debris flow volume for an event with a frequent to average return period. Following, we evaluate the expected volumes based on sections from the upper catchment to the retention dam. Above the major confluence at 1870 m a.s.l. a clear delineation of the torrent becomes more and more challenging. On the orographic left sub-

catchment (section G) is an old damaged retention dam. A failure of the check dam would increase the volume by 200 m^3. Further up in the orographic left subcatchment, we estimate a sediment potential of several 100 m ^{3}. Overall we consider about 500 m^3 originating from the orographic left sub-catchment.

The uppermost section of the orographic right sub-catchment (section F) is characterized by several gullies. We estimated the erosive capacity of these gullies to be 0.5 to 1 m^3/m. This may result in a volume of 500 m^3 of solids. In the lower part of this sub-catchment, the gullies merge to form a torrent. Here we estimate an erosion of 2 m^3/m over a length of 250m. This also adds up with another 500 m^3/m to 1,000 m^3 at the confluence from the orographic right.

Between the confluence and the widening (section E-D), where the larger debris flows leave the torrent, the slopes are very steep and unstable. We estimated 1 m^3/m from each side of the slope and 2 m^3/m from the torrent bed. Over a length of 500 m, this gives a total erosion of 2,000 m^3.

Summing all eroded volumes, we reach an expected debris flow volume of 4,500 m^3 if both sub-catchments are active. The remaining lower part (section A-C) is regularly deepened. The current position of the torrent bed is higher than the level of the debris flow cone. We do not expect significant erosion rates here. Larger events tend to erode and may leave the torrent bed if they exceed the capacity of the torrent. Smaller events tend to deposit in the torrent. This fill may allow subsequent events to overflow more easily.

For an event with a frequent to average return period, we expect a volume of 3,000 - 5,000 m^3. Due to the limitations of the catchment size, we do not expect events with rare or very rare return period to be of a significant greater magnitude. Estimates of runoff are difficult. In general, we can conclude that a specific discharge of 10 m^3/s would have enough space in the channel. If multiple surges from different sub-catchments have a similar timing, the debris flow discharge can easily increase to as much as 20 m^3/s. Geotest (2007) has estimated a maximal potential debris volume of 7'000 m^3 for a frequent event, which exceeds our estimation of 3'000 - 5,000 m^3. In Landschaft Davos and Tiefbauamt Graubünden (2000) they have estimated a debris volume of 2000 m^3 for a very rare flood scenario. In the event cadastre only debris volume up to 4000 m^3 have been observed (Geotest, 2007). The large variation emphasise the need for more reproducible hazard assessment routines. However, terms such as a potential debris volume should not be used in relation to return periods.

3.2. Fraschmardin

The second study torrent Fraschmardin has an area of 1.06 km^2. The lithology consists of Mönchalp Augengneis. The rock mechanical properties of this gneiss lead to a grain size distribution, which is dominated by large boulders originating from the strongly rock-fall prone head wall at 2600 m a.s.l. There is one main channel leading from the head wall down to the Landquart river. We expect one main surge to be decisive for the resulting magnitude. The morphology of the torrent changes with the elevation of the catchment. The uppermost part of the catchment is rather steep ($>35°$) and the frequent rockfall activity leads to an unlimited supply of debris. Fresh not yet weathered surfaces indicate recent activity. Around 1800 m a.s.l. the actual torrent becomes incised. Regular debris flow

events will follow the clearly defined torrent. However, the rough morphology of the cone indicates the occurrence of major events in the past. From the newspaper Davoser Zeitung (1900) we know about past events that posed a threat to the hamlet of Monbiel. The catchment size is rather small and the respective hydrological runoff is limited. However, in the uppermost area immense quantities of debris are available. Frequent smaller debris flow events may only mobilize minor parts of the available debris stored in the uppermost catchment. Thur (2019) have estimated a debris flow size of 5'000-7'000 m^3 for very frequent events. Frequent events are expected to bring 10'000 m^3 and rare events 50'000 m^3 (Geo7, 2000).

Following, we evaluate our field-based assessment of the Fraschmardin torrent. As it can be seen in Figure 3.1 the torrent devolves into a scree slope in the uppermost catchment. Debris originating from repetitive rockfall are temporarily stored in this uppermost section and are available for further transport. Along this part of the torrent it is not possible to extract specific torrential elements such as embankments. Furthermore, it is very challenging to predict where and how much the material will be eroded. We assume an erosion of 10 m width and 1 m depth over a length of 600 m. Deeper erosion is not expected because the bedrock is close to the surface. This results in a total erosion volume estimate of 6000 m^3.

Amongst an elevation of 2000 m a.s.l., the topography is still very steep. The bedrock is very close to the surface. As a result, there is only a limited supply of loose debris in the channel available for further transport. We have generally assumed a depth erosion of 20 cm over a width of 5 m. This results in an erosion rate of 1 m^3/m. Summing these measures, the total volume is 700 m^3. A separate evaluation of the slope processes is not required. At 1900 m a.s.l. a second minor side channel joins the main channel, which we did not evaluate further. Bellow 1800 m a.s.l. (section H) the torrent is more incised and the rock is a good distance below the torrent level, except for a few places where the torrent flows on the bedrock. Since erosion is limited in these places, the total erosion is also limited.

Between the confluence and the upper forest road at 1550 m a.s.l. (section F-G), the slope on the orographic right is partially consolidated with an obstruction of stone blocks. In addition, at 1650 m a.s.l., there is a dam that is intended to redirect a overflown debris flow back into the main channel. The embankment obstruction is damaged in some places. Failure of this protective structure would increase the amount of available solids and increase the possibility of an outbreak of a major debris flow below the protective dam. While such a scenario is unlikely, it is considered as a possibility for a debris flow event with a rare to very rare return period.

There are some minor landslides on the orographic left side. The depth of these landslides is about 0.5 m. The area-based potential for further landslides is severely limited. There is no potential for a large volume of new landslides to develop. Therefore, it was decided to include these potential volumes as part of the embankment processes. Overall, 2 m^3/m is assumed to erode from depth and an additional 1 m^3/m is assumed to erode from the embankment, including shallow landslides on the orographic left. Because in some regions the torrent bed is bedrock, a length reduction factor of 0.7 is applied to the entire section. Multiplying the length of 310 m, an erosion rate of 2.8 m^3/m, and the reduction factor of 0.7 results in a total volume of about 600 m^3. In case of total failure

of the protective structure, the volume of debris flow may drastically increase. To better distinguish the part with the presence of bedrock, we have divided this section into two parts (F and G) for our automatic extraction of torrential properties.

Below the upper forest road at 1550 m a.s.l. (section D-E), the behavior of the stream changes. The abrupt incision immediately below the road indicates the potential for further depth erosion. Here the bedrock must be well below the torrent bed. The potential for further depth erosion in the unconsolidated sediments needs to be considered. We estimate an average depth erosion rate of 3 m^3/m over a length of 400 m. The embankment is not very stable. In between the debris flows, the torrent is repeatedly filled with loose debris and shrubs that slide into the reach of the torrent as it can be seen in Figure 3.1. Because the embankment is quite steep and unstable, we estimated an additional erosion rate of 2 m^3/m for the embankment. We assume that several small spontaneous landslides are triggered during an event. Specifically, we assume that 10 landslides with an average area of 6 by 10 m and a depth of 0.5 m are triggered. This adds 300 m^3 to the volume of 2000 m^3, resulting in a total volume of 2300 m^3 for section D-E.

Below an altitude of 1460 m a.s.l. (section A-C), the slopes become more stable. As the torrent turns sharply to the left, the speed of the debris flow will decrease. We expect that only extreme events will break out in this curve. Overall, we expect that large events can still erode large volumes. Nevertheless, we observe a shift from the erosion-dominated regime to the transport- and deposition-dominated regime. The lateral levees are becoming larger. It is not always clear how strong the anthropogenic influence on the torrent geometry is. However, it is clear that the human influence increases in downslope direction. We can observe clear traces of excavation of the channel bed and consolidation of the embankments. For this roughly generalized lowermost section, we assume an additional erosion volume of maximal 500 m^3.

Summing all these quantities, we reach an expected debris flow volume of 10'000 m^3 for a frequent to average scenario. Compared to Davos Arelen, we expect rare events to be significantly larger. Especially in the upper catchment, the available debris volume is generally unlimited. If the initial debris flow can transport larger volumes, the erosion volume downstream and the probability of breakout increases, especially in the narrow bends and near the damaged structures. Geo7 (2000) have estimated a debris potential for a frequent scenario of up to 10'000 m^3.

Methods

In order to answer the research questions using the example of the two study sites Fraschmardin and Arelen, the methods applied are subsequently presented. In a first section, we present the platforms used and show how we processed the data. In order to quantify geomorphological changes and characterize torrent properties, we need to classify the point cloud into ground and non-ground points. This is done with two different routines using Terrasolid and LAStools software. The resulting high-resolution DTM is then used for automatic determination of torrential properties. The derived values are compared with geometrical measurements of the torrent. The respective scripts can be found in the Appendix B. Based on classified point clouds from several flight missions, the point clouds are corrected for systematic errors and distances are calculated using the M3C2 algorithm. The results are elaborated in the next chapter 5.

4.1. UAV Flights and Dense Point Cloud Processing

In this section, we introduce all the different platforms we used within this thesis. We also outline the data acquisition and processing workflow. Depending on the local conditions during the acquisition, we used slightly different settings such as flight altitude, transition time, and camera aperture.

4.1.1. Used Platforms

For our data collection we used different UAVs depending on the terrain and the task. The Table 4.1 shows the characteristics of the different platforms used. The RTK-PPK UAVs used for this study have the advantage over traditional solutions that highly accurate terrain models can be produced without the use of GCPs.

Table 4.1. Overview Table of used UAVs in scope of this thesis.

Name	**WingtraOne**	**DJI Phantom 4 RTK**	**DJI Mavic Air 2**	**DJI Matrice 600 Pro**
UAV Type	Tailsitter VTOL	Quadrocopter	Quadrocopter	Hexacopter
Dimensions	125 x 68 x 12 cm	46 x 35 x 24 cm	18 x 25 x 8 cm	169 x 152 x 74 cm
Weight	4.3 kg	1.4 kg	0.57 kg	10 kg
Maximal payload weight	800 g	-	-	5.5 kg
Battery capacity	198 Wh	89 Wh	40 Wh	780 Wh
Maximal flight speed	16 m/s	14 m/s	12 m/s	18 m/s
Wind resistance	12 m/s	10 m/s	10.5 m/s	8 m/s
Sensor	42 MP Sony RX1R II	1" CMOS, 20 MP	1/2" CMOS, 48 MP	Riegl miniVUX-3UAV
Focal length (35mm equivalent)	35 mm	24 mm	24 mm	
Location service	PPK	RTK	GPS	PPK
Price	∼35'000 CHF	∼5'000 CHF	∼1'000 CHF	∼150'000 CHF

Figure 4.1. Different used UAVs: DJI M600 pro with Riegl miniVUX-3UAV, Phantom 4 RTK, Wingtra One, DJI Mavic Air 2 (from upper left to lower right).

4.1.1.1. WingtraOne

The tailsitter VTOL WingtraOne is able to take-off, transit and land autonomously. With the WingtraOne it is possible to complete high-precision aerial surveys and mapping tasks. The high-end camera setup enables precise imaging even at high altitudes. The image position is post-processed with PPK.

4.1.1.2. DJI Phantom 4 RTK

The DJI Phantom 4 RTK is a widely used UAV in photogrametrical imagery. The low price and all-in-one processing enables the use for a wide audience. Compared to the WingtraOne, the image quality is lower. Beside the low cost a further advantage is the Phantom's ability to follow the terrain during flight, resulting in a more constant differential flight altitude. It cannot cover large areas of several km^2.

4.1.1.3. DJI M600 pro with Riegl miniVUX-3UAV

The DJI Matrice 600 Pro is a professional UAV. We have mounted a RIEGL miniVUX-3UAV airborne laser scanner, which is a high-precision laser operating at 300 kHz with a field of view of 120°, resulting in a dense point pattern on the ground. The resulting point density varies with the flight altitude and pattern. Different scanner configuration with respect to lines per second, scan repetition rates are

used depending on the operating velocity and altitude. The scanner is extremely light and compact (1.5 kg) and stores up to five target echoes per laser shot. Internal IMU and GNSS enable PPK positioning with centimetre accuracy.

4.1.1.4. DJI Mavic Air 2

The DJI Mavic Air 2 is a small and compact consumer-grade UAV. It does not offer precise RTK-PPK tracking and is primarily used to capture oblique pictures to investigate steep, inaccessible sections of torrents (see Figure 4.3 c). The high image quality and rough location information allows a-posteriori backtracking of the image.

4.1.2. Data Acquisition

Good weather conditions, that is no strong winds (>10 m/s), fog, precipitation or harsh sun/shadow contrasts, are a pre-requisite for successful data acquisition. In the Davos Arelen area, 36 Control Points (CPs) were painted on flat rocks, concrete or roads, which we used to quantify and compare the measurement error of the UAV missions. The positions of the CPs were measured with a Stonex S800 GNSS rover mounted on a tripod (see Figure 4.2). At the other four study sites, we set up a view detachable CPs. This practice allows a short quality assessment of the data.

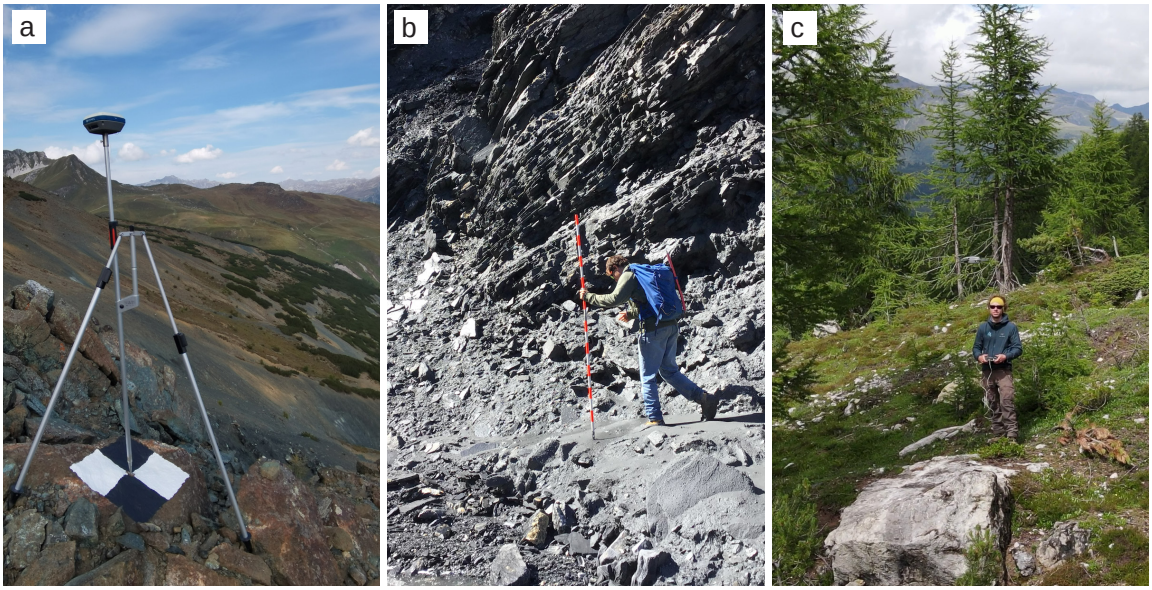


Figure 4.2. a) Measurement of CP with GNSS in Davos Arelen. b) Field measurements in Carrerabach. c) UAV flight with DJI Mavic Air 2 in Val Greva.

4.1.2.1. Data Acquisition with Wingtra

We planned the flight for the Wingtra UAV using WingtraPilot. The area to be surveyed is imported with a KML file. Fine adjustments can be made directly in the mobile app by dragging vertices. Image overlap was set to 80% along track and 60% across track and a flight altitude of approximately 200 m was chosen. This resulted in a ground sampling distance (GSD) of approximately 3 cm. Depending

on the take-off area, the transition altitude was adjusted. For successful flight missions the take-off area should be flat and the entire flown area should be visible during the flight. With the WingtraOne UAV, we flew parallel flight lines, whereby the closest point to the surface defined the height of the respective line. After the flight, the geographical position is stored in the metadata of the camera image. In a second step, the images are post-processed with WingtraHub. With the PPK workflow, the images are corrected to centimeter accuracy after the flight. The Wingtra PPK raw measurements of the satellite locations are compared with raw measurements from a reference base station. A high-precision geolocation is calculated for each image. WingtraHub PPK processing provides the location data in WGS84 with ellipsoidal elevation. For this study we reprojected the coordinate and elevation reference system to CH1903+/LV95 with LN02. The high-precision coordinates are converted from global coordinate system CHTRF95/ETRF93 to the projected coordinate system CH1903+ with LN02 using the Swisstopo service Reframe. Finally, the converted CSV file can be imported into Agisoft.

4.1.2.2. Data Acquisition with DJI Phantom 4 RTK

With the DJI Phantom 4 RTK UAV, we perform flight planning with the DJI controller. The area of interest is defined by dragging the vertices. Defining a flight plan that is accepted by the planning application can be a challenging task as the software does not give accurate error messages. We performed the acquisition with a single grid plan and thus only collected images along one direction, comparable to the flight settings of the Wingtra UAV. Since the UAV flies with RTK, the highly accurate location is already stored in the metadata of the image. The high-precision coordinates can be imported directly into the Structure-from-Motion software Agisoft, requiring no further post-processing. We have used this UAV in the strongly incised catchment of the Carrerabach and in Arelen to be able to compare accuracy of the resulting dense point clouds of WingtraOne and DJI Phantom 4 RTK.

4.1.2.3. Data Acquisition with DJI M600 pro with a Riegl miniVUX-3UAV

In order to be able to put the results of the photogrammetric UAV in relation to the LiDAR-based UAV, we carried out some reference flights with a LiDAR UAV in the Arelen catchment. We flew with the LiDAR DJI M600 pro equipped with a Riegl miniVUX-3UAV LiDAR using different flight settings. In total we flew four different flight plans (see Figure 4.3 a). The flight planning was done with the flight planning software UgCS (Version 4.2.156).

We wanted to assess optimal flight settings for creating a high-resolution DTM in alpine terrain with a LiDAR UAV. Therefore, we flew with different setups twice in the lower and upper catchment (see Figure 4.3).

4.1.3. Photogrammetric Data Processing

We process the images with Agisoft Metashape (version 1.6.5.11249), a stand-alone software that photogrammetrically processes digital images and generates spatial 3D data (Agisoft, 2021). Metashape

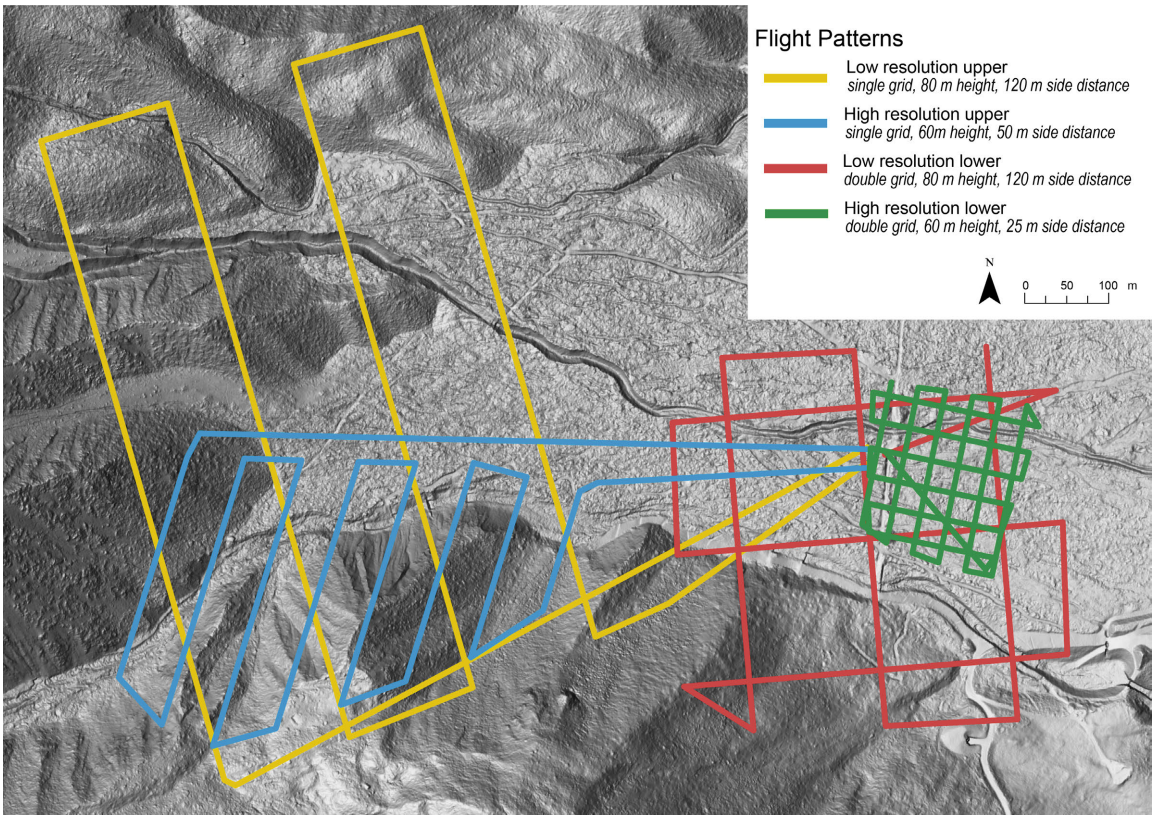


Figure 4.3. Flight pattern of LiDAR UAV flights. To crosscompare the resulting quality certain flight lines of the of the lower two flight patterns were selected.

is based on Structure-from-Motion (Koenderink and van Doorn, 1991; Verhoeven, 2011; Westoby et al., 2012). The software provides a complete photogrammetric workflow. The first processing step is called alignment and includes aerial triangulation and bundle blockadjustment. Metashape searches for feature points on multiple images and assigns them to tie points across the images. In addition, the camera position for each image is determined and the internal and external camera calibration parameters are refined using the standard routine of Agisoft. The result of this routine is a sparse point cloud and a set of camera positions (Agisoft, 2021).

Highly uncertain tie-points can be removed with gradual selection. We generally propose to delete tie-points with a projection error higher than 0.4 and in a second step all tie-points with a reconstruction uncertainty higher than 10. This procedure deletes erroneous tie-points that typically occur e.g. along forested areas.

The filtered sparse point cloud is used to determine depth maps based on dense stereo matching. The previously estimated camera positions are used to determine the depth information for each camera. All results are merged into a single dense point cloud. According to Bühler et al. (2016), the parameters *quality* and *depth filtering* have the strongest influence on the resulting point cloud. The quality indicates the desired level of detail of the reconstruction. Depth filtering removes outliers from the point cloud, which often arise from noisy or blurred images and poor texture of the scene. We used a high quality and aggressive depth filtering.

4.1.4. LiDAR Data Processing

Applanix IMU trajectories are corrected with PosPac 8.6 UAV considering correction data of the AGNES systems. Afterwards, the raw LiDAR data is processed with Riprocess (version 1.8.7.). The corrected trajectories from PosPac are applied and optimized within Riprecision. Further the points are colorized and the respective flight line is assigned as an attribute. During post processing, points below the ground are removed within LAStools. To be able to compare the flights in the upper and lower catchment (see Figure 4.3), we only included certain flight lines of the missions in the lower catchment for the data analysis: To achieve the same resolution for the green and blue flight pattern, we selected only every second flight line in the W-E direction of the green flight pattern. To reduce the resolution of the red double grid and achieve the same resolution as the yellow grid, we selected all flight lines in the W-E direction, therefor we get a single grid resolution with a lateral separation of 120 m.

4.2. Ground Classification of LiDAR Point Clouds

For LiDAR point clouds exists a standard processing solution serving as comparison data set. Within LAStools low lying noise points are removed and ground points are classified using the lasground functionality in LAStools. A non-default *step size* of 3 m is found to be working best for the classification of shrub forest. To be able to compare the flights in the upper and lower catchment (see Figure 4.3), we only included certain flight lines of the missions in the lower catchment for the data analysis: To achieve the same resolution for the green and blue flight pattern, we selected only every second flight line in the W-E direction of the green flight pattern. To reduce the resolution of the red double grid and achieve the same resolution as the yellow grid, we selected all flight lines in the W-E direction, therefor we get a single grid resolution with a lateral separation of 120 m.

4.3. Ground Classification of Photogrammetric Point Clouds

In the scope of this work, we have tested different ground classification routines. Many conventional algorithms do not perform accurately with photogrammetric data, as they were developed based on LiDAR data. However, the best results are obtained by combining RGB and another LiDAR-based classification routine (Anders et al., 2019). We have developed two approaches that provide reliable results. Both routines use the RGB color information of photogrammetric data to assist the classification routine. In the following, we outline the classification workflow of these two developed approaches.

4.3.1. Approach A: Point Cloud Classification with Terrasolid

TerraScan is a powerful software solution for processing laser scanning point clouds (Terrasolid, 2021). Here we aim for a classification routine via the generation of a smooth surface, which then can be classified disregarding remaining noise. In a first step, low reliable, noisy and isolated points are removed (see Figure 4.4 b). All points are sorted by x and y. If there are points with a high average

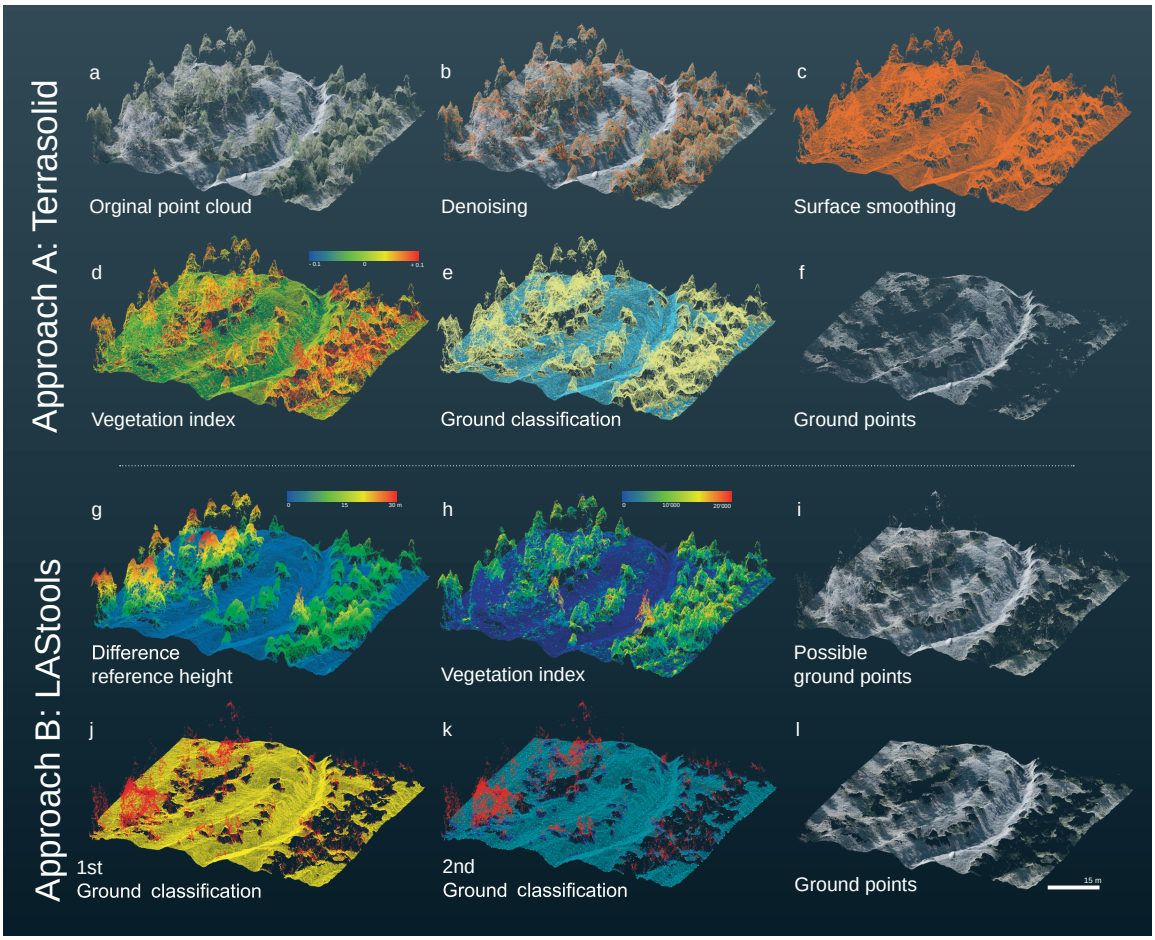


Figure 4.4. Terrasolid and LAStools workflows at an example area of the Arelen catchment. a) Original point cloud. b) Least reliable, noisy and isolated points are removed. c) A smoothed surface is generated for the subsequent ground classification. d) Generation of vegetation index. e) Ground classification f) Final classified ground points. g) Calculation of height difference to reference height model h) Generation of vegetation index. i) Potential ground points identified as non-green points and green point with less than 25 cm difference to the reference height model. j) 1^{st} ground classification with *step size* 1 m. k) 2^{nd} ground classification on tapered ground with *step size* 6 m and noise filtering and removal of all points 6 m above reference height model and 1 m above 1^{st} ground classification. l) Final classified ground points.

reliability ($>$10) in sphere of 0.5 m and its reliability value difference is at least 40% of average, points with low reliability are removed. The reliability is provided by the confidence parameter of Agisoft. Isolated and low noise points up in air or under the ground are deleted: If a point within groups of ≤ 15 points is more than 0.4 m vertically lower, but still within 2 m radius along xy plane to others, it is classified as a low point. If there are less than 15 points within a range of 3 m, points are considered as isolated. A smoothed surface is generated, which is used for the subsequent ground classification (see Figure 4.4 c). Potential surface points, that fit to locally smooth surfaces within a sphere of 5 cm, are extracted. Points with a high reliability are weighted more heavily. The potential surface points are smoothed along xyz. 25-30 neighboring points are used to fit a 2nd degree surface, whereby the maximum displacement of a point in any direction is 20 cm. Areas of high density are thinned out; for a volume of 7 by 7 by 7 cm, one central point is kept. Once again low and isolated points are removed. Points within groups of ≤ 10 points which are more than 0.4 m vertically lower than the others, but still within 1.5 m radius along xy plane to others, are treated

as low point. If there are less than 15 points within 2 m, they are considered as isolated (see Figure 4.4 c).

The now generated smoothed surface is used for the ground classification routine (see Figure 4.4 d-f). A vegetation index based on the RGB information of the points is calculated to optimise the ground classification algorithm: $(2 \cdot G - R - B)/(2 \cdot G + R + B)$ (Terrasolid, 2021). The values of this normalized vegetation index lie in the interval [-1,1]. Values above 0.05 are identified as green and thus likely to be vegetation. The aggregated colours are then smoothed by averaging the colour value by its nearest neighbours (average over a maximum of 10 points within a radius of 15 cm). We expect a maximum distance of any ground point of 60 m. A maximum permissible terrain angle of 75 degrees and an iteration angle of 15 as classification maxima was chosen. The maximum distance of a point to the triangle plane is set to 1 m. The iteration angle is reduced if the edge length is smaller than 8 m (Terrasolid, 2021).

4.3.2. Approach B: Point Cloud Classification with LAStools

LAStools is a dedicated tools for LiDAR point cloud post processing, allowing to tailor the workflow to the specific needs. We aim for a ground classification taking into consideration all potential ground points, even below the uppermost vegetation layer, which is applicable in steep, densely vegetated terrain (LAStools, 2022).

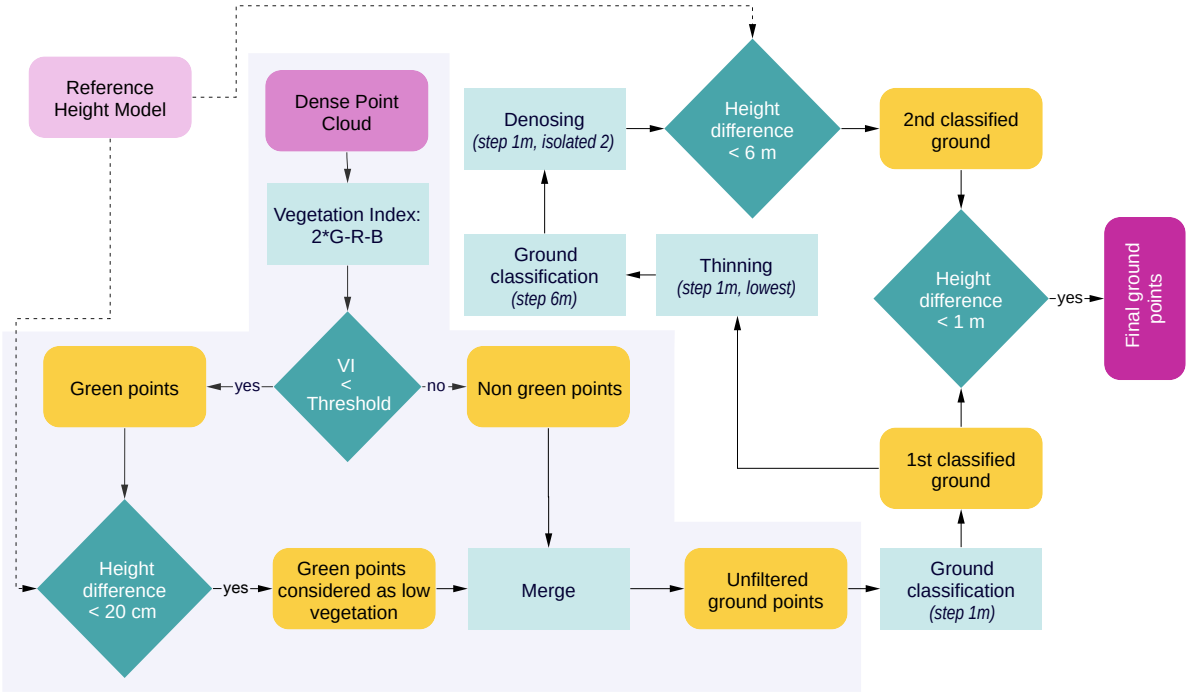


Figure 4.5. Flowchart illustrating the workflow of the LAStools routine. In a first step the potential ground points (low vegetation and bare ground) are selected (greyed background). In a subsequent iterative TIN densification with different step sizes remaining noise points are removed.

The algorithm is based on a color classification, a reference height model and repeated TIN densification (see Figure 4.5). The degree of green is defined by the vegetation index $2 \cdot G - R - B$ (Anders

et al., 2019) (see Figure 4.4 h). To be able to differentiate between grass and higher vegetation in steep terrain, we used a reference terrain model from Swisstopo (e.g. SwissSURFACE3D (Swisstopo, 2022b)) (see Figure 4.4 g). For the further processing we select suited ground points: i) non green points and ii) green-points within 20 cm, the uncertainty of the reference height model. A drawback is the remaining of dark points in the shade of high vegetation. Within the unfiltered ground points we applied repeated TIN densification routines using various step sizes to reduce those artifacts.

A first ground classification with a *step size* of 1 m is performed (see Figure 4.4 j). A thinning algorithm is applied to obtain the lowest point within an area of 1 m^2. A second, coarser ground classification with a *step size* of 6 m is applied upon this thinned ground and a noise algorithm deletes points, if there are less than 2 points in 1 m^3. Further, we calculate once again a height deviation to the reference height model: All point clusters that have a larger distance than 6 m to the reference model, such as trees, are deleted.

In a final calculation step, the difference between the first and the second rough ground classification is calculated. All points from the first classification that are more than 1 m above the second coarse classification (red points in Figure 4.4 k) are deleted. The remaining points from the 1^{st} ground classification (yellow points in Figure 4.4 j) are then finally considered as ground points (LAStools, 2022). All threshold values were elaborated empirically.

Once the ground points are classified, the point cloud is rasterised into a DTM with a 10 cm grid spacing. The resulting high-resolution DTM can then be further used in various processing steps of the hazard assessment, e.g. for the automatic extraction of torrential properties.

4.4. Automatic Extraction of torrential Properties

One of the objectives of this thesis is the extraction of torrent properties based on high resolution DTMs. In the previous section, two workflows to generate such DTMs were proposed. Subsequently, we present an algorithm that extracts torrential features in a automatic workflow. Flowcharts which promote the understanding of the algorithm can be found in appendix A).

To corroborate the automatic approach, we additionally estimated geometric torrential properties in the field (see Figure 4.2 b).

4.4.1. Delineation of the Torrent Bed

To extract the torrent bed, the Terrasolid digital terrain model is resampled to a resolution of 25 cm and then smoothed three times with a mean filter using 3-pixel neighbourhood. Sinks are removed in the smoothed DTM and derivatives such as flow direction and flow accumulation are calculated. After iterative testing, we decided to consider flow accumulation values greater than 40'000 as torrents. However, this threshold value is highly depending on the stream network characteristics of the catchment. The torrent pixels and flow direction allow the calculation of a polyline representing the streamline of the torrent. In order to calculate representative cross-sections, the polyline is smoothed with the PAEK algorithm (Polynomial Approximation with Exponential Kernel) with a floating path of 10 m length.

Transects with a width of 100 m (large enough to cover the whole "torrential extent") are generated perpendicular to the smoothed polyline. The spacing in between the transects is 0.5 m, dense enough to describe the cross-section geometry continuously along the torrent. The transects are converted into points to extract the height attribute, which is than assigned back to the respective transect line.

A major difficulty is to manage the confluence of different torrents. Since the transects have a width of 100 m, they tend to interact with the neighbouring torrent channel, which likely has a slightly different elevation. The confluence situation is shown in Figure 4.7 a at the example of the Arelen torrent. The torrent in subcatchment A has a higher elevation than the torrent in subcatchment B and would therefore influence the torrent bed extraction of the torrent bed in subcatchment B. To solve this problem, each section had to be calculated separately within the domain of its watershed. However, this effect distorts the embankment calculation at the confluence later in the process.

The corresponding pour point for the calculation of the watershed is defined by the 5th lowest point within the corresponding transect. Which means that the potential pour point is 2.5 m above the lowest transect point. Like this it can be assured that the accurate sub-catchment is calculated. The final pour point for the calculation is the deepest point within a search radius of 1 m around the potential pour point (see Figure 4.7 a). The raster catchment is converted to a polygon and buffered by 30 cm to represent the catchment boundary more realistically. Finally, the transects are clipped to the extent of the buffered catchment polygon and individually iterated.

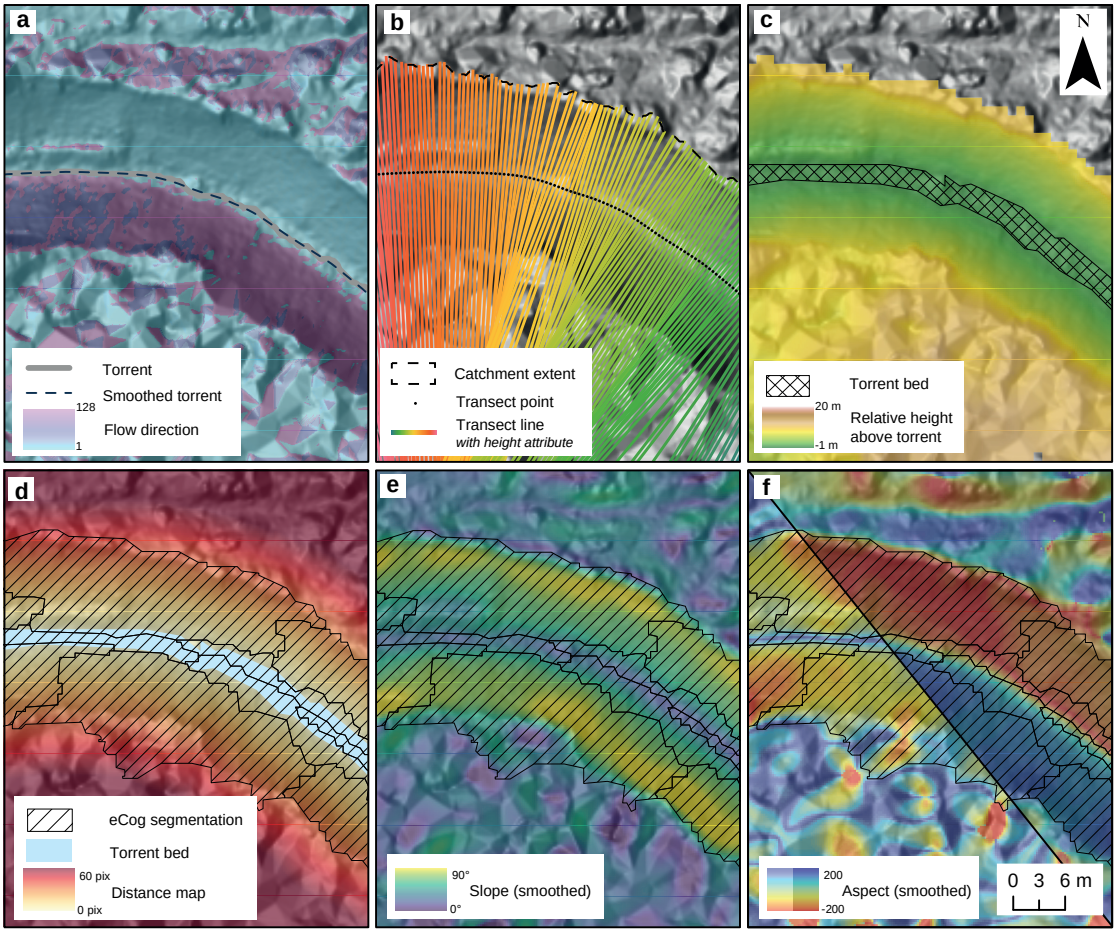


Figure 4.6. Extraction of torrent bed and input parameter which are used to run the eCognition segmentation. a) Based on a standard hydrology workflow applied on smoothed DTM the torrent is extracted and smoothed. b) Each 0.5 m a transect line (100 m length) is generated and clipped to the catchment extent. c) The transect lines are rasterised based on the height attribute and subtracted from the DTM. A condition of less than 25 cm relative height above torrent defines the torrent bed area. d) A distance map representing the distance to the torrent bed is calculated in eCogniton. The hatched surface indicates the resulting segmented torrential area. e) The slope is directly derived from the smoothed DTM. f) The two aspects are reclassified, so they represent together the aspect continuously.

The polylines, which are clipped to the extent of the catchment, are then rasterised with the DTM value (coming from the point) on a resolution of 1 m. This clipped raster represents the elevation of the torrent bed over a width of 100 m or up to the boundary of the calculated catchment. The torrent bed raster is resampled to a resolution of 25 cm. The relative elevation above the torrent bed is calculated from the difference between this grid and the DTM. We set a threshold of 25 cm difference to produce a binary raster that excretes the torrent bed polygon. These are merged with each other at the end of the iteration. We smoothed the joined polygons with an effective area algorithm of 1 m. Some polygons meet the 25 cm difference criteria outside of the torrent bed, especially on the fan apex. Using an intersection selection of the polygons with the smoothed torrent line, we can select the appropriate polygons. Within 2 m all polygons are being aggregated. This aggregation is needed to combine the polygons at the confluence. However, we got some unwanted aggregations. As for example if the two polygons of the two sub-catchment are closer than 2 m (see Figure 4.7 a).

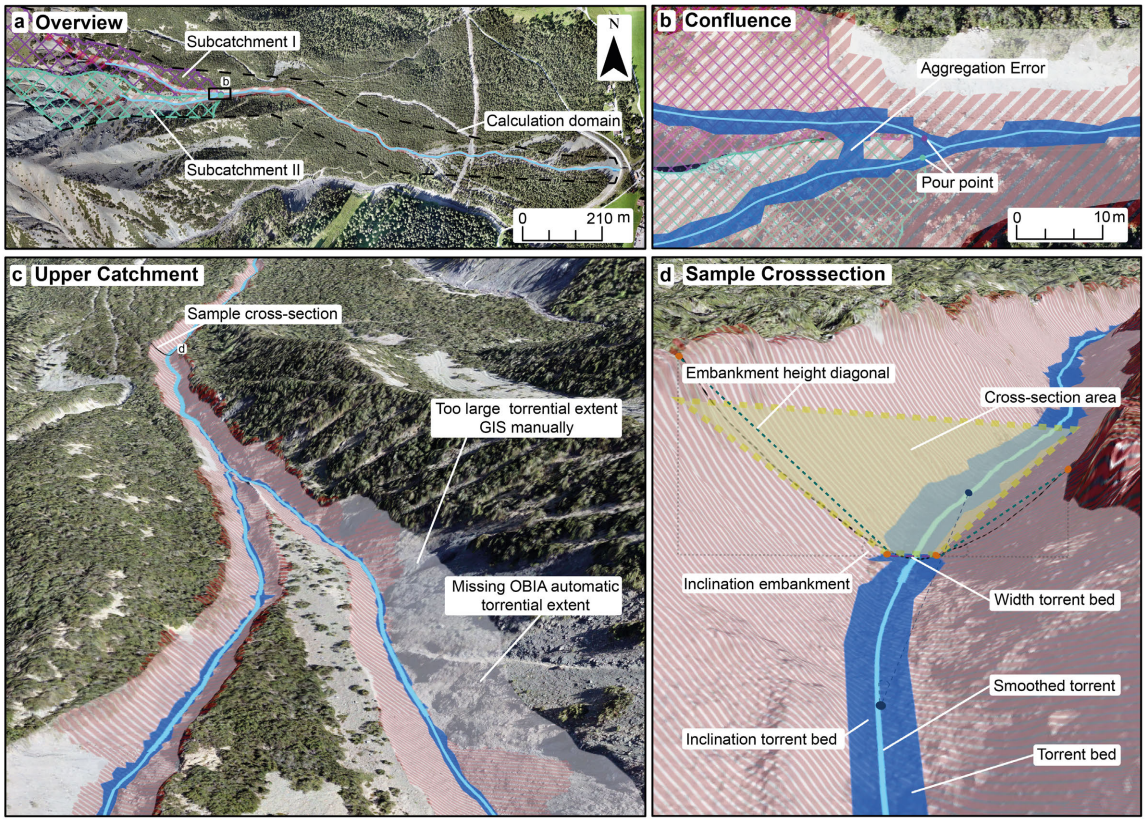


Figure 4.7. a) To prevent the interaction with the neighbouring sub-catchment the catchment areas are used to limit the calculation domain, this leads to calculation errors at the confluence. b) Overview of the calculation domain. c) Each 0.5 m all torrential properties are calculated based on the torrent bed and the embankment polygons. The cross-section area is a trapezoid surface, considering the mean of both embankment heights. The inclination of the torrent bed is calculated base on a 10 m horizontal distance. The OBIA based classification is partly limited due to triangulated DTM surfaces caused by dense vegetation. d) In the upper catchment the classification of the embankments is limited by steep adjacent slopes.

4.4.2. Delineations of Embankments with Object based Image Analysis

Based on the derivatives from the high-resolution DTM, such as slope and aspect, we generated objects with similar properties in the Object based Image Analysis (OBIA) software eCognition (Trimble, 2022) (see Figure 4.6).

The original DTM is resampled to a resolution of 25 cm and then smoothed 10 times with a mean filter of 3 by 3 cells. Hydrological depressions are filled and slope and aspect are calculated. In eCognition, the grid value is used to segment homogeneous objects. The aspect values are reclassified since there is a shift between 0 and 360°. The two generated reclassification maps, consisting of 130 classes, are congruent, meaning that the effect between 0 and 360° is no longer present. This calculation is done in a Python script, which is found in B.3. In eCognition a primary segmentation considering the slope and the generated torrent bed shapefile is used to classify background cells that will be ignored within the further classification procedure. The objects covering the torrent bed shapefile are assigned to the class torrent bed. In the domain of the unclassified objects a distance map is calculated, which represents the distance to the class torrent bed class. Based on the two associated aspect files, the slope and the distance map, a multi-resolution segmentation is performed. This procedure minimises

the average heterogeneity of the image objects at a given resolution. We weighted the slope layer the most with 15. Both aspect layers are weighted with 2 and the distance map is weighted with 4. To obtain the torrent geometries, the torrent bed shapefile was used as a thematic layer. For the multi-resolution classification we used 80 as scale parameter and 0.5 for shape and 0.7 for compactness. Those parameter we identified through iterative testing and visual interpretation.

This segmentation creates homogeneous objects that represent the torrent bed and both embankments. One of the most difficult tasks is to automatically select the correct objects and assign them to the embankment class. Based on a repetitive process of trial and error, we created a list of different conditions. The mean slope of the object must be between 10° and 90°. The mean distance map index must be less than 30. The 10th and 90th quantile of the distance map must be lower than 12 respectively 90, and the standard deviation must not exceed 18 (see 4.2). All objects that completely meet these requirements are exported as polygons and used in the following Python script as automatically generated "torrential extent".

Table 4.2. Criteria used to classify the "torrential extent" within eCognition.

Entity	Criteria
Mean slope	>10 and <90
Mean distance map	<30
10th percentile distance map	<12
90th percentile distance map	<90
Standard deviation distance map	<18

4.4.3. Calculation of Torrent Properties

Our goal was to write a script that would allow fully automatic extraction of torrent characteristics that would work in different torrent catchments. The respective Python script can be found in B.4.

Since the classification of the slopes tend to be erroneous, we have set up some modifications to obtain more realistic values:

(i) Some (<2%) automatically segmented "torrential extent" polygons extend far beyond the actual extent. With a 20 m buffer around the torrent bed, we can crop the eCognition output to the extent of the 20 m buffer file. With this workflow, we can prevent extreme outliers.

(ii) Some other (∼5%) automatically segmented polygons remain within the extent of the torrent bed. To prevent a breakdown of the algorithm, we merged the torrent bed buffered with 1 m with the "torrential extent" polygon from eCognition. With this procedure we created a polygon, which represents the embankments continuously.

Finally, because we classified all values 0.2 m above the smoothed torrent as the torrent bed, we applied a negative buffer of 0.2 m to obtain more realistic values. After those modifications we iterate over all section and transects. For each geometry type (embankment (left, right) and torrent bed), all results are stored in a Numpy array. The clipped transect files (already generated within the last script for the torrent bed polygon) are clipped again to the extent of the merged eCognition output. They are then joined to a feature class where all transect polylines are collected.

Points are generated every 25 cm along the trimmed transect file. The elevation value from the DTM is stored as an attribute to the point feature. The statistics tool is used to determine the median and maximum transect ObjectID within the section. Using the maximum transect, we know how many transects are within the section. And the median transect helps us to select the right polygon on the left and right side: The point with the lowest ObjectID within the median transect is used to select the right embankment polygon, and the highest ObjectID respectively for the left embankment polygon. To make this selection procedure work, the two sides (left/right) of the embankment polygon must be cut into separate parts.Therefore, we selected the first and last transects and created a buffer of 0.5 m around those polylines and erase them from the embankment file. In addition, we also intersect the manipulated embankment polygon with the buffered watershed polygon. After applying the multi to single polygon routine we are able to select the right and left embankment polygon based on the median transect with select by location of 1 m distance.

For each section, the generated point file was clipped by the three polygons (embankment left, embankment right, torrent bed). Using a while loop, we iterate over each transect and compute the torrent bed, left embankment, and right embankment properties. The calculations of the right and left embankment properties are redundant. A selection analysis selects all points that belong to the transect selected in the loop. Applying the statistics tool, we search for the minimum and maximum DTM values and for the point with the highest and lowest ObjectID. The height is calculated by subtracting the maximum and minimum DTM values. The width derives from the subtraction of the highest ObjectID and the lowest ObjectID multiplied by the cell size of 0.25m. By applying the pythagoras on the two parameters the diagonal distance is resulting. The inclination is the sinus arc from the division of height and diagonal distance.

For the calculation of the properties of the torrent bed, all points of the respective transect within the torrent bed point file are selected. Mean, standard deviation, maximum and minimum values are calculated. From these values, the parameters torrent bed width, mean height, and standard deviation are derived.

Based on torrent bed and embankment properties the cross-section area is automatically derived (see Figure 4.7 c). This measure should be understood as a measure for the capacity of the torrent at a specific cross-section. To not be strongly influenced by to large or to small classified embankment polygons we use the mean height of both embankment polygons for the height of the trapezoidal surface. The width is derived from the mean of the torrent bed width and the sum of torrent bed width, embankment width left and embankment width right. The mean torrent inclination is calculated over a linear distance of 10 m. The 10th transect below and above the specific transect are selected. From the height difference and the length of 10 meters the inclination is derived. By performing these iterations over all transects and sections, the calculated values are appended to the Numpy arrays. From these values, a combined Pandas data frame is generated. Based on attribute join, the Pandas data frame is linked to the feature class in which all polylines from each section were iterative selected.

4.5. Change Detection

Photogrammetric UAV point cloud data can be classified into ground points and subsequently a high resolution DTM can be derived. Such generated DTMs are not strongly robust, especially in vegetated areas some noise is likely to remain and cause irregular triangulations. In chapter 2 we introduced various methods to derive differences between point clouds. In this thesis we focus on the difference measure with the M3C2 algorithm (Lague et al., 2013). As the algorithm is based on points it calculates only distances along ground classified surfaces and is not depending on interpolations. As the distances measure is orientated in normal direction, steep embankments slopes are represented more realistic. Other distance measures such as C2M or DoD are not further introduced. Nevertheless, we compared results from C2M and DoD with M3C2 distance measures at the example of a debris flow event at Fraschmardin.

For an accurate change detection, properly co-registered point clouds are essential. There is a immense variety of different proposed registration workflows. Depending on the magnitude and type of error between the two acquisitions different methods may have to be applied. Following, we present three workflows that we experienced as applicable with photogrammetric point clouds. Overall, the science of co-registration and removal of systematic errors in the data has to be seen as an art. There is not one specific solution, which works out for all cases.

4.5.1. Co-Registration with ICP

Within small study extents and along highly pronounced terrain often no systematic error correction is needed. A simple ICP (Iterative Closest Point Algorithm) (Besl and McKay, 1992) can be applied to co-register the point clouds. If there are areas, where we can assume that no change occurs, it is advisable to restrict the ICP calculations on those areas. If this is not the case the algorithm has to be applied on the whole region. This imposes the risk that areas are compared with each other, which should not be compared, and the results become even worse. However, results of such ICP routines have always to be analysed carefully.

4.5.2. Co-Registration with common Tie-Points

Cook and Dietze (2019) introduced a novel technique to co-register the two point clouds directly along the process of tie-point matching. All pictures of both acquisitions are aligned together in Agisoft. The dense cloud generation is done using the tie-points of the specific acquisition. As the two acquisitions are using common tie-points for stereo-matching they tend to match accurately on each other. However, this procedure produces two different camera calibration settings and is therefore not capable to correct systematic errors with a spatial dependency. We applied this registration workflow at the example of the debris flow event in 2019 in Fraschmardin.

4.5.3. Systematic Error Modelling

As we compare two point clouds of different flight missions we have to check for a systematic error in the DoD. If there is a clear spatial dependency, such as doming effects, we need to model this error and make a correction. This error arises from systematic errors in the estimate of the camera geometric model during the photogrammetric processing (Hastedt and Luhmann, 2015). In the following, we present the systematic error handling applied in this work.

Using the Python script provided by James et al. (2020) we export a tie point cloud with the associated uncertainties from Agisoft. We convert the tie-point coordinates into a MATLAB matrix and calculate the difference between those two data sets. Based on this difference we were able to compute an error surface, which represents the systematic error between those two flight missions. We used linear least square 22 polynominal function with robust bisquare. We added the resulting error surface as scalar field to the tie point cloud of the second flight mission.

In CloudCompare we were able to interpolate the scalar field values of the error surface on the dense point cloud of the second flight mission. Like this the systematic error is interpolated to a scalar field of the second dense point cloud. Subsequently the z-coordinates of the dense point cloud are saved as scalar field. A new scalar field is derived based on the summation of the systematic error and the z-coordinate of the second point cloud. Finally, this new scalar field is set as z-coordinates of the second point cloud. Once the correction of the systematic error is applied we may run an additional co-registration in CloudCompare using ICP (Besl and McKay, 1992).

4.5.4. M3C2 Distance

To run the M3C2 algorithm with precision maps we interpolate the derived uncertainty measures from the tie-point cloud as a scalar field to the respectively dense point cloud. In densely vegetated terrain there are too little tie-points for the interpolation. In such cases we do not derive a precision map. If it is possible to derive a distance uncertainty, areas of significant change can be extracted. However, for the application along vegetated torrents this is often not applicable due to the effect of occlusion by vegetation. For our case studies we have used a cylinder and projection area of 0.5 m, a point cloud density of 5 cm and an uncertainty estimation of 10 cm.

Results

In the following chapter we evaluate the accuracy of the dense point cloud and assess the quality of the derived DTMs. Since the accuracy of the classified photogrammetric point cloud is low along highly vegetated areas, we compare the results with LiDAR flights in the catchment of Davos Arelen. Based on the DTMs generated with Terrasolid, we present the results of the calculation of the automatic torrential properties workflow.Furthermore, we present examples of the Fraschmardin, Arelen and Carrerabach catchments using difference calculations, where we could quantify interesting mass-wasting processes.

5.1. Accuracy of Dense Photogrammetric Point Clouds

To quantify the accuracy of dense photogrammetric point cloud data we performed a test flight with the WingtraOne and the DJI Phantom 4 RTK on 25.08.2021 in the Arelen catchment. All 36 local reference points were used as control points (CPs) for the processing of the measurement data with Agisoft. For this analysis we compare filtered and non filtered tie-points. The results of the processing report are depicted in Figure 5.1. A summary of the errors is presented in Table 5.1.

Table 5.1. Overview of error measures of UAV flight with DJI Phantom 4 RTK and WingtraOne.

	overall error	x-error	y-error	z-error	yaw-error	pitch-error	roll-error	camera location error
WingtraOne	8.4 cm	2.23 cm	5.35 cm	6.08 cm	1.59°	1.12°	0.92°	3.8 cm
DJI Phantom 4 RTK	9.46 cm	1.82 cm	6.42 cm	6.72 cm	2.76°	0.38°	0.64°	1.72 cm

The flight altitude with the Wingta UAV was 230 m in average with a velocity of 16 m/s. The tie point density is 0.38 points/m^2. With the DJI Phantom 4 RTK the average altitude was 146 m and the velocity 8 m/s. The resulting tie point density was 0.56 points/m^2. From one WingtraOne image, it was possible to generate 802 tie-points; from one DJI Phantom 4 RTK image 920 tie-points could be generated. The overall mean error over all 36 CP is 8.4 cm for the WingtraOne (see Figure 5.1 a) and 9.5 cm for the DJI Phantom 4 RTK (see Figure 5.1 e). Considering the elevation error, the error for the WingtraOne is 6.1 cm and for the DJI Phantom 4 RTK 6.7 cm. The grid spacing of the DSM is 6.65 cm/pix and a point density of 226 points/m^2 for the WingtraOne data and 8.58 cm/pix with a point density of 136 points/m^2 for the DJI Phantom 4 RTK. The camera location of the Phantom 4 RTK (see Figure 5.1g) is more accurate with an average error of 1.7 cm. Especially along the upper flight lines the uncertainties become larger. This is likely due to the increased wind in higher altitudes. The camera orientation error is particularly low at the end of the flight lines. The yaw error of the DJI UAV is with 2.7° significantly larger compared to the pitch (0.38°) and roll (0.64°) error, which leads to a total error of 2.9° (see Figure 5.1 h). The camera location accuracy of the WingtraOne (see Figure 5.1 c) flight has an average error of 3.8 cm. The errors are larger compared to the DJI Phantom 4 RTK flight along all dimensions. We explain this uncertainty by the

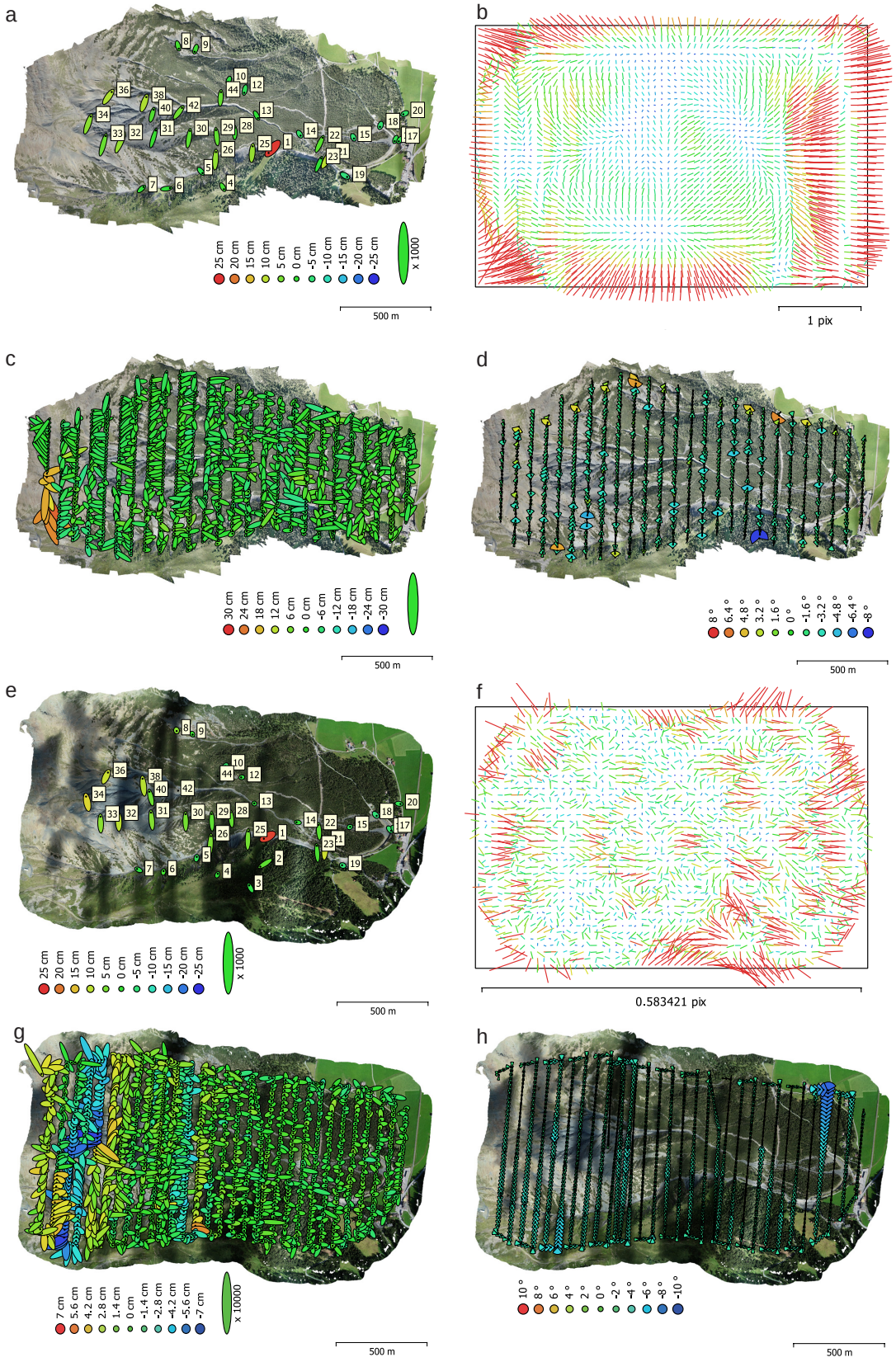


Figure 5.1. Results of dense cloud processing in Agisoft with Wingtra One (a-d) and DJI Phantom 4 RTK (e-h) images. a/e: Three-dimensional error of CPs, b/f: Camera distortion, c/g: Three-dimensional error of camera location d/h: Camera location error.

higher flight velocity. The overall camera orientation error is lower in relation to the DJI Phantom 4 RTK flight. The yaw (1.6°) and pitch (1.1°) are a little larger than the roll error of 0.9° (see Figure 5.1 d). The more stable flight conditions may be explained by the aerodynamic properties of the fixed-wing UAV.

We found that the accuracy of the camera location error is more pronounced along the WingtraOne flight. However, the resulting error based on the 36 CPs and the resulting dense point density is higher (with the Wingtra flight). In addition, the ratio between the amount of tie-point/image is lower with the WingtraOne flight. We explain these results by the difference in focal length (24mm DJI Phantom 4 RTK and 35mm WingtraOne). As we have gradually selected the accurate tie-points along both acquisitions with setting a threshold for the projection error by 0.4 and for the reconstruction uncertainty by 10, we reduced the amount of tie points to 290 for one image for the WingtraOne flight and 340 tie-points for the DJI Phantom 4 RTK flight. This reduction could reduce the overall error by 0.1 cm for the DJI Phantom 4 RTK measurement. For the WingtraOne we found an overall accuracy improvement of 8.2 cm, which is a reduction of approximately 0.2 cm.

To summarize: the WingtraOne is able to capture large areas. Due to the high quality of the mounted camera (see Figure 5.1 b) the resulting point density and grid spacing of the DTM is better than of the DJI Phantom 4 RTK acquisition, which carries a low cost action camera (see Figure 5.1 f). The WingtraOne UAV flies twice as fast as the DJI Phantom 4 RTK. The high velocity leads to a decrease in the camera location accuracy, however the influence on the overall accuracy is still better compared to the DJI Phantom 4 RTK. We could fly the whole catchment within two flights starting from a parking at the catchment outlet. For the Phantom 4 RTK in total six flights were needed. Three different take-off locations evenly distributed over the catchment were used, because the software limits the flight altitude to 500 m above take off point. If the terrain permits it, we propose to fly with the VTOL fixed-wing WingtraOne. The UAV provides highly accurate results with a short time demand for the acquisition.

5.2. Quality of different UAV-based Digital Terrain Models

To create a DTM-based on UAV photogrammetric data, we developed and tested many workflows. Finally, we have developed two algorithms that give consistent results. With LAStools we have developed a routine to classify points based on greenness and relative height deviation to a reference terrain model. In addition, we examined a Terrasolid point classification routine, which generates a smoothed surface and classifies this surface within a subsequent step. Below we compare the resulting ground classification of the two routines and relate them to LiDAR-UAV acquisitions and Swisstopo products SwissALTI3D (Swisstopo, 2022a) and SwissSURFACE3D (Swisstopo, 2022b). The SwissALTI3D is a digital elevation model, which describes the landsurface without vegetation and development. It is updated every 6 years. Along our study sites there is currently only an old version with a 2 m resolution available. Depending on the altitude the DTM is either derived from stereo correlation (up to 3 m spatial resolution) or LiDAR acquisition with 0.5 m resolution. The second used product is the SwissSURFACE3D. It describes all natural and man-made objects of the

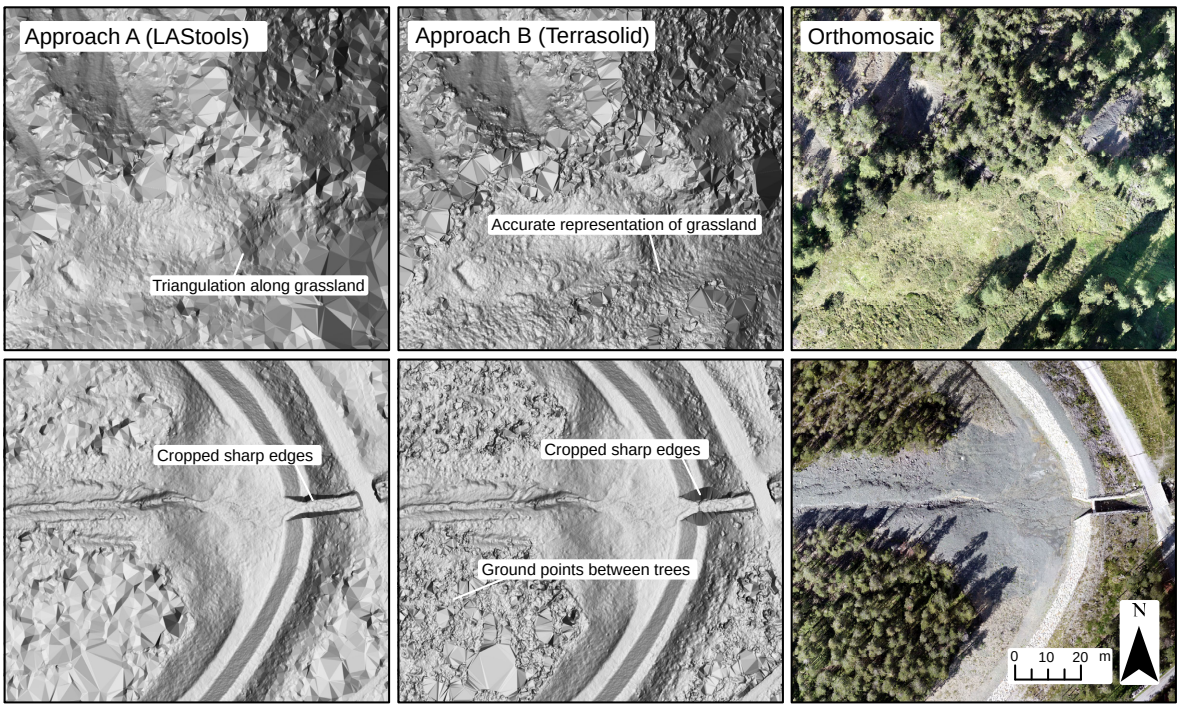


Figure 5.2. Qualitive comparison of LAStools and Terrasolid. Low vegetation such as grass is often interpolated with the Terrasolid algorithm. Sharp edges from constructions are stronger cut with than Terrasolid.

surface in the form of a classified point cloud. The minimum guaranteed ground points density is 5 points/m^2, the average ground point density is 15-20 points/m^2. The planimetric accuracy is $\pm$20 cm and the altimetric accuracy is $\pm$10 cm. For the subsequent analysis all classified ground points are rasterized to a 10 cm resolution DTM.

When analysing the hillshades in Figure **??** we can highlight the advantages and disadvantages of each routine. One of the major advantages of the LAStools-based algorithm is the reliable distinction between grass vegetation and shrubs. The Terrasolid routine tends to classify grasslands as vegetation. In contrast, the LAStools routine uses the reference height and can therefore correctly distinguish grassland from shrub forest in steep terrain. With the LAStools routine we had difficulties with the non-green points in the shade of the vegetation remaining after the classification. We therefore had to use a very coarse and aggressive noise removal algorithm to remove points in these shaded areas. LAStools has the disadvantage that edges of prominent surfaces such as steep rock faces, dams and retention basins are cut off stronger than observed with Terrasolid (see Figure **??**). In general, both algorithms retain some high vegetation points that are wrongly classified as ground. Figure **??** has a spatial resolution of 10 cm. Resampling the DTM to a 1 m grid gives a smoother and less erroneous surface. However, there are limitations to both photogrammetry-based methods.

In the following section we quantify the advantages of LiDAR UAV data. We evaluate the quality of two different LiDAR flight resolutions in terms of the density of classified ground points and the resulting ground elevation and compare it with the two photogrammetric routines and Swisstopo products in the Arelen catchment.

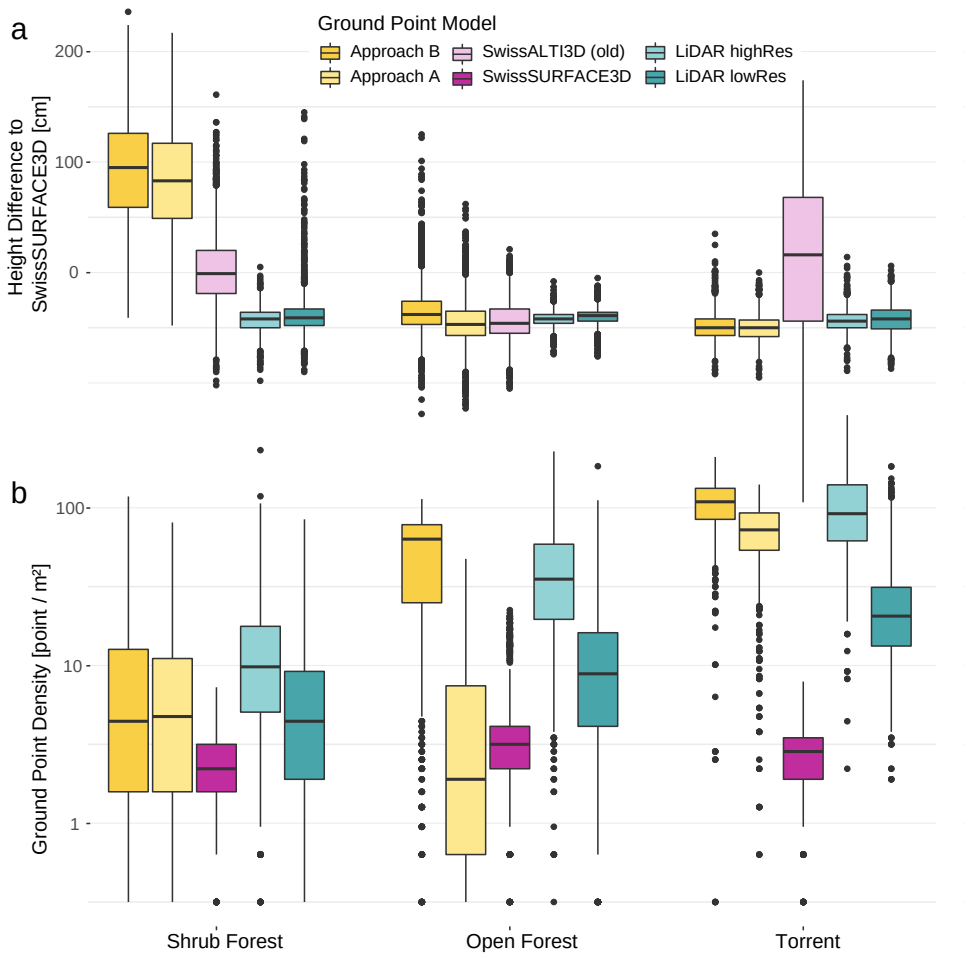


Figure 5.3. Boxplots for three sample areas using 2000 random sample points (shrub forest, open forest and an area covering the torrent). The Terrasolid (Approach A) and LAStools (Approach B) routine are based on photogrammetric point data originating from a Wingtra UAV flight. The LiDAR data (DJI M600 pro with a Riegl miniVUX-3UAV) is classified with a standard LAStools routine. a) Height deviation in comparison to the swissSURFACE3D data. b) Ground point density of different ground point clouds.

5.2.1. Ground Point Density

The ground point density describes the number of points classified as ground per square meter. We compare three different land cover types: Shrub forest, open forest and torrential terrain. For each of these sample areas, 2,000 random sample points were generated.

Across all types of point cloud generation, we observe low point densities within shrub forest areas. Looking at the boxplots in Figure 5.3, the proportion of ground points with Terrasolid and LAStools is rather high. It should be noted that points classified as ground are not necessarily true ground points. From Figure 5.3 we can anticipate that photogrammetricaly derived Terrasolid and LAStools ground points are generally 1 to 1.5 m higher than the ground points of SwissSURFACE3D in shrub forest. The density of SwissSURFACE3D ground points is very consistent with a ground point density of 5-20 points/m^2. LiDAR UAV derived point clouds tend to have a much higher ground point density along vegetated terrain.

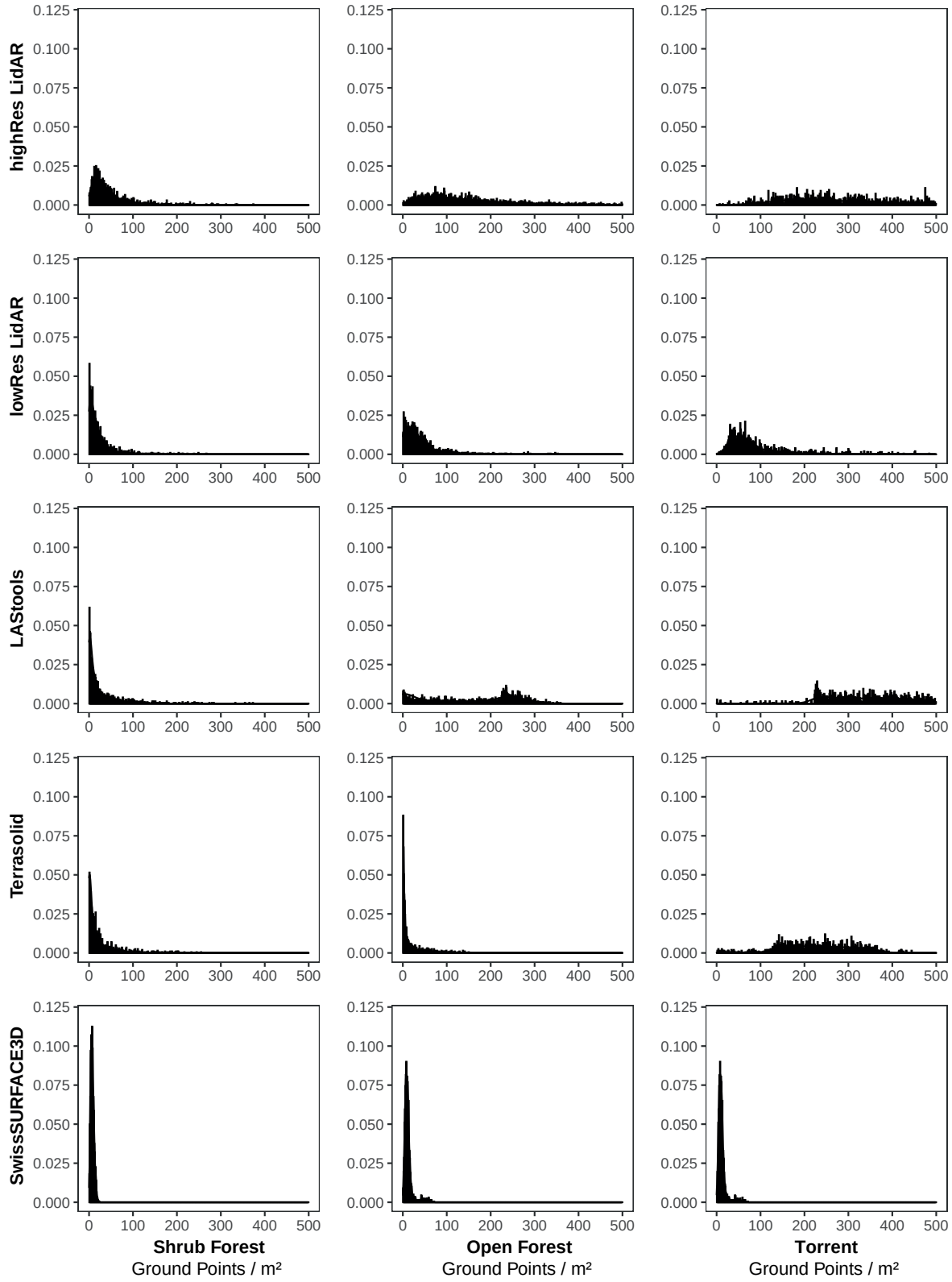


Figure 5.4. Histograms indicating the ground point density of different point cloud routines, for three sample area using 2000 random sample points (shrub forest, open forest and an area covering the torrent).

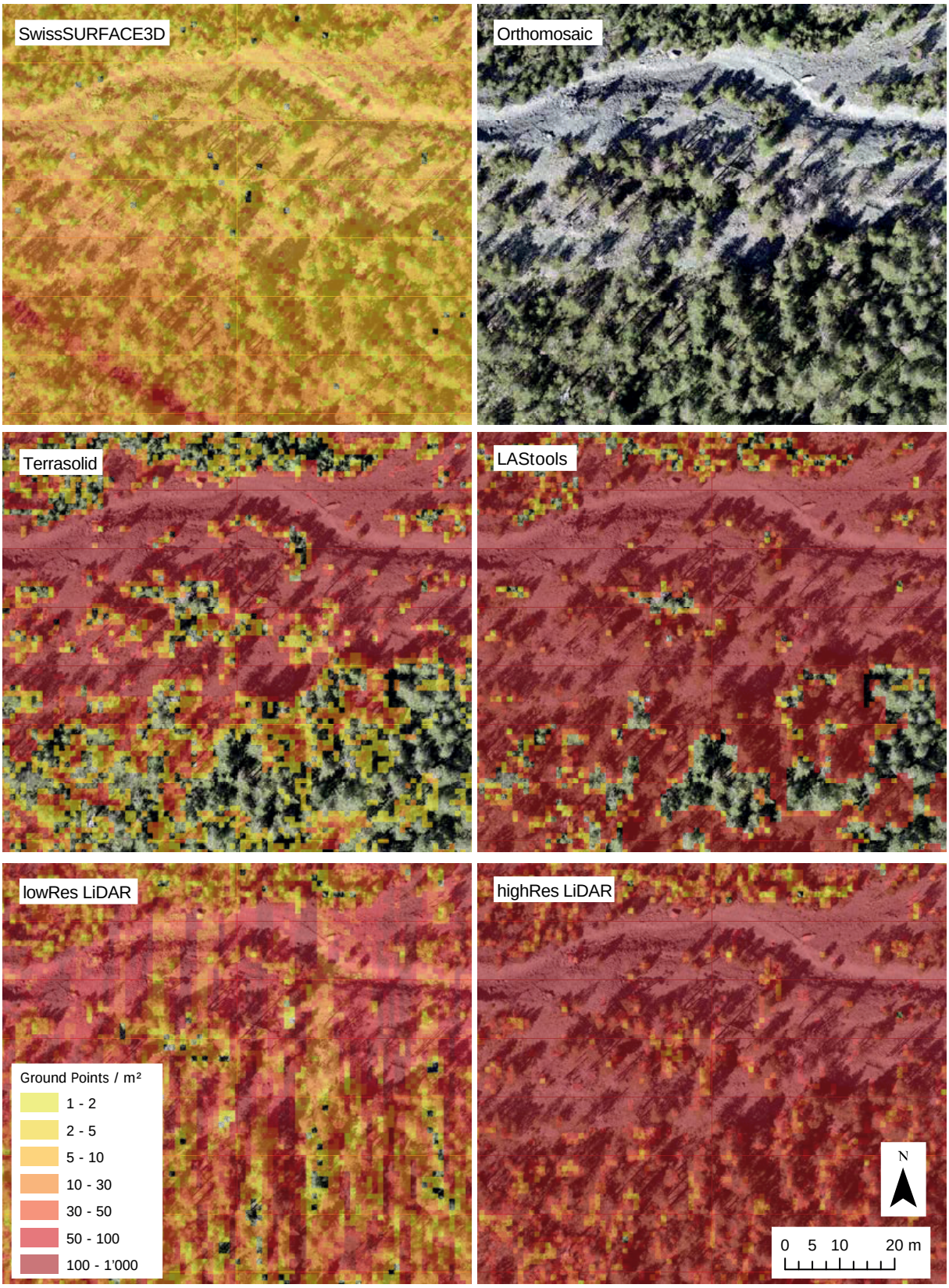


Figure 5.5. On this map a partly vegetated debris flow cone is shown. Each map shows the ground point density of the different point cloud sources (SwissSUFRACE3D, Terrasolid, LAStools, low resolution LiDAR, high resolution LiDAR).

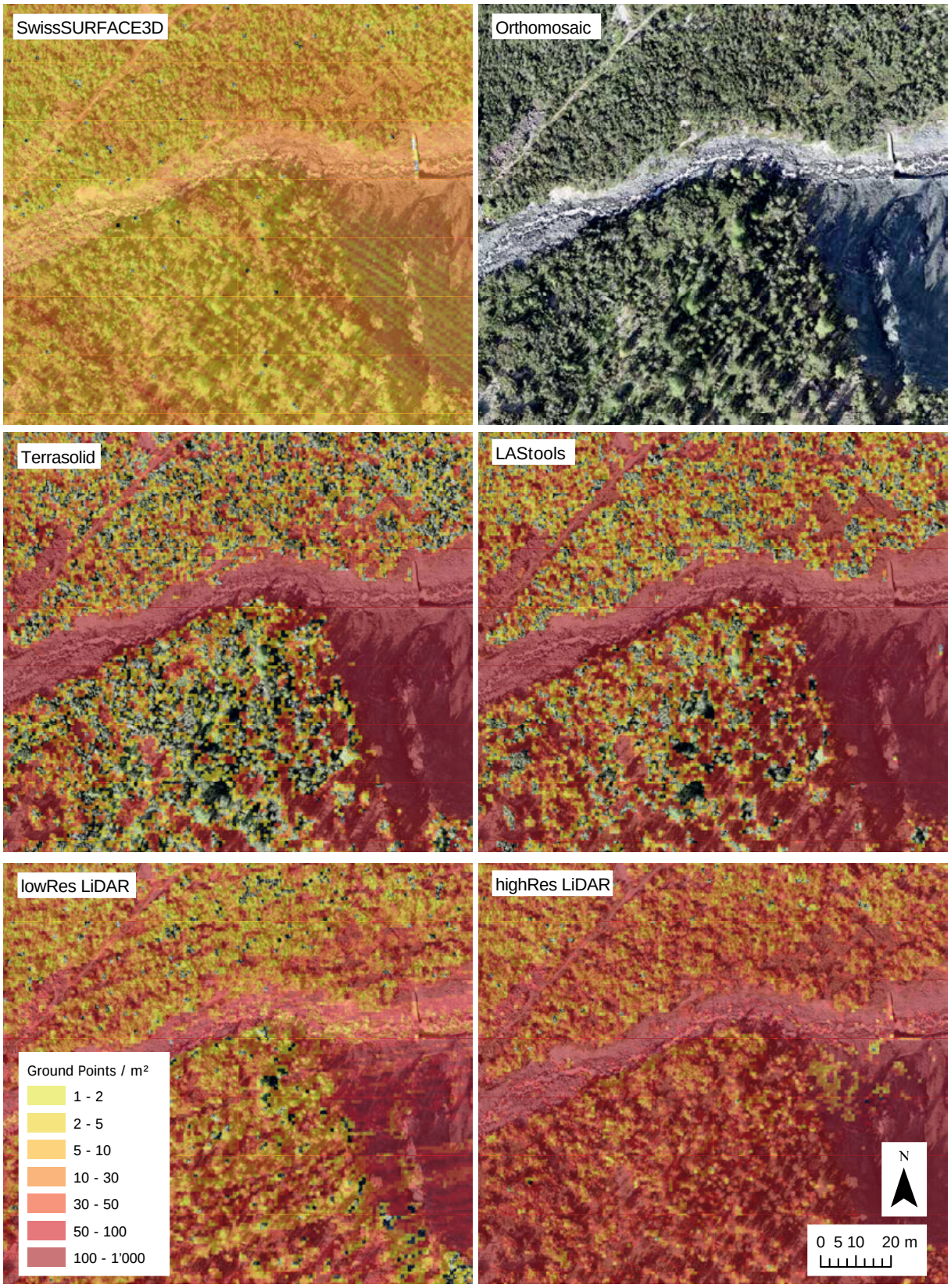


Figure 5.6. Each map shows the ground point density of the different point cloud sources (SwissSUFRACE3D, Terrasolid, LAStools, low resolution LiDAR, high resolution LiDAR).

In shrub forests, the ground point densities of the Terrasolid routine (approach A) and the LAStools routine (approach B) are very similar. In open forests we examine large variations. The LAStools routine has a high ground point density in open forest. Below trees, we find no ground points in either routine. However, in the open areas between trees we observe a higher density of ground points along the results of the LAStools routine. Here, the Terrasolid routine shows many areas with only a few ground points. We also see that the areas without ground points under trees are generally larger in the Terrasolid output (see Figure 5.5).

The boxplots of the low and high resolution LiDAR data in Figure 5.3 show a higher ground point density than the SwissSURFACE data. The high-resolution version contains about 2.7 times more data points than the low-resolution one. In open terrain such as the torrent area, the difference between the two resolutions is the greatest. In densely vegetated shrub forests, the deviations are smaller.

Photogrammetric point clouds have in principle a higher point density than LiDAR data. This can be well observed along open terrain such as in the torrent. The LAStools-based approach B tends to have more ground points than the Terrasolid approach A. This is explained by the fact, that the surface is smoothed and thinned out within the Terrasolid workflow. The histograms in Figure 5.4 clearly show that for the photogrammetric and low-resolution LiDAR data in the shrub forest and open forest, most points are around 0 m elevation.

The low-resolution LiDAR flights have only one flight direction. This makes the data strongly dependent on the relief and large objects such as tall trees. When the LiDAR UAV flies along steep terrain the angle of view is greater in downslope direction. In upslope direction the view angle is strongly limited and for example large tree can cause a major shadow (occlusion). However, along flat terrain we observe the same effect but less pronounced. The shadow effect of large trees can be reduced with an increased flight altitude or a denser flight pattern. But for this purpose the UAV-based LiDAR is not strong enough. In the SwissSURFACE3D data this effect is not that pronounced, because the flight altitude is much higher. The ground density of the SwissSURFACE3D varies between 5 and 20 points/m^2 and is not dependent on the terrain shape and vegetation (see Figure 5.6).

5.2.2. Digital Terrain Model

A DTM is computed based on the classified point cloud. The hillshades visible in Figure 5.9 and 5.8 are derived from a DTM with 10 cm spatial resolution. For reference, the hillshades from SwissSURFACE3D and SwissALTI3D are also shown. It can be seen clearly that the resolution of SwissALTI3D is much lower than that of the others. It can be further seen that the photogrammetric UAV products have a very high resolution in open areas. In overgrown terrain, many triangles indicate that only few ground points are found. Looking at the hillshade, it seems that these points are actually on the ground. In fact, the photogrammetric ground points in the shrub forest area are $\sim$1.4 m above the height of the LiDAR-based ground points. This effect is clearly visible in the boxplots in Figure 5.3, where the newly available SwissSURFACE3D data is plotted in relation to the other derived surface models.

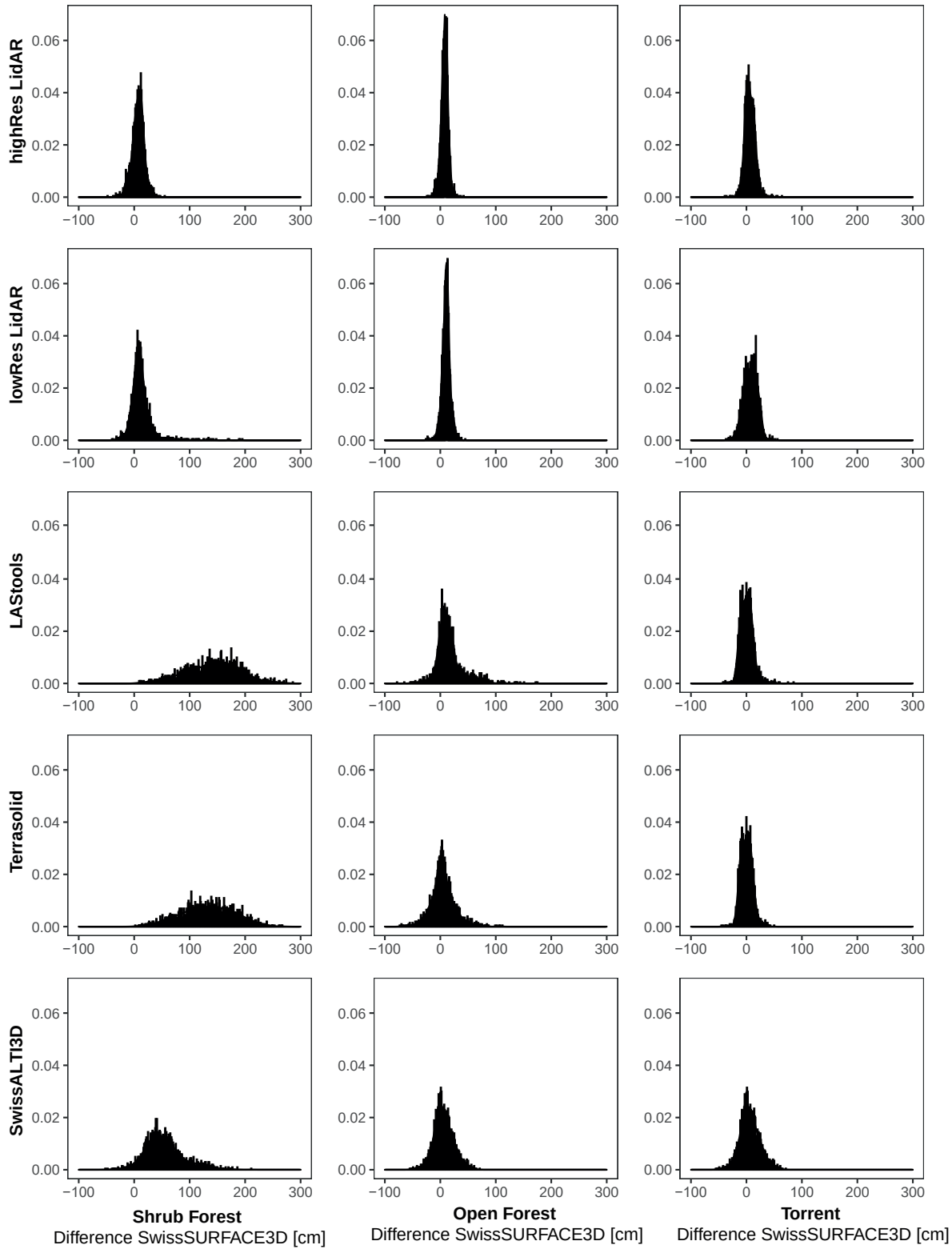


Figure 5.7. Histograms indicating the deviation of the height in comparison to the SwissSUFACE3D data, for three sample areas using 2000 random sample points (shrub forest, open forest and an area covering the torrent).

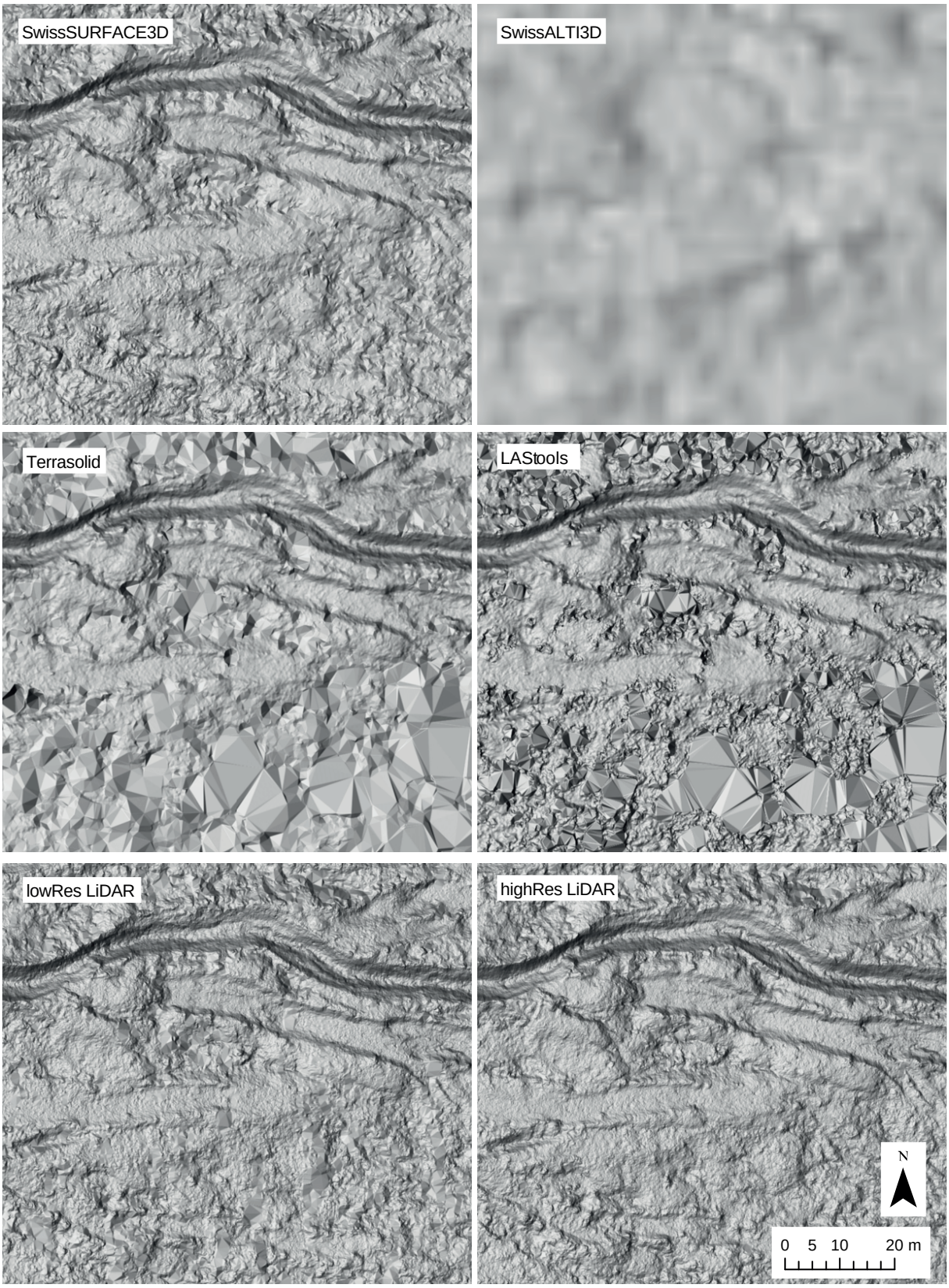


Figure 5.8. Each map shows the multi-directional hillshade of the different DTMs (SwissSUFRACE3D, Terrasolid, LAStools, low resolution LiDAR, high resolution LiDAR).

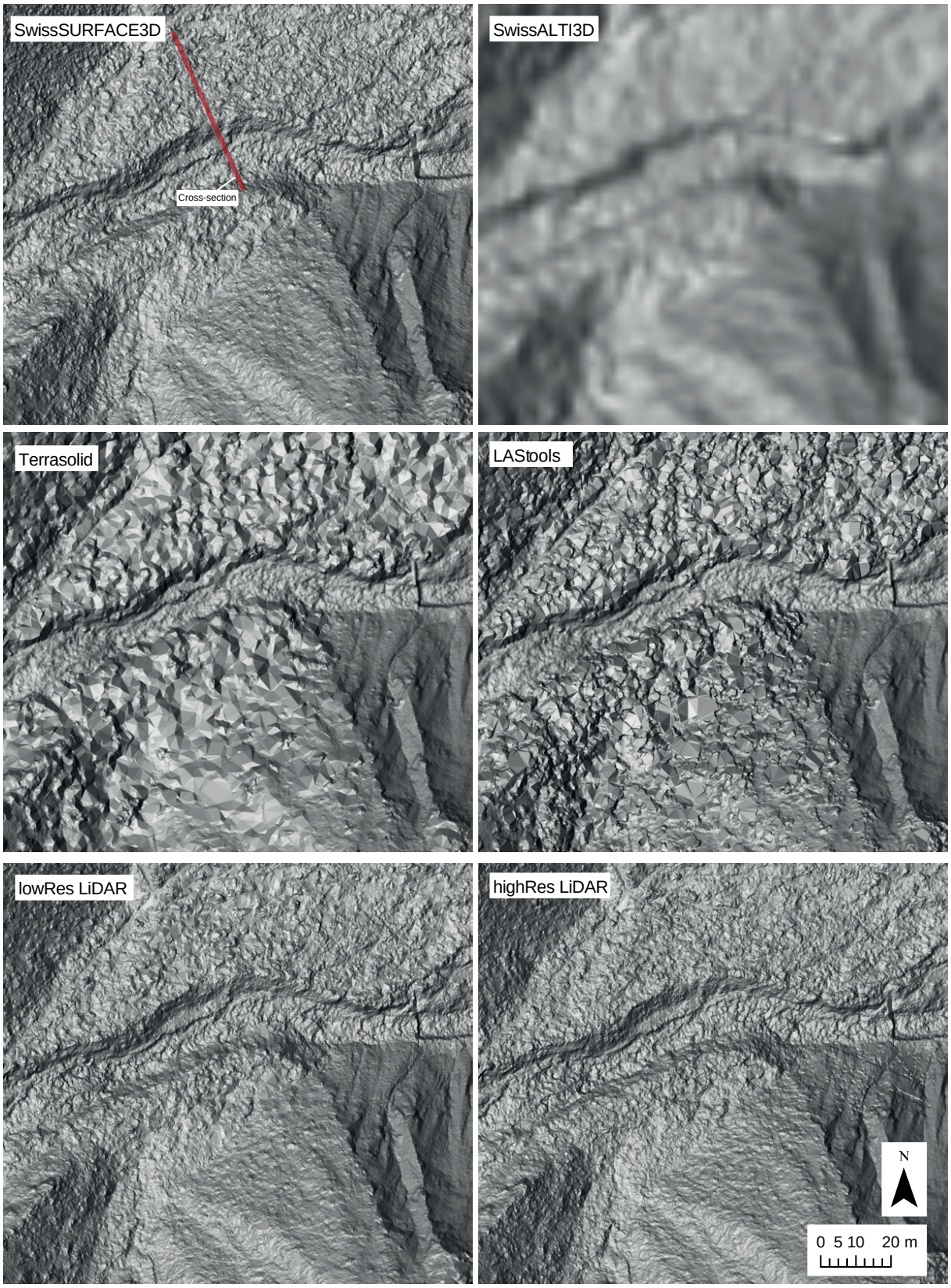


Figure 5.9. Each map shows the multi-directional hillshade of the different DTMs (SwissSUFRACE3D, Terrasolid, LAStools, low resolution LiDAR, high resolution LiDAR). The red line indicates the analysed cross-section in Figure 5.10.

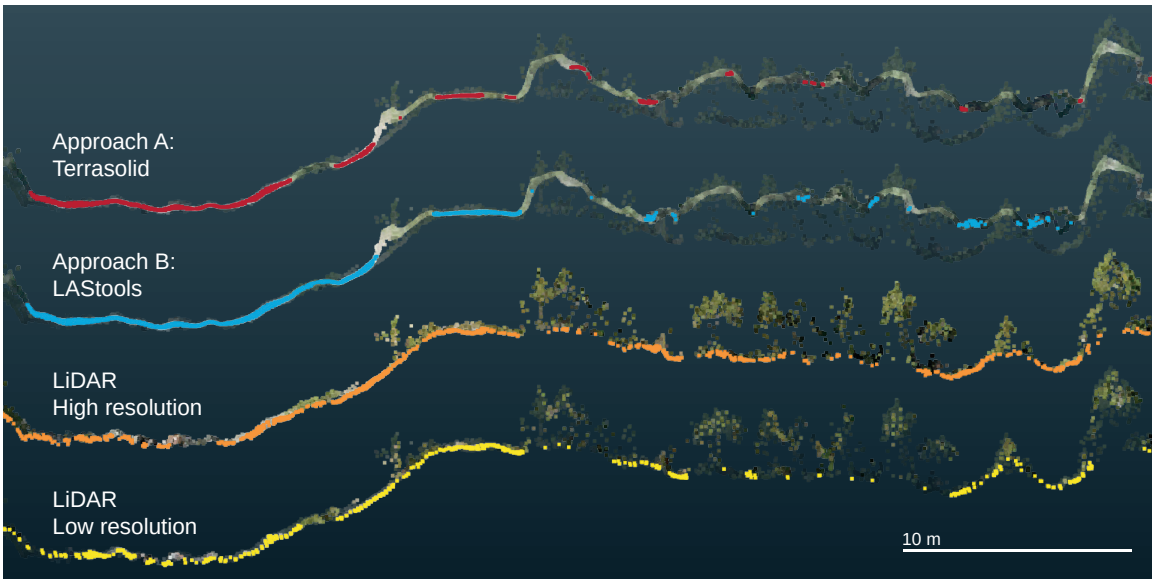


Figure 5.10. Cross-section of red line in Figure 5.9. Along densely vegetated shrub forest photogrammetric derived routines are strongly limited, classified ground points (color highlighting) tend to be clearly above the LiDAR-based ground points. The ground point density of the high resolution LiDAR along the densely vegetated areas is clearly superior compared to the low resolution survey. For better comparison, all ground points are plotted with the unclassified high-resolution LiDAR point cloud in the background.

The interquartile range of the deviation to Terrasolid and LAStools along shrub forests is between 100 and 170 cm. The whisker range is up to 260 cm. The median of the deviation from SwissALTI3D and SwissSURFACE3D is 50 cm. The median value of the LiDAR UAV is about the same as the SwissSURFACE3D. The data of the high-resolution version is lower. In the open forest, the differences in the quartile are between $\pm$20 cm deviation to the SwissSURFACE3D along all surfaces models. However, the median estimate of the Terrasolid is the closest to the SwissSURFACE3D. In the torrential area, the deviation from SwissSURFACE3D is marginal. The interquartile range of the UAV-derived data is $\pm$10 cm. The SwissALTI3D data have an interquartile range of 0 to 120 cm. This clearly shows the limitations of a DTM with 2 m resolution.

Another way to illustrate the differences between the approaches is to visualise the points as cross-sections (see Figure 5.10). The red line in Figure 5.9 indicates the profile from which the cross-sections are extracted. Ground points from four different data types are presented. Along the shrub forest the Terrasolid-based routine has a few very sparse ground points, all of them above the LiDAR ground points. The LAStools algorithm has a few more ground points. The ground points in the low-resolution LiDAR version are really on the ground, but are sparser than in the high-resolution version. Around the torrent, the photogrammetrically derived ground points are the densest. A significant proportion of those points is above the lowest points of the high-resolution LiDAR version. The few ground points of the Terrasolid routine however, are mostly close to the actual ground. The low-resolution LiDAR ground points are close to the real ground. They are not as dense as the high-resolution points, but still represent the terrain accurately.

We can conclude that in open terrain such as torrents and open forests, the quality of photogrammetric data is comparable to LiDAR UAV data. Along densely vegetated terrain the two photogrammetric

algorithms are severely limited. In shrub forest the classified ground is on average 1.4 m above the ground. In dense forests with tall trees, both routines are unable to detect ground points below the trees. However, for the use of torrent applications, photogrammetric data can be considered sufficient. If the torrent is surrounded by dense vegetation, a low resolution LiDAR flight may significantly improve the quality of the derived ground points.

5.3. Torrential Properties

We have developed an algorithm that extracts information on the characteristics of torrents based on a high-resolution DTM. Providing information on torrent geometry usually involves a lot of fieldwork. Assessment approaches such as Frick et al. (2011) suggest surveying torrent geometry in the field-based on homogeneous sections. Our newly developed algorithm calculates all geometry measures based on the high-resolution DTM (previously classified with Terrasolid) every 0.5 m. This results in a continuous dataset. We are able to extract further properties such as the slope of the embankment, the cross-section area or the standard deviation of the torrent bed heights. Such measures can be helpful in determining torrent sections. This division into homogeneous sections could be used for assessment methods such as Frick et al. (2011) and Gertsch (2009).

The automatic classification of the embankment with the OBIA software eCognition is limited, especially for vegetated areas near the torrent and steep adjacent slopes. To quantify these uncertainties, we ran the classification routine on automatically classified polygons from eCognition and on polygons we manually drew in ArcGIS (Esri, 2022). In general, it must be said that these results are only based on one subjective estimate. We do not know an absolute truth. We can only compare different estimates with each other. To obtain more robust data sets, the model would have to be tested on a large scale. As the generation of our data is not redundant it is not possible to make general assumptions. Nevertheless, we compare the results of the automatic OBIA-based approach and a workflow where polygons are drawn manually. In addition, we accessed the torrent geometry during a field survey. We divided the torrent into homogeneous sections and estimated a representative value for each geometry attribute. This is a very rough estimate with large uncertainties. We expect an uncertainty of 30% for all estimates based on metric measures. For the slope estimate, we added and subtracted 3° from the estimated field value. The subdivision was done during the field assessment and could have been done differently considering the newly obtained data. Boxplots are used to discuss the variability within the data along defined sections considering the estimated field, OBIA and GIS derived values.

We calculated standard geometry measures such as embankment height and torrent bed width. These measures are used in field-based methods such as Frick et al. (2011). Since we dispose of all geometry measures, we were also able to derive other geometry attributes such as the cross-section area, which provides information about the capacity of the torrent. The inclination of the torrent bed can be seen as an indication of the expected erosion capacity: The greater the slope, the higher the erosion may be. The results of the different torrent characteristics are presented below.

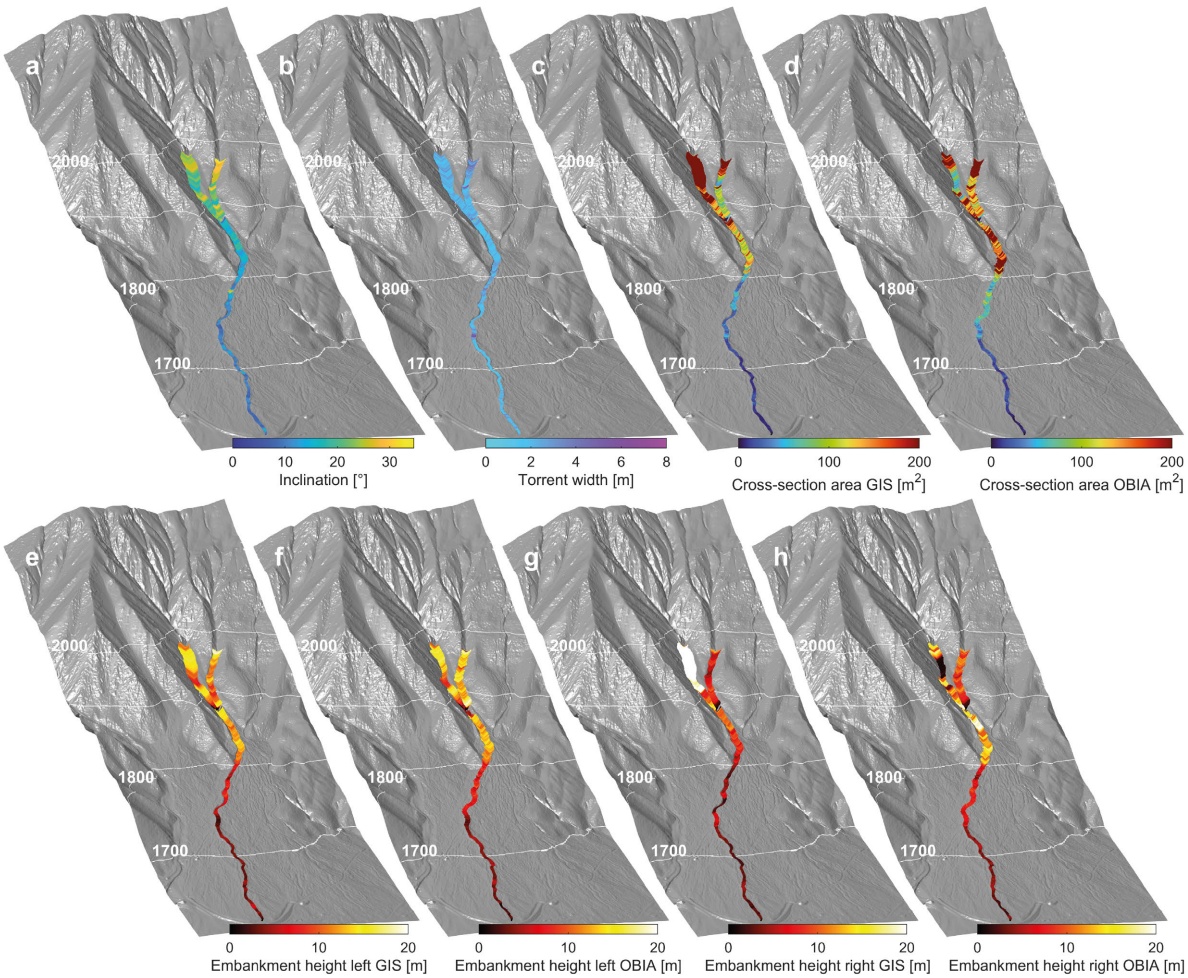


Figure 5.11. Extracted torrential properties of Arelen: a) Inclination torrent bed, b) Iorrent bed width, c) Cross-section area based on manually drawn Embankment polygon in GIS, d) Cross-section area based on automatic OBIA, e) Embankment height left based on manually drawn embankment polygon in GIS, f) Embankment height right based on automatic OBIA, g) Embankment height right based on manually drawn embankment polygon in GIS, h) Embankment height right based on automatic OBIA.

5.3.1. Torrent Bed Inclination

The slope of the torrent bed provides valuable information, such as dividing the catchment into homogeneous sections, the ability to trigger debris flows and the expected erosive capacity. The lower catchment below the fan apex of Davos Arelen has a slope inclination of 5 to 15°. The check dam at an altitude of 1780 m a.s.l. is clearly visible due to a concentrated steeper slope of 30°. Between the cone apex and the confluence, the slope ranges from 12 to 20°. The area above this point is inclined more steeply. With average inclinations of 30° and more. Most of the terrain is steep enough to trigger debris flows.

In Fraschmardin we could characterise the properties of the torrent only below 1780 m a.s.l. (see Figure 3.1 a). In this part of the torrent, the average slope is between 5 and 35°. The uppermost section is consistently about 30° steep. Debris flows can easily be triggered by sufficient hydrological discharge if enough debris is present. Between the confluence (1670 m a.s.l.) and the upper forest road (1560 m a.s.l.) the slope is between 15 and 30°. At the altitude of the forest road, we observe

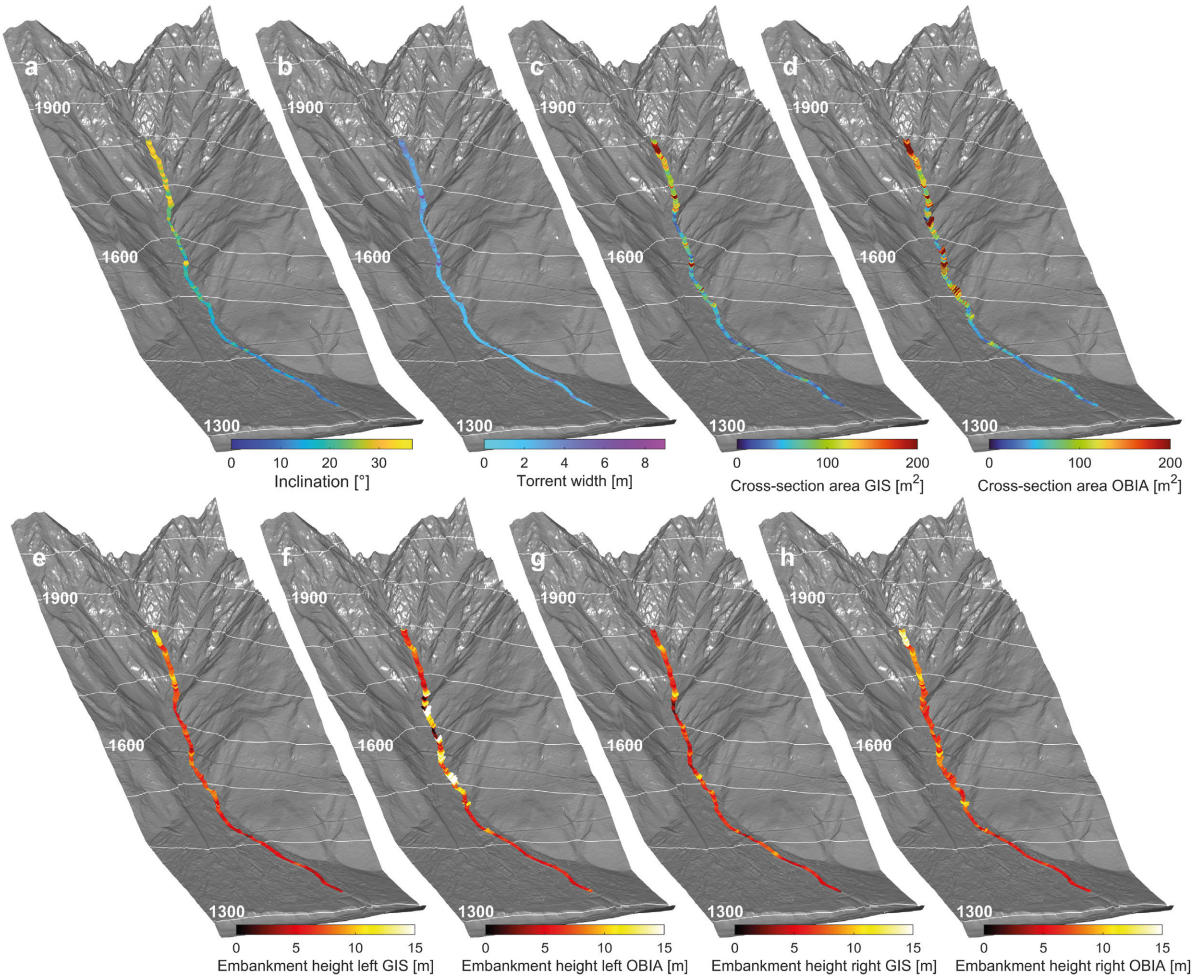


Figure 5.12. Extracted torrential properties of Fraschmardin: a) Inclination torrent bed, b) Torrent bed width, c) Cross-section area based on manually drawn Embankment polygon in GIS, d) Cross-section area based on automatic OBIA, e) Embankment height left based on manually drawn embankment polygon in GIS, f) Embankment height right based on automatic OBIA, g) Embankment height right based on manually drawn embankment polygon in GIS, h) Embankment height right based on automatic OBIA.

a steep and abrupt incision with a slope of more than 35°. Below this point, up to an altitude of 1460 m a.s.l., the slope varies between 15 and 25°. This is still a range of steepness in which debris flows can be triggered, especially if smaller landslides or wood lead to a jam. Below this point, the slope is between 5 and 15°, except at the lower crossing of the forest road (1410 m a.s.l.), where the slope is more than 20°, probably due to a man-made obstruction near the bridge (see Figure 5.12). In the field we selected a representative cross-section for each section, on which we measured the slope 10 m upslope and downslope. In Figure 5.13 we show the average value of the two measured slopes and add an uncertainty range of $\pm 3°$. The field-based measurements are in the same range as the automatically extracted measurement. The slope changes gradually over the torrent sections. The slope tends to increase in the upper catchment.

5.3.2. Torrent Bed Width

We define the width of the torrent bed as the horizontal distance between the two lower ends of the embankment. Large boulders and collapsed embankments can have a large influence on the resulting width. In Arelen (see Figure 5.14) the calculations of the torrent bed width varies by 1 m. It is not possible to define a clear trend along the sections. Estimates from the field range between 1.5 and 3 m. In the catchment area of the Fraschmardin (see Figure 5.13) the calculated torrent width is somewhat larger than in Arelen. The measurements derived from the DTM show a slightly larger torrent bed width in the lower section than in the middle section. In the upper sections, a trend towards a wider torrent bed can again be observed. This can be explained by the fact that in the lower section the torrent bed was artificially widened to increase capacity. In the upper part, the torrent tends to be wider and less incised. The field estimates range from 2 to 4 m, the calculated widths from 2 to 3 m. We assume that these values are in a similar range.

5.3.3. Embankment Height

To estimate the area exposed to torrent erosion, we are interested in the actual diagonal distance of the embankment. To do this, we extract the height difference and the vertically measured width. From these two measures we derive the distance between the lower and upper edge of the embankment. As described in section 4.4, we aimed to extract the embankment geometry fully automatic using the OBIA software eCognition and an ArcGIS-based Python script. This automated approach, which relies on OBIA, is severely limited, especially in the upper catchment, where the embankment gradually transits into the adjacent hillslopes. It is difficult to clearly delineate this boundary. Further, we encountered difficulties in regard to vegetation near the torrent, where the photogrammetrically derived DTM tends to be highly interpolated.

In Davos Arelen below the current fan apex at an elevation of 1740 m a.s.l. (sections A-C), the embankment heights of both sides are equivalent and consistent across the GIS, OBIA and field-based method. In sections D and E up to the confluence, we observe a difference between the OBIA, GIS and field-based methods on the orographic right side. The automatically derived OBIA tends to classify the embankment over the "torrential extent". On the left side, the values are consistent.

In the uppermost section the OBIA-derived embankment height on the orographic right side above the confluence deviates strongly from the manually drawn polygon. Figures 4.7 and 5.11 show that the GIS estimate of the embankment is too large. The OBIA approach has many values that are too low in a certain part of the section, with estimates close to the torrent bed. Nevertheless, the OBIA method results in a median value that is close to the field observation. It should be noted that it is also difficult to clearly delineate the embankment in the field. The uppermost section on the orographic right side shows very consistent results.

In the Fraschmardin catchment, embankment measurements below 1460 m a.s.l. (sections A-C) are consistent for all three methods. Above this altitude, the automatic OBIA method tends to classify too large areas as embankments (see Figure 5.11 h and f). This can be observed especially in densely vegetated or steep and rocky terrain. In sections D to G we observe extremely large uncertainties

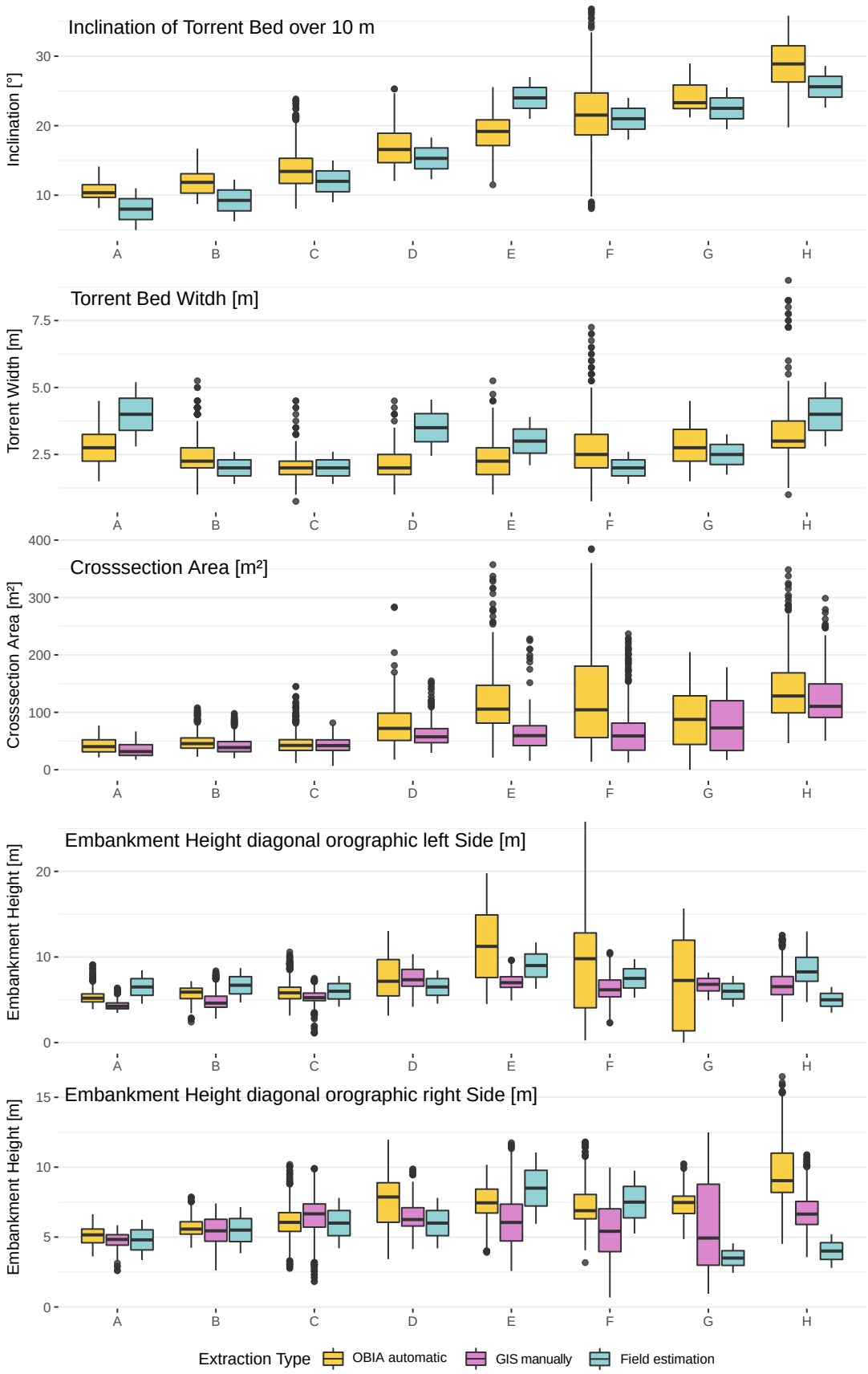


Figure 5.13. Comparison of the torrent parameters of the Fraschmardin catchment. OBIA and manually extracted GIS values displayed according to their distribution, field measurements with their survey uncertainties, respectively. The resulting values of torrent bed width and inclination are consistent along the field- and GIS-based extraction. The OBIA-based embankment measurements show a particularly large variation along densely vegetated and landslide-prone areas.

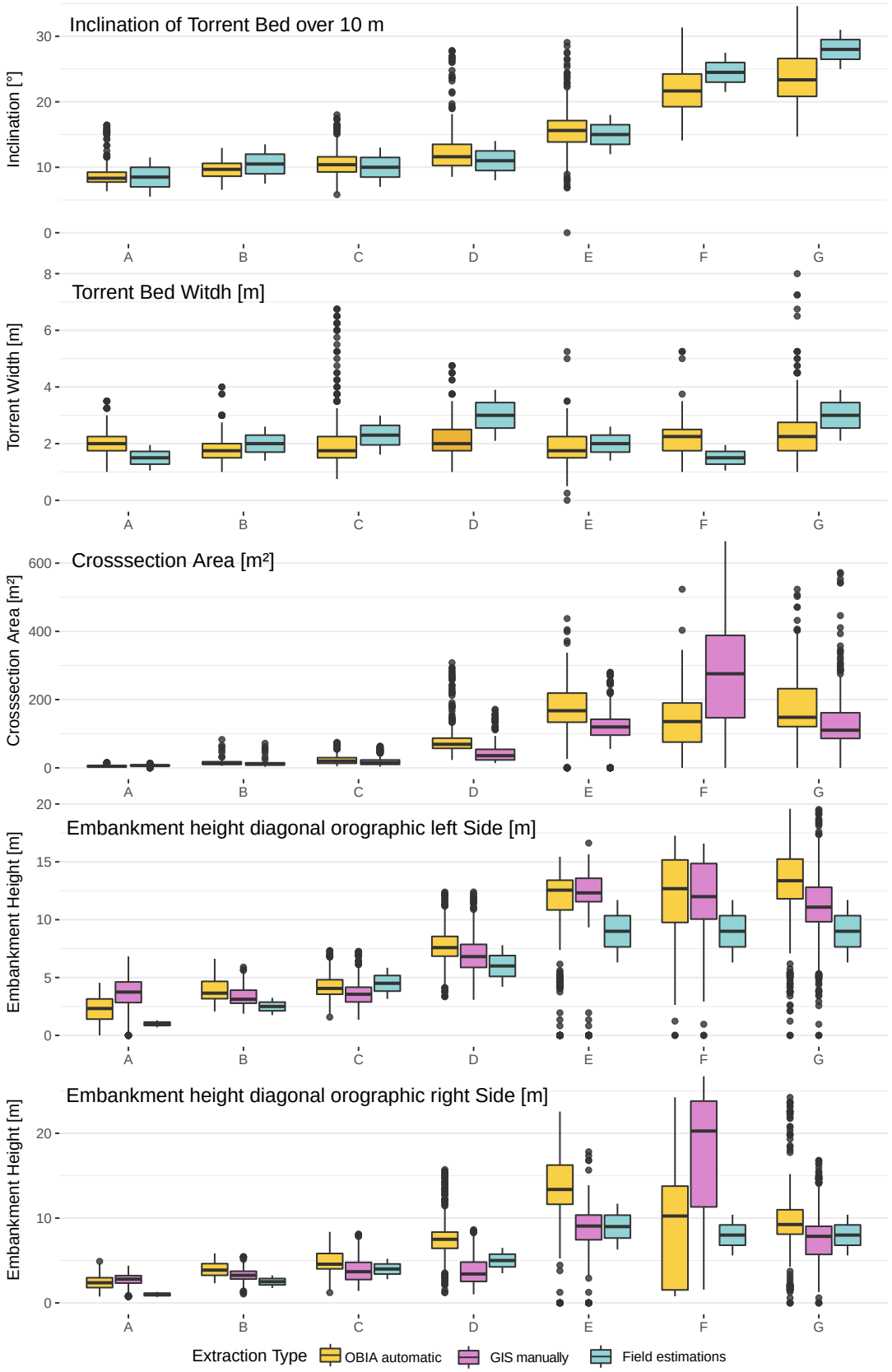


Figure 5.14. Comparison of torrent characteristics of the Arelen catchment. The resulting values of torrent bed width and inclination are consistent along the field- and GIS-based extraction. Along the old debris-flow cone (sections A-D), the embankments extracted automatically with OBIA and manually with GIS and the derived crosssectional area are largely consistent. One of the uppermost sections F on the embankment of the orographic right side has large discrepancies.

on the left side of the embankment. This is probably due to landslides and forested areas. On the orographic right side, the OBIA embankment height is less scattered. An exception are section D and the uppermost section H. Section D is along densely vegetated terrain. And in the uppermost section H, the automatic OBIA approach has classified the adjacent slope as embankment. This results in the embankment height being several metres too high. In the field we estimated an embankment height on the right side of 4 m. The OBIA method classifies an embankment height more than twice as high. Such clearly miss-classified embankments have a strong influence on the resulting cross-section area. In the other sections, the extracted values generally coincide with the field estimates. For some sections, such as Section E, the field-based estimate is probably overestimated. Since the measurement was carried out in a part of the torrent where the dimensions are above average.

5.3.4. Cross-section Area

The cross-section area results from the embankment heights and the width of the torrent. It provides a measure of the capacity of the torrent. Errors caused by inaccurate classification of the torrent embankments have a major influence on the resulting cross-section area. The cross-section area is defined as the trapezoidal area. The height is the average of the two embankment heights and the width is the average of the torrent bed width and the distance between the two top edges of the embankment. It is important to note that it should not be understood as the wetted surface during an event. Rather, it can be understood as a measure of the holding capacity.

In Arelen the cross-section area becomes larger in the upper catchment. Sections A-C show very similar cross-section areas comparing the manually drawn polygon and the OBIA-based classification. For sections below the actual fan apex, the differences between the methods are marginal. Section D represents the section on the old debris flow cone, where the torrent is still strongly incised and the delineation of the embankment is easily possible. Nevertheless, the vegetation leads to a larger cross-section area based on the OBIA method compared to the GIS-based approach. In section E we observe a similar effect, but more pronounced and with greater variability. The torrent section on the orographic right side above the confluence we see observe large deviations between the GIS and OBIA approaches. This section is well visible in the Figure 4.7. The classification with OBIA is very variable. In some areas there is barely any embankment classified, in other areas the algorithm worked well. The values of the GIS-based method are much higher. It is likely that we have drawn the polygon too high. However, this is a good example to also show the difficulties of drawing a polygon manually. In section G, the results of the two methods are consistent.

At Fraschmardin, the results of the GIS-based approach show consistent results. In the uppermost part above the small confluence, the resulting cross-section values vary between 70 and more than 200 m^2. The variability is very high in this area. In such steep terrain, a too large embankment polygon has an immense influence on the resulting calculated area, especially if the adjacent slopes are steep. Between the small confluence (1670 m a.s.l.) and the upper forest road, the cross-section area ranges from 20 to 150 m^2. The low values come from the narrows where the stream runs on bedrock and the orographic left side is bedrock and the right side is consolidated with large boulders. The OBIA-based calculation is more variable in this area. Just at the upper forest road,

the cross-section area increases drastically with values well above 200 m^2. Further downslope the cross-section area varies between 60 and 150 m^2. At some specific transects the values are well above or below this range, which are likely to be erroneous outliers. Below an altitude of 1470 m a.s.l. the cross-section area is consistently below 80 m^2. Looking at the results of the OBIA approach, we can visually see that the values are less consistent. Erroneous transects with extremely high cross-section values are more frequently, especially in forested and steep rocky areas. The boxplots in Figure 5.13 describe the same pattern. In the lowest three sections (A-C), the cross-section areas resulting from the GIS and OBIA approaches are almost equivalent. In sections D - F (1460 - 1670 m a.s.l.) the differences between the two approaches are most pronounced. This is due to densely vegetated areas and landslides, which make it difficult to delineate the embankment area. In the uppermost section above 1670 m a.s.l., the derived values of both GIS and OBIA are probably overestimated. In this section the torrent is very wide and the adjacent slopes rise steeply. A tiny misclassification has an major impact on the resulting cross-section area.

5.4. Change Detection with M3C2

We have applied the M3C2 method on different data sets. In the Arelen catchment we are comparing the situation between 2020 and 2021. Some minor landslides and gully erosion processes in the upper catchment can be quantified. In the Fraschmardin catchment we are comparing the UAV data sets from 2018 and 2019. In the year 2019 a major debris flow occurred and we were able to quantify the erosion and deposition behaviour. Here we compare three different difference calculation approaches with each other. In summer 2021 we had the occasion to be able to survey the Carrerabach catchment just before and after a debris flow event.

5.4.1. Case Study Arelen: Minor Landslides and Gully Erosion

In the catchment Davos Arelen we compared two Wingtra UAV flights from the year 2020 and 2021. The dense point clouds were classified following the classification routine of Terrasolid. Based on the derived DTMs we were able to detect a systematic error. The error was modelled on the basis of tie points resulting in a modelled error ranging $\pm$20 cm. Once the error model was applied and the two points clouds were co-registered with ICP, we calculated the M3C2 with 50cm radius and 5cm point density. In Figure 5.15 also a M3C2 of the non corrected point cloud is presented. The blue areas at the lower and upper end and the red area in the middle part indicate clearly the overall systematic error. The distance uncertainty, which derives from the tie points of both flight missions varried between 0 and 50 cm. Along the vegetated areas and steep terrain we observed higher uncertainties. The shallower parts at the lower and upper end of the catchment specify low uncertainty.

Evaluating the M3C2 distance we still observed a lot of noise along the vegetated areas. A minor failure of the embankment represents one of the greatest differences between 2020 and 2021. Further some erosion processes in the upper catchment could be quantified. In the lower area we see a red/blue line crossing the study extent. This is a construction of new water pipe, rather than a blunder error. Mass wasting processes with larger magnitude of more than 15 cm could be visualised

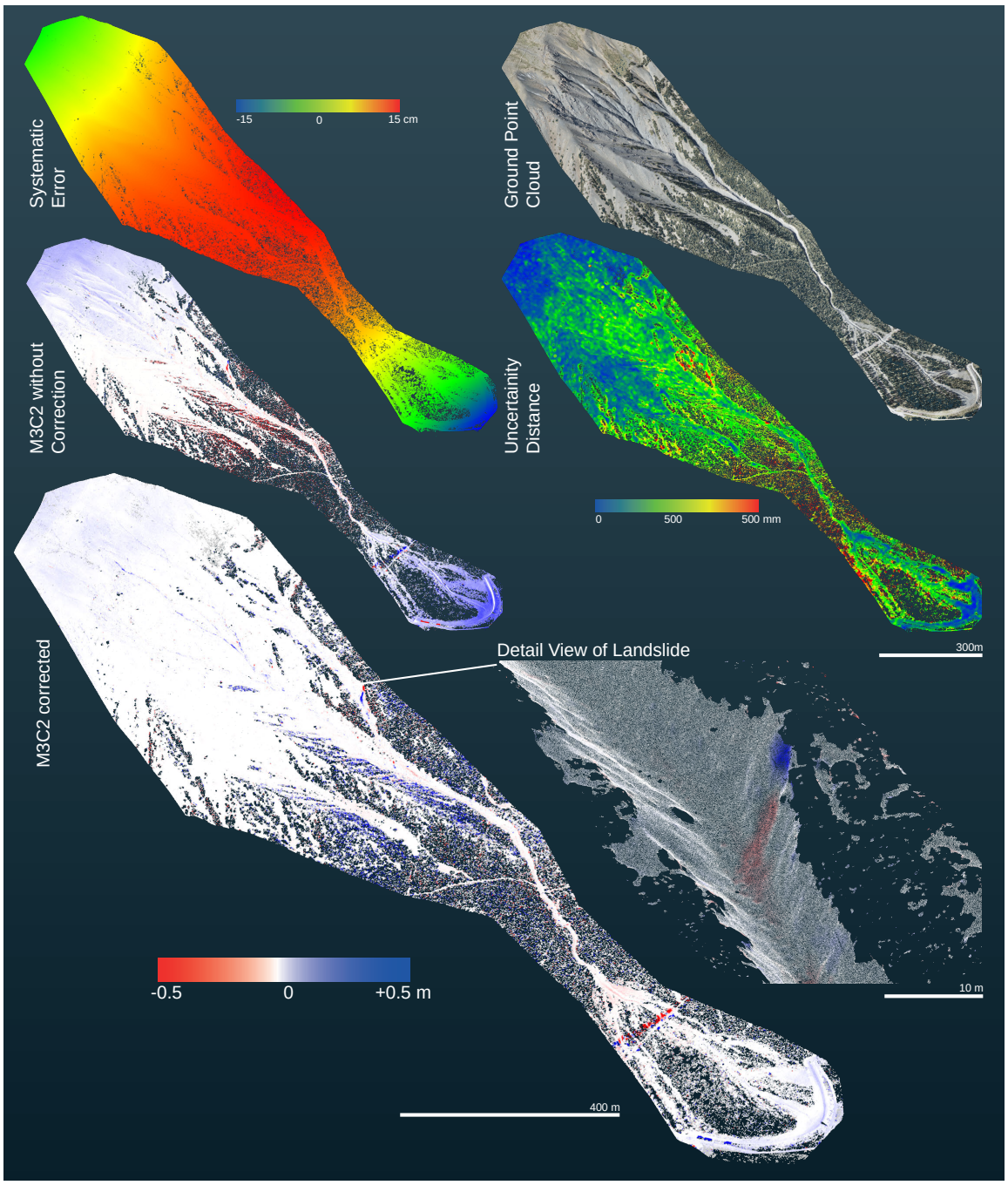


Figure 5.15. Point cloud based visualisation of the Arelen catchment. The various point clouds indicate calculation steps, which were applied to derive a corrected cloud based M3C2 difference.

and quantified in this example. With the application of the error model we were able to push the limits of change detection. The treatment of the point cloud with the error model allows to bend the point cloud. With the subsequent ICP the be aligned point cloud is turned and translated until it matches the reference point cloud.

5.4.2. Case Study Fraschmardin: Debris Flow 2019

In summer 2019 we were able to capture a debris flow event, as we have flown the middle to lower part of the catchment already in 2018. On Figure 5.16 a we present the standard vertical difference of the two DSMs. Along the vegetated areas the data is very noisy. To extract an accurate measure of erosion and deposition the DSM would need to be cut precisely to the extent of the torrent. On areas where vegetation was removed during the event, the measure is imprecise as well. Evaluating the erosion magnitude along the torrent we observe especially large values on the side at the embankments. This effect is due to the steepness of the terrain. To provide a brief example: For an embankment with an average inclination of 60°, we observe 40 cm erosion (measured vertically to the torrential surface). This would lead to a derived DoD difference of 80 cm. As the DoD is calculated on raster basis and therefore always orthogonal, the resulted heights are overestimated strongly along steep embankments. Normally the erosion is highest in depth and less pronounced to the side. In Figure 5.16 b the M3C2 difference from the LAStools-based algorithm derived classified ground point cloud is presented. There is almost no vegetation noise. The derived erosion values seem to be realistic. Nevertheless, the application of the M3C2 algorithm to mountain torrents has some limitations. Occlusion due to vegetation causing holes in the data and subsequently no values can be derived. The difference can only be computed within areas that were classified as ground in both surveys.

The third computation is based on cloud to mesh (C2M) difference calculation (see Figure 5.16 c). From the classified ground point cloud of the year 2018 we have generated a mesh and calculated the difference from the mesh to the classified ground point cloud 2019. The mesh generation has the advantage of creating a continuous surface along the point cloud. Difficulties like holes due to vegetation before the event can be solved following this approach. Along densely vegetated parts some noise is still remaining. The reason why it is noisier than the M3C2 difference can be explained by the continuous surface from the reference flight mission. Every noise point from the second flight leads to a distance calculation. In contrast to the M3C2, where the projected cylinder area has to match points in the second point cloud. The routine takes the closest point for the difference calculation. In areas where the surface has changed its orientation significantly a difference can be still calculated as long as the second point cloud provides some points to compare with. That can be seen nicely in the close-up view: In embankments prone to smaller landslides and along depositions on areas, which were vegetated before, it is still possible to derive a distance. This implies the important question, if those distances represent the reality. The mesh surface is an interpolation of the reference point cloud. It comes with the advantage that a distance can still be derived, also when the reference point cloud is partly hidden due to occlusion. The interpolated mesh surface is not an accurate representation of the real ground. Values derived from the interpolated part can not be taken as an accurate measure. The C2M algorithm calculates the shortest distance between each point and the mesh surface. The distance is calculated in a 3D space. In this manner embankments are not overestimated as strongly as with the DoD approach, but as it calculates the shortest distance to the mesh, which may deviate from the normal direction of the surface. Further, the mesh may be interpolated. The calculated distance is likely to differentiate from reality.

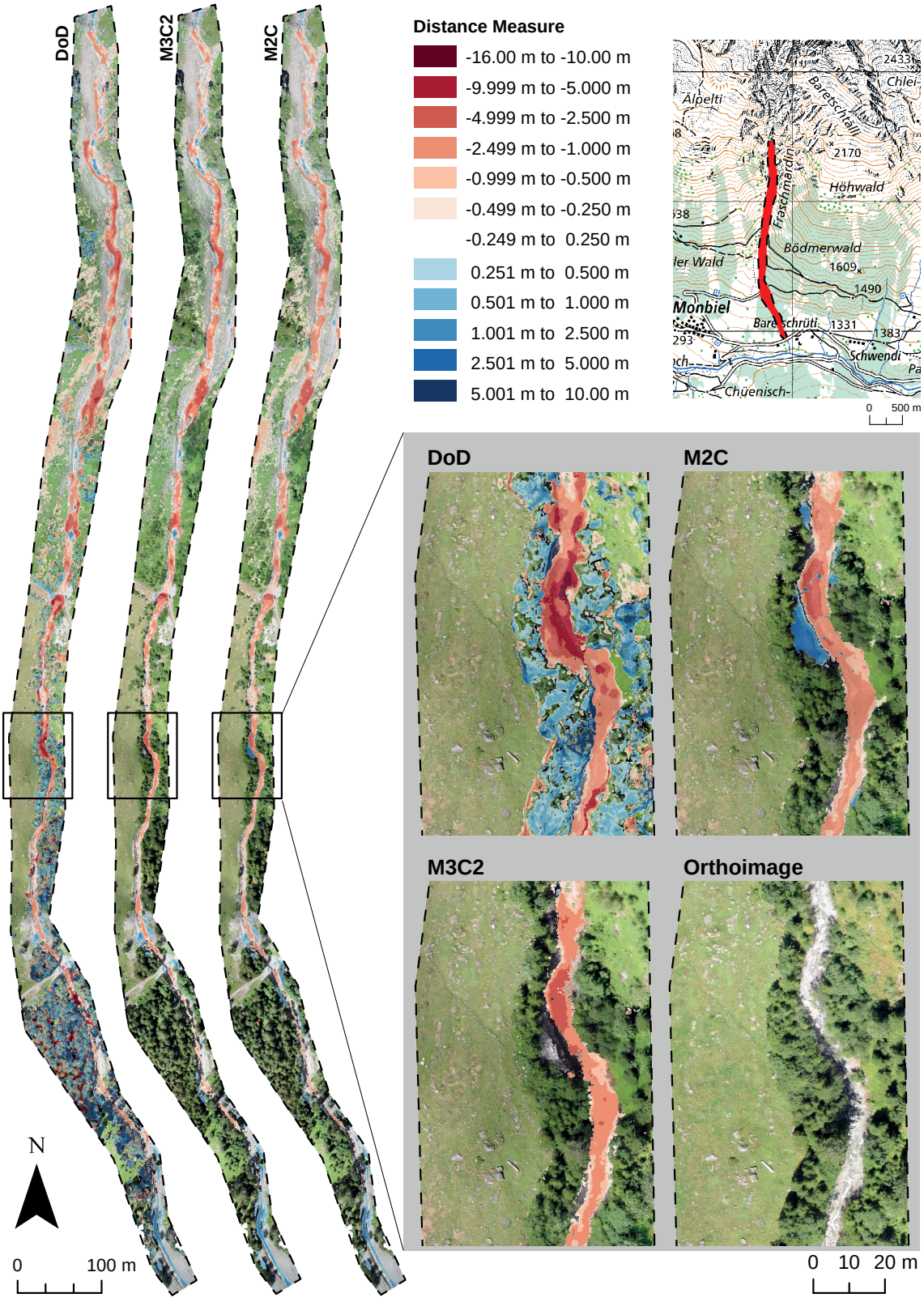


Figure 5.16. Calculated difference of before and after the debris flow in summer 2019 based on DoD, C2M and, M3C2 methods. The M3C2 method represents change most accurately, however the method is strongly limited by occlusion due to vegetation and miss classified ground surface.

In the close-up view we see a left turn which presents a fully different result along all three difference calculation methods. Comparing the two orthoimages. we see that there was a lot of forest before the event and now there is a pile of debris which was deposited during the event. Based on the DoD we interpret the whole area as erosion, explained by the difference between the top of the trees and the deposited pile of debris. Regarding the M3C2 there was no distance calculation executed in the area. The algorithm is only able to derive a distance when the cylinder matches points in the second point cloud. In this case, there are no points that can be compared to the reference point cloud. Overall, we can conclude that the M3C2 method provides the most accurate and precise measure. It is limited by occlusion and strongly changing surface orientation. Those distances which could be derived are correct, except to some noise along low vegetation, which can be neglected.

Based on the M3C2 distance (see Figure 5.17) calculation we found 19'700 m^3 deposition and 40'400 m^3 erosion along the study extent. The proportion of deposition is mainly concentrated on lower and upper part of the studied area.

5.4.3. Case Study Carrerabach: Debris Flow 2021

We flew only a small part of the catchment at Carrerabach (see Figure 5.17). The whole catchment is more than 20 km^2 large and the upper catchment is strongly incised. Photogrammetric flight mission are therefore strongly limited. With the DJI Phantom 4 RTK we surveyed the area "in da Bendera" on the 15.08.2021. On the 17.08.2021 a debris flow occurred and we were able to fly the same extent again one day later on the 18.08.2021. As the terrain is strongly incised, systematic errors are minor and could be neglected. The point cloud is classified with the Terrasolid-based algorithm and co-registred with the ICP routine. Distance uncertainty is derived by error estimates of the tie-point cloud. Based on the co-registered and classified ground point cloud we calculated the M3C2 distance using a projection area of 0.5 m^2 and a sampling size of 5 cm.

Along this torrent section of 500 m, we measured an erosion volume of 4'400 m^3 and a minor deposition volume of 300 m^3. In this small extent we have been able to observe various processes. The major volumes originate from landslides, which were most likely destabilized during a debris flow event. In the run-out area of the landslide there is no deposition. This brings us to the assumption that multiple surges of the debris flow followed subsequently. Gully erosion processes also contributed significantly to the erosion volume. And lastly depth erosion eroded a great amount of material. Along the torrent bed we observe a repetitive fill up of the bed in between the debris flow events. Further depth erosion is limited due to the bedrock. This section is likely one of the most active along the torrent.

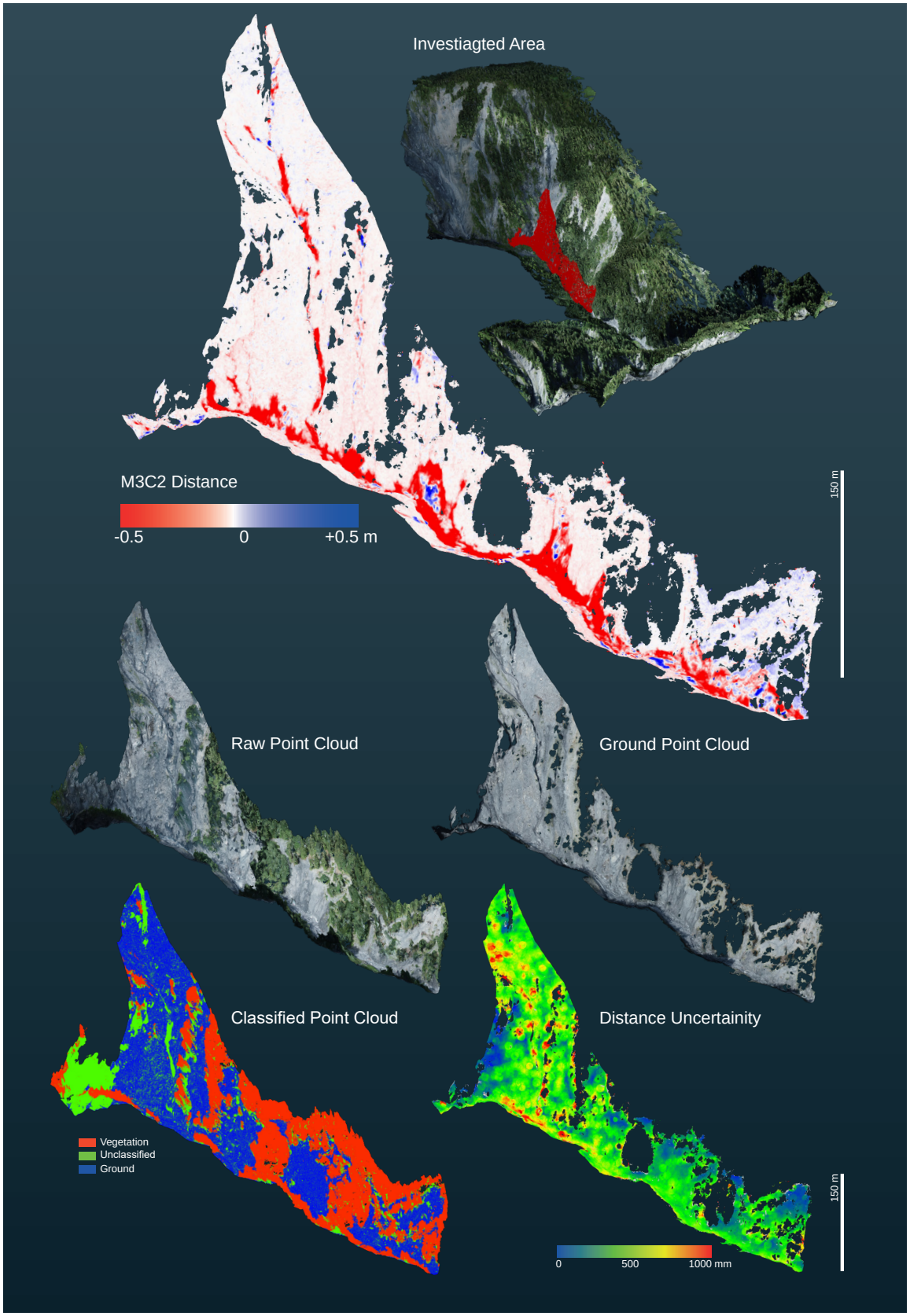


Figure 5.17. At 17.08.2021 a debris flow in the Carrerabach catchment occurred. A M3C2 distance is derived from the point clouds classfied with Terrasolid-based routine.

Discussion

In the following chapter we discuss how the elaborated processing of UAV data can be used to improve debris flow hazard assessment. Firstly, we review the data acquisition process. We discuss limitations and advantages of the different point classification algorithms. Secondly, we propose an hazard assessment workflow to exploit UAV measurement data. Limitations of the automated extraction of torrential properties are critically discussed and ongoing limitations for a reproducible estimation of the entrainment are specified. Automated UAV-based methods facilitate a more objective hazard assessment; nonetheless fieldwork will remain indispensable. Fieldwork, however, can be concentrated at specific critical locations in the torrent. The ability to fly with a UAV to the area of interest saves time and effort. Especially if we relate the costs and emissions emitted by a helicopter flight. In Table 6.1 we summarize the advantages and the disadvantages that are associated with UAV-based torrent assessment.

Table 6.1. Advantages and disadvantages of UAV-based assessment of torrents.

	Pro	**Contra**
Costs	Cheaper than helicopter inspections	High set up costs (software, computer, UAV, training)
Time demand	Fieldwork can be adressed more specifically	Additional workload (UAV flight 2-3h; processing 1h)
Planning	Can be combined with fieldwork	Stable weather, no snow, little leaves as possible
Reproducibility	Clear data basis available	Same classification algorithms need to be applied
LAStools Routine	Accurate on rough surfaces and grassland	Need of accurate coarse reference height model
Terrasolid Routine	No need of reference height model	Erroneous classification of grassland and surface smoothing
LiDAR UAV	Produces accurate DTM in dense vegetation	High costs (100'000 CHF), only small areas
WingtraOne	Captures large areas	High costs (30'000 CHF), limited by vegetation
DJI Phantom 4 RTK	Flies along the terrain, not too expensive	Limited by vegetation, only small areas
DJI Mavic Air2	Cheap, investigation of inaccessible terrain	Not applicable for photogrammetry
Torrent Analysis	Overview, objective segmentation of sections	Limited to clearly defined torrent beds
Impact Analysis	Less field measurements are needed	Limited near bridge culvatures and dikes
Simulation	High spatial and temporal resolution	Additional workload, inconsistent properties of DTM
Erosion	Contributing area can be derived automatically	Depth is not possible to derive automatically
Change Quantification	Sophisticated distance measure with M3C2	Limited by occlusion due to vegetation

6.1. Data Acquisition

We tested three different UAVs to generate DTMs in steep alpine terrain. The WingtraOne uses Tailsitter VTOL technology. It is capable of flying over large areas. Other photogrammetric UAVs such as DJI Phantom 4 RTK, are limited in their aerial extent. In steeply incised terrain, however, the DJI Phantom 4 RTK has several advantages. The quadrocopter UAV is able to turn within a small area and follows the terrain more precisely. In comparison, the WingtraOne does not change its flight altitude along a flightline. It always takes the highest point of each flight line as the flying altitude. The resulting ground measurement accuracy is comparable because the WingtraOne carries a high quality camera with a minor lens distortion.

Generally the higher the UAV flies the lower the ground sampling distance (GSD) becomes. In forested areas it is advisable not to fly too low, due to the tilt effect of large trees. With high flight altitude (>200 m) we did not observe this tilt effect of the trees so distinctly. We have only worked with pictures taken in nadir perspective. Especially in forested areas oblique pictures have the advantage that there are not as many disturbing shadow effects (Díaz et al., 2020). In areas with deciduous trees the UAV flights should be organized for leaf free seasons. Further improvements may be achieved with the application of different flight altitudes and circular flight patterns (Roncoroni et al., in review)

Snow cover is another problem. In late winter and early spring it is very likely that there is still some snow in the torrent. Especially if the catchment is prone to avalanche activity. Dense avalanche deposits and compacted snow layers melt slowly and therefore hinder UAV flights to find torrent geometries. By the time this snow has melted, it is very likely that deciduous trees have already sprouted their leaves. In autumn the timing can be equally challenging. Often, by the time of the first snow fall trees have not yet shed their leaves. On southern slopes the snow often remelts, allowing UAV flights with less pronounced vegetation effects. On less sunnier slopes, the snow often remains after the first snowfall. In lower altitude pre-alpine areas fog limits the use of UAVs. Weather conditions such as strong winds must always kept in mind. Furthermore, as the trees tend to move in the wind accurate reference points in the crown are often missing. In windy conditions UAVs tend to fly too slow or too fast. Further, they consume a lot more energy as they have to fly against the wind. Data acquisition using photogrammetric UAVs in debris flow prone catchments is demanding for various reasons. Flights need to be carried out in a time window without snow, little wind, no fog and as little green leaves as possible. Once the UAV flight is executed, dense point clouds are calculated. In a subsequent step, the point clouds are classified into ground and vegetation points.

We discovered large differences between LiDAR and photogrammetric UAVs when producing ground surface DTMs. LiDAR sensors are able to penetrate dense vegetation, but are more expensive (recall that UAVs can be lost and never recovered in critical weather situations). However, photogrammetric-based point clouds are limited to the vegetation surface. Therefore, if the area of interest is vegetated LiDAR-based workflows provide a more accurate ground point cloud with less post-processing. In catchments where vegetation plays a minor role, photogrammetric-based UAV flights are more efficient, time-wise and with regards to the overall costs. As forests often cover only parts of the torrent, it may be advisable to make use of LiDAR techniques to assess the vegetated torrent areas and use a photogrammetric technique in the remaining area.

6.2. Point Cloud Classification

We briefly introduced various approaches to classify point cloud data. The results based on photogrammetric point cloud data are often not satisfying, especially in steep terrain and densely vegetated areas. Routines only based on either TIN densification or native filtering do not provide reliable results in alpine terrain. These algorithms have difficulties to distinguish steep and rough hillslopes from shrub forest such as green alder (*Alnus viridis*) or the shrub form of mountain pine (*Pinus*

mugo). As stated by Anders et al. (2019) the combination of RGB-based classification with other algorithms, which were developed for LiDAR data, provide the most reliable results. Based on RGB values shrubs can be correctly distinguished from rocks. The use of a NIR-band based camera is likely able to further facilitate the classification of vegetation with photogrammetric classification routines. de Haas et al. (2020) has also used the LAStools software to classify vegetation along the debris flow channel at Illgraben (VS). They only aimed to remove low noise and overhanging vegetation. Low noise points were filtered with more than 0.1 m below a smoothed 20th height perecentile surface with a *step size* of 0.5m. Afterwards, a ground classification with *ultra fine* settings was applied. This procedure allows the removal of over-hanging vegetation along the torrent. We have tested this approach along the Arelen catchment. Large, dense vegetation patterns will remain and highly steep terrain tends to be classified as vegetation using these settings. We can conclude, that this approach works for spare vegetated areas along more or less flat terrain. To classify a whole alpine watershed with forested parts the application of this routine would not be effective. However, our developed LAStools approach may be improved with a better noise removal.

Following we are discussing the advantages and disadvantages of the Terrasolid and LAStools-based approaches developed for the purpose of this thesis. The two applied photogrammetric routines have different workflows. Terrasolid-based approach A aims to generate a smooth terrain surface and applies the classification algorithm on this smoothed surface. The Terrasolid routine smooths the surface strongly and aggregates the points along the smoothed surface. Like this the TIN densification to classify ground points becomes more robust. With the LAStools algorithm (approach B) the goal is pursued to keep all potential ground points also below the uppermost vegetation layer. This noisy surface is more challenging for TIN densification algorithms. A lot of noise remains after the first TIN densification. Especially along dense vegetation patterns non-green points remain in the shadow of trees. A second TIN densification routine, was applied on a coarse surface to generate a smoothed surface representing the real ground accurately, subsequently all points above 1 m of this coarse surface are deleted. With this two-folded routine it was possible to preserve the highly dense photogrammetric surface.

Along both routines a RGB vegetation index is used to address green areas, which are likely to be vegetation. The main difference between the two approaches can be found in the smoothing of the surface. The non-smoothed LAStools approach represents the surface in more details, however the classification of the ground is therefore more limited. Two subsequent ground classifications are needed to properly remove vegetation. This difference of the two routines can be well observed along open forests. By evaluating the final quality of the DTMs we observe a higher point density of the LAStools-based ground point clouds along open areas such as torrents and open forests. Approach A has the major advantage that it does not need a reference height model to differentiate between low and high vegetation. However, the routine is therefore not very robust in differentiating grass from high vegetation. In steep and highly rough terrain such as mountain torrents, the Terrasolid routine tends to remove sharp edges of coarse blocks more drastically. Very steep and high artificial constructions such es check dams are cut within both routines.

Both elaborated approaches are based on various processing steps. This structure holds the advantage that the algorithm can be adjusted finely. If needed there are many parameters to fine optimize. However, this incorporates a entire understanding of the behaviour of such complex algorithms. Simple algorithms such as Zhang et al. (2016) can be applied easily, but as soon as the terrain becomes more complex the results are often not applicable. Both routines have their advantages and disadvantages. The LAStools routine needs further improvement with the handling of noise. With this improvement, the second ground classification could be probably defined less aggressive. The major limitation for the Terrasolid routine is the missclassification of grass as high vegetation. To improve those results an additional parameter, which is defined by the roughness and confidence of green points, could be used to differentiate between grass and higher vegetation. However, the combination of both methods may lead to the most robust results for some applications. For different sites both routines need to be tested and likely some parameters might need adjustment.

6.3. Hazard Assessment with support of UAV derivatives

Hazard analyses vary widely from country to country (Jakob, 2005). In Switzerland there does not exists a clear general guideline (BAFU, 1997). In general, the swiss hazard assessment is based on three pillars: i) Field investigations and analysis of geodata ii) Study of past events (eg. event cadastre) iii) physical and mathematical modeling (Petracheck and Kienholz, 2003). We propose an hazard assessment approach to exploit UAV measurement data. While a holistic debris flow hazard assessment method is absent to date in the scientific literature, main principles thereof are elaborated in various contributions (e.g. Allen et al., 2022; Fuchs et al., 2008; Hungr et al., 1984; Hürlimann et al., 2008; Jakob, 2005; Petracheck and Kienholz, 2003; Rimböck et al., 2013) aiming to outline debris flow hazard assessment procedures. We are aware, that in actual engineering practice the assessment of debris flow hazards might often diverge. Our discussion is based on our UAV analysis of the two catchments Arelen and Fraschmardin, where we examined only the lower part of the catchment. However, we work under the assumption that UAV-based data is available and that it can be used for the assessment of the whole catchment. We investigate cost and time savings compared to conventional assessment approaches. The initial costs are approximately 30'000 CHF. This is required to purchase a UAV such as the WingtraOne. Further the software packages to process the data such as Agisoft, Terrasolid and LAStools cost an additional 5'000 CHF. Some software products must be purchased annually. We did not account for training costs to learn how to pilot the UAV, rather the costs for a sufficient powerful computer. We expect that those initial costs can be written off in a year, assuming the UAV is used frequently and that helicopter flights can be discontinued. We further expect that the application of UAV data will allow to address fieldwork more specifically. This saves additional money and time.

With the automated extraction of the torrential properties, we were able to calculate various derivations that may support hazard assessment in future. The hazard assessment process is complex, the data derived from UAV flights can be used in different processing steps of the hazard assessment. Subsequently, we go trough the whole hazard assessment procedure. We differentiate between torrent and runout analysis.

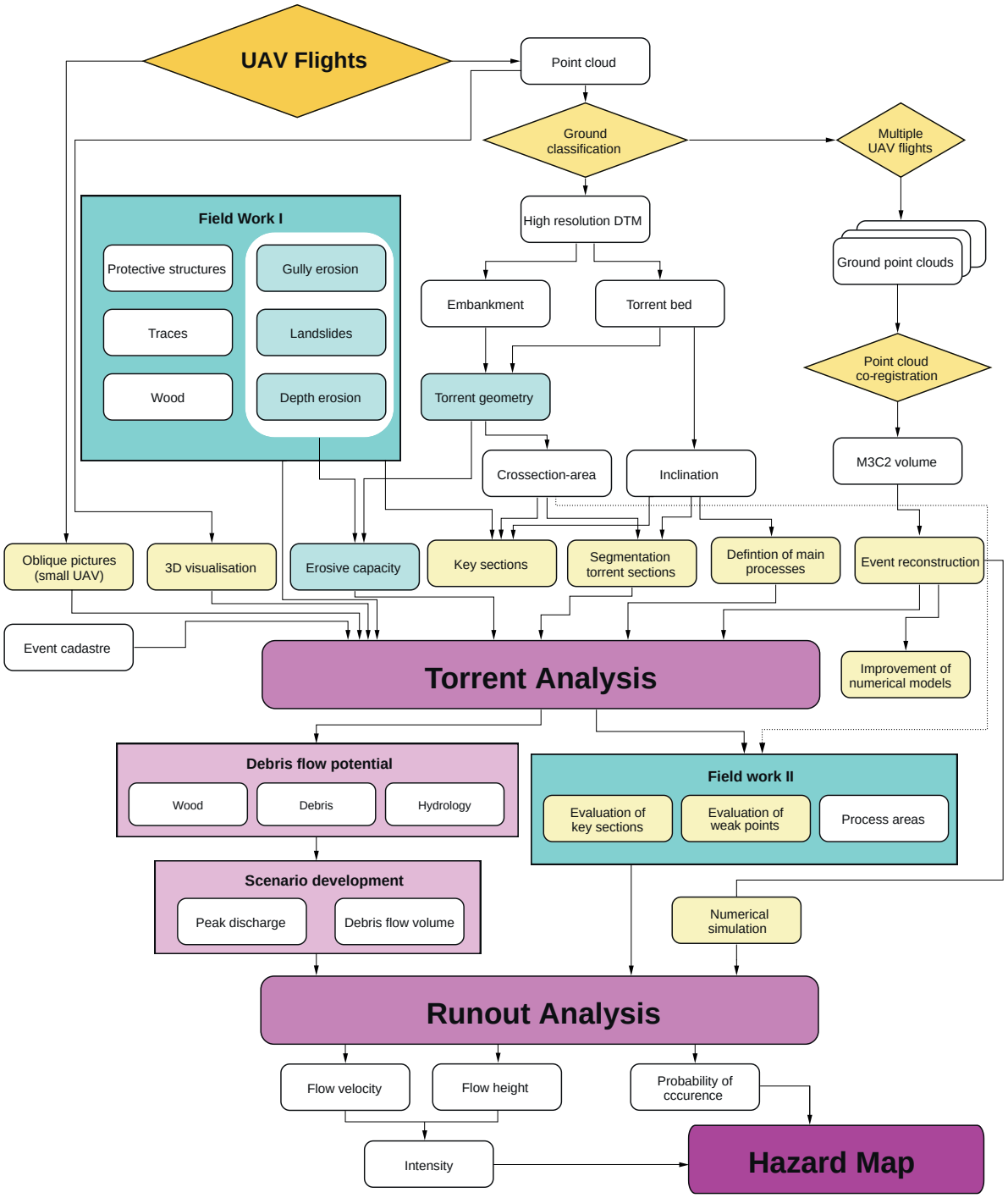


Figure 6.1. Suggested hazard assessment for the use with UAV data. The automatically derived high-resolution DTM is used to calculate geometry attributes, segment torrent sections, and determine key sections. Specific entities such as gullies, landslides, and depth erosion require field-based evaluations. Oblique imagery and 3D point visualizations, as well as event cadastres, contribute further to the torrent analysis. If available, multiple UAV flights allow for the reconstruction of past events and may further improve numerical simulations. Along the runout analysis, the intensity associated with a specific probability of occurrence is determined based on expert on-site assessments and simulations of the relevant processes.

6.3.0.1. Torrent Analysis

The torrent analysis involves a qualitative and quantitative characterization of the catchment. Three important components – hydrology, sediment type and distribution and alluvial/dead wood – are assessed.

The scope of the torrent analysis must be clearly defined in close cooperation with the authorities. Important sources such as the event and protective structure cadastres are consulted. Hydrological calculations are used to estimate torrent runout. For the assessment of deadwood, the respective forest area is important. With regard to sediments, the terrain and torrent bed are analysed.

At this stage of the analysis, UAV-based products offer the possibility of a fast, reliable and objective assessment of the catchment. The raw, dense point cloud provides a three-dimensional visualisation of the catchment. With small UAVs steep, inaccessible terrain can be investigated. With multiple flight missions, active areas can be targeted. Therefore, the point clouds need to be co-registered. Once M3C2 distances are derived, sediment transport patterns provide an indication of the expected frequency of future events. Further M3C2 difference reconstructions of debris flow events can help to improve numerical models and calibrate parameters for site-specific numerical simulations along the subsequent runout analysis.

Attempts to estimate a debris flow volume based on empirical relationships are severely limited (Rickenmann, 1999). Those empirical relationships are based on local conditions in specific regions. These relationships are not always applicable to other regions without further consideration. However, field-based geomorphological assessment are highly indispensable for a sophisticated hazard assessment. Automated quantification of torrent characteristics facilitates efficient planning of fieldwork. Key parameters such as the cross-section area and inclination can be used to distinguish homogeneous sections, which can be further used within geomorphologic assessment methods (eg. SEDEX (Frick et al., 2011)). However, it remains difficult to quantify the sediment supply based on remote sensing data. Further, automatic characterizations of grain-size distributions may provide an indication on the erodability of the materials. The division of the torrent into homogeneous sections should also consider the occurrence of landslides, probability of wood jamming, the presence of bed debris and the potential for bed surface erosion. These factors need to be accounted manually. More applied studies are needed to be able to derive robust erosion estimates based on remote-sensing data. Certainly fieldwork in the catchment will remain essential in future for a profound hazard assessment. However, UAV remote sensing data helps to organize the fieldwork more efficiently. The ability to be able to fly directly to the areas of interest saves a lot of time.

In the first field visit, protective structures and old traces of previous debris flows are identified. The newly available 3D visualisation of torrent characteristics helps to identify important key sections. The introduced automated extraction of torrential properties holds some limitations: The cross-section area derives from a simplified trapezoidal area considering the average embankment height of both sides. To make the cross-section calculation more robust, both sides are considered. The effect of misclassified embankments would be more pronounced if only the lower of two embankments were considered in the calculation. Further, near confluences the calculation domain is limited to

the calculated watershed of the sub-catchment and along steep hillslopes the-based classification is severely restricted. With further development efforts these limitations need to be addressed.

Beside field investigations and the analysis of geodata, the study of past events forms a third important pillar of the analysis. If an event cadastre is available, it provides valuable information on the characteristics of the debris-flow prone catchment.

After the initial field investigations and the first study of the catchment characteristics, the main scenarios for hydrology, sediments and driftwood have to be defined. However, the debris load and thus also the associated annual return period of a debris flow event depends on the event-triggering factors. The duration and intensity of precipitation, the combination with snowmelt and the pre-moisture in the soil have a major influence on the size of the expected debris load (Baer et al., 2017; Bennett et al., 2014; Hirschberg et al., 2019). Further, event sequence scenarios have to be defined and considered. Thereby, the interactions of threshold processes, process transformations and process linkages need to be taken into account (Allen et al., 2022).

6.3.0.2. Runout Analysis

Within the runout analysis the site-specific scenarios are used to determine the intensity of the debris flow at a specific location for a specific probability of occurrence. Therefore the hazard areas are transferred into intensity maps based on expert based assessment in the field and simulations for each relevant process source. UAV-based observations can provide an overview of the critical torrent sections and weak points, which need to be assessed in more detail, usually in a second block of fieldwork. While the measurement of critical cross-sections can largely be done on the basis of UAV data, other torrent cross-sections near bridge culverts and dikes still require closer on-site inspection.

When assessing the intensity of a specific debris flow event, the potential debris flow volume is the most important parameter (Fuchs et al., 2008). For a reliable estimation of the expected debris flow volume geomorphological assessments are indispensable. The estimated debris flow volume can then be used to derive the associated peak discharges, flow velocities and runout distances. With numerical models such as RAMMS, a spatial distribution of debris flow susceptibility within a user-defined boundary condition can be derived based on a mathematical formulation that defines the material flow behavior. UAV-based high resolution DTM provide a highly accurate basis for sophisticated simulations. A major difficulty remains to accurately determine the input parameters. Simulations are only as good as their input (Jakob, 2005). If available, multi temporal UAV data allow an accurate calculation of the required parameters. However, it is very important to critically validate the simulation results, especially regarding possible sediment transport ratios and debris flow scenarios. Based on the combination of fieldwork and simulation results, the debris flow volume and the associated process-relevant parameters such as peak discharge and flow velocity can be determined.

The frequency of a debris flow event may be described by several probability concepts (Fuchs et al., 2008). This makes it especially challenging to compare different hazard assessments with each other. The relation to past events is likely to be the most reliable indication (Rickenmann, 1999). With

regard to the ongoing climate change, it may become questionable how reliable past events are for the assessment of events in the up-coming decades, especially with regard to thawing permafrost (Patton et al., 2019; Stoffel et al., 2014). However, repetitive UAV flights enable to record debris flow events precisely and provide a great data basis to improve the understanding of the effect of climate change on the magnitude and frequency of future debris flow events.

6.4. Estimation of Debris Flow Magnitude and Erosion Depths

Both the Gertsch and SEDEX methods apply geomorphological assessments to determine expected debris flow erosion rates. Both methods rely strongly on field observations, experience and expert knowledge. At first inspection, the Gertsch method appears to be more complicated and involved than the SEDEX method. However, once the GIS data is prepared and the evaluation matrices are understood, the method guides the user step by step through the entire assessment. Difficult parameters such as the erosion rate are elaborated in a reproducible and comprehensive way. The results are therefore transparent and the argumentation can be easily followed by the approval authorities. Although the SEDEX method defines more precisely the sources of sediment production, a clear division must be made between embankment, torrent bed, gullies and landslide activity. This exact definition of the sediment source can lead to some discussion, as it is sometimes difficult to differentiate between landslide and embankment. This unclear separation and the fully expert based estimation of the erosion depth often produces large uncertainties.

How much material will be eroded therefore remains a highly difficult question. A method based on remote sensing data would be helpful. However, the same difficulties we encounter in the assessment method are also found in the UAV analysis: It is also difficult to differentiate clearly between embankment and adjacent slope, especially in vegetated terrain and torrent sections with steep adjacent slopes. In our analysis, we found that the derived median erosion values along the torrent segments are often close to the field-based estimates. This result suggests that we can consider the derived geometrical measures of the torrent to be accurate enough to be used for the assessment of the expected debris flow volume. The determination of the erosion capacity, which transforms the automatic derived contributing area to an estimation of a contributing volume, remains an on-going challenge. The contributing volume can be understood as the potential erosion volume. Based on the summation of the potential erosion volume and subtracting expected deposited volumes along the channel, we can determine the debris flow volume at the valley floor for a given scenario.

The inclination of the torrent bed can provide a rough estimate of the potential erosive power of a debris flow (Rickenmann, 2014). Steep inclinations are associated with higher debris flow velocities and therefore higher basal shear stresses. However, these higher shear stresses must be compared to the material properties of the torrent bed. A fundamental challenge to accurately estimate erosion rates is an accurate quantification of the bed material properties. For this, the grain size distribution may provide some indication on the sediment availability and erosion behaviour. We overview three different approaches:

- Carrivick and Smith (2019) summarize three main approaches using Structure-from-Motion to map grain-sizes. The first method is called "Photo-sieving" and implemented in various software packages such as BASEGRAIN (Detert and Weitbrecht, 2012). It aims to map individual surface grains form close-range overhead pictures. There are some studies (eg. Langhammer et al. (2017)), which derived georeferenced orthophoto mosaic from low-altitude UAV surveys with resulting GSD of 1.5 cm. The same application with our studied catchments would not be possible due to disturbing shadow effects and a to coarse GSD.
- The second method uses grain-size proxies from images. Empirical relationships and image properties are elaborated Carbonneau et al. (2004). Recent studies (Tamminga et al., 2015; Woodget and Austrums, 2017; Woodget et al., 2018) have applied this method on UAV Structure-from-Motion data. Woodget and Austrums (2017) have worked with a orthomosaic GSD of 1 cm.
- The third method uses grain sizes proxies from topography. Empirical relationships between surface characteristics and the topography are used to determine the grain size (Heritage and Milan, 2009; Schneider et al., 2015). This method depends not as strongly on shadow effects as it is derived from DTM data. To characterize the roughness accurately a suitable grid spacing and neighborhood has to be chosen (Florinsky and Kuryakova, 2000; Sappington et al., 2007). According to Brožová et al. (2021) to evaluate the roughness of a terrain regarding natural hazards such as avalanche a DTM with 1 m grid spacing and neighborhood of 7 m is well suited. To investigate the roughness in fluvial gravel beds, Groom et al. (2018b) states that the investigation areas should be around 16·D_{50A}, where D_{50A} is the median grain size of the bed topography and the grid spacing should be 1 mm at largest. However, other studies applied UAV-based Structure-from-Motion to determine the grain size with a grid spacing up 10 cm (Groom et al., 2018a; Pearson et al., 2017).

To characterize grain size along debris flow prone torrents, the resolution of the DTM probably should be increased compared to our available data. Methods which derive from the topography are likely to provide more consistent results, as they do not depend on limiting shadow effects of the images. Clearly more quantitative studies are necessary to find an appropriate workflow for debris flow prone catchments.

Grain size analysis alone cannot be used to make general statements of the erosive capacity of a specific debris flow torrent. Current methods such as those developed by Gertsch (2009) provide only a rough estimate of the bed erodibility or the possible erosive power of the debris flow. The depth of sediments available for erosion can hardly be derived from remote sensing data which captures only surface data. Remote sensing methods could be further developed and automated, if an estimation of the sediment source, including depth, is available. Berger et al. (2018) have developed an approach, which estimates the potential bed load transport at the scale of a hazard indication map. The contributing area was derived from the cross-sectional area of the torrent and the expected hydrological runoff. The erosive capacity was estimated in the field and added as an additional parameter. However, Berger et al. (2018) encountered several limitations. A further difficulty appears when attempting to estimate the volume of sediments originating from adjacent hill

slope processes. Gertsch (2009) does not state how to determine these volumes. It is only pointed out that for an accurate assessment, traces such as previous landslides, tilted trees, tension cracks, soil wetness and the availability of solids should be evaluated in the field. In comparison, the SEDEX method of Frick et al. (2011) likewise depends on expert knowledge. The SEDEX method gives a clear description how the geometrical volume of such processes is calculated when assessing hill slope processes such as landslides and rill erosion. It appears that the characterization of the available sediments, using either the Gertsch or SEDEX methods, must be derived in large parts from field studies. Both methods remain highly subjective.

Overall, we can conclude that our newly developed method enables an objective calculation of the contributing area, which can directly be applied within the SEDEX framework. For a clear and reliable estimation of entrainment further research in this field is highly indispensable. Multiple UAV flights can provide a highly valuable data sets, where the entrainment behavior of torrents can be validated. At the example of the debris flow Fraschmardin 2019, we could observe that the pattern of erosion and deposition in the catchment meets the division into section which was done in the field and evaluated on the basis of the automatically derived torrential properties (specifically cross-section area and inclination). A new method could be developed, which combines useful elements of Gertsch (2009) and Frick et al. (2011) with newly available automated extracted torrential properties.

6.5. Quantification of Change Detection

To accurately determine debris flow volumes, differences between measured surfaces from two or more UAV flights (pre- and post-event) must be matched as exactly as possible. UAV flights using RTK-PPK are often already highly accurate (Zhang et al., 2019). Nevertheless, an analysis of the systematic errors remains indispensable. These errors tend to be most pronounced at the edge of the photogrammetrical surveys. It is therefore advisable to fly over a larger area to minimize errors concerning the region of interest. A brief check of the systematic errors on the basis of the two DSMs is essential for any change detection analysis. If errors are not removed, there remains a high risk that these will be erroneously interpreted as either erosion or deposition (Lane et al., 2003).

Systematic errors need to be corrected when they exceed the level of detection, which falsifies the quantification of erosion and deposition processes. However, to quantify debris flow processes there is often no need to correct systematic errors as a debris flow causes normally larger differences than 30 cm in z-direction, which is often approximately the accuracy level of unprocessed RTK-PPK data. Nevertheless, we found that with a sophisticated error correction and precise co-registration the level of detection of our photogrammetric UAV derived terrain models can be reduced to approximately 15 cm. With this accuracy it is even possible to quantify surface erosion processes.

Co-Registration and correction of systematic errors is often a very subjective process. Depending on the data, different user workflows and requirements may lead to the best results. If the area of interest is only covering a small part of the study area or the systematic error is not strongly pronounced, it is possible to just use a constant shift value to correct the systematic error. In such cases often only a tiny doming or tilt effect is present and the data can be co-registered with ICP

(Besl and McKay, 1992). In steep alpine terrain tiny shifts in x or y-direction lead to large errors in z-direction. It is therefore often very demanding to fully co-register the data correctly. However, ICP routines can help to reduce such errors.

At the example of the Arelen catchment we modelled the errors between two flight missions on the basis of two tie-point clouds. In noisy data sets with significant vegetation or shadows the generation of an accurate error model based on tie-point clouds is limited. In such cases stable ground points need to be selected manually. Another approach that may lead to improved results is the application of a Monte Carlo simulation proposed by James et al. (2017a) to identify appropriate camera calibration processing settings.

Cook and Dietze (2019) tested a further method for co-registration. Along this proposed workflow all pictures of both flight missions are used for the tie-point generation. Through the use of those common tie-points the resulting dense point clouds are automatically aligned more accurately. However, this method does not tackle systematic errors as it is rather modelling remaining systematic differences between two flight missions or aims to optimize the bundle adjustment as it is proposed by James et al. (2017a). We have found that this approach matches the two point clouds along vertical and horizontal shifts. Systematic errors with a spatial dependency are not corrected. Along certain applications, such as the debris flow event in Fraschmardin, the combined alignment approach of Cook and Dietze (2019) leads to sufficiently accurate results.

Errors vary depending on terrain, camera quality, accuracy of camera calibration parameters, flight pattern and camera orientation. The high-end quality camera of the WingtraOne allows a better estimate of the camera calibration parameters in the bundle adjustment, compared to the noisier lens of the Phantom 4 RTK. Distortion effects are major in the border area, it is therefore advisable to define a buffer of 20% around the main area of interest. In steep incised terrain systematic errors are often from minor importance.

For the debris flow event at Fraschmardin, we compared C2M, M3C2 and DoD difference calculations. Based on the results, we propose to apply the M3C2 algorithm for a reliable estimation of erosion and deposition. Due to vegetation, the algorithm is limited to occlusion. Therefore, the calculated volume tends to be underestimated. We observed a further limitation when analyzing rough surfaces. These surfaces are seldom identical along different surveys. Roughness also affects the computation of surface normal orientation, which generally leads to a potential overestimation of distances (Bae et al., 2009). As the normal orientation of the surface is changing strongly (as for example the case of deposition and release of landslides), the computation of M3C2 distances is limited. Bernard et al. (2021) propose to overcome this limitation by first locating areas of significant change using M3C2 calculated along 3D normal vectors. In a subsequent step the areas with significant positive or negative change are extracted and a M3C2 distance is calculated based on the vertical normal direction. With this procedure the limitation of shadow effects due to the normal orientation can be avoided. From the vertical normal calculations along the areas of significant change a more accurate volume can be calculated (Bernard et al., 2021). However, for the application with photogrammetric data, especially along vegetated areas, this method is not applicable. Effects of occlusion due to vegetation and erroneously classified ground surfaces limit the M3C2 calculation significantly. We

found that along vegetation segments, it does not make sense to derive a statistical significant change threshold. However, the proposed error propagation by Winiwarter et al. (2021) may lead to an improvement of the definition of the level of significant change detection. Further research on that would be needed. With the use of LiDAR data, the proposed workflow by Bernard et al. (2021) introduces a significant increase in the accuracy for the calculation of volume change. For photogrammetric data along vegetated terrain, we infer that this additional calculation effort is not worthwhile.

Along large catchments with less inclined regions not the entire debris flow volume of eroded sediments will reach the channel outlet. Indices such as the sediment connectivity can provide a rough understanding on degree of connection between the different landform elements (Heckmann et al., 2018).

The extent to which different catchment areas in an alpine watersheds are interconnected has a major influence on the reach and intensity of mass movements. In particular, the type of connectivity between slope and channel has a significant influence on the sediment input of torrents. Connectivity is dependent on a structural component, which depends on the morphology of the surface, and a process component, which is related to the mode of transport and material composition. Over longer time units, the process component shapes the surface of the landscape (Cavalli et al., 2017, 2013).

The sediment connectivity index developed by Cavalli et al. (2013) is designed for fluvial sediment transport. Processes such as rockfalls and avalanches cannot be represented. The index of Cavalli et al. (2013) is based exclusively on an elevation terrain model. Rindsfueser (2020) has applied graph theory introduced by Fressard and Cossart (2019) to refine sediment connectivity with a special focus of structural elements such as barriers and buffers along a torrential catchment. Further Tiranti et al. (2018) have developed a semi-quantitative assessment method to quantify the sediment volumes originating by slope processes and contributing to the sediment cascade of the mainstream of large mountain valleys. Sediment source areas where mapped manually within the approach of Tiranti et al. (2018). Other studies such as Meyer et al. (2014) have aimed to model debris flow release zones at large scale based on derivations of digital terrain models. However, further research is needed to be able to incorporate the availability and erodibility of sediments accurately into the sediment connectivity models, especially at high spatial resolution. Therefore, different approaches may have to be combined with each other. How UAV derived terrain models can be used to calculate the sediment connectivity needs to be tested in detail. We expect limitations along densely vegetated areas.

In summary, repeated UAV flights can contribute significantly to a better understanding of ongoing processes in the catchment with respect to climate change. Annual flights also increase the probability of detecting a debris flow event and thus improve numerical simulations. Along clearly defined and not too densely vegetated torrents, the characteristics of debris flows can be reliably extracted with a automatic procedure. In complex terrain the automatic workflow is limited and in the uppermost part of the catchment the expected volumes have to be estimated manually. However, the model needs now to be further tested on a broader basis.

Change detection with photogrammetric UAV data is severely limited by vegetation. High-precision methods, such as those presented by Bernard et al. (2021), cannot be usefully applied to erroneous results from photogrammetric data. For these data along vegetated areas, the standard M3C2 approach provides sufficiently accurate results. Both photogrammetric classification algorithms presented provide consistent results. Depending on the terrain, the better performing algorithm should be chosen. A combination of the two may also lead to better results.

Conclusions

The study of how to meaningfully integrate UAV survey data into hazard assessment procedures is a key research problem in natural hazard science and engineering. UAVs are likely to bring along many advantages and introduce a major step towards a more reproducible and objective hazard assessment. Repeated photogrammetric UAV flights provide therefore a highly valuable data source to understand how the torrent will evolve over time. Fieldwork provides unique insights into the character of the catchment and will therefore remain an important element within the assessment procedure. To properly capture the morphology of the landscape, the obtained point cloud data must be classified into ground and non-ground points. Both presented classification routines provide reliable results in alpine terrain. However, there are distinct limitations, such as the removal of spurious vegetation effects, that must be accurately quantified. Although bushes and trees are well detected, the Terrasolid method has difficulties to classify grassland and highly rough, steep terrain. The LAStools routine makes use of a reference height algorithm to offset this problem. Both routines cut off sharp edges. We found, for example, that in shrub forests, the photogrammetric derived results tended to be 1.4 m above the reference height of the LiDAR-based acquisitions. Of course, LiDAR-based measurements could be used exclusively to identify missing ground points underneath dense vegetation. However, this is associated with higher costs.

We developed an algorithm to characterize torrent properties using previously classified high resolution terrain data. The automated extraction of the torrent properties provides accurate results and therefore can be used to support field measurements. However, a fully automated embankment extraction based on OBIA is challenging, especially when the adjacent hill slopes are rising steep. Along terrain with a significantly changing slope, such as on the debris flow cone, the automated delineation of the outer embankment border is robust. The results provided by the OBIA-based embankment extraction are often highly diverse. Nevertheless, the median value over a whole segmented section are in the same range as the field-based estimations. However, if the terrain is highly complex, it is advisable to manually delineate the "torrential extent". In conclusion, an accurate estimation of torrent processes and erosive capacity is a challenging task. Field work will remain indispensable, especially to assess the impact along the key torrent sections.

UAV-based products can be used by hazard engineers in a variety of ways. The raw three-dimensional point cloud provides a first overview of the overall catchment. Torrent properties, such as cross-sectional area and torrent bed inclination, can be derived from the algorithms. The data can be used to characterize the torrent and to segment the torrent into homogeneous sections. This allows the engineer to concentrate on the relevant key sections. Alpine terrain is highly challenging for UAV flights. A survey incorporates a few hours workload. The weather needs to be stable and there should be no snow and and as few leaves as possible on the trees.

The use of UAVs allows long-term monitoring of critical torrents using multiple UAV flights. Long-term torrent data that captures pre- and post-event states can be accurately quantified by calculating M3C2 differences using time-spaced point clouds. The accuracy of the UAV data can be enhanced as sets of two-flight missions are co-registered and systematic errors are identified and removed. Even without debris flow events between UAV missions, valuable information about the ongoing processes can be extracted. Erosion and deposition patterns of debris flow events facilitates the back-calculation of events and thus the definition of site-specific model input parameters. This will certainly contribute to the further improvement of numerical models, specifically the improvement of debris flow entrainment algorithms. The collection of temporal data of debris flow prone catchments is further essential to quantify the ongoing effects of climate change.

Further interdisciplinary research that combines remote sensing techniques and hazard assessment procedures is indispensable. The Terrasolid-based algorithm needs further improvement along the differentiation of high vegetation and grassland. Furthermore, the noise handling along the LAStools-based routine requires improvement. In a next step, the classification routines to characterize the torrent properties need to be applied to a wider variety of different study sites. Once a large data basis of many different torrents with UAV and manual measurements is available, the algorithm to automatically extract the torrent perimeter could be coupled to machine learning approaches. Finally, we note that studies to quantify torrent grain size distributions and sediment connectivity certainly require more research, as this will improve our technical ability to couple UAV measurements to existing torrent assessment approaches. Overall, further work is essential with regard to a more reproducible hazard assessment of debris flow prone catchments. However, UAV data and their introduced derivations have the potential to enable a step towards a more objective hazard assessment.

Bibliography

Abele, G. (1974). *Bergstürze in den Alpen: Ihre Verbreitung, Morphologie und Folgeerscheinungen*. Number 25 in Wissenschaftliche Alpenvereinshefte. München.

Adams, M., Fromm, R., and Lechner, V. (2016). High-Resolution Debris Flow Volume Mapping with Unmanned Aerial Systems (UAS) and Photogrammetric Techniques. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B1:749–755.

Agisoft (2021). *Metashape User Manual - Professional Edition Version 1.6.5.11249*. Agisoft LLC, https://www.agisoft.com/downloads/user-manuals/.

Allen, S., Frey, H., Haeberli, W., Huggel, C., Chiarle, M., and Geertsema, M. (2022). Assessment principles for glacier and permafrost hazards in mountain regions.

Anders, N., Valente, J., Masselink, R., and Keesstra, S. (2019). Comparing Filtering Techniques for Removing Vegetation from UAV-Based Photogrammetric Point Clouds. *Drones*, 3(3):61.

Bae, K.-H., Belton, D., and Lichti, D. (2009). A closed-form expression of the positional uncertainty for 3D point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):577–590.

Baer, P., Huggel, C., McArdell, B. W., and Frank, F. (2017). Changing debris flow activity after sudden sediment input: a case study from the Swiss Alps. *Geology Today*, 33(6):216–223.

BAFU (1997). *Berücksichtigung der Hochwassergefahren bei raumwirksamen Tätigkeiten*. Bundesamt für Umwelt (BAFU).

BAFU (2016). *Schutz vor Massenbewegungsgefahren. Vollzugshilfe für das Gefahrenmanagement von Rutschungen, Steinschlag und Hangmuren*. Bundesamt für Umwelt (BAFU).

Becker, C., Rosinskaya, E., Häni, N., D'Angelo, E., and Strecha, C. (2018). Classification of Aerial Photogrammetric 3D Point Clouds. *Photogrammetric Engineering & Remote Sensing*, 84(5):287–295.

Bennett, G. L., Molnar, P., McArdell, B. W., and Burlando, P. (2014). A probabilistic sediment cascade model of sediment transfer in the Illgraben. *Water Resources Research*, 50(2):1225–1244.

Berger, C., Mani, P., Pauli, M., and Caduff, U. (2018). Praxistaugliche modellansätze zur abschätzung der geschiebelieferung am beispiel braunsbach. *WASSERWIRTSCHAFT*.

Bergmeister, K., Suda, J., Hübl, J., and Rudolf-Miklau, F. (2009). *Schutzbauwerke gegen Wildbachgefahren: Grundlagen, Entwurf und Bemessung, Beispiele*. John Wiley & Sons.

Bernard, T. G., Lague, D., and Steer, P. (2021). Beyond 2D landslide inventories and their rollover: synoptic 3D inventories and volume from repeat lidar data. *Earth Surface Dynamics*, 9(4):1013–1044.

Besl, P. and McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256.

Brodu, N. and Lague, D. (2012). 3D terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology. *ISPRS Journal of Photogrammetry and Remote Sensing*, 68:121–134.

Brožová, N., Baggio, T., D'Agostino, V., Bühler, Y., and Bebi, P. (2021). Multiscale analysis of surface roughness for the improvement of natural hazard modelling. *Natural Hazards and Earth System Sciences*, 21(11):3539–3562.

Bühler, Y., Adams, M. S., Bösch, R., and Stoffel, A. (2016). Mapping snow depth in alpine terrain with unmanned aerial systems (UASs): potential and limitations. *The Cryosphere*, 10(3):1075–1088.

Carbonneau, P. E., Lane, S. N., and Bergeron, N. E. (2004). Catchment-scale mapping of surface grain size in gravel bed rivers using airborne digital imagery. *Water Resources Research*, 40(7).

Carrivick, J. L. and Smith, M. W. (2019). Fluvial and aquatic applications of Structure from Motion photogrammetry and unmanned aerial vehicle/drone technology. *WIREs Water*, 6(1).

Cavalli, M., Crema, S., Trevisani, S., and Marchi, L. (2017). GIS tools for preliminary debris-flow assessment at regional scale. *Journal of Mountain Science*, 14(12):2498–2510.

Cavalli, M., Trevisani, S., Comiti, F., and Marchi, L. (2013). Geomorphometric assessment of spatial sediment connectivity in small Alpine catchments. *Geomorphology*, 188:31–41.

Cignoni, P., Rocchini, C., and Scopigno, R. (1998). Metro: Measuring Error on Simplified Surfaces. *Computer Graphics Forum*, 17(2):167–174.

Cook, K. L. and Dietze, M. (2019). Short communication: A simple workflow for robust low-cost UAV-derived change detection without ground control points. *Earth Surface Dynamics*, 7(4):1009–1017.

Costa, J. E. (1984). *Developments and Applications of Geomorphology*, chapter Physical Geomorphology of Debris Flows, pages 268–317. Springer Berlin Heidelberg, Berlin, Heidelberg.

Cucchiaro, S., Cavalli, M., Vericat, D., Crema, S., Llena, M., Beinat, A., Marchi, L., and Cazorzi, F. (2018). Monitoring topographic changes through 4D-structure-from-motion photogrammetry: application to a debris-flow channel. *Environmental Earth Sciences*, 77(18):632.

Davoser Zeitung (1900). Murgangereignis Monbiel 21./22.07.1900. 24.07.1900.

de Haas, T., Nijland, W., de Jong, S. M., and McArdell, B. W. (2020). How memory effects, check dams, and channel geometry control erosion and deposition by debris flows. *Scientific Reports*, 10(1):14024.

Detert, M. and Weitbrecht, V. (2012). Automatic object detection to analyze the geometry of gravel grains – a freestand-alone tool. *River Flow*.

Ducard, G. J. J. and Allenspach, M. (2021). Review of designs and flight control techniques of hybrid and convertible VTOL UAVs. *Aerospace Science and Technology*, 118:107035.

Díaz, G. M., Mohr-Bell, D., Garrett, M., Muñoz, L., and Lencinas, J. D. (2020). Customizing unmanned aircraft systems to reduce forest inventory costs: can oblique images substantially improve the 3D reconstruction of the canopy? *International Journal of Remote Sensing*, 41(9):3480–3510.

Esri (2022). *ArcGIS Pro geoprocessing tool reference*. ESRI Inc., https://pro.arcgis.com/en/pro-app/latest/tool-reference/main/arcgis-pro-tool-reference.htm.

Florinsky, I. and Kuryakova, G. (2000). Determination of grid size for digital terrain modelling in landscape investigations—exemplified by soil moisture distribution at a micro-scale. *International Journal of Geographical Information Science*, 14(8):815–832.

Frank, F., McArdell, B. W., Oggier, N., Baer, P., Christen, M., and Vieli, A. (2017). Debris-flow modeling at Meretschibach and Bondasca catchments, Switzerland: sensitivity testing of field-data-based entrainment model. *Natural Hazards and Earth System Sciences*, 17(5):801–815.

Fressard, M. and Cossart, E. (2019). A graph theory tool for assessing structural sediment connectivity: Development and application in the Mercurey vineyards (France). *Science of The Total Environment*, 651:2566–2584.

Frick, E., Kienholz, H., and Romang, H. (2011). *SEDEX Anwenderhandbuch*. Geographisches Institut der Universität Bern.

Frick, E., Kienholz, H., and Roth, H. (2008). Sedex - eine methodik zur gut dokumentierten Abschätzung der Feststofflieferung in Wildbächen. *Wasser Energie Luft*, 2.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2001). *The Elements of Statistical Learning*, volume 1. Springer Series in Statistics New York.

Fuchs, S., Kaitna, R., Scheidl, C., and Hübl, J. (2008). The Application of the Risk Concept to Debris Flow Hazards. *Geomechanics and Tunnelling*, 1(2):120–129.

Geo7 (2000). Guteachten Fraschmardintobel. Technical report.

Geotest (2007). Davos Arelenbach Murgang Gefahrenbeurteilung und Massnahmenvorschläge. Technical Report G0522.1.

Gertsch, E. (2009). *Geschiebelieferung alpiner Wildbachsysteme bei Grossereignissen - Ereignisanalysen und Entwicklung eines Abschätzverfahrens*. PhD thesis, Universität Bern.

Gertsch, E., Lehmann, C., and Spreafico, M. (2012). Methods for the estimation of erosion, sediment transport and deposition in steep mountain catchments. *International Commission for the Hydrology of the Rhine Basin*, (II-21).

Giordan, D., Adams, M. S., Aicardi, I., Alicandro, M., Allasia, P., Baldo, M., De Berardinis, P., Dominici, D., Godone, D., Hobbs, P., Lechner, V., Niedzielski, T., Piras, M., Rotilio, M., Salvini, R., Segor, V., Sotier, B., and Troilo, F. (2020). The use of unmanned aerial vehicles (UAVs) for en-

gineering geology applications. *Bulletin of Engineering Geology and the Environment*, 79(7):3437–3481.

Giordan, D., Hayakawa, Y., Nex, F., Remondino, F., and Tarolli, P. (2018). Review article: the use of remotely piloted aircraft systems (RPASs) for natural hazards monitoring and management. *Natural Hazards and Earth System Sciences*, 18(4):1079–1096.

Gojcic, Z., Schmid, L., and Wieser, A. (2021). Dense 3D displacement vector fields for point cloud-based landslide monitoring. *Landslides*, 18(12):3821–3832.

Gojcic, Z., Zhou, C., and Wieser, A. (2020). F2S3: Robustified determination of 3D displacement vector fields using deep learning. *Journal of Applied Geodesy*, 14(2):177–189.

Graf, C., Christen, M., and M (2019). An overview of a decade of applied debris-flow runout modeling in Switzerland: challenges and recommendations. In *7th International Conference on Debris-Flow Hazards Mitigation*.

Groom, J., Bertin, S., and Friedrich, H. (2018a). Assessing Intra-Bar Variations in Grain Roughness Using Close-Range Photogrammetry. *Journal of Sedimentary Research*, 88(5):555–567.

Groom, J., Bertin, S., and Friedrich, H. (2018b). Evaluation of DEM size and grid spacing for fluvial patch-scale roughness parameterisation. *Geomorphology*, 320:98–110.

HADES (1992). Hydrologischer atlas der schweiz.

Hastedt, H. and Luhmann, T. (2015). INVESTIGATIONS ON THE QUALITY OF THE INTERIOR ORIENTATION AND ITS IMPACT IN OBJECT SPACE FOR UAV PHOTOGRAMMETRY. In *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume XL-1-W4, pages 321–328. Copernicus.

Heckmann, T., Cavalli, M., Cerdan, O., Foerster, S., Javaux, M., Lode, E., Smetanová, A., Vericat, D., and Brardinoni, F. (2018). Indices of sediment connectivity: opportunities, challenges and limitations. *Earth-Science Reviews*, 187:77–108.

Heritage, G. L. and Milan, D. J. (2009). Terrestrial Laser Scanning of grain roughness in a gravel-bed river. *Geomorphology*, 113(1):4–11.

Hirschberg, J., McArdell, B. W., Badoux, A., and Molnar, P. (2019). Analysis of rainfall and runoff for debris flows at the Illgrabencatchment, Switzerland. In *7th International Conference on Debris-Flow Hazards Mitigation*, pages 693–700.

Hungr, O., Evans, S. G., Bovis, M. J., and Hutchinson, J. N. (2001). A review of the classification of landslides of the flow type. *Environmental and Engineering Geoscience*, 7(3):221–238.

Hungr, O., Morgan, G. C., and Kellerhals, R. (1984). Quantitative analysis of debris torrent hazards for design of remedial measures. *Canadian Geotechnical Journal*, 21(4):663–677.

Hunziker, G., Berger, C., and Berwert-Lopes, J. (2021). *Modul Wildbäche Praxiskus Gefahrenbeurteilung gravitative Naturgefahren*. Fachleute Naturgefahren Schweiz FAN.

Hürlimann, M., Coviello, V., Bel, C., Guo, X., Berti, M., Graf, C., Hübl, J., Miyata, S., Smith, J. B., and Yin, H.-Y. (2019). Debris-flow monitoring and warning: Review and examples. *Earth-Science Reviews*, 199:102981.

Hürlimann, M., Rickenmann, D., Medina, V., and Bateman, A. (2008). Evaluation of approaches to calculate debris-flow parameters for hazard assessment. *Engineering Geology*, 102(3):152–163.

Iverson, R. M. (1997). The physics of debris flows. *Reviews of Geophysics*, 35(3):245–296.

Jakob, M. (2005). *Debirs-flow Hazards and Related Phenomena*, chapter Debris-fow hazard analysis, pages 411–443. Praxis Publishing.

Jakob, M. and Hungr, O. (2005). *Debirs-flow Hazards and Related Phenomena*. Praxis Publishing.

James, M. R., Antoniazza, G., Robson, S., and Lane, S. N. (2020). Mitigating systematic error in topographic models for geomorphic change detection: accuracy, precision and considerations beyond off-nadir imagery. *Earth Surface Processes and Landforms*, 45(10):2251–2271.

James, M. R., Robson, S., d'Oleire Oltmanns, S., and Niethammer, U. (2017a). Optimising UAV topographic surveys processed with structure-from-motion: Ground control quality, quantity and bundle adjustment. *Geomorphology*, 280:51–66.

James, M. R., Robson, S., and Smith, M. W. (2017b). 3-D uncertainty-based topographic change detection with structure-from-motion photogrammetry: precision maps for ground control and directly georeferenced surveys. *Earth Surface Processes and Landforms*, 42(12):1769–1788.

Johnson, A. M. and Rahn, P. H. (1970). Mobilization of debris flows. *Zeitschrift für Geomorphologie Supplementband*, 9:168–186.

Keilig, K.-P., Dietrich, A., and Krautblatter, M. (2019). Comparison of Multi-temporal Elevation Models of a Debris-Flow Channel. In Shakoor, A. and Cato, K., editors, *IAEG/AEG Annual Meeting Proceedings, San Francisco, California, 2018 - Volume 1*, pages 275–282. Springer International Publishing.

Kienholz, H., Frick, E., and Gertsch, E. (2010). Assessment tools for mountain torrents: SEDEX and bed load assessment matrix. In *Internationales symposion interpraevent in the Pacific Rim - Taipei*, pages 245–256. Internationale Forschungsgesellschaft Interpraevent.

Koenderink, J. J. and van Doorn, A. J. (1991). Affine structure from motion. *Journal of the Optical Society of America*, 8(2):377–385.

Kraus, K. and Pfeifer, N. (1998). Determination of terrain models in wooded areas with airborne laser scanner data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 53(4):193–203.

Lague, D., Brodu, N., and Leroux, J. (2013). Accurate 3D comparison of complex topography with terrestrial laser scanner: Application to the Rangitikei canyon (N-Z). *ISPRS Journal of Photogrammetry and Remote Sensing*, 82:10–26.

Landschaft Davos and Tiefbauamt Graubünden (2000). Wassergefahrenstudie Davos. Technical Report 2779.00-1.

Lane, S. N., Westaway, R. M., and Murray Hicks, D. (2003). Estimation of erosion and deposition volumes in a large, gravel-bed, braided river using synoptic remote sensing. *Earth Surface Processes and Landforms*, 28(3):249–271.

Langhammer, J., Bernsteinová, J., and Mirijovský, J. (2017). Building a High-Precision 2D Hydrodynamic Flood Model Using UAV Photogrammetry and Sensor Network Monitoring. *Water*, 9(11):861.

LAStools (2022). *Efficient LiDAR Processing Software*. rapidlasso GmbH, http://rapidlasso.com/LAStools.

Maisch, M. (1981). *Glazialmorphologische und gletschergeschichtliche Untersuchungen im Gebiet zwischen Landwasser- und Albulatal (Kt. Graubünden, Schweiz)*. PhD thesis, University Zurich.

Melton, M. A. (1965). The Geomorphic and Paleoclimatic Significance of Alluvial Deposits in Southern Arizona. *The Journal of Geology*, 73(1):1–38.

Meyer, N. K., Schwanghart, W., Korup, O., Romstad, B., and Etzelmüller, B. (2014). Estimating the topographic predictability of debris flows. *Geomorphology*, 207:114–125.

Meyrat, G., McArdell, B., Ivanova, K., Müller, C., and Bartelt, P. (2022). A dilatant, two-layer debris flow model validated by flow density measurements at the Swiss illgraben test site. *Landslides*, 19(2):265–276.

Monserrat, O. and Crosetto, M. (2008). Deformation measurement using terrestrial laser scanning data and least squares 3D surface matching. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(1):142–154.

Nex, F. and Remondino, F. (2014). UAV for 3D mapping applications: a review. *Applied Geomatics*, 6(1):1–15.

Niculia, M., Margarint, M. C., and Tarolli, P. (2020). Using UAV and LiDAR data for gully geomorphic changes monitoring. volume 23 of *Remote Sensing of Geomorphology*, pages 271–315. Elsevier.

Olsen, M., Kuester, F., Chang, B., and Hutchinson, T. (2010). Terrestrial laser scanning-based structural damage assessment. *Journal of Computing in Civil Engineering*, 24(3):264–272.

O'Neal, M. A. and Pizzuto, J. E. (2011). The rates and spatial patterns of annual riverbank erosion revealed through terrestrial laser-scanner surveys of the South River, Virginia. *Earth Surface Processes and Landforms*, 36(5):695–701.

Passalacqua, P., Belmont, P., Staley, D. M., Simley, J. D., Arrowsmith, J. R., Bode, C. A., Crosby, C., DeLong, S. B., Glenn, N. F., Kelly, S. A., Lague, D., Sangireddy, H., Schaffrath, K., Tarboton, D. G., Wasklewicz, T., and Wheaton, J. M. (2015). Analyzing high resolution topography for advancing the understanding of mass and energy transfer through landscapes: A review. *Earth-Science Reviews*, 148:174–193.

Patton, A. I., Rathburn, S. L., and Capps, D. M. (2019). Landslide response to climate change in permafrost regions. *Geomorphology*, 340:116–128.

Pearson, E., Smith, M. W., Klaar, M. J., and Brown, L. E. (2017). Can high resolution 3D topographic surveys provide reliable grain size estimates in gravel bed rivers? *Geomorphology*, 293:143–155.

Petracheck, A. and Kienholz, H. (2003). Hazard assessment and mapping of mountain risks in Siwtzerland. In Rickenmann, D. and Chen, C.-L., editors, *Proceedings of the Third International Conference on Debris-Flow Hazards Mitigation: Mechanics, Prediction, and Assessment*, volume 2.

Phillips, C. J. and Davies, T. R. H. (1991). Determining rheological parameters of debris flow material. *Geomorphology*, 4(2):101–110.

Pierson, T. C. (1986). *Hillslope Processes*, chapter Flow behavior of channelized debris flows, Mount St. Helens, Washington, pages 269–296. Binghamton Geomorphology Symposium 16.

Pierson, T. C. (2005). *Debirs-flow Hazards and Related Phenomena*, chapter Hyperconcentrated flow — transitional process between water flow and debris flow, pages 159–202. Praxis Publishing, Berlin, Heidelberg.

Rickenmann, D. (1995). Beurteilung von Murgängen. *Schweizer Ingenieur und Architekt*, 48:1104–1108.

Rickenmann, D. (1999). Empirical Relationships for Debris Flows. *Natural Hazards*, 19(1):47–77.

Rickenmann, D. (2001). Murgänge in den Alpen und Methoden zur Gefahrenbeurteilung. *Wasser - Katastrophe - Mensch. Proceedings 31. IWASA, internationales Wasserbau-Symposium*, page (27 pp.).

Rickenmann, D. (2014). *Methoden zur quantitativen Beurteilung von Gerinnepozessen in Wildbächen*, volume 9. Eig. Forschungsanstalt für Wald, Schnee und Landschaft WSL.

Rickenmann, D. (2016). *Methods for the Quantitative Assessment of Channel Processes in Torrents (Steep Streams)*. CRC Press, London.

Rimböck, A., Barben, M., Gruber, H., Hübl, J., Moser, M andRickenmann, D., Schober, S., and Schwaller, G. (2013). *OptiMeth – Beitrag zur optimalen Anwendung von Methoden zur Beschreibung von Wildbachprozessen*, volume 1. Internationale Forschungsgesellschaft INTERPRAEVENT.

Rindsfueser, N. (2020). Erfassung der strukturellen Sedimentkonnektivität anhand der Graphentheorie Anwendung des Residual Flow Index in einem alpinen Einzugsgebiet. Master's thesis, University Berne.

Roncoroni, M., Mancini, D., Kohler, T. J., Miesen, F., Gianini, M., Battin, T. J., and Lane, S. N. (in review). Centimeter-scale mapping of phototrophic biofilms in glacial forefields using visible band ratios and UAV imagery. *International Journal of Remote Sensing and Remote Sensing Letters*.

Rosser, N., Petley, D., Lim, M., Dunning, S., and Allison, R. (2005). Terrestrial laser scanning for monitoring the process of hard rock coastal cliff erosion. *Quarterly Journal of Engineering Geology and Hydrogeology*, 38(4):363–375.

Rusnák, M., Sládek, J., Kidová, A., and Lehotský, M. (2018). Template for high-resolution river landscape mapping using UAV technology. *Measurement*, 115:139–151.

Sappington, J. M., Longshore, K. M., and Thompson, D. B. (2007). Quantifying Landscape Ruggedness for Animal Habitat Analysis: A Case Study Using Bighorn Sheep in the Mojave Desert. *The Journal of Wildlife Management*, 71(5):1419–1426.

Schneider, J. M., Rickenmann, D., Turowski, J. M., and Kirchner, J. W. (2015). Self-adjustment of stream bed roughness and flow velocity in a steep mountain channel. *Water Resources Research*, 51(10):7838–7859.

Schürch, P., Densmore, A. L., Rosser, N. J., Lim, M., and McArdell, B. W. (2011). Detection of surface change in complex topography using terrestrial laser scanning: application to the Illgraben debris-flow channel. *Earth Surface Processes and Landforms*, 36(14):1847–1859.

Signer, A., Maggetti, M., Keller, F., Winkler, W., Weissert, H., and Pfiffner, O. (2018). Blatt 1197 Davos. – Geoloischer Atlas Schweiz 1: 25 000, Erläuterungen 156.

Spreafico, M., Lehmann, C., and Naef, O. (1996). *Empfehlung zu Abschätzung von Feststofffrachten in Wildbächen*. Number 4. Arbeitsgruppe für operationelle Hydrologie, Bern.

Stocker, A. (2013). Geschiebeabschätzung im dorfbach und wildibach mit den geschiebeabschätzverfahren gertsch und sedex. *Mattertal – ein Tal in Bewegung. Publikation zur Jahrestagung der Schweizerischen Geomorphologischen Gesellschaft*.

Stoffel, M., Mendlik, T., Schneuwly-Bollschweiler, M., and Gobiet, A. (2014). Possible impacts of climate change on debris-flow activity in the Swiss Alps. *Climatic Change*, 122(1):141–155.

Swisstopo (2022a). *SwissALTI3D, das hochpräzise digitale Höhenmodell der Schweiz*. Bundesamt für Landestopografie swisstopo, https://www.swisstopo.admin.ch/de/geodata/height/alti3d.html.

Swisstopo (2022b). *SwissSURFACE3D, das digitale Oberflächenmodell der Schweiz*. Bundesamt für Landestopografie swisstopo, https://www.swisstopo.admin.ch/de/geodata/height/surface3d-raster.html.

Takahashi, T. (2007). *Debris Flow: Mechanics, Prediction and Countermeasures*. Taylor & Francis, London.

Tamminga, A. D., Eaton, B. C., and Hugenholtz, C. H. (2015). UAS-based remote sensing of fluvial change following an extreme flood event. *Earth Surface Processes and Landforms*, 40(11):1464–1476.

Terrasolid (2021). *TerraScan User Guide*. Terrasolid Ltd, https://terrasolid.com/guides/tscan.pdf.

Thur (2019). Massnahmenkonzept Fraschmardinbach. Technical report.

Tiranti, D., Crema, S., Cavalli, M., and Deangeli, C. (2018). An Integrated Study to Evaluate Debris Flow Hazard in Alpine Environment. *Frontiers in Earth Science*, 6.

Tobler, D., Kull, I., Jacquemart, M., and Haehlen, N. (2014). Hazard Management in a Debris Flow Affected Area: Case Study from Spreitgraben, Switzerland. In Sassa, K., Canuti, P., and Yin, Y., editors, *Landslide Science for a Safer Geoenvironment*, pages 25–30, Cham. Springer International Publishing.

Trimble (2022). *User Guide eCognition Developer*. Trimble Geospatial, https://docs.ecognition.com.

Varnes, D. J. (1978). Slope movement types and processes. *Special report 176: Landslides: Analysis and Control, Transportation Research Board*, 176:11–33.

Verhoeven, G. (2011). Taking computer vision aloft – archaeological three-dimensional reconstructions from aerial photographs with photoscan. *Archaeological Prospection*, 18(1):67–73.

Westoby, M. J., Brasington, J., Glasser, N. F., Hambrey, M. J., and Reynolds, J. M. (2012). 'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology*, 179:300–314.

Winiwarter, L., Anders, K., and Höfle, B. (2021). M3C2-EP: Pushing the limits of 3D topographic point cloud change detection by error propagation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 178:240–258.

Woodget, A. S. and Austrums, R. (2017). Subaerial gravel size measurement using topographic data derived from a UAV-SfM approach. *Earth Surface Processes and Landforms*, 42(9):1434–1443.

Woodget, A. S., Fyffe, C., and Carbonneau, P. E. (2018). From manned to unmanned aircraft: Adapting airborne particle size mapping methodologies to the characteristics of sUAS and SfM. *Earth Surface Processes and Landforms*, 43(4):857–870.

Zhang, H., Aldana-Jague, E., Clapuyt, F., Wilken, F., Vanacker, V., and Van Oost, K. (2019). Evaluating the potential of post-processing kinematic (PPK) georeferencing for UAV-based structure-from-motion (SfM) photogrammetry and surface change detection. *Earth Surface Dynamics*, 7(3):807–827.

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., and Yan, G. (2016). An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sensing*, 8(6):501.

Zimmermann, M. (1997). *Murganggefahr und Klimaänderung - ein GIS-basierter Ansatz*. vdf Hochschulverlag AG, Zürich.

Personal Declaration

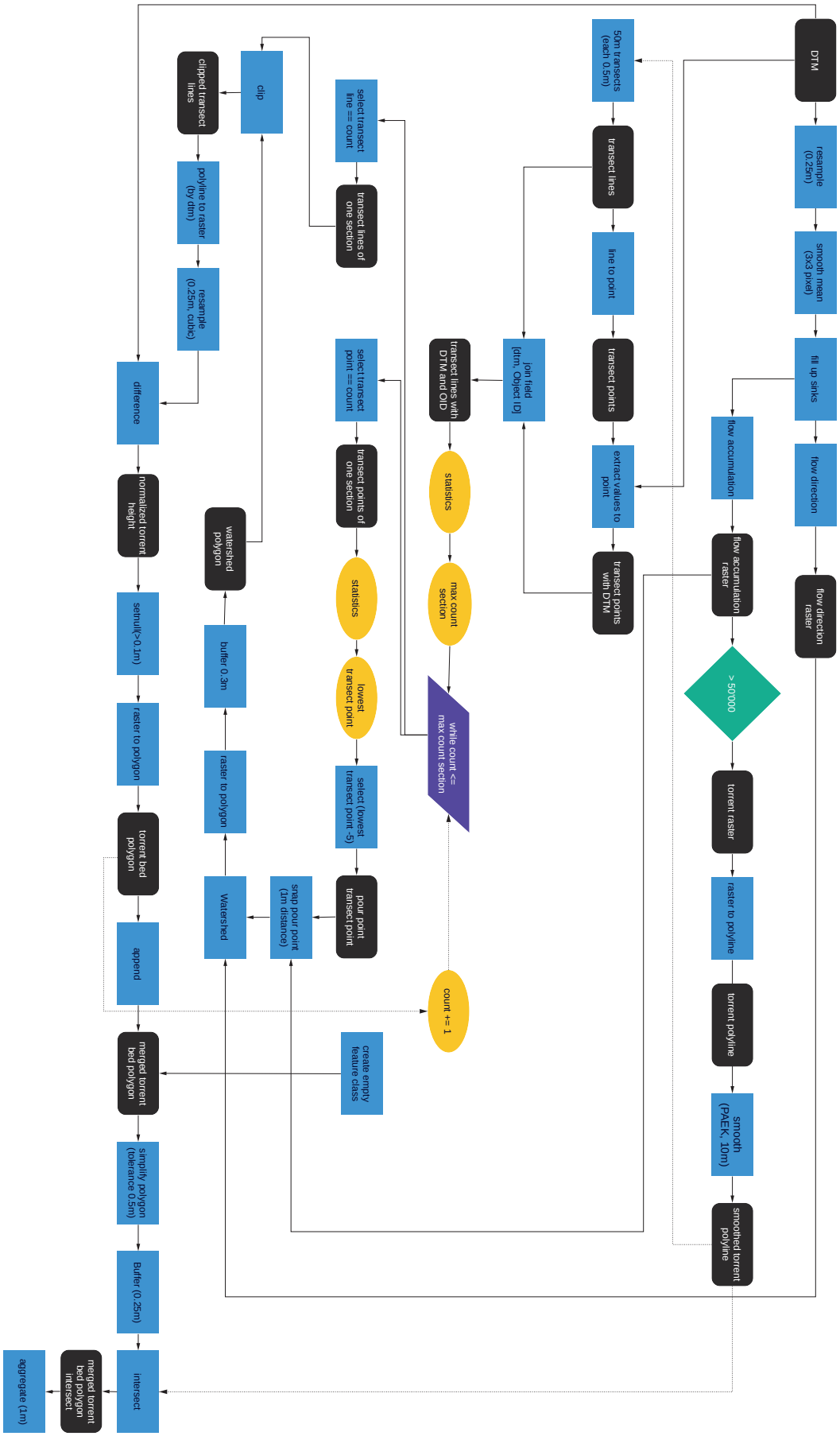
I hereby declare that the submitted Thesis is the result of my own, independent work.
All external sources are explicitly acknowledged in the Thesis.

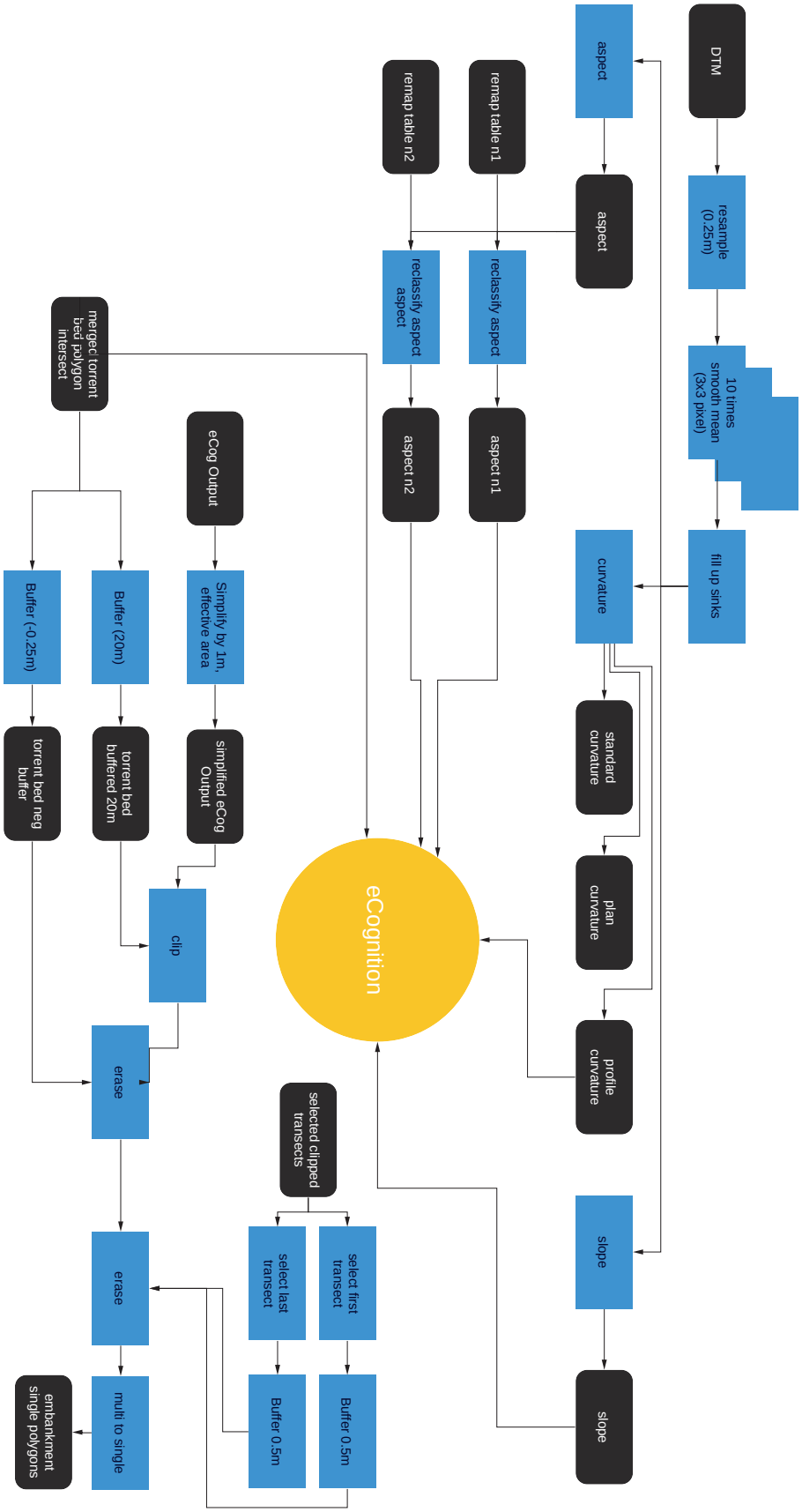
Davos, 30th April 2022

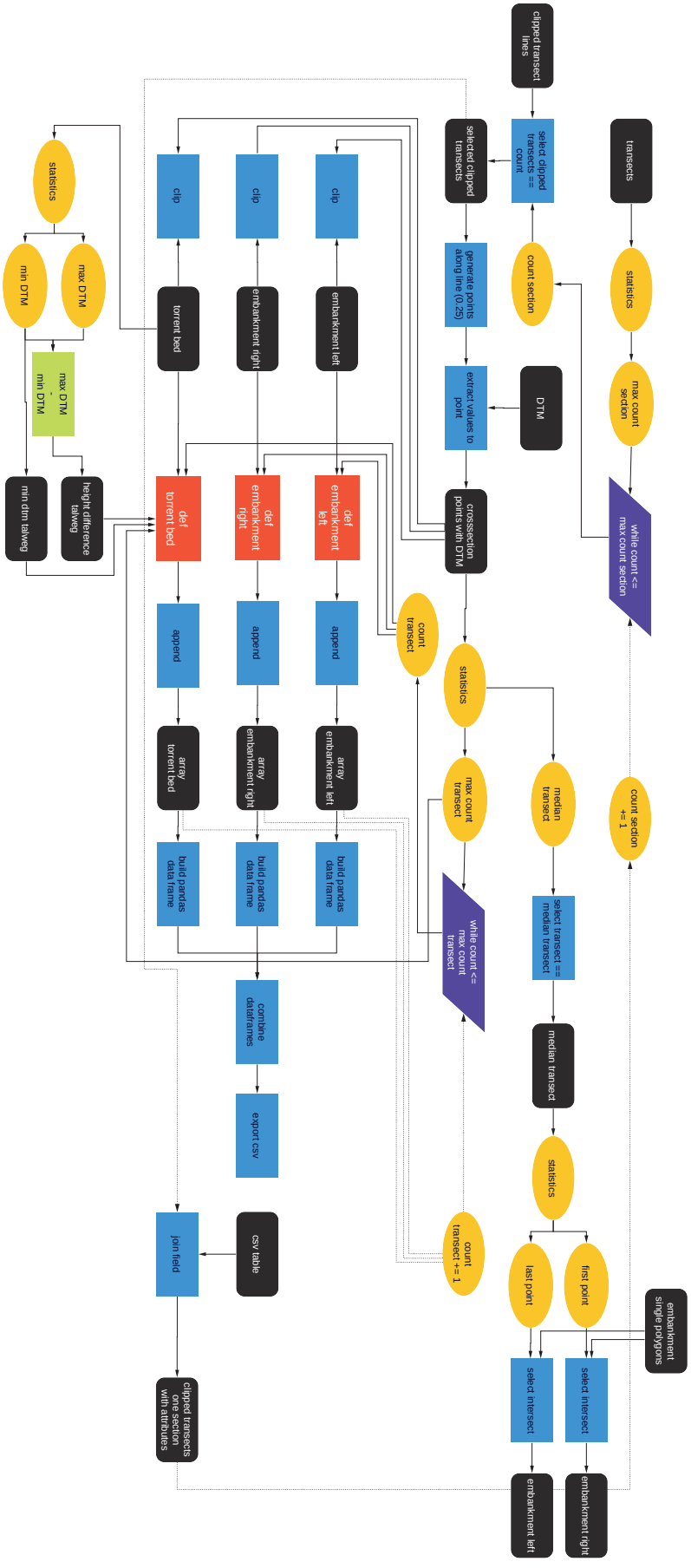
Gregor Rafael Schmucki

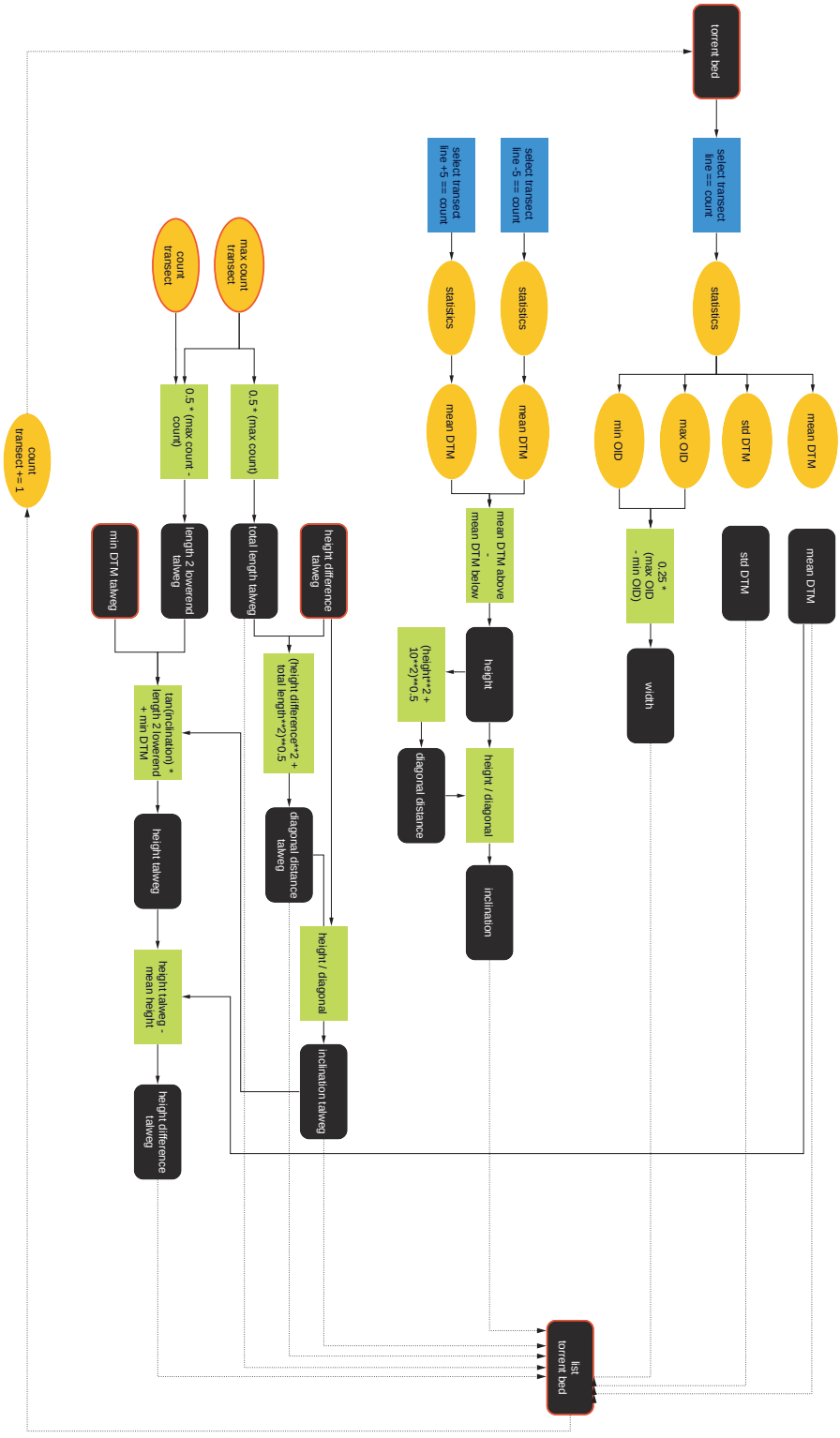
Appendices

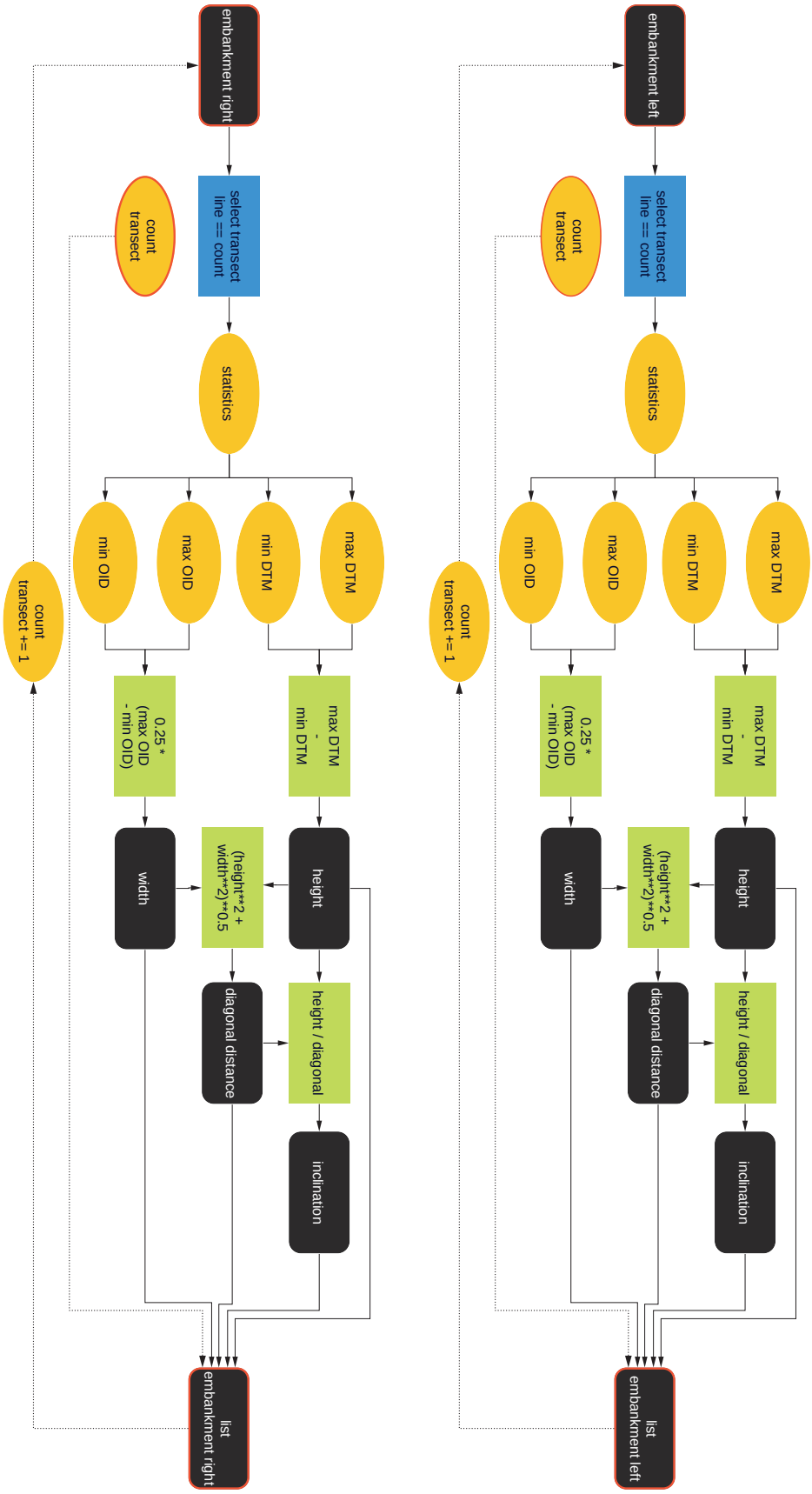
Flowchart Torrential Properties


DTM
resample (0.25m)
smooth mean (3x3 pixel)
fill up sinks
flow direction
flow direction raster
flow accumulation
flow accumulation raster
> 50000
torrent raster
raster to polyline
torrent polyline
smooth (PAEK, 10m)
smoothed torrent polyline
50m transects (each 0.5m)
transect lines
line to point
transect points
extract values to point
transect points with DTM
join field [dtm, Object ID]
transect lines with DTM and OID
statistics
max count section
while count <= max count section
count += 1
select transect line == count
transect lines of one section
select transect point == count
transect points of one section
statistics
lowest transect point
select (lowest transect point -5)
pour point transect point
snap pour point (1m distance)
Watershed
raster to polygon
buffer 0.3m
watershed polygon
clip
clipped transect lines
polyline to raster (by dtm)
resample (0.25m, cubic)
difference
normalized torrent height
setnull(<0.1m)
raster to polygon
torrent bed polygon
append
merged torrent bed polygon
create empty feature class
simplify polygon (tolerance 0.5m)
Buffer (0.25m)
intersect
merged torrent bed polygon intersect
aggregate (1m)


DTM
resample (0.25m)
10 times smooth mean (3x3 pixel)
fill up sinks
aspect
aspect
remap table n2
remap table n1
reclassify aspect aspect
reclassify aspect
aspect n2
aspect n1
curvature
standard curvature
plan curvature
profile curvature
slope
slope
merged torrent bed polygon intersect
eCog Output
Buffer (-0.25m)
Buffer (20m)
Simplify by 1m, effective area
torrent bed neg buffer
torrent bed buffered 20m
Simplified eCog Output
clip
eCognition
erase
selected clipped transects
erase
select last transect
select first transect
Buffer 0.5m
Buffer 0.5m
multi to single
embankment single polygons


clipped transect lines
select clipped transects == count
selected clipped transects
transects
statistics
max count section
count section
generate points along line (0.25)
extract values to point
DTM
crosssection points with DTM
while count <= max count section
count section += 1
count transect
statistics
max count transect
median transect
while count <= max count transect
select transect == median transect
median transect
statistics
last point
first point
count transect += 1
embankment single polygons
select intersect
select intersect
embankment left
embankment right
clip
clip
clip
torrent bed
embankment right
embankment left
statistics
min DTM
max DTM
max DTM - min DTM
min dtm talweg
height difference talweg
def torrent bed
def embankment right
def embankment left
append
append
append
array torrent bed
array embankment right
array embankment left
build pandas data frame
build pandas data frame
build pandas data frame
combine dataframes
export csv
csv table
join field
clipped transects one section with attributes


torrent bed
select transect line == count
statistics
mean DTM
std DTM
max OID
min OID
0.25 * (max OID - min OID)
width
select transect line -5 == count
statistics
mean DTM
select transect line +5 == count
statistics
mean DTM
mean DTM above - mean DTM below
height
(height**2 + 10**2)**0.5
diagonal distance
height / diagonal
inclination
count transect
max count transect
0.5 * (max count - count)
0.5 * (max count)
length 2 lowerend talweg
total length talweg
min DTM talweg
height difference talweg
(height difference**2 + total length**2)**0.5
diagonal distance talweg
height / diagonal
inclination talweg
tan(inclination) * length 2 lowerend + min DTM
height talweg
height talweg - mean height
height difference talweg
list torrent bed
count transect += 1


embankment left
select transect
line == count
count
transect
statistics
min OID
max OID
min DTM
max DTM
count
transect += 1
0.25 *
(max OID
- min OID)
max DTM
-
min DTM
width
height
(height**2 +
width**2)**0.5
diagonal distance
height / diagonal
inclination
list
embankment left
embankment right
select transect
line == count
count
transect
statistics
min OID
max OID
min DTM
max DTM
count
transect += 1
0.25 *
(max OID
- min OID)
max DTM
-
min DTM
width
height
(height**2 +
width**2)**0.5
diagonal distance
height / diagonal
inclination
list
embankment right

Scripts

B.1. LAStools Classification Routine

This Python script classifies into ground points and generates a DTM.

```
import subprocess
import time
from datetime import datetime
import os
import arcpy
import random
import arcpy.management as DM
import arcpy.cartography as CA
import math
from arcpy.sa import *
from arcpy import env
from datetime import datetime
from arcpy.ia import *


#runtime of scipt
global start_time
start_time = datetime.now()


# domain of lasTools applications
# if the following error is raised: the system cannot find the specific path -> the path of the bin folder is likely to be wrong
lasTools_version = r"C:\bin_version_2021_01\bin"




def las2las_vegetation_index(input_n, ground_points_name, working_dir):
    """
    classifies ground points with the help of a reference height model

    variables that may have to be changed in the script:

    ****most important! Run script first and check in which attribute the hight values are saved. ****

    lasTools_version:   Where the programm folder is stored (has to be on the C drive)

    startdir:           the folder where all the study_site_dir is located

    hight attribute:    depending on the output of agisoft, the calculated hight attribute will be stored with a different
                        atribute number. (This attribute will be overwritten, when in the second part of the script, the
                        hight value of a coarse suface is calculated


    green_index:        index which is used to distingluish between green and non-green rgb values

    confidence_index:   if there are confidence values, points with low confidence can be deleted. Therefore the section
                        about confidence values has to be untagged.


    To delete some of the noise, two approaches are considered:

    A)  Building a croase point cloud, to build kind of a new ground. From this new ground all points above a certain
        threshold will be deleted. This procedure is done twice, once with a very croase ground pointcloud and a high
        threshold. At a second stage the ground pointcloud is selected to be thinner and the threshold lower.

    B)  To delete thinner noise LasNoise is applied. In a cuboid of 2x2x1m the amount of points will be counted.
        If there are less points than given in the noise parameter, the points in the cuboid will be deleted.


    gen_ground:         Size of the generalised ground model. (croaser raster and higher hight)

    hight_delete_above: Hight, which above all points will be deleted. (croaser raster and higher hight)

```

```
    gen_ground_2:       Size of the generalised ground model. (finer raster and lower hight)

    hight_delete_ab_2:  Hight, which above all points will be deleted.(finer raster and lower hight)


    noise:              minimum of points in a cuboid of 2x2X1m, if there are less points, the points will be deleted.
    low_veg_hight:      the green (vegetation) areas are divided into to two classes. Low vegetation underneath a certain
                        threshold.



    the bare ground (non green areas) and the areas which are green, but low vegetation are merged together.
    As the merged point cloud has still a lot of noise, noise filtering approaches have to be applied.
    """

    hight_attribute = "4"
    green_index = 1500
    confidence_index = 2
    diff_swissALT3D = 0.25  # difference to reference point cloud

    gen_ground = 6  # m
    hight_delete_above = 600  # cm

    gen_ground_2 = 3  # m
    hight_delete_above_2 = 1  # cm

    noise = 3000

    cores = 50

    # index point cloud for faster processing
    input_name = str(input_n) + ".laz"
    input_file = os.path.join(working_dir, input_name)

    todo = lasTools_version + r"\lasindex -i {} -cores 50".format(input_file)
    p = subprocess.Popen(todo, shell=True)
    p.wait()

    print("index finished")

    # tile point clould for faster processing
    input_name = str(input_n) + ".laz"
    input_file = os.path.join(working_dir, input_name)

    output_name = str(input_n) + ".laz"
    output_file = os.path.join(working_dir, output_name)

    todo = lasTools_version + r"\lastile -i {} -tile_size 50 -buffer 5 -flag_as_withheld -cores 50 -o {}".format(
        input_file, output_file)
    p = subprocess.Popen(todo, shell=True)
    p.wait()

    # calculation of vegetation index
    input_name = str(input_n) + "*.laz"
    input_file = os.path.join(working_dir, input_name)

    todo = lasTools_version + r"\las2las64 -i {} -copy_G_into_register -0 -scale_register 0 2.0 -copy_R_into_register 1 " \
                              r"-subtract_registers 0 1 0 -copy_B_into_register 1 -subtract_registers 0 1 0 " \
                              r"-copy_register_into_intensity 0 -odix _vegindex_2G-RB -olaz -cores {}".format(
        input_file, cores)
    p = subprocess.Popen(todo, shell=True)
    p.wait()

    # classify all green points with intensity higher than green_index as vegetation (3) and others as ground (2)
    input_name = str(input_n) + "*_vegindex_2G-RB.laz"
    input_file = os.path.join(working_dir, input_name)

    todo = lasTools_version + r"\las2las64 -i {} -set_classification 0 -classify_intensity_above_as {} 3 -
      classify_intensity_below_as {} 2 -odix _green -olaz -cores {}".format \
        (input_file, green_index, green_index, cores)

    p = subprocess.Popen(todo, shell=True)
    p.wait()


    # difference to reference height model and saves output in folder all_points
    input_name = str(input_n) + "*_vegindex_2G-RB_green.laz"
    input_file = os.path.join(working_dir, input_name)
    ground_points_file = os.path.join(working_dir, ground_points_name)

```

```
    all_points_folder = os.path.join(working_dir, "all_points")
    if not os.path.exists(all_points_folder):
        os.makedirs(all_points_folder)
    output_name = str(input_n) + "*_all_points.laz"
    output_file = os.path.join(all_points_folder, output_name)

    todo = lasTools_version + r"\lasheight64 -i {} -store_as_extra_bytes -ground_points {} -odir {} -odix _all_points -olaz -cores {}".format(
        input_file, ground_points_file, all_points_folder, cores)

    p = subprocess.Popen(todo, shell=True)
    p.wait()

    print("green index finished" + str(datetime.now() - start_time))


    # keep green points with less than defined distance to reference height model
    input_name = str(input_n) + "*_vegindex_2G-RB_green_all_points.laz"
    input_file = os.path.join(all_points_folder, input_name)

    todo = lasTools_version + r"\las2las64 -i {} -keep_class 3 -drop_attribute_above {} {} -odix _ground -olaz -odir {} -cores {}".format(
        input_file, hight_attribute, diff_swissALT3D, working_dir, cores)

    p = subprocess.Popen(todo, shell=True)
    p.wait()

    # points from last calculation step are reclassfied to class 2
    input_name = str(input_n) + "*_vegindex_2G-RB_green_all_points_ground.laz"
    input_file = os.path.join(working_dir, input_name)

    low_veg_folder = os.path.join(working_dir, "low_not_veg")
    if not os.path.exists(low_veg_folder):
        os.makedirs(low_veg_folder)
    output_name = str(input_n) + "*_low_not_veg.laz"
    output_file = os.path.join(low_veg_folder, output_name)

    todo = lasTools_version + r"\las2las64 -i {} -set_classification 2 -odir {} -odix _lnv -olaz -cores {}".format(
        input_file, low_veg_folder, cores)

    p = subprocess.Popen(todo, shell=True)
    p.wait()

    print("Wiese als Ground klassifziert" + str(datetime.now() - start_time))

    # merge selected low green points with all primary ground points (with vegetation index)

    dir_a = low_veg_folder

    dir_b = all_points_folder

    list_a = os.listdir(low_veg_folder)
    list_b = os.listdir(all_points_folder)


    merge_folder = os.path.join(working_dir, "merge")
    if not os.path.exists(merge_folder):
        os.makedirs(merge_folder)

    run = 0

    for f in list_a:
        file_name_no_extention_a = os.path.splitext(os.path.basename(f))[
            0]  # takes file name, without .laz (first element)
        elements_a = file_name_no_extention_a.split("_")
        id_a = str(elements_a[0]) + "_" + str(elements_a[1]) + "_" + str(elements_a[2])
        for l in list_b:
            file_name_no_extention_b = os.path.splitext(os.path.basename(l))[
                0]  # takes file name, without .laz (first element)
            elements_b = file_name_no_extention_b.split("_")
            id_b = str(elements_b[0]) + "_" + str(elements_b[1]) + "_" + str(elements_b[2])
            if id_a == id_b:
                output_name = str(id_b) + "_merge.laz"

                run += 1
                print("run")
                print(run)

                input_file = os.path.join(low_veg_folder, f)
                merge_file = os.path.join(all_points_folder, l)
```

```
            print("input")
            print(input_file)
            print("merge")
            print(merge_file)
            print("output")
            print(merge_folder)

            todo = lasTools_version + r"\lasmerge64 -i {} {} -o {} -odir {}".format(input_file, merge_file,
                                                                                    output_name, merge_folder)
            # have to be -o !

            p = subprocess.Popen(todo, shell=True)
            p.wait()

# LasDublicate delete all duplicate points
# points from file, which are added first into the merge are kept (>changed classification of low green vegetation)
input_name = str(input_n) + "*_merge.laz"
input_file = os.path.join(merge_folder, input_name)

todo = lasTools_version + r"\lasduplicate64 -i {} -odix _dub -olaz -odir {} -cores {}".format(input_file,
                                                                                              working_dir, cores)

p = subprocess.Popen(todo, shell=True)
p.wait()

# drop class 3 (all vegetation points with a higher difference than defined threshold
input_name = str(input_n) + "*_merge_dub.laz"
input_file = os.path.join(working_dir, input_name)

todo = lasTools_version + r"\las2las -i {} -olaz -odix _ground -keep_class 2 -cores {}".format(input_file,
                                                                                               cores)  # -drop_class 3
p = subprocess.Popen(todo, shell=True)
p.wait()


# set classification to 1
input_name = str(input_n) + "*_merge_dub_ground.laz"
input_file = os.path.join(working_dir, input_name)

todo = lasTools_version + r"\las2las64 -i {} -set_classification 1 -odix _c -olaz -cores {}".format(input_file,
                                                                                                    cores)

p = subprocess.Popen(todo, shell=True)
p.wait()

# first ground classification with setp 1
input_name = str(input_n) + "*_merge_dub_ground_c.laz"
input_file = os.path.join(working_dir, input_name)

todo = lasTools_version + r"\lasground_new64 -i {} -step 1 -odix _lgn -olaz -cores {}".format(input_file, cores)

p = subprocess.Popen(todo, shell=True)
p.wait()

print("first ground classfication done" + str(datetime.now() - start_time))

# make thin ground from the classfied ground
input_name = str(input_n) + "*_merge_dub_ground_c_lgn.laz"  # las ground ausgelassen!
input_file = os.path.join(working_dir, input_name)


todo = lasTools_version + r"\lasthin64 -i {} -odix _thin -olaz -step 1 -lowest -keep_class 2 -cores {}".format(
    input_file, output_file, cores)

p = subprocess.Popen(todo, shell=True)
p.wait()

# second ground classification with step 1
input_name = str(input_n) + "*_merge_dub_ground_c_lgn_thin.laz"
input_file = os.path.join(working_dir, input_name)

todo = lasTools_version + r"\lasground_new64 -i {} -step 6 -odix _lgn -olaz -cores {}".format(input_file, cores)

p = subprocess.Popen(todo, shell=True)
p.wait()

# keep class 2
input_name = str(input_n) + "*_merge_dub_ground_c_lgn_thin_lgn.laz"
input_file = os.path.join(working_dir, input_name)
```

```
302
303     todo = lasTools_version + r"\las2las64 -i {} -keep_class 2 -odix _c -olaz -cores {}".format(input_file, cores)
304
305     p = subprocess.Popen(todo, shell=True)
306     p.wait()
307
308     print("2nd ground classfication done" + str(datetime.now() - start_time))
309
310     # remove noise in thinned 2nd ground classsified points
311     input_name = str(input_n) + "*_merge_dub_ground_c_lgn_thin_lgn_c.laz"
312     input_file = os.path.join(working_dir, input_name)
313
314     todo = lasTools_version + r"\lasnoise -i {} -step 1 -isolated 2 -remove_noise -odix _n -olaz -cores {}".format(input_file,
        cores)
315
316     p = subprocess.Popen(todo, shell=True)
317     p.wait()
318
319     # height difference between 2nd and reference height model
320     # if no points below 6 m are found script crashes
321     input_name = str(input_n) + "*_merge_dub_ground_c_lgn_thin_lgn_c_n.laz"
322     input_file = os.path.join(working_dir, input_name)
323
324     ground_points_file = os.path.join(working_dir, ground_points_name)
325
326     todo = lasTools_version + r"\lasheight -i {} -odix _h -olaz -ground_points {} -do_not_store_in_user_data -keep_below 6 -
       cores {}".format(
327         input_file, ground_points_file,
328         cores)
329
330     p = subprocess.Popen(todo, shell=True)
331     p.wait()
332
333
334
335     # merge thinned ground points
336     input_name = str(input_n) + "*_merge_dub_ground_c_lgn_thin_lgn_c_n_h.laz"
337     input_file = os.path.join(working_dir, input_name)
338
339     output_name = str(input_n) + "_merged_thin_lgn"
340
341     todo = lasTools_version + r"\lasmerge64 -i {} -olaz -drop_withheld -o {} -odir {}".format(
342         input_file, output_name, working_dir)
343
344     p = subprocess.Popen(todo, shell=True)
345     p.wait()
346
347     # with las hight compute difference from 2nd thin ground points  and 1st ground points
348     # everything which is higher than 1 m above the twice classified ground is classified as class 7
349     # for the further classification of the ground, only class 2 is kept
350     input_name = str(input_n) + "*_merge_dub_ground_c_lgn.laz"
351     input_file = os.path.join(working_dir, input_name)
352
353     ground_points_name = str(input_n) + "*_merged_thin_lgn.laz"
354     ground_points_file = os.path.join(working_dir, ground_points_name)
355
356     todo = lasTools_version + r"\lasheight -i {} -odix _reclass -olaz -ground_points {} -do_not_store_in_user_data -
       classify_above 1 7 -cores {}".format(
357         input_file, ground_points_file,
358         cores)
359
360     p = subprocess.Popen(todo, shell=True)
361     p.wait()
362
363     # LasThin to make the point cloud less heavy
364     input_name = str(input_n) + "*_merge_dub_ground_c_lgn_reclass.laz"
365     input_file = os.path.join(working_dir, input_name)
366
367     todo = lasTools_version + r"\lasthin64 -i {} -odix _thin -olaz -keep_class 2 -step 0.1 -cores {}".format(
368         input_file, cores)
369
370     p = subprocess.Popen(todo, shell=True)
371     p.wait()
372
373     input_name = str(input_n) + "*_merge_dub_ground_c_lgn_reclass_thin.laz"
374     input_file = os.path.join(working_dir, input_name)
375
376     output_name = str(input_n) + "_merged"
377
378     todo = lasTools_version + r"\lasmerge64 -i {} -olaz -drop_withheld -o {} -odir {}".format(
```

```
        input_file, output_name, working_dir)

    p = subprocess.Popen(todo, shell=True)
    p.wait()

    # generate dtm and hillshade
    input_name = str(input_n) + "_merged.laz"
    input_file = os.path.join(working_dir, input_name)

    output_name = str(input_n) + "_dem.tif"
    output_file = os.path.join(working_dir, output_name)

    todo = lasTools_version + r"\blast2dem -i {} -o {} -elevation -step 0.1 -merged -drop_withheld".format(input_file,
                                                                                                           output_file)

    p = subprocess.Popen(todo, shell=True)
    p.wait()

    output_name = str(input_n) + "_hillshade.tif"
    output_file = os.path.join(working_dir, output_name)

    todo = lasTools_version + r"\blast2dem -i {} -o {} -hillshade -step 0.1 -merged -drop_withheld".format(input_file,
                                                                                                           output_file)

    p = subprocess.Popen(todo, shell=True)
    p.wait()




############################### RUN Script Vegetation Classification ###################################
# folder
vegetation_classification_start_dir = r"Z:\Debris_Flow_Project\_Python_Data"
vegetation_classification_name = "fraschmardin2019v5"
vegetation_classification_dir = os.path.join(vegetation_classification_start_dir, vegetation_classification_name)

# run function
las2las_vegetation_index("fraschmardin19comb", "merged_lidar_ground_v2.las", vegetation_classification_dir)
```

B.2. Terrasolid Classification Routine

This commands are used to classify ground points with Terrasolid.

```
FnScanClassifyClass("Any",1,0)
FnScanSort(12)
FnScanCutLowRel("Any",1,10,10,0.500,40)
FnScanClassifyLow(1,7,15,0.40,2.00,0)
FnScanClassifyIsolated("1",7,15,"1",3.00,0)

FnScanDeleteCass(7,0)
FnScanDeleteCass(10,0)
FnScanClassifySurface("1",8,0.050,1,0)
FnScanSmoothenYyz("8",0.200,"",0,0)
FnScanThinPoints("8",9,2,0,0.070,0.070,0)
FnScanClassifyLow(9,7,10,0.40,1.50,0)
FnScanClassifyIsolated("9",7,15,"8",2.00,0)

FnScanDeleteCass(1,0)
FnScanDeleteCass(7,0)
FnScanDeleteCass(8,0)
FnScanDistVbdvi("9",0.050)
FnScanSmoothenDist("9","",0.150,10,0)
FnScanClassifyDistance("9",17,0.000,9999.000,0)
FnScanClassifyGround("9",2,"2",1,60.0,75.00,15.00,1.00,1,8.0,0,2.0,0,0
```

B.3. eCognition Input

This script generates input files for eCognition and the subsequent Python script B.4 based on basic high resolution DTM data.

```
#ESRI Import
import arcpy
import time
import random
import time
import random
import arcpy.management as DM
import arcpy.cartography as CA
import os
import math
from arcpy.sa import *
from arcpy import env
from datetime import datetime
from arcpy.ia import *

#NUMPY Import
import matplotlib.pyplot as plt
import numpy
from scipy.interpolate import UnivariateSpline
import pandas as pd
import csv
import numpy as np

def process_data_fun(extent, asc_n1_file, asc_n2_file, dtm_file, tmp_dir):

    print("hallo")
    # global count, counts the amount of script runs
    global g_c
    g_c += 1  # global count +1
    arcpy.AddMessage("run: " + str(g_c))

    # ArcGIS variables, used in the whole script
    env.overwriteOutput = True
    arcpy.CheckOutExtension("spatial")
    arcpy.CheckOutExtension("ImageAnalyst")
    arcpy.CheckOutExtension("ArcGIS 3D Analyst")
    env.extent = extent
    env.pyramid = None
    env.parallelProcessingFactor = "100%"

    # start_time used to estimate the calculation time
    global start_time

    start_time = datetime.now()

    transect(dtm_file, tmp_dir)

    derivations(dtm_file, asc_n1_file, asc_n2_file, tmp_dir)









def torrent_extraction(GW, dtm_resample_file, tmp_dir):

    #smooth dem 3 times
    smooth_dtm_file = os.path.join(tmp_dir, "smooth_dtm.tif")
    neighborhood = NbrRectangle(3, 3, "CELL")
    smooth_dtm_con = FocalStatistics(dtm_resample_file, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file)

    smooth_dtm_file_2 = os.path.join(tmp_dir, "smooth_dtm_2.tif")
    neighborhood = NbrRectangle(3, 3, "CELL")
    smooth_dtm_con = FocalStatistics(dtm_resample_file, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_2)

    smooth_dtm_file_3 = os.path.join(tmp_dir, "smooth_dtm_3.tif")
    neighborhood = NbrRectangle(3, 3, "CELL")
    smooth_dtm_con = FocalStatistics(dtm_resample_file, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_3)

    # Fill up the sinks
    arcpy.AddMessage("Filling up sinks...")
```

```
    fill_file = os.path.join(tmp_dir, "hydro_sink.tif")
    out_fill = Fill(smooth_dtm_file_3)
    out_fill.save(fill_file)

    # FlowDirection
    arcpy.AddMessage("Calculating Flow Direction...")
    flowdirection_file = os.path.join(tmp_dir, "hydro_flowdirection.tif")
    drop_raster_file = os.path.join(tmp_dir, "hydro_drop_raster.tif")
    out_flowdirection = FlowDirection(fill_file, "NORMAL", drop_raster_file, "D8")
    out_flowdirection.save(flowdirection_file)


    # FlowAccumulation
    arcpy.AddMessage("Calculating Flow Accumulation...")
    flowaccumulation_file = os.path.join(tmp_dir, "hydro_flowaccumulation.tif")
    out_flowaccumulation = FlowAccumulation(flowdirection_file)
    out_flowaccumulation.save(flowaccumulation_file)

    # Extract torrent, based on GW
    arcpy.AddMessage("Extracting Torrent...")
    torrent_file = os.path.join(tmp_dir, "torrent.tif")
    out_torrent = Con(Raster(flowaccumulation_file) > GW, 1)
    out_torrent.save(torrent_file)

    # Feature to Torrent
    feature_torrent_file = os.path.join(tmp_dir, "torrent.shp")
    StreamToFeature(torrent_file, flowdirection_file, feature_torrent_file, "SIMPLIFY")

    # Smooth Torrent with PEAK 10 Meters
    smooth_torrent_file = os.path.join(tmp_dir, "torrent_smooth.shp")
    arcpy.cartography.SmoothLine(feature_torrent_file, smooth_torrent_file, "PAEK", "10 Meters")

    arcpy.AddMessage("torrent ok")
    arcpy.AddMessage(str(datetime.now() - start_time))

    return smooth_torrent_file


#wieso ist bei aggregate polygons eine linie als erstes file? dies ergibt fehler beim zusammenfueren

def transect(dtm_org_file, tmp_dir):

    # Resample and Rename DTM File
    arcpy.AddMessage("Preparing DTM File...")
    dtm_resample_file = os.path.join(tmp_dir, "dtm.tif")
    arcpy.Resample_management(dtm_org_file, dtm_resample_file, "0.25", "CUBIC")

    # run torrent_extraction function, with defined GW
    torrent = torrent_extraction(3000000, dtm_org_file, tmp_dir)

    # create transects
    arcpy.AddMessage("creating transects...")
    transect_file = os.path.join(tmp_dir, "transect.shp")
    arcpy.management.GenerateTransectsAlongLines(torrent, transect_file, "0.5 Meters", "100 Meters")

    # create transect points
    arcpy.AddMessage("creating points for each transect...")
    transect_points_file = os.path.join(tmp_dir, "transect_points.shp")
    arcpy.FeatureToPoint_management(transect_file, transect_points_file)

    #add dtm file
    ExtractMultiValuesToPoints(transect_points_file,
                               [[dtm_resample_file]], "BILINEAR")


    # join field
    arcpy.JoinField_management(transect_file, "FID", transect_points_file, "ORIG_FID",
                               ["dtm"])

    # join field
    arcpy.JoinField_management(transect_points_file, "ORIG_FID", transect_file, "FID",
                               ["ORIG_FID"])

    # calculates the amount of sections
    section_stat_file = os.path.join(tmp_dir, "section_stat.csv")
    arcpy.Statistics_analysis(transect_file, section_stat_file, [["ORIG_FID", "MAX"]])

    with arcpy.da.SearchCursor(section_stat_file, ["MAX_ORIG_FID"]) as cursor:
        for row in cursor:
            max_count_section = row[0]
```

```
            print(max_count_section)

    count_section = 0


    # create empty feature class
    arcpy.management.CreateFeatureclass(tmp_dir, "merged_poly.shp", "POLYGON", "", "", "", transect_file)
    merged_poly_file = os.path.join(tmp_dir, "merged_poly.shp")

    # iterate over each section
    while count_section <= max_count_section:

        arcpy.AddMessage("section {}...".format(count_section))

        # select transect file
        select_transect_file = os.path.join(tmp_dir, "select_transect_{}.shp".format(count_section))
        arcpy.Select_analysis(transect_file, select_transect_file, "ORIG_FID={}".format(count_section))

        # select transect points
        select_transect_points_file = os.path.join(tmp_dir, "select_transect_points_{}.shp".format(count_section))
        arcpy.Select_analysis(transect_points_file, select_transect_points_file, "ORIG_FID_1={}".format(count_section))

        # find lowest transect point; for pour point
        pour_stat_file = os.path.join(tmp_dir, "pour_stat_{}.csv".format(count_section))
        arcpy.Statistics_analysis(select_transect_points_file, pour_stat_file, [["ORIG_FID", "MAX"]])

        # pour point is 2.5m heigher than lowest point
        with arcpy.da.SearchCursor(pour_stat_file, ["MAX_ORIG_FID"]) as cursor:
            for row in cursor:
                lowest_point = row[0]
                print("lowest_point")
                print(lowest_point)
                pour_point = lowest_point - 5

        # select pour point
        pour_point_select_file = os.path.join(tmp_dir, "pour_point_select_{}.shp".format(count_section))
        arcpy.Select_analysis(select_transect_points_file, pour_point_select_file, "ORIG_FID={}".format(pour_point))

        # link to flowdirection and accumulation file
        flowdirection_file = os.path.join(tmp_dir, "hydro_flowdirection.tif")
        flowaccumulation_file = os.path.join(tmp_dir, "hydro_flowaccumulation.tif")

        # snap real pour point within 1 m distance
        pour_point_snap_file = os.path.join(tmp_dir, "pour_point_snap_{}.tif".format(count_section))
        pour_point_snap_con = SnapPourPoint(pour_point_select_file, flowaccumulation_file, 1) #dieser wert setzen wir tiefer
    um zuweisung zum falschen catchment zu vermeiden
        pour_point_snap_con.save(pour_point_snap_file)

        # watershed
        watershed_file = os.path.join(tmp_dir, "watershed_{}.tif".format(count_section))
        watershed_con = Watershed(flowdirection_file, pour_point_snap_file)
        watershed_con.save(watershed_file)

        # polygon watershed
        watershed_poly_file = os.path.join(tmp_dir, "watershed_{}.shp".format(count_section))
        arcpy.RasterToPolygon_conversion(watershed_file, watershed_poly_file, "NO_SIMPLIFY")

        watershed_poly_buffer_file = os.path.join(tmp_dir, "watershed_buffer_{}.shp".format(count_section))
        arcpy.Buffer_analysis(watershed_poly_file, watershed_poly_buffer_file, "0.3 Meters", "", "",  "", "", "GEODESIC")

        # clip transects to the extent of the watershed polygon
        clip_transect_file = os.path.join(tmp_dir, "clip_transect_{}.shp".format(count_section))
        arcpy.Clip_analysis(select_transect_file, watershed_poly_buffer_file, clip_transect_file)


        # polyline to raster
        torrent_height_file = os.path.join(tmp_dir, "torrent_height_{}.tif".format(count_section))
        arcpy.PolylineToRaster_conversion(clip_transect_file, "dtm", torrent_height_file,
                                          "MAXIMUM_LENGTH", "dtm", 1)

        torrent_height_resample_file = os.path.join(tmp_dir, "torrent_height_resample_{}.tif".format(count_section))
        arcpy.Resample_management(torrent_height_file, torrent_height_resample_file, "0.25", "CUBIC")

        # difference of two rasters
        norm_torrent_height_file = os.path.join(tmp_dir, "norm_torrent_height_{}.tif".format(count_section))
        norm_torrent_height_con = Raster(dtm_resample_file) - Raster(torrent_height_resample_file)
        norm_torrent_height_con.save(norm_torrent_height_file)

        # binary raster, with threshold at 20 cm
        bin_file = os.path.join(tmp_dir, "bin_{}.tif".format(count_section))
```

```
        bin_con = SetNull(Raster(norm_torrent_height_file) < -1, SetNull(Raster(norm_torrent_height_file) > 0.2, 1))
        bin_con.save(bin_file)

        # convert binary raster to polygon
        poly_file = os.path.join(tmp_dir, "poly_{}.shp".format(count_section))
        arcpy.RasterToPolygon_conversion(bin_file, poly_file, "NO_SIMPLIFY")

        # append poly_file to merged_poly_file
        arcpy.management.Append(poly_file, merged_poly_file, "NO_TEST")


        count_section += 1

    # link to file calculated further above
    merged_poly_file = os.path.join(tmp_dir, "merged_poly.shp")

    # smooth merged_poly_file with 1 m effective area
    smooth_torrent_bed_file = os.path.join(tmp_dir, "smooth_torrent_bed.shp")
    CA.SimplifyPolygon(merged_poly_file, smooth_torrent_bed_file, "EFFECTIVE_AREA", 1)

    # link to file calculated further above
    smooth_torrent_file = os.path.join(tmp_dir, "torrent_smooth.shp")

    # for the select operation a feature layer is needed
    # make a layer from the feature class
    arcpy.management.MakeFeatureLayer(smooth_torrent_bed_file, "merged_poly_feature")

    # select all polygons, which intersect with the smooth torrent file
    arcpy.management.SelectLayerByLocation("merged_poly_feature", "INTERSECT", smooth_torrent_file, None,
                                           "NEW_SELECTION", "NOT_INVERT")

    # write the selected features to a new featureclass
    merged_poly_intersect_file = os.path.join(tmp_dir, "aggregate_torrent_bed.shp")
    arcpy.CopyFeatures_management("merged_poly_feature", merged_poly_intersect_file, '', None, None, None)

    # agregate all selected files within 2 m distance
    aggregate_torrent_bed_file = os.path.join(tmp_dir, "aggregate_torrent_bed_fin.shp")
    arcpy.cartography.AggregatePolygons(merged_poly_intersect_file, aggregate_torrent_bed_file, 2)



def derivations(dtm_org_file, remap_n1_file, remap_n2_file, tmp_dir):


    # Resample and Rename DTM File
    arcpy.AddMessage("Preparing DTM File...")
    dtm_resample_file = os.path.join(tmp_dir, "dtm_05.tif")
    arcpy.Resample_management(dtm_org_file, dtm_resample_file, "0.25", "CUBIC")

    # Focal Statistic: smooth dem
    smooth_dtm_file_1 = os.path.join(tmp_dir, "smooth_dtm_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(dtm_resample_file, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_1)

    smooth_dtm_file_2 = os.path.join(tmp_dir, "smooth_dtm_2_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_1, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_2)

    smooth_dtm_file_3 = os.path.join(tmp_dir, "smooth_dtm_3_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_2, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_3)

    smooth_dtm_file_4 = os.path.join(tmp_dir, "smooth_dtm_4_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_3, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_4)

    smooth_dtm_file_5 = os.path.join(tmp_dir, "smooth_dtm_5_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_4, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_5)

    smooth_dtm_file_6 = os.path.join(tmp_dir, "smooth_dtm_6_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_5, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_6)

```

```
    smooth_dtm_file_7 = os.path.join(tmp_dir, "smooth_dtm_7_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_6, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_7)

    smooth_dtm_file_8 = os.path.join(tmp_dir, "smooth_dtm_8_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_7, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_8)

    smooth_dtm_file_9 = os.path.join(tmp_dir, "smooth_dtm_9_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_8, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_9)

    smooth_dtm_file_10 = os.path.join(tmp_dir, "smooth_dtm_10_05.tif")
    neighborhood = NbrRectangle(5, 5, "CELL")
    smooth_dtm_con = FocalStatistics(smooth_dtm_file_9, neighborhood)
    smooth_dtm_con.save(smooth_dtm_file_10)



    # fill up the sinks
    arcpy.AddMessage("Filling up sinks...")
    fill_file = os.path.join(tmp_dir, "hydro_sink.tif")
    out_fill = Fill(smooth_dtm_file_10)
    out_fill.save(fill_file)

    # slope
    slope_file = os.path.join(tmp_dir, "slope_05.tif")
    out_slope = Slope(fill_file)
    out_slope.save(slope_file)

    # aspect
    aspect_file = os.path.join(tmp_dir, "aspect_05.tif")
    out_aspect = Aspect(fill_file)
    out_aspect.save(aspect_file)

    # curvature (not neeeded)
    curv_std_file = os.path.join(tmp_dir, "curv_std_05.tif")
    curv_prof_file = os.path.join(tmp_dir, "curv_prof_05.tif")
    curv_plan_file = os.path.join(tmp_dir, "curv_plan_05.tif")
    arcpy.Curvature_3d(fill_file, curv_std_file, 1, curv_prof_file, curv_plan_file)


    # reclassify aspect n1
    aspect_n1_file = os.path.join(tmp_dir, "aspect_n1.tif")
    outreclass1 = ReclassByASCIIFile(aspect_file, remap_n1_file)
    outreclass1.save(aspect_n1_file)

    # reclassify aspect n2
    aspect_n2_file = os.path.join(tmp_dir, "aspect_n2.tif")
    outreclass2 = ReclassByASCIIFile(aspect_file, remap_n2_file)
    outreclass2.save(aspect_n2_file)


def main():

    #input = arcpy.GetParameterAsText(0)
    #output = arcpy.GetParameterAsText(1)
    #...

    # Extent
    # defines within CH1903+ the outer edges of the calculation domain

    # Arelen
    extent = "2782000 1189000 2785000 1191000"



    ######################################

    # in the following section all sources are listed.
    # im folgenden Code Abschnitt sind alle Inputdatensaete aufgefuert und verlinkt.

    # folder start_dir defines the main folder
    start_dir = r"D:\Debris_Flow_Project\eCognition\norm_torrent_height\_Data\Arelen"


    # DTM with 0.5 m x 0.5 m cell size
```

```
    dtm_dir = "DTM"
    dtm_name = "arelen_terra_2021_large_20211220.tif"
    dtm_file = os.path.join(os.path.join(start_dir, dtm_dir), dtm_name)

    # folder where reclassifcation matrix is stored
    asc_dir = "ASC"
    asc_n1_name = "remap_2_8125_asci_n1.txt"
    asc_n1_file = os.path.join(os.path.join(start_dir, asc_dir), asc_n1_name)

    asc_n2_name = "remap_2_8125_asci_n2.txt"
    asc_n2_file = os.path.join(os.path.join(start_dir, asc_dir), asc_n2_name)


    # output folder
    tmp_dir = os.path.join(start_dir, "_Calculations_v3002_2021_large_terra_2m")

    # with env.snapRaster all raster files are aligned properly
    env.snapRaster = dtm_file


    # calls the calculation
    process_data_fun(extent, asc_n1_file, asc_n2_file, dtm_file, tmp_dir)


if __name__ == "__main__":

    g_c = 0         # count
    print("start")
    main()
    print("Done!")
```

B.4. Torrent Values Calculation

This script calculates the torrential properties based on eCognition output.

```
#ESRI Import
import arcpy
import time
import random
import arcpy.management as DM
import arcpy.cartography as CA
import os
import math
from arcpy.sa import *
from arcpy import env
from datetime import datetime
from arcpy.ia import *


#NUMPY Import
import matplotlib.pyplot as plt
import numpy
from scipy.interpolate import UnivariateSpline
import pandas as pd
import csv
import numpy as np


def process_data_fun(extent, dtm_file, ecog_file, transect_file, torrent_bed_file , tmp_dir):

    # global count, counts amount of runs
    global g_c
    g_c += 1  # global count +1
    arcpy.AddMessage("run: " + str(g_c))

    # ArcGIS variables, which are used within the whole script
    env.overwriteOutput = True
    arcpy.CheckOutExtension("spatial")
    arcpy.CheckOutExtension("ImageAnalyst")
    arcpy.CheckOutExtension("ArcGIS 3D Analyst")
    env.extent = extent
    env.pyramid = None
    env.parallelProcessingFactor = "100%"

    # start_time calculation
    global start_time
```

```
42
43      start_time = datetime.now()
44
45      # create geodatabase
46      arcpy.CreateFileGDB_management(tmp_dir, "torrent_values.gdb")
47      torrent_values_gdb_lu = os.path.join(tmp_dir, "torrent_values.gdb")
48
49      # will save intermediate results in memory
50      torrent_values_gdb = "in_memory"
51
52      calculate_values(ecog_file, dtm_file, transect_file, torrent_bed_file, tmp_dir, torrent_values_gdb, torrent_values_gdb_lu)
53
54
55
56  def calculate_values(ecog_file, dtm_file, transect_file, torrent_bed_file, tmp_dir, torrent_values_gdb, torrent_values_gdb_lu)
        :
57
58      # counts amount of transect
59      count_id_transect = 0
60
61      # negative buffer of torrent_bed
62      #(>torrent bed is calculated to wide, with this reduction it is closer to reality and there will be always values for the
          embankment)
63      neg_buffer_torrent_bed_file = os.path.join(tmp_dir, "clip_neg_buffer_torrent_bed.shp")
64      arcpy.Buffer_analysis(torrent_bed_file, neg_buffer_torrent_bed_file, "-0.2 Meters", "FULL", "", "ALL", "", "PLANAR")
65
66      # positive buffer of torrent_bed (>cuts wrongly classified embankment polygon to a maximal distance 20 m to the torrent
          bed)
67      pos_buffer_torrent_bed_file = os.path.join(tmp_dir, "clip_pos_buffer_torrent_bed.shp")
68      arcpy.Buffer_analysis(torrent_bed_file, pos_buffer_torrent_bed_file, "20 Meters", "FULL", "", "ALL", "", "PLANAR")
69
70      # small positive buffer of torrent_bed (>needed for aggregation)
71      pos_1m_buffer_torrent_bed_file = os.path.join(tmp_dir, "clip_pos_1m_buffer_torrent_bed.shp")
72      arcpy.Buffer_analysis(torrent_bed_file, pos_1m_buffer_torrent_bed_file, "1 Meters", "FULL", "", "ALL", "", "PLANAR")
73
74      # simplify ecognition output
75      ecog_simplify_file = os.path.join(tmp_dir, "ecog_simplify.shp")
76      CA.SimplifyPolygon(ecog_file, ecog_simplify_file, "EFFECTIVE_AREA", 1)
77
78      # clip simplified ecog by 20 m
79      clip20m_ecog_file = os.path.join(tmp_dir, "clip20m_ecog.shp")
80      arcpy.Clip_analysis(ecog_simplify_file, pos_buffer_torrent_bed_file, clip20m_ecog_file)
81
82      # merge cliped and simplified eCog output with 1m positive torrent_bed buffer
83      merged_ecog_file = os.path.join(tmp_dir, "merged_ecog.shp")
84      arcpy.Merge_management([clip20m_ecog_file, pos_1m_buffer_torrent_bed_file], merged_ecog_file)
85
86
87      # aggregate ecognition output
88      ecog_aggregate_file = os.path.join(tmp_dir, "ecog_aggregate.shp")
89      arcpy.cartography.AggregatePolygons(merged_ecog_file, ecog_aggregate_file, 1)
90
91
92      # erase torrent bed from ecog output
93      embankment_multi_file = os.path.join(tmp_dir, "embankment_multi.shp")
94      arcpy.analysis.Erase(ecog_aggregate_file, neg_buffer_torrent_bed_file, embankment_multi_file)
95
96
97
98
99      # calculates the amount of sections
100     section_stat_file = os.path.join(tmp_dir, "section_stat.csv")
101     arcpy.Statistics_analysis(transect_file, section_stat_file, [["ORIG_FID", "MAX"]])
102
103     with arcpy.da.SearchCursor(section_stat_file, ["MAX_ORIG_FID"]) as cursor:
104         for row in cursor:
105             max_count_section = row[0]
106             print(max_count_section)
107
108
109     # starts at count section 0
110     count_section = 0
111
112
113
114
115
116     # create an empty list for dfs
117     small_dfs = []
118
```

```
    # create empty feature class, in which the merged transects will be saved
    arcpy.management.CreateFeatureclass(tmp_dir, "merged_transects.shp", "POLYLINE", "", "", "", transect_file)
    merged_transects_file = os.path.join(tmp_dir, "merged_transects.shp")

    # create array to save torrent properties
    v_embakment_left = [[None, None, None, None, None]]
    v_embakment_right = [[None, None, None, None, None]]
    v_torrent_bed = [[None, None, None, None, None, None, None, None, None, None, None]]

    # Iteration over all sections
    while count_section <= max_count_section:


        arcpy.AddMessage("section {}...".format(count_section))

        # clip (from last transect iteration; clipped by the watershed polygon)
        clip_transect_file = os.path.join(tmp_dir, "clip_transect_{}.shp".format(count_section))

        # clip feature transects by simplified ecognition output
        clip_ecog_transect_file = os.path.join(tmp_dir, "clip_ecog_trans_{}.shp".format(count_section))
        arcpy.Clip_analysis(clip_transect_file, merged_ecog_file, clip_ecog_transect_file)

        # append transect (which are clipped by eCog Output) to merged_transects_file
        arcpy.management.Append(clip_ecog_transect_file, merged_transects_file, "NO_TEST")


        # creating points along transect line with 25 cm distance
        points_one_section_file = os.path.join(torrent_values_gdb,
                                               "points_transect_{}".format(count_section))
        arcpy.management.GeneratePointsAlongLines(clip_ecog_transect_file, points_one_section_file, "DISTANCE", 0.25)

        # calculate DTM value for each point
        ExtractMultiValuesToPoints(points_one_section_file,
                                   [[dtm_file]], "BILINEAR")


        # calculates the amount of clipped transects in a section and its median transect
        transect_stat_file = os.path.join(torrent_values_gdb, "transect_stat_{}".format(count_section))
        arcpy.Statistics_analysis(points_one_section_file, transect_stat_file, [["ORIG_FID", "MAX"], ["ORIG_FID", "MEDIAN"]])

        with arcpy.da.SearchCursor(transect_stat_file, ["MAX_ORIG_FID"]) as cursor:
            for row in cursor:
                max_count_transect = row[0]

        with arcpy.da.SearchCursor(transect_stat_file, ["MEDIAN_ORIG_FID"]) as cursor:
            for row in cursor:
                median_count_transect = row[0]

        # set count_transect to 0
        count_transect = 0


        # select median transect in section
        select_median_transect_file = os.path.join(torrent_values_gdb, "select_transect_points_{}".format(count_section))
        arcpy.Select_analysis(points_one_section_file, select_median_transect_file, "ORIG_FID={}".format(median_count_transect
    ))

        # calculates first and last point in transect
        # a buffer around the first and last transect, will devide the embankment left and right side
        embankment_stat_file = os.path.join(torrent_values_gdb, "embankment_stat_{}".format(count_section))
        arcpy.Statistics_analysis(select_median_transect_file, embankment_stat_file, [["OID", "MIN"], ["OID", "MAX"]])

        with arcpy.da.SearchCursor(embankment_stat_file, ["MIN_OID"]) as cursor:
            for row in cursor:
                OID_embankment_right = row[0]

        with arcpy.da.SearchCursor(embankment_stat_file, ["MAX_OID"]) as cursor:
            for row in cursor:
                OID_embankment_left = row[0]

        # select point to intersect with embankment left
        select_embankment_left_point_file = os.path.join(tmp_dir, "select_embankment_left_point_{}.shp".format(count_section))
        arcpy.Select_analysis(select_median_transect_file, select_embankment_left_point_file, "OID={}".format(
    OID_embankment_left))

        # select point to intersect with embankment right
        select_embankment_right_point_file = os.path.join(tmp_dir, "select_embankment_right_point_{}.shp".format(count_section
    ))
        arcpy.Select_analysis(select_median_transect_file, select_embankment_right_point_file, "OID={}".format(
    OID_embankment_right))
```

```
195
196
197         #select first transect
198         select_first_transect_file = os.path.join(tmp_dir, "select_first_transect_{}.shp".format(count_section))
199         arcpy.Select_analysis(clip_transect_file, select_first_transect_file, "FID=0")
200
201         #select last transect
202         select_last_transect_file = os.path.join(tmp_dir, "select_last_transect_{}.shp".format(count_section))
203         arcpy.Select_analysis(clip_transect_file, select_last_transect_file, "FID={}".format(max_count_transect))
204
205         # buffer first transect
206         buffer_first_transect_file = os.path.join(tmp_dir, "buffer_first_transect_{}.shp".format(count_section))
207         arcpy.Buffer_analysis(select_first_transect_file, buffer_first_transect_file, "0.5 Meters", "FULL", "", "ALL", "",
208                               "PLANAR")
209
210         # buffer last transect
211         buffer_last_transect_file = os.path.join(tmp_dir, "buffer_last_transect_{}.shp".format(count_section))
212         arcpy.Buffer_analysis(select_last_transect_file, buffer_last_transect_file, "0.5 Meters", "FULL", "", "ALL", "",
213                               "PLANAR")
214
215         # erase first and last transect line of each section
216         embankment_erase1_file = os.path.join(tmp_dir, "embankment_erase1_{}.shp".format(count_section))
217         arcpy.analysis.Erase(embankment_multi_file, buffer_first_transect_file, embankment_erase1_file)
218         embankment_erase2_file = os.path.join(tmp_dir, "embankment_erase2_{}.shp".format(count_section))
219         arcpy.analysis.Erase(embankment_erase1_file, buffer_last_transect_file, embankment_erase2_file)
220
221         # clip embankment_file by watershed
222         watershed_file = os.path.join(tmp_dir, "watershed_buffer_{}.shp".format(count_section))
223         embankment_watershed_file = os.path.join(tmp_dir, "select_first_transect_{}.shp".format(count_section))
224         arcpy.analysis.PairwiseClip(embankment_erase2_file, watershed_file, embankment_watershed_file)
225
226
227         # embankment multi to single file
228         embankment_single_file = os.path.join(tmp_dir, "embankment_single_{}.shp".format(count_section))
229         arcpy.management.MultipartToSinglepart(embankment_watershed_file, embankment_single_file)
230
231         # make a layer from the feature class (>needed for select by location)
232         arcpy.management.MakeFeatureLayer(embankment_single_file, "embankment_single_left_{}".format(count_section))
233
234         # select polygon, which intersect embankment left point
235         arcpy.management.SelectLayerByLocation("embankment_single_left_{}".format(count_section), "WITHIN_A_DISTANCE",
236             select_embankment_left_point_file, "1 Meters", "NEW_SELECTION", "NOT_INVERT")
237
238         # write the selected features to a new featureclass
239         embankment_left_file = os.path.join(torrent_values_gdb, "embankment_left")
240         arcpy.CopyFeatures_management("embankment_single_left_{}".format(count_section), embankment_left_file, '', None, None,
    None)
241
242
243         # make a layer from the feature class (>needed for select by location)
244         arcpy.management.MakeFeatureLayer(embankment_single_file, "embankment_single_right_{}".format(count_section))
245
246         # select polygon, which intersect embankment right point
247         arcpy.management.SelectLayerByLocation("embankment_single_right_{}".format(count_section), "WITHIN_A_DISTANCE",
248                                                select_embankment_right_point_file, "1 Meters",
249                                                "NEW_SELECTION", "NOT_INVERT")
250
251         # write the selected features to a new featureclass
252         embankment_right_file = os.path.join(torrent_values_gdb, "embankment_right")
253         arcpy.CopyFeatures_management("embankment_single_right_{}".format(count_section), embankment_right_file, '', None,
254                                       None, None)
255
256
257         # clip transect points by embakment_left
258         select_embakment_left_file = os.path.join(torrent_values_gdb, "select_embakment_left")
259         arcpy.analysis.PairwiseClip(points_one_section_file, embankment_left_file, select_embakment_left_file)
260
261         # clip transect points by embakment_right
262         select_embakment_right_file = os.path.join(torrent_values_gdb, "select_embakment_right")
263         arcpy.analysis.PairwiseClip(points_one_section_file, embankment_right_file, select_embakment_right_file)
264
265         # clip transect points by torrent_bed
266         select_torrent_bed_file = os.path.join(torrent_values_gdb, "torrent_bed")
267         arcpy.analysis.PairwiseClip(points_one_section_file, torrent_bed_file, select_torrent_bed_file)
268
269
270         # calculates points statistics for Talweg calculation (parameter not used in thesis)
271         points_stat_talweg_file = os.path.join(torrent_values_gdb,
272                                        "talweg_{}".format(count_section))
273         arcpy.Statistics_analysis(select_torrent_bed_file, points_stat_talweg_file,
```

```
                                [["dtm_1", "MAX"], ["dtm_1", "MIN"]])

        with arcpy.da.SearchCursor(points_stat_talweg_file, ["MAX_dtm_1"]) as cursor:
            for row in cursor:
                max_dtm_talweg = row[0]

        with arcpy.da.SearchCursor(points_stat_talweg_file, ["MIN_dtm_1"]) as cursor:
            for row in cursor:
                min_dtm_talweg = row[0]


        # calculate height diff Talweg
        height_diff_talweg = float(max_dtm_talweg) - float(min_dtm_talweg)



        # iterations over transect by transect
        while count_transect <= max_count_transect:

            arcpy.AddMessage("section {}... transect {}....".format(count_section, count_transect))

            #adding values to v_embakment_left with np.append (the return value from the embakment_left is a python array)
            v_embakment_left = np.append(v_embakment_left,
                                         embakment_left(select_embakment_left_file, count_section, count_transect,
    count_id_transect, tmp_dir,
                                                        torrent_values_gdb), axis=0)

            #adding values to v_embakment_right with np.append (the return value from the embakment_right is a python array)
            v_embakment_right = np.append(v_embakment_right,
                                         embakment_right(select_embakment_right_file, count_section, count_transect,
    count_id_transect, tmp_dir,
                                                        torrent_values_gdb), axis=0)

            #adding values to v_torrent_bed with np.append (the return value from the torrent_bed is a python array)
            v_torrent_bed = np.append(v_torrent_bed,
                                         torrent_bed(select_torrent_bed_file, count_section, count_transect, count_id_transect
    , max_count_transect,
                                            height_diff_talweg, min_dtm_talweg, max_dtm_talweg, tmp_dir, torrent_values_gdb),
    axis=0)


            arcpy.AddMessage("section {} transect {} done".format(count_section, count_transect))

            print(count_id_transect)

            count_transect += 1
            count_id_transect += 1

        count_section += 1


    # combine multiple pandas df
    df_embakment_left = pd.DataFrame(v_embakment_left,
                                     columns=["id_transect", "e_l_height", "e_l_width", "e_l_diagonal",
                                              "e_l_inclination"])
    df_embakment_right = pd.DataFrame(v_embakment_right,
                                      columns=["id_transect", "e_r_height", "e_r_width", "e_r_diagonal",
                                               "e_r_inclination"])
    df_torrent_bed = pd.DataFrame(v_torrent_bed,
                                  columns=["id_transect", "count_section", "t_mean_height", "t_std_height", "t_width",
                                           "t_diagonal",
                                           "t_inclination", "talweg_inclincation", "talweg_torrent_diff",
                                           "talweg_path_distance", "talweg_linear_distance"])

    # combine df
    df_comb = pd.merge(df_embakment_left, df_embakment_right, on='id_transect', how='left')
    df_comb2 = pd.merge(df_comb, df_torrent_bed, on='id_transect', how='left')

    large_df = df_comb2.iloc[1:, :]

    # calculate additional variabel volume_cs (crosssection area)
    large_df["volume_cs"] = ((large_df["e_l_height"] + large_df["e_r_height"]) / 2) * (
                large_df["t_width"] * (large_df["t_width"] + large_df["e_l_width"] + large_df["e_r_width"]) / 2)

    #rename column in pandas df
    large_df.rename(columns={"id_transect": "FID"}, inplace=True)

    large_df["FID"] = pd.to_numeric(large_df["FID"], errors="coerce", downcast="integer")


```

```
    # converts large_df to a .csv table
    table_comb = os.path.join(tmp_dir, "table_comb_{}.csv".format(count_section))
    large_df.to_csv(table_comb)

    #join csv table to merged_transects_file
    fields = ["count_section", "e_l_height", "e_l_width", "e_l_diagonal", "e_l_inclination", "e_r_height", "e_r_width", "
      e_r_diagonal",
            "e_r_inclination", "t_mean_height", "t_std_height", "t_width", "t_diagonal",
            "t_inclination", "talweg_inclincation", "talweg_torrent_diff", "talweg_path_distance",
            "talweg_linear_distance", "volume_cs"]
    arcpy.management.JoinField(merged_transects_file, "FID", table_comb, "FID", fields)



    print(large_df)

    arcpy.AddMessage("join sucessful")




def embakment_left(select_embakment_left_file, count_section, count_transect, count_id_transect, tmp_dir, torrent_values_gdb):

    # select one transect line
    select_transect_line_file = os.path.join(torrent_values_gdb,
                                             "select_transect_line_{}_{}".format(count_section, count_transect))
    arcpy.Select_analysis(select_embakment_left_file, select_transect_line_file, "ORIG_FID={}".format(count_transect))

    #count
    count_select = float(str(arcpy.GetCount_management(select_transect_line_file)))
    arcpy.AddMessage("transect {}-{} has {} records".format(count_section, count_transect, count_select))

    # if there are less than 2 points an empty np_array is generated
    if count_select < 2:
        list = [[count_id_transect, None, None, None, None]]

    else:
        # calculates points statistics
        points_stat_file = os.path.join(torrent_values_gdb, "embakment_left_{}_{}".format(count_section, count_transect))
        arcpy.Statistics_analysis(select_transect_line_file, points_stat_file, [["dtm_1", "MAX"], ["dtm_1", "MIN"], ["OID", "
      MAX"], ["OID", "MIN"]])

        with arcpy.da.SearchCursor(points_stat_file, ["MAX_dtm_1"]) as cursor:
            for row in cursor:
                highest_point_dtm = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MIN_dtm_1"]) as cursor:
            for row in cursor:
                lowest_point_dtm = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MAX_OID"]) as cursor:
            for row in cursor:
                highest_point_id = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MIN_OID"]) as cursor:
            for row in cursor:
                lowest_point_id = row[0]

        height = float(highest_point_dtm) - float(lowest_point_dtm)
        width = 0.25 * (float(highest_point_id) - float(lowest_point_id))
        diagonal = (height**2 + width**2)**0.5

        if diagonal > 0:
            radians = math.asin(height / diagonal)
            inclination = math.degrees(radians)
        else:
            inclination = 0

        list = [[count_id_transect, height, width, diagonal, inclination]]

        arcpy.AddMessage("embakment left transect {}-{} records: {}".format(count_section, count_transect, list))

    return list

def embakment_right(select_embakment_right_file, count_section, count_transect, count_id_transect, tmp_dir, torrent_values_gdb
    ):
    # select one transect line
    select_transect_line_file = os.path.join(torrent_values_gdb,
                                             "select_transect_line_{}_{}".format(count_section, count_transect))
```

```
    arcpy.Select_analysis(select_embakment_right_file, select_transect_line_file, "ORIG_FID={}".format(count_transect))

    # count
    count_select = float(str(arcpy.GetCount_management(select_transect_line_file)))
    arcpy.AddMessage("transect {}-{} has {} records".format(count_section, count_transect, count_select))

    # if there are less than 2 points an empty np_array is generated
    if count_select < 2:
        list = [[count_id_transect, None, None, None, None]]

    else:
        # calculates points statistics
        points_stat_file = os.path.join(torrent_values_gdb,
                                        "embakment_right_{}_{}".format(count_section, count_transect))
        arcpy.Statistics_analysis(select_transect_line_file, points_stat_file,
                                  [["dtm_1", "MAX"], ["dtm_1", "MIN"], ["OID", "MAX"], ["OID", "MIN"]])

        with arcpy.da.SearchCursor(points_stat_file, ["MAX_dtm_1"]) as cursor:
            for row in cursor:
                highest_point_dtm = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MIN_dtm_1"]) as cursor:
            for row in cursor:
                lowest_point_dtm = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MAX_OID"]) as cursor:
            for row in cursor:
                highest_point_id = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MIN_OID"]) as cursor:
            for row in cursor:
                lowest_point_id = row[0]

        height = float(highest_point_dtm) - float(lowest_point_dtm)
        width = 0.25 * (float(highest_point_id) - float(lowest_point_id))
        diagonal = (height ** 2 + width ** 2) ** 0.5

        if diagonal > 0:
            radians = math.asin(height / diagonal)
            inclination = math.degrees(radians)

        else:
            inclination = 0

        list = [[count_id_transect, height, width, diagonal, inclination]]

        arcpy.AddMessage("embakment right transect {}-{} records: {}".format(count_section, count_transect, list))

    return list

def torrent_bed(select_torrent_bed_file, count_section, count_transect, count_id_transect, max_count_transect,
                    height_diff_talweg, min_dtm_talweg, max_dtm_talweg, tmp_dir, torrent_values_gdb):
    # select one transect line
    select_transect_line_file = os.path.join(torrent_values_gdb,
                                             "select_transect_line_{}_{}".format(count_section, count_transect))
    arcpy.Select_analysis(select_torrent_bed_file, select_transect_line_file, "ORIG_FID={}".format(count_transect))

    # count
    count_select = float(str(arcpy.GetCount_management(select_transect_line_file)))
    arcpy.AddMessage("transect {}-{} has {} records".format(count_section, count_transect, count_select))

    # if there are less than 2 points an empty np_array is generated
    if count_select < 2:
        list = [[count_id_transect, count_section, None, None, None, None, None, None, None, None, None]]

    else:
        # calculates points statistics
        points_stat_file = os.path.join(torrent_values_gdb,
                                        "torrent_bed_{}_{}".format(count_section, count_transect))
        arcpy.Statistics_analysis(select_transect_line_file, points_stat_file,
                                  [["dtm_1", "MEAN"], ["dtm_1", "STD"], ["OID", "MAX"], ["OID", "MIN"]])

        with arcpy.da.SearchCursor(points_stat_file, ["MEAN_dtm_1"]) as cursor:
            for row in cursor:
                mean_dtm = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["STD_dtm_1"]) as cursor:
            for row in cursor:
                std_dtm = row[0]
```

```
        with arcpy.da.SearchCursor(points_stat_file, ["MAX_OID"]) as cursor:
            for row in cursor:
                max_point_id = row[0]

        with arcpy.da.SearchCursor(points_stat_file, ["MIN_OID"]) as cursor:
            for row in cursor:
                min_point_id = row[0]

        # select transect line 5m above (-10id)
        transect_5m_above = count_transect - 10
        if transect_5m_above < 0:
            transect_5m_above = 0

        select_transect_line_above_file = os.path.join(torrent_values_gdb,
                                               "select_transect_line_above_{}_{}".format(count_section, count_transect))
        arcpy.Select_analysis(select_torrent_bed_file, select_transect_line_above_file,
                              "ORIG_FID={}".format(transect_5m_above))

        # calculates points statistics
        points_stat_above_file = os.path.join(torrent_values_gdb,
                                      "torrent_bed_above_{}_{}".format(count_section, count_transect))
        arcpy.Statistics_analysis(select_transect_line_above_file, points_stat_above_file,
                                  [["dtm_1", "MEAN"]])

        with arcpy.da.SearchCursor(points_stat_above_file, ["MEAN_dtm_1"]) as cursor:
            for row in cursor:
                mean_dtm_above = row[0]

        # test if local variabel is assigned
        try:
            mean_dtm_above
        except NameError:
            mean_dtm_above = max_dtm_talweg


        # select transect line 5m below (+10id)
        transect_5m_below = count_transect + 10
        if transect_5m_below > max_count_transect:
            transect_5m_below = max_count_transect - 1

        select_transect_line_below_file = os.path.join(torrent_values_gdb,
                                                       "select_transect_line_below_{}_{}".format(count_section,
                                                                                                 count_transect))
        arcpy.Select_analysis(select_torrent_bed_file, select_transect_line_below_file,
                              "ORIG_FID={}".format(transect_5m_below))

        # calculates points statistics
        points_stat_below_file = os.path.join(torrent_values_gdb,
                                      "torrent_bed_below_{}_{}".format(count_section, count_transect))
        arcpy.Statistics_analysis(select_transect_line_below_file, points_stat_below_file,
                                  [["dtm_1", "MEAN"]])

        with arcpy.da.SearchCursor(points_stat_below_file, ["MEAN_dtm_1"]) as cursor:
            for row in cursor:
                mean_dtm_below = row[0]

        # test if local variabel is assigned
        try:
            mean_dtm_below
        except NameError:
            mean_dtm_below = min_dtm_talweg


        height_diff = float(mean_dtm_above) - float(mean_dtm_below)

        length_diff = 10
        diagonal = (height_diff ** 2 + length_diff ** 2) ** 0.5

        if diagonal > 0:
            radians = math.asin(height_diff / diagonal)
            inclination = math.degrees(radians)
        else:
            inclination = 0

        # width
        width = 0.25 * (float(max_point_id) - float(min_point_id))

        mean_height = mean_dtm

        std_height = std_dtm
```

```
587
588         # calculation of talweg inclination
589         length_2_lowerend_talweg = 0.5 * (max_count_transect - count_transect)
590         length_diff_talweg = 0.5 * (float(max_count_transect))
591         diagonal_talweg = (height_diff_talweg ** 2 + length_diff_talweg ** 2) ** 0.5
592
593         radians = math.asin(height_diff_talweg / diagonal_talweg)
594         inclination_talweg = math.degrees(radians)
595
596         # calculation height and height difference to talweg
597         height_talweg = (math.tan(inclination_talweg) * length_2_lowerend_talweg) + min_dtm_talweg
598
599         diff_talweg = float(height_talweg) - float(mean_height)
600
601
602         list = [[count_id_transect, count_section, mean_height, std_height, width, diagonal, inclination,
603                     inclination_talweg, diff_talweg, diagonal_talweg, length_diff_talweg]]
604
605         arcpy.AddMessage("torrent bed transect {}-{} records: {}".format(count_section, count_transect, list))
606
607     return list
608
609
610
611 def main():
612
613     #input = arcpy.GetParameterAsText(0)
614     #output = arcpy.GetParameterAsText(1)
615     #...
616
617     # Extent
618     # defines within CH1903+ the outer edges of the calculation domain
619
620     # Arelen
621     extent = "2782000 1189000 2785000 1191000"
622
623
624
625     ######################################
626
627     # in the following section all sources are listed.
628
629     # folder start_dir defines the main folder
630     start_dir = r"D:\Debris_Flow_Project\eCognition\torrent_values\Arelen20211220"
631
632     dtm_dir = "DTM"
633     dtm_name = "dtm.tif"
634     dtm_file = os.path.join(os.path.join(start_dir, dtm_dir), dtm_name)
635
636     ecog_dir = "ECOG"
637     ecog_name = "torrent_extent_2021_exact.shp"
638     ecog_file = os.path.join(os.path.join(start_dir, ecog_dir), ecog_name)
639
640     transect_dir = "TRANSECT"
641     transect_name = "transect.shp"
642     transect_file = os.path.join(os.path.join(start_dir, transect_dir), transect_name)
643
644     bed_dir = "BED"
645     bed_name = "aggregate_torrent_bed_fin_s2.shp"
646     bed_file = os.path.join(os.path.join(start_dir, bed_dir), bed_name)
647
648
649
650     # output folder
651     #(! in this folder clip_transect_file of all sections have to manually copied from eCog preperation script !)
652     tmp_dir = os.path.join(start_dir, "_Calculations_v4_exact2_s2")
653
654     # with env.snapRaster all raster files are aligned properly
655     env.snapRaster = dtm_file
656
657
658     # calls the calculation
659     process_data_fun(extent, dtm_file, ecog_file, transect_file, bed_file, tmp_dir)
660
661
662 if __name__ == "__main__":
663
664     g_c = 0         # count
665     print("start")
666     main()
```

```
667     print("Done!")
```

B.5. Systematic Error Model MATLAB

This script determines the systematic error surface based on the tie-point differences of two surveys.

```
close all ;
clear all ;

load arelen20200626_cook2.mat ;
load arelen20210825_cook2.mat ;
load colzweimeter.mat ;

% loading point clouds as matrix
Wolfgang1 = projectcomb20202021ptprec2020 ;
Wolfgang2 = projectcomb20202021ptprec2021 ;

% derive information
%disp(Wolfgang1(1:10,1))
minx=min(Wolfgang1(:,1));
maxx=max(Wolfgang1(:,1));
miny=min(Wolfgang1(:,2));
maxy=max(Wolfgang1(:,2));
x=[minx:1:maxx];
y=[miny:1:maxy];
[X Y]=meshgrid(x,y);
Z1=griddata(Wolfgang1(:,1),Wolfgang1(:,2),Wolfgang1(:,3),X,Y);
Z2=griddata(Wolfgang2(:,1),Wolfgang2(:,2),Wolfgang2(:,3),X,Y);
ZO=Z2-Z1;
 ZO(ZO<-5)=-9999;
% ZO(ZO>5)=-9999;
xm=X(:);
ym=Y(:);
zm=ZO(:);
data=[xm,ym,zm];
% data=sortrows(data,3);
% nz=nansum(data(:,3)<-9998);
% data(1:nz,:)=[];
% xm=data(:,1);
% ym=data(:,2);
% zm=data(:,3);
[xData, yData, zData] = prepareSurfaceData(xm, ym, zm );
ft = fittype( 'poly22' );
opts = fitoptions( 'Method', 'LinearLeastSquares');
opts.Robust = 'Bisquare';
[fitresult, gof] = fit( [xData, yData], zData, ft, opts );
ZE=feval(fitresult,X,Y);

%new corrected difference
ZN=(Z2-Z1)-ZE; %gleich wie (Z2-ZE)-Z1 !

%new z heights for 2nd pc
CZ2 = Z2-ZE;


%creat new point cloud z2
newz = CZ2(:) ;
correction = ZE(:) ;
newpc = [xm, ym, newz, correction] ;
dlmwrite('arelen_2021_cor.asc',newpc,'precision','%.6f');

figure ;
mesh(ZE)
zlabel("deviation z [m]")
%ylabel("CH1903+")
%xlabel("CH1903+")
title("Systematic Error Arelen 2020/2021")

figure ;
imagesc(Z2-Z1)
zlabel("deviation z [m]")
%ylabel("CH1903+")
%xlabel("CH1903+")
title("Difference Arelen 2020/2021")
%colormap(zweimeter)

```

```
figure ;
imagesc(ZN)
zlabel("deviation z [m]")
%ylabel("CH1903+")
%xlabel("CH1903+")
title("corrected Difference Arelen 2020/2021")
%colormap(zweimeter)
```

B.6. Torrent Values Graphic MATLAB

This script visualizes the generated torrential properties in a 3D MATLAB figure.

```
clc
clearvars
%close all


%% Set up the Import Options and import the data
opts = delimitedTextImportOptions("NumVariables", 28, "Encoding", "UTF-8");

% Specify range and delimiter
opts.DataLines = [1, Inf];
opts.Delimiter = ",";

% Specify column names and types
opts.VariableNames = ["OID_", "Id", "ORIG_FID", "Id_1", "e_l_hight", "e_l_width", "e_l_diagon", "e_l_inclin", "e_r_hight", "e_r_width", "e_r_diagon", "e_r_inclin", "count_sect", "t_mean_hig", "t_std_high", "t_width", "t_diagonal", "t_inclinat", "talweg_inc", "talweg_tor", "talweg_pat", "talweg_lin", "volume_cs", "RASTERVALU", "POINT_X", "POINT_Y", "POINT_Z", "POINT_M"];
opts.VariableTypes = ["double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "double", "string", "double", "double", "double", "double", "double"];

% Specify file level properties
opts.ExtraColumnsRule = "ignore";
opts.EmptyLineRule = "read";

% Specify variable properties
opts = setvaropts(opts, "volume_cs", "WhitespaceRule", "preserve");
opts = setvaropts(opts, "volume_cs", "EmptyFieldRule", "auto");

% Import the data
tabletest2nd = readtable("Z:\Debris_Flow_Project\_MATLAB\fromAndrin_20211210\matlab_arelen_ecog_20211224_v5.csv", opts); %path fo
tabletest = readtable("Z:\Debris_Flow_Project\_MATLAB\fromAndrin_20211210\matlab_arelen_exact_20211224.csv", opts); %path fo


colormap jet


field = "e_r_hight";



%% plot energy dissipation - Figure 1

comp = figure(1);
clf



set(comp,'units','normalized','outerposition',[0.01 0.25 0.99 .65]) % maximize figure to screen with tool bar
set(gcf,'color','w');
FS = 20;

fw = 0.25;
fh = 0.90;



%colormap jet

%% import TIF DEM
% read DEM
[dem, R_dem] = geotiffread('swissSURFACE3D_1m_arelen.tif'); %swissSURFACE3D_1m_arelen.tif

```

```
cellsize = R_dem.CellExtentInWorldX;
nrows = R_dem.RasterSize(1);
ncols = R_dem.RasterSize(2);
Xmin = R_dem.XWorldLimits(1);
Xmax = R_dem.XWorldLimits(2);
Ymin = R_dem.YWorldLimits(1);
Ymax = R_dem.YWorldLimits(2);

% generate mesh coordinates for plotting purposes with DEM
xgv = Xmin:cellsize:Xmax-cellsize; %length(X);
ygv = Ymin:cellsize:Ymax-cellsize; %length(Y);
[X,Y] = meshgrid(xgv,ygv);

%% flip and turn dem (no idea why needed...)
dem1mclip_up = flipud(dem)-3;   % up down flip and 3 m downwards


%%
clf
ax0 = axes('Position',[.05 .05 .95 .95]); %0.115
%[C1,h1] = contour3(ax1,X,Y,dem1mclip_up,400); hold on
%set(h1,'LineColor',[.2 .2 .2]); % line color of contour plot


s = mesh(xgv,ygv,dem1mclip_up); hold on
%colormap([.8 .8 .8])
view([45 20]); % -52  %arelen 45 20 %fraschmardin 45 20 %from copy: -46 24]


C0(:,:,1) = ones(size(X,1),size(X,2)).*.7; % red
C0(:,:,2) = ones(size(Y,1),size(Y,2)).*.7; % green
C0(:,:,3) = ones(size(dem1mclip_up,1),size(dem1mclip_up,2)).*.7; % blue
mesh(X,Y,dem1mclip_up,C0)




%% weird ordering idx = 135,136,137,138
axes(ax0)

%% subfigure 1
clf

maxID = max(tabletest.ORIG_FID); %maximale anzahl CS
n= maxID;



field_variabe = tabletest.e_l_diagon;
ax1 = axes('Position',[0.05 0.05 fw fh]); % 0.38
set(gca,'FontSize',14)


mesh(X,Y,dem1mclip_up,C0); hold on;
camlight('right')

contour3(X,Y,dem1mclip_up+1,[1300:100:2000],'-w'); hold on;
annotation('textbox',...
    [0.1355 0.258 0.03 0.02],...
    'Color',[1 1 1],...
    'String',{'1700'},...
    'LineStyle','none',...
    'FontWeight','bold',...
    'FontSize',14,...
    'FitBoxToText','off');

annotation('textbox',...
    [0.115 0.395 0.03 0.02],...
    'Color',[1 1 1],...
    'String',{'1800'},...
    'LineStyle','none',...
    'FontWeight','bold',...
    'FontSize',14,...
    'FitBoxToText','off');

annotation('textbox',...
    [0.075 0.6404 0.03 0.02],...
    'Color',[1 1 1],...
    'String',{'2000'},...
    'LineStyle','none',...
```

```
    'FontWeight','bold',...
    'FontSize',14,...
    'FitBoxToText','off');


for mm = 1:n
        idx = find(tabletest.ORIG_FID == mm); %selection von einer CS
        B = sortrows([tabletest.POINT_X(idx),tabletest.POINT_Y(idx),tabletest.RASTERVALU(idx)],1); %müssen sortiert werden, um
       querverbindungen zu verhindern

    col = field_variabe(idx);  % This is the color, vary with resulting velocity

    surface([B(:,1)';B(:,1)'],[B(:,2)';B(:,2)'],...
        [B(:,3)';B(:,3)'],[col';col'],...
        'facecol','no',...
    'edgecol','interp',...
    'linew',0.5);
    hold on;

end

view([73,18]) %fraschmardin -20 13 %arelen 52 32
daspect([1 1 1])
grid on


% Set the remaining axes properties
cbar= colorbar;
set(cbar,'Location','southoutside',...
    'Position', [0.185792349726776 0.0950764006791171 0.107468123861566 0.0186757215619694],...
    'FontSize',12,...
    'FontName','Arial');
caxis([0 20])

cbar.Label.String = 'Embankment height left GIS [m]';
set(cbar.Label,'FontSize',13)
colormap(ax1,'hot');

axis off

% Create textbox
annotation('textbox',...
    [0.05929 0.7791 0.020250758955677 0.0543293718166383],...
    'Color',[1 1 1],...
    'String',{'e'},...
    'LineStyle','none',...
    'FontWeight','bold',...
    'FontSize',22,...
    'FontName','Arial',...
    'FitBoxToText','off',...
    'EdgeColor','none');

%% subfigure 2
maxID = max(tabletest2nd.ORIG_FID); %maximale anzahl CS
n= maxID;


field_variabe = tabletest2nd.e_l_diagon;

offset = 0.15;
ax2 = axes('Position',[0.05+offset 0.05 fw fh]); % 0.38
set(gca,'FontSize',14)
colormap(ax2,flipud(cork0))

mesh(X,Y,dem1mclip_up,C0); hold on;
camlight('right');

contour3(X,Y,dem1mclip_up+1,[1300:100:2000],'-w'); hold on;
annotation('textbox',...
    [0.1355+offset 0.258 0.03 0.02],...
    'Color',[1 1 1],...
    'String',{'1700'},...
    'FontWeight','bold',...
    'FontSize',14,...
    'LineStyle','none',...
    'FitBoxToText','off');

annotation('textbox',...
    [0.115+offset 0.395 0.03 0.02],...
    'Color',[1 1 1],...
```

```
    'String',{'1800'},...
    'FontWeight','bold',...
    'FontSize',14,...
    'LineStyle','none',...
    'FitBoxToText','off');

annotation('textbox',...
    [0.075+offset 0.6404 0.03 0.02],...
    'Color',[1 1 1],...
    'String',{'2000'},...
    'FontWeight','bold',...
    'FontSize',14,...
    'LineStyle','none',...
    'FitBoxToText','off');



for mm = 1:n
        idx = find(tabletest2nd.ORIG_FID == mm); %selection von einer CS
        B = sortrows([tabletest2nd.POINT_X(idx),tabletest2nd.POINT_Y(idx),tabletest2nd.RASTERVALU(idx)],1); %müssen sortiert
      werden, um querverbindungen zu verhindern

    col = field_variabe(idx);  % This is the color, vary with resulting velocity

    surface([B(:,1)';B(:,1)'],[B(:,2)';B(:,2)'],...
        [B(:,3)';B(:,3)'],[col';col'],...
        'facecol','no',...
    'edgecol','interp',...
    'linew',0.5);
    hold on;

end

view([73,18]) %fraschmardin -20 13 %arelen 52 32  %arelen new 73 18
daspect([1 1 1])
grid on


% Set the remaining axes properties
cbar= colorbar;
set(cbar,'Location','southoutside',...
    'Position', [0.185792349726776+offset 0.0950764006791171 0.107468123861566 0.0186757215619694],...
    'FontSize',12,...
    'FontName','Arial');

caxis([0 20]) %for CS area

cbar.Label.String = 'Embankment height left OBIA [m]';
set(cbar.Label,'FontSize',13)
colormap(ax2,'hot');


axis off


% Create textbox
annotation('textbox',...
    [0.059287795992714+offset 0.7791 0.020250758955677 0.0543293718166383],...
    'Color',[1 1 1],...
    'String',{'f'},...
    'LineStyle','none',...
    'FontWeight','bold',...
    'FontSize',22,...
    'FontName','Arial',...
    'FitBoxToText','off',...
    'EdgeColor','none');
%% subfigure 3
maxID = max(tabletest.ORIG_FID); %maximale anzahl CS
n= maxID;


field_variabe = tabletest.e_r_diagon;

offset = 2*0.15;
ax3 = axes('Position',[0.05+offset 0.05 fw fh]); % 0.38
set(gca,'FontSize',14)


mesh(X,Y,dem1mclip_up,CO); hold on;
camlight('right');
```

```
297
298 contour3(X,Y,dem1mclip_up+1,[1300:100:2000],'-w'); hold on;
299 annotation('textbox',...
300     [0.1355+offset 0.258 0.03 0.02],...
301     'Color',[1 1 1],...
302     'String',{'1700'},...
303     'FontWeight','bold',...
304     'FontSize',14,...
305     'LineStyle','none',...
306     'FitBoxToText','off');
307
308 annotation('textbox',...
309     [0.115+offset 0.395 0.03 0.02],...
310     'Color',[1 1 1],...
311     'String',{'1800'},...
312     'FontWeight','bold',...
313     'FontSize',14,...
314     'LineStyle','none',...
315     'FitBoxToText','off');
316
317 annotation('textbox',...
318     [0.075+offset 0.6404 0.03 0.02],...
319     'Color',[1 1 1],...
320     'String',{'2000'},...
321     'FontWeight','bold',...
322     'FontSize',14,...
323     'LineStyle','none',...
324     'FitBoxToText','off');
325
326
327 for mm = 1:n
328         idx = find(tabletest.ORIG_FID == mm); %selection von einer CS
329         B = sortrows([tabletest.POINT_X(idx),tabletest.POINT_Y(idx),tabletest.RASTERVALU(idx)],1); %müssen sortiert werden, um
        querverbindungen zu verhindern
330
331     col = field_variabe(idx);  % This is the color, vary with resulting velocity
332
333     surface([B(:,1)';B(:,1)'],[B(:,2)';B(:,2)'],...
334         [B(:,3)';B(:,3)'],[col';col'],...
335         'facecol','no',...
336     'edgecol','interp',...
337     'linew',0.5);
338     hold on;
339
340 end
341
342 view([73,18]) %fraschmardin -20 13 %arelen 52 32
343 daspect([1 1 1])
344 grid on
345
346
347 % Set the remaining axes properties
348
349 cbar= colorbar;
350 set(cbar,'Location','southoutside',...
351     'Position', [0.185792349726776+offset 0.0950764006791171 0.107468123861566 0.0186757215619694],...
352     'FontSize',12,...
353     'FontName','Arial');
354
355 caxis([0 20]) %for CS area
356
357 cbar.Label.String = 'Embankment height right GIS [m]';
358 set(cbar.Label,'FontSize',13)
359 colormap(ax3,'hot');
360
361
362 axis off
363
364
365 % Create textbox
366 annotation('textbox',...
367     [0.059287795992714+offset 0.7791 0.020250758955677 0.0543293718166383],...
368     'Color',[1 1 1],...
369     'String',{'g'},...
370     'LineStyle','none',...
371     'FontWeight','bold',...
372     'FontSize',22,...
373     'FontName','Arial',...
374     'FitBoxToText','off',...
375     'EdgeColor','none');
```

```
376
377 %% subfigure 4
378 maxID = max(tabletest2nd.ORIG_FID); %maximale anzahl CS
379 n= maxID;
380
381
382 field_variabe = tabletest2nd.e_r_diagon;
383
384 offset = 3*0.15;
385 ax4 = axes('Position',[0.05+offset 0.05 fw fh]); % 0.38
386 set(gca,'FontSize',14)
387
388
389 mesh(X,Y,dem1mclip_up,C0); hold on;
390 camlight('right');
391
392 contour3(X,Y,dem1mclip_up+1,[1300:100:2000],'-w'); hold on;
393 annotation('textbox',...
394     [0.1355+offset 0.258 0.03 0.02],...
395     'Color',[1 1 1],...
396     'String',{'1700'},...
397     'FontWeight','bold',...
398     'FontSize',14,...
399     'LineStyle','none',...
400     'FitBoxToText','off');
401
402 annotation('textbox',...
403     [0.115+offset 0.395 0.03 0.02],...
404     'Color',[1 1 1],...
405     'String',{'1800'},...
406     'FontWeight','bold',...
407     'FontSize',14,...
408     'LineStyle','none',...
409     'FitBoxToText','off');
410
411 annotation('textbox',...
412     [0.075+offset 0.6404 0.03 0.02],...
413     'Color',[1 1 1],...
414     'String',{'2000'},...
415     'FontWeight','bold',...
416     'FontSize',14,...
417     'LineStyle','none',...
418     'FitBoxToText','off');
419
420
421
422 for mm = 1:n
423         idx = find(tabletest2nd.ORIG_FID == mm); %selection von einer CS
424         B = sortrows([tabletest2nd.POINT_X(idx),tabletest2nd.POINT_Y(idx),tabletest2nd.RASTERVALU(idx)],1); %müssen sortiert
      werden, um querverbindungen zu verhindern
425
426     col = field_variabe(idx);  % This is the color, vary with resulting velocity
427
428     surface([B(:,1)';B(:,1)'],[B(:,2)';B(:,2)'],...
429         [B(:,3)';B(:,3)'],[col';col'],...
430         'facecol','no',...
431     'edgecol','interp',...
432     'linew',0.5);
433     hold on;
434
435 end
436
437 view([73,18]) %fraschmardin -20 13 %arelen 52 32
438 daspect([1 1 1])
439 grid on
440
441
442 % Set the remaining axes properties
443 axes1 = comp;
444
445 cbar= colorbar;
446 set(cbar,'Location','southoutside',...
447     'Position', [0.185792349726776+offset 0.0950764006791171 0.107468123861566 0.0186757215619694],...
448     'FontSize',12,...
449     'FontName','Arial');
450
451 caxis([0 20]) %for CS area
452
453 cbar.Label.String = 'Embankment height right OBIA [m]';
454 set(cbar.Label,'FontSize',13)
```

```
colormap(ax4,'hot');


axis off

% Create textbox
annotation('textbox',...
    [0.059287795992714+offset 0.7791 0.020250758955677 0.0543293718166383],...
    'Color',[1 1 1],...
    'String',{'h'},...
    'LineStyle','none',...
    'FontWeight','bold',...
    'FontSize',22,...
    'FontName','Arial',...
    'FitBoxToText','off',...
    'EdgeColor','none');

%% export

%export_fig('Z:\Debris_Flow_Project\_MATLAB\export_torrential_properties\fig_arelen_42_3','-png')

f = gcf;
%exportgraphics(f,'fig_arelen_42_4.pdf','ContentType','vector',...
%               'BackgroundColor','none')

exportgraphics(f,'fig_arelen_42_6.png','Resolution',500)
```

B.7. DTM Statistics R

This script compares the elevation and ground point density of the different point cloud routines and visualizes them.

```
library(tidyverse)
library(patchwork)


#set working directory
setwd("D:/_Polybox/_UZH/Masterthesis/Statistics/Statistics_Masterarbeit/Data")

#read data
data_legfoere <- read.csv("rp_legfoere_v4.csv", header = TRUE) %>%
        select(swisurface, terra, swialt, lastool, lidarlow, lidarhigh, d_lowres_n, d_highre_n, d_swisur_n, d_terra_n, d_
      lastoo_n) %>%
        mutate(origin = "legfoere")


data_torrent <- read.csv("rp_torrent_v4.csv", header = TRUE) %>%
        select(swisurface, terra, swissalt3d, lastools, lidarlow_n, lidarhig_n, d_lowres_n, d_highre_n, d_swisur_n, d_terra_n,
       d_lastoo_n) %>%
        mutate(origin = "torrent") %>%
        rename(swialt = swissalt3d, lidarlow = lidarlow_n, lidarhigh = lidarhig_n, lastool = lastools)

data_openforest <- read.csv("rp_forest_open_v4.csv", header = TRUE)%>%
        select(swisurface, terra, swissalt, lastool, lidarlow_n, lidarhig_n, d_lowres_n, d_highre_n, d_swisur_n, d_terra_n, d_
      lastoo_n) %>%
        mutate(origin = "openforest") %>%
        rename(swialt = swissalt, lidarlow = lidarlow_n, lidarhigh = lidarhig_n)


# rename(new = old)

## combine data
data_comb <- data_legfoere %>%
        rbind(data_torrent)%>%
        rbind(data_openforest)


## data wrangling
data_comb[data_comb == -9999] <- 0


data_comb_density <- data_comb %>%
        pivot_longer(cols = starts_with("d_"),
                     names_to = "d_routine",
                     values_to = "density")
```

```
41
42 data_comb_density$d_routine = factor(data_comb_density$d_routine, levels = c("d_lastoo_n", "d_terra_n", "d_swisur_n", "d_
       highre_n", "d_lowres_n"))
43
44
45
46 data_comb <- data_comb %>%
47         mutate(diff_terra = (terra - swisurface) * 100,
48                diff_swialt = (swialt - swisurface) * 100,
49                diff_lastool = (lastool - swisurface) * 100,
50                diff_lidarlow = (lidarlow - swisurface) * 100,
51                diff_lidarhigh = (lidarhigh - swisurface) * 100) %>%
52         pivot_longer(cols = starts_with("diff_"),
53                      names_to = "routine",
54                      values_to = "height_diff")
55
56 data_comb$routine = factor(data_comb$routine, levels = c("diff_lastool", "diff_terra", "diff_swialt", "diff_lidarhigh", "diff_
       lidarlow"))
57
58
59 ggplot( data_comb, aes(x=origin, y=height_diff, fill=routine)) +
60         theme_bw() +
61         geom_boxplot() +
62         #theme(legend.position="none") +
63         scale_fill_brewer(palette="BrBG")
64         #scale_y_continuous(trans='log10')
65
66
67
68
69
70
71 ggplot( data_comb_density, aes(x=origin, y=density, fill=d_routine), log10="y") +
72         theme_bw() +
73         geom_boxplot() +
74         #theme(legend.position="none") +
75         scale_fill_brewer(palette="BrBG") +
76         scale_y_continuous(trans='log10')
77
78
79 #Creating Histograms
80 h1 <- data_comb_density %>%
81         filter(d_routine == "d_highre_n" & origin == "legfoere") %>%
82         ggplot( aes(density)) +
83         theme_bw() +
84         geom_histogram(aes(y = ..density..), binwidth = 1,
85                        col="black",
86                        fill="black") +
87         geom_density() +
88         labs(title= element_blank(), x=element_blank(), y="highRes LidAR") +
89         xlim(c(0,500)) +
90         ylim(c(0,0.12)) +
91         #theme(axis.title.x = element_blank()) +
92         theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
93         theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())
94
95
96 h2 <- data_comb_density %>%
97         filter(d_routine == "d_lowres_n" & origin == "legfoere") %>%
98         ggplot( aes(density)) +
99         theme_bw() +
100        geom_histogram(aes(y = ..density..), binwidth = 1,
101                       col="black",
102                       fill="black") +
103        geom_density() +
104        labs(title= element_blank(), x=element_blank(), y="lowRes LidAR") +
105        xlim(c(0,500)) +
106        ylim(c(0,0.12)) +
107        #theme(axis.title.x = element_blank()) +
108        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
109        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())
110
111 h3 <- data_comb_density %>%
112        filter(d_routine == "d_lastoo_n" & origin == "legfoere") %>%
113        ggplot( aes(density)) +
114        theme_bw() +
115        geom_histogram(aes(y = ..density..), binwidth = 1,
116                       col="black",
117                       fill="black") +
118        geom_density() +
```

```
        labs(title= element_blank(), x=element_blank(), y="LASTools") +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h4 <- data_comb_density %>%
        filter(d_routine == "d_terra_n" & origin == "legfoere") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x=element_blank(), y="Terrasolid") +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h5 <- data_comb_density %>%
        filter(d_routine == "d_swisur_n" & origin == "legfoere") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x="Ground Points/m² Shrub Forest", y= "SwissSURFACE3D") +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


####
h6 <- data_comb_density %>%
        filter(d_routine == "d_highre_n" & origin == "openforest") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


h7 <- data_comb_density %>%
        filter(d_routine == "d_lowres_n" & origin == "openforest") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h8 <- data_comb_density %>%
        filter(d_routine == "d_lastoo_n" & origin == "openforest") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
```

```
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h9 <- data_comb_density %>%
        filter(d_routine == "d_terra_n" & origin == "openforest") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h10 <- data_comb_density %>%
        filter(d_routine == "d_swisur_n" & origin == "openforest") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x="Ground Points/m² Open Forest", y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())



####
h11 <- data_comb_density %>%
        filter(d_routine == "d_highre_n" & origin == "torrent") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


h12 <- data_comb_density %>%
        filter(d_routine == "d_lowres_n" & origin == "torrent") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h13 <- data_comb_density %>%
        filter(d_routine == "d_lastoo_n" & origin == "torrent") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
```

```
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h14 <- data_comb_density %>%
        filter(d_routine == "d_terra_n" & origin == "torrent") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h15 <- data_comb_density %>%
        filter(d_routine == "d_swisur_n" & origin == "openforest") %>%
        ggplot( aes(density)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x="Points/m² Torrent ", y=element_blank()) +
        xlim(c(0,500)) +
        ylim(c(0,0.12)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


(h1+h6+h11)/(h2+h7+h12)/(h3+h8+h13)/(h4+h9+h14)/(h5+h10+h15)

###

yb = 0.07
#Creating Histograms for Hight Difference to SwissSURFACE
h1 <- data_comb %>%
        filter(routine == "diff_lidarhigh" & origin == "legfoere") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x=element_blank(), y="highRes LidAR") +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())



h2 <- data_comb %>%
        filter(routine == "diff_lidarlow" & origin == "legfoere") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x=element_blank(), y="lowRes LidAR") +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h3 <- data_comb %>%
        filter(routine == "diff_lastool" & origin == "legfoere") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
```

```
        geom_density() +
        labs(title= element_blank(), x=element_blank(), y="LASTools") +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h4 <- data_comb %>%
        filter(routine == "diff_terra" & origin == "legfoere") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x=element_blank(), y="Terrasolid") +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h5 <- data_comb %>%
        filter(routine == "diff_swialt" & origin == "legfoere") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x="Shrub Forest Difference SwissSURFACE3D [cm]", y= "SwissSURFACE3D") +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


####
h6 <- data_comb %>%
        filter(routine == "diff_lidarhigh" & origin == "openforest") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


h7 <- data_comb %>%
        filter(routine == "diff_lidarlow" & origin == "openforest") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h8 <- data_comb %>%
        filter(routine == "diff_lastool" & origin == "openforest") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
```

```
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h9 <- data_comb %>%
        filter(routine == "diff_terra" & origin == "openforest") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h10 <- data_comb %>%
        filter(routine == "diff_swialt" & origin == "openforest") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x="Open Forest Difference SwissSURFACE3D [cm]", y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())



####
h11 <- data_comb %>%
        filter(routine == "diff_lidarhigh" & origin == "torrent") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


h12 <- data_comb %>%
        filter(routine == "diff_lidarlow" & origin == "torrent") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h13 <- data_comb %>%
        filter(routine == "diff_lastool" & origin == "torrent") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
```

```
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h14 <- data_comb %>%
        filter(routine == "diff_terra" & origin == "torrent") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x= element_blank(), y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())

h15 <- data_comb %>%
        filter(routine == "diff_swialt" & origin == "openforest") %>%
        ggplot( aes(height_diff)) +
        theme_bw() +
        geom_histogram(aes(y = ..density..), binwidth = 1,
                       col="black",
                       fill="black") +
        geom_density() +
        labs(title= element_blank(), x="Torrent Difference SwissSURFACE3D [cm]", y=element_blank()) +
        xlim(c(-100,300)) +
        ylim(c(0, yb)) +
        #theme(axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.grid.major.y = element_blank(), panel.grid.minor.y = element_blank())


(h1+h6+h11)/(h2+h7+h12)/(h3+h8+h13)/(h4+h9+h14)/(h5+h10+h15)
```

B.8. Boxplot Torrent Sections R

This script takes the geometrical measures of the torrent classified by section and cross compares them with automatically OBIA, manually GIS and field-based measurements.

```
#set working directory
setwd("D:/_Polybox/_UZH/Masterthesis/Statistics/Statistics_Masterarbeit/transect_data_arelen")
library(ggplot2)

#read data
ecog_section_a <- read.csv("clip_s3_ecog_section_A.csv", header = TRUE)
ecog_section_b <- read.csv("clip_s3_ecog_section_B.csv", header = TRUE)
ecog_section_c <- read.csv("clip_s3_ecog_section_C.csv", header = TRUE)
ecog_section_d <- read.csv("clip_s3_ecog_section_D.csv", header = TRUE)
ecog_section_e <- read.csv("clip_s3_ecog_section_E.csv", header = TRUE)
ecog_section_f <- read.csv("clip_s3_ecog_section_F.csv", header = TRUE)
ecog_section_g <- read.csv("clip_s3_ecog_section_G.csv", header = TRUE)
#ecog_section_h <- read.csv("clip_s3_ecog_section_H.csv", header = TRUE)
#ecog_section_i <- read.csv("clip_s3_ecog_section_I.csv", header = TRUE)
#ecog_section_j <- read.csv("clip_s3_ecog_section_J.csv", header = TRUE)



exact_section_a <- read.csv("clip_s3_exact_section_A.csv", header = TRUE)
exact_section_b <- read.csv("clip_s3_exact_section_B.csv", header = TRUE)
exact_section_c <- read.csv("clip_s3_exact_section_C.csv", header = TRUE)
exact_section_d <- read.csv("clip_s3_exact_section_D.csv", header = TRUE)
exact_section_e <- read.csv("clip_s3_exact_section_E.csv", header = TRUE)
exact_section_f <- read.csv("clip_s3_exact_section_F.csv", header = TRUE)
exact_section_g <- read.csv("clip_s3_exact_section_G.csv", header = TRUE)
#exact_section_h <- read.csv("clip_s3_exact_section_H.csv", header = TRUE)
#exact_section_i <- read.csv("clip_s3_exact_section_I.csv", header = TRUE)
#exact_section_j <- read.csv("clip_s3_exact_section_J.csv", header = TRUE)


field_df <- read.csv("field_values_v5_arelen.csv", header = TRUE, sep = ";")

```

```
#add extraction type as new column to the data
ecog_section_a$extraction <- "eCognition"
ecog_section_b$extraction <- "eCognition"
ecog_section_c$extraction <- "eCognition"
ecog_section_d$extraction <- "eCognition"
ecog_section_e$extraction <- "eCognition"
ecog_section_f$extraction <- "eCognition"
ecog_section_g$extraction <- "eCognition"
#ecog_section_h$extraction <- "eCognition"
#ecog_section_i$extraction <- "eCognition"
#ecog_section_j$extraction <- "eCognition"

ecog_section_a$section <- "A"
ecog_section_b$section <- "B"
ecog_section_c$section <- "C"
ecog_section_d$section <- "D"
ecog_section_e$section <- "E"
ecog_section_f$section <- "F"
ecog_section_g$section <- "G"
#ecog_section_h$section <- "G"
#ecog_section_i$section <- "H"
#ecog_section_j$section <- "I"

exact_section_a$extraction <- "exact"
exact_section_b$extraction <- "exact"
exact_section_c$extraction <- "exact"
exact_section_d$extraction <- "exact"
exact_section_e$extraction <- "exact"
exact_section_f$extraction <- "exact"
exact_section_g$extraction <- "exact"
#exact_section_h$extraction <- "exact"
#exact_section_i$extraction <- "exact"
#exact_section_j$extraction <- "exact"

exact_section_a$section <- "A"
exact_section_b$section <- "B"
exact_section_c$section <- "C"
exact_section_d$section <- "D"
exact_section_e$section <- "E"
exact_section_f$section <- "F"
exact_section_g$section <- "G"
#exact_section_h$section <- "G"
#exact_section_i$section <- "H"
#exact_section_j$section <- "I"


# #adding field measurements
# field_df <- data.frame(section = character(),
#                        t_width = double(),
#                        extraction = character())
# new_row <- c("A",double(3),"field")
# field_df[nrow(data) + 1, ] <- new_row
#
# field_df %>% add_row(section = "A", t_width = 3,
#                      extraction = "field")
#
# field_section_a$section <- "A"
# field_section_a$t_width <- 3
#field_section_a$extraction <- "field"

field_df$extraction <- "field"

merged_df <- do.call("rbind", list(ecog_section_a, ecog_section_b, ecog_section_c, ecog_section_d, ecog_section_e, ecog_
    section_f, ecog_section_g,
                   exact_section_a, exact_section_b, exact_section_c, exact_section_d, exact_section_e, exact_section_f,
    exact_section_g,
                   field_df))

#delete entries which are not needed
merged_df$t_width[merged_df$extraction == "eCognition"] <- NA
merged_df$t_inclinat[merged_df$extraction == "eCognition"] <- NA


#ggplot torrent bed width
ggplot(merged_df, aes(x=section, y=t_width, fill=extraction)) +
       geom_boxplot(alpha=0.8) +
       scale_fill_brewer(palette="Paired") +
       labs(fill = "Extraction Type") +
       ggtitle("Torrent Bed Witdh [m]") +
       xlab("Section") + ylab("Torrent Width [m]") +
```

```
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999"))
        #stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")


#ggplot CS Area
ggplot(merged_df, aes(x=section, y=volume_cs, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Crosssection Area [m^2]") +
        xlab("Section") + ylab("Crosssection Area [m^2]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999"))
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")


#ggplot Embankment height diagonal orographic right side [m]
ggplot(merged_df, aes(x=section, y=e_r_diagon, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Embankment height diagonal orographic right side [m]") +
        xlab("Section") + ylab("Embankment Height [m]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999"))
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")

#ggplot Embankment height diagonal orographic left side [m]
ggplot(merged_df, aes(x=section, y=e_l_diagon, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Embankment height diagonal orographic left side [m]") +
        xlab("Section") + ylab("Embankment Height [m]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999"))
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")


#ggplot Inclincation Torrent Bed over 10 m
ggplot(merged_df, aes(x=section, y=t_inclinat, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Inclination of Torrent Bed over 10 m") +
        xlab("Section") + ylab("Inclination [?]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999"))
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")

#########

#ggplot torrent bed width
p2 <- ggplot(merged_df, aes(x=section, y=t_width, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Torrent Bed Witdh [m]") +
        xlab("Section") + ylab("Torrent Width [m]") +
        scale_fill_manual(values=c("#E69F00", "#999999")) +
        theme_bw() +
        theme(plot.title = element_text(vjust = - 10)) +
        theme(plot.title = element_text(hjust = 0.02)) +
        theme(legend.position="none",
              axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.border = element_blank())
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")

#ggplot torrent inclincation
p1 <- ggplot(merged_df, aes(x=section, y=t_inclinat, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Inclination of Torrent Bed over 10 m") +
        xlab("Section") + ylab("Inclination [°]") +
        scale_fill_manual(values=c("#E69F00", "#999999")) +
        theme_bw() +
        theme(plot.title = element_text(vjust = - 10)) +
        theme(plot.title = element_text(hjust = 0.02)) +
        theme(legend.position="none",
              axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.border = element_blank())
```

```
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")


#ggplot CS Area
p3 <- ggplot(merged_df, aes(x=section, y=volume_cs, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Crosssection Area [m^2]") +
        xlab("Section") + ylab("Crosssection Area [m^2]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999")) +
        theme_bw() +
        theme(plot.title = element_text(vjust = - 10)) +
        theme(plot.title = element_text(hjust = 0.02)) +
        theme(legend.position="none",
              axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.border = element_blank())
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")


#ggplot Embankment height diagonal orographic right side [m]
p4 <- ggplot(merged_df, aes(x=section, y=e_r_diagon, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Embankment height diagonal orographic right side [m]") +
        xlab("Section") + ylab("Embankment Height [m]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999")) +
        theme_bw() +
        theme(plot.title = element_text(vjust = - 10)) +
        theme(plot.title = element_text(hjust = 0.02)) +
        theme(legend.position="bottom",
              axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.border = element_blank())
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")

#ggplot Embankment height diagonal orographic left side [m]
p5 <- ggplot(merged_df, aes(x=section, y=e_l_diagon, fill=extraction)) +
        geom_boxplot(alpha=0.8) +
        scale_fill_brewer(palette="Paired") +
        labs(fill = "Extraction Type") +
        ggtitle("Embankment height diagonal orographic left side [m]") +
        xlab("Section") + ylab("Embankment Height [m]") +
        scale_fill_manual(values=c("#E69F00", "#56B4E9", "#999999")) +
        theme_bw() +
        theme(plot.title = element_text(vjust = - 10)) +
        theme(plot.title = element_text(hjust = 0.02)) +
        theme(legend.position="none",
              axis.title.x = element_blank()) +
        theme(panel.grid.major.x = element_blank(), panel.grid.minor.x = element_blank()) +
        theme(panel.border = element_blank())
#stat_summary(fun.y=mean, geom="point", shape=20, size=14, color="red", fill="red")


p1/p2/p3/p5/p4
```