Technische Universität München
Fakultät für Informatik
Lehrstuhl für Computer Grafik und Visualisierung

Visualization of Coherence and Variation in Meteorological Dynamics

Michael Alexander Kern

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Prof. Dr. Stephan Günnemann
Prüfer der Dissertation:	1. Prof. Dr. Rüdiger Westermann
	2. Prof. Dr. Tobias Günther
	Friedrich-Alexander-Universität Erlangen-Nürnberg

Die Dissertation wurde am 17.07.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 06.11.2020 angenommen.

To my family and friends

Abstract

In ensemble forecasting, multiple numerical simulations with slightly perturbed initial conditions are run to infer the range of possible future weather conditions. These ensemble weather forecasts are a common way to assess the predictability of weather simulation models afflicted with uncertainties due to measurement inaccuracies or approximation errors in physical models. With increasing computational power and higher model resolutions, it is progressively possible for weather centers to simulate atmospheric processes on a much smaller scale. Although numerical simulations output physical quantities on several vertical atmospheric layers, daily operations in weather forecasting still focus on the (manual) 2D analysis of weather charts for single height levels. To the best of our knowledge, forecasters lack the tools to objectively identify features in 3D and, instead, rely on established 2D detection methods. Hence, meteorologists actively look for novel techniques to examine features in terms of their 3D structure, their temporal evolution, and their relationship to other atmospheric processes. Furthermore, visualization techniques are required to assess the uncertainty of feature occurrence and structure in ensemble weather forecasts. In this thesis, we present techniques to identify 3D atmospheric features in ensemble simulations and introduce visualization techniques to depict their spatial coherence and variation. We closely collaborate with domain experts, such as forecasters from the European Weather Centre for Medium-Range Forecasts, to design appropriate detection and visualization techniques. We also demonstrate the benefit of our methods by applying our detection methods to real-world case scenarios. With this, we facilitate the full visual analysis of atmospheric features in 3D, which has not been possible before.

In particular, we propose detection and visualization techniques for two atmospheric features deemed important for weather forecasting. The first detection method deals with jet-stream core lines, which are the lines of maximum wind speed and considered the driving force for significant global and local weather events. Here, we present a robust 3D detection algorithm to extract these lines from volumetric wind fields. Although closely related to ridge detection, it avoids both extensive blurring and the computation of consistently oriented eigenvectors. Instead, our algorithm exploits the local wind direction to construct a consistent local frame-of-reference, which is used to identify the locations of wind maxima. The second detection method is concerned with atmospheric fronts. They separate air masses of different characteristics, such as temperature, and play a vital role in weather forecasting as they are associated with severe weather events. However, no automated method has yet been introduced to automatically identify fronts in 3D. Instead, forecasters still rely

on 2D approaches to detect fronts on 2D weather maps. In this work, a visualization tool is proposed to extract fronts in 3D, filter them by frontal characteristics, such as the strength of temperature and moisture gradients, and map the uncertainty of feature occurrence to opacity. Fronts are not only a single feature but rather a zone of high thermal gradients between air masses (frontal zone). Thus, a method to visualize frontal zones and to analyze the statistics of physical quantities within this zone is further introduced.

The extraction of jet stream core lines in a weather ensemble prediction yields line features of vastly different topology. Typically, these lines are disconnected, vary in orientation and length, and are not spatially aligned in 3D space. Unfortunately, the simultaneous display of such line features with so-called spaghetti plots yields cluttered images and fails to communicate the actual major and minor trends in the ensemble data. In this thesis, different strategies to cluster line sets based on their similarity and present methods to extract line representatives from each cluster are employed. For instance, clustering is applied to the input fields, which the lines have been extracted from, or directly operates on the line geometry. We also introduce an implicit line representation, called vector-to-closest-point volume, which is independent of the orientation and location of line features in 3D space. In addition to that, we derive "mean" representations from visitation maps, which denote the frequency of feature occurrences, and from central tendency fields, which describe points most central in between line sets.

If thousands of line features have to be explored for the first time, a pre-selection of features or line clustering is not always possible. Instead, users often map line characteristics to transparency to accentuate features-of-interest, while fading out lines deemed unimportant. Unfortunately, transparency rendering of large line sets requires the alpha-blending of multiple transparent layers in correct visibility order, which is a computationally expensive task. In our work, we present a comprehensive study of different GPU and CPU rendering techniques to render large line sets with transparency. We compare all rendering techniques in terms of their run-time performance, memory consumption, and image quality. Based on the results of this study, we finally discuss the suitability of algorithms wrt. the complexity of line sets and concerning different transparency settings.

Zusammenfassung

Die Ensembleprognose beinhaltet das Rechnen mehrerer Simulationsläufe mit leicht veränderten Anfangsbedingungen, um die Menge aller möglichen Wetterbedingungen in der Zukunft zu berechnen. Diese Ensembles von Wettervorhersagen werden gebraucht, um die Vorhersagbarkeit von Wettersimulationsmodellen aufgrund der zugrundeliegenden Unsicherheiten in Messungen und Approximationsfehlern in physikalischen Modellen abzuschätzen. Mit immer steigender Rechenleistung der Computer und höheren Modellauflösungen ist es zunehmend möglich, atmosphärische Merkmale auf einer viel kleineren Skala zu simulieren. Obwohl die numerischen Simulationen physikalische Größen für mehrere atmosphärische Höhenlagen berechnen, benutzt die operationelle Vorhersage hauptsächlich 2D Analysemethoden, um Wetterkarten für einzelne Höhenlagen zu untersuchen. Im Speziellen fehlen den Meteorologen Werkzeuge, um einzelne Merkmale in 3D zu detektieren, was zur Folge hat, dass in der Wettervorhersage auf etablierte 2D Detektionsmethoden zurückgegriffen werden muss. Daher sind Meteorologen an neuen Techniken interessiert, um die Merkmale in Hinblick auf ihrer 3D Struktur, ihrer zeitlichen Entwicklung und ihrer Beziehung zu anderen atmosphärischen Prozessen zu untersuchen. Außerdem werden Visualisierungstechniken benötigt, um die Unsicherheit über das Auftreten von Merkmalen und ihrer Struktur in Ensembleprognosen abzuschätzen. In dieser Doktorarbeit präsentieren wir Techniken, um atmosphärische Merkmale von Ensemblesimulationen in 3D zu detektieren, und stellen Visualisierungsmethoden vor, um die räumliche Kohärenz und Variation von Merkmalen darzustellen. Wir arbeiten dabei eng mit Fachexperten, z.B. mit Wettervorhersagern am Europäischen Wetterdienst für mittelfristige Wettervorhersagen, zusammen, um geeignete Methoden und Visualisierungen zu entwerfen. Zudem wird der Nutzen unserer Methoden dargestellt, indem wir die Detektionsalgorithmen auf echte Wetterszenarien anwenden. Dadurch ist zum ersten Mal eine neue 3D Analyse von atmosphärischen Merkmalen möglich.

Wir schlagen Konzepte zur Detektion und Visualisierung von zwei für wichtig erachtete Merkmale vor. Zunächst werden die Kernlinien des Jetstreams identifiziert, wobei die Kernlinien das Windmaximum in einem Jetstream sowohl in der horizontalen, als auch in der vertikalen Richtung repräsentieren. Die Windmaxima-Linien von Jetstreams sind die antreibende Kraft von globalen und lokalen Wetterprozessen und werden mit heftigen Wetterereignissen assoziiert. Wir präsentieren einen robusten Detektionsalgorithmus, um diese Kernlinien aus volumetrischen Windfeldern zu extrahieren. Obwohl dieser Algorithmus eng verwandt mit “Ridge Detection” ist, vermeidet er sowohl das ausgiebige Verwischen der Daten als auch die Berechnung von konsistent orientierten Eigenvektoren. Stattdessen wird die lokale Windrichtung ausgenutzt, um Windmaxima in wohlbestimmten lokalen

Bezugssystemen zu identifizieren. Zudem ermöglicht unser Tool die Analyse der Linien und deren Zusammenhang mit anderen atmosphärischen Prozessen. Die zweite Detektionsmethode behandelt atmosphärische Fronten. Fronten trennen Luftmassen mit unterschiedlichen Eigenschaften, wie z.B. Temperatur, und spielen eine essenzielle Rolle in Wetterprognosen, da sie mit heftigen Wetterereignissen in Verbindung stehen. Nichtsdestotrotz werden Fronten von Meteorologen lediglich auf 2D Wetterkarten detektiert. Nach unserem Wissenstand existieren bis zum heutigen Zeitpunkt keine Methoden, um Fronten in 3D zu erkennen. Wir präsentieren ein Visualisierungswerkzeug, um 3D Fronten zu extrahieren und sie auf Basis ihrer Charakteristiken, wie z.B. die Stärke der thermischen Gradienten, zu filtern. Die Visualisierung von Unsicherheiten beim Filtern der Fronten wird mithilfe von Transparenz realisiert. Zudem sind Fronten nicht nur harte Grenzen, sondern Übergangszonen, welche durch hohe Unterschiede zwischen zwei Luftmassen gekennzeichnet sind. Wir präsentieren daher eine Methode, um die Übergangszonen darzustellen und um Statistiken von physikalischen Merkmalen innerhalb der Zone zu analysieren.

Die Extraktion von Windmaxima von Jetstreams in Ensembleprognosen liefert Linien von stark unterschiedlicher Topologie. Typischerweise sind die Linien unterbrochen, variieren in Bezug auf die Orientierung und Länge, und befinden sich nicht in gleicher räumlicher Lage in 3D. Unglücklicherweise liefert das gleichzeitige Darstellen von Linien mit so genannten Spaghetti Plots stark überladene Bilder und ist somit nicht geeignet, um lokale und globale Trends aufzuzeigen. In dieser Doktorarbeit benutzen wir verschiedene Strategien, um eine Menge an Linien basierend auf ihrer Ähnlichkeit zu gruppieren, und wir extrahieren bestmögliche repräsentative Linien für jede Gruppe. Das Gruppieren erfolgt entweder auf den Eingabefeldern, welche zur Detektion verwendet werden, oder direkt auf der Liniengeometrie. Wir stellen zudem eine implizite Repräsentation von Linien vor, die unabhängig von der Orientierung und räumlichen Lage der Linien ist. Zudem berechnen wir künstliche Linien-Repräsentationen auf Basis von“ Visitation Maps”, welche die Häufigkeit des Auftretens eines Merkmals an einem Ort anzeigen, und von Zentralitäts-Feldern, welche die zentrale Tendenz eines Raumpunktes innerhalb des Liniendatensatzes angeben.

Werden tausende von Linien zum ersten Mal exploriert, ist eine Vorauswahl an Linien oder eine Gruppierung der Daten eventuell nicht möglich. Daher wird oftmals Transparenz eingesetzt, um interessante Merkmale hervorzuheben und andere weniger wichtige Merkmale visuell auszublenden. Das Darstellen von großen Liniendatensätzen mit Transparenz benötigt jedoch das korrekte Alpha-Blending von zahlreichen transparenten Ebenen pro Pixel in korrekter Sichtbarkeitsordnung, was einen hohen Rechenaufwand darstellt. Wir präsentieren eine umfassende Studie zu verschiedenen GPU- und CPU-Rendering Techniken, welche sich hauptsächlich auf das Darstellen von großen Liniendatensätzen mit Transparenz fokussiert. Es werden die Techniken in Bezug auf ihre Laufzeitleistung, Speicherverbrauch und Bildqualität verglichen. Anhand der Ergebnisse diskutieren wir, welche Methodiken in Bezug auf die Datengröße und Transparenzeinstellung verwendet werden sollen.

Acknowledgments

Above all, I would like to thank Prof. Dr. Rüdiger Westermann. He gave me the opportunity to do my Ph.D. at the Technical University of Munich and offered me this thesis to conduct scientific visualization of meteorological data. I also want to give him great credit for his major support, empathy, and great supervision during the entire 4 years of my Ph.D. The communication was excellent and the weekly discussion highly contributed to the design of promising work. Due to his help, I was able to successfully publish four peer-reviewed papers related to the scope of my project. Next, I also thank Dr. Marc Rautenhaus for his supervision during my studies in informatics and during the first two years of my Ph.D. I have learned a lot from him and he allowed me to advance my knowledge not only in computer science but also in meteorology. I highly acknowledge his major support at the beginning of my Ph.D. His guidance and recommendations helped me a lot to successfully manage my time and work as a research assistant.

In addition to that, it was also a pleasure for me to collaborate with a forecaster from a weather center to understand the processes of operational forecasting. Hence, I want to thank Tim Hewson from the European Centre of Medium-Range Weather Forecasts, who helped me to design detection algorithms for numerical weather data. He was also responsible for the application studies and always provided us with useful ideas and suggestions to facilitate the publication of visualization papers with a strong application in meteorology. I additionally thank the German Research Foundation (DFG) for their funding of this research project. All work in this thesis was done within the subproject "Visualization of coherence and variation in meteorological dynamics" of the Transregional Collaborative Research Center SFB/TRR 165 "Waves to Weather".

I am also thankful for Filip Sadlo, who was my first collaborator within the trans-regional research project Waves-To-Weather. He helped me to learn the basics of flow visualization techniques and, in particular, the fundamentals of ridge detection. I additionally want to give high credit to all the co-authors of my papers. They significantly contributed to the design and writing processes and thus greatly enhanced the papers' quality. Among those co-authors, I want to emphasize my former students Christoph Neuhauser and Torben Maack. They did excellent work and it was a pleasure for me to work with them. Due to the accomplishments during their bachelor thesis, we could publish a comparative study of line rendering techniques with transparency.

I further appreciate all my colleagues and thank them for a wonderful and exciting time at the

computer graphics and visualization chair. Among those are Henrik Masbruch, Dr. Alexander Kumpf, Dr. Mathias Kanzler, Dr. Marie-Lena Eckert, Martin Ender, Christian Reinbold, Lukas Prantl, Steffen Wiewel, Dr. Florian Ferstl, Dr. Johannes Kehrer, Sebastian Weiß, Kevin Höhlein, Rachel Mengyu Chu, Bianca Tost and many more. Furthermore, I want to thank Susanne Weitz and Sebastian Wohner for their support wrt. to organizational stuff and technical problems, respectively.

Last but not least, this thesis is mainly dedicated to my family and friends. I want to first thank my entire family, including my parents, my brother, my uncle, and my grandmother, for their overwhelming support and guidance, especially in stressful times. In particular, my parents always helped me with difficult decisions, provided everything I needed to successfully finish my Ph.D., and made my life much easier. Without their support, it would have been impossible for me to achieve the desired goals in my life, and above all, to obtain a doctoral degree. Thank you to Sarah Jacobs, for all her love and support to finish this thesis. Special thanks to all of my friends who supported me during this time: Stefan Föhring, Jan Krohn, Michael Kubitza, Georg Guba, Marian Izsak, Philipp Krickl, Peter Kreitmaier, Philipp Mautes, Constantin Noichl, Isabella Pomp-Noichl, Korbinian Hübner, Manuel Lechner, Julia Eingärtner, Florian Pfeifer, Philipp Fixmer, Sabine Andrä, Kathi Six, and many more. I enjoyed all the events that we spent together and you all contributed to the success of my life.

Contents

1 Introduction

Weather represents the chaotic physical processes generated by a complex and chaotic fluid system, called the atmosphere [Stu17]. The chaos in the atmosphere emanates from various interactions between a myriad of physical processes. For instance, differences in temperature (caused by solar radiation) cause pressure differences, which create strong winds. Weather forecasting is a daily practice at weather centers to predict the day-to-day evolution of the atmosphere and inherent weather patterns, and to estimate the likelihood of severe weather events, such as thunderstorms. Due to highly powerful computers at computation centers, numerical weather prediction (NWP) models are used to generate hourly weather forecasts. NWP makes use of mathematical models to (deterministically) predict the state of the atmosphere based on observed (and precomputed) weather conditions [WH06]. A NWP model is a computer simulation that takes into account a particular set of dynamic equations and initial physical parameterizations to generate deterministic weather forecasts. As NWP depends on initial weather conditions, it is an initial-value problem, which means that the initial atmospheric conditions need to be set a-priori to conduct the numerical simulation. The initial weather conditions are, in major part, based on a set of observations. Such observations are comprised of measurements from radiometers on satellites, radiosonde data, and flight level data from aircrafts [Stu17]. Unfortunately, observations are innately uncertain and can only provide an estimate of the atmospheric state. Although data assimilation schemes exist which reduce the uncertainty in observations by correcting systematic errors in measurements, a certain amount of uncertainty in the initial conditions still remains. Edward Lorenz discovered that atmospheric motions are inherently unpredictable due to the uncertainty in initial conditions. He encountered that the computation of atmospheric motion equations is highly sensitive to initial conditions, which means that small perturbations in initial conditions may lead to the amplification of errors in deterministic weather forecasts. This implies that there is a limit to the predictability of weather. Up to this point in time, it is assumed that the limit in predictability is in order of approximately two weeks [WH06].

In recent years, the increasing computational power of computers and improvements in measurements and data assimilation fostered a better understanding of atmospheric processes and enabled NWP models to simulate physical processes on a much finer scale. However, the uncertainty in models remains due to the initial-value problem in NWP and, thus, the simulated processes in medium- and long-term forecasts with slightly varying initial conditions can highly differ. Therefore, the assessment of variation and coherence, as well as a deep understanding of atmospheric processes is a key challenge in atmospheric science to further improve numerical weather models and daily forecasts in the future.

1.1 Feature Inference and Visualization in Numerical Weather Ensembles

Given the underlying uncertainty in NWP models and their sensitivity to initial conditions, ensemble forecasting is a common tool at weather forecast centers to reveal the dependency of weather forecasts on different atmospheric conditions. Here, forecast centers repeatedly run multiple simulations of the atmosphere, but with slightly perturbed initial conditions for each simulation run. This yields an ensemble of forecasts of the future atmospheric state for a certain time step. The ensemble is used to analyze the spatial and temporal evolution of possible significant weather events and indicates the likelihood of (significant) weather events. Among those weather events are the occurrence of local moving air currents with high wind speeds (jet-streams) or the zone of sudden changes in air mass characteristics (fronts). For instance, jet-streams and fronts can be sketched as 3D structures which evolve over time and are situated at different height levels of the atmosphere. As many definitions for the same features exist in meteorology, forecasters are interested in a reliable and objective location scheme to identify the 3D structure of features. This detection scheme can be used to document the interplay of these features with surrounding atmospheric processes and to assess the temporal evolution of certain weather events triggered by previous events.

With increasing complexity in ensemble simulations, however, trend inference becomes notoriously difficult. Forecasters commonly attempt to reduce the complexity of weather simulations by both restricting the analysis on single height levels and conducting a manual analysis of weather events on 2D charts. Typically, they plot several simulation outcomes along an atlas of images or display features simultaneously in a single plot, such as a spaghetti plot, to infer trends in weather forecasts. However, atmospheric features can vastly differ in location, shape, and orientation either along the vertical axis or in the entire ensemble, which makes it difficult to analyze features at all height levels and forecast runs at the same time. In addition to that, the 2D analysis of single height levels does not facilitate the analysis of the vertical evolution of features. This is an important aspect in weather forecasting as meteorologists still do not completely understand how the vertical shape structure or

evolution could have an impact on surrounding weather events at different height levels. Although NWP models simulate atmospheric processes on several height levels, forecasters at weather centers rely on established 2D detection and analysis methods. Consequently, they are still actively looking for methods to properly extract significant 3D features from ensemble simulations. Furthermore, they want to infer the coherence and variation of features in ensembles to convey minor and major trends in the weather ensemble forecast.

1.2 Feature Similarity and Clustering

Another challenge in ensemble analysis is the assessment of coherence and variation of extracted features. For instance, wind maxima in jet-streams can be extracted as 3D line features. Their occurrence is further restricted via filtering to specific characteristics, such as a minimum wind speed of 40 ms^{-1}. These filtered lines can consist of many small disconnected line segments of arbitrary topology and orientation. Since errors in NWP models are amplified over time, the location and orientation of features can highly vary for medium-range weather forecasts, that is simulation runs for 3 to 10 days after the initial simulation time. The likelihood of occurrence of features, as well as the detection of possible interruptions in shape and orientation of features, thus depends on the amount of agreement across all ensemble forecasts.

Furthermore, atmospheric features are detected based on multiple physical quantities. These quantities are represented by volumetric scalar fields produced by NWP models. Hereby, feature extraction is based either on the unprocessed raw scalar fields or new derived fields, such as gradients of wind speeds. Unfortunately, in an ensemble of scalar fields, the identified 3D features can vastly differ from member to member due to the aforementioned uncertainty in simulations. To overcome the uncertainty, forecasters thus use a simpler approach. They average the scalar fields over the entire ensemble and extract the features on the computed mean scalar fields. In ensembles of high variations, however, taking the mean is not appropriate, since smaller (local) features and outliers can be completely removed after averaging. Furthermore, averaged scalar fields may suggest long, continuous features with no interruptions, although some ensemble members indicate multiple disconnected features. And, the location of features can be influenced by averaging which causes the detection algorithm to identify features at locations slightly shifted from their “original” position.

To avoid data aggregation, meteorologists cluster the scalar fields before feature extraction to divide the field ensembles into groups of similar field characteristics. This facilitates a cluster-based analysis of ensembles, which is conducted as an operational routine at the European Centre for Medium-Range Weather Forecasts (ECMWF) [FC11] in 2D and proofed to be an important tool to understand the predictability of atmospheric processes. However, the quality of clustering highly depends on the

variation in the input fields and on the used similarity metrics between ensemble members. With high-dimensional scalar fields of high variation, clustering often fails to reveal disjoint groups of similar characteristics and, thus, does not provide satisfying results. As 3D features can also vary in shape and orientation, it is also not clear how to best represent the similarity of vastly different features. Hence, clustering of shape ensembles requires the definition and usage of appropriate similarity metrics between ensemble of shapes. Furthermore, forecasters actively look for automated methods to infer major and minor feature trends from an ensemble of high varying simulation runs with appropriate clustering methods, and to indicate outliers and statistical properties in the shape ensemble.

In operational routines at weather centers, forecasters make use of spaghetti plots to plot all extracted features simultaneously for a given time step and actively search for feature clusters in the plot. Finally, they manually deduce a mean or median representation from each cluster and consider this the most likely feature for the predicted time step. Spaghetti plots, however, suffer from cluttered visual results, especially in 3D, and hamper the interpretation of features for multiple ensemble members, also in the context of identifying possible interruptions between single features. The deduction of mean features is therefore highly aggravated by overlapping features in local cluster groups. To improve this feature deduction, approaches are required to automatically extract best feature representatives or artificial features from cluster groups.

1.3 Comparative Ensemble Visualization with Transparency

Another problem arises when a myriad of features is extracted. The high number of features is either due to multiple occurrences in the ensemble data or due to thousands of ensemble members available for a certain day. For such a scenario, the existence of groups with similar feature characteristics is not known beforehand and clustering may not be applicable to spot groups in the data. The simultaneous display of features in 3D, however, suffers from high occlusion effects and cluttered visual plots, which makes the interpretation of large feature sets impossible. Therefore, interactive visual exploration tools are required to properly examine features-of-interest. Especially in visualization, the need for efficient techniques to render and explore large line sets in 3D is still an active prominent research field. Here, major applications range from the analysis of particle paths in flow fields, or optimal transport of moving vehicles, to exploring neural connections in brains. Prior work [GRT13, LGP14, OLK*14, BGG19] has shown that rendering large line sets with transparency is a highly effective tool to reveal occluded features hidden by multiple line sets and to infer important structures in highly cluttered line sets. In terms of exploration, users typically select transfer functions to map data values along the features to transparency and color. This is useful to interactively accent important structures during the exploration, while retaining the overall contextual information about surrounding line features deemed less important.

Transparency rendering of large feature sets, unfortunately, is computationally expensive. After all objects have been projected onto the viewport grid, the correct per-pixel color and opacity of all parts of the objects falling into the same pixel has to be computed. In terms of transparency, algorithms for alpha-blending have to be performed to blend transparent objects in correct visibility order. To enforce the correct order, all transparent layers falling into the same pixel are sorted along the view direction wrt. the distance (depth) to the viewer, either in front-to-back (ascending) or back-to-front (descending) order. The pixel position and order of the transparent parts of objects can be determined in two ways: a) objects are projected and rasterized onto the viewport pixel grid (object-order) and all parts of the objects falling into the same pixel are sorted in a second pass, or b) rays are shot from the pixel centers into the scene (image-order), where the rays along their path intersect with the objects, which implicitly defines the visibility order. As the visibility order of objects to the camera is view-dependent, it has to be recomputed for each frame during rendering, making it a severe performance bottleneck in visualization. Another problem is the high memory consumption, as thousands of transparent layers have to be stored and sorted per pixel for high viewport resolutions. Hence, researchers in visualization constantly look for novel ways to efficiently render large line sets with transparency. With transparency rendering, researchers intend to keep the image quality high while reducing the costs for computation and memory requirements.

1.4 Contributions

In close collaboration with meteorologists and operational forecasters from ECMWF, several contributions are presented that attempt to handle all previously mentioned research questions. Features, such as atmospheric fronts and jet-stream core lines, exert a major impact on surrounding weather events. Yet, the meteorologists identify these features on 2D height levels only and forecasters lack the knowledge about how the 3D shape of these features is related to changes in nearby atmospheric processes. In this thesis, two detection methods are proposed to extract 3D features from scalar fields. The first is a robust method to extract jet-stream core lines in 3D using a consistent local frame-of-reference, which is orthogonal to the local wind field, to compute points of maximum wind speed gradients (c.f. [KHS*18]). The second approach extends an existing 2D automated detection method of atmospheric fronts to 3D and improves its filtering framework to better identify and visualize the borders and zone of high thermal gradients (c.f [KHS*19]). To indicate major and minor trends in an ensemble of vastly different line features, three frameworks to cluster jet-stream core lines are suggested. While one framework clusters on the derived scalar fields used for detection, the remaining two either cluster directly on the extracted feature geometry or operate on an implicit representation of lines independent of the features' topology and orientation. Alternatively to clustering, proxy fields are generated to retrieve "artificial" line features from a line set (see [KW19a]). Regarding the visu-

alization of transparency, an extensive comparative study of different rendering techniques that are able to render transparent line sets, is conducted. The algorithms are compared in terms of run-time performance, memory consumption, and image quality. (cf. [KNM*20]). In particular, the specific contributions of this thesis are detailed in the following list:

- In [KHS*18], a robust 3D detection algorithm is proposed to detect and visualize jet-stream core lines in 3D for the first time. The algorithm works differently from the original ridge detection algorithm, where maximum lines are identified by computing the first derivative along the local frame-of-reference based on the eigenvectors of the 3D Hessian. This involves the computation of second derivatives prone to noise in the data, which requires extensive blurring beforehand. In contrast to ridge detection [EGM*94, Lin98, PS08], extensive blurring of the input fields is avoided and core lines are directly extracted from the wind fields. The algorithm exploits the assumption that maximum lines are located at narrow angles to adjacent wind directions and neglects the vertical wind component as the vertical extent of the atmosphere is much smaller than the horizontal extent. Therefore, wind maxima are identified in a vertical plane perpendicular to the local wind field.
- In [KHS*19], the 2D objective front detection method by Hewson [Hew98] is extended to 3D, which provides the very first automated identification and visualization of atmospheric weather fronts in form of 3D frontal surfaces. It further proposes a framework to fuzzy filter weather fronts by mapping height-dependent thermal gradients to transparency. Furthermore, the zone of high thermal gradients in the vicinity of frontal surfaces is visualized, while its characteristics and statistics are determined and visualized using traced "normal curves" and statistical plots.
- All methods have been integrated into the open-source visualization tool "Met.3D", developed by Rautenhaus et al. [RKSW15, Rau20]. This software is highly optimized for meteorological applications and combines several established visualization techniques to interactively depict numerical weather data both in 2D and 3D. In this thesis, Met.3D was used to analyze how jet-stream core lines are related to local streamlines extracted from the wind fields and how they affect surrounding atmospheric conditions. Regarding weather fronts, a combined visualization is proposed to analyze the relationship between the vertical structure of frontal surfaces and surrounding atmospheric processes, such as nearby precipitation. The benefit of the proposed visual analysis of atmospheric features is demonstrated by means of real-world case studies.
- Regarding ensembles of jet-stream core lines, it is discussed in [KW19a] how clustering can be used to identify trends in a set of highly complex line features. Two different clustering approaches are discussed wrt. their potential to convey characteristic trends in weather forecast ensembles. The first approach clusters on derived scalar fields which are used for feature detection. Since these fields are volumetric and high-dimensional, a dimensionality reduction

method, such as principal component analysis, is utilized to preprocess the fields. Regions not containing any feature are removed from the data to further reduce the complexity of the input data. The second approach either directly clusters on line geometry using a line-specific similarity metric or computes a topology-invariant representation of line features based on a vector-to-closest-point field. All clustering approaches are compared to ridge lines extracted from proxy fields which are computed based on the line features in 3D space. In particular, two proxy fields are generated from an ensemble of lines: a) density fields indicating the frequency of feature occurrence and b) centrality fields accenting regions most central to all line features. Finally, the advantages and drawbacks of the method are discussed by means of real-world case scenarios, where all results are compared to features extracted from the best estimate of the atmosphere.

- For transparency rendering, a comparison study of exact and approximate object- and image-order rendering techniques for large transparent line sets is provided in [KNM*20]. The study is comprised of an in-detail discussion of rendering techniques with respect to several important aspects in scientific visualization. Among those aspects are real-time performance, to infer the level of interactivity, and the algorithm-specific memory consumption based on required model-specific acceleration structures during preprocessing and rendering. In addition to that, the image quality of approximate techniques is further analyzed based on different transparency and color settings, and their frame-to-frame coherency is examined during animation. Among all object-order techniques, which make use of GPU rasterization, Per-Pixel Linked List (LL) [YHGT10], Multi-Layer Alpha Blending (MLAB) [SV14], and Moment-Based Order-Independent Transparency (MBOIT) [MKKP18] are considered. For image-order techniques, based on ray tracing, voxel-based line ray tracing [KRW18], CPU-based ray tracing using the General Tubes method [HWU*19] and Embree's built-in Bezier curves [WWB*14], and a GPU-based implementation using NVIDIA's RTX ray tracing API [NVI18] are discussed. Finally, the study provides a summary of the strengths and weaknesses of each rendering technique, so that readers gain an understanding of when to use a certain technique for a given transparency setting and data size. The implementations and data sets used in this work are publicly available in [KN20, MK20].

1.5 Outline

The remainder of this thesis is structured as follows. In Chapter 2, prior work in the context of the research task related to this thesis is discussed. Among those topics are jet-stream core line detection, the objective identification of fronts, cluster-based analysis of shape ensembles, and transparency rendering. The fundamentals of all methods are explained in Chapter 3 to provide the readers with a basic understanding of the theory required and used in all published papers. All published papers and the individual contributions of the authors are summarized in Chapter 4. The findings of all papers and an outlook to future work is provided in Chapter 5. All published papers associated with the research topics of this thesis are appended at the end of this work in Chapter 7.

1.6 List of Publications

All the methods and visualization techniques proposed in this thesis have been originally published in the following peer-reviewed conference papers and journal articles:

- **M. KERN**, T. HEWSON, F. SADLO, R. WESTERMANN, AND M. RAUTENHAUS:
  "Robust Detection and Visualization of Jet-Stream Core Lines in Atmospheric Flow",
  IEEE Transactions on Visualization and Computer Graphics, 24(1):893–902. 5, 2018
  doi:10.1109/TVCG.2017.2743989
- **M.KERN**, T. HEWSON, A. SCHÄFLER, R. WESTERMANN, AND M. RAUTENHAUS:
  "Interactive 3D Visual Analysis of Atmospheric Fronts",
  IEEE Transactions on Visualization and Computer Graphics, 25(1):767-776, 2019.
  doi:10.1109/TVCG.2018.2864806
- **M. KERN** AND R. WESTERMANN:
  "Clustering Ensembles of 3D Jet-Stream Core Lines",
  In *Vision, Modeling and Visualization (2019)*, Schulz H.-J., Teschner M., Wimmer M., (Eds.), The Eurographics Association.
  doi:10.2312/vmv.20191321
- **M. KERN**, C. NEUHAUSER, T. MAACK, H. MENGJIAO, W. USHER, AND R. WESTERMANN:
  "A Comparison of Rendering Techniques for 3D Line Sets with Transparency",
  IEEE Transactions on Visualization and Computer Graphics (2020).
  doi:10.1109/TVCG.2020.2975795

2 Related Work

This chapter discusses prior work that is concerned with feature detection and ensemble visualization, and that is closely related to the work in this thesis. The ideas for our research projects, i.e., atmospheric feature detection or cluster analysis, were further governed by publications in both the visualization and meteorological community.

2.1 Jet-Stream Core Lines

The extraction of jet-stream core lines — lines with maximum wind speed along the horizontal and vertical dimension — is very similar to the extraction of extremum lines in scientific visualization. In flow visualization, for instance, ridge detection is a prominent mathematical framework to identify extremum lines in scalar fields. In meteorology, prior work has dealt with the definition of jet-stream characteristics and the identification of jet-stream core lines in 2D.

2.1.1 Line Features in Visualization

In scientific visualization, 3D, 4D scalar, or vector fields are often mapped to geometric primitives, such as lines or surfaces. These visual mappings reduce the dimension of the data while increasing information inference. In particular, lines can be extracted from vector fields to visually depict the chaotic and turbulent behavior in simulated flow fields. In contrast to 2D manifolds, lines do not suffer from self-occlusion effects and are thus features to effectively describe the motion of flow. Among the line features are lines of maximum or minimum values (extremum lines), ridge lines, vortex core lines, and separation or attachment lines. Extremum lines, for instance, represent the locations where the underlying scalar field, such as wind magnitude, is locally maximal or minimal. Ridge lines are a special case of extremum lines and are discussed in Sect. 3.2.1. Attachment and separation lines are

lines along which the 3D flow separates or attaches from or to a solid surface. These features are important concerning the aerodynamical design of an aircraft [Rot00].

Vortex core lines are a prominent line feature in flow visualization and an important component in a turbulent flow. In 3D turbulent vector fields, they can be imagined as the centers (straight or curved axis) of swirling motion around which mass-less particles are orbiting. Although vortex core lines have been well studied, there is still no general unique definition to automatically detect those. There are several variants of definitions and algorithms to identify vortex core lines. A widely used definition is taken from Sujudi and Haimes [SH95] who define vortex core lines as the loci where the vector field is parallel to one real eigenvector of the vector field's Jacobian, while the remaining pair of eigenvectors needs to be complex.

Peikert and Roth [PR99] provided a mathematical framework to identify variants of line features with the "Parallel Vectors" operator. They introduced the operator "||" as $v \parallel w$ for two n-dimensional vector fields v and w. Solutions S for the operator $\parallel$ are the set of all locations x where two vectors have the same direction: $S = \{x : v(x) = 0\} \cup \{x : \exists \lambda, w(x) = \lambda v(x)\}$. For instance, they used the operator to reformulate Sujudi and Haimes's vortex core line definition and identified vortex core lines as zero curvature loci where the velocity field v is parallel to the velocity acceleration, with $v \parallel (\nabla v) \cdot v$. For the latter, they proofed that this is equivalent to the definition of Sujudi and Haimes. In terms of the vector parallelism operator, another widely used vortex core line definition is given by Levy et al. [LDS90]. Here, one component of the vorticity vector is required to be parallel to the local velocity vector whilst the remaining two vorticity vectors are perpendicular to it. The Parallel Vectors operator can also be used to identify separation and attachment lines [KHL99] or to define bifurcation lines, which are situated at the crossing line of two 2D manifolds of streamlines, where one manifold represents the streamlines converging to the intersection line, and the other represents streamlines diverging from it [Rot00, MSE13].

Roth [Rot00] provided an overview of several algorithms that compute features with the Parallel Vectors operator. First, all lines, where the parallelism of the vectors and its derivative holds, are found at each cell and connected to lines. If these lines intersect with a cell twice, they can be connected to the actual line feature. If there are more intersections, heuristics are applied to connect those lines. Oster et al. [ORT18a, ORT18b] extended this approach to detect vortex lines in 3-dimensional tensor fields, also called Eigenvector trajectories. They formalized the "Parallel Eigenvector Operator" in a similar way to the "Parallel Vectors Operator" for tensor fields. Post et al. [PVH*03] and recently Günther and Theisel [GT18] provided an overview of existing state-of-the-art techniques to detect line features in flow visualization, including vortex core lines or separation and attachments lines in steady and unsteady vector fields.

Ridge Detection Ridges and valleys, sometimes also called “creases” or mountains, are prominent line features in scalar fields. For a 2D height field, ridges or valleys are the loci along a line where the slope — that is the magnitude of the height field derivative — is locally minimal compared to points having the same elevation [PR99]. More general for n-dimensional scalar fields, ridges (valleys) are locations where the scalar value at a local point is locally maximal (minimal) along n-1 dimensional orthogonal axes that define a cross-section cutting through the point. The first mathematical formulation of ridges (and valleys) in the context of surface topography dates back to early work by de Saint-Vernant and Breton [dSV52, Bre54].

In computer vision and image analysis, ridge detection has been evolved to an established technique and is a widely used tool to characterize and automatically identify the inner structure of objects in 2D images, complementary to edges of objects. Usually, the intensity of images is interpreted as 2D height fields, where ridges are the maxima of image intensities. The first general formulation of ridges for images was given by Haralick [Har83]. He proposed to fit a polynomial surface based on the pixel-neighborhood and compute the directional derivatives of this surface analytically to obtain ridge lines. Later, Eberly [EGM*94] reformulated the approach on 2D images and used directional derivatives along the eigenvectors of the Hessian of pixel intensities. Lindberg [Lin98] described ridges for images similar to Eberly, with the focus on the analysis and automated selection of an appropriate image scale-space. The image-scale space here represents the set of images parameterized by the size of smoothing kernels. Smoothing is typically applied to images as a pre-processing step as the computation of second derivatives and eigenvectors is error-prone to high fluctuations in scalar fields and may lead to very small-scale line features. Pizer et al. [PBC*94] looked for medial axes of 2D objects in images. They defined the medial axis as the ridge in topographical watersheds. This follows the definition of Blum and Nagel [BN78], where the medial axis is a symmetry axis dividing an object into two symmetrical parts. The impact of image-scale space on ridge detection and the existence of connector curves, which connect disjoint line segments of ridges, was further studied by Damon. [Dam98, Dam99]. Majer [Maj00] provided an overview of different feature detection algorithms and investigated the behavior of ridge line detection in different image-scale spaces. To improve the robustness of ridge line detection, Lopez-Molina et al. [LMdUB*15] proposed to operate directly on smoothing kernel functions and approximate the derivatives of anisotropic Gaussian kernels to detect ridge lines in images.

In flow visualization, ridge detection was successfully applied to detect extremum line and surface features, e.g., to identify vortex core lines [SKA99, SWH05, Sah09] or flow separation [SLM05]. It was further used by Sahner et al. [SWTH07] to visualize vorticity and strain, or by Sadlo et al. [SP07, SP09] to display separation regions of different flow in unsteady vector fields. An efficient approach to compute and filter d-dimensional ridges from n-dimensional fields was introduced by Peikert and Sadlo [PS08]. They have shown that ridge lines are implicitly given by the parallelism between the

gradient vector and the eigenvectors of the Hessian matrix.

Recently Bader et al. [BSB*20] extracted bands (extremum lines) of potential vorticity scalar fields with a 2-pass predictor-corrector algorithm. First, their approach starts with stepping into the gradient direction of the vorticity and afterward corrects the position by identifying the exact maximum (minimum) within a cross-section perpendicular to the gradient.

2.1.2 Jet-Stream Core Lines in Meteorology

The analysis of jet-stream core lines in meteorology dates back to the 50s and 60s. In 1957, Endlich and McLean [EM57] conducted fifty flights to design an empirical model of the jet-stream core lines. During their flights, they empirically measured the average wind and temperature fields near the areas of maximum wind speed and documented the results. One decade later, researchers described jet-streams and their major characteristics and provided a concept to detect jet-streams based on wind fields, i.e., as described in the books by Reiter [Rei63] and Palmen and Newton [PN69]. In particular, they introduced the concept of the "layer of maximum wind" (LMW), an approach to detect and analyze the jet-stream axis along which the wind speed is highest at a given altitude [Wor92]. The LMW method played a key role in weather forecasting in the United States [SD05] and is still used to produce significant weather charts (SIGWX) for daily flight operations.

Jet-stream events: In the context of climatology, several automated extraction methods have been developed to detect events of jet-stream occurrences wrt. large-scale weather analyses. Some authors focus on the identification of core lines, while others identify regions of major characteristics of jet-streams, like areas with wind speed above a defined threshold (usually above $20\,\mathrm{ms}^{-1}$).

Koch et al. [KWD06] examined layers between 100 and 400 hPa and vertically averaged the wind speeds between these pressure levels. Here, all horizontal grid points where the averaged wind speed exceeds a user-defined threshold, such as $30\,\mathrm{ms}^{-1}$, are marked to identify the jet-stream locations. Archer and Caldeira [AC08] extended this approach and incorporate mass to compute a mass-flux weighted vertical average of wind speed, pressure, and latitude for a more robust identification of jet-streams. Martius [Mar14] and Limbach et al. [LSW12] also used a height-dependent threshold for wind speed while Limbach used a graph-based and connected component approach to detect and track the spatio-temporal evolution of jet-stream areas in 3D. Barton and Ellis [BE09] and Pena-Ortiz et al. [POGR*13] assumed a vertical (eastward or westward) motion of jet-streams around 300 hPa and 100 – 400 hPa, respectively. Their method computes and marks for each longitudinal band the exact single latitude location of the maximum wind speed. Pena-Ortiz et al., in particular, used latitude-altitude cross-sections to locate 3D wind maxima for different pressure levels. Gallego et al. [GRGH*05] focused on the Southern Hemisphere and also assumed a vertical jet propagation. They defined the jet-stream to be the closed geostrophic streamline of maximum wind velocity averaged

in the meridional direction. The geostrophic wind, in this context, is the theoretical wind direction that results from the balance between pressure and the Coriolis force without taking into account (surface) friction.

Layer-of-Maximum-Wind: Strong and Davis [SD05, SD07, SD08] analyzed jet-streams using the notion of LMW. They assumed a zonal orientation of jet-streams and identified the location of maximum wind speeds at each single pressure level either for each latitude or along each meridian using finite differences in zonal direction. However, the line geometry was not created or used in their analysis. Manney et al. [MHD*11, MHD*14] assumed a zonal orientation and constructed latitude-altitude cross-sections to locate wind speed maxima across the zonal direction for pressure levels between 100 and 400 hPa. They also did not use the line geometry but just cataloged the locations of the jets for climatology analysis.

Rikus [Rik15] used an image-based approach to identify jet-streams. He proposed to apply minimum and maximum filters to the wind magnitudes of each latitude-pressure cross-section and to locate the maximum points where the difference between minimum and maximum images exceeded a certain threshold. Molnos et al. [MMP*17] used Rikus' method to calibrate their graph-based approach to detect continuous closed jet-streams. They interpreted the horizontal domain as a connected graph, where each neighboring grid point is connected via an undirected edge. Each edge contains several weights determined by wind speed, the angle of the wind to the edge normal, and the current location in latitude. Based on edge weights, Dijkstra's algorithm is used to compute the shortest path along the Northern Hemisphere and the line geometry is generated, based on this path, to finally obtain the jet-stream core lines.

Berry et al. [BTH07] used wind shear to detect jet-stream core lines from 2D wind fields on the isosurface along the set of points with a constant 2-PV (potential vorticity) units. Here, the total wind speed gradient is projected onto the wind normal direction and compute the wind shear. The core line is then defined as the zero-isocontour of this wind shear. Additionally, their algorithm requires all jet core line candidate points to be maximal and to exceed a certain wind speed threshold. As these criteria do not hold for all altitudes — wind speeds may increase or decrease in upper atmospheric levels and extrema get less well-defined — Spennsberger et al. [SSL17] extended this approach and introduced a new filter criterion. The product between the wind shear and wind shear gradient is computed along the normal direction. Afterward, all candidate points are identified where the wind shear gradient exceeds a certain negative threshold. Consequently, the wind maxima are well-defined, which allows to emphasize long line features.

Many of the aforementioned algorithms to detect jet-stream core lines are not Galilean invariant, only operate on specific height levels in 2D, and assume that jet-streams are always oriented in zonal direction. However, on a smaller scale, jet-streams and their core lines can be interrupted and oriented

in both zonal or meridional direction. In this thesis, we refrain from using any of these simplified assumptions and instead focus on the extraction of wind maxima lines in wind vector fields. We further regard all vertical pressure levels and provide an algorithm to identify 3D jet-stream core lines. This algorithm and the visualization of core line characteristics are detailed in [KHS*18].

2.2 Atmospheric Weather Fronts

Our work is inspired by several methods that have been published to automatically detect 2D front features in meteorology. Since there is no unique and common definition of fronts, an overview is provided to summarize all existent definitions of fronts. Eventually, prior work concerned with the automated front detection and the analysis of weather fronts is discussed in the following.

2.2.1 Definition of Atmospheric Fronts and Frontal Zones

Although fronts have thoroughly been studied for almost a century, there is still no agreement on a single unique definition of fronts. The "classical" definition considers fronts to be the horizontal boundary between two distinct air masses [Stu17]. In particular, a front separates air masses that have similar characteristics in terms of temperature, moisture, and stability [TS19b]. This traditional notion dates back to "Bergen school" published by Bjerknes and Solberg in 1922 [BS22].

Fronts: The American Meteorological Society (AMS) defines fronts as *"the interface of or transition zone between air masses of different density"* [Ame18]. The density of air depends on the parameters, such as temperature, moisture, and pressure. AMS emphasizes that in most cases the front almost always *"separates air masses of different temperature"*. The World Meteorlogical Organization (WMO) proposes that fronts are the ***"interface or transition zone between air masses of different densities (pressure, temperature, humidity).*** **1) Line of intersection of the surface separating two air masses with another surface or with the ground."** [Wor92]. Hewson [Hew98] describes the 3D boundary of a front as ***"thin layer, or non-rigid slab-like region, in three-dimensional space, within which there are [...] large horizontal gradients in the thermal characteristics."***

Frontal Zone: In meteorology, fronts are not always assumed to be a simple line or surface feature. Instead, they are defined as the transition zone of high thermal gradients, which is the zone where the thermal characteristics change the most. This zone is called **frontal zone**. According to WMO, a frontal zone is the ***"atmospheric transition layer separating two air masses in which the properties are intermediate between those of the air masses."*** [Wor92]. AMS states that a frontal zone is simply the *"transition zone"*, without any further explanation. Martin [Mar06] discovered that their length is

significantly greater than their breadth and thus provided a better understanding of the frontal zone structure. In general, the frontal zone can be regarded as the transition zone between the warm and cold-side boundaries of a front at all vertical levels, as sketched by [Hew98]. Prior work has also encountered that discontinuities of other parameters, such as wind, mainly occur at the warm-side boundary (see [Hew98, Stu17]). Hence, the "actual" front locations are commonly identified at the warm-side boundaries of the frontal zone. Fronts are further classified into two categories: warm fronts which bring warm air along with their movement into the wind direction, or cold fronts which bring cold air. Moreover, fronts are distinguished by their location on the globe. For instance, polar fronts are situated near the poles and tropical fronts are located near the equator. Researchers also actively look for fronts on small scales (mesoscale, around 5 to 500km) and larger scales (synoptic, in the order of 1000km) for weather forecasting and climatology. For further explanations, we refer to publications like [SB15, Stu17].

One popular schematic of fronts and their location is given by the "Norwegian" model from the Bergen school in 1920 [BS22] which shows the typical model of fronts around mid-latitude cyclones. The model is still used in operational forecasting and synoptic meteorology. The "Norwegian" model was later extended by the "Shapiro-Keyser" model [SK90]. The Shapiro-Keyser model uses a different definition of the structure of "occlusion processes", which are the merging processes of warm and cold fronts. An overview of models and concepts of fronts is given by Schultz and Vaughan [SV11]. Alternatively, researchers also considered vorticity maxima to identify frontal features or used the total derivative of thermal horizontal gradients as an indicator for the emergence of fronts ("frontogenesis", see [SD95, SB15]).

2.2.2 Objective Front Detection

Tracking and detection of features is important for operational weather forecasting, for example, as emphasized by Hewson [Hew98] and Schultz and Blumen [SB15]. However, no clear definition of fronts has been found, yet, mainly because fronts are identified in regions of different thermal characteristics. For instance, prior research has demonstrated that forecasters frequently recognize different front features from the same data set based on their experience, as seen in work by Mass [Mas91] and Sanders [SD95]. Schemm et al. [SRS15] compared front detection schemes based on either thermal gradients or wind discontinuities and demonstrated how the results differ when applied to the same data set. Recently, Thomas and Schultz [TS19b, TS19a] made a comparison study of different front definitions. They investigated all methods and thermal quantities used in the context of objective front detection, also in relation to global climatology, and discussed their advantages and drawbacks. They concluded that there is no best method to identify fronts.

The first formulation of a parameter to identify fronts dates back to 1936. Petterssen [Pet36] first introduced the term frontogenesis, which is the formation process of fronts over time, to facilitate

frontal analysis. He incorporated not only thermal quantities but also wind fields to identify regions of frontal zones. The frontogenesis is described as the Lagrangian rate of change of the horizontal gradient magnitude. With this definition, one can use thresholds to relate the change of magnitude to the wind fields and to manually identify potential frontal regions. Later, Renard and Clark were the first who introduced the objective detection of 2D atmospheric fronts [RC65] based on thermal quantities. They introduced the "notion" of the Thermal Front Parameter (TFP), which is the directional derivative of thermal gradient magnitude along the unit direction of the thermal gradient. An in detail discussion about TFP is given in Sect. 3.3. TFP was mainly used to detect zones at the warm or cold air boundaries of the frontal zone.

Tim Hewson [Hew98] further extended that approach and used the TFP as a filter to distinguish between the cold and warm air boundaries. His method automatically detects the boundaries of a frontal zone by using the third (directional) derivative of the thermal gradient. Fronts are also filtered by the magnitude of the thermal gradient, also known as frontal strength. Besides, Hewson sketched the general 3D structure of fronts for multiple altitudes (pressure levels) and demonstrated the evolution of the detected fronts for distinct pressure levels. However, he proposed not to detect the full 3D structure at once but to extract all fronts from multiple altitudes separately. To estimate the 3D structure, the resulting set of fronts is first mapped to a different color per altitude and then displayed at once, similar to a spaghetti plot, on a 2D weather chart. He also pointed out that computing the third derivative is highly prone to noisy data. Hence, he suggested averaging the second derivative at a local grid point. His work was further improved in Hewson and Titley work [HT10], which is still operationally run at the ECMWF to produce products for weather forecasting. Jenkner et al. [JSS*09] also made use of the TFP but only regarded the gradient magnitude maximum where $TFP = 0$. They also used smoothing to produce stable results and filtered by the frontal strength. In contrast to Hewson et al., they additionally removed stationary (= non-moving) fronts over time. Simmonds et al. [SKTB11] defined fronts by detecting temporal wind shifts in the data. They assumed that fronts are oriented zonal and identified fronts at locations where the meridional wind velocity is altered about 2, 4, or 6 ms^{-1}. Hope et al. [HKP*14] also compared 6 different methods to detect fronts, including self-organizing maps (SOMs). A SOM is a type of an artificial neural network which produces a low-dimensional representation (often a 2D map) of the input space using neurons associated with weights. Neurons are shifted so that the neurons corresponding to similar types are positioned close together, and vice versa.

In terms of climatology, several authors evaluated the automated front detection on research questions in climatology ([Kas03, BRJ11, PGM*14, SNM*16]). Most recently, Thomas and Schultz [TS19b] evaluated different objective front detection schemes and different thermal quantities. They used the approaches proposed by Hewson, Jenkner, and Petterson's frontogenesis parameter and applied them to temperature and equivalent potential temperature. Finally, they compared the results in the context

of interpretation for climatology.

Related to frontal zone detection in scientific visualization, Kniss et al. [KHGR02] ran a case study to identify frontal zones from multi-variate volumetric data. They regarded multiple physical quantities, like temperature or humidity, and mapped those quantities to color and opacity using multi-dimensional transfer functions. After this mapping, the values were rendered in 3D by using classical direct volume rendering. While domain experts could manually identify frontal zones with their method, they did not propose an automated detection of such zones.

2.2.3 Extremal Surface Features in Visualization

The detection of line features from 2D scalar fields is well-known in visualization. In the context of fronts, this would relate to extremum lines along which the local thermal gradient or the second derivative of a thermal quantity is maximal. Note that extremum line features are similar to ridge detection, which was already discussed in Sect. 2.1.1 and is further explained in Sect. 3.2.1.

In general, the extraction of surface type features in flow visualization is a common tool to identify chaotic and turbulent 3D structures from flow fields. Here, visualization techniques help to reduce the dimensional complexity of spatio-temporal scalar or vector fields and facilitates the 3D analysis of turbulent (non-)steady flow fields. Among those features are the traces of curves, which emanate from constantly seeded particles released into the flow from a given curve line. All particles seeded from this curve are tracked and sequentially connected over time to form a surface. Among these surfaces are stream, streak, smoke, or time surfaces. An overview of the extraction of such features is given by Edmunds et al. [ELC*12].

Related to 3D front features, our work is similar to the extraction of 3D ridge surfaces from scalar fields. For ridge (valley) surfaces the definition of ridges is more relaxed and only one orthogonal axis has to contain a local maximum (minimum) in a 3D scalar field. Furst et al. introduced the “marching cores”algorithm [FPE96] to find 2D ridge surfaces (creases) in a 4D space, where the 4th dimension is scale space. Furst and Pizer extended their approach with “marching ridges” [FP01] to extract one or two-dimensional ridges from high-dimensional data. They assumed that ridges only intersect a cell twice and orientation of eigenvectors is always given. They also made use of principal component analysis (PCA) to estimate the local orientation of the eigenvectors. Ridges are found by tracing sign changes of the directional derivative, which is characterized, here, by the product between an eigenvector and the scalar field gradient. In visualization, ridge surface detection has been applied to specific 3D volumetric data. Kindlmann et al. [KTW07, KESW09] extracted ridge surfaces to find skeletal structures in data from diffusion tensor Magnetic Resonance Imaging (MRI). They used the classical “marching cubes” algorithm [LC87] to extract surfaces. The directional derivative field is considered to be the scalar field for ridge detection. Their approach can then trace eigenvectors along

cell edges and can impose an eigenvector orientation at each cell. Sadlo and Peikert [SP07] also used "marching cubes" on an adaptive grid to identify ridge surfaces in unsteady vector fields. Their ridges define regions that separate flow areas of different characteristics. To define the orientation of eigenvectors, they used the definition from "marching ridges". Sahner et al. [SWTH07] made use of ridge detection to extract 1D and 2D vortex and strain skeletons. In contrast to classical ridge detection, they used a watershed-based definition for feature extraction. Schultz et al. [STS10] also operated on diffusion tensor MRI data and took into account the non-orientability of eigenvector systems, especially in cases where the Hessian matrix degenerates, that is lines where two eigenvectors are equal. They used the Hessian degeneracy to define boundaries of ridge surfaces and presented an algorithm to find intersection points of ridges and their boundaries. Finally, normals were estimated to identify 2D ridges. A more recent overview of the detection and efficient rendering of extremal surfaces in 3D fields is given by Kindlmann et al. [KCH*18].

In computer graphics and visualization, ridge detection has been used as a tool to derive specific extremal surface features. For instance, Süssmuth and Greiner [SG07] used ridge surface detection to reconstruct meshes from noisy point cloud scans. They extracted ridges from a precomputed density function of the points, which was generated by applying a smoothing kernel to the point cloud. Ferstl et al. [FBTW10] identified separating streak surfaces in time-varying flow fields with the help of ridge detection. They applied ridge line detection to finite-time Lyapunov exponent (FTLE) fields and considered the identified ridge lines to be the seeding curves for the streak surface generation.

In the meteorological community, to the best of our knowledge, there has not been proposed any automated detection scheme that identified front features for all vertical pressure levels in 3D. Regarding the prior work from flow visualization, ridge surface detection is one option to identify maxima lines from first- or second-order thermal gradient fields. However, ridge surface detection would require the computation of third- or fourth-order derivatives of a thermal quantity, for example, with finite differences. This would accumulate approximation errors and would yield cluttered features without the use of extensive blurring.

In this thesis, we adapt the algorithm of Hewson [Hew98] to identify 3D frontal surfaces and propose a visualization tool to plot the characteristics of the front along its surface and its corresponding frontal zone. For front detection, we focus on thermal quantities that best represent changes both in temperature and magnitude. We also propose to smooth the data beforehand using a Gaussian smoothing kernel with weights adapted to the distance of grid points per latitude. An in-detail discussion of our algorithm is provided in [KHS*19].

2.3 Clustering-Based Ensemble Visualization

The cluster-based analysis of ensembles of shape is closely related to ensemble visualization, a branch of the more general field of uncertainty visualization. As already explained in Chapter 1, uncertainty in the data occurs due to approximate simulation models with erroneous initial conditions. The uncertainty of the model is covered by computing an ensemble of simulation runs with slightly perturbed initial conditions. Several surveys exist that list the most important techniques in ensemble visualization, up to this point in time. For this, compare the surveys by MacEachren et al. [MRH*05], Bonneau et al [BHJ*14], and Potter et al. [PRJ12]. Another overview of ensemble and multi-dimensional visual data analysis is given by Love et al. [LPK05], where appropriate statistical distribution and shape descriptors for scalar-valued ensembles are discussed. Alternatively, Kehrer and Hauser [KH13] proposed different visualization techniques for multi-variate, spatio-temporal varying data.

2.3.1 Ensemble Visualization

Prior work from the last years has proposed methods to infer major and minor trends in ensembles of spatio-temporal varying ensembles of scalar and vector-valued fields. For instance, in terms of closed isoline ensembles, Whitaker et al. [WMK13] suggested replacing spaghetti plots, which often hamper trend inference due to occlusion effects, with contour box plots to visually emphasize confidence regions in the data. Mizargar et al. [MWK14] extended contour box plots to arbitrary curves in 2D and 3D. They measured the depth (centrality) of a single curve shape within an ensemble of shapes to determine band depths in the data. Demir et al. [DJW16] introduced translation and rotation-invariant closest point representations to visualize the central tendency of multi-dimensional ensemble of shapes in 3D. Similar to our work, the median of the closest point volumes is computed to produce a mean representation of the shape ensemble. Bruckner and Möller [BM10b] proposed the computation of signed distance functions to analyze different iso-contours of the same scalar field.

In some prior work, Gaussian mixture models (GMM) or mixtures of probability density functions were used to visualize time-varying data and its evolution in distribution. To name a few, Liu et al. [LLBP12] represented 3D volumes with per-voxel distributions using GMMs and made use of a Monte-Carlo approach to render volumes with GMMs. Jarema et al. [JDKW15] proposed lobular glyphs based on the GMMs of vector-valued ensemble data. Dutta et al. [DS15] performed feature classification and tracking based on modeling of mixture Gaussian distributions. Wang et al. [WLW*17] computed the probability density function of values and their occurrence probability using Spatial GMMs. Demir et al. [DDW14] suggested transforming 3D data points into a linearized vector representation to visualize the points' distributions with bar charts. Pfaffelmoser et al. [PRW11] extended the approach of Pöthkow and Hege [PH11] to visualize probabilistic 3D isosurfaces with volume ray casting. Here, they assumed a Gaussian distribution, accounted for homogeneous and anisotropic correlations, and

determined the probability of isosurface crossing using a stochastic distance function. The geometrical and positional uncertainty of an isosurface is computed by measuring the distance to the mean isosurface in standard deviation units (normal curves). Pöthkow and Hege [PH13] and Athawale et al. [ASE16] allowed non-Gaussian distributions in ensemble data and used location-wise kernel density estimators of non-parametric distributions to measure the data spread in surface and vector-valued fields features while incorporating correlations in the data with principal component analysis. Hazarika et al., see [HBS18] and very recently in [HDSC19], presented a copula-based framework for large multivariate data sets, where the domain is partitioned and statistical quantities are computed over the partitions.

Furthermore, there has been more work on the visualization of statistical summaries of the data, such as from Love et al. [LPK05] who used parametric distribution models or distance metrics (here Kullback-Leibler divergence) to visualize multi-variate data. A summary of different plots to visualize statistical data and their uncertainty is provided by Potter et al. [PKRJ10]. Hoellt et al. [HMZ*14] introduced a visual analysis system that explores ensemble forecasts to predict the surface of an ocean. Here, statistics are computed to visualize the spatio-temporal uncertainty and probability of ocean surfaces in 3D.

2.3.2 Cluster-Based Visualization of Shape Ensembles

The simultaneous display of ensemble of shapes, especially for a myriad of lines and surfaces, increases the visual complexity in 3D plots due to occlusion effects of several complex surfaces. This hampers the exploration of ensembles wrt., for instance, local feature inference. Clustering is one approach to reduce the complexity of such shape ensemble plots. Here, the features are first divided into groups of similar characteristics. Afterward, visual mappings can be applied or statistics can be computed to further analyze every single group or line set. In statistics, clustering is a well-known effective tool to detect groups of samples with similar characteristics in a data set. The similarity between different samples depends on the underlying similarity metric, such as the Euclidean or Mahalanobis distance. An introduction to clustering is provided by books like in Everitt et al. [ELL09].

In terms of shape ensembles, Bruckner and Möller [BM10a] used density-based clustering to identify similar volumetric time sequences in physically-based ensemble simulations. Thomas and Natarajan [TN14] computed shape descriptors of isosurface contours and measured the similarity between these descriptors to cluster the data. Oeltze et al. [OLK*14] compared k-Means, hierarchical, and spectral clustering to cluster streamlines and chose the best representative streamline of a cluster based on the shortest distance to all other cluster members. Zhang et al. [ZHQL16] and Moberts et al. [MVv05] provided overviews of suitable similarity metrics to measure the distance between the geometry of line curves. Clustering using the Hausdorff metric, or mean vertex-wise closest point distance, was ap-

plied in prior work to group streamlines [RT12] in flow-fields and fiber-bundles [BBP*05] in diffusion tensor imaging.

Beham et al. [BHGK14] used hierarchical clustering to group similar geometric shapes. Reh et al. [RGK*13] clustered similar pores in industrial X-ray computed tomography data into mean objects (Mobjects) and then visualized the per-voxel probability of belonging to a Mobject using transfer functions. Hummel et al. [HOGJ13] proposed minimum spanning trees to compare the parcel transport in flow ensembles for trend inference. Clustering was further used by Carr et al. [CDD06] to group iso-contours in scalar fields.

Ferstl et al. [FKRW16, FBW16] proposed to visualize ensembles of closed isocontours and streamlines with variability plots. To group similar shape ensembles, they made use of principal component analysis and clustering and estimated confidence regions with signed distance functions or multivariate normal distributions. In [FKRW17], different time steps of the same ensemble were clustered with hierarchical clustering to convey the temporal change of clusters in ensembles. Demir et al. [DKW16] rendered silhouettes of an ensemble of isosurfaces and used clustering to show members of similar contours. Recently, Favelier et al. [FFST19] visualized the spatial variability of critical points in ensembles based on statistics on their occurrence.

2.3.3 Cluster-Based Analysis in Meteorology

A general overview of clustering techniques and applications in meteorology is given by Wilks [Wil11], while Nocke et al. [NSB04] discuss some visualization techniques for clustered climate data. Operational clustering is employed in weather forecasting at the ECMWF, as described in [FC11]. In particular, three different time windows for a static domain are used to cluster an ensemble of scalar fields of geopotential height. The results are visualized in matrix plots, which are comprised of forecast maps of all cluster representatives. In meteorology, cluster-based analysis of ensembles has been successfully employed to improve the understanding of various aspects in terms of atmospheric predictability. For instance, Frame et al. [FAGM11] clustered jet wind profiles using k-means clustering, and compared the clusters to pre-identified weather-regimes to observe the climatological probability of eddy-driven jet events over the North Atlantic. Strehl and Ghosh [SG02] applied different clustering techniques to one single ensemble and combined the results to one single cluster result. Bordoloi et al. [BKS04] proposed realization- and distribution-based hierarchical clustering to reduce the amount of information to be visualized. More Recently, Kumpf et al. [KTB*18] used multiple k-Means clustering on ensemble fields with slightly varying clustering domains and analyzed the robustness of clustering results.

In this thesis, we present a framework to cluster ensembles of jet-stream core lines. Clustering is either applied directly to the extracted core line features or applied to the derived scalar fields

used to identify core lines. We also propose methods to extract best representatives of core lines by computing proxy fields derived from the ensemble of extracted core line features. An in-detail discussion of clustering and the extraction of best representatives is given in [KW19a].

2.4 Rendering with Transparency

Techniques to render data sets with transparency have been well studied in the visualization community for the last decades. Some surveys (cf. [MCTB11, Wym16]) exist that deal with the performance and image quality of rasterization-based transparency rendering techniques. Transparency rendering, in general, represents the approach to render multiple transparent layers onto the viewport pixel grid in correct visibility order. Here, the transparent layer (or a fraction of the object's geometry), which is projected and rasterized onto a pixel of the viewport, is called a fragment.

Vasilakis et al. [VVP20] recently discussed different algorithms that work on multiple fragments per pixel to employ rendering techniques, such as order-independent transparency. However, they mainly focused on rather simple scenes in the context of real-time graphics effects for games. The scenes consist of a small number of transparent layers, with homogeneous colors and a few spatially extended objects, which makes the depth complexity (= the number of transparent layers in a scene) rather low. In our work, however, we deal with data that combines thousands of transparent line sets with high depth complexity. Hence, it is difficult to assess the suitability of the surveyed algorithms in [MCTB11, Wym16, VVP20] for our complex line sets with thousands of transparent layers and varying colors. To the best of our knowledge, no comparison study has been done yet to infer the suitability of transparency rendering techniques wrt. rendering of large line sets with transparency.

Among the many algorithms that exist to correctly render transparent line sets, we list the prior work which is closely related to the comparison study in this thesis. In the following, we classify the algorithms into object- and image-order techniques.

2.4.1 Object-Order Techniques

Object-order approaches mainly work on objects projected onto the viewport pixel grid and the fragments produced by rasterization. All fragments that fall into the same pixel have to be sorted in correct visibility order and must be finally blended to obtain the final color. One of the very basic algorithms was introduced by Everitt [Eve01]. He proposes Depth Peeling, where two depth-buffers are used to successively render the sequential depth layers of a transparent object. Depth in this context means the distance of a layer to the viewer. While the first depth-buffer keeps track of the closest objects in a scene, the second depth-buffer stores the information about the closest fragments from the last iteration. The second depth-buffer is particularly used to cut off fragments from all previous depth

layers. While this algorithm is exact, rendering thousands of layers is highly time-consuming and, thus, does not guarantee results in real-time. In earlier work, Carpenter [Car84] introduced the A-Buffer. It is comprised of data structures to store an unordered set of fragments and to determine the amount of area that a fragment is covering (pixel coverage). The fragments falling into the same pixel are sorted by depth and averaged based on the coverage mask. An extension of the A-Buffer and a very popular technique nowadays are per-pixel linked lists (LL), introduced by Yang et al. [YHGT10]. Here, fragments of arbitrary amount are gathered in a buffer on the GPU where fragments falling into the same pixel are connected via linked lists while the pointer to the first fragment of a pixel is stored in a second buffer. After all fragments have been gathered, they are sorted and blended in correct order. With LL, however, the fragment buffer can easily exceed video memory bounds, as the number of fragments is not limited. This makes LL not suitable to render many transparent layers at high viewport resolutions.

To overcome this issue, researchers suggested algorithms that operate on bounded memory only and focus on a fixed number of fragments per pixel. One exemplary work is the k-Buffer, proposed by [BCL*07], where only the k fragments closest to the camera are stored and blended per pixel. Here, fragments are merged heuristically if there are more than k transparent layers per pixel. Similar to the k-Buffer, Salvi et al. [SML11] introduced Adaptive Transparency which also assumes bounded memory and takes into account k fragments per pixel. Salvi et al. created a compressed visibility representation by pre-computing the transmittance function per pixel during fragment merging while keeping the error of compression low. Their algorithm keeps track of the correct order of fragments by inserting them based on their depth in a list, which sorts fragments in back-to-front order. An extension of this approach is called Hybrid Transparency, proposed by Maule et al. [MCTB13] where the set of fragments for a pixel is split into a core and tail part. For a total of n pixel fragments, a subset of k nearest fragments is sorted and blended in the core while the $n-k$ remaining fragments in the tail are blended and merged heuristically. Core and tail are finally merged with the background color. During our experiments, we discovered that this approach has severe problems with scenes comprised of thousands of transparent layers. In our study, we propose a solution where a scene is divided into depth buckets and the merge heuristic is adapted to better handle large line sets. Salvi and Vaidyanathan [SV14] proposed Multi Layer Alpha Blending (MLAB) where only a very few fragments are handled per pixel. Here, k nodes (combined fragments) are sorted in a blending array comprised of color, depth, and transmittance along a view. If the blending array is fully occupied with k nodes, they merge incoming fragments that are close in terms of distance and transmittance.

Fourier opacity mapping was introduced by Jansen and Bavoil [JB10] where the light transmittance in participating media is approximated by a low-frequency distribution, here a Fourier series, dependent on the depth. This was used to efficiently compute shadows, however, colors were ignored. Recently, Rojo et al. [BGG19] adapted Fourier opacity mapping for importance-based transparency

rendering and also considered colored fragments in their Fourier approximation. Münstermann introduced Moment-Based Order-Independent Transparency [MKKP18], where the per-pixel transmittance is approximated by power moments. They also used a logarithmic scale for the light absorbance to enable additive compositing of fragments and thus to enforce order-independence implicitly.

Some work dealt with rendering transparent layers using multiple samples per pixel, for instance, to create a smoothed, anti-aliased rendering result. For instance, Stochastic Transparency [ESSL10] is a multi-pass approach using random sub-pixel patterns and the fragment's opacity to compute weights based on their estimated pixel coverage. Consequently, fragments are rendered and blended based on their final alpha-to-coverage. McGuire and Bavoil [MB13] proposed Weighted-Blended Order-Independent Transparency and focused on an improved alpha-compositing of fragments. For the alpha-compositing, they computed weights based on pixel coverage and the distance to the viewer. Other methods are Stochastic Layer Alpha Blending (Wyman [Wym16]) and Phenomenological Transparency (McGuire and Mara [MM17]). Wyman proposed to insert fragments based on a coverage mask and the probability that a fragment is visible based on this mask. McGuire and Mara included the approximation of physical processes to obtain realistic effects of transparent objects. However, these techniques do not scale well performance-wise with an increasing depth complexity and are, thus, not suitable for large line sets.

Alternatively, particle-based approaches, such as [SK12], and voxel-based rendering techniques, as proposed in [CNLE09, LK10], have been introduced to render opaque and transparent geometry. However, these methods are not suitable for highly space-filling line sets as they effectively increase the number of render-able primitives and require modifications to account for fine-detailed geometry and sharp outlines. Hence, we did not consider them further in our evaluation study.

2.4.2 Image-Order Techniques

In contrast to rasterization-based techniques, image-order approaches operate on the viewport's pixel-grid and traverse translucent media along the rays from the viewer's pixel grid into the scene through the pixel centers. This is called ray casting or ray tracing. Recent developments and advances in GPU hardware have shown that ray tracing has become a serious alternative to rasterization in terms of real-time rendering, especially in scenes with multiple triangulated models. In the last decade, a lot of improvements have been achieved in this field, for instance, with enhanced space partitioning schemes, traversal algorithms, or efficient boundary volume hierarchies, such as [WIK*06, WMS06, WJA*17] for CPU, and [AL09, LGS*09, LK10, PBD*10] for GPU. Wald et al. [WKJ*15] further used ray-tracing in combination with tree-based search structures to efficiently locate particles and ray-particle intersections. Recently, Kanzler et al. [KRW18] proposed a voxel-based GPU rendering technique, called voxel ray casting, to render large 3D line sets with global illumination effects and transparency. They approximated line sets with voxel-based representations of compressed size and used voxel

ray tracing to efficiently render line sets. For voxelization, line-voxel intersections are quantized at the voxel faces and discrete locations of these intersections are computed to enable high traversal efficiency. The voxelized line models also serve as acceleration structures to visualize line sets with global illumination effects, such as ambient occlusion.

Instead of using voxel-based approximations, line sets can be modeled as analytical or polygonal tubes. Acceleration structures such as kD-trees or bounding volume hierarchies can be additionally used to efficiently determine locations of ray-tube intersections. In this context, OSPRay [WJA*17] and OptiX [PBD*10] have been introduced for CPU and GPU, respectively, and are prominent tools to perform ray tracing. OSPRay builds upon Intel's Embree ray tracing kernel functions [WWB*14], which has integrated support to render lines with fixed radii or Bézier curve primitives. Han et al. [HWU*19] extended OSPRay and supported the rendering of general tube primitives with varying radii, bifurcations, and transparency. Very recently, NVIDIA introduced the RTX ray tracing API [NVI18] based on OptiX to conduct hardware-accelerated ray tracing on the most recent graphics cards. In particular, RTX hardware supports ray tracing applications with hardware-accelerated ray-triangle intersection tests and intelligent shader re-scheduling of individual compute branches for rays of varying computational workload. Optix and RTX also provide an interface to implement custom intersection shaders for arbitrary geometry which can be used to intersect rays with analytical tubes. In our work, we compare the three major variants of image-order techniques, including voxel ray casting, CPU ray tracing (using Embree and OSPRay), and RTX ray tracing.

In [KNM*20], several object- and image-order techniques to render line sets of varying sizes with different transparency and color settings are evaluated. The size of our line sets ranges from 10000 lines to 400 K highly dense trajectories. Each technique and its results are compared wrt. performance, memory consumption, and image quality, and their frame-to-frame coherency is analyzed. In addition to that, two improvements for LL and MLAB are provided. The former technique is enhanced with an improved fragment sorting strategy, such as with shell sort and quick sort. The latter divides the scene into depth buckets and performs MLAB on these buckets separately to enhance the final image quality.

3

Fundamentals and Methods

In this chapter, we briefly discuss the theory and basics of the concepts and techniques either used in our published papers or closely related to our proposed techniques. Among those are feature detection algorithms, the definition of normal curves, dimensionality reduction, clustering, and alpha compositing. First, we discuss the meteorological data and its characteristics used within the context of our research projects.

3.1 Numerical Weather Data

For feature detection and visualization of weather ensemble data, we make use of NWP data from the ensemble prediction component (ENS) of the Integrated Forecast System (IFS) [LP08]. It provides an unperturbed control forecast and 50 ensemble members perturbed in the initial state of the simulation. For jet-stream core line and weather front detection, we retrieved NWP weather ensemble data from the Meteorological Archival and Retrieval System (MARS) for an area over the North Atlantic and Western Europe on the Northern Hemisphere. Our NWP data is defined on a so-called hybrid-sigma model levels, where the vertical levels are determined by a hybrid sigma-pressure coordinate system (introduced by Simmons et al. [SB81, SS83]). We preferred hybrid-sigma model levels over regular pressure levels for feature detection for the following reasons. In real-world cases, for instance, pressure levels do not follow the terrain's surface, especially over mountains, but, instead, can intersect with the terrain, such as with mountains (cf. orange dotted line in Fig. 3.1). They thus require the incorporation of additional boundary conditions during the simulation of near-surface physical processes. Hybrid sigma-pressure model levels, in contrast, avoid the boundary conditions and allow the definition and simulation of physical fields, such as temperature or wind, along smooth terrain-following levels. Furthermore, studies have shown that an increased vertical resolution at the lowest boundary of the atmosphere, referred to as atmospheric boundary layer, is helpful to bet-

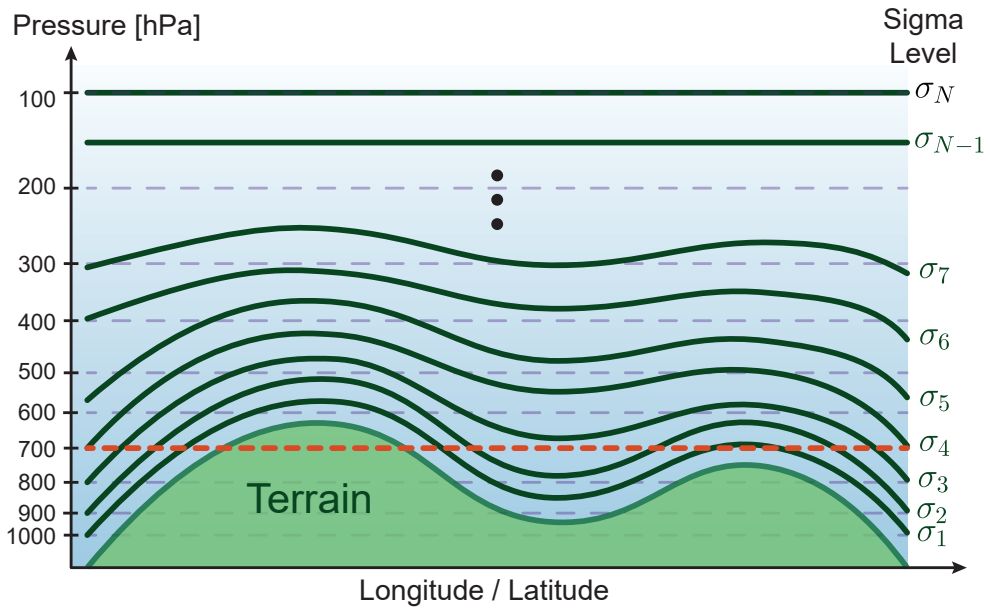


Figure 3.1: *Sketch of the hybrid sigma-pressure levels and regular pressure levels in NWP data used by the ECMWF models. The vertical axis represents the altitude, here represented by pressure in hPa. The green section represents the surface topography. Dark green bold lines depict the altitude of all N sigma-pressure levels σ_i. Regular pressure levels are depicted by purple dotted lines. Here, the mismatch between sigma-pressure levels and pressure levels can be observed. While the former levels mainly follow the terrain at vertical layers close to the surface, pressure levels remain at the same altitude and intersect with the terrain (compare red dotted line at 700 hPa).*

ter capture atmospheric processes important for (seasonal) forecasts of processes, such as winds or sea-flux influenced by the surface terrain or sea (see [Tei99]).

The horizontal resolution of the NWP data is regular in longitude and latitude, which means that adjacent grid points are equidistant. In our work, the horizontal distance of the data is 1.0°, which is approximately 111 km. Regarding the vertical resolution, the data has a fixed number of vertical layers K defined on the terrain-following sigma coordinates. Sigma coordinates are defined through a set of coefficients and surface pressure at a certain longitude-latitude grid point. They also consider the exponential decrease of pressure with increasing distance to the Earth's surface. Hence, the distances between adjacent vertical layers increase exponentially with height and are thus non-uniform. As the irregularity in the vertical dimension needs to be considered in our detection schemes, for instance, in terms of derivative computations, we will briefly sketch the theory behind hybrid sigma-pressure grids in the following. Hereby, we mainly follow the work of Ritchie et al. [RTS*94] and of Untch and Hortal [UH04].

At each horizontal grid point, let k be the current vertical layer. Further assume that we know the current surface pressure p_s at the surface grid point below the layer k. Let a_k and b_k be predefined

constant coefficients, also called hybrid coefficients. Then, the actual pressure p_k at level k is defined as $p_k = a_k + b_k \cdot p_s$. This means that the influence of the surface pressure is high near the lower bound of the atmosphere, with the strongest influence at the surface (given $b_k = 1$ and $a_k = 0$). For upper vertical levels, the influence of p_s, represented by b_k, decreases with height for the upper atmospheric layers and becomes zero at the top of the atmosphere. The coefficient a_k exerts more influence on sigma levels at upper atmospheric heights. This implies that at altitudes where $b_k = 0$, i.e., for vertical levels at $P \geq 200\,\text{hPa}$, the sigma levels rather resemble flat constant pressure levels. The shape of terrain-following hybrid sigma-pressure levels is depicted in Fig. 3.1.

In our work, we also compared results from our clustering and proxy geometry approaches to the best estimates of the atmosphere produced by the ERA5 re-analysis method(see [HBB*19]). ERA5 data is provided in high-resolution pressure levels, with a regular grid in the horizontal while using constant pressure levels for the vertical dimension.

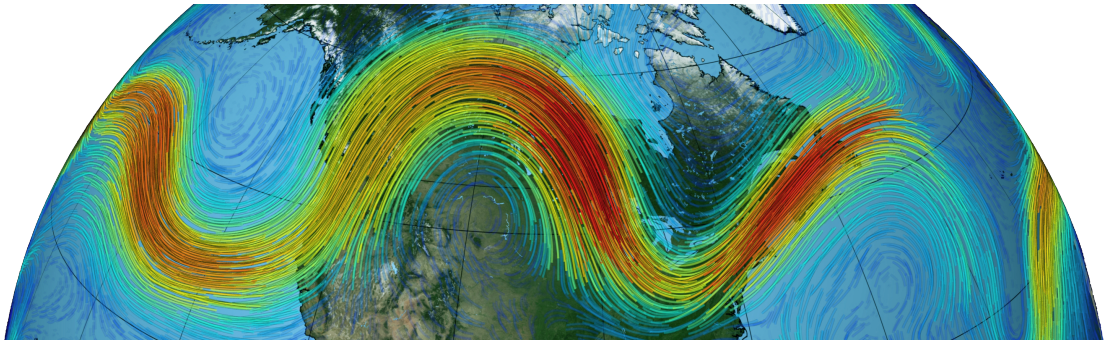


Figure 3.2: *Visualization of the polar jet-stream over North America. The colored streamlines indicate the motion of airflow around the Northern Hemisphere. Color encodes the wind magnitude (blue is the lowest, red is highest wind speed). This figure was adapted from [Sch20], where the visualization was created by NASA's Scientific Visualization Studio. The Blue Marble data is courtesy of Reto Stockli (NASA).*

3.2 Detection of Jet-Stream Core Lines

Jet-streams are meandering air currents of high wind speeds moving from west to east [Stu17] over the entire globe. Their high wind speeds range from 25 to 100 ms^{-1}. The WMO defines jet-streams as ***"flat tubular, quasi horizontal, current of air generally near the tropopause"*** [Wor92]. The tropopause is the lower atmospheric layer located at altitudes ranging from 9 km at the poles to 17 km at the equator [Wor92]. Two common strong jet-stream systems are the polar jets, near 50° to 60° latitude, and subtropical jets near 30° latitude [Stu17]. An example of a polar jet system is indicated in Fig. 3.2.

The axis of a jet-stream is located ***"along a line of maximum speed and which is characterized by great speeds and strong vertical and horizontal wind shear"*** [Wor92]. Its core line is determined as the ***"line along which the wind speeds are maximum both in the vertical and horizontal"*** [Wor92]. In our work, we want to follow the official WMO definition and in [KHS*18] present an automated detection algorithm to identify the lines along which the wind speed is maximal both in the horizontal and vertical direction. This algorithm is closely related to the "classical" ridge detection algorithm, which can be employed to identify extremum lines from scalar fields. Hence, ridge detection can be applied to wind magnitude fields to obtain jet-stream core lines. In the following, we discuss the fundamentals of ridge detection and explain the basics of our jet-stream core line detection algorithm.

3.2.1 Ridge Detection

The detection of extremum lines has been well studied in visualization. In computer vision, ridge detection is an established and effective tool to detect the interior or symmetrical axes of closed object boundaries [PBC*94], and is employed in flow visualization to identify extremum and vortex core lines [PS08]. A widely used approach in computer vision and flow visualization is to interpret images

and volumetric data as height fields. The extremum lines are represented by the ridges of the corresponding height field. In terms of topographic ridges, this represents all loci where the slope of the surface is locally minimal compared to surrounding points on the same elevation level [PR99]. Haralick formalized ridges as the loci along the path in the direction of the largest second directional derivative [Har83]. The definition of k-dimensional height ridges was reformulated by Eberly [EGM*94] and Lindeberg [Lin98], and later extended by an alternative formulation, the parallel vectors [PR99]. In principle, ridge detection makes use of the first derivative and the Hessian matrix of a scalar field, where a height ridge can be regarded as a local maximum with relaxed criteria. For instance, height ridges are located where the scalar field is maximal in at least one direction. For extremum lines in wind fields, we are concretely looking for 1D ridge lines in 3D scalar fields, where the scalars at each grid point denote the local wind magnitude.

Let $\sigma : \mathbb{R}^3 \rightarrow \mathbb{R}$ be the function of our scalar field mapping a 3D position $\vec{p} = (x, y, z)$, defined on a Cartesian grid, to the corresponding wind magnitude $\sigma(\vec{p})$. Furthermore, let $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})$ be the vector of the first-order derivatives of a function into the orthogonal directions $\vec{x}$, $\vec{y}$, and $\vec{z}$. The 3×3 Hessian matrix of the scalar field is defined as $H := \nabla\nabla^T\sigma$, with its eigenvectors $\vec{\xi}_i$ and their corresponding eigenvalues $\lambda_i = \nabla^2_{\vec{\xi}_i}\sigma$. Note that the eigenvectors form an orthonormal basis as the Hessian H is symmetric. The directional derivative of a function σ into the direction of $\vec{s}$ is described as

$$\nabla_s\sigma = (\vec{s} \cdot \nabla)\,\sigma = s_x \frac{\partial \sigma}{\partial x} + s_y \frac{\partial \sigma}{\partial y} + s_z \frac{\partial \sigma}{\partial z}, \tag{3.1}$$

where $\vec{s}$ is a unit vector. Following the definition of Eberly [EGM*94], the eigenvectors ξ_i of the Hessian H are sorted by the corresponding eigenvalues in ascending order, with $\lambda_1 \leq \lambda_2 \leq \lambda_3$. Moreover, Eberly defines that ridge lines, extracted from a 3D scalar field, are located within the local-frame spanned by the two eigenvectors $\vec{\xi}_1, \vec{\xi}_2$ corresponding to the two smallest eigenvalues. The scalar field is derived into the direction of $\vec{\xi}_1$ and $\vec{\xi}_2$ and height ridges are thus defined at all points $\vec{p}$ where the following two equations hold:

$$\nabla_{\xi_1}\sigma(\vec{p}) \;=\; \nabla_{\xi_2}\sigma(\vec{p}) \;= 0 \tag{3.2}$$

$$\lambda_1, \;\; \lambda_2 < 0. \tag{3.3}$$

In the notion of parallel vectors [PR99], height ridges are equivalently defined as the loci where the first derivative of the scalar field is parallel to the eigenvector $\vec{\xi}_3$, which implies that $\nabla\sigma(\vec{p}) \,||\, \vec{\xi}_3$ and eq. 3.3 are fulfilled. Here, ξ_3 is considered to be parallel to the tangent of a ridge line. The remaining two eigenvectors $\vec{\xi}_1, \vec{\xi}_2$ are assumed to be orthogonal to the height ridge and span a 2D local $(\vec{\xi}_1, \vec{\xi}_2)$-plane perpendicular to the ridge line tangent. Within this plane, the gradient of the

scalar field must be maximal in both eigenvector directions (see Eq. 3.2). In practice, however, points satisfying Eq. 3.2 and Eq. 3.3 do not exist. Angles between $\vec{\xi}_3$ and the scalar field gradient are often larger than 25 degrees. For this reason, Peikert and Roth [PS08] proposed to relax Eq. 3.2 and allow that ridge lines can deviate from $\vec{\xi}_3$ at a maximum angle of 40°.

Although ridge detection is a powerful tool to identify maximum lines, the required computations of the first and second derivatives are, however, sensitive to noise in the data. In addition to that, the eigenvectors of the Hessian at degenerated points can be inconsistently oriented, especially at grid points where two eigenvalues of H are equal (as described by Peikert and Sadlo [PS08]). Related to the inconsistency in orientation, it was observed that the identification of ridges lacks robustness and may lead to false-positive or false-negative disconnections of lines (cf. Fig. 2 in [KHS*18]). Concerning the sensitivity to noise, Lindeberg [Lin98] discussed ridge detection applied to different image-scale spaces, which are generated by the convolution of Gaussian smoothing kernels. However, smoothing must be selected carefully to retain features-of-interest on the desired image-scale while obtaining meaningful results. In contrast to ridge detection applied to scalar fields, we propose a slightly different approach and exploit the local wind directions, which is explained in the following.

3.2.2 Maximum Lines from Wind Fields

We propose an alternative approach to detect jet-stream core lines from 3D wind fields by exploiting the local wind directions. In a jet-stream core, we regard the local air parcel momentum at each grid point and assume that maximum lines (height ridges) are defined at loci where the local flow momentum is higher than the flow momentum in the vicinity of the wind maximum. This follows the notion of the jet-stream axis having a zero horizontal and vertical shear [BTH07]. Here, filter criteria are further applied to eliminate minima, saddle points, and light wind conditions. Regarding the zero shear, a core line can be imagined as the line passing through the point of maximal wind magnitude in a local frame-of-reference perpendicular to the 3D wind direction. A schematic of this is given in Fig. 3.3(b). Since the vertical extent of the atmosphere (e.g. 10 km) is much smaller than the horizontal extent (e.g. 1000 km), the vertical component of wind is negligible. Hence, the local frame-of-reference is always vertical (parallel to the vertical z-axis) while the line features are quasi-horizontal. In theory, the momentum p at a grid point is defined as $p = \rho\vec{v}$, where ρ represent the air density and $\vec{V} \in \mathbb{R}^3$ is the velocity. Since we assume that the density ρ is locally constant, the momentum of flow is maximal if the wind speed $|\vec{V}|$ resolved into the local wind direction is maximal within this normal plane.

In [KHS*18], we propose to build a local frame-of-reference to obtain the candidate points for the final jet-stream core lines. In contrast to ridge detection, where the local frame is constructed as (ξ_1, ξ_2)-plane normal to the ridge line tangent, we construct a normal plane which is orthogonal to the local wind direction, and due to the weak vertical wind component, parallel to the z-axis. In contrast

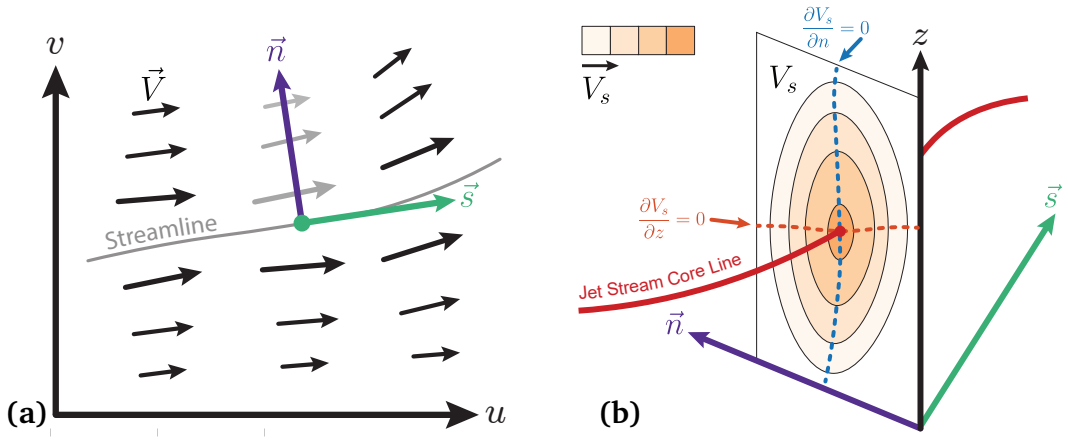


Figure 3.3: ***(a)*** *The depiction of a wind vector field and the local coordinate system that is used to construct the normal plane (similar to Fig. 3 in [KHS*18]). Vector $\vec{s}$ represents the direction of flow and is tangential to the streamline (green arrow), $\vec{n}$ is the vector normal to $\vec{s}$ (purple arrow).* ***(b)*** *A schematic of the spanned normal plane intersecting with the jet-stream core line (red) in 3D (image was adapted from Fig. 3 in [KHS*18]). The bright orange colors represent the velocities resolved into the direction $\vec{s}$ within the plane. The dotted lines denote the zero lines of the derivatives along the z-axis (orange) and along the $\vec{n}$-axis (blue). The jet-stream core line is defined where both zero-derivative lines intersect.*

to ridge detection, which is applied to scalar fields, our algorithm directly uses the wind vector field to obtain jet-stream core lines. Our algorithm is similar to the objective identification of objective fronts [Hew98] and the detection of 2D jet cores [BTH07]. Assume that the NWP data is located in a Cartesian longitude-latitude-pressure grid with terrain-following vertical levels, where a point $\vec{p} = (x, y, z)$ is the position in this coordinate system; $\vec{x}$ and $\vec{y}$ represent the axes along longitude and latitude, respectively. The vertical axis $\vec{z}$ is defined by the hybrid-sigma pressure coordinates. The 3D wind vector output of the NWP model is defined as $\vec{V} = (u, v, \omega)$. The horizontal wind components u and v represent the winds in meridional (eastward in $\vec{x}$) and zonal (northward in $\vec{y}$) direction, respectively. The units are in meters per second (ms^{-1}). The vertical component ω is defined in Pascal per second (Pa s^{-1}) and follows the z-axis. When converted to ms^{-1}, the velocity of ω, however, is significantly smaller than the horizontal wind components with velocities in the order of 10^3. Consequently, ω is not regarded in our detection algorithm (with $\omega = 0$).

In our work, we focus explicitly on the horizontal wind components ($\vec{V} \in \mathbb{R}^2$) at each grid point of our 3D wind field. Let the velocity be $\vec{V} = (u, v)$, where $\vec{n}$ is the vector normal to $\vec{V}$. Since we regard the momentum into the local wind direction at a grid point $\vec{p}$, let $\vec{s} \in \mathbb{R}^2$ be a unit vector parallel to local wind direction (cf. Fig. 3.3(a). At each grid point, $\vec{s}$ is equivalent to $\vec{V}$, which means that $V_s = |\vec{V}|$ is the local wind magnitude. For all other surrounding grid points in the horizontal and vertical vicinity of a grid point, the wind magnitudes are resolved into the direction of $\vec{s}$, which means

that $V_s = \vec{V} \cdot \vec{s}$. Therefore, our local frame-of-reference is finally defined as $(\vec{n}, \vec{z})$-plane spanned by the vectors $\vec{n}$ and $\vec{z}$. A depiction of this frame is provided in Fig. 3.3(b). In contrast to ridge detection, our plane is always consistently defined by the wind directions and does not suffer from the inconsistent orientation of vectors spanning the frame-of-reference, as already mentioned with the orientation of eigenvectors in Sect. 3.2.1. All points considered to be potential candidates for jet-stream core lines are defined by the following equation:

$$\nabla_n V_s \; = \; \frac{\partial}{\partial z} V_s = 0. \tag{3.4}$$

The computation of the directional derivative $\nabla_n V_s$ and $\frac{\partial}{\partial z} V_s$ for a non-uniform vertical coordinate system is described in Sect. 3.2.3. To finally obtain all jet-stream core lines, we compute the Hessian matrix and its eigenvalues to identify the loci of all maxima points. Let $\nabla^N = (\frac{\partial}{\partial n}, \frac{\partial}{\partial z})$ be the vector of the partial derivatives of the resolved wind speed V_s within the ($\vec{n}$-$\vec{z}$)-plane. The Hessian matrix $H_N \in \mathbb{R}^{2\times 2}$ in the ($\vec{n}$-$\vec{z}$)-plane is defined as

$$H_N = \nabla^N (\nabla^N)^T V_s = \begin{bmatrix} \frac{\partial^2 V_s}{\partial n^2} & \frac{\partial^2 V_s}{\partial n \partial z} \\ \frac{\partial^2 V_s}{\partial z \partial n} & \frac{\partial^2 V_s}{\partial z^2} \end{bmatrix} = \begin{bmatrix} \nabla_n^2 V_s & \nabla_n \frac{\partial}{\partial z} V_s \\ \frac{\partial}{\partial z} \nabla_n V_s & \frac{\partial^2}{\partial z^2} V_s \end{bmatrix} \tag{3.5}$$

Note that the mixed partial derivatives $\nabla_n \frac{\partial}{\partial z} V_s$ and $\frac{\partial}{\partial z} \nabla_n V_s$ are equal, which implies that H_N is symmetric and the eigenvalues are real. A grid point is thus considered to be a wind maximum if both eigenvalues λ_1 and λ_2 are negative. For a 2×2 matrix, we can use the second partial derivative test, according to [TF98], by computing the determinant of H_N:

$$D = det(H_N) = (\nabla_n^2 V_s)\,(\frac{\partial^2}{\partial z^2} V_s) - (\nabla_n \frac{\partial}{\partial z} V_s)^2. \tag{3.6}$$

All points $\vec{p}$ are considered to be a local maximum if $D > 0$ and $\nabla_n^2 V_s < 0$. After the identification of all jet-stream core line candidates, we filter the lines utilizing further wind field characteristics. Based on discussions with meteorologists, we identify core lines with wind speeds of at least $40\,\mathrm{ms}^{-1}$ as they consider such lines to be related to significant weather events. Furthermore, maximum lines are not required to be tangential to the streamlines of a wind field, which was also discovered in works by Berry et al. [BTH07] and is closely related to the problem of ridge detection in [PS08]. Hence, we also filter lines where the angle between the line tangent and the local wind vector is at most 40°. For more details, we refer to our published paper [KHS*18]. In the next section, we will briefly explain how to compute the partial derivatives in NWP models.

3.2.3 Computation of Directional Partial Derivatives

According to eq. 3.1, the directional derivative along the normal wind direction $\vec{n}$ is

$$\nabla_n V_s = (\vec{n} \cdot \nabla) V_s = n_x \frac{\partial V_s}{\partial x} + n_y \frac{\partial V_s}{\partial y}. \tag{3.7}$$

On a discrete grid, central differences are used to compute the approximate derivatives along the x- and y-axis. Given that the function V_s is only sampled at the x-coordinate, while the y- and z-coordinate are remained constant, the partial derivative of V_s into direction $\vec{x}$ is defined as follows:

$$\frac{\partial V_s(x)}{\partial x} = \frac{V_s(x+h_x) - V_s(x-h_x)}{2h_x}, \tag{3.8}$$

where h_x is the distance between two grid points in longitude direction. The central differences for the y-axis with grid point distance h_y are computed equivalently. In the NWP data, V_s is defined in ms^{-1}, while the horizontal grid points are given in ° along the sphere. To obtain consistent derivatives, we convert ° to kilometers (km) and wind speeds to $\mathrm{km\,s}^{-1}$. Assuming a global perfect sphere, the distance between two latitude points is approximately $h_y = 1.112 \cdot 10^2$ km per ° on the entire globe. However, the distance between two longitude points h_x is equivalent to h_y but constantly decreases towards the poles, with $h_x = 0$ at the pole. A simple approximation is to scale h_x with the cosine of the current latitude: $h_x = h_y \cdot cos(y)$.

Vertical derivatives constitute a different scenario. As we explained in Sect. 3.1, vertical layers are unevenly spaced in the $\vec{z}$-direction, which implies that the distances $h_z \in \mathbb{R}$ between two grid points in the vertical are non-uniform. For a given layer k, it holds that $h_z^{k+1} \neq h_z^{k-1}$, where h_z^{k+1} and h_z^{k-1} are the distances to the next (upper) layer $k+1$ and previous (lower) layer $k-1$. Following the paper of Sundqvist and Veronis [SV70], we can obtain the finite differences on non-uniform intervals using the Taylor expansion. Let V_s^k be the resolved wind velocity at layer k. Then the equations T_1 and T_2 are defined as follows:

$$T_1 \equiv V_s^{k+1} = V_s^k + h_z^{k+1} \cdot \frac{\partial V_s^k}{\partial z} + \frac{(h_z^{k+1})^2}{2} \cdot \frac{\partial^2 V_s^k}{\partial^2 z} + \dots . \tag{3.9}$$

$$T_2 \equiv V_s^{k-1} = V_s^k - h_z^{k-1} \cdot \frac{\partial V_s^k}{\partial z} + \frac{(h_z^{k-1})^2}{2} \cdot \frac{\partial^2 V_s^k}{\partial^2 z} - \dots . \tag{3.10}$$

The central differences to approximate the 1st order derivative is defined as:

$$T_1 - T_2 \equiv V_s^{k+1} - V_s^{k-1} = (h_z^{k+1} + h_z^{k-1}) \frac{\partial V_s^k}{\partial z} + \left(\frac{(h_z^{k+1})^2 - (h_z^{k-1})^2}{2} \right) \frac{\partial^2 V_s^k}{\partial^2 z} + \dots \tag{3.11}$$

$$\implies \frac{\partial V_s^k}{\partial z} = \frac{V_s^{k+1} - V_s^{k-1}}{h_z^{k+1} + h_z^{k-1}} + \epsilon. \tag{3.12}$$

$$\epsilon = \left(\frac{(h_z^{k+1} + h_z^{k-1})(h_z^{k+1} - h_z^{k-1})}{2(h_z^{k+1} + h_z^{k-1})} \right) \frac{\partial^2 V_s^k}{\partial^2 z} + ... = \left(\frac{(h_z^{k+1} - h_z^{k-1})}{2} \right) \frac{\partial^2 V_s^k}{\partial^2 z} + ... \tag{3.13}$$

The numerical truncation error ϵ is $\mathcal{O}(h_z^{k+1} - h_z^{k-1})$, which means the error reduces with smaller distances between grid points. To achieve consistent derivatives in the vertical, coinciding with the velocities in $\mathrm{km\,s^{-1}}$ and with $h_z^k \leq 1$, we converted the pressure coordinates to height levels in km. Note that for $h_z^{k+1} = h_z^{k-1} = h_z$, the term for $\frac{\partial^2 V_s}{\partial z^2}$ vanishes and the truncation error becomes $\mathcal{O}(h_z^2)$, which is equivalent to the error in classical central differences on regular grids.

The numerical truncation error for non-uniform grids is further reduced by eliminating $\frac{\partial^2 V_s}{\partial z^2}$. This can be achieved by substituting Eq. 3.9 and Eq. 3.10 with $(h_z^{k-1})^2$ and $(h_z^{k+1})^2$, respectively, and subtracting the equations from each other:

$$\begin{aligned} (h_z^{k-1})^2 \cdot T_1 - (h_z^{k+1})^2 \cdot T_2 &\equiv (h_z^{k-1})^2 V_s^{k+1} - (h_z^{k+1})^2 V_s^{k-1} = \\ = [(h_z^{k-1})^2 - (h_z^{k+1})^2] V_s^k &+ \frac{[(h_z^{k+1})^2 (h_z^{k-1})^2] (h_z^{k+1} + h_z^{k-1})}{6} \frac{\partial^3 V_s^k}{\partial z^3} + ... \end{aligned} \tag{3.14}$$

$$\implies \frac{\partial V_s^k}{\partial z} = \frac{(h_z^{k-1})^2 V_s^{k+1} - (h_z^{k+1})^2 V_s^{k-1} - [(h_z^{k-1})^2 - (h_z^{k+1})^2] V_s^k}{h_z^{k+1} h_z^{k-1} (h_z^{k+1} + h_z^{k-1})} + \epsilon, \tag{3.15}$$

$$\epsilon = \frac{[(h_z^{k+1})^2 (h_z^{k-1})^2 (h_z^{k+1} + h_z^{k-1})]}{6 h_z^{k+1} h_z^{k-1} (h_z^{k+1} + h_z^{k-1})} \frac{\partial^3 V_s^k}{\partial z^3} + ... = \frac{h_z^{k+1} h_z^{k-1}}{6} \frac{\partial^3 V_s^k}{\partial z^3} + ... \tag{3.16}$$

Based on Eq. 3.16, the truncation error ϵ is thus $\mathcal{O}(h_z^{k+1} h_z^{k-1})$. For $h_z^k < 1\,\mathrm{km}$, this achieves an overall better approximation for the vertical gradients, in particular for layers towards the lower atmosphere. Hence, eq. 3.12 should be preferred for layers with $\Delta_z > 1$ km or, when computed in different units, like meters.

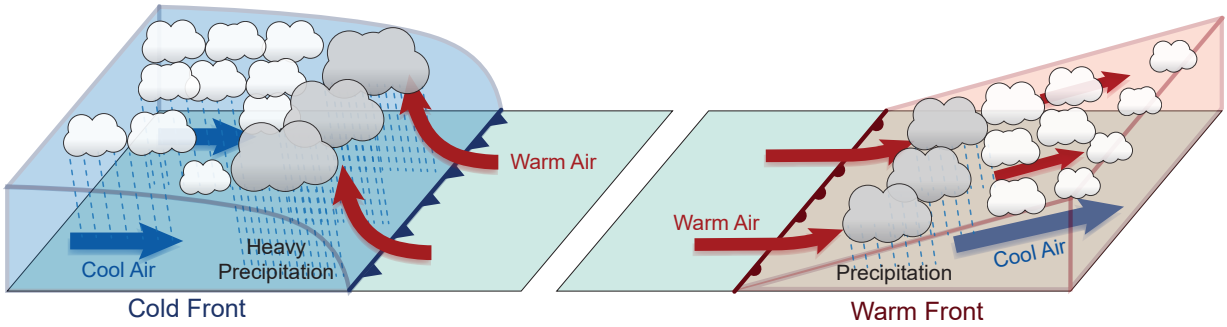


Figure 3.4: *Schematic of the structure of atmospheric fronts in 3D. Thick arrows indicate the direction of flow for cool air (blue) and warm air (red). The left image shows a sketch of a cold front where cool air suppresses warm air. The moist warm air quickly rises and induces cloud formation due to condensation. This causes heavy precipitation near the frontal surface. The right image depicts a warm front. The warm moist air slowly ascends and suppresses cooler air. Here, moderate cloud formation and precipitation is caused by the warm front.*

3.3 Detection of Atmospheric Weather Fronts

Atmospheric fronts are the boundary layers that separate two air masses of different characteristics. Fronts are notorious for causing extreme weather events, such as heavy rain, near the border of cold and warm air, and are usually associated with abrupt changes in local weather conditions. For instance, compare the scenario of a cold front in Fig. 3.4 where cool air quickly moves towards warm air. Here, the dry cold air mass suppresses the warm moist air, as cold air has a much higher density than warm air. Consequently, the warm air rapidly ascends and is cooled down on its way to higher altitudes, which promotes the formation of convection cells near the boundary of the two air masses. The cooling of moist warm air leads to the condensation of water, which leads to cloud formation and supports the genesis of significant weather events, such as thunderstorms with lightning and hail. A sketch of the two major front types, which either advect cold air towards warm air (cold front) or vice versa (warm front) is depicted in Fig. 3.4.

As discussed in Sect. 2.2.1, several definitions and approaches exist to define fronts. The first formalization and objective identification of atmospheric fronts from numerical weather data was proposed by Renard and Clark [RC65]. They used the term *"numerical fronts"*, based on the extraction from numerical weather data, and followed the notion that fronts are the loci at the warm-air boundary along which the thermal gradients are high. Moreover, they explained that fronts are the discontinuities in first-order thermal and moisture derivatives. Regarding numerical weather data, they aim at detecting all loci along which the magnitude of the gradient of the first-order derivative changes most rapidly. In their work, they introduced a "frontal parameter" locator to enable the automated detection of atmospheric fronts on 2D numerical weather data. As this locator is used in several prior publications, we refer to this locator as the "thermal front parameter" (TFP) in the following. The

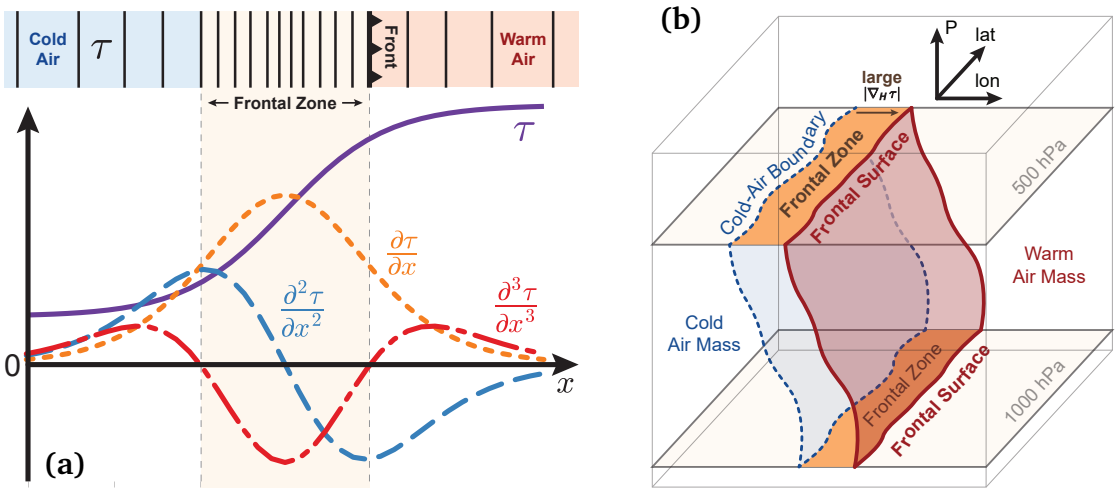


Figure 3.5: ***(a)*** *Illustration of the along-front derivatives of a thermal quantity τ (adapted version of Fig. 3 in [KHS*19]). Note that the frontal surface (front) is located where the third thermal derivative is zero and the second thermal derivative is negative.* ***(b)*** *Depiction of a frontal surface in 3D elongated along the vertical (adapted version of Fig. 2 in [KHS*19]). The frontal zone (region of large thermal gradients) is highlighted in orange. It is located at the horizontal plane that intersects both with the surface at the cold-air boundary (blue) and the frontal surface situated at the warm-air side (red).*

TFP of a thermal or moisture quantity τ, also referred to as thermal parameter, is defined as

$$[L] \quad TFP_\tau = -\nabla_H |\nabla_H \tau| \cdot \vec{g} = -\nabla_H |\nabla_H \tau| \cdot \frac{\nabla_H \tau}{|\nabla_H \tau|} = 0, \tag{3.17}$$

where $\vec{g}$ is a unit vector pointing into the direction of the thermal gradient $\nabla\theta$ and $\nabla_H = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ is the horizontal gradient in grid space. Based on the TFP, the separation of air masses is defined by looking into the direction of the gradient of the thermal parameter and locating the extremal point of its second derivative. In [RC65], those areas were identified where the TFP close to zero and where the magnitude of ∇_H exceeds a certain minimum threshold.

A schematic of a simplified 1D thermal quantity, its gradients, and the location of fronts and the frontal zone is given in Fig. 3.5(a). Here, the function graphs of τ, the absolute first-order gradient $G_\tau = |\nabla_H| \approx \left|\frac{\partial \tau}{\partial x}\right|$, and the curvature $C_\tau = \frac{\partial^2 \tau}{\partial x^2}$ are plotted below. In this example, the quantity τ is linearly increasing along the direction of its gradient represented by the $\vec{x}$-direction. The black isolines denote the quantized values of τ. The area of high gradients is represented by closely-aligned isolines and represents the frontal zone. According to the definition of Renard and Clark, the front (or frontal surface) is then located at the warm-air boundary of the frontal zone, marked as the black thick line on the right of the frontal zone. Comparing the function graphs, this location corresponds to the extremal points in the curvature which describes the most rapid increase or decrease in gradient

magnitude. The warm-air boundary is, hence, marked by the location where the curvature is minimal. To identify such locations, the derivative of the curvature $\frac{\partial^3 \tau}{\partial x^3}$ must be zero and C_τ must be negative. This definition is used in the approach by Hewson et al. [Hew98], which is further explained in Sect. 3.3.2. The first problem, however, in front detection is the choice of an appropriate thermal parameter from which the fronts are identified.

3.3.1 Thermal Quantity

As already indicated in Sec. 2.2.1, many front detection algorithms exist which make use of different thermal quantities. Since we follow the notion of Hewson [Hew98] and the official definition of the WMO, our objective is to automatically identify fronts at points where (sharp) discontinuities appear in both temperature and moisture. A simple and obvious approach would be to use temperature T for the front identification. However, T quickly changes in areas of complex topology, like mountain ranges, as described by Thomas and Schultz [TS19b]. For instance, the air cools down ascending a mountain and heats up descending due to expansion and compression. This would lead to sharp gradients near the mountain ridges and to a false classification of fronts. In contrast to this, potential temperature (θ) — the temperature of an air parcel brought to a reference surface pressure level — is conserved during air-lifting and air-sinking, and does not introduce gradients over complex topography. Furthermore, it better captures moisture effects on air density. Another advantage of θ is that the dynamics and kinematics of fronts are historically defined by means of potential temperature (e.g., compare [SS56, Eli62]). Thomas and Schultz concluded that the gradient of θ can be directly associated with the lift of moist air which is related to cloud formation and precipitation. It is, therefore, an important variable to diagnose fronts (cf. the section "choice of quantity" in [TS19b]).

A drawback of θ is that it is not conserved during moist adiabatic processes, which represent the processes in the atmosphere related to air containing water. For instance, an effect such as condensation (due to air parcel lifting) or evaporation (due to air parcel dropping) occurs, when moist air ascends or descends, and causes θ to increase or decrease, respectively (see [Stu17]). Hence, prior work, such as [Hew98] favored the use of thermal quantities that are conserved during moist adiabatic processes and better capture temperature and moisture (humidity). The main two alternatives here are equivalent potential temperature (θ_e) and wet-bulb potential temperature (θ_w) which are defined based on pressure, temperature, and moisture. These quantities are useful as they tend to produce gradients higher than those of θ in regions with high moisture gradients [TS19b] and may lead to a better front detection.

To shed light on the suitability of different thermal quantities, Hewson [Hew98] compared the results of a front detection using either θ or θ_w, and showed that the detected fronts from both quantities matched manually diagnosed fronts. Based on this, he concluded that both can be used in the analysis and recommended using θ_w as it better traces air masses of different temperature and

moisture characteristics due to conservation during moist processes. Moreover, he argued that θ_w is more frequently used by forecasters than θ in the Met Office to identify surface fronts in model predictions. However, Thomas and Schultz [TS19b] recently argued that θ_e and θ_w are not always conserved during moist processes and claim that both quantities do not necessarily constitute the best quantity to trace different air masses. According to them, moist processes are often accompanied by stronger vertical velocities, whereas fronts are mainly identified on a quasi-horizontal plane and neglect this relationship. Therefore, it is not clear whether incorporating moist processes is beneficial to automated front detection. Based on their extensive study with different quantities and methods, they concluded that θ or θ_w / θ_e have different advantages or disadvantages and should be used depending on the type of areas. For instance, they recommended the usage of θ_w in areas of high moisture (tropics, subtropics), but θ should be used to identify fronts at lower atmospheric levels.

Since we focus on areas over the North Atlantic between Europe and the United States of America in our analyses and adapt the methods by Hewson [Hew98], we decided to use θ_w to identify fronts in our work. Note that wind is not regarded in our detection methods, as wind-based detection methods, such as [SKTB11], cannot be used to identify fronts near cloud or precipitation bands, as these bands are rather associated with high thermal gradients. Therefore, Hewson et al. argued that wind is not suitable to automatically identify atmospheric fronts [Hew98, TS19b].

3.3.2 Objective 2D Front Detection

Regarding Eq. 3.18, ridge surface detection applied to the G_τ field can be used to obtain the location of fronts. However, this would require the computation of its Hessian matrix, which is comprised of third- and fourth-order derivatives of τ. Unfortunately, its computational accuracy suffers from the approximation error induced by finite differences. Hence, we refrain from using ridge surface detection and mainly follow the notion of the objective front detection, proposed by Hewson and Titley [Hew98, HT10]. They defined fronts at the warm-air boundary of the thermal gradients and identify locations where the third-order thermal derivative is zero. Their front locator $[L]$ is defined as the first-order derivative of the horizontal thermal curvature $C_\tau = \nabla_H |\nabla_H \tau|$ in the direction of the thermal curvature:

$$[L] \quad \frac{\partial (C_\tau)_s}{\partial s} = 0, \quad with\, \hat{s} = \pm \frac{C_\tau}{|C_\tau|}, \tag{3.18}$$

where $\hat{s}$ is the 5-five point mean unit axis of the thermal curvature for a local grid point. The 5-point mean axis is the average direction of C_τ and computed by taking into account the C_τ directions at the adjacent two grid points in both $\vec{x}$- and $\vec{y}$-direction. Hewson proposed this mean axis because the computation of second derivatives is prone to noise in the data. This makes the detection scheme based on $[L]$ highly sensitive to variation in the data and may produce very fine-scale cluttered small fronts. Hence, the mean axis $\vec{s}$ can be used to compute the locations of fronts in a more simple

approach. First, all gradients $\nabla_H \tau$ are resolved into the direction of $\vec{s}$: $Gs_\tau = (\nabla_H \tau) \cdot \vec{s}$. Next, the divergence along the direction $\vec{s}$ is computed with:

$$D_\tau = \left| \frac{\partial Gs_\tau}{\partial x} + \frac{\partial Gs_\tau}{\partial y} \right|. \tag{3.19}$$

The front locations are obtained by setting $D_\tau = 0$. Hewson showed that this "along-vector divergence" in the gradient vector field can be used to obtain the front locations. Furthermore, it is more numerically stable as it avoids the computation of third-order derivatives. An in-detail discussion of vector divergence and shear is provided in the appendix of [Hew98]. As fronts are defined on the warm-air boundary, the fronts are located in regions with negative C_τ. The candidate points for fronts are thus filtered by the TFP (based on eq. 3.17):

$$[M1] \quad TFP_\tau = -C_\tau \cdot \frac{\nabla_H \tau}{|\nabla_H \tau|} > K_1, \tag{3.20}$$

where K_1 is a user-defined threshold in units of Kelvin (K) per 100 km^2 (~ 0.3 in [Hew98]).

Forecasters intend to detect fronts in regions of significant thermal gradients. For this, Hewson proposed to measure the strength of a front by following the direction of the horizontal thermal gradient within the frontal zone at the current pressure level, which is referred to as the baroclinic zone. He estimated the next point traced along thermal gradient by a single-step integration scheme which uses the direction of C_τ and the grid length (the distance between two adjacent points) of the NWP data. Hence, he identified fronts using the following filter equation:

$$\begin{aligned} S_\tau^{\sim ABZ} &= |\nabla_H \tau|^{(\sim ABZ)} = |\nabla_H \tau| + \kappa |C_\tau| \\ [M2] \quad S_\tau^{\sim ABZ} &> K_2, \end{aligned} \tag{3.21}$$

where $S_\tau^{\sim ABZ}$ denotes the frontal strength in the adjacent baroclinic zone (ABZ), κ is a fraction of the grid point distance, and K_2 is a user-defined threshold in units of K per 100 km. Hewson proposed to set $\kappa = \frac{1}{\sqrt{2}}$ to achieve best results, while K_2 was set to values between 0.95 and 1.3.

Sometimes, streams of moist and warm air occur near extratropical cyclones. These streams are characterized by fast ascending air, which causes heavy precipitation when moved over cooler air [Stu17]. As they are associated with high moisture gradients in θ_w, but are not considered as fronts by forecasters, Hewson and Titley [HT10] proposed to filter out those points by measuring the strength of θ in the adjacent baroclinic zone:

$$\begin{aligned} S_\theta^{\sim ABZ} &= |\nabla_H \theta|^{(\sim ABZ)} = |\nabla_H \theta| + \kappa |C_\theta| \\ [M3] \quad S_\theta^{\sim ABZ} &> K_3, \end{aligned} \tag{3.22}$$

where K_3 is a user defined threshold in units of K per 100 km. The locators L and the filter criteria $[M1]$, $[M2]$, and $[M3]$ are used to obtain the final front features.

Fronts are further categorized into **"warm fronts"**, where warm air is advected towards areas of colder air, and **"cold fronts"**, which bring cold air and suppress warm air masses (cf. schematics in Fig. 3.4). According to Hewson [Hew98], this can be defined by tracking the geostrophic thermal advection. Geostrophic in this context denotes the direction of winds under the assumption that the Coriolis force and the pressure gradient force are in balance. The geostrophic thermal advection (A_{G_τ}) is defined as

$$A_{G\tau} = -V \cdot \nabla_H \tau, \tag{3.23}$$

where V is the geostrophic wind velocity. All points with $A_{G\tau} > 0$ are considered to be warm fronts, whereas negative points with $A_{G\tau} < 0$ represent cold fronts.

3.3.3 Extension of Front Detection to 3D

To extend this approach to the third dimension, one can assume that the vertical component of the thermal gradients needs to be taken into account, as well. As already discussed in Sec. 2.2.1, the official definitions of fronts, however, only consider the horizontal thermal gradients. By looking at thermal gradient profiles and vertical cross-sections of thermal quantities, it can be presumed that fronts are also associated with vertical gradients. However, as the vertical extent of the atmosphere is much smaller than the horizontal extent of the Earth's surface, the strong vertical gradients are much greater than strong horizontal gradients, i.e., 10 K/100 m versus 10 K/100 km. Regarding the definition of along-thermal-gradient fronts in eq. 3.18, including a strong vertical component would cause the thermal gradient direction to point more into the direction of the vertical axis, which would yield quasi-horizontal frontal surfaces. This is a contradiction to the official definition, as fronts separate two air masses in the horizontal dimension and would be considered to be the vertical boundary between these air masses. Strong vertical thermal gradients also occur in anticyclonic weather systems — fair weather usually accompanied by a high-pressure system — which inherits large-scale subsidence (descending air) [Stu17, WH06], also called anticyclonic inversion. This is considered to be the "opposite" of frontal processes, and thus not declared to be a front by forecasters. In our work, we make use of the detection and filtering equations proposed by Hewson (explained in Sect. 3.3.2) and improve the filtering to obtain 3D fronts across all vertical levels. A schematic of a 3D front is depicted in Fig. 3.5(b).

Feature Candidates: Due to the aforementioned mismatch between horizontal and vertical thermal gradients, we decided to only take into account the horizontal gradients in our 3D detection method. At each grid point in our 3D volume, we compute the locator quantity $[L]$ (cf. eq 3.18) and retrieve

the zero-isosurfaces using either ray casting or the "Marching Cubes" algorithm [LC87]. The locations at the zero-isosurfaces are regarded as the candidates for the final fronts.

Fuzzy Filtering: To obtain the candidates located at the warm-side boundary, we make use of the TFP (c.f. eq 3.17 and eq. 3.20) and check whether the negated TFP is positive. We also determine the frontal strength to filter out weak front features. Similar to Hewson, we compute $[M2]$ in Eq. 3.21 and $[M3]$ in Eq. 3.22 at each candidate location. Unfortunately, the filter criterion $[M2]$ and $[M3]$ estimate the along-front magnitude of the thermal gradient by regarding just one single integration step. This does not represent the full distribution of thermal gradients well along the front, since the breadth of the zone can be larger than one grid point. Furthermore, the definition of a single filter threshold for K_2 and K_3 to binary filter front candidates across all vertical levels is not possible as thermal gradients greatly vary with height, with the strongest thermal gradients occurring near the surface. Hence, we replace the hard filter thresholds K_2 and K_3 and, instead, offer a flexible framework to "softly" filter the frontal strength criteria. This means that fronts are filtered by manually defining an interval of possible values for S_τ and S_θ. For each parameter S_τ and S_θ, users can define a transfer function that maps a range of frontal strength values to opacity. With this, they can interactively explore how the strength is related to frontal surfaces along the vertical axis and can adjust the transfer function until the most meaningful features can be extracted. For further details of this filtering process, we refer to the paper in [KHS*19]. In the next section, we describe how to define the frontal strength by using the concept of normal curves traced along the thermal gradient.

3.3.4 Computation of Frontal Strength with Normal Curves

We consider the frontal strength to be the absolute change of τ along the shortest path from the warm-air to the cold-air boundary through the frontal zone. As fronts are quasi-orthogonal to the thermal gradient, a simple approach would be to traverse the frontal zone along the surface normal until the cold-air boundary is hit. However, a direct line along the surface normal of a frontal surface does not guarantee the shortest path through the typical type of frontal zones. In particular, Hewson mentioned that it has been observed that most atmospheric fronts are curved (cf. type 3 fronts in [Hew98]). Hence, fronts are characterized by non-uniform changes in thermal gradients and by undulations in the shape of frontal surfaces and their frontal zone. The shape of a typical front is sketched in Fig. 3.6(a). Hence, we improve Eq. 3.21 and Eq. 3.22 and make use of so called "normal curves" to estimate the strength of the frontal zones.

Normal curves are trajectories traced through the underlying scalar field along the gradient of the scalar field, which implies that they are always locally tangential to the direction of the corresponding scalar gradient (cf. blue curves in Fig. 3.6(a)). For example, Pfaffelmoser et al. [PRW11] used normal

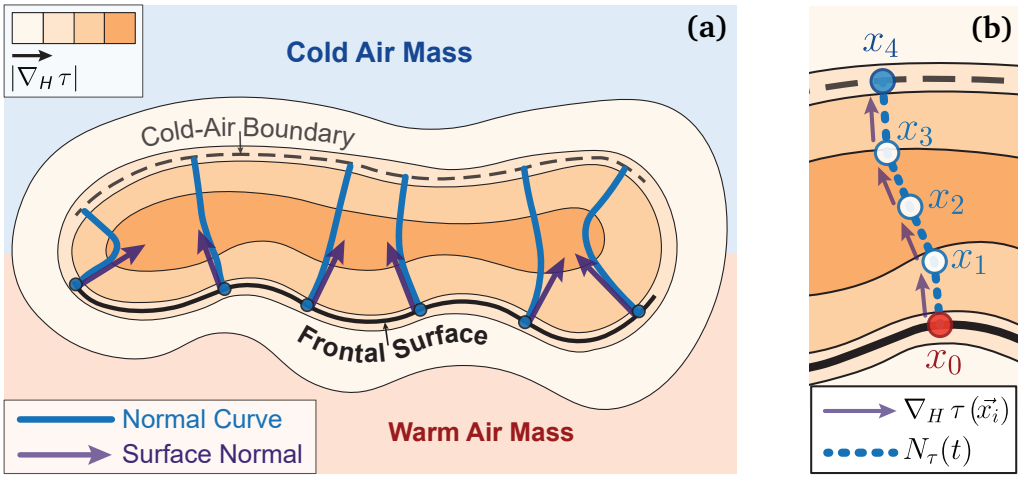


Figure 3.6: ***(a)*** *This schematic shows a horizontal cross-section of a thermal gradient field and its corresponding 2D frontal surface. Blue lines represent the normal curves traced from the frontal surface to the cold-air boundary. Purple arrows indicate the frontal surface normal at the starting point of a normal curve.* ***(b)*** *Close-up view of the frontal zone in (a). It depicts the traversal of a single normal curve through the frontal zone using the Euler integration scheme. The red point indicates the start position of the curve at the frontal surface, while the blue point is the intersection point of the curve with the cold-air boundary.*

curves to measure the spatial distance of two isosurfaces in a stochastic distance field, and Rautenhaus et al. [RKSW15] made use of normal curves to visualize the distribution of quantities within isosurfaces. As the frontal surfaces are retrieved by computing directional derivatives along the thermal gradient, we mainly follow the notion that the orientation of frontal surfaces and its counter-part at the cold-air boundary is always quasi-orthogonal to the thermal gradient. Consequently, we make use of the normal curve in the $|G_\tau|$ field to facilitate the shortest path traversal through frontal zones.

As sketched in Fig. 3.5(b), the frontal zone is associated with the frontal surface at a fixed vertical level and lies in the horizontal plane of high thermal gradients. It is constrained by the cold-air and warm-air boundaries at the corresponding vertical level. Starting from a point on the frontal surface, the normal curve is horizontally traced through the G_τ scalar field until the cold-air boundary is hit. Based on this, let $\tau(\vec{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}$ be the thermal field in the horizontal, and let the normal curve $N_\tau(t) : \mathbb{R} \rightarrow \mathbb{R}^2$ be the curve function within the frontal zone. $N_\tau(t)$ is parameterized by a scalar $t \in \mathbb{R}$, defined for the interval $[0, t_N]$, and follows the direction of the first-order horizontal thermal gradient $\nabla_H \tau$. Note that the z-value (height) of the normal curve is always equivalent to the z-value of the start position on the frontal surface, and is omitted in the following definitions. Further assume that the function $g_\tau(\vec{x}) := \nabla_H \tau(\vec{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ denotes the vector field of all horizontal gradients.

$N_\tau(t)$ is thus defined by the following ordinary differential equation (ODE):

$$\begin{aligned} &\frac{\delta N_\tau}{\delta t}(t) = g_\tau(N_\tau(t)) \quad , \quad N_\tau(0) = \vec{x_0} \\ \iff &N_\tau(t) = \vec{x_0} + \int_0^t g_\tau(N_\tau(s))\,\mathrm{d}s \end{aligned} \tag{3.24}$$

where $\vec{x_0} \in \mathbb{R}^2$ is a start position of the normal curve located on the frontal surface, and t_N is the final value of t at the intersection point between the normal curve and the cold-air boundary. The ODE can be solved by using the Euler method and the Riemann sum:

$$N_\tau(t) \approx \vec{x_0} + \sum_{i=0}^{n-1} g_\tau(N_\tau(t_i))\,\Delta_t, \text{ with } \Delta_t = \frac{t_n}{n},\ 0 \le n \le N\,, \tag{3.25}$$

where t_0 is the start position of the normal curve with $N_\tau(t_0) = \vec{x_0}$. A visual schematic of the normal curve traversal can be observed in Fig. 3.6(b). Next, the length of the normal curve needs to be computed to obtain the frontal strength. This length is equivalent to the arc length of the normal curve. Let L_{N_τ} be the arc length of the normal curve defined on the interval $t \in [0, t_N]$, which is computed as follows:

$$L_{N_\tau} = \int_0^{t_N} \left|\left|\frac{N_\tau(t)}{\delta t}\right|\right| \mathrm{d}t = \int_0^{t_N} \left|\left|g_\tau(N_\tau(\vec{x}))\right|\right| \mathrm{d}t \approx \sum_{i=0}^{N-1} \left|\left|g_\tau(N_\tau(t_i))\right|\right| \Delta_t, \tag{3.26}$$

where $||\,.\,||$ is the Euclidean distance. With $N_\tau(0) = \vec{x_0}$ and $N_\tau(t_N) = \vec{x_N}$, the final equation for the frontal strength S_τ is given by the following equation:

$$[M2] \quad S_\tau = \frac{|\tau(\vec{x_N}) - \tau(\vec{x_0})|}{L_{N_\tau}} \tag{3.27}$$

$[M3]$ is similar to Eq. 3.27 with the thermal parameter $\tau = \theta$ and the user-defined threshold K_3:

$$[M3] \quad S_\theta = \frac{|\theta(N_\theta(t_N)) - \theta(N_\theta(0))|}{L_{N_\theta}} \tag{3.28}$$

Regarding the selection of filter criteria for S_τ and S_θ, for instance via transfer functions, we refer to our paper [KHS*19].

3.3.5 Data Smoothing

As mentioned in Sect. 3.3.2, high-resolution NWP data facilitates the detection of fine-scale features but also inherits fine-scale variations of thermal gradients, which hampers the detection of smooth features. As forecasters desire to detect smooth frontal surfaces at synoptic scale, averaging of thermal quantities prior to the detection is a common method to facilitate a proper detection of smooth fronts. For instance, Jenkner et al. [JSS*09] explicitly use a 2D smoothing kernel to average the thermal quantity at a grid point. In their work, the new quantity τ at a 2D grid point (x, y) is averaged as follows:

$$\tau_{(x,y)} = \frac{1}{2}\tau'_{(x,y)} + \frac{1}{8}(\tau'_{(x+1,y)} + \tau'_{(x-1,y)} + \tau'_{(x,y+1)} + \tau'_{(x,y-1)}), \tag{3.29}$$

where τ' is the unsmoothed thermal quantity. This approach assumes that the distances between grid points in the horizontal are uniform across the entire globe, which is not true as the distances between points in meridional direction decrease towards the poles.

To take into account the non-uniform distances per latitude, we consider all points within a user-defined geometric distance from the current grid point and apply a 2D Gaussian smoothing kernel to compute the averaged values. This smoothing approach does not depend on the resolution of the grid and is related to the length scales of the detected frontal features, which possess a length of 100 km to 1000 km. Hence, the 2D Gaussian kernel uses a standard deviation $\sigma = \sigma_x = \sigma_y = 100\,\text{km}$ for both horizontal axes x and y.

$$\begin{aligned} \tau_{(x,y)} &= \sum_{x_N}\sum_{y_N} G(x_N, y_N)\,\tau'_{(x_N,y_N)} \ , \ \text{with} \\ G(x_N, y_N) &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x_N - x)^2 + (y_N - y)^2}{2\sigma^2}\right). \end{aligned} \tag{3.30}$$

x_N and y_N denote the grid points within the user-defined geometrical distance while $x_N - x$ is the distance between x and x_N in the meridional direction; the distance in zonal direction y is computed equivalently. For further details, we refer to our published paper [KHS*19].

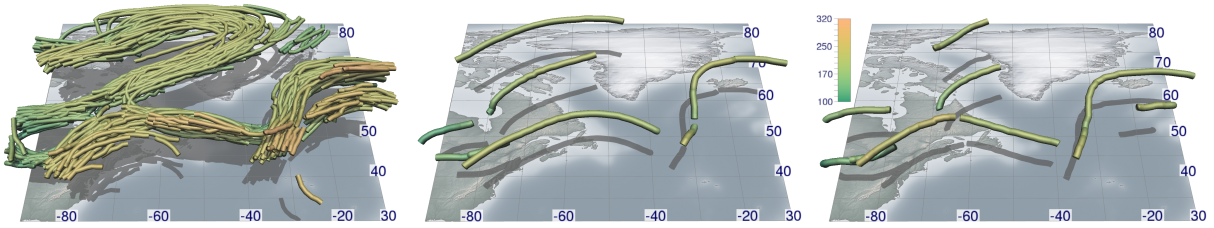


Figure 3.7: *The left image depicts a spaghetti plot of jet-stream core lines detected in wind ensemble data for September 28, 2016, from initial simulation time on September 25, 2016. The core lines are rendered as 3D tubes and colored by their altitude in pressure (hPa) (cf. inset on the right). The other two images display jet-stream core lines extracted from the wind field in ensemble member 15 (middle) and member 44 (right).*

3.4 Clustering Ensembles of Line Features

Identifying subsets of 3D features with similar characteristics, where each subset of highest within-group similarity is referred to as a cluster, is a challenging task for weather ensemble data. Regarding jet-stream core lines, for instance, medium-range forecasts for at least the next three days already produce highly different wind fields for each ensemble member. Hence, the line features extracted from these fields can vastly differ wrt. certain line characteristics from member to member. Among those characteristics are the number of vertices representing the line, their location in space and time, their orientation, and their topology. A depiction of a real ensemble of jet-stream core lines is depicted in Fig. 3.7. Simple metrics, such as Euclidean distance, are not appropriate to compare vastly different line sets. Therefore, the definition of similarity between ensembles of complex line sets needs to be defined to facilitate clustering. It is also possible to cluster directly on the scalar fields, which are used to identify line features. Alternatively, clustering can be applied to an implicit line representation, which is independent of the orientation and location of line sets. Nevertheless, all these variants of clustering strategies involve the identification of similar subsets in a very high-dimensional data set. In this case, clusters cannot be easily determined as many data elements — the data entries (vertices, proxy fields) characterizing a line — per ensemble member can be correlated, which means that no clear distinction between ensemble members can be made. In the following, we discuss the basics of clustering, similarity metrics of line sets, dimensionality reduction techniques, and alternative representations for line sets.

3.4.1 Clustering Approaches

Clustering is an effective tool in data mining to identify clusters of similar characteristics in multi-dimensional data. Based on a distance or similarity metric, data observations are assigned to the cluster closest wrt. their features. Observations here describe the group of features contained in one single ensemble member. The assignments can be conducted in binary form, where each observation is assigned to only one cluster, or in a soft manner, where observations belong to all clusters with a certain probability. In this thesis, we briefly discuss two well-known variants of binary clustering that were also used in one of our papers [KW19a]. For a more detailed explanation of clustering, we refer to the book "Cluster Analysis" by Everitt [ELL09].

K-Means

K-Means is a prominent clustering algorithm and, due to its simplicity and low computational costs, it is often used in data analysis to cluster multi-dimensional data. The term "k-Means" was first introduced by MacQueen [Mac67], while the idea dates back to Steinhaus [Ste56]. The most used variant of k-Means is the one from Lloyd [Llo82] which was formalized in 1957 and published decades later, while the equivalent approach was published earlier by Forgy [For65]. k-Means aims to minimize the Euclidean distance between two observations, formalized by the following cost function:

$$\arg\min_{C} \sum_{i=1}^{K} \sum_{x \in C_i} ||x - \mu_i||, \tag{3.31}$$

where $||\cdot||$ denotes the Euclidean distance or L_2-Norm between two vectors, μ_i is the vector representing the center (mean) of the i-th cluster C_i, and K is the user-defined number of clusters to be identified. Regarding Lloyd's algorithm, an initial set of K cluster centers is selected, where the cluster center μ_i represents a randomly chosen observation. The remaining observations are assigned to the closest initial cluster centers. After the initialization, the cluster centers are recomputed by averaging the observations belonging to a cluster, and all observations are re-assigned to the cluster with the cluster center closest to them. These steps are repeated until the optimization converges to a minimum.

Finding an optimal solution with k-Means, however, is an NP-hard problem, as it usually runs into a local but not global minimum. Hence, k-Means suffers from slow convergence and has to be conducted multiple times to achieve the results considered most optimal. To increase the convergence speed, different variants have been proposed to chose an improved initial configuration of cluster centers. For instance, "k-Means++" [AV07] uses a probabilistic approach to enforce keeping initial cluster centers at a maximum distance to each other. For more detail on alternative approaches, such as k-Medians, and additional initialization of k-Means, we refer to [ELL09].

Agglomerative Hierarchical Clustering

Instead of assigning observations to a fixed number of clusters, a hierarchy of a cluster series can be produced from the data set. The hierarchy runs from a single cluster comprised of all N observations to N clusters containing only one single observation [ELL09]. In this context, the pair-wise similarities (proximities, distances) between observations are stored in a symmetric matrix S with entries s_{ij} denoting the similarity between data element i and j. In agglomerative methods, the hierarchy of clusters is built in a bottom-up fashion, where most similar individual clusters, starting from the individual observations at the beginning, are sequentially merged to a new cluster until one global cluster is attained in the last step. This results in a binary tree of clusters, where each node represents the similarity of its underlying two clusters (children). Note that the root node combines the last two clusters with maximal dissimilarity between those two. The final number of clusters can be obtained by cutting the tree at a certain similarity or tree level.

Various merge heuristics, also called linkage methods, exist to define which two clusters need to be fused at each iteration step. Among those is single linkage clustering, also called nearest neighbor clustering, where clusters with the smallest pair-wise distances are combined. The similarity matrix S is updated by removing the distances for the merged clusters and inserting the new cluster where the new similarity at the parent node represents the minimum distance between the merged clusters. The counterpart of single linkage is complete linkage, where the farthest cluster-to-cluster distances are computed and used for fusion and similarity updates. In our work, we make use of Ward's [Jr.63] method for cluster merges to minimize the increase of the within-cluster variance with each onward fusion step. To achieve this, the pair-wise squared Euclidean distances between two observations are computed and stored in S. Afterwards, the measure for the distances between a cluster C_k and the fusion $(l \sqcup m)$ of two other clusters C_l and C_m is computed by the Lance-Williams formula:

$$s_{k(l \sqcup m)} = \alpha_l s_{lk} + \alpha_m s_{mk} + \beta s_{lm} + \gamma |s_{lk} - s_{mk}|, \quad \text{with} \tag{3.32}$$

$$\alpha_l = \frac{n_l + n_k}{n_l + n_m + n_k} \quad , \quad \alpha_m = \frac{n_m + n_k}{n_l + n_m + n_k} \quad , \quad \beta = \frac{-n_k}{n_l + n_m + n_k} \quad , \quad \gamma = 0, \tag{3.33}$$

where n_i is the number of elements contained in cluster C_i. Here, the cluster C_l and C_m with minimal distance $s_{k(l \sqcup m)}$ to cluster C_k is preferred during agglomeration.

3.4.2 Comparison of Line Topology

For an ensemble of shapes, the next step is to define the (dis-)similarity of line sets between two ensemble members. Corouge et al. [CGG04] introduced different types of similarity metrics for line sets that are spatially close to each other, such as with fibers in muscle diffusion tensor Magnetic

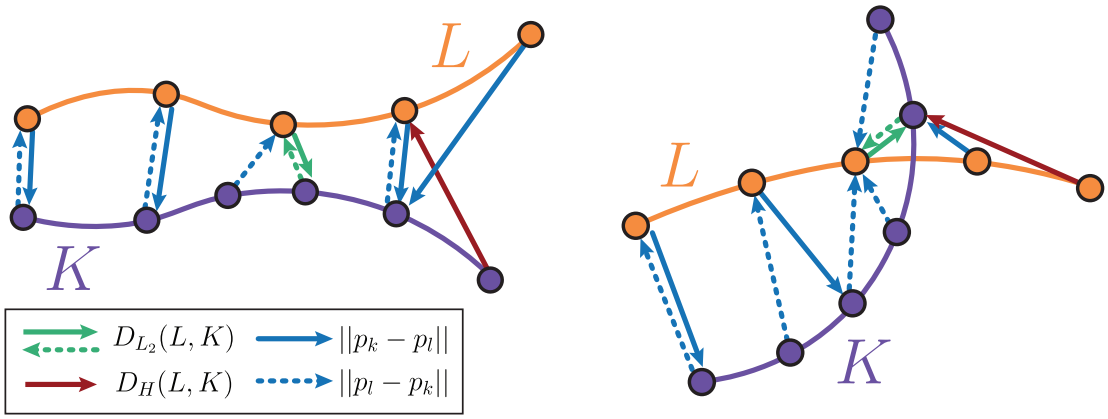


Figure 3.8: *Sketch of two different sets of lines (L and K), their line vertices (circles with black contours), and their minimal vertex-wise distances (arrows). Left image: L and K are quasi-parallel to each other. Right image: L and K intersect and exhibit a completely different orientation. The vertex-wise distances define the similarity between to line sets. The green arrow depicts the minimal vertex-wise Euclidean distance, while the red arrow represents the corresponding Hausdorff distance.*

Resonance Imaging (MRI). A simple approach to measure the similarity between those fibers (or lines), here denoted as L and K for two arbitrary lines, is to compute the minimal distance between the pair-wise distances of the 3D line points p_l and p_k

$$D_{L_2}(L,K) = \min_{p_l \in L, p_k \in K} d(p_l, p_k), \tag{3.34}$$

where the distance between two points is defined as the Euclidean distance

$$d(p_l, p_k) = ||p_l - p_k||. \tag{3.35}$$

While this is easy to compute, the downside of this metric is that the distance between non-spatially aligned line sets is non-uniform in terms of point-wise distances, and the minimum distance is thus not representative. For example, the scenario in Fig. 3.8(b) depicts the pair-wise minimal distances between the vertices of two cutting lines K and L. Both lines are spatially distant to each other in most parts, however, D_{L_2} computes the minimal distance at the line overlap and falsely suggests that K and L are as similar to each other as in Fig. 3.8(a). In contrast to D_{L_2}, Corouge et al. proposed to compute the average of the minimal pair-wise closest distances between two lines sets, which is called the mean closest-point distance (MCPD):

$$\begin{aligned} D_M(L,K) &= \text{mean}(d_{min}(L,K), d_{min}(K,L)), \\ d_{min}(L,K) &= \text{mean}_{p_l \in L} \min_{p_k \in K} d(p_l, p_k) \end{aligned} \tag{3.36}$$

While MCPD better estimates the point-wise similarity of arbitrary line sets, its computation is most expensive for a large set of lines. A more efficient alternative to MCPD is the Hausdorff Distance (HD), named after the mathematician Felix Hausdorff, which was first introduced in the book "Grundzüge der Mengenlehre" [Hau14]. With HD, two different sets of points are considered similar if the longest of all point-wise minimal distances is small. HD $D_H(L,K)$ is thus defined as:

$$\begin{aligned} D_H(L,K) &= \max\{d_m(L,K), d_m(K,L)\} \\ d_m(L,K) &= \max_{p_l \in L} \min_{p_k \in K} d(p_l, p_k) \end{aligned} \tag{3.37}$$

A sketch of D_H between two line sets is shown in Fig. 3.8 (compare red arrows).

Ensemble comparison

MCPD and HD can be used to determine the similarity of the lines of one ensemble member to the lines of any other ensemble member. However, a metric has to be defined to measure the (dis-)similarity of two ensemble members based on the used distance metrics. In the following, let Ω be a set of ensemble members with ensemble members $n, m \in \Omega$, and let $\hat{L_n}$ and $\hat{L_m}$ be the set of lines of the corresponding ensemble members. Furthermore, $D_X(L,K)$ denotes an arbitrary metric representing the distance between two single lines L and K, where $X \in \{H, M\}$ for HD and MCPD, respectively. Two ensemble members are compared in terms of line similarity by computing the average of all minimal line-pair-wise distances D_X. The member-wise similarity $D_\omega(\hat{L_m}, \hat{L_n})$ between ensemble member m and n is defined as follows:

$$\begin{aligned} d_e(L, \hat{L_i}) &= \min\left\{D_X(L\,, K)\,, \text{ for } K \in \hat{L_i}\right\} \\ D_{mn} := D_\omega(\hat{L_m}, \hat{L_n}) &= \text{mean}\left\{ \max_{L'_m \in \hat{L_m}} \left\{d_e(L'_m, \hat{L_n})\right\}, \max_{L'_n \in \hat{L_n}} \left\{d_e(L'_n, \hat{L_m})\right\} \right\}. \end{aligned} \tag{3.38}$$

$D_\omega = d_{mn} = d_{nm}$ denotes the distance of lines in ensemble member m to the lines in ensemble member n and can be used in clustering to set up a symmetric distance matrix comprised of all member-wise dissimilarities to facilitate hierarchical clustering of ensembles of shapes. For further details, we refer to the corresponding published paper in [KW19a].

3.4.3 Implicit Line Representation

The downside of the direct line-to-line comparison is that lines can highly vary in topology, location, and orientation. Metrics such as HD and MCPD fail to compare lines in scenarios where lines partly coincide and mainly diverge from each other. Locations where the line sets of all ensemble members mostly coincide, independent of the orientation and location of a line, cannot be reflected by simply

considering similarities based on the point-wise distances of lines. Hence, a more implicit representation of lines is required to better indicate at which grid points the ensemble members concerning their line sets are most similar. In the following, assume that the domain of our ensemble members is discretized into a regular voxel grid in the horizontal and vertical. The resolution of this grid is chosen according to the smallest distance between two adjacent horizontal and vertical grid points.

Density Volumes

One approach is to compute a visitation map, proposed by Bürger et al. [BFMW12], where at each grid point the number of lines crossing the voxel boundaries for all ensemble members is counted. The number of line crossings represents the amount of local density at a voxel. Since the lines are defined by discrete points (vertices) in the data domain, the connection line between the endpoints of a line segment may pass several voxels. Hence, the lines need to be rasterized onto the predefined voxel grid. This can be accomplished, for instance, by computing the intersection points of the line segments with the boundaries of the voxels, which is computationally expensive. To overcome costly intersection tests for each line, a more efficient GPU implementation can be used. Bürger et al. [BFMW12] assumed that each line vertex is a spherical particle with a fixed diameter and baked this particle into the voxel grid using the rasterization on the GPU. The fixed diameter is represented by a splat-kernel, in our work with a size of 4 voxels, to effectively rasterize lines. Furthermore, rendering with splat-kernel also facilitates the generation of smooth density distributions. The final densities are normalized in a final second pass after all lines have been rasterized to produce a normalized density volume. With this, regions of high density can be visualized, such as with ray casting, to indicate locations of high similarity across the entire ensemble. A sketch of an ensemble of lines and the corresponding densities at each grid point is given in Fig. 3.9(a).

Vector-To-Closest Point Volumes

Another alternative is the generation of vector-to-closest point (VCP) volumes [AMT*12]. Here, at each grid point, the direction and distance to the line feature closest to the voxel center are computed for all ensemble members. Let $\text{vcp}(\vec{x})_i$ be the vector to the closest line feature in ensemble member i at grid point $\vec{x}$. The VCP volume is comprised of a set of VCPs for all N ensemble members, with $\text{VCP}(\vec{x}) = \left\{vcp_1(\vec{x}), vcp_2(\vec{x}),\ \ldots, vcp_N(\vec{x})\right\}$. A schematic of VCPs for an ensemble of lines is given in Fig. 3.9(b). The advantage of VCPs is that they enable users to reconstruct the locations of either individual or all line sets by computing the minimal distance to the closest feature in all ensemble members:

$$d_{min}(\vec{x}) = \min\left\{||vcp_i(\vec{x})||,\ vcp_i(\vec{x}) \in VCP(\vec{x})\right\}. \tag{3.39}$$

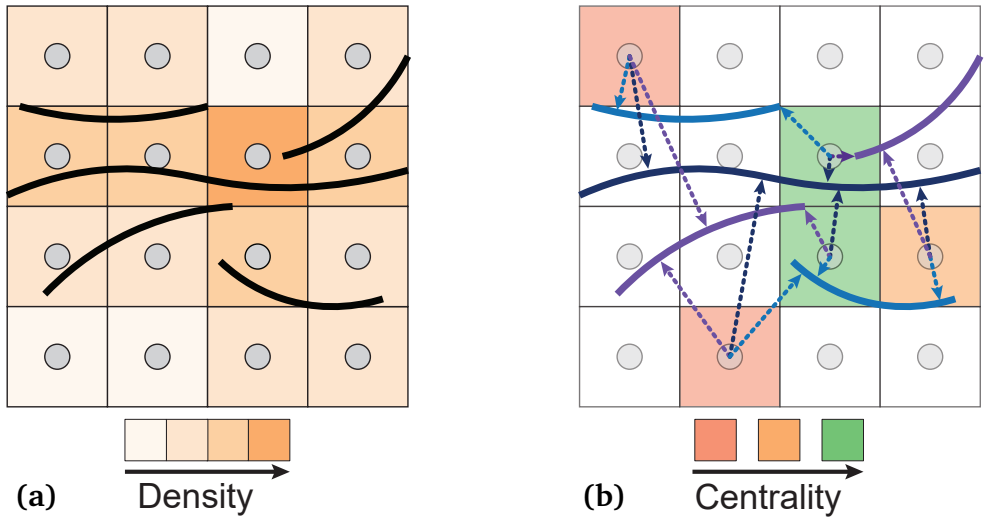


Figure 3.9: ***(a)*** *Sketch of a density volume (visitation map) from an ensemble of lines (similar to Fig. 6 in [KW19a]). The bright orange colors represent the numbers of line crossings at each voxel, which reflects the density.* ***(b)*** *Depiction of the vectors that point to the line features closest to a voxel center (dotted arrows) in an ensemble of 3 line sets (blue, dark blue, and purple lines). The central tendency is related to the distribution of these vectors, where color denotes the level of centrality (red = lowest, orange = medium, green = highest).*

Furthermore, VCPs can be clustered by directly comparing the vector differences of all member-wise $vcp_i(\vec{x})$. That is, the more similar the vectors and their distances to the closest line feature are at a voxel, the more similar the ensemble members can be considered. A problem with clustering on VCP volumes is that $d_{min}(\vec{x})$ is high for voxels that never cross any lines in all ensemble members, or whose distances to the nearest feature are high for the entire feature ensemble. To overcome this, one can reduce the number of grid points considered during clustering by removing grid points where the minimal distance of the vectors to the closest feature exceeds a user-defined distance threshold δ: $d_{min}(\vec{x}) > \delta$.

Central Tendency

VCP volumes are also useful to estimate the central tendency of location, which denotes the location of points that lie in between the line features in all ensemble members. Centrality was introduced by Demir et al. [DJW16] to obtain the most central locations of an ensemble of closed isosurfaces. To determine most central points, the mean vector $\vec{\mu}$ of the VCP($\vec{x}$) set needs to exhibit the least vector length. It is computed as follows:

$$\vec{\mu}(\vec{x}) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{vcp}_i(\vec{x}). \tag{3.40}$$

A point is also considered to be central if the maximum distance $d_{max}(\vec{x})$ to the farthest close-by feature in all ensemble members is low. Furthermore, another factor f is introduced, which represents the percentage of ensemble members, which have a small distance $||vcp_i(\vec{x})||$ to the next closest features. With this, the centrality $\psi(\vec{x})$ of a voxel is computed as

$$\psi(\vec{x}) = \max\left\{1 - d_{max}(\vec{x}), \sqrt{f \cdot (1 - ||\vec{\mu}(\vec{x})||)}\right\}. \tag{3.41}$$

Note that the values $d_{max}(\vec{x})$ and $\vec{\mu}(\vec{x})$ lie in the range $[0, 1]$, as they have been normalized beforehand. A sketch of the centrality per voxel is provided in Fig. 3.9(b)). Consequently, ridge detection can be used to identify locations of high centrality or high frequency of feature occurrence to retrieve artificial "mean" representation of line ensembles. For further discussion of the advantages and disadvantages of implicit line representations and ridge lines in proxy fields, we refer to our published paper in [KW19a].

3.4.4 Dimensionality Reduction

Clustering on NWP ensemble weather data or implicit line representations, such as VCP volumes, is challenging, as the number of data elements per ensemble member is much higher than the number of ensembles. For instance, assume that the data matrix $X \in \mathbb{R}^{N \times D}$ needs to be clustered where N represents the ensemble members (in our experiments 51) characterized by D data elements, also referred to as features, in the volumetric data. In our experiments, D represents the number of voxels of the used scalar fields, i.e., wind fields of size $D \approx 131 \times 66 \times 70$ (longitude × latitude × hybrid levels), or the number of voxels in the precomputed voxel representations of line sets. For high-dimensional data with $D \gg N$ a clear distinction of individual ensemble members is challenging because multiple data elements can be interrelated. Here, the notion of dimensionality reduction is to transform X to a new matrix $X' \in \mathbb{R}^{N \times C}$ using a very low-dimensional set of data elements $C \ll D$, so that most of the data's variance is retained. The transformation creates a new subset of data elements that are (ideally) uncorrelated and ordered so that the first variables contain the highest variation in the original data. Hence, dimensionality reduction helps to separate ensemble members of different characteristics by using only a small subset of transformed data elements. Different variants of methods exist to reduce the dimension of high-dimensional data using a linear or non-linear transformation. In the following, we present the two most commonly known and used reduction methods.

Principal Component Analysis

In principal component analysis (PCA), the idea is to find the major axes of variance, named principal components (PCs), in high-dimensional data sets. In the following, we briefly discuss PCA while

following the in-detail explanations from the book of Jolliffee [Jol11]. For this, the variance of the D data elements and their covariances with all other $D-1$ data elements needs to be compared. Since variance is measured by computing the squared deviation of data elements from their mean, every row in the data point matrix X with entries x_{ij} needs to be zero-centered around its column-wise mean $\overline{\mu}$, which yields the new zero-centered matrix $\overline{X}$:

$$\begin{aligned} \overline{X} &= X - \kappa\overline{\mu}^T \text{ , where} \\ \overline{\mu} = (\mu_0, \mu_1, ..., \mu_D) \in \mathbb{R}^{D\times 1} &\text{ , } \mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij} \text{ , and} \\ \kappa \in \mathbb{R}^{N\times 1} &\text{ , } \kappa_i = 1 \text{ for } i = 1, ..., n. \end{aligned} \tag{3.42}$$

Given matrix $\overline{X}$, the covariance matrix $\Sigma \in \mathbb{R}^{D\times D}$ can be computed with

$$\Sigma = \frac{1}{N-1}\overline{X}^T\overline{X}. \tag{3.43}$$

To find the set of PCs, we can use eigendecomposition to obtain the normalized eigenvectors ξ_j and their corresponding eigenvalues λ_j. The normalized eigenvectors of the covariance matrix form an orthonormal basis, where each $\xi_j \in \mathbb{R}^{D\times 1}$ represents an orthogonal axis fitted to the data. Given that each λ_j represents the variance of the linear combination of data elements, we can use the eigenvectors with the largest eigenvalues (variance) to transform $\overline{X}$ to a new score matrix X':

$$X' = \overline{X}E \text{ , } x'_{ij} = \overline{x}_i \cdot \xi_j. \tag{3.44}$$

where $E \in \mathbb{R}^{D\times C}$ is a matrix containing the C largest eigenvectors in each column, with $C \leq D$. This equation describes a linear transformation of $\overline{X}$ so that the data is aligned with the eigenvector basis. Note that the original data matrix can be obtained by:

$$X = \kappa\overline{\mu}^T + X'E^T. \tag{3.45}$$

PCA can also be achieved by conducting a singular value decomposition (SVD) on the zero-centered matrix $\overline{X}$. SVD is often preferred for very high-dimensional data as it is computationally faster. Here, $\overline{X}$ can be factorized as follows:

$$\overline{X} = U\Lambda W^T, \tag{3.46}$$

where $U \in \mathbb{R}^{N\times N}$ is comprised of left singular vectors, $\Lambda \in \mathbb{R}^{N\times D}$ is a diagonal matrix containing the singular values σ_i along its diagonal entries Λ_{ii}, and $W \in \mathbb{R}^{D\times D}$ contains the right singular vectors. Since the singular values of Λ are the square roots of the eigenvalues of the covariance matrix X, it follows that the eigenvalues are defined by $\lambda_i = \sigma_i^2$. Hence, the full score matrix X' can be computed

as follows:

$$X' = XW = U\Lambda W^T W = U\Lambda. \tag{3.47}$$

A truncated version of a score matrix $X'_C \in \mathbb{R}^{N\times C}$ can be obtained by just keeping the C columns with the largest singular values in the reduced $\Lambda_C \in \mathbb{R}^{C\times C}$ matrix:

$$X' = U_C\Lambda_C = XW_C, \tag{3.48}$$

where $U_C \in \mathbb{R}^{N\times C}$. The value C is determined by computing the amount of explained variance for each linear combination of the data elements. Assume that the matrices E (with the eigenvectors), U, and Σ_C (with the singular values σ_i) are sorted by the eigenvalues λ_i in descending order: $\lambda_0 \geq \lambda_1 \geq ... \geq \lambda_C$. The explained variance can be computed as follows:

$$\text{exVar}_i = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j} \tag{3.49}$$

C is determined by finding the lowest index C where the sum of all variances explains at least p-percentage of the total variance:

$$\sum_{i=1}^{C} \text{exVar}_i \geq p \tag{3.50}$$

t-Distributed Stochastic Neighbor Embedding

Maaten and Hinton introduced t-Distributed Stochastic Neighbor Embedding (t-SNE), an adapted version of Stochastic Neighbor Embedding (SNE) [HR03], to visualize high-dimensional data. In t-SNE the data is projected to low dimension (e.g., to 2D or 3D) to produce nice visual results of complex data while the local similarity between data elements is still retained visually after the projection. In contrast to SNE, Maaten and Hinton use a symmetric version of SNE by minimizing a single Kullback-Leibler divergence between joint probability distributions in high- and low-dimension. Additionally, they make use of a t-Student distribution instead of a Gaussian to obtain the similarities of two data elements in low-dimension. In the following, we briefly summarize the findings of the paper by Maaten and Hinton [MH08].

In SNE, the Euclidean distances between data elements are transformed into conditional probabilities which represent the similarity between pairs of data elements. Assume that each local point x_i the neighborhood to all other points is defined relative to the Gaussian probability distribution σ_i which is centered at x_i. The similarity of element x_i to x_j is described as the probability that x_i would

choose x_j as its neighbor with

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2/2\sigma_i^2)}. \tag{3.51}$$

The probability $q_{j|i}$ for the low-dimensional counterpart is computed similarly (see [MH08]). The major objective of t-SNE is to minimize the distance between the high- (P_i) and low-resolution probability distribution (Q_i), which can be expressed via the Kullback-Leibler divergence $KL(P_i||Q_i)$. The cost function C for the minimization problem is defined as:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} \tag{3.52}$$

To estimate the variance σ_i of the Gaussian centered at point x_i, a fixed perplexity for the probability distribution P_i is used:

$$\text{Perp}(P_i) = 2^{H(P_i)} \ , \ H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}, \tag{3.53}$$

where $H(P_i)$ is the Shannon entropy. The perplexity can be used to smoothly estimate the number of preferred neighbors, with values between 5 and 50. With t-SNE, Maaten and Horton proposed to use a symmetric SNE to improve the optimization of the cost function C. The symmetric probabilities $p_{ij} = p_{ji}$ and $q_{ij} = q_{ij}$ for points x_i and x_j for high- and low-dimensional space are defined as:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n} \ , \ q_{ij} = \frac{\exp(-||y_i - y_j||^2)}{\sum_{k \neq i} \exp(-||y_k - y_l||^2)} \tag{3.54}$$

The following cost function which describes the Kullback-Leibler divergence between P_i and Q_i is minimized:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{3.55}$$

The final projection results are performed by choosing an initial location of points in low-dimensional space and sequentially minimizing C until convergence. A drawback of this method is that the results depend on the initial set of point locations and the used perplexity, which makes it difficult to control the outcome of t-SNE projections of different data sets.

We used both PCA and t-SNE in our work to improve the results for jet-stream core line clustering based on scalar fields and line geometry. For further details on the selected parameters for PCA (explained variance) and t-SNE (perplexity) within this context, we refer to our published paper [KW19a].

3.5 The Visualization Tool "Met.3D"

In this thesis, we make use of the open-source visualization "Met.3D", developed by Marc Rautenhaus et al. [RKSW15, RGSW15]. It was originally designed to provide tools to visually analyze ensemble data during aircraft-based field campaigns, but over time has become a powerful visualization tool with increasing capabilities. Within the trans-regional collaborative research project Waves-To-Weather, Met.3D has become an integral part of common visual analysis tools, is actively developed, and used in ongoing research projects. In particular, Met.3D provides interactive two- and three-dimensional visualization tools to facilitate the 3D visual analysis of numerical weather prediction data and other atmospheric model data sets. It makes use of state-of-the-art techniques in computer graphics, such as alpha compositing, direct volume rendering, volume ray casting, streamline and pathline computation, particle tracing, and clipping planes to visualize data along horizontal or vertical 2D planes. It also uses a multi-window layout to facilitate the analysis of several atmospheric processes and ensemble data in parallel. The front-end (GUI) and back-end (data request and processing) are completely separated, which means that an interactive rendering of the data is possible while additional data can be retrieved in the background. Once the data is retrieved, it is cached in memory to avoid re-computations of data and multiple hard drive accesses. It also provides a flexible modular framework to process and filter ensemble data or single scalar fields. The filters are represented by separate modules, where each module performs a particular computational task. With this, the filters can be combined in multiple ways to implement a sequence of data filtering techniques. For instance, ensemble data can be aggregated to the mean, or smoothing of scalar fields can be combined with threshold filtering to focus on certain atmospheric features.

Within the project of this thesis, we have integrated the detection of jet-stream core lines, the extraction of 3D atmospheric weather fronts, and the clustering of jet-stream core lines into the tool Met.3D. All gradient, filtering, and smoothing algorithms have been implemented as modular filter techniques. We have also entirely integrated our visualization techniques into Met.3D. Therefore, the screenshots and renderings, which are shown in our published paper [KHS*18, KHS*19, KW19a], were completely rendered with Met.3D.

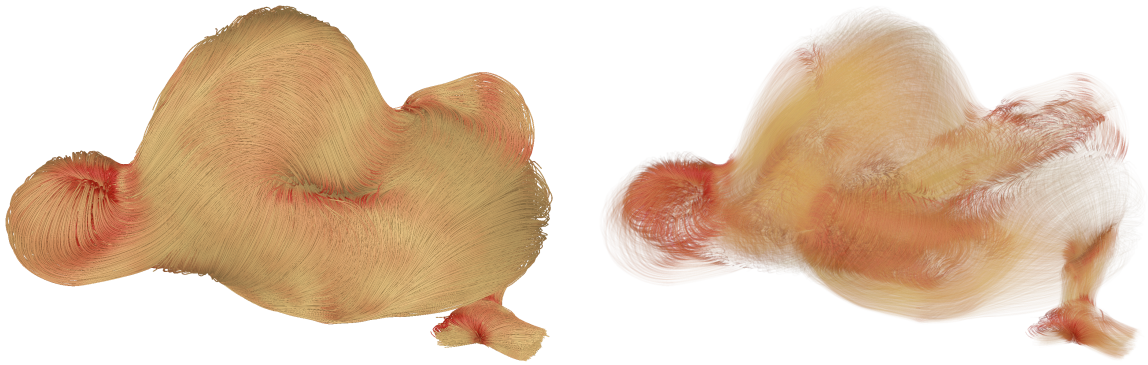


Figure 3.10: *3D rendering of 10000 dense streamlines randomly seeded on the inside of an aneurysm. Color encodes the amount of vorticity of the underlying flow field at each streamline vertex (beige = lowest, red = highest vorticity). The left image depicts lines rendered fully opaque. The right image shows lines rendered with transparency, where vorticity is mapped to opacity. Here, it can be seen that lines of high vorticity in the interior of the aneurysm, which are hidden by other lines in the opaque rendering, can be revealed by fading out low-vorticity lines with transparency.*

3.6 Rendering Line Sets with Transparency

In flow visualization, line features, such as streamlines, are extracted to describe the motion of flow in flow fields. The lines are commonly rendered as opaque 3D (analytic) tubes with a fixed radius. At each line vertex, certain attributes are stored, such as line curvature, vorticity, or velocity. These attributes are mapped to color and displayed on the tube geometry. For a myriad of line sets, however, rendering the entire line data set fully opaque fails to communicate trends in the data, because opaque rendering suffers from multiple occlusion effects and hampers the visual analysis of features-of-interest available in the data. Instead, transparency rendering is used during the visual exploration of large line data sets. For instance, users typically map line characteristics to transparency with user-defined transfer functions to accentuate features-of-interest in large line sets, while fading out lines deemed less important. This procedure improves the visual exploration of specific line features while keeping the overall flow context in their surrounding. Unfortunately, transparency rendering is highly computationally expensive, as all transparent lines need to be blended in correct visibility order. Therefore, researchers look for rendering techniques to display large 3D line sets with millions of transparent layers most realistically and efficiently. In the following, we briefly discuss the theory of alpha-compositing and discuss the two major variants of transparency rendering algorithms.

3.6.1 Alpha Compositing

In computer graphics, Porter and Duff [PD84] introduced the well-known equation to formalize the overlay of two transparent layers, also termed alpha blending or alpha compositing. Alpha (α) in

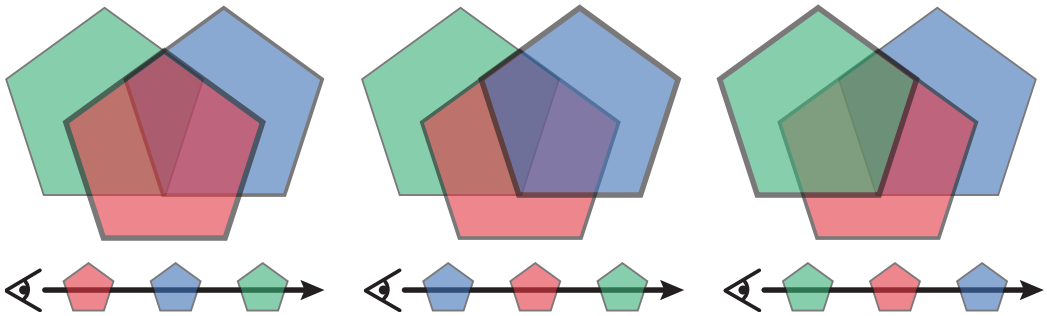


Figure 3.11: *The resulting colors produced by blending transparent planar pentagons (colored in red, green, and blue) in different orders. The blending order is indicated by the arrow plotted below the images. Comparing the kite-like shape, where all three pentagons overlap, it can be observed that all three scenarios yield a different color after alpha blending.*

this context denotes the opacity of a material, or more specifically, the amount of light that is not transmitted through the layer. The more opacity a layer possesses, the less light (color) can be seen from objects concealed by this layer in the direction of reflected light towards the viewer. Based on the functionality of monitors, color in computer graphics is commonly defined in the additive color space RGB, where light is classified into red, green, and blue light. In RGB space, the addition of all of the three light components yields the final color seen on the screen. In the following, the color for a pixel i is defined as $C_i = (r_i, g_i, b_i, \alpha_i)$, where the first three components represent the three light components, and α_i is the corresponding opacity. Porter and Duff defined alpha compositing for rasterization in RGB-space, where the geometry of an object is projected and rasterized onto a 2D pixel grid on the screen, also called fragment or pixel candidate. Every transparent fragment that falls into a pixel after rasterization has to be blended with all other transparent fragments falling into the same pixel in correct visibility order.

Alpha compositing in RGB-space in correct order is vital as this operation is not commutative, which means that blending different colors in arbitrary order produces different transparent colors, as illustrated in Fig. 3.11. Let P and N designate two different fragments falling into the same pixel, with the corresponding colors C_p and C_n and opacities α_p and α_n. Furthermore, assume that fragments are sorted in back-to-front order, hence the distance z_n of the next closest fragment to the viewer is less than the distance z_p of the previous fragment. The blending equation to obtain the final color C_o and opacity α_o after compositing the transparent fragments p and n is defined as follows:

$$C_o = \frac{C_n \alpha_n + (1-\alpha_n) C_p \alpha_p}{\alpha_n + (1-\alpha_n) \alpha_p}. \tag{3.56}$$

This equation is also known as the OVER operator [PD84] A convenient way is to store the color of a fragment pre-multiplied with its opacity. Let $c_i = C_i \alpha_i$ be the pre-multiplied color component

$c_i = (r \cdot \alpha_i, g \cdot \alpha_i, b \cdot \alpha_i)$ for a fragment i with color components red, green, and blue. Then, the blending equation can be written as:

$$\begin{aligned} &[\text{OVER}] \\ c_o &= c_n + (1-\alpha_n)\, c_p \\ \alpha_o &= \alpha_n + (1-\alpha_n)\, \alpha_p \end{aligned} \tag{3.57}$$

Equivalently, an explicit expression for the recursive OVER operator is, cf. Münstermann [MKKP18]:

$$C_o = \sum_{i=1}^{n-1} c_l \cdot \prod_{k=i}^{n-1} (1-\alpha_k), \text{ with } z_k < z_l, \tag{3.58}$$

where $T_i = \prod_{k=i}^{n-1}(1-\alpha_k)$ denotes the transmittance of fragment i, which is the complement to opacity.

The counterpart of the OVER operator is the UNDER operator, where fragments are blended in front-to-back order going from the closest to the farthest fragment. Assume that the colors are pre-multiplied with alpha and the fragment n is blended under the previously blended fragment r with opacities α_n and α_r, respectively. The blending equation for front-to-back blending is then defined as:

$$\begin{aligned} &[\text{UNDER}] \\ c_r &= c_r + \alpha_r\, c_n \\ \alpha_r &= (1-\alpha_n)\, \alpha_r \end{aligned} \tag{3.59}$$

A detailed explanation of this formula and its derivation is provided by Bavoil and Myers [BM08]. In the following, we assume that the lines are either represented by triangle meshes denoting line tubes with a fixed radius or represented by analytic tubes defined at each line segment.

Order-Independent Transparency

A simple approach to render transparent objects is to sort the triangle meshes of all displayed objects in either front-to-back or back-to-front order and to make use of the corresponding blending equations to obtain the final colors per pixel. However, triangle meshes cannot be simply ordered by the distance to the viewer (depth) as they exhibit different spatial extents and orientations and can also overlap.

Therefore, a variety of rendering techniques has been proposed within the last two decades to solve the problem of a correct visibility order for objects of arbitrary shapes. Within this context, order independent transparency (OIT) deals with the design of algorithms that strive to correctly blend transparent objects which are processed in arbitrary order, so that the objects' geometry in the scene does not need to be sorted beforehand. OIT is mainly linked to object-order techniques which

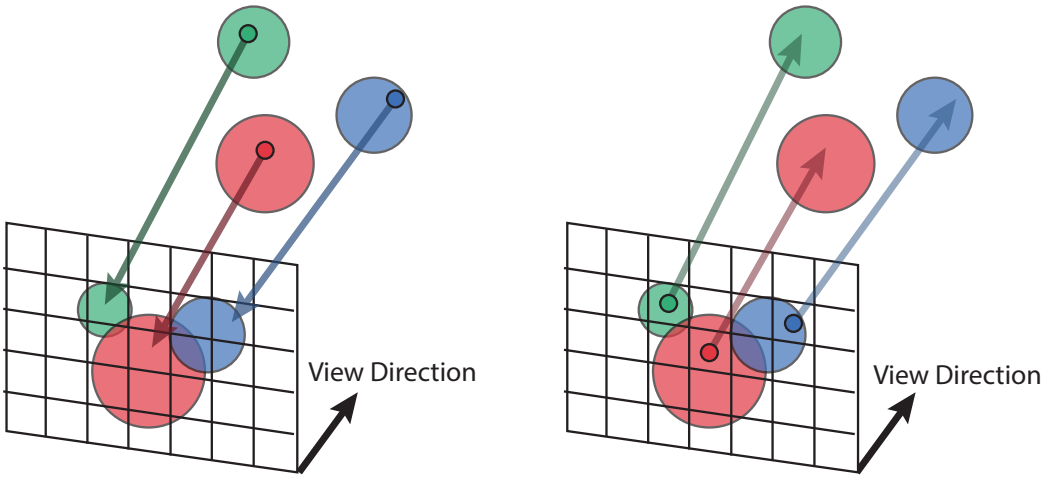


Figure 3.12: *A simplified sketch of two rendering techniques obtaining the final per-pixel color on a pixel grid (viewport) with fixed resolution. Left: Depiction of object-order rendering, where the geometry of the objects is projected and rasterized onto the pixel grid. Right: Image-order rendering, where a ray is shot from a pixel center into the scene. The color of the pixel is defined by computing the intersection point of the ray with an object and obtaining the object's color value at that point.*

operate on triangles projected onto the viewport pixel grid (cf. Fig. 3.12(a)). A first option is to render the scene layer-by-layer with Depth Peeling (DP) [Eve01], where at each iteration the closest fragments are rendered and blended with the current color buffer, while their depth values are stored in a second depth buffer. The second buffer is used in the next iteration to peel away the previous closest fragments and to blend the next layer of fragments with the color buffer. The rendering time of DP, however, is heavily increasing with the complexity of line sets. Thus, DP should not be used to interactively explore large line sets.

Many other variants of object-order techniques are comprised of two phases. In the first part, all fragments falling into the pixel grid are gathered in a local or global buffer, which stores the distance of a fragment to the viewer (depth), its color, and its opacity or transmittance. In the second part, all gathered fragments belonging to the same pixel are sorted by depth and finally blended using either the OVER or UNDER operator, depending on the type of order. Regarding the gathering procedure, different approaches exist to store and sort fragments for each pixel. For instance, a global buffer is used, which theoretically is supposed to be large enough to hold all fragments generated during a render pass for a given viewport size. However, this is not always possible in practice as the amount of video memory on the GPU is limited, which poses a serious problem in scenes with thousands of triangles producing millions of fragments. Alternatively, some techniques only store a fixed number of fragments in a local per-pixel array and merge new fragments into this array heuristically. This approach supports rendering on hardware with low video memory. A detailed list of different OIT

techniques was already given in 2.4. Further detailed evaluation of the performance of OIT algorithms wrt. line rendering with transparency is provided in our evaluation paper [KNM*20].

3.6.2 Ray Tracing

Another class of rendering techniques is called ray tracing. With ray tracing, rays are shot into the scene from the pixel centers of the image pixel grid (cf. schematic in Fig. 3.12(b)). While the scene is traversed along the ray, all possible intersection points with any object in the scene are computed. Usually, this requires testing millions of rays against thousands of objects, which is highly computationally expensive and ineffective, as a major part of the objects' geometry is missed by a single ray. Hence, acceleration structures are used to improve the performance of ray traversals through the scene. Among those acceleration structures are boundary-volume-hierarchies (BVH). Here, a simple boundary volume (BV), such as a sphere or an axis-aligned bounding box, is defined for each part of the tube geometry. The BVH is constructed by sequentially merging the nearest two boundary volumes in a bottom-up fashion until the root node of the hierarchy contains all BVs and geometries. BVs are useful to quickly test whether a ray can hit a triangle of an object within this volume. The constructed BVH can be employed to quickly obtain all triangles that can intersect with a ray, which facilitates a fast computation of all intersection points. Since the rays start at the pixel centers, which are closest to the viewer, the front-to-back order of all intersection points is implicitly determined by the BVH traversal. Thus, the UNDER operator can be used in this case to conduct alpha blending.

Another alternative is to use a voxel representation of line sets. Here, lines are rasterized into a 3D grid with a fixed voxel resolution in Euclidean space. Each voxel locally stores all line segments that intersect with it. The voxelization of lines is conducted in a preprocessing step and used as (acceleration) data structure during ray tracing. For instance, voxels that do not intersect with any lines are marked empty and can be skipped during ray traversal. On the other hand, when a ray hits a non-empty voxel, all line segments intersecting with this voxel are retrieved and the ray is analytically intersected with the tube of a line segment to finally render the corresponding line tubes. An in-detail discussion of such algorithms is also provided in our comparison study [KNM*20].

In the next chapter, we will provide the summaries, author contributions, and copyrights of all four appended publications associated with this thesis and we will then conclude this thesis in Chapter 5 with a discussion of our results and an outlook for potential future work.

4 Summary of Papers

4.1 Robust Detection and Visualization of Jet-Stream Core Lines in Atmospheric Flow

Summary

Although jet-streams, meandering air currents of high wind speed, and their core lines (maxima lines of wind speed) have been well studied for decades in meteorology, there is still no automated method to identify these line features. To the best of our knowledge, forecasters manually analyze 2D charts to retrieve core lines, which are defined as height ridges of wind speed. However, tools are missing to identify and analyze core lines in 3D. In this paper, we present a robust detection method to identify 3D jet-stream core lines in atmospheric dynamics. Our method exploits the wind direction to define a local, well-defined frame-of-reference and identifies core lines within this frame. In contrast to ridge detection, our method is more stable and avoids the computation of the Hessian and its eigenvectors. In close collaboration with meteorologists, we have conducted a meteorological analysis of the jet-stream core lines and surrounding atmospheric processes through real-world case scenarios, for the first time. As this analysis was not possible before, we also show the benefit of our detection method to forecasting and research. Full publication is provided in Sec. 7.1.

Author Contribution

The first author was responsible for the implementation of the detection algorithm and visualization techniques. In collaboration with Prof. Dr. Filip Sadlo, the comparison of the detection algorithm with “classical” ridge detection was discussed and compared. Dr. Marc Rautenhaus and Tim Hewson conducted the meteorological analysis using the detection algorithm and were responsible for the case studies. In discussions with Prof. Dr. Rüdiger Westermann and Dr. Marc Rautenhaus, we designed

the visualization techniques used to depict the 3D line features in combination with other atmospheric processes.

4.2 Interactive 3D Visual Analysis of Atmospheric Fronts

Summary

Atmospheric fronts are boundaries between two air masses of different characteristics, such as temperature or moisture. They are fundamental features in meteorology as they are associated with abrupt weather changes and severe weather events, like thunderstorms or heavy precipitation. Although fronts are conceived as 3D surfaces in meteorological concepts, existing objective methods only focused on the detection and visualization of fronts in 2D. In this paper, we extend an existing 2D detection method to 3D and propose a visualization technique to analyze the 3D structure of fronts, as well as the relation between fronts and related atmospheric processes. For the detection method, we use wet-bulb potential temperature as the primary thermal quantity to identify both high temperature and moisture gradients. As thermal gradients vary with height, we allow the users to softly filter fronts by mapping their characteristics, such as frontal strength, to opacity via transfer functions. We further introduce normal curves to visualize the transition zone of high thermal gradients between two air masses (frontal zone). To conduct a statistical analysis of single fronts and their associated frontal zone, we estimate the volume of the frontal zone in 3D by identifying grid boxes intersecting with normal curves traced from a selected front. We also demonstrate the benefit of our method by means of an idealized water planet simulation and two-real world case scenarios. Full publication is provided in Sec. 7.2.

Author Contribution

The major contribution of the first author is the implementation of front detection and soft filtering of front features, the computation of frontal zone volumes, and the implementation of visualization techniques to depict frontal surfaces and frontal zones. Tim Hewson, Andreas Schäfler, and Dr. Marc Rautenhaus were mainly responsible for the case studies and domain expert feedback. Visualization techniques and the improvement of the front detection schemes were designed in close collaboration with Prof. Dr. Rüdiger Westermann and Dr. Marc Rautenhaus.

4.3 Clustering Ensembles of 3D Jet-Stream Core Lines

Summary

Jet-stream core lines, identified in wind fields, can vastly differ in topology and orientation and can possess single or multiple interruptions. Especially in ensembles of wind fields, these structures can highly differ in major parts between ensemble members and thus show high variation. In this paper, we examine different clustering approaches to infer trends in an ensemble of jet-stream core lines. Three major clustering strategies are proposed: (a) we cluster on the scalar fields, which are used to identify core lines, (b) we cluster on the 3D line sets using the Hausdorff distance, (c) we compute a line-implicit representation of core lines and cluster on this representation. After clustering, we obtain and visualize the best representative line feature in a cluster. Furthermore, we compute two proxy fields to describe the amount of density at a point or the central tendency in the data. The density is represented by a 3D visitation map, which is the frequency of feature occurrences at a point, while the central tendency is computed by the ensemble of vector-to-closest-points. Ridge detection is applied to the proxy fields to obtain "artificial" representatives of an ensemble of core lines. We compare the clustering and artificial representatives to core lines extracted from re-analysis data and discuss the potential of our techniques in terms of trend inference. Full publication is provided in Sec. 7.3.

Author Contribution

The first author was mainly responsible for the implementation of clustering frameworks for jet-stream core lines, the generation of proxy fields and implicit line representations, ridge line extraction in proxy fields, and the evaluation of these techniques. Discussions with Prof. Dr. Rüdiger Westermann led to the final design of the clustering frameworks and visualization techniques.

4.4 A Comparison of Rendering Techniques for 3D Line Sets with Transparency

Summary

Transparency rendering is a common technique to explore large sets of particle trajectories in flow fields. Based on the line properties or the attributes assigned with the trajectory curves, transparency is used to reveal features-of-interest while fading out all other remaining trajectories. To create a proper visualization of transparent lines, transparency rendering requires the correct alpha compositing of myriads of transparent layers which have to be sorted before alpha blending in correct visibility order. However, sorting and blending for thousands of layers per pixel is highly computationally expensive and becomes increasingly difficult. In this paper, we present an extensive comparison study of transparency rendering techniques for transparent 3D line sets on the GPU or CPU. We examine different variants of line rendering algorithms that either accurately compute or approximate the blending equation. All techniques are compared concerning their run-time performance, memory consumption, and image quality. It is also analyzed how the complexity of line sets along with different transparency and color mappings affect the performance and quality of the rendering techniques. We also propose a better sorting algorithm to quickly retrieve the correct visibility order of fragments on the GPU with Per-Pixel Fragment Lists and divide the scene into buckets to enhance the quality of Multi-Layer Alpha Blending. In the end, we provide an in-detail discussion of all algorithms to assist the users in choosing the appropriate rendering technique for different transparency settings. Full publication is provided in Sec. 7.4.

Author Contribution

The first author was responsible for the entire comparison study of all techniques. This includes all performance tests, the analysis of memory consumption, and the assessment of image quality followed by a user study. Christoph Neuhauser designed a software, called PixelSyncOIT, to implement and compare all object-order techniques and voxel ray casting presented in the paper. In discussions with the first author, he designed an improved version of Multi-Layer-Alpha Blending with bucket sorting. Torben Maack implemented GPU ray tracing of transparent line sets using the RTX API via Vulkan on the latest NVIDIA RTX graphics cards. Mengjiao Han and Will Usher helped to implement CPU ray tracing, and, in particular, Mengjiao Han conducted performance tests on the CPU. Together with Prof. Dr. Rüdiger Westermann, the major content of this paper was created.

5 Final Discussion

5.1 Future Work

The visual analysis of atmospheric features can further be improved in the future by taking into account both the spatial and temporal evolution of atmospheric features in ensemble weather forecasts. For instance, our detection methods could be extended to a more probabilistic approach, where the probability at a local point indicates the likelihood of feature occurrence in an ensemble data or forecast series. Optical flow techniques could also be used to track individual sets of features and to convey possible split and merge events of features over time. In addition to that, existing or new formulas for the genesis and dissolution of atmospheric features could be incorporated into the automated detection process. We have also encountered that feature characteristics can greatly vary with height. For instance, features near the surface tend to have stronger characteristics than those in the upper atmosphere. Here, it would be interesting to investigate whether the filter criteria could be modified to automatically adapt to the altitude or location of a feature. Regarding atmospheric fronts, we have encountered that frontal surfaces suffered from multiple interruptions due to vanishing horizontal gradients. This is mainly because only the 2D horizontal thermal gradients are considered during the detection of frontal surfaces. It could be investigated in the future whether the vertical component of thermal gradients or additional parameters could be used to improve this detection scheme.

Concerning the clustering of ensembles of jet-stream core lines, it was not always possible with binary clustering to assign lines to specific disjoint groups. Therefore, the binary classification of groups in ensembles of lines could be replaced by fuzzy clustering, where each ensemble member is assigned with a probability that it belongs to a certain group. This probability could be used to visualize the uncertainty in an ensemble of shapes. Furthermore, there is room for improvement for the comparison of line sets with vastly different topology. As the Hausdorff metric and mean-closest point distance do not always best represent the (dis-)similarity of line sets, the design of a better

similarity metric or an alternative (implicit) representation of line sets is desirable to improve line clustering.

In terms of transparency rendering, we have encountered during the user study that the comparison of image quality between the result of approximate techniques and ground truth renderings is not trivial. Image quality metrics, such as peak-signal-to-noise ratio or the structured similarity index have often failed to communicate whether the quality is sufficient to interpret large line sets, while avoiding any misinterpretations of the data. In the future, it will be interesting to conduct further user-studies to investigate within the context of scientific visualization, how the visual perception of large line sets is affected by rendering artifacts produced by approximate rendering techniques.

Within the last years, the application of neural networks (deep learning) has greatly emerged for nearly every different research area. Regarding the definition of fronts, for instance, meteorologists have not yet agreed on the question of which physical quantities should be used to objectively identify fronts. Neural networks could help to automatically identify features from a set of several physical quantities. 2D weather charts from weather centers could be used here to provide the networks with information about manually identified atmospheric features during their training process. Given that neural networks can identify these features, this procedure could support forecasters to better understand how quantities are related to atmospheric events and may help the meteorological community to improve existing feature definitions in the future. Neural networks have also successfully been trained to deliver a score that measures the similarity between two images wrt. visual perception, such as with the Learned Perceptual Image Patch Similarity (LPIPS) [ZIE*18]. Hence, networks could also be employed to estimate the similarity of different line or surface features extracted from numerical weather data. However, the similarity of line features needs to be defined, for instance, by experts to inform the networks during training which pairs of line sets are considered to be (dis-)similar based on their topology and orientation.

In terms of clustering, existing dimensionality reduction and clustering techniques fail to communicate trends in thousands of numerical weather ensembles, either due to the massive dimensionality and noise in the data or due to high computational costs. Hence, autoencoding strategies, such as (conditional) variational autoencoder [KW19b, SLY15], could be employed to infer a low-dimensional representation from high-dimensional ensemble data. Variational autoencoders learn to reproduce their original input while inferring a statistical distribution encoded in a low-dimensional mapping. The statistical distributions are often expressed as Gaussian distribution with mean and standard deviation. Thus, the learned distributions could be used to describe the uncertainty of feature occurrence in the data.

5.2 Conclusion

In this thesis, we have proposed detection methods and visualization techniques to automatically extract and visualize 3D atmospheric features and their characteristics in numerical weather data. This facilitates a novel 3D visual analysis of atmospheric processes, which has not been possible beforehand in meteorology, as meteorologists relied on classical 2D approaches operating on single height levels. Concerning feature detection, we have proposed a robust 3D detection method to identify jet-stream core lines from volumetric wind fields [KHS*18]. Our method is similar to ridge detection but exploits the local wind direction to define a consistent frame-of-reference. With this, we have demonstrated that our method is numerically stable and does not require any specific image-scale space to properly identify core line features. We also employ color and tube-thickness mapping to depict the distribution of atmospheric quantities along the displayed line features. In [KHS*19], we have also introduced a novel full 3D detection and interactive visualization technique to examine 3D atmospheric fronts. Based upon the classical 2D objective front detection [Hew98], we retrieve all loci of the 3D frontal surface at each vertical level and render its geometry with Marching Cubes or isosurface ray casting. We further make use of normal curves to depict the frontal zone and employ color-mapping to display physical quantities along the frontal surface or normal curves. Single front features can also be selected to collect statistical information of atmospheric quantities within the associated frontal zone. Since the characteristics of fronts vary with height levels, our framework allows users to softly filter front features by mapping the used filter criteria to transparency.

To demonstrate the benefit of our detection methods, we have conducted several meteorological analyses of real-world case scenarios and have shown that both our 3D depiction and analysis of atmospheric features provide forecasters with new insights into the relationship between different physical processes. Our visualization methods are not only useful for forecasting and meteorological analyses but can also be used to automatically produce improved significant weather charts (SIGWX) used worldwide for commercial aviation purposes. In summary, we are confident that our novel 3D visualization methodologies for the depiction and analysis of atmospheric features can help to improve the understanding of atmospheric processes and their interrelationship at all height levels.

We have also investigated the suitability of clustering to convey trends in an ensemble of vastly different jet-stream core lines [KW19a]. In particular, three different clustering strategies have been introduced which operate on either a) derived scalar fields, b) line geometry, or c) vector-to-closest point volumes. Comparing the clustering results with trends contained in reanalysis data, we have shown that our proposed strategies have the potential to reveal trends in an ensemble of line features. Moreover, we have proposed methods to retrieve the frequency of feature occurrence via visitation maps and obtain the central tendency in the ensemble data from vector-to-closest-point volumes. Ridge detection can be used on these proxy fields to obtain artificial representations of a specific set

of jet-stream core lines. Our experiments have shown that these representations can also compete with clustering methods in terms of trend inference. However, no clear recommendation for any clustering technique or proxy geometry can be made as the results highly depend on the complexity of the line sets. Nevertheless, we have found that a line-implicit representation is the most promising technique to cluster vastly different line geometry as it can operate on line sets independent of their topology and orientation.

Decoupled from specific atmospheric features, we have also conducted an exhaustive study of rendering techniques to display large line sets with transparency [KNM*20]. In this study, we have compared different CPU and GPU algorithms wrt. run-time performance, memory consumption, and image quality. The examined algorithms make use of either rasterization (object-order) or ray tracing (image-order) and precisely or approximately compute the alpha-blending equation for thousands of transparent layers with color. Based on our comparison study, we have proposed an improved sorting strategy for Per-Pixel Linked Lists and an enhanced version Multi-Layer-Alpha-Blending (MLAB) by dividing the scene into depth buckets. Furthermore, we have presented an algorithm to render transparent line sets on the latest RTX graphics cards using the provided RTX rendering API.

The major findings of our study are that approximate rasterization-based approaches can attain rendering results of acceptable image quality in real-time, while keeping the memory consumption low. On the other hand, image-based approaches provide high-quality renderings, but come at the expense of huge memory requirements and rather low run-time performance. Nevertheless, we have encountered that transparency itself hampers the visual perception of spatial relationships between multiple line sets. As already shown by Kanzler et al. [KRW18], global illumination (GI) effects, such as ambient occlusion and self-shadowing, help to improve the visual perception of lines. As GI effects cannot be easily integrated into object-order techniques, image-order approaches should be preferred in this context. If GI effects are not desired, we suggest using MLABDB for scenes with a moderate depth complexity and can recommend Moment-Based Order-Independent Transparency for highly large line sets depicted with multiple colors and transparency settings. Regarding approximate techniques for object- and image-order rendering techniques, we have also conducted an informal user-study to evaluate whether their image quality is acceptable wrt. scientific visualization. While approximate rendering techniques produce errors in the final image, users have stated that they are still able to locate features-of-interest and have concluded that the visual perception of line sets, despite approximation errors, is retained. We also see potential in the usage of hardware-accelerated ray tracing with NVIDIA RTX graphics cards, as all transparent lines sets, used in our comparison, can be rendered in real-time on the GPU. This makes the design of GPU-based ray tracing exciting for future scientific visualization projects.

6 Bibliography

[AC08] ARCHER C. L., CALDEIRA K.: Historical trends in the jet streams. *Geophys. Res. Lett. 35*, 8 (Apr. 2008), L08803+.

[AL09] AILA T., LAINE S.: Understanding the efficiency of ray traversal on gpus. In *Proceedings of the Conference on High Performance Graphics 2009* (New York, NY, USA, 2009), HPG '09, Association for Computing Machinery, p. 145–149.

[Ame18] AMERICAN METEOROLOGICAL SOCIETY: "front". glossary of meteorology. `http://glossary.ametsoc.org/wiki/Front`, 2018. Accessed 31 March 2018.

[AMT*12] AUER S., MACDONALD C. B., TREIB M., SCHNEIDER J., WESTERMANN R.: Real-time fluid effects on surfaces using the closest point method. *Computer Graphics Forum 31*, 6 (2012), 1909–1923.

[ASE16] ATHAWALE T., SAKHAEE E., ENTEZARI A.: Isosurface visualization of data with nonparametric models for uncertainty. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (Jan 2016), 777–786.

[AV07] ARTHUR D., VASSILVITSKII S.: K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms* (USA, 2007), SODA '07, Society for Industrial and Applied Mathematics, p. 1027–1035.

[BBP*05] BLAAS J., BOTHA C. P., PETERS B., VOS F. M., POST F. H.: Fast and reproducible fiber bundle selection in dti visualization. In *VIS 05. IEEE Visualization, 2005.* (Oct 2005), pp. 59–64.

[BCL*07] BAVOIL L., CALLAHAN S. P., LEFOHN A., COMBA J. A. L. D., SILVA C. T.: Multi-fragment effects on the gpu using the k-buffer. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2007), I3D '07, ACM, pp. 97–104.

[BE09] BARTON N. P., ELLIS A. W.: Variability in wintertime position and strength of the North Pacific jet stream as represented by re-analysis data. *Int. J. Climatol. 29*, 6 (May 2009), 851–862.

[BFMW12] BÜRGER K., FRAEDRICH R., MERHOF D., WESTERMANN R.: Instant visitation maps for interactive visualization of uncertain particle trajectories. In *Proceedings Visualization and Data Analysis 2012* (2012), vol. SPIE 8294, pp. 82940P–82940P–12.

[BGG19] BAEZA ROJO I., GROSS M., GUENTHER T.: Fourier opacity optimization for scalable exploration. *IEEE Transactions on Visualization and Computer Graphics* (2019), 1–1.

[BHGK14] BEHAM M., HERZNER W., GRÖLLER M. E., KEHRER J.: Cupid: Cluster-based exploration of geometry generators with parallel coordinates and radial trees. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1693–1702.

[BHJ*14] BONNEAU G.-P., HEGE H.-C., JOHNSON C., OLIVEIRA M., POTTER K., RHEINGANS P., SCHULTZ T.: Overview and state-of-the-art of uncertainty visualization. In *Scientific Visualization*. Springer, 2014, pp. 3–27.

[BKS04] BORDOLOI U., KAO D., SHEN H.-W.: Visualization techniques for spatial probability density function data. *Data Science Journal 3* (01 2004), 153–162.

[BM08] BAVOIL L., MYERS K.: Order independent transparency with dual depth peeling.

[BM10a] BRUCKNER S., MÖLLER T.: Result-driven exploration of simulation parameter spaces for visual effects design. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1468–1476.

[BM10b] BRUCKNER S., MÖLLER T.: Isosurface similarity maps. *Computer Graphics Forum 29*, 3 (2010), 773–782.

[BN78] BLUM H., NAGEL R. N.: Shape description using weighted symmetric axis features. *Pattern Recognition 10*, 3 (1978), 167 – 180. The Proceedings of the IEEE Computer Society Conference.

[Bre54] BRETON D. C.: Note sur les lignes de faîte et de thalweg. *Comptes rendus hebdomadaires des séances de l'académie des Sciences 39* (1854), 647–648.

[BRJ11] BERRY G., REEDER M. J., JAKOB C.: A global climatology of atmospheric fronts. *Geophys. Res. Lett. 38*, 4 (Feb. 2011), L04809+.

[BS22] BJERKNES J., SOLBERG H.: Life cycles of cyclones and the polar front theory of atmospheric circulation. *Geofys. Publ. 3* (1922), 1–18.

[BSB*20] BADER R., SPRENGER M., BAN N., RÜDISÜHLI S., SCHÄR C., GÜNTHER T.: Extraction and visual analysis of potential vorticity banners around the alps. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (Jan 2020), 259–269.

[BTH07] BERRY G., THORNCROFT C., HEWSON T.: African easterly waves during 2004—Analysis using objective techniques. *Mon. Wea. Rev. 135*, 4 (Apr. 2007), 1251–1267.

[Car84] CARPENTER L.: The a -buffer, an antialiased hidden surface method. *SIGGRAPH Comput. Graph. 18*, 3 (Jan. 1984), 103–108.

[CDD06] CARR H., DUFFY B., DENBY B.: On histograms and isosurface statistics. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1259–1266.

[CGG04] COROUGE I., GOUTTARD S., GERIG G.: Towards a shape model of white matter fiber bundles using diffusion tensor mri. In *2004 2nd IEEE International Symposium on Biomedical Imaging: Nano to Macro (IEEE Cat No. 04EX821)* (2004), pp. 344–347 Vol. 1.

[CNLE09] CRASSIN C., NEYRET F., LEFEBVRE S., EISEMANN E.: Gigavoxels: Ray-guided streaming for efficient and detailed voxel rendering. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2009), I3D '09, ACM, pp. 15–22.

[Dam98] DAMON J.: Generic structure of two-dimensional images under gaussian blurring. *SIAM Journal of Applied Mathematics 59* (1998), 97–138.

[Dam99] DAMON J.: Properties of ridges and cores for two-dimensional images. *J. Math. Imaging Vis. 10*, 2 (Mar. 1999), 163–174.

[DDW14] DEMIR I., DICK C., WESTERMANN R.: Multi-charts for comparative 3d ensemble visualization. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (Dec 2014), 2694–2703.

[DJW16] DEMIR I., JAREMA M., WESTERMANN R.: Visualizing the central tendency of ensembles of shapes. In *SIGGRAPH Asia 2016 Symposium on Visualization* (New York, NY, USA, 2016), SA '16, ACM.

[DKW16] DEMIR I., KEHRER J., WESTERMANN R.: Screen-space silhouettes for visualizing ensembles of 3d isosurfaces. In *2016 IEEE Pacific Visualization Symposium (PacificVis)* (2016), pp. 204–208.

[DS15] DUTTA S., SHEN H.-W.: Distribution driven extraction and tracking of features for time-varying data analysis. *IEEE transactions on visualization and computer graphics 22*, 1 (2015), 837–846.

[dSV52] DE SAINT-VENANT M.: Surfaces à plus grande pente constituées sur des lignes courbes. *Bulletin de la Société Philomathématique de Paris* (1852), 24–30.

[EGM*94] EBERLY D., GARDNER R., MORSE B., PIZER S., SCHARLACH C.: Ridges for image analysis. *Journal of Mathematical Imaging and Vision 4*, 4 (1994), 353–373.

[ELC*12] EDMUNDS M., LARAMEE R. S., CHEN G., MAX N., ZHANG E., WARE C.: Surface-based flow visualization. *Computers & Graphics 36*, 8 (2012), 974 – 990. Graphics Interaction Virtual Environments and Applications 2012.

[Eli62] ELIASSEN A.: On the vertical circulation in frontal zones. *Geofys. publ 24*, 4 (1962), 147–160.

[ELL09] EVERITT B. S., LANDAU S., LEESE M.: *Cluster Analysis*, 4th ed. Wiley Publishing, 2009.

[EM57] ENDLICH R. M., MCLEAN G. S.: The structure of the jet stream core. *Journal of Meteorology 14*, 6 (1957), 543–552.

[ESSL10] ENDERTON E., SINTORN E., SHIRLEY P., LUEBKE D.: Stochastic transparency. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, ACM, pp. 157–164.

[Eve01] EVERITT C.: Interactive order-independent transparency. *NVIDIA Corporation 2* (10 2001).

[FAGM11] FRAME T. H. A., AMBAUM M. H. P., GRAY S. L., METHVEN J.: Ensemble prediction of transitions of the north atlantic eddy-driven jet. *Quarterly Journal of the Royal Meteorological Society 137*, 658 (2011), 1288–1297.

[FBTW10] FERSTL F., BURGER K., THEISEL H., WESTERMANN R.: Interactive separating streak surfaces. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov 2010), 1569–1577.

[FBW16] FERSTL F., BÜRGER K., WESTERMANN R.: Streamline variability plots for characterizing the uncertainty in vector field ensembles. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (Jan 2016), 767–776.

[FC11] FERRANTI L., CORTI S.: New clustering products. *ECMWF Newsletter 127* (2011), 6–11.

[FFST19] FAVELIER G., FARAJ N., SUMMA B., TIERNY J.: Persistence atlas for critical point variability in ensembles. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan 2019), 1152–1162.

[FKRW16] FERSTL F., KANZLER M., RAUTENHAUS M., WESTERMANN R.: Visual analysis of spatial variability and global correlations in ensembles of iso-contours. *Computer Graphics Forum (Proc. EuroVis) 35*, 3 (2016), 221–230.

[FKRW17] FERSTL F., KANZLER M., RAUTENHAUS M., WESTERMANN R.: Time-hierarchical clustering and visualization of weather forecast ensembles. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan 2017), 831–840.

[For65] FORGY E.: Cluster analysis of multivariate data: Efficiency versus interpretability of classification. *Biometrics 21*, 3 (1965), 768–769.

[FP01] FURST J. D., PIZER S. M.: Marching ridges. In *SIP* (2001), Hamza M. H., (Ed.), IASTED/ACTA Press, pp. 22–26.

[FPE96] FURST J. D., PIZER S. M., EBERLY D. H.: Marching cores: a method for extracting cores from 3d medical images. In *Proceedings of the Workshop on Mathematical Methods in Biomedical Image Analysis* (June 1996), pp. 124–130.

[GRGH*05] GALLEGO D., RIBERA P., GARCIA-HERRERA R., HERNANDEZ E., GIMENO L.: A new look for the southern hemisphere jet stream. 607–621.

[GRT13] GÜNTHER T., RÖSSL C., THEISEL H.: Opacity optimization for 3d line fields. *ACM Trans. Graph. 32*, 4 (July 2013).

[GT18] GÜNTHER T., THEISEL H.: The state of the art in vortex extraction. *Computer Graphics Forum 37*, 6 (2018), 149–173.

[Har83] HARALICK R. M.: Ridges and valleys on digital images. *Computer Vision, Graphics, and Image Processing 22*, 1 (1983), 28 – 38.

[Hau14] HAUSDORFF F.: *Grundzüge der Mengenlehre*, vol. 7. von Veit, 1914.

[HBB*19] HERSBACH H., BELL W., BERRISFORD P., HORÁNYI A., J. M.-S., NICOLAS J., RADU R., SCHEPERS D., SIMMONS A., SOCI C., DEE D.: Global reanalysis: goodbye era-interim, hello era5. 17–24.

[HBS18] HAZARIKA S., BISWAS A., SHEN H.: Uncertainty visualization using copula-based analysis in mixed distribution models. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan 2018), 934–943.

[HDSC19] HAZARIKA S., DUTTA S., SHEN H., CHEN J.: Codda: A flexible copula-based distribution driven analysis framework for large-scale multivariate data. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan 2019), 1214–1224.

[Hew98] HEWSON T. D.: Objective fronts. *Met. Apps 5*, 1 (1998), 37–65.

[HKP*14] HOPE P., KEAY K., POOK M., CATTO J., SIMMONDS I., MILLS G., MCINTOSH P., RISBEY J., BERRY G.: A comparison of automated methods of front recognition for climate studies: A case study in southwest western australia. *Monthly Weather Review 142*, 1 (Jan. 2014), 343–363.

[HMZ*14] HÖLLT T., MAGDY A., ZHAN P., CHEN G., GOPALAKRISHNAN G., HOTEIT I., HANSEN C. D., HADWIGER M.: Ovis: A framework for visual analysisof ocean forecast ensembles. *IEEE Transactions on Visualization and Computer Graphics 20*, 8 (2014), 1114–1126.

[HOGJ13] HUMMEL M., OBERMAIER H., GARTH C., JOY K.: Comparative visual analysis of Lagrangian transport in CFD ensembles. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2743–2752.

[HR03] HINTON G. E., ROWEIS S. T.: Stochastic neighbor embedding. In *Advances in Neural Information Processing Systems 15*, Becker S., Thrun S., Obermayer K., (Eds.). MIT Press, 2003, pp. 857–864.

[HT10] HEWSON T. D., TITLEY H. A.: Objective identification, typing and tracking of the complete lifecycles of cyclonic features at high spatial resolution. *Met. Apps 17*, 3 (2010), 355–381.

[HWU*19] HAN M., WALD I., USHER W., WU Q., WANG F., PASCUCCI V., HANSEN C. D., JOHNSON C. R.: Ray Tracing Generalized Tube Primitives: Method and Applications. *Computer Graphics Forum* (2019).

[JB10] JANSEN J., BAVOIL L.: Fourier opacity mapping. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, ACM, pp. 165–172.

[JDKW15] JAREMA M., DEMIR I., KEHRER J., WESTERMANN R.: Comparative visual analysis of vector field ensembles. In *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on* (Oct 2015), pp. 81–88.

[Jol11] JOLLIFFE I.: *Principal Component Analysis*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 1094–1096.

[Jr.63] JR. J. H. W.: Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association 58*, 301 (1963), 236–244.

[JSS*09] JENKNER J., SPRENGER M., SCHWENK I., SCHWIERZ C., DIERER S., LEUENBERGER D.: Detection and climatology of fronts in a high-resolution model reanalysis over the alps. *Meteorological Applications 17*, 1 (Mar. 2009), n/a.

[Kas03] KASPAR M.: Objective frontal analysis techniques applied to Extreme/Non-extreme precipitation events. 605–631.

[KCH*18] KINDLMANN G., CHIW C., HUYNH T., GYULASSY A., REPPY J., BREMER P.-T.: Rendering and extracting extremal features in 3d fields. *Computer Graphics Forum 37*, 3 (2018), 525–536.

[KESW09] KINDLMANN G. L., ESTEPAR R. S. J., SMITH S. M., WESTIN C.: Sampling and visualizing creases with scale-space particles. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov 2009), 1415–1424.

[KH13] KEHRER J., HAUSER H.: Visualization and visual analysis of multifaceted scientific data: A survey. *IEEE Transactions on Visualization and Computer Graphics 19* (03 2013), 495–513.

[KHGR02] KNISS J., HANSEN C., GRENIER M., ROBINSON T.: Volume rendering multivariate data to visualize meteorological simulations: A case study. In *Proceedings of the Symposium on Data Visualisation 2002* (Goslar, DEU, 2002), VISSYM '02, Eurographics Association, p. 189–ff.

[KHL99] KENWRIGHT D. N., HENZE C., LEVIT C.: Feature extraction of separation and attachment lines. *IEEE Transactions on Visualization and Computer Graphics 5*, 2 (April 1999), 135–144.

[KHS*18] KERN M., HEWSON T., SADLO F., WESTERMANN R., RAUTENHAUS M.: Robust detection and visualization of jet-stream core lines in atmospheric flow. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (2018), 893–902.

[KHS*19] KERN M., HEWSON T., SCHÄFLER A., WESTERMANN R., RAUTENHAUS M.: Interactive 3d visual analysis of atmospheric fronts. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 1080–1090.

[KN20] KERN M., NEUHAUSER C.: chrismile/PixelSyncOIT: CPU and GPU algorithms for large space-filling 3D Line Sets with Transparency, Feb. 2020. URL: `https://doi.org/10.5281/zenodo.3634907`.

[KNM*20] KERN M., NEUHAUSER C., MAACK T., HAN M., USHER W., WESTERMANN R.: A comparison of rendering techniques for 3d line sets with transparency. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1.

[KRW18] KANZLER M., RAUTENHAUS M., WESTERMANN R.: A voxel-based rendering pipeline for large 3d line sets. *IEEE Transactions on Visualization and Computer Graphics* (2018), 1–1.

[KTB*18] KUMPF A., TOST B., BAUMGART M., RIEMER M., WESTERMANN R., RAUTENHAUS M.: Visualizing confidence in cluster-based ensemble weather forecast analyses. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan 2018), 109–119.

[KTW07] KINDLMANN G., TRICOCHE X., WESTIN C.-F.: Delineating white matter structure in diffusion tensor mri with anisotropy creases. *Medical image analysis 11* (11 2007), 492–502.

[KW19a] KERN M., WESTERMANN R.: Clustering Ensembles of 3D Jet-Stream Core Lines. In *Vision, Modeling and Visualization* (2019), Schulz H.-J., Teschner M., Wimmer M., (Eds.), The Eurographics Association.

[KW19b] KINGMA D. P., WELLING M.: An introduction to variational autoencoders. *CoRR abs/1906.02691* (2019).

[KWD06] KOCH P., WERNLI H., DAVIES H. C.: An event-based jet-stream climatology and typology. *Int. J. Climatol. 26*, 3 (2006), 283–301.

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, July 1987), vol. 21 of *SIGGRAPH '87*, ACM, pp. 163–169.

[LDS90] LEVY Y., DEGANI D., SEGINER A.: Graphical visualization of vortical flows by means of helicity. *AIAA Journal 28*, 8 (1990), 1347–1352.

[LGP14] LAWONN K., GÜNTHER T., PREIM B.: Coherent View-Dependent Streamlines for Understanding Blood Flow. In *EuroVis - Short Papers* (2014), The Eurographics Association.

[LGS*09] LAUTERBACH C., GARLAND M., SENGUPTA S., LUEBKE D., MANOCHA D.: Fast bvh construction on gpus. *Computer Graphics Forum 28*, 2 (2009), 375–384.

[Lin98] LINDEBERG T.: Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision 30*, 2 (1998), 117–156.

[LK10] LAINE S., KARRAS T.: Efficient sparse voxel octrees. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games* (2010), pp. 55–63.

[LLBP12] LIU S., LEVINE J. A., BREMER P.-T., PASCUCCI V.: Gaussian mixture model based volume visualization. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV)* (2012), IEEE, pp. 73–77.

[Llo82] LLOYD S. P.: Least squares quantization in pcm. *IEEE Trans. Inf. Theory 28* (1982), 129–136.

[LMdUB*15] LOPEZ-MOLINA C., DE ULZURRUN G. V.-D., BAETENS J., DEN BULCKE J. V., BAETS B. D.: Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing 116* (2015), 55 – 67.

[LP08] LEUTBECHER M., PALMER T.: Ensemble forecasting. *J. Comput. Phys. 227*, 7 (Mar. 2008), 3515–3539.

[LPK05] LOVE A. L., PANG A., KAO D. L.: Visualizing spatial multivalue data. *IEEE Computer Graphics and Applications 25*, 3 (2005), 69–79.

[LSW12] LIMBACH S., SCHÖMER E., WERNLI H.: Detection, tracking and event localization of jet stream features in 4-D atmospheric data. *Geosci. Model Dev. 5*, 2 (Apr. 2012), 457–470.

[Mac67] MACQUEEN J.: Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics* (Berkeley, Calif., 1967), University of California Press, pp. 281–297.

[Maj00] MAJER P.: *A statistical approach to feature detection and scale selection in images*. PhD thesis, Niedersächsische Staats-und Universitätsbibliothek, 2000.

[Mar06] MARTIN J. E.: *Mid-Latitude Atmospheric Dynamics: A First Course*, 1 ed. Wiley, June 2006.

[Mar14] MARTIUS O.: A Lagrangian analysis of the northern hemisphere subtropical jet. *J. Atmos. Sci. 71*, 7 (Mar. 2014), 2354–2369.

[Mas91] MASS C. F.: Synoptic frontal analysis: Time for a reassessment? *Bulletin of the American Meteorological Society 72*, 3 (Mar. 1991), 348–363.

[MB13] MCGUIRE M., BAVOIL L.: Weighted blended order-independent transparency. *Journal of Computer Graphics Techniques (JCGT) 2*, 2 (December 2013), 122–141.

[MCTB11] MAULE M., COMBA J. A. L. D., TORCHELSEN R. P., BASTOS R.: Technical section: A survey of raster-based transparency techniques. *Comput. Graph. 35*, 6 (Dec. 2011), 1023–1034.

[MCTB13] MAULE M., COMBA J. A., TORCHELSEN R., BASTOS R.: Hybrid transparency. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2013), I3D '13, ACM, pp. 103–118.

[MH08] MAATEN L. V. D., HINTON G.: Visualizing data using t-sne. *Journal of machine learning research 9*, Nov (2008), 2579–2605.

[MHD*11] MANNEY G. L., HEGGLIN M. I., DAFFER W. H., SANTEE M. L., RAY E. A., PAWSON S., SCHWARTZ M. J., BOONE C. D., FROIDEVAUX L., LIVESEY N. J., READ W. G., WALKER K. A.: Jet characterization in the upper troposphere/lower stratosphere (UTLS): applications to climatology and transport studies. *Atmospheric Chemistry and Physics 11*, 12 (June 2011), 6115–6137.

[MHD*14] MANNEY G. L., HEGGLIN M. I., DAFFER W. H., SCHWARTZ M. J., SANTEE M. L., PAWSON S.: Climatology of upper Tropospheric–Lower stratospheric (UTLS) jets and tropopauses in MERRA. *J. Climate 27*, 9 (Jan. 2014), 3248–3271.

[MK20] MAACK T., KERN M.: Rtx ray tracing of transparent 3d line sets, Feb. 2020. URL: `https://doi.org/10.5281/zenodo.3637621`.

[MKKP18] MÜNSTERMANN C., KRUMPEN S., KLEIN R., PETERS C.: Moment-based order-independent transparency. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 1*, 1 (May 2018), 7:1–7:20.

[MM17] MCGUIRE M., MARA M.: Phenomenological transparency. *IEEE Transactions on Visualization and Computer Graphics 23*, 5 (May 2017), 1465–1478.

[MMP*17] MOLNOS S., MAMDOUH T., PETRI S., NOCKE T., WEINKAUF T., COUMOU D.: A network-based detection scheme for the jet stream core. *Earth System Dynamics 8*, 1 (Feb. 2017), 75–89.

[MRH*05] MACEACHREN A., ROBINSON A., HOPPER S., GARDNER S., MURRAY R., GAHEGAN M., HETZLER E.: Visualizing geospatial information uncertainty: What we know and what we need to know. *Cartography and Geographic Information Science 32* (07 2005), 139–160.

[MSE13] MACHADO G. M., SADLO F., ERTL T.: Local Extraction of Bifurcation Lines. In *Vision, Modeling & Visualization* (2013), Bronstein M., Favre J., Hormann K., (Eds.), The Eurographics Association.

[MVv05] MOBERTS B., VILANOVA A., VAN WIJK J. J.: Evaluation of fiber clustering methods for diffusion tensor imaging. In *VIS 05. IEEE Visualization, 2005.* (Oct 2005), pp. 65–72.

[MWK14] MIRZARGAR M., WHITAKER R. T., KIRBY R. M.: Curve boxplot: Generalization of boxplot for ensembles of curves. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2654–2663.

[NSB04] NOCKE T., SCHUMANN H., BÖHM U.: Methods for the visualization of clustered climate data. *Computational Statistics 19* (02 2004), 75–94.

[NVI18] NVIDIA: Nvidia rtx ray tracing developer documentation. `developer.nvidia.com/rtx/raytracing`, 2018. [Online; accessed 24-May-2020].

[OLK*14] OELTZE S., LEHMANN D. J., KUHN A., JANIGA G., THEISEL H., PREIM B.: Blood flow clustering and applications invirtual stenting of intracranial aneurysms. *IEEE Transactions on Visualization and Computer Graphics 20*, 5 (May 2014), 686–701.

[ORT18a] OSTER T., RÖSSL C., THEISEL H.: Core lines in 3d second-order tensor fields. *Computer Graphics Forum 37*, 3 (2018), 327–337.

[ORT18b] OSTER T., RÖSSL C., THEISEL H.: The Parallel Eigenvectors Operator. In *Vision, Modeling and Visualization* (2018), Beck F., Dachsbacher C., Sadlo F., (Eds.), The Eurographics Association.

[PBC*94] PIZER S. M., BURBECK C. A., COGGINS J. M., FRITSCH D. S., MORSE B. S.: Object shape before boundary shape: Scale-space medial axes. *Journal of Mathematical Imaging and Vision 4* (1994), 303–313.

[PBD*10] PARKER S. G., BIGLER J., DIETRICH A., FRIEDRICH H., HOBEROCK J., LUEBKE D., MCALLISTER D., MCGUIRE M., MORLEY K., ROBISON A., STICH M.: Optix: A general purpose ray tracing engine. *ACM Transactions on Graphics* (August 2010).

[PD84] PORTER T., DUFF T.: Compositing digital images. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1984), SIGGRAPH '84, Association for Computing Machinery, p. 253–259.

[Pet36] PETTERSSEN S.: *Contribution to the theory of frontogenesis*. Geofysiske Publikasjoner. Cammermeyer, 1936.

[PGM*14] PINTO J. G., GÓMARA I. N., MASATO G., DACRE H. F., WOOLLINGS T., CABALLERO R.: Large-scale dynamics associated with clustering of extratropical cyclones affecting western europe. *Journal of Geophysical Research: Atmospheres 119*, 24 (Dec. 2014), 13,704–13,719.

[PH11] PÖTHKOW K., HEGE H.: Positional uncertainty of isocontours: Condition analysis and probabilistic measures. *IEEE Transactions on Visualization and Computer Graphics 17*, 10 (2011), 1393–1406.

[PH13] PÖTHKOW K., HEGE H.-C.: Nonparametric models for uncertainty visualization. *Computer Graphics Forum 32*, 3 (2013), 131–140.

[PKRJ10] POTTER K., KNISS J., RIESENFELD R., JOHNSON C. R.: Visualizing summary statistics and uncertainty. In *Proceedings of the 12th Eurographics / IEEE - VGTC Conference on Visualization* (Chichester, GBR, 2010), EuroVis'10, The Eurographs Association & John Wiley & Sons, Ltd., p. 823–832.

[PN69] PALMEN E., NEWTON C. W.: *Atmospheric circulation systems: their structure and physical interpretation (International Geophysics)*. Academic Press, Aug. 1969.

[POGR*13] PENA-ORTIZ C., GALLEGO D., RIBERA P., ORDONEZ P., ALVAREZ-CASTRO M. D.: Observed trends in the global jet stream characteristics during the second half of the 20th century. *J. Geophys. Res. Atmos. 118*, 7 (Apr. 2013), 2702–2713.

[PR99] PEIKERT R., ROTH M.: The "parallel vectors" operator-a vector field visualization primitive. In *Visualization '99. Proceedings* (Oct 1999), pp. 263–532.

[PRJ12] POTTER K., ROSEN P., JOHNSON C. R.: From quantification to visualization: A taxonomy of uncertainty visualization approaches. In *Uncertainty Quantification in Scientific Computing*. 2012, pp. 226–249.

[PRW11] PFAFFELMOSER T., REITINGER M., WESTERMANN R.: Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields. *Computer Graphics Forum 30*, 3 (2011), 951–960.

[PS08] PEIKERT R., SADLO F.: Height ridge computation and filtering for visualization. In *2008 IEEE Pacific Visualization Symposium* (March 2008), pp. 119–126.

[PVH*03] POST F. H., VROLIJK B., HAUSER H., LARAMEE R. S., DOLEISCH H.: The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum 22*, 4 (2003), 775–792.

[Rau20] RAUTENHAUS M.: Met.3d - interactive 3d visualization of meteorological (ensemble) simulations. `https://met3d.wavestoweather.de/`, July 2020. [accessed 06-July-2020].

[RC65] RENARD R. J., CLARKE L. C.: Experiments in numerical objective frontal analysis 1. *Monthly Weather Review 93*, 9 (Sept. 1965), 547–556.

[Rei63] REITER E. R.: *Jet-stream meteorology*, 1st ed. University of Chicago Press, 1963.

[RGK*13] REH A., GUSENBAUER C., KASTNER J., GRÖLLER M. E., HEINZL C.: Mobjects–a novel method for the visualization and interactive exploration of defects in industrial xct data. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2906–2915.

[RGSW15] RAUTENHAUS M., GRAMS C. M., SCHÄFLER A., WESTERMANN R.: Three-dimensional visualization of ensemble weather forecasts – part 2: Forecasting warm conveyor belt situations for aircraft-based field campaigns. *Geoscientific Model Development 8*, 7 (2015), 2355–2377.

[Rik15] RIKUS L.: A simple climatology of westerly jet streams in global reanalysis datasets part 1: mid-latitude upper tropospheric jets. 1–26.

[RKSW15] RAUTENHAUS M., KERN M., SCHÄFLER A., WESTERMANN R.: 3-d visualization of ensemble weather forecasts – Part 1: The visualization tool Met.3d (version 1.0). *Geosci. Model Dev. Discussions 8*, 2 (Feb. 2015), 2101–2160.

[Rot00] ROTH M.: *Automatic Extraction of Vortex Core Lines and Other Line-Type Features for Scientific Visualization*. PhD thesis, Swiss Federal Institute of Technology, ETH Zürich, 2000.

[RT12] RÖSSL C., THEISEL H.: Streamline embedding for 3d vector field exploration. *IEEE Transactions on Visualization and Computer Graphics 18*, 3 (March 2012), 407–420.

[RTS*94] RITCHIE H., TEMPERTON C., SIMMONS A., HORTAL M., DAVIES T., DENT D., HAMRUD M.: Implementation of the semi-lagrangian method in a high resolution version of the ecmwf forecast model. 68. URL: `https://www.ecmwf.int/node/11939`.

[Sah09] SAHNER J.: *Extraction of Vortex Structures in 3D Flow Fields*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, 2009.

[SB81] SIMMONS A. J., BURRIDGE D. M.: An energy and angular-momentum conserving vertical finite-difference scheme and hybrid vertical coordinates. *Monthly Weather Review 109*, 4 (1981), 758–766.

[SB15] SCHULTZ D. M. D., BLUMEN W.: *SYNOPTIC METEOROLOGY | Fronts*. Elsevier, 2015, pp. 337–343.

[Sch20] SCHINDLER T. L.: The polar jet stream. `https://svs.gsfc.nasa.gov/3864`, July 2020. [accessed 06-July-2020].

[SD95] SANDERS F., DOSWELL C. A.: A case for detailed surface analysis. *Bulletin of the American Meteorological Society 76*, 4 (Apr. 1995), 505–521.

[SD05] STRONG C., DAVIS R. E.: The surface of maximum wind as an alternative to the isobaric surface for wind climatology. *Geophys. Res. Lett. 32*, 4 (Feb. 2005), L04813+.

[SD07] STRONG C., DAVIS R. E.: Winter jet stream trends over the northern hemisphere. *Q.J.R. Meteorol. Soc. 133*, 629 (Oct. 2007), 2109–2115.

[SD08] STRONG C., DAVIS R. E.: Variability in the position and strength of winter jet stream cores related to northern hemisphere teleconnections. *J. Climate 21*, 3 (Feb. 2008), 584–592.

[SG02] STREHL A., GHOSH J.: Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research 3*, Dec (2002), 583–617.

[SG07] SUESSMUTH J., GREINER G.: Ridge Based Curve and Surface Reconstruction. In *Geometry Processing* (2007), Belyaev A., Garland M., (Eds.), The Eurographics Association.

[SH95] SUJUDI D., HAIMES R.: *Identification of swirling flow in 3-D vector fields*. 1995.

[SK90] SHAPIRO M. A., KEYSER D.: Fronts, jet streams, and the tropopause. In *Extratropical Cyclones. The Erik Palmen Memorial Volume*, Newton C., Holopainen E. O., (Eds.). American Meteorological Society, Boston, MA, 1990, pp. 167–191.

[SK12] SAKAMOTO N., KOYAMADA K.: Stoachastic approach for integrated rendering of volumes and semi-transparent surfaces. In *Proceedings - 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, SCC 2012* (11 2012), pp. 176–185.

[SKA99] STRAWN R. C., KENWRIGHT D. N., AHMAD J.: Computer visualization of vortex wake systems. *AIAA Journal 37*, 4 (1999), 511–512.

[SKTB11] SIMMONDS I., KEAY K., TRISTRAM BYE J. A.: Identification and climatology of southern hemisphere mobile fronts in a modern reanalysis. *J. Climate 25*, 6 (Sept. 2011), 1945–1962.

[SLM05] SHADDEN S. C., LEKIEN F., MARSDEN J. E.: Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena 212*, 3-4 (2005), 271 – 304.

[SLY15] SOHN K., LEE H., YAN X.: Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28*, Cortes C., Lawrence N. D., Lee D. D., Sugiyama M., Garnett R., (Eds.). Curran Associates, Inc., 2015, pp. 3483–3491.

[SML11] SALVI M., MONTGOMERY J., LEFOHN A.: Adaptive transparency. In *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics* (New York, NY, USA, 2011), HPG '11, ACM, pp. 119–126.

[SNM*16] SCHEMM S., NISI L., MARTINOV A., LEUENBERGER D., MARTIUS O.: On the link between cold fronts and hail in switzerland. *Atmos. Sci. Lett. 17*, 5 (May 2016), 315–325.

[SP07] SADLO F., PEIKERT R.: Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (Nov 2007), 1456–1463.

[SP09] SADLO F., PEIKERT R.: *Visualizing Lagrangian Coherent Structures and Comparison to Vector Field Topology*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 15–29.

[SRS15] SCHEMM S., RUDEVA I., SIMMONDS I.: Extratropical fronts in the lower troposphere–global perspectives obtained from two automated methods. *Quarterly Journal of the Royal Meteorological Society 141*, 690 (2015), 1686–1698.

[SS56] SAWYER J. S., SUTTON O. G.: The vertical circulation at meteorological fronts and its relation to frontogenesis. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences 234*, 1198 (1956), 346–362.

[SS83] SIMMONS A. J., STRÜFING R.: Numerical forecasts of stratospheric warming events using a model with a hybrid vertical coordinate. *Quarterly Journal of the Royal Meteorological Society 109*, 459 (1983), 81–111.

[SSL17] SPENSBERGER C., SPENGER T., LI C.: Upper tropospheric jet axis detection and application to the boreal winter 2013/14. *Mon. Wea. Rev.* (Mar. 2017).

[Ste56] STEINHAUS H.: Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci 1*, 804 (1956), 801.

[STS10] SCHULTZ T., THEISEL H., SEIDEL H.: Crease surfaces: From theory to extraction and application to diffusion tensor mri. *IEEE Transactions on Visualization and Computer Graphics 16*, 1 (2010), 109–119.

[Stu17] STULL R.: *Practical Meteorology: An Algebra-based Survey of Atmospheric Science*, v1.02b ed. Creative Commons License, Copyright 2017, 2018 by Roland Stull, University of British Columbia, Vancouver, BC, 2017.

[SV70] SUNDQVIST H., VERONIS G.: A simple finite-difference grid with non-constant intervals. *Tellus 22*, 1 (1970), 26–31.

[SV11] SCHULTZ D. M., VAUGHAN G.: Occluded fronts and the occlusion process: A fresh look at conventional wisdom. *Bulletin of the American Meteorological Society 92*, 4 (Apr. 2011), 443–466.

[SV14] SALVI M., VAIDYANATHAN K.: Multi-layer alpha blending. In *Proceedings of the 18th Meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2014), I3D '14, ACM, pp. 151–158.

[SWH05] SAHNER J., WEINKAUF T., HEGE H.-C.: Galilean Invariant Extraction and Iconic Representation of Vortex Core Lines. In *EUROVIS 2005: Eurographics / IEEE VGTC Symposium on Visualization* (2005), Brodlie K., Duke D., Joy K., (Eds.), The Eurographics Association.

[SWTH07] SAHNER J., WEINKAUF T., TEUBER N., HEGE H. C.: Vortex and strain skeletons in Eulerian and Lagrangian frames. *IEEE Transactions on Visualization and Computer Graphics 13*, 5 (Sept 2007), 980–990.

[Tei99] TEIXEIRA J.: The impact of increased boundary layer vertical resolution on the ecmwf forecast system. 55.

[TF98] THOMAS G. B., FINNEY R. L.: *Calculus and Analytic Geometry*, 9 ed. Addison Wesley, 1998, ch. 12.8, pp. 970 – 979.

[TN14] THOMAS D., NATARAJAN V.: Multiscale symmetry detection in scalar fields by clustering contours. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2427–2436.

[TS19a] THOMAS C. M., SCHULTZ D. M.: Global climatologies of fronts, airmass boundaries, and airstream boundaries: Why the definition of "front" matters. *Monthly Weather Review 147*, 2 (2019), 691–717.

[TS19b] THOMAS C. M., SCHULTZ D. M.: What are the best thermodynamic quantity and function to define a front in gridded model output? *Bulletin of the American Meteorological Society 100*, 5 (2019), 873–895.

[UH04] UNTCH A., HORTAL M.: A finite-element scheme for the vertical discretization of the semi-lagrangian version of the ecmwf forecast model. *Quarterly Journal of the Royal Meteorological Society 130*, 599 (2004), 1505–1530.

[VVP20] VASILAKIS A. A., VARDIS K., PAPAIOANNOU G.: A Survey of Multifragment Rendering. *Computer Graphics Forum* (2020).

[WH06] WALLACE J. M., HOBBS P. V.: *Atmospheric Science: An Introductory Survey*, 2 ed. Academic Press, Feb. 2006.

[WIK*06] WALD I., IZE T., KENSLER A., KNOLL A., PARKER S. G.: Ray tracing animated scenes using coherent grid traversal. *ACM Transactions on Graphics* (2006), 485–493.

[Wil11] WILKS D. S.: *Statistical Methods in the Atmospheric Sciences*, 3rd ed. Academic Press, June 2011.

[WJA*17] WALD I., JOHNSON G. P., AMSTUTZ J., BROWNLEE C., KNOLL A., JEFFERS J., GÜNTHER J., NAVRÁTIL P.: OSPRay – A CPU Ray Tracing Framework for Scientific Visualization. *IEEE Transactions on Visualization and Computer Graphics* (2017).

[WKJ*15] WALD I., KNOLL A., JOHNSON G. P., USHER W., PASCUCCI V., PAPKA M. E.: Cpu ray tracing large particle data with balanced p-k-d trees. In *IEEE SciVis* (2015).

[WLW*17] WANG K.-C., LU K., WEI T.-H., SHAREEF N., SHEN H.-W.: Statistical visualization and analysis of large data using a value-based spatial distribution. In *2017 IEEE Pacific Visualization Symposium (PacificVis)* (2017), IEEE, pp. 161–170.

[WMK13] WHITAKER R. T., MIRZARGAR M., KIRBY R. M.: Contour boxplots: A method for characterizing uncertainty in feature sets from simulation ensembles. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec 2013), 2713–2722.

[WMS06] WOOP S., MARMITT G., SLUSALLEK P.: B-KD Trees for hardware accelerated ray tracing of dynamic scenes. In *GH '06: Proceedings of the 21st ACM SIGGRAPH/EUROGRAPHICS symposium on Graphics hardware* (New York, NY, USA, 2006), ACM, pp. 67–77.

[Wor92] WORLD METEOROLOGICAL ORGANIZATION: *International Meteorological Vocabulary*, 2nd ed. WMO. WMO, 1992.

[WWB*14] WALD I., WOOP S., BENTHIN C., JOHNSON G. S., ERNST M.: Embree: A kernel framework for efficient cpu ray tracing. *ACM Trans. Graph. 33*, 4 (July 2014), 143:1–143:8.

[Wym16] WYMAN C.: Stochastic layered alpha blending. In *ACM SIGGRAPH 2016 Talks* (New York, NY, USA, 2016), SIGGRAPH '16, ACM, pp. 37:1–37:2.

[YHGT10] YANG J. C., HENSLEY J., GRÜN H., THIBIEROZ N.: Real-time concurrent linked list construction on the gpu. In *Proceedings of the 21st Eurographics Conference on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2010), EGSR'10, Eurographics Association, pp. 1297–1304.

[ZHQL16] ZHANG H., HOU Y., QU D., LIU Q.: Correlation visualization of time-varying patterns for multivariable data. *IEEE Access 4* (2016), 4669–4677.

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 586–595.

7

Accepted Versions of Published Papers

The accepted versions of all four publications connected to this thesis are appended in this chapter, as they are an essential part of this publication-based thesis. While the fundamentals of our methods were explained in Chapter 3, these publications detail all the developed methodologies and analyses mentioned in Sec. 1.4. Note that the original and camera-ready versions of the papers are subject to the corresponding journals they have been published in. In particular, “Robust Detection and Visualization of Jet-Stream Core Lines in Atmospheric Flow” 7.1, “Interactive 3D Visual Analysis of Atmospheric Fronts” 7.2, and “A Comparison of Rendering Techniques for 3D Line Sets with Transparency” 7.4 are governed by the copyright of IEEE Transactions on Visualization and Computer Graphics, and “Clustering Ensembles of 3D Jet-Stream Core Lines” 7.3 is liable to the copyright of the Eurographics Association.

Robust Detection and Visualization of Jet-stream Core Lines in Atmospheric Flow

Michael Kern, Tim Hewson, Filip Sadlo, Rüdiger Westermann, and Marc Rautenhaus

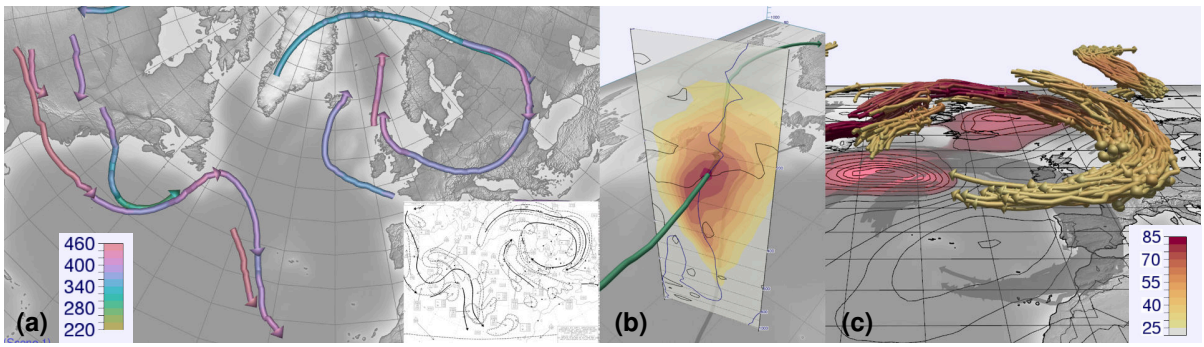


Fig. 1. **(a)** Jet-stream core lines extracted from a 3D wind field with our method, colored by flight level (hft). To the best of our knowledge, these features are still created manually by domain experts in operational weather forecasting and provided via 2D maps, as shown in the inset. **(b)** Normal plane perpendicular to the wind vector, used by our method as a local coordinate frame to extract jet cores. Color shows wind speed (ms^{-1}). **(c)** 3D "spaghetti plot" visualization of an ensemble of jet-stream cores (color shows wind speed as in (b)) enables an improved analysis of their spatial structure, forecast uncertainty, and relation to further atmospheric features including, e.g., mean-sea level pressure (MSLP; black contour lines, red regions indicate where pressure is below 1000 hPa).

Abstract— Jet-streams, their core lines and their role in atmospheric dynamics have been subject to considerable meteorological research since the first half of the twentieth century. Yet, until today no consistent automated feature detection approach has been proposed to identify jet-stream core lines from 3D wind fields. Such 3D core lines can facilitate meteorological analyses previously not possible. Although jet-stream cores can be manually analyzed by meteorologists in 2D as height ridges in the wind speed field, to the best of our knowledge no automated ridge detection approach has been applied to jet-stream core detection. In this work, we –a team of visualization scientists and meteorologists– propose a method that exploits directional information in the wind field to extract core lines in a robust and numerically less involved manner than traditional 3D ridge detection. For the first time, we apply the extracted 3D core lines to meteorological analysis, considering real-world case studies and demonstrating our method's benefits for weather forecasting and meteorological research.

Index Terms—Meteorology, weather forecast, jet-stream, feature detection

1 INTRODUCTION

The improvement of weather forecasts and climate change projections depends heavily on documenting and understanding complex three-dimensional structures in the atmosphere. A key component of those structures is the jet-stream. Jet-streams are regions of high wind speed, typically encountered near to the top of our principal weather systems, at altitudes of about 8-16km. As well as determining the general weather type –such as blocking and storm tracks– they also exert a strong dynamical influence on severe weather events, such as extreme windstorms [2]. Jet-streams are also related to clear-air turbulence (CAT), important for daily aviation operations. In this respect, jet-stream core lines –lines of maximum wind speed– are operationally depicted as fundamental atmospheric structures on significant weather (SIGWX) charts used by pilots [53]. Even though a concise definition of jet-streams has long been provided by the World Meteorological Organization (WMO) [54], jet-stream core lines in operational weather forecast settings are, similar to other atmospheric features including fronts, still identified manually. This process is time-consuming, requires expertise, and does not allow for a full analysis of the 3D geometry of the jet-stream core lines and how this relates to features of the surrounding atmosphere.

In this article, we approach the still unsolved question of how jet-stream core lines can be objectively identified from three-dimensional numerical weather prediction (NWP) data in an automated, robust manner, and visualized in a way that can benefit a subsequent in-depth meteorological analysis of the model atmosphere.

1.1 Problem Description

A jet-stream is officially defined by the WMO as a *"flat tubular, quasi-horizontal, current of air generally near the tropopause, whose axis is along a line of maximum speed and which is characterized by great speeds and strong vertical and horizontal wind shears"* [54]. Its core line is defined as the *"line along which the wind speeds are maximum both in the vertical and in the horizontal"* [54]. Yet in spite of the pivotal role that jet-streams and their core lines play, as a driving force in atmospheric dynamics, we are unaware of any objective three-dimensional identification methodology built on this definition that can be applied in a practical way to meteorological analysis.

- *Michael Kern, Rüdiger Westermann and Marc Rautenhaus are with the Computer Graphics & Visualization Group, Technische Universität München, Garching, Germany. E-mail: {michi.kern, westermann, marc.rautenhaus}@tum.de*
- *Tim Hewson is with the European Centre for Medium-Range Weather Forecasts, Reading, UK. E-mail: tim.hewson@ecmwf.int*
- *Filip Sadlo is with the Visual Computing Group, Heidelberg University, Heidelberg, Germany. E-mail: sadlo@uni-heidelberg.de*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

Such automated detection and visualization of jet-stream core lines is much needed because of the following:

- Using a single 3D visualization, with a fly-through capability, it would provide a greatly increased capacity to understand the key relationships and dependencies between multiple atmospheric processes and the jet.
- It would provide a new way of understanding and critically assessing the uncertainty inherent in ensemble forecasts, and notably occasions when there is insufficient spread, that results in 'forecast busts' around the world. Jet-stream behavior is implicated in studies of these forecast failures.
- Automated core-line extraction is much needed to assist with the generation of SIGWX charts for aviation; currently the manual procedure relies heavily on representation of a single jet level.

A specific unsolved issue in meteorology that can benefit from automated jet-stream core detection and for which we discuss the application of our method is the modulation of the jet-stream by other atmospheric processes and the influence of such modulation on the predictability of downstream weather. Specific examples for such processes include extreme convection over the United States [41], and Warm Conveyor Belts (WCBs, airstreams in extratropical cyclones that lift warm and moist air from near the surface to the upper troposphere; e.g., [7]) over the north Atlantic, which can both modulate the jet-stream and cause large uncertainties in predictions for European weather.

1.2 Contribution

The jet-stream core line definition provided by the WMO relates to the definition of three-dimensional height ridges encountered in different contexts in flow visualization [48], [46], [43]. Yet, while a number of studies in meteorology have proposed methods to detect –mainly two-dimensional– jet-stream features (cf. Sect. 2), to the best of our knowledge no ridge detection method has been applied to the automated extraction of 3D jet-stream core lines.

In this work, we propose a robust 3D detection method for jet-stream core lines in NWP data that directly reflects the official WMO definition and that relates to height ridge computation. By exploiting the fact that jet-stream core lines are at very narrow angles to the wind direction and the fact that the vertical wind component is negligible compared to the horizontal in large-scale atmospheric flow, we can determine the core lines as wind speed maxima in vertical planes perpendicular to the horizontal wind direction. Thus, in contrast to "classical" ridge detection algorithms, which determine maxima in vertical planes spanned by the eigenvectors of the 3D Hessian matrix, our approach does not suffer from spurious variations due to noise, and it can be enforced explicitly that the planes are consistently oriented. As a consequence, the jet cores extracted by our method are more robust, i.e., less disjointed and cluttered. Furthermore, our method does not require excessive blurring of the underlying field and can work on the original data. Fig. 2 demonstrates the differences in the extracted jet cores using both methods.

We integrate our new detection method into the interactive 3D meteorological ensemble visualization tool "Met.3D" [38], facilitating combination of the detected features with further atmospheric visualizations, and propose a number of visualizations of the core lines that help with the analysis of NWP data to investigate the motivating meteorological research questions. In particular, we visualize

- 3D jet-stream core lines in combination with 3D depictions of atmospheric processes including clouds,
- 3D spaghetti plots of jet-stream core lines extracted from ensemble weather forecasts to depict forecast uncertainty with respect to the jets,
- the relation of the core lines to local streamlines and surrounding atmospheric conditions (such as cloud water content or surface pressure),
- an automated SIGWX jet-stream product.

We apply the proposed techniques to analysis of NWP data from the European Centre for Medium-Range Weather Forecasts (ECMWF), demonstrating insight that can be gained regarding the posed meteorological research questions.

2 Related Work

Our work relates to research connected to jet-streams and their detection in the atmospheric sciences, and to the extraction of line features in flow visualization. Concerning the latter, ridge detection is of particular importance.

2.1 Jet-stream Detection

Major references for jet-streams and their characteristics, including previous research and a description of manual analysis methods, date back to the books by Reiter [39] and Palmen and Newton [34]. Here, the jet-stream axis on a 2D chart was introduced as the "line of maximum wind speed". The "layer of maximum wind" (LMW) was introduced as a method to analyze the 3D jet-stream axis (=core). The LMW was operationally used for weather forecasting in the U.S. (cf. [49]) and is also utilized today in operational production of SIGWX charts (Sect. 5.1).

Automatic extraction of jet-streams has been mainly investigated in the past 15 years, primarily to compile climatologies. Some authors try to identify core lines, others simply use speed thresholds. Often the classification of a model grid point as belonging to a jet-stream has been considered sufficient. Also, studies commonly look first for the maximum wind in the vertical, as in the LMW concept.

Koch et al. [20] counted events for each horizontal grid point where the average wind speed between 100 and 400 hPa exceeds a threshold, whilst Archer and Caldeira [1] used mass-flux weighted averages to determine jet-stream events per horizontal grid point. Similar height-dependent thresholding on wind speed was used by Limbach et al. [23] and Martius [31]. Meanwhile Schiemann et al. [45], Pena-Ortiz et al. [36] and Barton and Ellis [3] compute jet-stream core events in various ways, but with the common assumption that jets must propagate west to east. Also for the purpose of a climatology, Gallego et al. [12] defined a criterion based on a geostrophic streamline of maximum average velocity to get jet-like streamlines circumventing the southern hemisphere.

Strong and Davis [49–51] used a notion similar to the LMW. The core is detected on their LMW equivalent by computing wind speed maxima via finite differencing in the y-direction only. However line geometry was not used. Manney et al. [29,30] constructed a climatology of jet-stream cores by cataloging wind speed maxima on longitudinal cross-sections (no detected line geometry). More recently, Molnos et al. [32] introduced a network-based scheme using shortest paths to detect jet-stream core as a continuous, globe-circumventing line. The method is calibrated using an image-based jet analysis technique by Rikus [40], and used to compute frequencies of jet-core occurrence. Recently, Spensberger et al [47] adapted a 2D criterion by Berry et al. [4] to 2D wind fields on a "dynamical tropopause", an isosurface of 2 PV (potential vorticity) units.

Most of the above methods are not Galilean invariant. They mostly depend on assuming a priori that the jet-stream exhibits certain characteristics; for example that it is westerly. In contrast, the methodology in this paper, which follows on principally from techniques described in Berry et al (2007), is Galilean invariant, and will identify jet-streams equally in all directions and at all atmospheric levels.

2.2 Line Features in Fluid Dynamics / Flow Visualization

Feature extraction is an important tool and an active branch of research in flow visualization. A particular reason for its importance is the comparably high dimensionality of vector fields—they add the difficulty of three-dimensional range visualization to the already difficult representation of three spatial and one temporal dimensions of their domain. The fact that flow fields, which are a primary source for vector fields, tend to exhibit turbulence and chaotic advection further complicates

their analysis, necessitating effective visualization approaches with as few parameters as possible.

Line-type features are particularly useful for this purpose due to several reasons: they are able to give a concise picture of flow fields, do not suffer from occlusion, and there are many variants that are conceptually free of parameters. A widely used and very successful line feature in flows is the vortex core lines. As no general definition of a vortex has been found so far, there exist also several competing definitions for vortex core lines. A vortex core line can be seen as the (possibly bent) axis of vortical fluid motion, representing the set of points around which massless particles in the flow swirl. A widely used definition for vortex core lines is that by Sujudi and Haimes [52]. Peikert and Roth [35] presented a mathematical framework, the parallel vectors operator, in which Sujudi and Haimes' definition can be formulated as the loci where the real eigenvector of the Jacobian of the vector field is parallel to the flow vector, with the additional requirement of the other two eigenvalues being complex. An other prominent vortex core line definition is that by Levy et al. [22], which, in this framework, requires the vorticity vector to be parallel to velocity. The parallel vectors operator, representing line-type features in general as the locations where two (derived) vector fields are parallel or anti-parallel, can also be used for defining separation lines and attachment lines [18], and the related bifurcation lines [27, 42].

In scalar fields, a prominent line-type feature is that of ridges and valleys. Ridges and valleys can be interpreted as generalized local extrema. Local extrema in n-dimensional scalar fields can be defined as points which exhibit a respective extremum in n orthogonal profile sections cutting through that point. If we relax this condition by one dimension, i.e., taking the set of points at which only $n-1$ orthogonal profile sections exhibit a local maximum (minimum), we obtain ridge (valley) lines. In early work in the context of surface topography, ridges and valleys were first mathematically described and idealized [5, 8]. Ridge extraction (we imply here also valleys, since valleys can be obtained by extracting ridges from the negated field) is widely applied in image analysis and computer vision, with a digital image treated as a scalar field. Ridges serve as characteristic structures in these domains, complementary to edges, within the boundaries of objects [14, 24, 28]. For ridge surfaces, i.e., where only one profile section has to exhibit a local maximum, Furst and Pizer [11] presented an approach for their extraction from 3D scalar fields, by tracing them through the volume. Ridge surfaces were applied to volumetric data by Kindlmann et al. [19] to visualize diffusion tensor MRI data. In the context of flow visualization, ridges have become a common tool to indicate and extract, e.g., vortex core lines [48], flow separation [46], by Sahner et al. [44] to visualize vorticity and strain, and by Sadlo et al. [43] to display separating regions of different flow behavior in unsteady vector fields. Peikert et al. [35] described an efficient and alternative way to compute and filter height ridges with an implicit formulation with respect to the eigenvectors of the Hessian.

2.3 Ridge Detection

A widely used formulation for ridges is that of height ridges by means of the gradient and the Hessian of a scalar field. Let $s(\vec{x})$ be the 3D scalar field where we want to extract ridge lines from. Height ridge lines according to Eberly [10] are defined by the parallel vectors operator as the loci where

$$\nabla s(\vec{x}) \parallel \vec{\varepsilon}_3\,, \tag{1}$$

i.e., where the major eigenvector $\vec{\varepsilon}_3$ of the Hessian $\nabla\nabla s(\vec{x})$ and the gradient of the scalar field are (anti-)parallel, with the additional condition that the two other eigenvalues need to be negative:

$$\lambda_1 < 0, \quad \text{and} \quad \lambda_2 < 0\,, \tag{2}$$

with $\lambda_1 \leq \lambda_2 \leq \lambda_3$, and $\vec{\varepsilon}_i$ being the eigenvector for eigenvalue λ_i.

In this formulation, $\vec{\varepsilon}_3$ is considered parallel to the ridge line tangent, and $\vec{\varepsilon}_1$ and $\vec{\varepsilon}_2$ perpendicular to it (note that the eigenvectors form an orthonormal system because the Hessian is symmetric). That is, we can consider $\vec{\varepsilon}_1$ and $\vec{\varepsilon}_2$ being normal vectors to the ridge line. One difficulty with this assumption is that it is typically never exactly met in practice. The angle α between the (anti-)parallel vectors $\nabla s(\vec{x})$ and $\vec{\varepsilon}_3$, and the feature tangent is, for well-defined ridges, commonly larger than 25 degrees. Peikert and Roth [35] use α as a quality criterion for extracted ridge lines, and filter their regions by imposing an upper threshold on this value, i.e., rejecting ridge line parts where this angle exceeds the threshold. It is not uncommon that the threshold needs to be set above 45 degrees to obtain useful results in practical data, which means that the eigenvectors may switch role regarding their "orthogonality" to the line feature tangent. An involved difficulty is that the gradient and the Hessian need to be estimated from the data, suffer from amplification of noise, and that obtaining smooth and at the same time interpolation-consistent derivatives is a difficult and often, as in the case of trilinear interpolation, impossible undertaking. All in all, although useful and widely employed, height ridges by means of eigenvectors of the Hessian tend to suffer with respect to robustness, false positives, and false negatives (disruptions)—often impeding effective visualization (compare Fig. 2). A main reason for these issues is the instability of the eigenvectors of the Hessian with respect to the aimed ridge line feature.

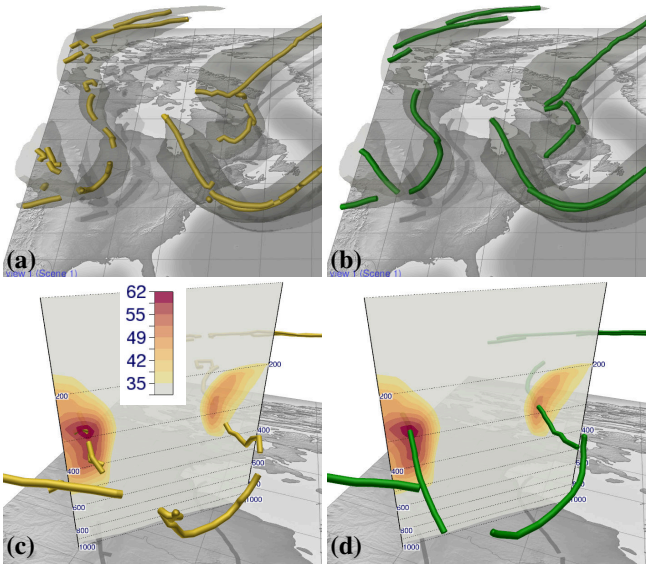


Fig. 2. 3D visualizations of maxima lines extracted from a 3D wind field. Shown are extracted lines via **(a)** the "classical" 3D ridge detection algorithm (yellow polylines) and **(b)** via our proposed method (green), restricted to regions with a wind speed of at least 40 ms^{-1} (enclosed by the dark-greyish isosurface). **(c)** Two maxima depicted by the contour lines of windspeed (color shows windspeed in ms^{-1}) in a vertical section are not properly detected by ridge detection. **(d)** Our method robustly extracts jet cores even where ridge detection fails.

In this work, we present, for the special case of scalar fields derived from vector fields, as is the case for jet-stream cores, an approach that avoids the computation of the Hessian except for masking purposes. Instead, we derive the required two normal directions across the ridge line from the vector field itself—and although these are, in general, also not strictly perpendicular to the resulting feature line, they are as continuous as the vector field—leading to superior results.

3 DETECTION METHOD

3.1 Definition and Assumptions

Based on the WMO definition, we define and interpret jet cores as follows: A jet core is a core of fluid having higher momentum than its surroundings. By examining the plane perpendicular to the momentum vector at every point in space, one can ascertain whether each point belongs to a jet core. In large-scale meteorology, momentum in the vertical plane is negligible. Therefore, jet cores lie in an approximately

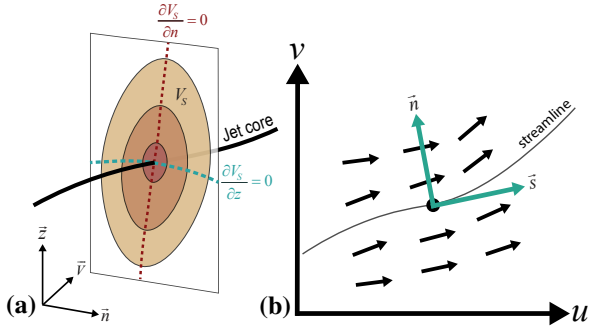


Fig. 3. **(a)** Sketch of the normal plane at a jet core intersection point. On the normal plane, colored contours of wind speed V_s (low speeds are yellow, high speeds red) are shown. The cyan (red) dotted line shows where the vertical (horizontal) derivative along the plane is zero. The jet core crosses the plane at the intersection of these zero lines. **(b)** Illustration of the local coordinate system in the 2D wind vector field used to construct the normal plane. $\vec{s}$ is the unit vector tangential to the current streamline, $\vec{n}$ is normal to it.

horizontal plane, and, thus, the normal planes are always vertical. If within its normal plane a point is a local maximum in resolved momentum then it belongs to a jet core. The connection of all maxima points form the jet core lines. In an NWP model, the momentum p per grid point is defined as $p = \rho\vec{v}$, where ρ denotes the density and $\vec{v}$ the velocity (or wind vector), both changing over time. Since we assume that the density ρ of the air is locally constant the resolved momentum at each grid point is maximal if the wind speed $|\vec{v}|$ is maximal within its normal plane. Fig. 3 illustrates the definition.

In our mathematical definition of jet cores we are employing the notion of following lines along which there is both zero horizontal and vertical shear. On a jet axis there is zero shear vorticity (as in [4]). And similarly in the vertical there is zero (speed) shear in parallel components. Additional masking is applied to further eliminate all wind speed minima and light wind conditions in general (also following [4]).

3.2 Data

We use NWP data from the ensemble prediction component (ENS) of the ECMWF Integrated Forecast System (IFS) (e.g., [21]), which comprises 50 perturbed members and an unperturbed control forecast. Data are available on a regular longitude–latitude grid in the horizontal, whilst in the vertical terrain-following hybrid sigma-pressure coordinates are used [37]. In this work, we use ECMWF ENS forecasts initialized at 00:00 UTC, 22 and 25 September 2016.

The output 3D wind vector $\vec{v} = (u, v, \omega)$, defined at grid points in longitude–latitude–pressure space, is composed of the horizontal wind-components u (eastward wind) and v (northward wind) defined in meters per second (ms^{-1}), and ω (vertical wind) defined in Pascals per second ($\mathrm{Pa\ s}^{-1}$). ω can be approximately converted to ms^{-1}, however, its order of magnitude is significantly smaller than that of the horizontal wind: $\mathcal{O}(0.05\ \mathrm{ms}^{-1})$ compared to $\mathcal{O}(50\ \mathrm{ms}^{-1})$. The vertical wind component thus has a negligible impact on both direction and magnitude of the 3D wind vector, we can hence ignore ω for jet core detection.

3.3 Mathematical Framework

Based on the official WMO definition and our understanding of jet cores in atmospheric flow, we elaborated a framework to mathematically define jet core lines in NWP model data. With these equations, we are able to robustly compute candidate points for jet cores and extract three-dimensional curve lines representing the jet-stream cores from a volumetric, gridded wind vector field.

Our method is built upon the notion of maxima ridges in 3D wind fields, i.e., we are aiming for the detection of local maxima points along a local coordinate frame (the normal plane) perpendicular to the current velocity vector. In principal, we have done so by following "classical" ridge extraction techniques. However, we found that computing eigenvectors in real-world 3D data sets tends to be unstable, resulting in jet cores that are commonly disjointed and cluttered (see Fig. 2 for a comparison). To avoid this, we introduce an alternative approach that, under the assumptions we make, can extract jet cores well aligned with those analyzed in operational weather forecasting, yet in a far more stable way.

Our method employs a fixed local frame-of-reference (Fig. 3) to locate our candidate points for the final jet core lines. As such it builds on the approach adopted by [15] to identify atmospheric front lines objectively, and [4] who identify 2D trough axes and 2D jet cores. Consider a local coordinate system $(\vec{s}, \vec{n})$ at a grid point X, where $\vec{s}$ is parallel to the local 2D (horizontal) wind vector $\vec{V} = (u, v)$, and $\vec{n}$ is normal to it. Then, the horizontal wind vector at each grid point can be split into two constituent components: $\vec{V} = (u, v) \equiv (V_s, V_n)$ where V_s is the wind speed along the vector $\vec{s}$ and V_n is the wind speed along $\vec{n}$. The magnitude V of the wind vector at X is $V = |\vec{V}| = V_s$, since $\vec{V}$ is parallel to $\vec{s}$ and V_n is locally zero.

For each grid point surrounding X, let V_s be the magnitude of the local wind vector resolved into direction $\vec{s}$. The jet-stream core co-locates with lines along which V_s is maximal within the local two-dimensional normal planes spanned by $\vec{n}$ and $\vec{z}$. This is given if the derivatives in the horizontal and vertical direction are both locally zero:

$$\frac{\partial V_s}{\partial n} = 0 \qquad (3) \qquad\qquad \frac{\partial V_s}{\partial z} = 0 \qquad (4)$$

These equations denote three-dimensional contorted isosurfaces (of zero shear vorticity), where the quasi-vertical isosurface is described by Eq. 3 and the quasi-horizontal sheet is described by Eq. 4. The set of all extremum points in 3D-space, which are considered the potential candidate points for the jet cores, is defined by the intersection of these two isosurfaces. Connections of these candidate points resemble polylines in 3D-space. We are interested in regions of high wind speeds, usually above about 40 ms^{-1}; therefore, core lines detected in light wind conditions need to be removed. We hence add an inequality mask (Eq. 5) to Eqs. 3 and 4 to focus on high wind speeds (this also helps to circumvent noise). For a given threshold α (in ms^{-1}) each candidate point has to satisfy:

$$V_s > \alpha \qquad (5)$$

From all extremum points that are candidates for the jet cores we need to filter out all those that are a local minimum or a saddle point. The type of the extremum point within the $\vec{n}$-$\vec{z}$-plane (local "normal plane") can be determined by the local Hessian matrix and the sign of its corresponding eigenvalues. The Hessian matrix H_N within the local normal plane coordinate frame is defined as follows:

$$H_N = \begin{bmatrix} \frac{\partial^2 V_s}{\partial n^2} & \frac{\partial^2 V_s}{\partial n \partial z} \\ \frac{\partial^2 V_s}{\partial z \partial n} & \frac{\partial^2 V_s}{\partial z^2} \end{bmatrix} \qquad (6)$$

Given the Hessian matrix H_N, a line point is said to be a local (convex/elliptic) maximum if the sign of both eigenvalues is negative. Hence, an additional mask is applied to all line points to extract only those where the following inequalities hold:

$$\lambda_0 < 0, \lambda_1 < 0 \qquad (7)$$

The computed eigenvalues λ_i are real since the Hessian matrix H_N is symmetric.

The major difference to ridge extraction techniques working directly on the 3D wind field is that our technique works in a fixed reference frame. Thus, we avoid working in the eigenvector-frame for both the computation of first and second order derivatives, and we can thereby avoid oscillations –due to oscillating eigenvectors– in the frame of reference for the first derivatives. This makes the technique significantly more robust for jet core extraction. Indeed this local coordinate-based

mechanism for noise removal aligns closely with the mechanism used in [15] to eliminate third-derivative noise during front detection (see his Figure 6).

3.4 Feature Extraction

We now describe the numerical computations involved to extract jet cores from 3D gridded wind fields.

3.4.1 Raw Features

We call the set of candidate points that satisfy Eqs. 3 and 4 "raw features" that potentially belong to a jet core. To extract raw features, we first compute the directional (horizontal) and vertical derivatives in 3D-space at every grid point X: First, the local coordinate frame $(\vec{s},\vec{n})$ is derived at X by using the unit vector parallel to the local horizontal wind vector and computing the orthogonal normal vector. Second, the local wind vectors at surrounding grid points are projected onto $\vec{s}$.

To solve Eq. 3, the directional derivative into direction $\vec{n}$, we apply the chain rule to obtain

$$\frac{\partial V_s}{\partial n} = (\vec{n}\cdot\vec{\nabla})V_s = n_x\frac{\partial V_s}{\partial x} + n_y\frac{\partial V_s}{\partial y}. \quad (8)$$

The partial derivatives are computed using finite differences, taking care of the geometric distance between two grid points in x and y-direction. As V_s is defined in ms^{-1} and the horizontal grid is defined in longitude–latitude space, we need to ensure consistency in the used distance metric. While the distance between two latitudinal points is constant over the globe at approximately 111 km per °, the longitudinal distance decreases towards the poles. Assuming a spherical globe, it can be computed by scaling the equatorial distance (111 km per °) by the cosine of the latitude.

The vertical derivative (Eq. 4) is computed via finite differences at each grid point X at level k into the direction of $\vec{z}$, by using the resolved wind vectors of the surrounding levels. Here, the vertical position of each grid point defined in pressure space needs to be converted to geometric height (in m) first.

3.4.2 Zero-Isosurface Crossing Extraction

A naive approach to compute the intersection of two isosurfaces implicitly defined in a 3D scalar field on a discrete voxel grid uses the Marching Cubes algorithm [26]. The triangle geometry of both isosurfaces is extracted, and per-voxel triangle intersection tests compute the intersection lines. Such an approach, however, would be very inefficient since we do not require the entire isosurface geometries but only the geometry of the raw line features.

We hence use the Marching Faces algorithm proposed by Ljung and Ynnerman [25] to implicitly extract intersection lines between two isosurfaces from two co-located scalar fields. Marching Faces traverses each voxel of a 3D grid and computes the intersection points of isosurface crossings at each voxel face, by computing crossings of the isolines of each isosurface at a face. Isolines are approximated by linear interpolation along the face edges (similar to Marching Squares), the crossing of two isolines is computed analytically.

Ljung and Ynnerman's approach first identifies all intersection points per voxel face in parallel, then joins points sharing a common voxel face to create a polyline. We have modified this approach and in our method directly trace the polylines through the grid. We traverse the grid along subsequent voxels and combine points that share a common face; tracing is stopped if a voxel does not contain a suitable intersection point or a looped curve is detected.

3.5 Filtering

After raw features have been extracted, the resulting lines are filtered to obtain features that represent local maxima in wind speed. First, all candidate points of too low wind speed are removed by applying Eq. 5 to each raw feature vertex. Results from two different windspeed thresholds are shown in Fig. 4a (40 ms $^{-1}$) and b (10 ms $^{-1}$). A smaller threshold increases the number of jet core lines as well as their length. For the application cases in this work, the domain experts were interested in jet cores with a velocity of at least 40 ms^{-1}; we use this threshold throughout this paper.

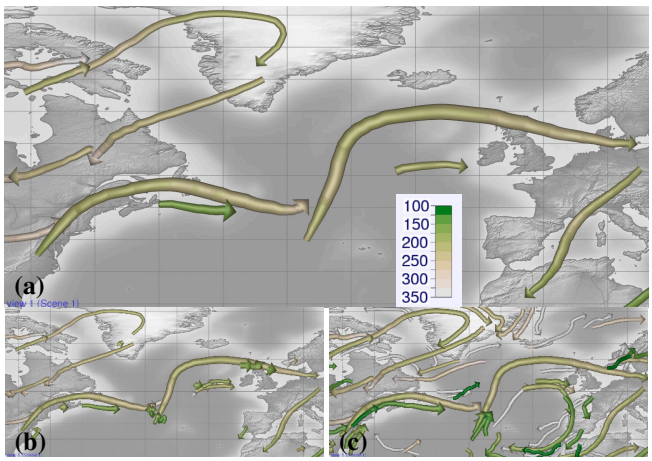


Fig. 4. 2D visualization of jet-stream core lines. **(a)** Core lines exceeding 40 ms^{-1}, filtered according to a minimum length of 500 km and a maximum angle of 55°. Tube thickness maps to the magnitude of the wind (thick lines indicate high wind speed), color maps to pressure elevation (colorbar in hPa, lower pressure corresponds to higher height). **(b)** Same as (a) but with length and angle filters disabled. **(c)** Same as (a) but with the wind speed threshold reduced to 10 ms^{-1} (length and angle filters enabled).

3.5.1 Hessian Computation

To determine which raw feature points belong to a local maximum, i.e., where V_s is locally maximal in the $\vec{n}$-$\vec{z}$-plane, we compute the Hessian matrix and its eigenvalues (Eq. 6) at each line vertex. Since the raw feature points are points in 3D-space and not located on the grid points, we compute the entries of H_N as follows: Second partial derivatives with respect to V_s are computed per grid point and are each stored in a separate grid. The second derivatives at a given line vertex are then obtained by tri-linear interpolation using the 8 grid points surrounding the vertex. As the Hessian matrix is approximated on a finite grid and its eigenvalues tend to oscillate, points can be falsely rejected (or accepted). Thus, we introduce a threshold β to soften the criterion in Eq. 7: $\lambda_i < \beta$, where β is a small positive value. Short line disconnections due to false rejections are in our method counteracted by a curve-following algorithm which keeps track of the eigenvalues along a core line. Falsely rejected points that are enclosed by two accepted points are subsequently corrected.

3.5.2 Geometric Length

Aviation centers are generally interested in jet-stream cores that at least extend over a certain distance; these cores are expected to have more influence on surrounding atmospheric conditions than short jet cores. We compute the geometric length of each core line in kilometers and remove lines whose length is below a user-specified threshold. Fig. 4a and b shows the effect of this filter; as expected, more short jet cores are detected when the length filter is omitted. For the remaining figures in this paper, we set this threshold to 500 km; shorter jet cores often resulted from small maxima regions and did not contribute to the analysis.

3.5.3 Angle Criterion

Numerical inaccuracies, in particular in regions in which the wind speed differences are small or the local maximum is ill-defined, can lead to misclassification of raw feature points, i.e., saddle or minima points are falsely detected as local maxima. These misclassified points can lie between two close jet cores and can be relocated to the same voxel. In such a case, our line tracing algorithm from Sec. 3.4.2 may combine the ends of two distinct lines so that the resulting jet cores

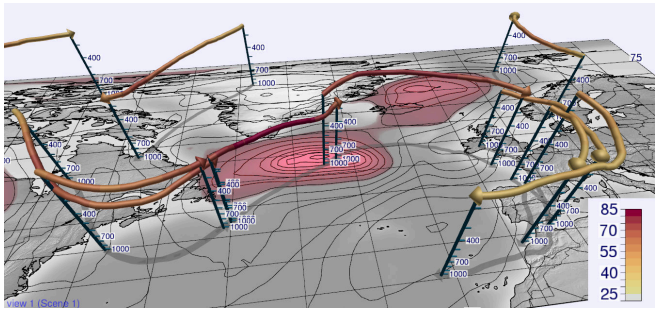


Fig. 5. Core lines rendered in 3D (colored by wind speed in ms^{-1}). Shadows cast to the ground and vertical drop-lines (labeled by pressure in hPa) significantly improve spatial perception of the view. Black surface contour lines and red highlighted regions show MSLP as in Fig. 1, aiding the analysis of the relation of the jet cores to surface weather systems.

can exhibit unphysical "bending". To filter such cases, we determine horizontal angle between each two core line segments and vertical pressure differences at the core line end segments, and apply filters to remove lines that exceed a user-defined angle or pressure difference threshold. Choosing high angle thresholds can result in more sharply bent core lines (especially in saddle point regions), as illustrated in Fig. 4c. For this work, we found thresholds of 55° and 10 hPa for angle and pressure, respectively, to yield good results in our examples.

3.6 Performance

Ljung and Ynnerman [25] showed the complexity of computing the intersection between two isosurfaces to be $\mathcal{O}(\sqrt{N})$, where N is the number of triangles in the isosurfaces (which in turn depends on grid dimensions and the characteristic of the considered scalar fields). Our subsequent filtering of the core line candidates is of order $\mathcal{O}(M)$ (M being the number of intersection points, i.e., vertices of the jet cores), yielding a total detection complexity of $\mathcal{O}(\sqrt{N}+M)$.

We have measured the performance of a CPU implementation of our detection algorithm in Met.3D [38] on a desktop computer equipped with an Intel Core i7 3770 processor with 4.0 GHz×4 cores, 32 GB RAM and an NVIDIA Geforce GTX 970. For an ENS forecast with a grid spacing of 1° in both latitude and longitude and a grid size of 131×66×70 cells, the isosurface intersection step took less than 300 ms for each single member of the 51 ensemble members. Core line filtering was performed on average in about 140 ms per member. A larger grid with a horizontal grid spacing of 0.15° and 268×669×72 cells required about 5 s for the isosurface intersection and 1.3 s for filtering.

4 Visualization Techniques

We have designed a number of visualization techniques to facilitate improved visual analysis of the detected 3D jet-stream cores in both 2D and 3D space. Our techniques support domain experts in their analysis and provide answers to important questions concerning jet cores: specifically their 3D shape, the strength of the wind along them, their orientation in relation to the wind direction, and their elevation.

4.1 Jet Core Rendering

Core line geometry is rendered in 2D and 3D as tubes. Fig. 4 shows how arrow glyphs placed at the end of each jet core line indicate their orientation; core line parameters can be encoded via tube thickness and color. This facilitates the simultaneous visualization of, (e.g.) wind speed and pressure elevation or flight level. For example, in Fig. 4, wind speed is mapped to tube thickness and pressure to color; the core lines are mainly located at pressure elevations between 200 and 300 hPa. Thin lines represent cores of weak wind speeds and likely small impact, whereas thick lines depict cores of potentially high impact.

Fig. 5 shows a 3D visualization, displaying the full 3D structure of the core lines. Notably, this 3D structure cannot be communicated via standard SIGWX charts. Since spatial perception in 3D renderings is crucial to meteorological analyses (cf. [38]), we render tube shadows cast by a directional light source from above onto the surface to show the horizontal location of the core lines. Based on feedback from domain experts in our author team, we additionally provide drop-lines to further improve spatial perception in the vertical. The drop-lines are vertical axes connecting the core line with the surface, placed at the endpoints of each core line. They are augmented by text-labeled tick marks at user-defined pressure levels to display quantitative elevation information (Fig. 5).

4.2 Jet Cores in Atmospheric Flow

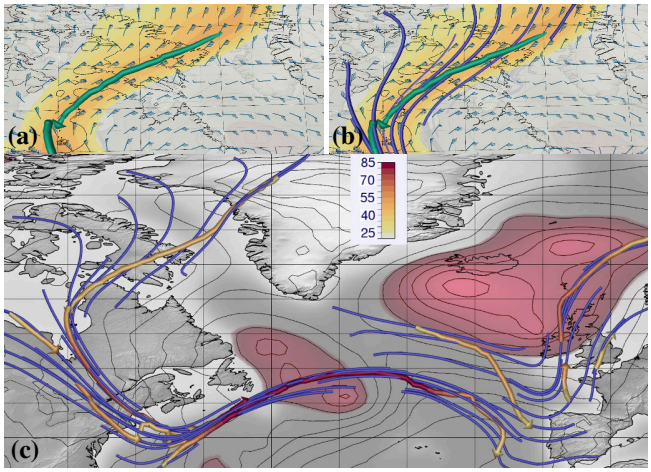


Fig. 6. Visualization of a jet-stream core line (green) and a horizontal section of the wind speed field (color in ms^{-1}), **(a)** in combination with wind barbs, **(b)** with additional blue-colored streamlines, seeded along the core line at a fixed interval, and **(c)** all jet core lines and their streamlines, with MSLP as in Fig. 5, over the North Atlantic. Such displays can help scientists to analyze the jet core orientation relative to the flow direction. In some locations the angle between the two is small, indicating that little lateral movement of the core is expected in the near future (e.g., east of Newfoundland), whereas in others the angle is large, suggesting jet core movement (e.g., near to and south-west of Ireland, where a trough and jet cores on its forward and rearward flanks are all moving east-northeast).

Jet-streams follow the large-scale, moving wave patterns in the atmosphere. Newton and Omoto [33] showed that due to energy considerations in a moving wave system, the jet-stream core line must meander across the flow's streamlines; the jet-stream wave can only move if there is a wind component normal to the core (cf. Fig. 11 in [33]), and indeed moves with a speed approximately equal to the jet-core-normal wind component. Equivalently, only in a stationary wave in which the jet core has a uniform speed is the core line expected to be everywhere tangential to the streamlines. To shed light on the strength of the meandering in real-world forecasts, and hence on the expected advection of a core with the wind, we provide options to visualize the cores in the surrounding flow field. The core lines can be embedded into visualizations using standard wind barbs (Fig. 6a) to provide map-based displays similar to those typically used in operational settings. Additionally, streamlines (started at intervals along the core) can highlight the deviation between the core and local flow direction (Fig. 6b and c).

Visualization of jet cores along with further atmospheric fields (Fig. 5 and Fig. 6b) provides entirely new possibilities to examine the relation of the cores to weather events of interest. For example, 2D surface fields including mean-sea-level pressure can be displayed as line and filled contours (Fig. 6b), whilst 3D fields including cloud water content can be visualized as 3D isosurfaces (Sect. 5.2). In such examples one can examine jet cores and their connection to cyclones and anticyclones, or the relationship of jet cores to extreme weather (such as heavy rain and strong surface winds).

4.3 Ensemble Uncertainty

The visualization techniques presented above show just a single forecast. In operational forecasting, ensembles of forecasts are in widespread use (e.g., [13]) and need to be analyzed to investigate the uncertainty represented by the forecasts. In particular, experts need to examine the variability and coherence of predicted weather conditions and, thus, with respect to our work, the spread of detected jet cores across all ensemble members.

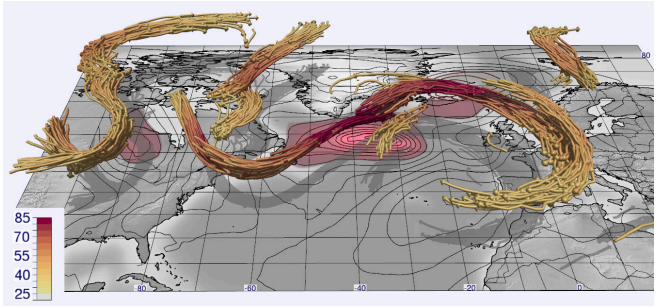


Fig. 7. 3D jet-stream core lines detected from the wind fields of 51 ensemble members of an ECMWF ENS forecast. Line color shows wind speed (ms^{-1}), MSLP is rendered as in Fig. 5.

To support such analyses, we provide "spaghetti plots", a simultaneous display of multiple members in a single image. Fig. 7 shows an example of a 3D spaghetti plot, including the core lines of all ensemble members of the considered forecast. The wider the jet cores are spread over the map, the more the forecast can be considered uncertain.

5 Results

To demonstrate the value of our method, we discuss two applications. The first application demonstrates the automatic generation of jet-streams for a SIGWX product. The second case considers a real-world ensemble forecast from the recent North Atlantic Waveguide and Downstream Impact Experiment (NAWDEX, [9]), an atmospheric research field campaign involving one of the authors. The analysis of ensemble behavior during the campaign cases is a major focus of the –at the time of writing ongoing– data analysis activities of the campaign.

5.1 Significant Weather Charts

Jet cores are marked as one component on official medium and high level SIGWX (significant weather) charts prepared for aviation purposes by meteorologists, following regulations of the International Civil Aviation Organization [17]. In practice, forecasters at the UK Met Office, one of the two world area forecast centers (WAFCs), perform this manually, broadly as follows (pers. comm., P. McGarry and D. Naylor):

1. Examine 2D fields of forecast maximum wind (in a vertical sense) depicted as isotachs (lines of constant wind speed) and vectors, supplemented by gridded wind data for various levels.
2. Draw jet core lines that broadly follow the speed maxima, but with a secondary consideration that the wind flags on the output chart, that by convention have to be shown parallel to the core line, do not depart too much from also being wind-parallel. Only include cores lines where wind speed exceeds 80 knots.
3. According to regulations in [17], add supplementary jet-related information, and also adjust to ensure correct prioritisation when depicting multiple hazards, and intelligibility for users - for example jet cores at two different levels cannot be overlaid.

Fig. 8 shows a comparison of an operationally issued SIGWX chart and jet-stream core lines detected by our approach; a second example is contained in Fig. 1. Having examined a number of cases we would describe the agreement between the SIGWX and our plots as very good. The main reasons for any discrepancies are as follows:

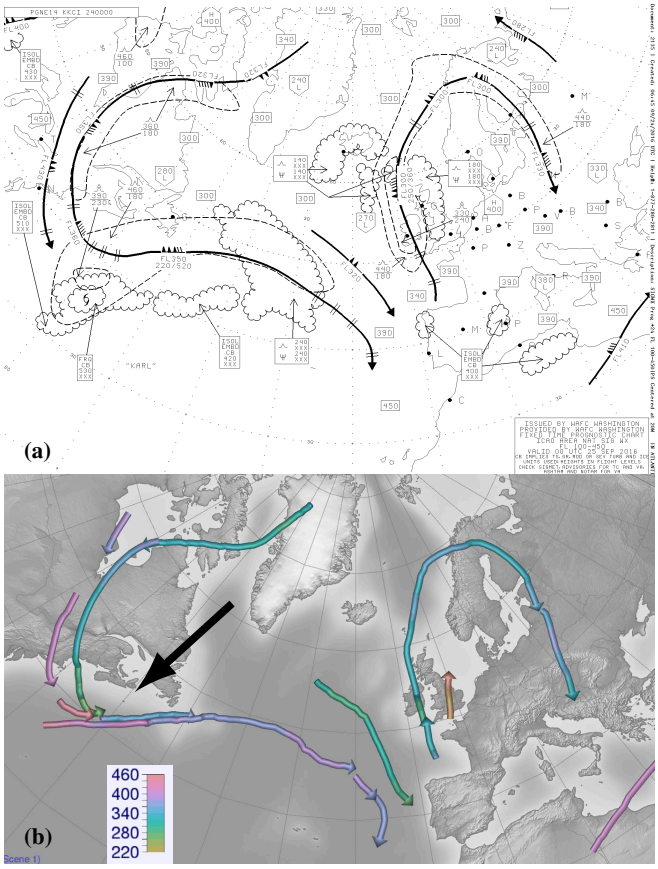


Fig. 8. Comparison of jet-stream core lines **(a)** manually identified by forecasters on operational significant weather charts, shown as thick solid arrows, with FLxxx notation alongside wind flags denoting flight level (xxx in hft), and **(b)** automatically detected by our method, colored by flight level (hft). Charts are valid at 00:00 UTC 25 September 2016; core lines in (a) are based on various forecast models available to the forecasters, those in (b) are based on the control analysis of the ECMWF ensemble forecast.

- "Artistic licence" by the SIGWX chart analyst, who has to combine multiple features intelligibly, with prioritisation, on their chart, abiding also by some official rules regarding overlaps.
- Differences in interpretation of available data between analysts (i.e., two forecasters given the same data would not produce the same chart).

In Fig. 8, the area south of Nova Scotia (black arrow in Fig. 8b) is interesting. Whilst our automated method picks out distinct jet cores at multiple levels (Fig. 8b), the manual method simplifies, showing just one jet core at 36000 ft (FL360), with a deep region of turbulence (FL180 to FL460, within the dashed line) probably added to cater for the multiple jets (Fig. 8a). Perhaps using our products the jet could be have been consigned to a more appropriate, lower level, and the turbulence region made more confined. Indeed we received the following general comment: "more information over the shape of the jets and potentially where they overlap could allow for more intelligent route planning to avoid turbulence/increase efficiencies" (pers. comm., S. Ramsdale, Chief Forecaster at the UK Met Office).

Thus our new 3D jet products can be used as helpful first guess fields to be rationalised by the SIGWX analysts. Overlapping jets, which are important and relatively frequent (see Sect. 5.2 below), are missing on the "wind maximum" field used in step 1 above, but with our method would be very visible.

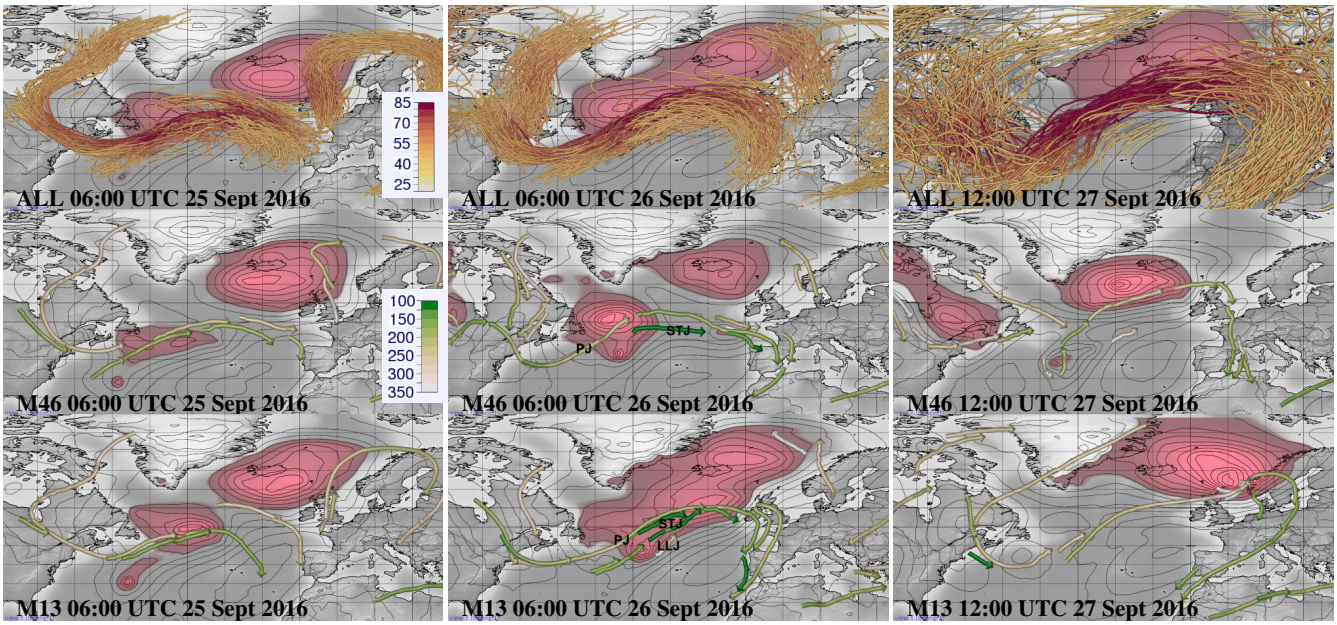


Fig. 9. Top row: "Spaghetti" representation of jet-stream core lines detected from the ECMWF ensemble forecast from 00:00 UTC 22 September 2016, valid at the indicated times. Jet-stream core lines are colored by wind speed (ms^{-1}), contours and red shading show sea level pressure (red shading indicates pressure below 1000 hPa). Middle and bottom rows: Temporal development (same time steps) in two selected members (M46 and M13). Jet-stream core lines are colored by pressure elevation (hPa). In M13, a strong cyclone develops that on 27 September hits Norway. Annotation on central panels relates to jet types; PJ for polar Jet, STJ for sub-tropical jet, LLJ for low-level jet (used also on Fig. 10).

5.2 Tropical Cyclone "Karl"

We consider a real-world case from the NAWDEX campaign that represents applications in both weather forecasting and atmospheric research into physical processes. The extratropical transition of Tropical Cyclone Karl occurred in late September 2016. The system was successfully observed in multiple research flights, but had posed significant difficulties for forecasting due to associated high uncertainty.

We focus on the ECMWF ensemble starting from 00:00 UTC 22 September 2016, and specifically on the behavior of Karl as it became an extra-tropical feature in those forecasts. The ensemble included very different outcomes. These outcomes are analyzed in relation to detected jet-stream cores. We show how identifying the 3D cores facilitates investigation of jet behavior in a way not possible with classical wind speed analysis at single levels.

The top row in Fig. 9 shows a spaghetti representation of jet cores in the ensemble, on 3 different days, with the ensemble mean surface pressure field (MSLP). Spread increases quite dramatically with time (in surface weather too, not shown). However the salient features remain clear, most notably the eastward migration of a strong jet (darker reds on core lines) into the mid Atlantic. Karl is visible on the first frame, in the MSLP field due south of Newfoundland, but then moves northeast beneath the jet(s), to potentially interact with them.

We illustrate two very different ways in which that jet interaction could have played out, using two ensemble members denoted M13 and M46. Plan view time series of jet cores and surface pressure for each are depicted in Fig. 9. Greens denote jet core altitude, darker being higher (pressure level in hPa on scale). We can see three types of jet, a polar jet (PJ) at high levels, a subtropical jet (STJ) attributable in part to outflow from Karl, at very high levels (commensurate with tropical air), and in one case also a low level jet (LLJ) close to Karl's center. Each jet may have more than one core. The behavior of the STJ relative to the PJ seems to play a pivotal role in determining subsequent evolution. Animation shows that the STJ in Fig. 9 (M46 06:00 UTC 26 Sept 2016) propagates rapidly forward away from Karl, turns anticyclonically, and reinforces the upper trough east of Iberia. Fig. 10a is a 3D view for 6h later – note how the high altitude STJ towers above other features, but is moving on, leaving the PJ behind. Conversely in Fig. 10b, 6h after Fig. 9 (M13 06:00 UTC 26 Sept 2016), the three STJ branches do not propagate forwards, and indeed the westernmost STJ branch moves north to become vertically aligned with the two PJ branches, as can be seen in the shadows, and indeed on Fig. 10c where the added section shows wind speed. This vertical stacking is commensurate with a "tropopause wall" developing, which in energetic terms is very conducive to rapid cyclogenesis should a surface low, in this case Karl, happen to move poleward of the (stacked) jet cores.

In the M46 case Karl died, as can be inferred from Fig. 9, in part because the jet configuration did not help its development. However in the M13 case Karl crossed the cores and developed very rapidly, becoming a sub-970 hPa low center with extreme surface winds near Norway (cf. Fig. 9). Fig. 10d shows a rendering of the jet cores, cloud field and MSLP, 24 h after Fig. 10b and c. The STJ migrated east as the development ensued, the PJ is still very strong, with it's left exit area near the low, assisting cyclogenesis. We also see a new mid level jet (MLJ) connecting up to the STJ (showing also that our code can identify altitude changes in jet level well). In addition there are two new LLJ cores at low levels. The lower one of these begins around 800 hPa, with hints of greater strength at its eastern end, reminiscent of the sting jet (SJ) phenomena implicated in many damaging European windstorms [6, 16], though further analysis would be needed to prove this connection.

We have seen that jet behavior is pivotal in this example, notably for the STJ. Further related research can focus on the role moist processes (for example) play in dictating jet behavior, which in turn feeds back on synoptic evolution. Other aspects that this case usefully reveals, also worthy of further study, are the mid level jet on Fig. 10d, and its upward connection, and the trough extension effect of the STJ in Fig. 10a. Thus 3D jet identification can highlight in a particularly compact and illuminating way new aspects of atmospheric structure that can be missed by classical 2D analysis methods.

Furthermore, regarding forecasting applications, domain expert S. Ramsdale (UK Met Office, pers. comm.) comments: "your approach...could be easily extended to show interactions between upper/surface level features in terms of perhaps vertical velocity around the cores, showing their penetration depth for development, allowing

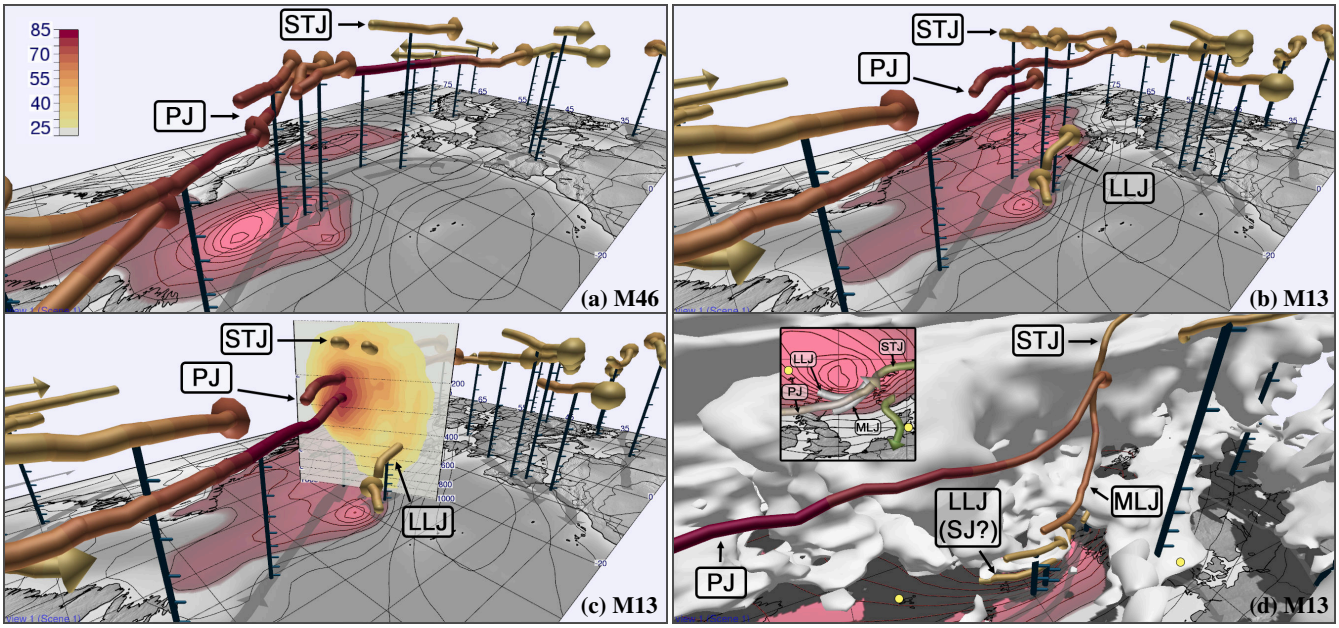


Fig. 10. Application of the proposed visualizations to the analysis of the extra-tropical transition of Tropical Cyclone Karl. Jet-stream core lines detected in the ECMWF ensemble forecast from 00:00 UTC 22 September 2016, valid 12:00 UTC 26 September 2016. Core lines are colored by wind speed (ms^{-1}), contours show sea level pressure (red shading indicates pressure below 1000 hPa): **(a)** 3D view for member 46, **(b)** 3D view for member 13, **(c)** The same as (b) but with an added vertical section colored by wind speed, **(d)** 3D structure of member 13 off the coast of Norway, valid 12:00 UTC 27 September 2016. White isosurface encloses cloud cover fraction larger than 0.55. Inset, for same time, extracted from lower right panel of Fig 9. Yellow dots (at sea level) added to aid registration.

for again more objective assessments of how differences in shape/speed may lead to differences in evolution". In addition he highlights the utility, for forecasting purposes, of real-time comparison of observations (e.g. winds from aircraft) with our jet cores.

For future research, one other area to highlight in which application of our method will be of interest is 'forecast failures'. Rodwell et al. [41] found that these often stem from modulation of downstream flow, and notably upper level jets, by mishandled convective outbreaks over North America. Our new tools will highlight the upscale effects of such convective errors in revealing ways, with spaghetti jet plots for example (see Fig. 9) likely to yield key insights in a fraction of the time it would ordinarily take to examine all ensemble members.

6 Conclusion

We have proposed a robust detection method for identifying jet-stream core lines in atmospheric flow, and have presented visualization techniques that facilitate analysis of 3D jet-stream behavior in a way not possible with classical meteorological wind speed analysis at single vertical levels. Our method is to some extent similar to 3D height ridge detection but exploits wind direction information to achieve increased stability and greater agreement with classical manual detection methods. We have developed our methodology within a team of visualization and atmospheric scientists, have demonstrated how the method behaves when fed with realistic wind fields from numerical weather forecasts, and have proposed 2D and 3D visualization techniques.

Detection and visualization has been incorporated into the open-source meteorological 3D ensemble visualization tool "Met.3D" to facilitate combination of the new jet features with visualization of other important meteorological phenomena, and in order to promulgate the general methodology into the meteorological community. We have demonstrated how our method supports analysis that relies on core line geometry, including investigation of core line relationship to streamlines and investigation of jet core uncertainty inherent in ensemble weather prediction.

Two case studies have highlighted the value of our method for meteorological applications. We examined the automatic identification of jet-stream core lines for global SIGWX charts used worldwide in aviation, and we examined closely the 3D jet-stream behavior during a specific weather case involving the extratropical transition of Tropical Cyclone Karl.

In conclusion, we are confident that our method will facilitate many new and valuable studies in atmospheric research, and that it will bring important benefits to operational weather forecasting. In our case study we have already identified interesting 3D jet-stream structures that are very relevant for whether or not extreme and damaging weather will develop at the surface. We are confident this will stimulate further meteorological research that addresses societal needs. Above all, we have achieved for the first time a compact, smooth, continuous 3D depiction of one of the most fundamental atmospheric features –the jet stream– that plays a pivotal role in determining world weather, and that even achieves frequent references in the media.

Acknowledgments

The research leading to these results has been done within the subproject "Visualization of coherence and variation in meteorological dynamics" of the Transregional Collaborative Research Center SFB/TRR 165 "Waves to Weather" funded by the German Research Foundation (DFG). The work was partly funded by the European Union under the ERC Advanced Grant 291372 SaferVis: Uncertainty Visualization for Reliable Data Discovery. Access to ECMWF prediction data has been kindly provided in the context of the ECMWF special project "Support Tool for HALO Missions".

References

[1] C. L. Archer and K. Caldeira. Historical trends in the jet streams. *Geophys. Res. Lett.*, 35(8):L08803+, Apr. 2008.
[2] C. Baehr, B. Pouponneau, F. Ayrault, and A. Joly. Dynamical characterization of the FASTEX cyclogenesis cases. *Q.J.R. Meteorol. Soc.*, 125(561):3469–3494, Oct. 1999.
[3] N. P. Barton and A. W. Ellis. Variability in wintertime position and strength of the North Pacific jet stream as represented by re-analysis data. *Int. J. Climatol.*, 29(6):851–862, May 2009.

[4] G. Berry, C. Thorncroft, and T. Hewson. African easterly waves during 2004—Analysis using objective techniques. *Mon. Wea. Rev.*, 135(4):1251–1267, Apr. 2007.
[5] D. C. Breton. Note sur les lignes de faîte et de thalweg. *Comptes rendus hebdomadaires des séances de l'académie des Sciences*, 39:647–648, 1854.
[6] K. A. Browning. The sting at the end of the tail: damaging winds associated with extratropical cyclones. *Q.J.R. Meteorol. Soc.*, 130(597):375–399, Jan. 2004.
[7] K. A. Browning and N. M. Roberts. Structure of a frontal cyclone. *Q.J.R. Meteorol. Soc.*, 120(520):1535–1557, Oct. 1994.
[8] M. de Saint-Venant. Surfaces à plus grande pente constituées sur des lignes courbes. *Bulletin de la Société Philomathématique de Paris*, pp. 24–30, 1852.
[9] Deutsches Zentrum für Luft- und Raumfahrt. NAWDEX - North Atlantic Waveguide and Downstream Impact Experiment. `http://www.pa.op.dlr.de/nawdex`, 2016. Accessed 26 July 2017.
[10] D. Eberly, R. Gardner, B. Morse, S. Pizer, and C. Scharlach. Ridges for image analysis. *Journal of Mathematical Imaging and Vision*, 4(4):353–373, 1994.
[11] J. D. Furst and S. M. Pizer. Marching ridges. In M. H. Hamza, ed., *SIP*, pp. 22–26. IASTED/ACTA Press, 2001.
[12] D. Gallego, P. Ribera, R. Garcia-Herrera, E. Hernandez, and L. Gimeno. A new look for the southern hemisphere jet stream. 24(6):607–621, 2005.
[13] T. Gneiting and A. E. Raftery. Weather forecasting with ensemble methods. *Science*, 310(5746):248–249, 2005.
[14] R. M. Haralick. Ridges and valleys on digital images. *Computer Vision, Graphics, and Image Processing*, 22(1):28 – 38, 1983.
[15] T. D. Hewson. Objective fronts. *Met. Apps*, 5(1):37–65, 1998.
[16] T. D. Hewson and U. Neu. Cyclones, windstorms and the IMILAST project. *Tellus A: Dynamic Meteorology and Oceanography*, 67(1):27128+, Jan. 2015.
[17] International Civil Aviation Organization. *Meteorological Service for International Air Navigation, Annex 3to the Convention on International Civil Aviation*. International Civil Aviation Organization, 999 University Street, Montréal, Quebec, Canada H3C 5H7, 17th ed., 2010.
[18] D. N. Kenwright, C. Henze, and C. Levit. Feature extraction of separation and attachment lines. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):135–144, 1999.
[19] G. Kindlmann. Semi-automatic generation of transfer functions for direct volume rendering. Master's thesis, Cornell University, USA, 1999.
[20] P. Koch, H. Wernli, and H. C. Davies. An event-based jet-stream climatology and typology. *Int. J. Climatol.*, 26(3):283–301, 2006.
[21] M. Leutbecher and T. Palmer. Ensemble forecasting. *J. Comput. Phys.*, 227(7):3515–3539, Mar. 2008.
[22] Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA*, 28(8):1347–1352, 1990.
[23] S. Limbach, E. Schömer, and H. Wernli. Detection, tracking and event localization of jet stream features in 4-D atmospheric data. *Geosci. Model Dev.*, 5(2):457–470, Apr. 2012.
[24] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. 30(2):117–156, 1998.
[25] P. Ljung and A. Ynnerman. Extraction of intersection curves from isosurfaces on co-located 3D grids. In *The Annual SIGRAD Conference. Special Theme - Real-Time Simulations. Conference Proceedings from SIGRAD2003*, number 10, pp. 23–28. Linköping University Electronic Press; Linköpings universitet, 2003.
[26] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, vol. 21 of *SIGGRAPH '87*, pp. 163–169. ACM, New York, NY, USA, July 1987.
[27] G. M. Machado, F. Sadlo, and T. Ertl. Local extraction of bifurcation lines. In *Proceedings of International Workshop on Vision, Modeling and Visualization (VMV)*, pp. 17–24, 2013.
[28] P. Majer. *A statistical approach to feature detection and scale selection in images*. Niedersächsische Staats-und Universitätsbibliothek, 2000.
[29] G. L. Manney, M. I. Hegglin, W. H. Daffer, M. L. Santee, E. A. Ray, S. Pawson, M. J. Schwartz, C. D. Boone, L. Froidevaux, N. J. Livesey, W. G. Read, and K. A. Walker. Jet characterization in the upper troposphere/lower stratosphere (UTLS): applications to climatology and transport studies. *Atmospheric Chemistry and Physics*, 11(12):6115–6137, June 2011.
[30] G. L. Manney, M. I. Hegglin, W. H. Daffer, M. J. Schwartz, M. L. Santee, and S. Pawson. Climatology of upper Tropospheric–Lower stratospheric (UTLS) jets and tropopauses in MERRA. *J. Climate*, 27(9):3248–3271, Jan. 2014.
[31] O. Martius. A Lagrangian analysis of the northern hemisphere subtropical jet. *J. Atmos. Sci.*, 71(7):2354–2369, Mar. 2014.
[32] S. Molnos, T. Mamdouh, S. Petri, T. Nocke, T. Weinkauf, and D. Coumou. A network-based detection scheme for the jet stream core. *Earth System Dynamics*, 8(1):75–89, Feb. 2017.
[33] C. W. Newton and Y. Omoto. Energy distribution near jet stream, and associated wave-amplitude relations. *Tellus*, 17(4):449–462, Nov. 1965.
[34] E. Palmen and C. W. Newton. *Atmospheric circulation systems: their structure and physical interpretation (International Geophysics)*. Academic Press, Aug. 1969.
[35] R. Peikert and F. Sadlo. Height ridge computation and filtering for visualization. In *2008 IEEE Pacific Visualization Symposium*, pp. 119–126, March 2008.
[36] C. Pena-Ortiz, D. Gallego, P. Ribera, P. Ordonez, and M. D. Alvarez-Castro. Observed trends in the global jet stream characteristics during the second half of the 20th century. *J. Geophys. Res. Atmos.*, 118(7):2702–2713, Apr. 2013.
[37] A. Persson and E. Andersson. *User guide to ECMWF forecast products, version 1.1*. European Centre for Medium-Range Weather Forecasts (ECMWF), Reading, UK, 2013.
[38] M. Rautenhaus, M. Kern, A. Schäfler, and R. Westermann. Three-dimensional visualization of ensemble weather forecasts – Part 1: The visualization tool Met.3D (version 1.0). *Geosci. Model Dev.*, 8(7):2329–2353, 2015.
[39] E. R. Reiter. *Jet-stream meteorology*. University of Chicago Press, 1st ed.
[40] L. Rikus. A simple climatology of westerly jet streams in global reanalysis datasets part 1: mid-latitude upper tropospheric jets. pp. 1–26, 2015.
[41] M. J. Rodwell, L. Magnusson, P. Bauer, P. Bechtold, M. Bonavita, C. Cardinali, M. Diamantakis, P. Earnshaw, A. Garcia-Mendez, L. Isaksen, E. Källén, D. Klocke, P. Lopez, T. McNally, A. Persson, F. Prates, and N. Wedi. Characteristics of occasional poor medium-range weather forecasts for Europe. *Bull. Amer. Meteor. Soc.*, 94(9):1393–1405, Feb. 2013.
[42] M. Roth. *Automatic Extraction of Vortex Core Lines and Other Line-Type Features for Scientific Visualization*. PhD thesis, Swiss Federal Institute of Technology, ETH Zürich, 2000.
[43] F. Sadlo and R. Peikert. Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1456–1463, Nov 2007.
[44] J. Sahner, T. Weinkauf, N. Teuber, and H. C. Hege. Vortex and strain skeletons in Eulerian and Lagrangian frames. *IEEE Transactions on Visualization and Computer Graphics*, 13(5):980–990, Sept 2007.
[45] R. Schiemann, D. Lüthi, and C. Schär. Seasonality and interannual variability of the westerly jet in the Tibetan plateau region. *J. Climate*, 22(11):2940–2957, June 2009.
[46] S. C. Shadden, F. Lekien, and J. E. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3-4):271 – 304, 2005.
[47] C. Spensberger, T. Spenger, and C. Li. Upper tropospheric jet axis detection and application to the boreal winter 2013/14. *Mon. Wea. Rev.*, Mar. 2017.
[48] R. C. Strawn, D. N. Kenwright, and J. Ahmad. Computer visualization of vortex wake systems. *AIAA journal*, 37(4):511–512, 1999.
[49] C. Strong and R. E. Davis. The surface of maximum wind as an alternative to the isobaric surface for wind climatology. *Geophys. Res. Lett.*, 32(4):L04813+, Feb. 2005.
[50] C. Strong and R. E. Davis. Winter jet stream trends over the northern hemisphere. *Q.J.R. Meteorol. Soc.*, 133(629):2109–2115, Oct. 2007.
[51] C. Strong and R. E. Davis. Variability in the position and strength of winter jet stream cores related to northern hemisphere teleconnections. *J. Climate*, 21(3):584–592, Feb. 2008.
[52] D. Sujudi and R. Haimes. Identification of swirling flow in 3D vector fields. In *12th AIAA Computational Fluid Dynamics Conference*, pp. 95–1715, 1995.
[53] C. Tyson and M. Strahan. *Representing WAFS Significant Weather (SIGWX) Data in BUFR, Version 4.3*. World Area Forecast Centres (WAFCs), London and Washington, 2013.
[54] World Meteorological Organization. *International Meteorological Vocabulary*. WMO. WMO, 2nd ed., 1992.

Interactive 3D Visual Analysis of Atmospheric Fronts

Michael Kern, Tim Hewson, Andreas Schäfler, Rüdiger Westermann, and Marc Rautenhaus

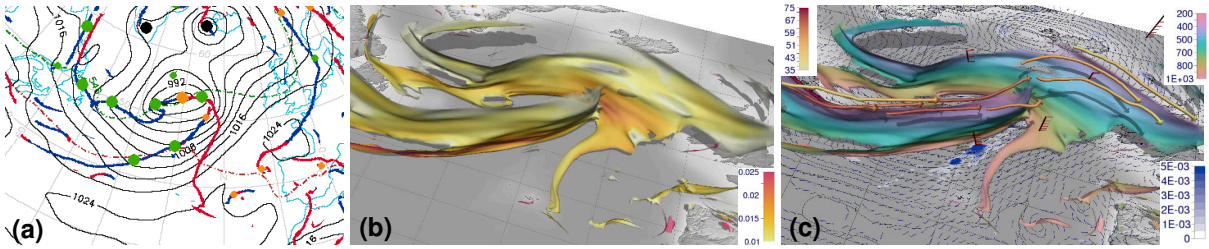


Fig. 1. Cyclone "Vladiana", 00:00 UTC 23 September 2016. **(a)** Objectively identified fronts at 1 km above ground from ECMWF operations, using the algorithm described in [15]. **(b)** 3D fronts identified and visualized with our method, color denotes frontal strength (K km^{-1}). **(c)** 3D fronts combined with further meteorological fields and features. Front color denotes pressure (hPa). Overlain are jet-stream core lines detected and visualized with the approach by Kern et al. [19], colored by wind speed (m s^{-1}). Surface contours show mean sea level pressure. Blue surface color shows precipitation (m h^{-1}). Wind barbs show surface wind.

Abstract— Atmospheric fronts play a central role in meteorology, as the boundaries between different air masses and as fundamental features of extra-tropical cyclones. They appear in numerous conceptual model depictions of extra-tropical weather systems. Conceptually, fronts are three-dimensional surfaces in space possessing an innate structural complexity, yet in meteorology, both manual and objective identification and depiction have historically focused on the structure in two dimensions. In this work, we –a team of visualization scientists and meteorologists– propose a novel visualization approach to analyze the three-dimensional structure of atmospheric fronts and related physical and dynamical processes. We build upon existing approaches to objectively identify fronts as lines in two dimensions and extend these to obtain frontal surfaces in three dimensions, using the magnitude of temperature change along the gradient of a moist potential temperature field as the primary identifying factor. We introduce the use of normal curves in the temperature gradient field to visualize a frontal zone (i.e., the transitional zone between the air masses) and the distribution of atmospheric variables in such zones. To enable for the first time a statistical analysis of frontal zones, we present a new approach to obtain the volume enclosed by a zone, by classifying grid boxes that intersect with normal curves emanating from a selected front. We introduce our method by means of an idealized numerical simulation and demonstrate its use with two real-world cases using numerical weather prediction data.

Index Terms—Meteorology, Atmospheric Fronts, Feature Detection

1 INTRODUCTION

In meteorology, fronts separate atmospheric air masses of different characteristics (e.g., warm and humid versus cold and dry; see, e.g., [56]). Indeed, fronts are among the most important features used in weather forecasting due to the associated weather activity, ranging from temperature changes to severe weather. They are most commonly denoted by manually-analyzed 2D line segments on weather maps. Ordinarily such lines represent fronts at the surface; an automatically generated example is provided in Fig. 1a. Conceptually, however, fronts are surfaces in 3D space, yet only very occasionally can one see fronts at upper levels marked on a standard chart. This is in spite of regular references to the importance of the vertical structure for surface weather [2, 6, 22, 24]. Identifying fronts and judging their 3D temporal evolution can thus be crucial for weather forecasting; however, analysts currently lack the tools and the time to investigate this in detail. Frontal structures have also been extensively studied in atmospheric research, mainly using manual analysis of 2D sections of related fields (including temperature, humidity, wind; e.g., [11, 12, 22, 23, 29, 48]), with a recent focal point being complex spatio-temporal structures and related processes (e.g., [32, 49]).

The subject of the present work is the objective detection and visualization of 3D frontal structures from numerical simulation output. The primary goals are facilitating further research to improve understanding of fronts and related weather systems, as well as application within forecasting. A number of studies have investigated the automated detection of 2D frontal lines from numerical simulation output, for weather forecasting, for creating climatologies, and for other applications (e.g., [3, 14, 15, 17]). However, to our knowledge, no previous method exists that can detect and visualize the full 3D structure of fronts, and facilitate a rapid analysis of vertical structure, frontal zone properties, and related atmospheric processes, such as vertical motion, moisture transport and precipitation (cf. Locatelli et al. [22]). Such an approach will be beneficial because:

- Using static 2D horizontal and vertical maps and sections, as now, is clearly a sub-optimal way to picture and analyze 3D reality. Hewson [14] pointed out the rich complexity in vertical structures in fronts, whilst Mulqueen and Schultz [32] referenced commonly occurring "double front systems". Such aspects are contrary to the widely accepted "Norwegian" conceptual model [4]. 3D visualization opens the door to investigate these inconsistencies, in so far as "reality" is represented within state-of-the-art numerical models. This is also in the spirit of Schultz and Vaughan's recom-

- *Michael Kern, Rüdiger Westermann and Marc Rautenhaus* are with the Computer Graphics & Visualization Group, Technische Universität München, Garching, Germany. (*M.R. is now with Universität Hamburg, Regional Computing Center, Hamburg, Germany.)
  E-mail: {michi.kern, westermann, marc.rautenhaus}@tum.de
- *Tim Hewson is with the European Centre for Medium-Range Weather Forecasts, Reading, UK. E-mail: tim.hewson@ecmwf.int*
- *Andreas Schäfler is with the Deutsches Zentrum für Luft- und Raumfahrt (DLR), Oberpfaffenhofen, Germany. E-mail: andreas.schaefler@dlr.de*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

mendation [49] that traditional conceptual models be revisited.

- In operational weather forecasting, accounting for the full 3D structure of fronts, using manually constructed 2D charts, is too time consuming. Typically, the most a forecaster can achieve in the available time is analysis of one or two cross sections. We would thus like to pave the way for increased operational use of 3D front visualization. For example, 3D visualization can enable a forecaster to rapidly check whether a rainband relates to particularly steep or even over-turning front topography aloft.

In this article, we –a team of visualization scientists and meteorologists– propose an approach to visualize and analyze frontal structures in 3D. We contribute:

- An adaptation and extension of the 2D objective front detection method of Hewson [14] to provide the first ever visualizations of continuous 3D frontal surfaces and associated meteorological features within real-world cyclones.
- A detection scheme to identify, determine and visualize characteristics of the frontal zone associated with a frontal surface based on tracing "normal curves".
- An interactive selection scheme to isolate an "interesting" front from a complex and potentially cluttered depiction of a large region and to determine statistical information about that frontal surface, its associated frontal zone, and related processes.
- Various visualizations that relate the frontal structure to associated atmospheric processes (e.g., vertical motion, moisture transport and precipitation).

We demonstrate how our method facilitates, for example, the analysis of the vertical "extent" of a front and distinction between low-level and upper-level fronts; the horizontal breadth of a frontal zone and strength of, e.g., temperature and humidity gradients; the slope of a frontal surface, undulations on it, and their influence on vertical motion; distribution of physical parameters within a frontal zone; and the relative locations of front, jet stream, and surface pressure distribution.

Our work has been partly motivated by ongoing analysis of mid-latitude cyclones and associated frontal systems observed during the atmospheric science field campaign "NAWDEX" [45], in which two of the authors were involved. This research on predictability of weather investigates how different physical processes can influence cyclone evolution. Fronts are an integral part of the NAWDEX cases and indeed of other cyclones, and rapid analysis of their 3D characteristics is required to gain improved understanding and improve future predictions.

2 Basics and Related Work

We first introduce a definition of atmospheric fronts and review related work on objective detection methods from the atmospheric sciences. The present article adds to the literature on visualization in meteorology, a comprehensive overview of which was recently presented in the survey by Rautenhaus et al. [36]. In other areas of visualization, surface-type features are of importance as well, and briefly reviewed below.

2.1 Definition of Atmospheric Fronts

The "traditional" notion of a front as the horizontal boundary between two air masses, i.e., volumes of air with nearly coherent characteristics (e.g., [56]), dates back to the "Bergen school", about 100 years ago (e.g., [4]). The glossary of the American Meteorological Society (AMS) defines a front as "the interface or transition zone between two air masses of different density", stressing that a front almost "invariably separates air masses of different temperature" but noting that many other features may distinguish a front, including a change in wind direction or a moisture discontinuity [1]. Hewson [14] describes this interface as a "thin layer, or non-rigid slab-like region, in three-dimensional space, within which there are [...] large horizontal gradients in the thermal characteristics". Fig. 2 illustrates the concept.

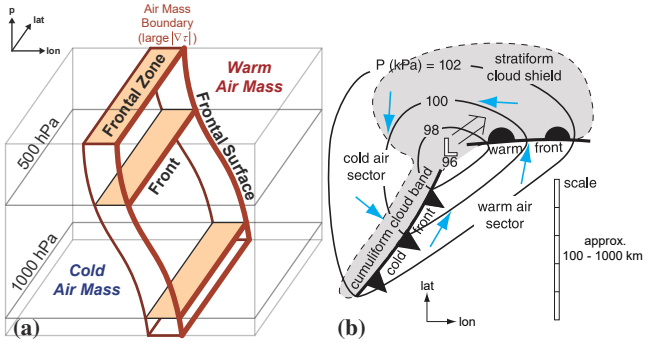


Fig. 2. **(a)** Illustration of a frontal surface and associated volumetric frontal zone separating warm and cold air masses in a 3D temperature field τ. The frontal surface (front) is located on the warm-air side of the frontal zone. **(b)** Components of a typical extratropical cyclone in the Northern Hemisphere, highlighting the pivotal role of cold and warm fronts. (Adapted from [56]. Copyright 2017 R. Stull, CC-NC-SA 4.0 license.)

Important in this respect is that only the *horizontal* thermal gradient is considered. Although fronts commonly slope (Fig. 2a), and so also have a vertical thermal gradient associated, a more extreme class of vertical gradients also occurs in the atmosphere and for this the meteorological cause is the "opposite" of frontal processes. Such gradients are due to large-scale subsidence (commensurate with settled weather) and are known as anticyclonic inversions. Clearly one must avoid misidentifying these as fronts. Typical values for an anticyclone-related vertical gradient are orders of magnitude greater than they are for a strong horizontal frontal gradient (e.g., 10 K / 100 m versus 10 K / 100 km, respectively). Meanwhile Milionis and Davies [31] showed that inversion frequency and average inversion strength (for a UK site) are both greater in anticyclonic than in cyclonic (frontal) conditions. These two points further emphasize the need to focus on just horizontal rather than 3D gradients when identifying fronts.

In Fig. 2a, where the vertical scaling is greatly amplified relative to the horizontal, edges of the 3D frontal region are marked in brown. In the 2D definition, intersection of the this region with pseudo-horizontal surfaces is denoted by a "frontal zone" (or "transition zone" in the AMS definition). According to Martin [27], the length of the frontal zone is significantly greater than its breadth. Note that in this paper, we will denote the full 3D frontal layer as the frontal zone. The warm side of the frontal zone on a level is the "front"; atmospheric dynamics dictate that this is where discontinuities in other parameters, such as the wind field, should typically lie (e.g., [14, 56]). Some authors have also considered vorticity maxima as an identifying feature, or recommended using "frontogenesis" (usually the total derivative with respect to time of the magnitude of the horizontal temperature gradient) to define frontal regions (e.g., [42, 50]).

A number of different types of fronts are distinguished (cf. [50, 56]); In the present paper, we consider synoptic-scale phenomena in extratropical cyclones (vs. mesoscale phenomena). Fig. 2b is a snapshot of the the location of fronts in a typical mid-latitude cyclone, approximately following the "Norwegian" model dating back to the Bergen school around 1920 [4]. The model still nowadays finds application in synoptic meteorology and operational forecasting. It was extended by the "Shapiro-Keyser" conceptual model [52], which differs in the structure of the "occlusion process", the "merging" of warm and cold fronts. An overview of the literature about fronts and related conceptual models is provided by, e.g., Schultz and Vaughan [49].

2.2 Objective Fronts

In operational meteorology, identification and tracking of fronts are crucial, as emphasized, e.g., by Hewson [14] and Schultz and Blumen [50]. As noted in Sect. 1, 2D surface fronts are usually manually analyzed by weather forecasters. Such analyses are commonly based on many

different parameters and inherently subjective; research has shown that different forecasters frequently recognize different fronts from the same data (e.g., [28, 43]). In this respect, Schultz and Blumen [50] pointed out that "part of the dilemma with frontal analysis is that the characteristics of fronts used for analysis are not clearly defined."

Hence, a number of studies have investigated the objective identification of frontal lines, dating back to a paper by Renard and Clarke [39] in the 1960s. Analogous to manual analysis, research focussed on two dimensions, and included application to forecasting (e.g., [14, 15]) and to compute climatologies of the occurrence of fronts (e.g., [3, 17, 47, 53]). A review of objective identification works up to the late 1990s is contained in Hewson [14]; more recent studies have also included investigation of specific processes related to the identified fronts (e.g., [16–18, 35, 46]). The method by Hewson and Titley [15] is operationally run at the European Centre for Medium-Range Weather Forecasts (ECMWF) to produce products including the one shown in Fig. 1a.

Common to most published methods is the identification of a front based on a thermal parameter τ that is representative of the considered air masses. A common choice is a moist potential temperature such as wet-bulb potential temperature θ_w or equivalent potential temperature θ_e, both essentially conserved under moist adiabatic ascent and hence essentially invariant to vertical motion in front vicinity. Details on these quantities can be found in meteorology textbooks (e.g., [56]). The frontal definitions referenced in Sect. 2.1 are directly applied to τ; the objective identification methods seek to find regions where the magnitude $M_\tau = |\nabla_h \tau|$ of the horizontal gradient of τ is large (the frontal zone) and subsequently the warm side of this zone, i.e., essentially parts of ridge lines in the first derivative of M_τ (cf. [14]). Some studies, however, employ simpler criteria. The method proposed in this article is based on the method by Hewson [14, 15], which can be considered representative of the current state of the art in meteorology.

2.3 Extremal Structures in Visualization

The existing 2D objective front detection methods extract line-type features; the method we propose generates surface-type features in 3D space. The extraction and visualization of line-type and surface-type features is also an essential tool and ongoing research branch in flow visualization. The importance of such originates from the high-dimensional nature of spatio-temporal varying flow fields –volumetric scalar and vector fields over time– and the difficulty to visually represent their inherent chaotic and turbulent structures in 3D. This requires effective visualization techniques to reduce the complexity of flow fields and leverage 3D analysis.

Related to the frontal features we extract are local extrema in n-dimensional scalar fields, i.e., 2D ridge (valley) lines and 3D ridge surfaces [5, 7]. 2D ridge (valley) lines are widely used in computer vision and image analysis to depict characteristic structures that exhibit a local maximum (minimum) along the transverse direction [13, 21, 26]. For ridge surfaces, this direction is derived locally from the full 3D tensor describing the field's variation in the surrounding. Furst and Pizer [10] proposed a technique, called Marching Ridges, to obtain ridge surfaces from three-dimensional scalar fields, by tracing their transverse direction through the volume. Kindlmann et al. [20] applied ridge surfaces to visualize diffusion tensor MRI data. In flow visualization, ridges serve as an indicator and approach to determine and extract vortex core lines [55], flow separations [51], or to visualize vorticity and strain [41]. Sadlo et al. [40] used ridge detection to reveal separating regions of different flow behavior in unsteady vector fields. Peikert et al. [33] proposed a method to compute ridge lines and ridge surfaces from n-dimensional scalar fields without the explicit computation of eigenvectors from the Hessian matrix. In our scenario, where the horizontal gradient magnitude is orders of magnitudes smaller than the 3D gradients, ridge surface extraction does not produce meaningful structures. Even in 2D, where in our case classical ridge line detection requires a fourth-order derivative, highly fuzzy and disconnected structures can occur.

3 Design Objectives and Method Overview

We propose a 3D visualization approach that enables meteorologists to explore, for the first time, the structure of atmospheric fronts which are fundamentally 3D features, and to examine frontal characteristics and related atmospheric processes. Our design, motivated the lack of such an approach, e.g., in the ongoing NAWDEX analyses (cf. Sect. 1), targets the following: (a) Analysis of the full 3D spatial structure of frontal surfaces (to judge, e.g., spatial coherence and steepness of the surface topography) in the context of the atmospheric environment (i.e., integration of the front display with existing meteorological visualizations). (b) Analysis of atmospheric quantities at the front location and in its vicinity (e.g., vertical motion, humidity). (c) Analysis of the frontal zone associated with a frontal surface, including its spatial structure and distribution of atmospheric quantities in the zone, to focus in on regions where the physical processes that drive adverse weather are concentrated. (d) Joint analysis of frontal structures with related processes and features (e.g., jet-streams, precipitation).

To achieve these goals, we show how one can depict, in 3D, frontal surfaces and frontal zones color-coded to represent various important atmospheric quantities and related properties. Due to known skepticism in the meteorological community regarding 3D visualization (cf. the discussion in Rautenhaus et al. [36]), consideration of spatial perception and interactivity was deemed important. To facilitate combined visualization with further atmospheric features, we have integrated our approach into the state-of-the-art open-source meteorological visualization tool "Met.3D" [30, 38]; by providing our method in an existing tool we also ease promulgation into the meteorological community.

The proposed detection method for 3D front surfaces (Sect. 4) follows the 2D approach by Hewson [14] and uses an arbitrary thermal parameter τ (cf. Sect. 2.2) selected by the user to compute feature candidate surfaces representing potential fronts. The candidates are filtered according to a number of criteria to obtain the final front surfaces. Since, to filter candidate features, the existing 2D methods discussed in Sect. 2.2 use "hard" threshold values selected based on the specific data investigated, we propose interactive adjustment of these criteria by the user to facilitate investigation of the effect of changing, e.g., the minimum strength of the thermal gradient. For visualization of the frontal zones associated with the surfaces (Sect. 5), "normal curves" are used to simultaneously display horizontal breadth of the zones, structure of the thermal gradient, and the distribution of any NWP quantity of interest. To remove clutter from a complex scene, a front feature of interest can be selected and displayed in isolation. Frontal-zone distributions of an NWP quantity of interest can be displayed for any selected front by means of a histogram. Finally, the depiction can be combined with visualizations of further features of interest, e.g., jet-stream core lines [19].

4 Detection and Visualization of Front Surfaces

This section describes the extraction, filtering, and visualization of 3D front surfaces. We introduce our method with data from a numerical simulation of an atmosphere on an "aquaplanet", a flat planet covered only by water. Details of the simulation are described by Schäfer and Voigt [44], it develops a series of cyclones that exhibit an idealized structure. One of these is selected to introduce our method with relatively "smooth" data before investigating real-world cases in Sect. 6. The aquaplanet simulation is available on a regular latitude–longitude grid with a grid spacing of 0.5° (approximately 50 km) in both horizontal dimensions; in the vertical 35 levels of constant pressure are available. Fig. 4a shows a horizontal map of θ_w of the selected region on the 925 hPa pressure surface, illustrating the clear separation of cold air masses in the north and warm air masses in the south.

At its core, our approach closely follows the method proposed for 2D front lines by Hewson [14]. His paper provides a thorough discussion of the mathematics involved; here, we provide a brief overview of the fundamental aspects and in particular discuss its extension to 3D.

4.1 Objective Detection of 2D Feature Candidates

Hewson's [14] 2D method is based on detecting the "boundaries" of regions of strong gradient magnitude in a thermal parameter τ, in

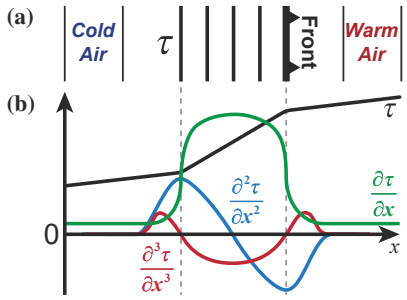


Fig. 3. Simplified 1D illustration of front detection based on a thermal parameter τ. **(a)** Schematic contour lines of τ, increasing to the right, with an increased thermal gradient in the middle. **(b)** Profiles of τ (black) and its derivatives (color) along a straight line through the field. The front, i.e., the warm air side of the region of increased thermal gradient, is detected as a minimum in the curvature (blue) of the thermal field.

his paper taken to be the wet bulb potential temperature θ_w. Fig. 3 illustrates the approach by assuming a simple 1D thermal gradient and a straight front geometry. In the example, τ increases linearly from left to right, the frontal line is located at the warm boundary of the frontal zone (i.e., the zone of high gradient magnitude of τ) shown in the middle. In this case, detection of the front location is a simple 1D problem. The frontal zone is bounded by the positions at which the thermal gradient magnitude, in this case $M_\tau = |\partial\tau/\partial x|$, changes most rapidly. That is, the extremal points in the gradient of M_τ, or where the third derivative of τ equals zero. In the example, the warm air side of the frontal zone, i.e., the front, is represented by the extremal point at which the thermal curvature $\partial^2\tau/\partial x^2$ is minimal.

In the general 2D case, fronts are curved and possess an along-front thermal gradient [14]. For this case, Hewson [14] formulates the "front locating equation" [L] as

$$L_\tau \equiv \frac{\partial(|\nabla_h|\nabla_h\tau||)_s}{\partial s} = 0, \quad with\ \hat{s} = \pm\frac{\nabla_h|\nabla_h\tau|}{|\nabla_h|\nabla_h\tau||} \tag{1}$$

where ∇_h denotes the horizontal gradient and $\hat{s}$ represents a unit axis (i.e., possessing only an orientation instead of a direction as a unit vector would) oriented along the gradient of the thermal gradient magnitude $M_\tau = |\nabla_h\tau|$. That is, essentially the ridge lines in the 2D height field represented by $|\nabla_h M_\tau|$ are sought. While classical ridge line detection [8] requires even a fourth-order derivative on τ and evaluation of the Hessian matrix, a simpler computational scheme was suggested by Hewson [14]. In short, his scheme derives a *five-point mean axis* $\hat{s}$ from the horizontal grid points surrounding a considered grid point, then evaluates the locating equation [L] by means of computing the "along-vector divergence" of $\nabla_h M_\tau$. Hewson [14] shows that the vector field $\nabla_h M_\tau$ possesses zero divergence along the frontal line; the notion of "along-vector divergence" is introduced for numerical stability, here simply all vectors at grid points employed for computation of the divergence are resolved into the direction of $\hat{s}$. Due to space limitations, we refer to Hewson [14] for a thorough description. In our approach, we follow the Hewson scheme to achieve consistency with 2D products that are operationally produced at ECMWF (cf. Fig. 1). Fig. 4a shows 2D feature candidates obtained for the idealized aquaplanet case.

4.2 Extension to 3D

The critical question that arises when raising the approach to three dimensions is whether vertical contributions to the gradient need to be considered. The meteorological definitions reviewed in Sect. 2 clearly only consider the horizontal gradient of τ. As discussed there this is primarily to avoid mis-representation of non-frontal features as fronts. Inspection of a vertical section through a frontal zone (Fig. 5a) may suggest that the locating equation [L] should be evaluated in the direction of a three-dimensional axis $\hat{s}$. Note, however, that in all depictions the vertical scale is massively exaggerated, with horizontal gradient magnitudes $\nabla_h M_\tau$ being approximately three orders of magnitude smaller than 3D gradients (Fig. 5b and c), which would again result in significantly different (often non-frontal) features being detected if a 3D gradient were used. We hence use only horizontal derivatives in the computation of our frontal features. A problem we encountered with this approach, however, is that some frontal surfaces may exhibit "holes" that appear in cases of locally reduced horizontal gradients of M_τ (an example is arrowed in Fig. 5). In the case shown, the definition of a front as a horizontal boundary between two air masses is not fulfilled. However, intuition may suggest that nevertheless a frontal surface should be drawn in this case; an issue that may require further investigations and possible re-assessments of the employed frontal definitions in the future.

Our approach to detect 3D candidates is hence: (a) At each grid point of the data, the locating equation [L] is evaluated as proposed by Hewson [14] to obtain a 3D scalar field of [L]. (b) 3D contouring methods (ray-casting, e.g., [9], or Marching Cubes [25]) are used to obtain raw candidate features that subsequently need to be filtered to obtain the desired frontal features. Fig. 4b shows the candidate surface features thus obtained.

4.3 Filtering of 3D Feature Candidates

The obtained candidate features need to be filtered to obtain those that actually represent frontal surfaces. Hewson [14] filters 2D candidates by means of two distinguishing characteristics: the curvature of the thermal parameter field, and the magnitude of the thermal gradient in the vicinity of the candidate feature as a measure of the frontal strength.

To obtain features at the warm-air side boundary of the frontal zone, all candidate features at the cold-side boundary are removed by keeping only the set of candidate vertices at which the curvature is negative. For this purpose, Renard and Clarke [39] introduced the "thermal front parameter" (TFP) as a negated curvature:

$$TFP_\tau \equiv -\nabla_h|\nabla_h\tau| \cdot \frac{\nabla_h\tau}{|\nabla_h\tau|} > K_1 \tag{2}$$

where K_1 is a user-defined threshold; it is required to be at least zero to obtain features on the warm air side of the frontal zone (cf. the blue line in Fig. 3 and note the negation). Hewson [14] sets K_1 to a small positive value to eliminate spurious features (criterion [M1] in his paper).

To require detected fronts to represent a minimum specified strength in terms of thermal gradient, Hewson [14] uses an additional filter criterion that estimates frontal strength by approximating the local magnitude of the gradient of τ on the cold side of the considered candidate feature point. Candidates are eliminated if they do not fulfill

$$S_\tau \equiv |\nabla_h\tau| > K_2 \tag{3}$$

where K_2 again is a user-defined threshold (criterion [M2] in [14]). In the approach by Hewson [14], values K_1 and K_2 are found specific to the employed NWP data and vertical levels. Candidate features are filtered "hard", i.e., all candidate points that do not meet the specified criteria are eliminated entirely. Fig. 4c illustrates the 2D case.

For detection of 3D frontal features, these filter criteria pose several disadvantages. First, estimation of frontal strength based on thermal gradient magnitude near the candidate feature is not necessarily representative of the gradient inside the frontal zone. Second, replacing a "hard" by a "soft" filtering, i.e., not discarding candidates completely but mapping the criterion to opacity, can help to smoothly "fade", e.g., regions of strong frontal strength to regions of weak strength, thus keeping additional information in the visualization. Third, suitable values of K_1 and K_2 may depend on the data or the specific structure in which the user is interested. In an interactive application as envisaged in our work, interactive adjustment is hence required. Fourth, filtering can be improved by considering further characteristics of the frontal zone. For instance, Hewson and Titley [15] suggested to eliminate fronts resulting primarily from moisture and not temperature gradients by adding a frontal strength criterion based on (*dry*) potential temperature.

Our approach improves upon all mentioned aspects. Contrary to estimating the strength of a front by evaluating Eq. 3 only by local

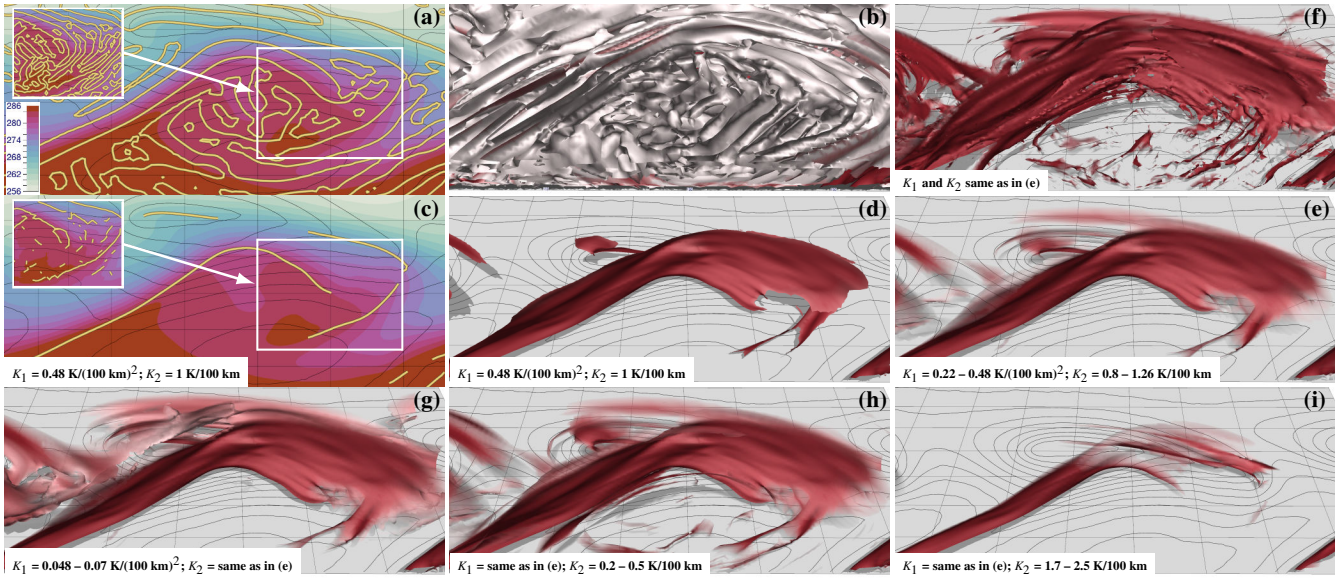


Fig. 4. 2D and 3D objective detection and filtering of front surface features (idealized aquaplanet dataset). Note non-sequential order of images. **(a)** Horizontal section at 925 hPa, showing θ_w (K, color), geopotential height (black contours), and 2D feature candidates (following [14], yellow lines). Gaussian smoothing with a distance of 100 km has been applied to θ_w. Inset compares detected features candidates if no smoothing is applied. **(b)** 3D feature candidates for the same case. **(c)** "Hard" filtering with the listed thresholds for K_1 and K_2 applied to the feature candidates in (a). Inset compares the unsmoothed field as in (a). **(d)** "Hard" filtering applied to the 3D feature candidates in (b). Black contours shows surface pressure. **(e)** As (d) but with "soft" filtered features. The listed range of K_1 and K_2 has been linearly mapped to opacity $[0-1]$. **(f)** The same as (e) but with decreased smoothing of θ_w (25 km compared to 100 km in (e)). Note the increased number of small-scale features. **(g-i)** Same as (e) but with different settings of the filter values K_1 and K_2 to demonstrate the sensitivity of detected front surfaces on these values. Detected features become larger for relaxed (g,h) filter values; more restrictive values allow focus on the stronger parts of a front (i).

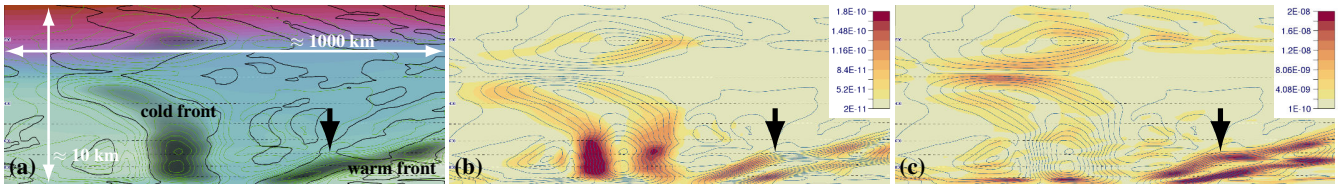


Fig. 5. Considering vertical contributions of derivatives to identify the boundaries of the zone of high thermal gradient leads to undesired features. Shown are vertical sections of **(a)** θ_w (K, color as in Fig. 4a), green contours show thermal gradient $M_\tau = |\nabla_h \tau|$, regions of large M_τ are shaded in black. Black contours denote $L_\tau = 0$, i.e., sections through the feature candidates. Note the apparent "hole" in the warm front (arrow), where the L_τ contour moves upward. **(b)** Magnitude of horizontal-only gradient $|\nabla_h |M_\tau||$ (K m^{-2}, color), blue contours show M_τ. The feature candidates in (a) are essentially ridges in $|\nabla_h |M_\tau||$. In the region of the "hole" $|\nabla_h |M_\tau||$ is weak. **(c)** When using the full 3D gradient $|\nabla |M_\tau||$, the "hole" is "filled", however, fundamentally different features are detected. E.g., the cold front (cf. (a)) vanishes. Note the much stronger gradient magnitude compared to (b).

approximation, we define as an estimate of frontal strength the *average* thermal gradient along a curved path through the frontal zone from the warm to the cold-air side. Here, we apply the concept of "normal curves" to traverse the frontal zone. Normal curves are traced through a scalar field following its gradient direction. They were used by Paffelmoser et al. [34] to measure the spatial distance between two isosurfaces, Rautenhaus et al. [38] used them to visualize the interior structure of isosurfaces. Fig. 6a illustrates the approach. A "straight line normal" can not always represent a path across the breadth of the frontal zone, particularly if a front is highly curved as is often the case when it is particularly active. A normal curve, on the other hand, runs everywhere parallel to the thermal gradient that we are interested in averaging across the zone. Again, only horizontal gradients are considered as by definition (cf. Sect. 2.1) the horizontal breadth of the zone is the quantity of interest, so all normal curves lie fully within "horizontal" planes (that is planes that accord with the vertical axis definition). Eq. 3 then becomes

$$S_\tau|_{\text{frontal zone}} \equiv \int_{NC} |\nabla_h \tau| ds > K_2 \qquad (4)$$

Traversal is stopped upon hitting the "cold-side" boundary, where $L_\tau = 0$ (L_τ is always negative within the frontal zone; cf. Fig. 3).

To construct the frontal surface visualizations, we implemented two rendering approaches. A ray-casting-based approach performs all filtering computations including normal curve integration on the GPU upon hitting a candidate surface (i.e., where $L_\tau = 0$); filtering, color- and opacity-mapping is performed per-pixel. A polygonal approach extracts surface mesh geometry using Marching Cubes, and performs filtering, color- and opacity-mapping per vertex. Rendering employs order-independent transparency. The approaches differ in particular with respect to performance. While the ray-casting approach requires more rendering time (for the cases considered here up to a few seconds on an NVidia Geforce GTX 970), the polygonal approach requires pre-processing time on the order of 10 seconds but subsequently facilitates adjustment of filtering criteria at interactive framerates.

Fig. 4d and e illustrates the difference of "hard" versus "soft" filtering for 3D frontal surfaces. "Soft" filtering is implemented with transfer functions that map K_1 and K_2 to opacity. This facilitates an interactive, user-guided filtering process (for polygonal rendering); the mapping of K_i to opacity can be quickly adjusted, and thus the sensitivity of the detected surfaces on, e.g., frontal strength, estimated. Fig. 4g-i shows

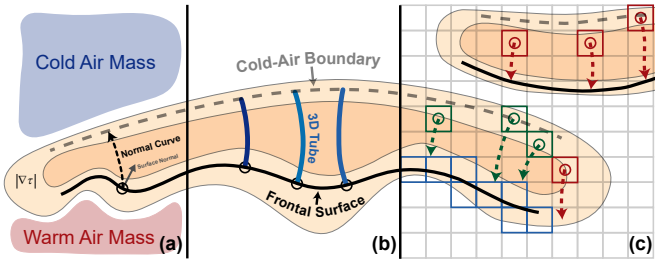


Fig. 6. Normal curves are used for multiple aspects of our approach. Shown is a schematic region of high thermal gradient (orange colors), with a frontal surface detected on its warm air side (thick black line). **(a)** Normal curves (dashed arrow) cast from the frontal surface through the frontal zone are used to obtain curve-integrated quantities including frontal strength. **(b)** For depiction of the frontal zone, normal curves are rendered as tubes. **(c)** A candidate grid cell belongs to the frontal zone associated with a selected frontal surface (cf. Fig. 8) if a "backward" normal curve started from the cell center intersects with the selected frontal surface (green cells and curves). Cells with curves intersecting with none or a different surface are discarded (red cells and curves).

examples of how the detected surfaces change when the transfer functions are changed. Hewson [14], specific to the data used in his study, suggests values for $K_1 = 0.3$ K/(100 km)^2 and $K_2 = 1.35$ K/100 km; in Fig. 4, values are of similar order but vary as noted.

4.4 Data Smoothing

Increasing horizontal resolution of NWP models (cf. [36]) has in recent years enabled the models to resolve increasingly smaller details of the atmosphere. While this for many application is a valuable property, for the detection of synoptic-scale fronts removal of small-scale gradients and variability in the data is desirable to obtain smooth features that represent the large-scale features well. Jenkner et al. [17], as well as others, used a simple smoothing filter assuming equally distributed grid points in terms of geometric distance. To account for grid points in a regular latitude–longitude grid to be closer together near the poles, we instead use a 2D Gaussian kernel to smooth, under consideration of all surrounding grid points, with a user-defined geometric distance. This approach removes sensitivity of our method to the grid resolution of the underlying data; instead, sensitivity to the smoothing distance is present. Fig. 4e and f illustrates the sensitivity of detected 3D features to changing the standard deviation of the kernel smoothing the θ_w field (Fig. 4a and b for 2D); as expected larger smoothing distances eliminate small-scale features. Typical length scales for the fronts of interest in this study are on the order of 100 to 1000 km (cf. Sect. 2.1 and Fig. 2), smoothing should be performed accordingly. For the cases presented in this paper, we selected a standard deviation of 100 km. Note that smoothing affects the settings of the filtering parameters K_1 and K_2, since with increased smoothing the thermal gradients are weakened.

4.5 Visualization of Front Surfaces and Related Processes

The detected frontal surfaces are rendered in Met.3D [38] and can be combined with existing displays of the system, e.g., horizontal maps and vertical cross-sections. Properties of the front and surrounding atmosphere (e.g., frontal strength and atmospheric variables interpolated to the frontal position) can be color-mapped onto the surfaces (cf. the tasks in Sect. 3). Fig. 7 provides examples. Note the use of shadows and movable vertical axes (Fig. 7a) to improve spatial perception, and combination with movable 2D sections through the atmosphere (Fig. 7c; cf. [38] for these functions). Color scales use the perceptional linear HCL color space [54] to conform with common visualization functionality in Met.3D (e.g., cf. the case study in [37]). Fronts are classified as cold or warm fronts, depending on whether the local wind advects temperature change towards cold or warm air, following Hewson [14]: $A_\tau = -V \cdot \nabla_h \tau$, where V denotes horizontal wind; Fig. 8a shows an example. Furthermore, our approach facilitates computation and display of the *frontal slope*, important to study, e.g., relationships between surface topography and precipitation (cf. [22]). Combination with visualizations of further atmospheric features including air parcel trajectories [37] and jet-stream core lines [19] is also readily available. The latter will be demonstrated in the Sect. 6.

5 Analysis of Frontal Zones and Individual Fronts

In addition to visualization of frontal surfaces, another unique feature of our approach is the interactive visual analysis of frontal zones associated with a frontal surface. The proposed techniques enable users to analyze the horizontal breadth of a frontal zone, as well as the distribution of atmospheric variables within (cf. task (c) in Sect. 3). To reduce clutter and to facilitate examination of an individual feature of interest, single front features can be selected and isolated.

5.1 Normal-Curve-Based Visualization

As described in Sect. 4.3, we evaluate frontal strength at a frontal surface point by integrating along a ' "normal curve" in the thermal gradient field. It is straightforward to use this approach to obtain further frontal-zone-averaged quantities, e.g., humidity. Values obtained can be color-mapped onto the frontal surface to display properties of the zone associated with the surface. Also, curve length can be used to provide color-coding of the zone's horizontal breadth.

Normal curves also offer a way to directly depict the frontal zone and the structure of its gradient field. Fig. 6b illustrates the concept, showing a set of curves that start on the frontal surface as 3D tubes. We follow Rautenhaus et al. [38] to generate seed points by computing the intersections between rays parallel to the coordinate axes and the frontal surface (see [38, Sect. 4.4] for details). For our application, an additional check is required to account for the filter criteria (cf. Sect. 4.3); potential seed points at which the identified front surface has been made transparent by the filtering transfer functions are eliminated. The density of displayed curves can be controlled by the user (cf. [38]).

Fig. 7b and c shows an example. The normal curves act as an indicator of the breadth of the frontal zone. Simultaneously, the structure of the thermal gradient field inside the zone is conveyed by the shape of the curves (cf. [38, Sect. 3.4]). Rendering the normal curves as 3D tubes also facilitates color-coding of scalar measures including gradient magnitude itself (cf. Fig. 7b and c) or arbitrary atmospheric quantities.

5.2 Selection of Individual Fronts

The simultaneous visualization of all front/frontal zone features detected in a scene can result in cluttered displays. We hence propose a mechanism to select and isolate a single front feature of interest for inspection – see Fig. 8. Picking is realized by casting a ray from the virtual camera into the scene through the selected pixel, and computing the first intersection of that ray with a computed frontal surface. At the intersection point, we determine the eight surrounding grid points (i.e., a "root" grid cell) in the 3D data grid, extract the local geometry of the picked surface using Marching Cubes on the front locator L_τ, and compute the front type (cold or warm, cf. Sect. 4.5). From the root cell, region-growing is used to iteratively identify all grid cells and front geometry that belong to the picked surface. To add a neighboring cell or surface triangle, we require adjacent triangles to form a watertight mesh and to belong to the same front type. Also, filtering criteria have to be observed; parts of the surface that are transparent in the rendered representation of the front are discarded. The approach is continued until no further cells need to be evaluated.

5.3 Statistical Properties of a Frontal Zone

To query statistical properties of a frontal zone (e.g., the distribution of an atmospheric parameter within the zone), a volumetric representation of the zone is required (i.e., a list of all grid cells that are part of the zone). The geometric representation of a selected frontal surface obtained in Sect. 5.2 can be used to approximately determine the volume that is enclosed by the corresponding frontal zone. Fig. 6c illustrates the approach: First, all grid cells intersected by the frontal surface are determined (blue cells in Fig. 6c). Next, a bounding box enclosing these cells is generated, which subsequently is enlarged to ensure that surrounding grid cells that may belong to the frontal zone are included.

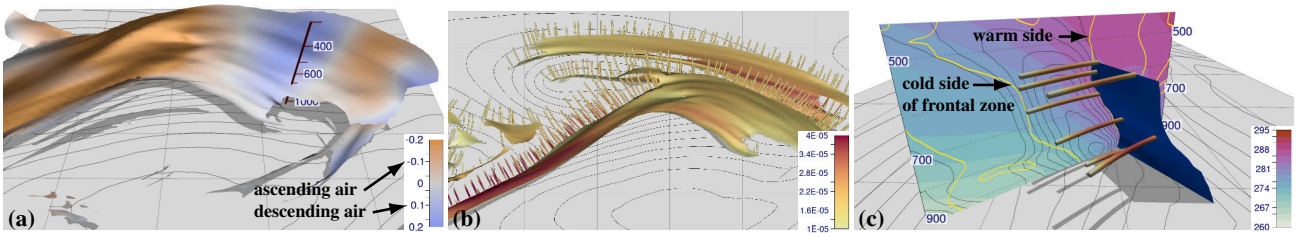


Fig. 7. Visualization of front parameters and frontal zone. **(a)** Vertical velocity in (Pa s^{-1}) as an example of an atmospheric quantity color-mapped onto the frontal surface. Note the ascending motion in particular along the left part of the front (negative values indicate ascent). **(b)** Normal curves rendered as tubes to visualize the frontal zone. Curves are color-coded with the magnitude of the thermal gradient. Black contour lines represent surface pressure. **(c)** Vertical section showing θ_w (K, color), thermal gradient (black contour lines) and the zero-isocontours (yellow) of the locator field, which bound potential frontal zones. Note how the normal curves extend horizontally from the frontal surface throughout the frontal zone; integration is stopped upon hitting the locator "contour" on the cold side of the zone.

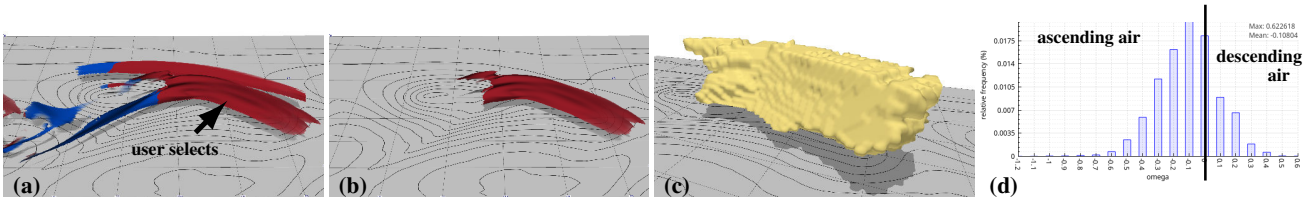


Fig. 8. Interactive selection of a front enables the user to visualize and analyze features of that particular front on its own. **(a)** The fronts are colored by their classification as cold (blue) and warm (red) front. The user selects the warm front, which **(b)** appears isolated. **(c)** Frontal zone detection (cf. Fig. 6c) identifies the displayed volume as the frontal zone of the selected warm front. **(d)** Distribution of atmospheric quantities in the frontal zone can be displayed by means of a histogram. The example shows vertical wind velocity ω (Pa s^{-1}).

Enlargement is done via a user-defined distance large enough to cover the scales of interest (cf. Fig. 2, for the study at hand features of sizes of order 100 km to 1000 km are of interest, a value we found suitable for the cases investigated here is 300 km). For each grid cell in the bounding box, we evaluate L_τ (Eq. 1) and keep those within the thermal gradient zone ($L_\tau < 0$) and those that intersect with the cold-side boundary ($L_\tau = 0$) as potential candidates. Since candidate cells in the bounding box can also belong to a different front feature in the vicinity of the selected front, correspondence of the candidate cell with the frontal zone associated with the selected surface needs to be confirmed. Here, "backward" normal curves prove useful. As shown in Fig. 6c, a normal curve is traversed from the center of each candidate grid cell into the direction of the warm-side air (i.e., towards the frontal surface). If the normal curve intersects the frontal surface, the candidate grid cell belongs to the frontal zone of interest. Curves not intersecting the frontal surface are terminated as soon as they leave the bounding box.

Fig. 8c shows the volume thus identified for the selected warm front in Fig. 8b. It is now straightforward to compute, e.g., histograms of data values within the frontal zone. Fig. 8d shows an example: A distribution of vertical velocity has been queried, revealing prevailing upward motion (i.e., negative velocities) in the frontal zone.

6 Results: Case Study and User Feedback

To demonstrate the value of our method, this section discusses first investigations of the 3D frontal structure of two mid-latitude cyclones that occurred during the 2016 North Atlantic Waveguide and Downstream Impact Experiment (NAWDEX, [45]), an atmospheric research field campaign in which two of the authors were involved. The analysis of the 3D structure of the observed cyclones is a major focus of the –at the time of writing ongoing– data analysis activities of NAWDEX (cf. [45]). Here, we consider the systems "Vladiana" and "Walpurga" that both crossed the North Atlantic in late September 2016. We use analysis data (i.e., the initial conditions of subsequent forecasts and thus the "best estimate" of the atmospheric situation at the considered time) obtained from ECMWF; all figures are produced from data on a regular latitude–longitude grid with a grid spacing of 0.5°, using 137 terrain-following model levels in the vertical. Gaussian smoothing has been applied to θ_w with a horizontal standard deviation of 100 km.

6.1 Cyclone "Vladiana"

Cyclone Vladiana preceded the extratropical transition of Tropical Cyclone "Karl", aspects of which were described by Kern et al. [19]. Our focus is on 00:00 UTC 23 September 2016, a time at which, after rapid intensification, Vladiana had evolved to maturity. Our initial objective is to compare its structure with idealized conceptual models and to investigate structural details of the fronts. Next, we show how the visualized 3D fronts facilitate investigation of the cyclone structure in a way not possible with classical front analysis at single levels.

Fig. 1 shows the operational objectively detected 2D fronts from ECMWF (Fig. 1a) and the 3D fronts from our method; Fig. 9a shows the detected 3D fronts classified as cold and warm fronts. As expected, the cold and warm 3D fronts show good agreement with the ECMWF surface fronts, but 3D visualization adds valuable insight into vertical extent and structure. The warm front appears as a downstream tilted, curved feature (red colors) on the eastern flank of the cyclone that separates northeastward moving warm air southwest of the front from westward moving cold air to its north. The westernmost part of this frontal surface is classified as a cold front, due to northerly winds on the rear side of the cyclone bringing cold air southward. At the cyclone center the lateral position of the meeting point of cold and warm fronts varies only slightly in the vertical, suggesting little tilt with height in the cyclone's frontal signature, which is probably symptomatic of a mature cyclone. Such aspects cannot be seen on a 2D front chart. Almost perpendicular to the warm front, the main cold front (blue colored) extends southwestward towards the US east coast. Several small red areas (detected warm front character) within the large cold front appear when the predominately southwesterly flow that crosses the frontal surface at small angles locally advects warm air in the direction of the cold air. When near the earth's surface these signify frontal waves or anti-waves [14] – note the orange spots on Fig. 1a. When higher up they denote upper-level-only equivalents of these [2]. In both cases a change in surface weather can be associated.

Fig. 1b and c and Fig. 9b and c show selected key meteorological parameters color-mapped onto the detected fronts, providing insight into atmospheric processes along the frontal features that could not be inferred from the existing 2D approaches. Pressure elevation (Fig. 1c) at a glance shows that the cold front as well as the northern part of the

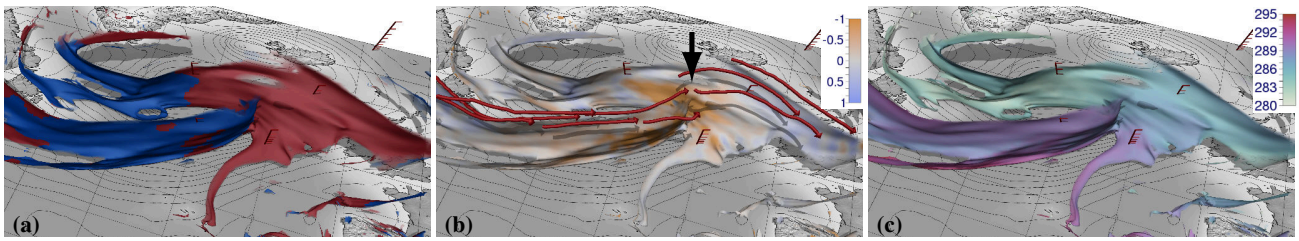


Fig. 9. Fronts of cyclone "Vladiana", at 00:00 UTC 23 September 2016. **(a)** Cold (blue) and warm (red) fronts. **(b)** Vertical wind velocity at the frontal surface (hPa s^{-1}, negative values denote ascent). Note stronger ascent near the triple point (arrow). Red lines show jet-stream core lines detected as in Kern et al. [19]. **(c)** Wet bulb potential temperature θ_w (K). Note almost constant values on each frontal surface.

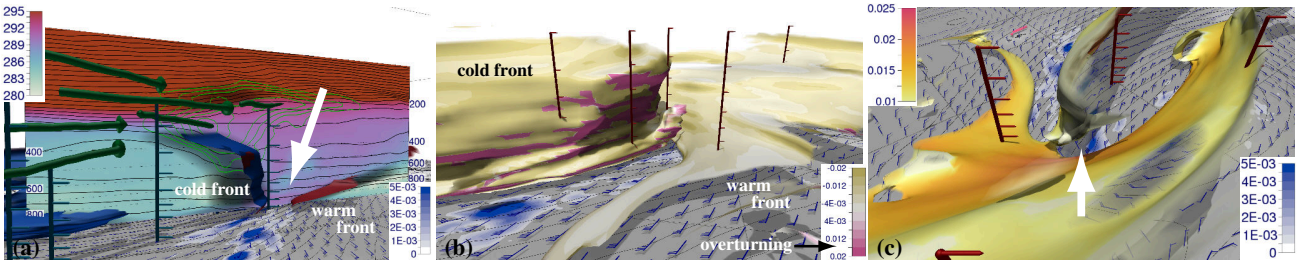


Fig. 10. Details of the fronts of "Vladiana", at 00:00 UTC 23 September 2016. **(a)** Vertical section through cold and warm front, showing θ_w (K, color), the θ_w gradient (black shading), contours of potential temperature (grey) and wind speed (green) and jet-stream core lines as in Fig. 9b. Blue ground surface shading shows precipitation (m h^{-1}). The arrow points to the warm sector. Note the jet-stream above the cold front. **(b)** Color mapping of frontal slope (ratio vertical to horizontal). Note the larger slopes for the cold front than for the warm front. The cold front even exhibits some overturning (lilac). **(c)** Looking southwestwards; "frontal fracture" between cold and warm front extends through much of the troposphere (arrow).

warm front are deep features that extend into the upper troposphere (above 400 hPa). The southern part of the warm front, however, appears vertically shallow at lower levels. Air masses tend to be displaced vertically in the vicinity of the frontal zones, as the fronts themselves propagate. Often vertical velocity magnitude reflects frontal slope [22]. Indeed here we generally see higher vertical velocities where there are steeper slopes, such as near the triple point (Fig. 9b). The ascent adiabatically cools the air until saturation is reached and clouds and precipitation form. Vertical velocities are also an atmospheric response to broadscale dynamical forcing; for example the "left exit" and "right entrance" regions beneath upper level jets commonly have large upward motion associated. By adding jet cores to such plots the interplay between features can be demonstrated in new ways (Fig. 9b). The conservation of wet bulb potential temperature θ_w is confirmed by almost constant values on the various frontal surfaces, shown in Fig. 9c. The different colors on the different fronts show that the warm airmass associated with each has a different intrinsic θ_w.

Fig. 10a shows a front-crossing vertical section of θ_w, along with the detected cold and warm fronts. Vladiana's warm sector (arrowed) appears clearly as an air mass with increased θ_w. Atmospheric dynamics tend to mean that zones of strong horizontal temperature gradients are accompanied by vertically increasing wind speeds, leading to high level jet streams (cf. [19]). The depiction in Fig. 10a shows jet cores above the cold front (cf. [52]).

The Norwegian conceptual model [4] describes the slope of typical cold fronts to be steeper (about 1 km vertical rise over a horizontal distance of 100 km) than that of typical warm fronts (1 km over 300 km). This idealized picture is generally reflected in Fig. 10b. However, we observe a large variability in frontal slope in particular in the cold front, where overturning is present locally. This may relate to local instability causing enhanced convective precipitation (cf. [22]). An investigation of vertical instability along frontal structures is one of the scientific aims of NAWDEX; future work will employ our method to improve interpretation of campaign observations in this regard.

Our 3D visualizations provide an effective means for comparison of cyclone types with conceptual models and associated cyclone characteristics. For example, the Norwegian and Shapiro-Keyser conceptual models differ with respect to the occlusion process, i.e., the merging of cold and warm fronts (cf. [49]). The Norwegian model describes the cold front to "catch up" with the warm front; a "triple point" is formed with an elevated occluded front extending to the cyclone center. In contrast, Shapiro and Keyser [52] describe a "frontal fracture" with the cold front being detached and almost perpendicular to a strong warm front north of it. Fig. 10c indicates very clearly a detachment, through depth, of the cold and warm fronts, which seems to suggests a Norwegian cyclone. However, care is needed in interpretation as at the base of an occlusion warm air has conceptually similar characteristics to the ubiquitous anticyclonic inversions, which we have tried to eliminate from our analysis. This may be an area for future work.

To gain further insight into the characteristics of the cold front, it is selected and the frontal zone is visualized using normal curves (Fig. 11a and b). Fig. 11b shows vertical velocity mapped to normal curves. As expected from Fig. 9b, ascent is represented close to the frontal surface. Our depiction shows how far this ascent extends horizontally into the frontal zone; on the back side descent is present, whilst closer to the cyclone center (left side of the figure), descent is stronger and closer to the frontal surface. The different breadths of the regions of ascent may relate to different breadths of rainbands. Frontal zone statistics become useful to obtain an estimate of the θ_w distribution within. As visible in Fig. 11c, θ_w values are smaller than the values mapped onto the front in Fig. 9c, as expected from the front definition.

6.2 Cyclone "Walpurga"

Cyclone Walpurga, the fifth system in September 2016 observed during NAWDEX, had reached southern Scandinavia at 12:00 UTC 29 September 2016. Its long cold front extended from southern Sweden over Denmark, northwestern France to the Azores. Fig. 12 shows the operational 2D fronts product from ECMWF, along with our 3D visualization. We consider an interesting feature that immediately is visible to the user in the 3D depiction: In the southernmost part of the front, the ECMWF detection shows a secondary cold front upstream of the cold front, seemingly detached and independent from the latter. The 3D depiction, however, shows that both fronts are actually connected at upper levels in the east and belong to the same structure. The primary front appears

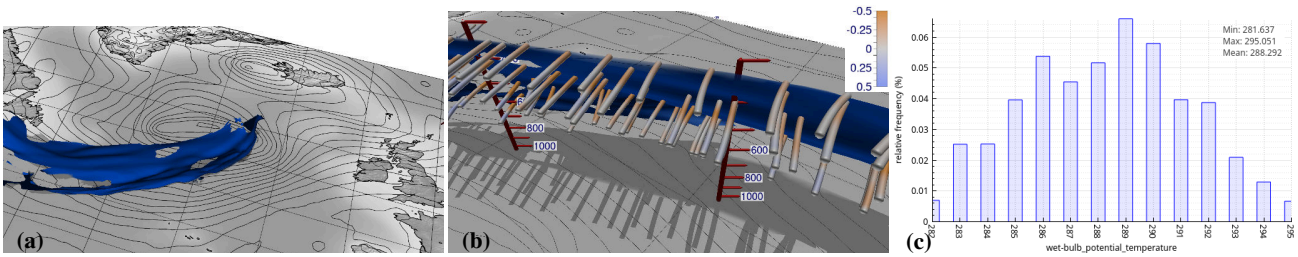


Fig. 11. The main cold front of "Vladiana" is investigated. **(a)** Selection of the cold front isolates the feature. **(b)** Normal curves colored by vertical velocity show ascending air close to the frontal surface (cf. Fig. 9b); at the rear side of the frontal zone descending air prevails. **(c)** Distribution of θ_w in the frontal zone. As expected, the values depicted on the frontal surface in Fig. 9c are on the warm side of the distribution.

to be very shallow (mostly below 900 hPa) whereas the trailing front is a deep feature vertically extending into the mid troposphere. This may be a case of a "double front" system as recently discussed by Mulqueen and Schultz [32]. While Mulqueen and Schultz only considered 2D surface maps, our method provides a means for a thorough systematic investigation of multiple-front cyclones. Only a full 3D perspective can clarify whether double fronts observed at the surface are individual frontal systems or interconnected.

6.3 Domain Expert Feedback

Domain experts, active in weather forecasting and meteorological research, were provided with some static 3D front images from this paper, and were asked to provide feedback; about 20 replied, from Europe and North America. In regard to potential use as a research tool the response overall was very positive (one said "this is really amazing!"). For real-time forecasting applications the response was also positive, though a few were concerned about speed of operation and usability. Meteorological training and media outlets were also mentioned as potential application areas.

Many indicated that tailored 3D front products could assist with aviation forecasting, specifically for major hazards like icing and "embedded convection", that usually occur within frontal cloud. Also cited were turbulence prediction, the "sting jet" extreme wind phenomena and snowbands. In research and forecasting many were interested in frontal slope, and 3D front characterization (ana- and kata-, frontogenetic or frontolytic, over-running and split), all of which can now be highlighted. Replies from research referred also to combining the 3D fronts with other variables and visualizations, such as front-normal winds, microphysical quantities, trajectories and vertical profiles. Interestingly, reactions to some plots varied markedly; one found Fig. 12 illuminating whilst another found it confusing. Evidently familiarization will be important.

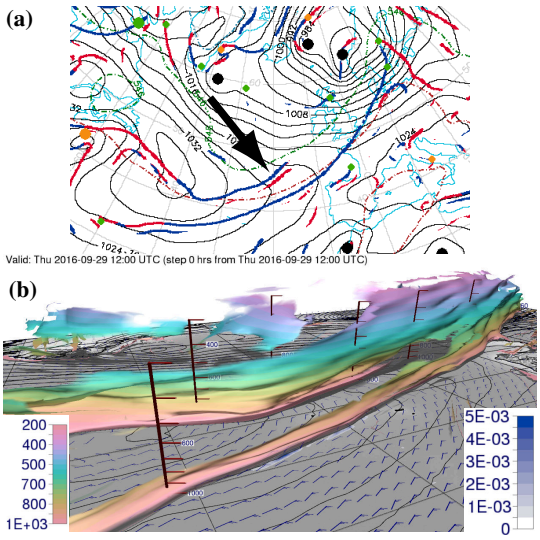


Fig. 12. Fronts of cyclone "Walpurga", at 12:00 UTC 29 September 2016. **(a)** ECMWF 2D objective surface fronts product. Note the seemingly detached front indicated by the arrow. **(b)** 3D fronts identified and visualized with our method, color denotes pressure elevation (hPa). Note how the two fronts that are seemingly independent in the 2D image are actually connected at upper levels.

7 Discussion and Conclusion

We have proposed a full 3D identification and visualization methodology for investigating atmospheric frontal structures, which are intrinsically 3D. Such analysis was not possible before, using classical 2D techniques. Our techniques include depiction of 3D frontal surfaces, depiction of frontal zones by means of normal curves, color-mapping of atmospheric quantities onto displayed features, and combination with further atmospheric fields and features. Frontal features of interest can be isolated interactively, and information about the distribution of atmospheric quantities within a frontal structure can be queried.

We briefly discuss our results. Our technique builds upon an established 2D method [14, 15] that has been run in operational environments for years, thereby achieving consistency with existing products. At the same time this might be considered a limitation, as somewhat different views exist within the meteorological community regarding how fronts should be defined (e.g., [42, 49]). Nonetheless, our portrayed structures do make physical sense, they do conform, to a large degree, with conventional ideas, and they do provide, in addition, many new insights. Two real-world cases from the 2016 NAWDEX atmospheric field campaign have highlighted the value of our method for meteorological analysis and shown in one case (Walpurga) a structural feature not previously documented. With respect to lessons learned during the design stage, we note that in particular the use of transparency for "soft" filtering was much appreciated by the domain experts in the author team (after discovering the strong sensitivity of "hard" filtered fronts to thresholds). Also, the isolated depiction of individual fronts proved very beneficial to reduce cluttering and to permit focus.

In conclusion, we are confident that our method will facilitate many new and valuable studies in atmospheric research, and that it will also benefit operational forecasting, particularly for adverse weather which often relates to fronts. Newly identified frontal characteristics will be subject to further investigations in the context of NAWDEX and beyond. This will include evaluation of traditional ideas by comparison of conceptual models and observations to the multiple 3D structures identified by our approach. We are confident this will stimulate further meteorological research to help reconcile different views on fronts, and to improve representation of frontal processes in numerical models.

Acknowledgments

The research leading to these results has been done within the subproject "Visualization of coherence and variation in meteorological dynamics" of the Transregional Collaborative Research Center SFB/TRR 165 "Waves to Weather" funded by the German Research Foundation (DFG). We are grateful to Sophia Schäfer for providing the "aquaplanet" dataset. ECMWF data has been kindly provided in the context of the ECMWF special project "Support Tool for HALO Missions".

REFERENCES

[1] American Meteorological Society. "front". glossary of meteorology. http://glossary.ametsoc.org/wiki/Front, 2018. Accessed 31 March 2018.

[2] M. J. Bader, G. S. Forbes, J. R. Grant, R. B. E. Lilley, and A. J. Waters. *Images in weather forecasting : a practical guide for interpreting satellite and radar imagery*. Cambridge University Press, 1995.

[3] G. Berry, M. J. Reeder, and C. Jakob. A global climatology of atmospheric fronts. *Geophys. Res. Lett.*, 38(4):L04809+, Feb. 2011.

[4] J. Bjerknes and H. Solberg. Life cycles of cyclones and the polar front theory of atmospheric circulation. *Geofys. Publ.*, 3:1–18, 1922.

[5] D. C. Breton. Note sur les lignes de faîte et de thalweg. *Comptes rendus hebdomadaires des séances de l'académie des Sciences*, 39:647–648, 1854.

[6] K. A. Browning and G. A. Monk. A simple model for the synoptic analysis of cold fronts. *Quarterly Journal of the Royal Meteorological Society*, 108(456):435–452, Apr. 1982.

[7] M. de Saint-Venant. Surfaces à plus grande pente constituées sur des lignes courbes. *Bulletin de la Société Philomathématique de Paris*, pp. 24–30, 1852.

[8] D. Eberly. *Ridges in Image and Data Analysis (Computational Imaging and Vision)*. Springer, 1996 ed., Sept. 1996.

[9] K. Engel, M. Hadwiger, J. Kniss, C. Rezk-Salama, and D. Weiskopf. *Real-Time Volume Graphics*. A K Peters, Wellesley, Mass., July 2006.

[10] J. D. Furst and S. M. Pizer. Marching ridges. In *2001 IASTED International Conference on Signal and Image Processing*, pp. 22–26, 2001.

[11] J. A. Grim, R. M. Rauber, M. K. Ramamurthy, B. F. Jewett, and M. Han. High-Resolution observations of the Trowal–Warm-frontal region of two continental winter cyclones. *Monthly Weather Review*, 135(5):1629–1646, May 2007.

[12] M. Han, S. A. Braun, Persson, and J.-W. Bao. Alongfront variability of precipitation associated with a midlatitude frontal zone: TRMM observations and MM5 simulation. *Monthly Weather Review*, 137(3):1008–1028, Mar. 2009.

[13] R. M. Haralick. Ridges and valleys on digital images. *Computer Vision, Graphics, and Image Processing*, 22(1):28 – 38, 1983.

[14] T. D. Hewson. Objective fronts. *Met. Apps*, 5(1):37–65, 1998.

[15] T. D. Hewson and H. A. Titley. Objective identification, typing and tracking of the complete life-cycles of cyclonic features at high spatial resolution. *Met. Apps*, 17(3):355–381, 2010.

[16] P. Hope, K. Keay, M. Pook, J. Catto, I. Simmonds, G. Mills, P. McIntosh, J. Risbey, and G. Berry. A comparison of automated methods of front recognition for climate studies: A case study in southwest western australia. *Monthly Weather Review*, 142(1):343–363, Jan. 2014.

[17] J. Jenkner, M. Sprenger, I. Schwenk, C. Schwierz, S. Dierer, and D. Leuenberger. Detection and climatology of fronts in a high-resolution model reanalysis over the alps. *Meteorological Applications*, 17(1):n/a, Mar. 2009.

[18] M. Kašpar. Objective frontal analysis techniques applied to Extreme/Non-extreme precipitation events. 47(3):605–631, 2003.

[19] M. Kern, T. Hewson, F. Sadlo, R. Westermann, and M. Rautenhaus. Robust detection and visualization of jet-stream core lines in atmospheric flow. *IEEE Trans. Visual. Comput. Graphics*, 24(1):893–902, 2018.

[20] G. Kindlmann. Semi-automatic generation of transfer functions for direct volume rendering. Master's thesis, Cornell University, USA, 1999.

[21] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. 30(2):117–156, 1998.

[22] J. D. Locatelli, J. E. Martin, and P. V. Hobbs. A wide cold-frontal rainband and its relationship to frontal topography. *Q.J.R. Meteorol. Soc.*, 120(516):259–275, Jan. 1994.

[23] J. D. Locatelli, M. T. Stoelinga, M. F. Garvert, and P. V. Hobbs. The IMPROVE-1 storm of 1–2 february 2001. part i: Development of a Forward-Tilted cold front and a warm occlusion. *Journal of the Atmospheric Sciences*, 62(10):3431–3455, Oct. 2005.

[24] J. D. Locatelli, M. T. Stoelinga, and P. V. Hobbs. Re-examination of the split cold front in the british isles cyclone of 17 july 1980. *Quarterly Journal of the Royal Meteorological Society*, 131(612):3167–3181, Oct. 2005.

[25] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, vol. 21 of *SIGGRAPH '87*, pp. 163–169. ACM, New York, NY, USA, July 1987.

[26] P. Majer. *A statistical approach to feature detection and scale selection in images*. Niedersächsische Staats-und Universitätsbibliothek, 2000.

[27] J. E. Martin. *Mid-Latitude Atmospheric Dynamics: A First Course*. Wiley, 1 ed., June 2006.

[28] C. F. Mass. Synoptic frontal analysis: Time for a reassessment? *Bulletin of the American Meteorological Society*, 72(3):348–363, Mar. 1991.

[29] C. F. Mass and D. M. Schultz. The structure and evolution of a simulated midlatitude cyclone over land. *Mon. Wea. Rev.*, 121(4):889–917, Apr. 1993.

[30] Met.3D contributors. Met.3D. http://met3d.wavestoweather.de, 2018. Accessed 31 March 2018.

[31] A. E. Milionis and T. D. Davies. The effect of the prevailing weather on the statistics of atmospheric temperature inversions. *International Journal of Climatology*, 28(10):1385–1397, Aug. 2008.

[32] K. C. Mulqueen and D. M. Schultz. Non-classic extratropical cyclones on met office sea-level pressure charts: double cold and warm fronts. *Weather*, 70(3):100–105, Mar. 2015.

[33] R. Peikert and F. Sadlo. Height ridge computation and filtering for visualization. In *2008 IEEE Pacific Visualization Symposium*, pp. 119–126, March 2008.

[34] T. Pfaffelmoser, M. Reitinger, and R. Westermann. Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields. *Comput. Graphics Forum*, 30(3):951–960, June 2011.

[35] J. G. Pinto, I. n. Gómara, G. Masato, H. F. Dacre, T. Woollings, and R. Caballero. Large-scale dynamics associated with clustering of extratropical cyclones affecting western europe. *Journal of Geophysical Research: Atmospheres*, 119(24):13,704–13,719, Dec. 2014.

[36] M. Rautenhaus, M. Bottinger, S. Siemen, R. Hoffman, R. M. Kirby, M. Mirzargar, N. Rober, and R. Westermann. Visualization in meteorology — a survey of techniques and tools for data analysis tasks. *IEEE Trans. Visual. Comput. Graphics*, preprint:1, 2018.

[37] M. Rautenhaus, C. M. Grams, A. Schäfler, and R. Westermann. Three-dimensional visualization of ensemble weather forecasts – Part 2: Forecasting warm conveyor belt situations for aircraft-based field campaigns. *Geosci. Model Dev.*, 8(7):2355–2377, 2015.

[38] M. Rautenhaus, M. Kern, A. Schäfler, and R. Westermann. Three-dimensional visualization of ensemble weather forecasts – Part 1: The visualization tool Met.3D (version 1.0). *Geosci. Model Dev.*, 8(7):2329–2353, 2015.

[39] R. J. Renard and L. C. Clarke. Experiments in numerical objective frontal analysis 1. *Monthly Weather Review*, 93(9):547–556, Sept. 1965.

[40] F. Sadlo and R. Peikert. Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1456–1463, Nov 2007.

[41] J. Sahner, T. Weinkauf, N. Teuber, and H. C. Hege. Vortex and strain skeletons in Eulerian and Lagrangian frames. *IEEE Transactions on Visualization and Computer Graphics*, 13(5):980–990, Sept 2007.

[42] F. Sanders. A proposed method of surface map analysis. *Monthly Weather Review*, 127(6):945–955, June 1999.

[43] F. Sanders and C. A. Doswell. A case for detailed surface analysis. *Bulletin of the American Meteorological Society*, 76(4):505–521, Apr. 1995.

[44] S. A. K. Schäfer and A. Voigt. Radiation weakens idealized midlatitude cyclones. *Geophysical Research Letters*, 45(6):2833–2841, Mar. 2018.

[45] A. Schäfler, G. Craig, H. Wernli, P. Arbogast, J. D. Doyle, R. McTaggart-Cowan, J. Methven, G. Rivière, F. Ament, M. Boettcher, M. Bramberger, Q. Cazenave, R. Cotton, S. Crewell, J. Delanoë, A. Dörnbrack, A. Ehrlich, F. Ewald, A. Fix, C. M. Grams, S. L. Gray, H. Grob, S. Groß, M. Hagen, B. Harvey, L. Hirsch, M. Jacob, T. Kölling, H. Konow, C. Lemmerz, O. Lux, L. Magnusson, B. Mayer, M. Mech, R. Moore, J. Pelon, J. Quinting, S. Rahm, M. Rapp, M. Rautenhaus, O. Reitebuch, C. A. Reynolds, H. Sodemann, T. Spengler, G. Vaughan, M. Wendisch, M. Wirth, B. Witschas, K. Wolf, and T. Zinner. The north atlantic waveguide and downstream impact experiment. *Bull. American Meteor. Soc.*, preprint, 2018.

[46] S. Schemm, L. Nisi, A. Martinov, D. Leuenberger, and O. Martius. On the link between cold fronts and hail in switzerland. *Atmos. Sci. Lett.*, 17(5):315–325, May 2016.

[47] S. Schemm and M. Sprenger. Frontal-wave cyclogenesis in the north atlantic –a climatological characterisation. *Q.J.R. Meteorol. Soc.*, 141(693):2989–3005, Oct. 2015.

[48] D. M. Schultz and C. F. Mass. The occlusion process in a midlatitude cyclone over land. *Mon. Wea. Rev.*, 121(4):918–940, Apr. 1993.

[49] D. M. Schultz and G. Vaughan. Occluded fronts and the occlusion pro-

cess: A fresh look at conventional wisdom. *Bulletin of the American Meteorological Society*, 92(4):443–466, Apr. 2011.

[50] D. M. D. Schultz and W. Blumen. *SYNOPTIC METEOROLOGY — Fronts*, pp. 337–343. Elsevier, 2015.

[51] S. C. Shadden, F. Lekien, and J. E. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3-4):271 – 304, 2005.

[52] M. A. Shapiro and D. Keyser. Fronts, jet streams, and the tropopause. In C. Newton and E. O. Holopainen, eds., *Extratropical Cyclones. The Erik Palmen Memorial Volume*, pp. 167–191. American Meteorological Society, Boston, MA, 1990.

[53] I. Simmonds, K. Keay, and J. A. Tristram Bye. Identification and climatology of southern hemisphere mobile fronts in a modern reanalysis. *J. Climate*, 25(6):1945–1962, Sept. 2011.

[54] R. Stauffer, G. J. Mayr, M. Dabernig, and A. Zeileis. Somewhere over the rainbow: How to make effective use of colors in meteorological visualizations. *Bull. Amer. Meteor. Soc.*, 96(2):203–216, Feb. 2015.

[55] R. C. Strawn, D. N. Kenwright, and J. Ahmad. Computer visualization of vortex wake systems. *AIAA journal*, 37(4):511–512, 1999.

[56] R. Stull. *Practical Meteorology: An Algebra-based Survey of Atmospheric Science*. Creative Commons License, Copyright 2017, 2018 by Roland Stull, University of British Columbia, Vancouver, BC, v1.02b ed., 2017.

Vision, Modeling, and Visualization (2019)
H.-J. Schulz, M. Teschner, and M. Wimmer (Eds.)

Clustering Ensembles of 3D Jet-Stream Core Lines

Michael Kern[1] and Rüdiger Westermann[1]

[1]Technische Universität München, Chair of Computer Graphics and Visualization, Germany

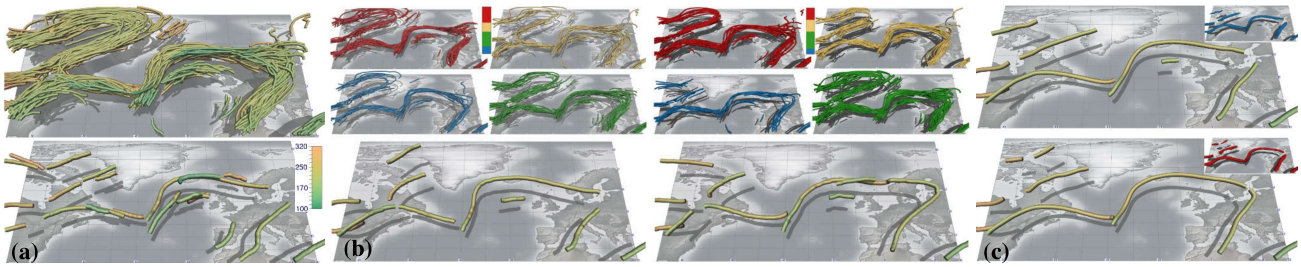


Figure 1: ***(a)*** *Top: Spaghetti plot of jet-stream core lines, extracted from numerical weather prediction data for September 28, 2016. Bottom: Ground truth of the core line on same day. Line features are colored by pressure level in hPa to infer vertical variation.* ***(b)*** *Top: 4 core line groups obtained from clustering on fields of directional wind speed derivatives (upper left image) and vector-to-closest point volumes (upper right image). Bottom: The best cluster representatives closest to the ground truth.* ***(c)*** *Ridge lines extracted from central tendency volume (inset, upper image) computed from case (a) and ridges from the visitation map (inset, lower image).*

Abstract

The extraction of a jet-stream core line in a wind field results in many disconnected line segments of arbitrary topology. In an ensemble of wind fields, these structures show high variation, coincide only partly, and almost nowhere agree in all ensemble members. In this paper, we shed light on the use of clustering for visualizing an ensemble of jet-stream core lines. Since classical approaches for clustering 3D line sets fail due to the mentioned properties, we analyze different strategies and compare them to each other: We cluster the 3D scalar fields from which jet-stream core lines are extracted. We cluster on a closest-point representation of each set of core lines. These representations are derived from the extracted line geometry and can be used independently of the lines orientation and topology. We cluster on the 3D line set using the Hausdorff distance as similarity metric. In the resulting clusters, we visualize core lines from the most representative ensemble member. We further compute ridges in a single 3D visitation map that is build from the ensemble of core lines, and we extract the most central core line from the ensemble closest-point representation. These "averages" are compared to the clustering results, and they are put into relation to ground truth jet-stream core lines at the predicted lead time.

1. Introduction

Nowadays, numerical weather forecasting is routinely performed at weather centers and research institutions. The improvement of weather forecasts depends heavily on documenting and understanding complex three-dimensional structures in the atmosphere. Therefore, these structures need to be defined and extracted from simulated physical fields. Once available, a three-dimensional (3D) visualization of these structures provides a greatly increased capacity to understand the key relationships and dependencies between multiple atmospheric processes. Beyond single forecasts, ensemble weather forecasts are well established in meteorology to understand and assess the uncertainty inherent in numerical weather predictions. Ensemble methods perform multiple simulations using perturbed initial conditions or different forecast models, to predict possible future states of the atmosphere. Analysis of the temporal evolution and variability of an ensemble forecast is then used to estimate the likelihood of certain weather events. For many important structures, i.e. line features, in the atmosphere, ensemble analysis becomes notoriously difficult. This is because such structures are often fuzzy and cluttered, consisting of many disconnected segments of arbitrary topology. An example is the jet-stream, regions of high wind speed, typically encountered near to the top of our principal weather systems. The extraction of jet-stream core lines can lead to many separate line features in 3D [KHS*18]. In an ensemble of wind fields, these features show high variation, coincide only partly, and almost nowhere agree in all ensemble members.

To determine and characterize major trends in operational routines, forecasters often actively look for some kind of mean or me-

dian solution for a certain feature in an ensemble, involving clustering features on the 2D chart in their mind if the patterns suggest so to do. We aim to automate this process and, in particular, lift it to 3D. This, however, is challenging for the type of feature we consider. So-called spaghetti plots, i.e., the simultaneous display of all ensemble members (see 1a), quickly produce visual clutter and cannot easily convey major trends, outliers, and statistical properties of the feature distribution. Statistical feature-based approaches, on the other hand, like contour and curve box-plots [WMK13; MWK14] or variability plots [FBW16] cannot be used in general for highly fragmented line sets with arbitrary topology.

In this work, we shed light on the use of clustering to determine the major trends in ensembles of jet-stream core lines with vastly different geometry and topology. We compare the results to the clustering of lines using the mean distance between any pair of core line (one in each cluster) as distance metric. For clustering, we use Agglomerative Hierarchical Clustering (AHC) with Ward's method [Jr63], which groups the initial core lines in bottom-up fashion. To prepare the data for clustering, the used fields are interpreted as high-dimensional data points and projected to lower dimension using either PCA, for derivative fields, or t-SNE, for closest point fields. The findings, in the form of the mean or most representative core line of each cluster, are put into relation to ground truth features obtained from re-analysis data that best represents the atmosphere data at the considered lead time.

In particular, we make the following contributions: First, we compare different clustering approaches and analyze their potential to convey major trends in a weather forecast ensemble to improve the predictability of certain weather situations. Second, we propose two novel representations for clustering meteorological features comprised of line sets with arbitrary topology: We cluster on the derivative fields from which features are extracted, and on a topology-invariant representation of line sets based on a closest point representation. This enables to operate on the extracted jet-stream core lines, regardless of their shape and topology. Third, we compare the results of clustering to those achieved via visitation maps, a density field showing the frequency of occurrence of line features, and via central tendency fields characterizing points most central in between sets of lines. Ridge detection in such volumes is used to extract approximate features which are then used in the comparative analysis. To demonstrate the practical relevance and usefulness of our work, we discuss the strengths and weaknesses of the methods by means of real-world scenarios using numerical weather simulation data from the European Center for Medium Weather Forecasting (ECMWF).

2. Related Work

The analysis of 3D line feature ensembles is closely related to ensemble visualization techniques. Parametric statistical distributions and distribution shape descriptors for scalar-valued ensembles were presented by Love et al. [LPK05]. Different variants of confidence regions were introduced to obtain major geometric trends in ensembles of closed isocontours and streamlines [WMK13; MWK14; FBW16; FKRW16]. Demir et al. [DJW16] proposed a closest-point representation to convey the central tendency of an multi-dimensional shape ensemble.

In a number of works, scalar- and vector-valued ensemble fields were modeled via mixtures of probability density functions to compactly classify complex distributions and their evolution over time [LLBP12; JDKW15; DS15; WLW*17]. Demir et al. [DDW14] propose to linearize 3D data points and visualize the data distributions using bar-charts. Poethkow and Hege [PH13] and Athawale et al. [ASE16] use location-wise estimators of non-parametric distributions from ensemble members to estimate the spread of surface and vector field features. Recently, Hazarika et al. [HBS18; HDSC19] presented a copula-based framework for large multivariate datasets, where they partition the domain and compute statistical quantities over those parts.

Alternatively, clustering has been used to group ensemble members, either fields or curves in such fields, regarding similar data characteristics [BM10; TN14; OLK*14; FBW16; FFST19]. The comparative studies by Zhang et al. [ZHQL16] and Moberts et al. [MVv05] provide overviews of similarity measures using geometric distances between curves. Strehl and Ghosh [SG02] apply different clustering techniques to one single ensemble, and combine the results into a single clustering. Ferstl et el. [FKRW17] cluster different time-steps of the same ensemble in a hierarchical way to convey the change of clusters over time. Clustering of atmospheric data is surveyed by Wilks [Wil11]. An example is the operational clustering of ensemble members at ECMWF [FC11], where forecast scalar fields of geopotential height are clustered in three different time windows for a static data region. Kumpf et al. [KTB*18] use multiple k-Means clusterings on ensemble fields, with slightly varying clustering domain, and analyze the robustness of clusters. Clustering using the Hausdorff metric was applied to group similar streamlines (cf. [RT12]) in flow-fields, and fiber bundles [BBP*05] in diffusion tensor imaging.

Building upon first definitions of jet-streams, e.g., [Rei63], were the "layer of maximum wind" (LMW) was defined to identify the 3D jet-stream axis representing the core line, many different methods have been developed to automatically identify jet-streams. For example, Strong and Davis [SD05] detect core lines on their LMW by computing wind speed maxima via finite differencing in the y-direction only. Molnos et al. [MMP*17] introduced a network-based scheme using shortest paths to detect a jet-stream core as a continuous, globe-circumventing line. Spensberger et al. [SSL17] consider 2D wind fields on a "dynamical tropopause", an isosurface of 2 potential vorticity units.

Aforementioned methods make a priori assumptions about certain jet-stream characteristics, such as being westerly oriented or fully connected large-scale features. Our approach builds upon the detection method by Kern et al. [KHS*18], which avoids such assumptions and detects arbitrarily oriented and possibly disconnected jet-stream core lines.

3. Data

We extract the jet-stream core lines by applying ridge detection to a given wind field, as proposed by Kern et al. [KHS*18]. Numerical weather prediction data, generated from the Numerical Weather Prediction (NWP) model at ECWMF, serve as input for feature detection. Ensemble forecasting is used to compute m predictions,

each comprised of multiple fields such as wind direction and precipitation. For jet-stream core line detection, we use the 2D wind components u, v and wind speed V_s for each vertical layer, From these inputs, we derive new fields from which the features are extracted (see eq. 1). Let $\vec{V} = (u, v)$ be the wind vector at a grid point x, and V_s be the wind speed resolved into the direction $\vec{s}$, the local wind direction at x. Let $\vec{n}$ be the vector normal to $\vec{s}$. The loci (candidates) of jet-stream core lines are defined at each grid point where V_s is maximal along $\vec{n}$ in the horizontal, and along $\vec{z}$ in the vertical:

$$\frac{\partial V_s}{\partial n} = 0, \frac{\partial V_s}{\partial z} = 0 \tag{1}$$

Upon detecting candidate lines via ridge extraction, filtering is applied to consider domain-specific constraints. In particular, domain experts look for jet-stream core lines with at least $V_s \geq 40ms^{-1}$ and large features with a feature length above $500km$. Minima lines of the wind field are removed by testing whether all eigenvectors of the 2D Hessian matrix at each line vertex are negative. Filtered ridge lines are shown in Fig. 1a and Fig. 2.

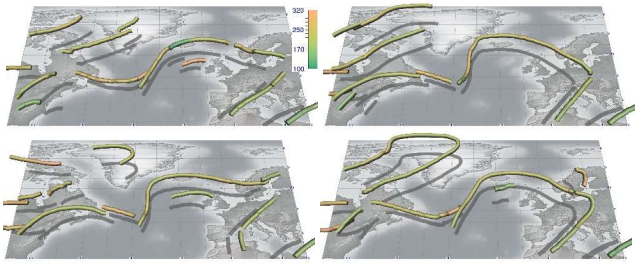


Figure 2: *Extracted jet-stream core lines rendered as 3D tubes for 4 different member of the ensemble in Fig. 1a. Images demonstrate the variation of line topology. Tubes are colored by pressure in hPa to show variation in the vertical.*

4. Clustering of Jet-Stream Core Lines

In this section, different strategies to cluster an ensemble of jet-stream core lines are examined and put into relation regarding the main trends they can convey. We consider clustering on the fields from which the core lines are extracted, and on the extracted line geometry. By using clustering, we overcome in particular the limitations of spaghetti plots (cf. Fig. 1a). In the shown scenario, spaghetti plots suggest long coherent features. By visualizing single members (cf. Fig. 2), however, it can be observed that the ensemble predicts vastly diverging features comprised of many disconnected line segments. We introduce three different strategies to better reveal the major trends in an ensemble of jet-stream core line of arbitrary geometry and topology. Fig. 3 gives an overview of these strategies, their application is shown in Fig. 1b. Furthermore, we compare the extracted line geometries to ridge lines in fields derived from a closest point representation, and ridges in visitation maps derived from the line geometry (cf. Fig. 1c). Both variants can be computed either for each cluster to provide cluster representatives, or for the entire ensemble.

4.1. Clustering of Physical Fields

The first approach operates on the original fields of wind directions and speed, as explained in Sec. 3 for m ensemble members. A naive

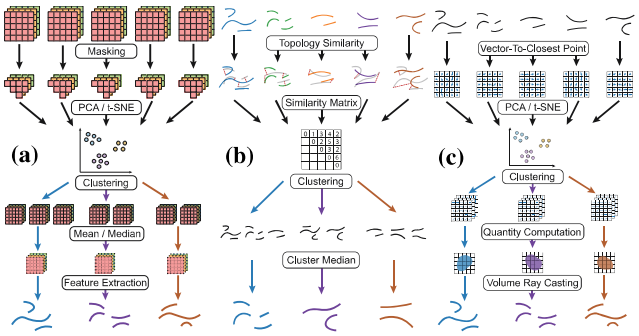


Figure 3: *Clustering jet-stream core lines based on* ***(a)*** *scalar volumes derived from initial fields,* ***(b)*** *geometric distance metrics for lines, and* ***(c)*** *closest point representations derived from line sets.*

approach is to first average all fields over the entire ensemble and perform feature detection in the averaged fields. Feature lines extracted from mean fields of two different scenarios are shown in Fig. 4a and b. For both days, averaging leads to long connected features, while small features vanish due to high variation in the data. In comparison to the ground truth, averaging is not able to convey important trends, such as the long jet-stream core line south of Greenland (Fig. 4b), for regions of high variation. We also discovered that clustering directly on the original scalar fields leads to unsatisfying results due to high variation in the data across vertical levels and ensembles. More specifically, inputs like wind direction and speed over a certain domain do not necessarily correlate locally or globally with the occurrence of line features.

To overcome this problem, we propose the following clustering strategy outlined in Fig. 3a. From the input field, we first derive quantities that describe the characteristics of jet-stream core lines and from which these lines are extracted. In particular, we use directional derivatives of wind (cf. 1) and wind speed as initial scalar fields for clustering. The latter is important as forecasters are only interested in features of high wind speed. Based on these scalar fields, we mask all grid points where the derivatives and wind speed do not approximately match the criteria listed in Sec. 3 for all ensemble members, and eventually remove these points from the initial fields. Next, we concatenate the initial fields and perform dimensionality reduction using Principal Component Analysis (PCA). Here, we only take into account the principal components representing at least 90% of data variance, similar to [FKRW16]. In PCA space, we use clustering to obtain groups of ensemble members with similar set of scalar fields. The fields of each cluster are then either averaged, or features are extracted from the most representative fields per cluster (selected similar to eq. 3).

We propose to choose cluster median over computing the mean for each cluster as the mean leads to less satisfying results with important smaller features being vanished. Fig. 4c,d show outcomes of clustering using the mean across all clusters. These images suggest fully connected line features whereas the ground truth in Fig. 4a,b (inset) reveals trends of interrupted features, for instance, over Florida. Contrary to that, using the scalar field median (cf. Fig. 1b and Fig. 7b) predict interruptions and convey more minor trends, thus, being more similar to the ground truth.

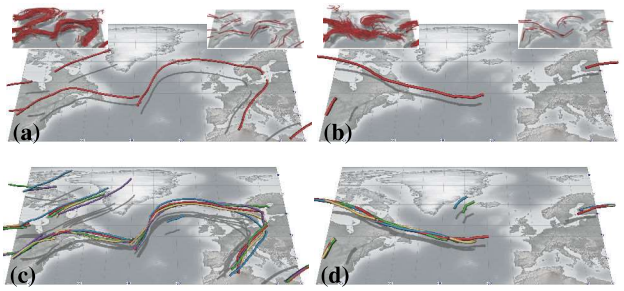


Figure 4: ***(a)*** *Jet-stream core lines extracted from mean input fields of the entire weather ensemble for (a) September 28, 2016, and* ***(b)*** *October 02, 2016, spaghetti plot and the ground truth are shown in the insets.* ***(c)*** *Mean representations for case (a) after averaging clusters obtained from directional derivative fields.* ***(d)*** *Results from clustering similar to (c) for use case (b).*

4.2. Geometry-Based Clustering

Next, we consider the scenario where only the ensemble of extracted jet-stream core lines is given as a set of lines per ensemble member. This requires the computation of feature-to-feature similarities regarding the shape of their line geometries. We introduce two approaches to perform clustering on 3D line feature ensembles.

4.2.1. Geometry Comparison

The first approach is illustrated in Fig. 3b. Given that each ensemble member contains several lines varying in shape and topology, we first measure member-to-member similarity by comparing pairs of lines vertex-wise. We use both the mean closest point distance (MCPD) and the Hausdorff metric (HD) for comparing two sets of lines, similar to [MVv05; BBP*05]. MCPD is the average of the closest point-wise distances while HD computes the maximum of all minimal closest distances between two non-empty subsets. In our experiments, we found that the results using both metrics were quite similar. Yet, since HD produces slightly better results in terms of trend prediction, we use this metric for topology comparison. In the following, we assume that $d_H(A,B)$ represents the Hausdorff distance between two point sets A and B. For each line in one ensemble member, we determine the line with minimum d_H to a line in the other member. For final comparison of two ensemble members, we take the average d_H between two ensemble members for all line-pair-wise minimal HDs. Let E be the ensemble set with ensemble members $i, j \in E$ consisting of line point sets Li, L_j. The member-wise similarity $d_s(L_i, L_j)$ is then defined as:

$$\begin{aligned} d_{He}(L,L_j) &= min_{L' \in L_j}\{d_H(L,L')\} \\ d_s(L_i,L_j) &= mean_{L \in L_i}\{max\{d_{He}(L,L_j), d_{He}(L_j,L)\}\} \end{aligned} \tag{2}$$

Based on d_s, we set-up a distance matrix containing the similarity values for each ensemble member pair used for line clustering.

After clustering, we choose the jet-stream core lines from the ensembles which best represent their clusters. Here, we consider the median of the cluster ensemble members to be the the most representative. An ensemble member serves as the cluster median if its similarity d_s is smallest to all other group members. Let $C \subset E$ be a cluster, then the representative line feature is defined as follows:

$$argmin_{i \in C}\{d_s(L_i,L_j) | j \in C \wedge i \neq j\} \tag{3}$$

Note that with the median we obtain actual line feature predictions and display those as possible outcomes for further analysis. An example of clustering results using HD as similarity metric is shown in Fig. 8a, the best cluster representatives are shown in Fig. 7c.

4.2.2. Implicit Line-Feature Representation

Similarity metrics operating on point-to-point distances are very sensitive to the topology and orientation of given line features, strongly influenced by outliers, especially when comparing several disconnected line features per ensemble member. To overcome this limitation, we make use of a method that first transforms a jet-stream core line of a single ensemble member into a domain-filling field in which this feature is given implicitly. Then, clustering operates on the ensemble of derived fields. Since jet-stream core lines are not closed and inside/outside classification is not possible, signed distance fields cannot be utilized for our purpose. Instead, as proposed by Demir et al. [DJW16], we compute at each grid cell the vectors to the closest points of each jet-stream core line in the ensemble, and cluster the resulting closest-point field (see Fig. 3c).

To transform jet-stream core lines to a closest-point representation, the ensemble domain is first discretized using a Cartesian grid of fixed resolution. Vcp volumes are then generated by using a bounding volume hierarchy for all line segments in a bottom-up fashion. At each grid point, we store m vcps pointing to the closest feature in each member (compare Fig. 6b). Vector magnitudes represent the distance to these closest feature. For clustering, we consider members to be similar at a grid point if all vectors point into the same direction and have a similar magnitude. To improve cluster results, vcp volumes are first preprocessed by filtering out grid points which do not point to any close-by feature, with vcp magnitude exceeding a user-defined minimum length λ with $||vcp_i(\vec{x})|| \geq \lambda$ for all members i. λ is set to half the grid distance. Dimensionality reduction is conducted via t-SNE [MH08] to project the vector volume dimension to 2D for improved cluster separation. Clustering is then performed on the resulting data points in 2D. We use t-SNE with a perplexity value of 5 for $m = 51$ ensemble members to maintain rather local trends with at least 5 close data points. Higher perplexity values force t-SNE to consider more global structures in the data, which we found to produce clusters of higher geometric variability.

To analyze different cluster characteristics, each cluster can be visualized separately, or new quantities can be derived from each cluster using the set of vcp volumes assigned to it. For visualization, we use GPU volume ray-casting to render the isosurface for locations at least half of the grid distance away from any line feature. The minimal distance d_{min} of a vcp set to a line feature is defined as

$$d_{min} = min\{||vcp_i(\vec{x})||, i \in C\} \tag{4}$$

Similar to Demir et al. [DW15], we employ a custom interpolation scheme combining tri-linear interpolation with a plane-based approach for smooth isosurface rendering.

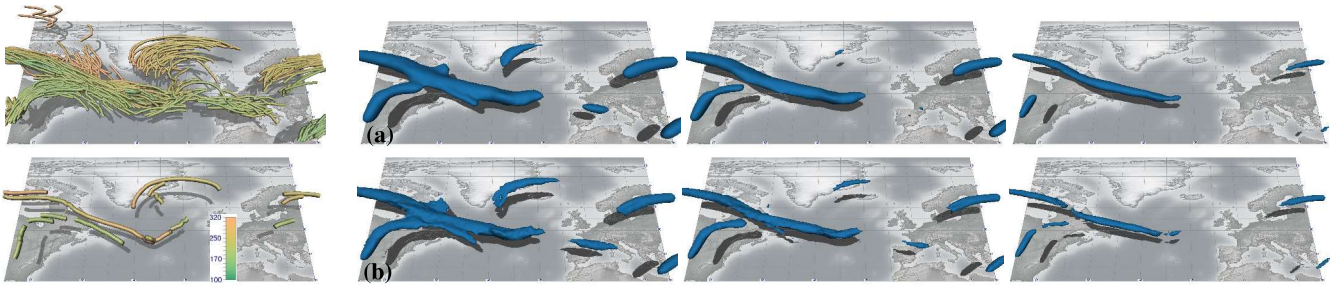


Figure 5: *Left most images: Jet-stream core line ensemble for lead time October 02, 2016 (top) and ground truth core line features (bottom). Top row **(a)**: Visitation map of line features for the entire ensemble with different thresholds for isosurface extraction ($\tau = 0.4/0.5/0.6$). Bottom row **(b)**: Central tendency of line features with different thresholds for isosurface ray-casting ($\sigma = 0.4/0.5/0.6$).*

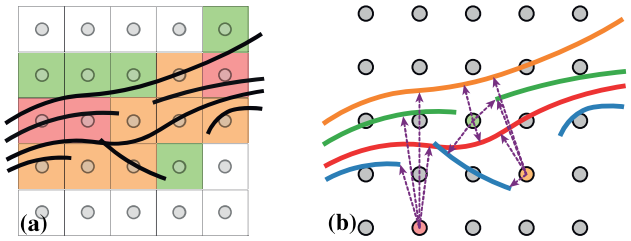


Figure 6: ***(a)** Schematic of a visitation map from a line set (black bold lines). Color represents amount of density related to the number of line crossings (red = 3 crossings, yellow = 2, green = 1). **(b)** Schematic of central tendency computed from closest point vectors (purple dashed arrows). Grid point color indicates amount of centrality (green = highest, orange = medium, red = lowest).*

5. Ridge Line Extraction in Proxy Fields

In addition to clustering, we convey trends in line sets by applying ridge line extraction in fields that are derived from given line geometries, e.g., in vcp fields. Extracted ridge lines represent artificial "mean" representations of the given features, conveying major and minor trends in the feature ensemble. Note that this is contrary to feature detection from averaged scalar fields (of the entire ensemble or single clusters), where minor trends in regions of high variance can vanish (cf. Fig. 4d with best representatives in Fig. 7 and further examined in Sec. 6). In the following, we discuss two different approaches to obtain such fields.

5.1. Visitation Map

Line geometry can be used to analyze the frequency of line feature occurrences within the ensemble domain. This is accomplished by rasterizing line segments into a discrete grid and counting the number of line crossings per voxel cell. Fig. 6a illustrates a visitation map with line feature ensembles of 4 different ensemble members. In particular, we implement visitation maps proposed by Bürger et al. [BFMW12] and used in [FBW16]. Visitation maps make use of a discrete voxel grid and count for each cell the number of line segments crossing the cell. We use rasterization of line vertices to build the visitation maps with each vertex assumed to be a particle – in this context called splat-kernel – of a user-defined diameter. Splat-kernels with support of 4 voxels were chosen for our data and the density values were normalized in a second pass to alleviate thresholding. Next, we use GPU volume ray-casting to obtain isosurfaces with a user-defined density. This reveals regions of high feature density and provides first insights about possible tendencies in the data. Fig. 5b shows a visitation map for 3 different thresholds where major trends of high frequent occlusions are revealed but interruptions are hardly detected.

We also extract ridge lines from visitation maps using classical ridge detection (cf. [PS08]). The ridge lines provide approximate line features which serve as possible predictions for the current ensemble (see Fig. 1c bottom and Fig. 7f,h). Ridge line extraction is restricted by a user-defined minimal density for visitation maps. This threshold has to be set carefully as high values lead to high information loss (see Fig. 5a for threshold comparison). In our work, we found thresholds between 0.4 and 0.6 most suitable for our data. We can compute visitation maps for either the entire ensemble or for each cluster individually. Note that ridge lines are approximations producing new features not contained in the ensemble. In Sec. 6, we compare ridge lines to feature lines obtained from clustering techniques to discuss their potential.

5.2. Central Tendency

We use vcp volumes to also determine the central tendency of the data. Central tendency (centrality) represents locations which are most central in the data set, meaning that the grid points lie in between all closest line features and inherit small magnitudes. An example of three different central tendencies is sketched in Fig. 6b for 4 ensemble members. For centrality, we additionally compute the mean vcp $\vec{\mu}$ at each cell:

$$\vec{\mu} = \overline{vcp}(\vec{p}) = \frac{1}{N}\sum_{i=1}^{N} vcp_i(\vec{p}) \tag{5}$$

Note that we refrain from visualizing the isosurface of $\vec{\mu}$ as this introduces artifacts in rendering, in example for grid points in between two line feature groups. The final central tendency σ of a grid point is computed as follows:

$$\begin{aligned} d_{max} &= max\{||vcp_i(\vec{x})||, i \in C\} \\ \sigma &= max\{1 - d_{max}, \sqrt{f \cdot (1 - ||\vec{\mu}||)}\}, \end{aligned} \tag{6}$$

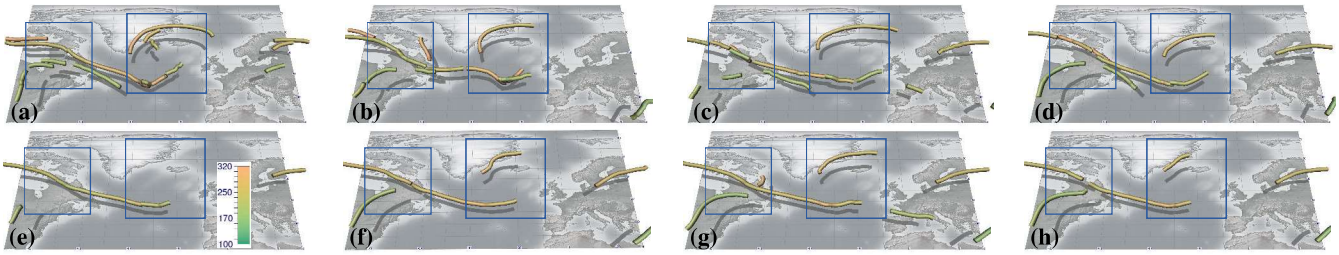


Figure 7: *Best cluster representatives for scenario on October 2, 2016 (lead time) simulated for September 28 (init time) from:* ***(a)*** *ground truth,* ***(b)*** *derivative field clustering,* ***(c)*** *geometry-based clustering,* ***(d)*** *vcp clustering,* ***(e)*** *ensemble mean,* ***(f)*** *ensemble visitation map,* ***(g)*** *ensemble centrality,* ***(h)*** *cluster centrality. Blue rectangles mark regions of varying feature prediction across all ridges.*

where f is the fraction of vcp vectors with a vector magnitude $||vcp_i(\vec{x})|| < \tau$, with τ being half the grid distance. The factor f is important as we want to filter out cases where many vcp of high magnitude cancel out each other, resulting in artifacts with falsely classified points of high centrality. The final rendering of central tendency for the entire ensemble is shown in Fig. 5b with 3 different threshold for σ. From the central tendency volume, we can also extract ridge lines, both for the ensemble and for each cluster. Compared to the visitation maps, extracting ridges from the central tendency differs from ridges in visitation maps in terms of trend inference. In example, features over Florida and Greenland (cf. Fig. 5b) are better revealed than with visitation maps (cf. Fig. 5a) when compared to ground truth. However, central tendency produced a jet-stream core line over Spain which has not been predicted by the ground truth, leading to false interpretations.

6. Results and Use Cases

We discuss the potential of all proposed techniques by means of real-world weather forecasting cases from the 2016 North Atlantic Waveguide and Downstream Impact Experiment (NAWDEX, [SCW*18]), an atmospheric research field campaign. Forecast data are obtained from the Ensemble Prediction System (ENS) at the ECMWF with 51 ensemble members. The data is a regular latitude-longitude grid with grid spacing of 1.0° and 70 terrain-aligned model levels in the vertical. To assess the potential of our methods in terms of trend prediction, we compare the results to jet-stream core lines extracted from re-analysis data obtained from ECMWF using the ERA5 re-analysis model [HB19]. ERA5 data is similar to the NAWDEX data with 0.5° grid spacing and 276 vertical levels.

To enable a proper quality comparison, we selected cases where trends in the ground truth are predicted by a subset in the ensemble forecast. Furthermore, we will examine weather forecasts 72 and 96 hours from initial simulation time because of the following reasons: a) these hours are commonly used by forecaster to create short- and medium-range forecasts, and b) the high variation of features prediction hampers forecasters to infer clear trends from standard spaghetti plots. We assess the quality of each approach by comparing the median MCPD of best cluster representatives (predictions closest to ground truth) and ridge lines to all line features in ERA5. To compare the average closest distance of lines contained in feature predictions from our approaches to the ground truth lines in ERA5, and to be insensitive to features not present in the reference, we found that MCPD is most suitable here for quality comparison.

Table 1: *Median of mean point-pair distances of best cluster representatives and ridge lines to ground truth for all techniques: mean over entire ensemble (MEAN), scalar field clustering with mean (SFCμ) and median (SFCM), geometry-based clustering (GC), vcp clustering (VC), cluster central tendency (VCC), ensemble central tendency (C), and cluster visitation map (V).*

Use cases	MEAN	SFCμ	SFCM	GC	VC	VCC	C	V
Sep. 28	4.58	3.58	2.61	2.25	2.62	2.66	2.79	3.08
Oct. 02	4.73	2.70	3.13	2.47	2.60	2.73	4.31	3.01

6.1. September 28, 2016

We regard a scenario of highly varying predictions on September 28 from initial time September 25, 2016, partly predicting either a closed curved or interrupted jet-stream core line over Greenland or Florida. Taking the mean over the entire ensemble (MEAN) does not convey information about interrupted features and minor feature trends (cf. Fig. 4a). Clustering the derived scalar fields and obtaining the mean over the clusters for feature detection led to improved results (cf. Fig. 4c), indicated by smaller values for MCPD (cf. Table 1 MEAN and SFCμ); however, those means did not convey any interruptions. We also applied all three clustering techniques to the data and obtained the best representative feature lines for comparison. In terms of cluster separation, vcp ensemble clustering managed to separate the line ensemble best compared to geometry-based scalar field clustering (shown Fig. 1b). All clustering techniques produced representatives which are similar to predictions in the reference, predicted multiple interrupted lines over Florida, and, thus, outperformed MEAN wrt. MCPD.

Next, we assess the quality of ridge lines identified in the visitation map of a cluster obtained from a geometry-based clustering (cf. Fig. 1c bottom), and ridge lines from the central tendency field (see Fig. 1c top). Both methods are able to provide satisfying results, however, the ridge line of the visitation map fails to convey some minor features (in example west of Great Britain), leading to slightly higher MCPD values. Yet, ridge lines are closer to the reference than lines from MEAN, and those from centrality can compete with all best cluster representatives. In Summary, we could improve MEAN wrt. MCPD with all cluster representatives and ridge lines, best results were achieved with cluster representatives (cf. Table 1).

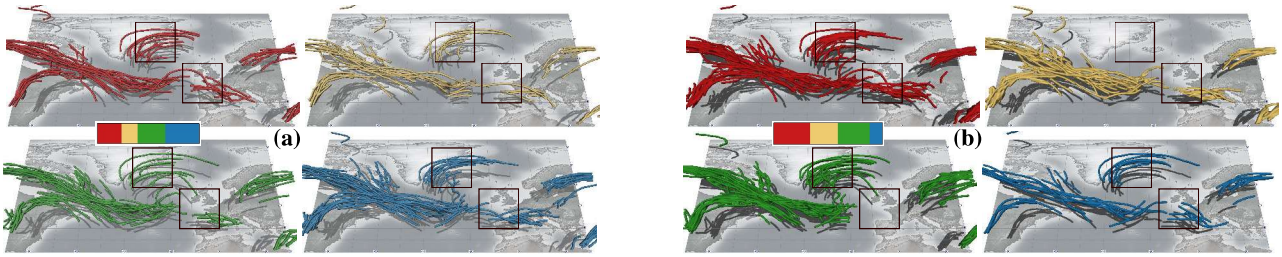


Figure 8: *Clustering results with 4 clusters (each cluster colored)* ***(a)*** *obtained from geometry-based clustering with Hausdorff distance, and* ***(b)*** *from clustering on vector-to-closest point volumes. Inset shows member distribution for all clusters. Black rectangles highlight major differences in trend prediction between both clustering techniques.*

6.2. October 02, 2016

We picked another scenario for October 2 from September 28, 2016 with highly varying predictions, in example feature occurrence south of Greenland, two merging or two distinct jet cores from Florida towards the North Atlantic, and multiple interruptions in between (cf. blue rectangles in Fig. 7). Similar to Sec. 6.1, taking the mean over the entire ensemble led to massive information loss and missed features compared to the reference (see Fig. 4b and d). Best representatives from all clustering techniques are shown in Fig. 7b,c and d. Computing the mean or median from scalar-field-based clustering produced results close to the ground truth, both visually and wrt. MCPD (see. column 2 and 3 in Table 1). All clustering techniques, again, were able to provide good results coinciding with the ground truth. In example, they correctly predicted two distinct features over Florida with multiple interruptions (cf. Fig. 7b,c, and d). In terms of cluster separation, we found that vcp clustering is superior to all other clustering techniques, in example, it excelled line set comparison with HD as shown in Fig. 8. Opposed to clustering, ridge lines in central tendency and visitation map volumes (cf. Fig. 7f,g, and h) could not properly reveal such interruptions in the data. In addition, centrality over the entire ensemble produced a feature over Spain not present in the reference (Fig. 7g), leading to the highest MCPD value compared to all other techniques except the MEAN. All approaches, except centrality, were able to highly outperform MEAN wrt. MCPD where best cluster representatives were closest to the reference.

7. Conclusion

In this work, we have introduced and compared different approaches to cluster 3D line sets of arbitrary orientation and topology for trend inference. In particular, we clustered (a) on derived scalar fields, (b) on line geometry using the Hausdorff metric, and (c) on vector-to-closest point representations invariant to line topology and orientation. We have shown that all clustering approaches have the potential to infer major and minor trends from weather ensemble forecasts close to trends contained in re-analysis data considered to be the ground truth. In addition, we have provided methods to analyze of feature occurrence frequency with visitation maps and derived central tendency fields from closest-point volumes to classify points most central between line sets. From these fields, we have extracted ridge lines as an artificial mean representation and have demonstrated that they can compete with clustering when applied to cluster results only, yet produced slightly inferior results. In terms of clustering, we can recommend two approaches: If the initial physical fields are given, we propose to operate on derived scalar fields characterizing points close to potential features. In case line geometry is provided only, clustering based on vector-to-closest-point representations is the most effective technique to obtain clear distinct clusters and trends in the data.

Note that the examination of cluster quality for each proposed clustering technique is beyond the scope of this work and requires further investigations in the future. This includes automated selection of best clusters and best cluster representatives by comparing multiple clustering results and incorporating information about additional physical quantities. However, our proposed techniques are not limited to jet-stream core lines and can be applied to arbitrary scalar fields, feature detection methods, and line sets. Note that parameters for t-SNE and ridge detection have to be adapted depending on number of data elements (ensemble members) and their dimensionality. We have integrated our techniques in the visualization tool "Met.3D" (see [FKRW17; KHS*18; KTB*18]), enabling domain experts to further judge the quality and likelihood of best representatives by plotting additional atmospheric processes along with jet-stream core lines. We are confident that this analysis can help domain experts in estimating the best predictions and trends from vastly diverging line features in the future.

8. Acknowledgements

This research has been done within the subproject A7 of the Transregional Collaborative Research Center SFB/TRR 165 Waves to Weather funded by the German Research Foundation (DFG).

References

[ASE16] ATHAWALE, T., SAKHAEE, E., and ENTEZARI, A. "Isosurface Visualization of Data with Nonparametric Models for Uncertainty". *IEEE Transactions on Visualization and Computer Graphics* 22.1 (Jan. 2016), 777–786 2.

[BBP*05] BLAAS, J., BOTHA, C. P., PETERS, B., VOS, F. M., and POST, F. H. "Fast and reproducible fiber bundle selection in DTI visualization". *VIS 05. IEEE Visualization, 2005.* Oct. 2005, 59–64 2, 4.

[BFMW12] BÜRGER, K., FRAEDRICH, R., MERHOF, D., and WESTERMANN, R. "Instant Visitation Maps for Interactive Visualization of Uncertain Particle Trajectories". *Proceedings Visualization and Data Analysis 2012*. Vol. SPIE 8294. 2012, 5.

[BM10] BRUCKNER, S. and MÖLLER, T. "Isosurface Similarity Maps". *Computer Graphics Forum* 29.3 (2010), 773–782 2.

[DDW14] DEMIR, I., DICK, C., and WESTERMANN, R. "Multi-Charts for Comparative 3D Ensemble Visualization". *IEEE Transactions on Visualization and Computer Graphics* 20.12 (Dec. 2014), 2694–2703 2.

[DJW16] DEMIR, I., JAREMA, M., and WESTERMANN, R. "Visualizing the Central Tendency of Ensembles of Shapes". *SIGGRAPH Asia 2016 Symposium on Visualization*. SA '16. ACM, 2016 2, 4.

[DS15] DUTTA, S. and SHEN, H.-W. "Distribution driven extraction and tracking of features for time-varying data analysis". *IEEE transactions on visualization and computer graphics* 22.1 (2015), 837–846 2.

[DW15] DEMIR, I. and WESTERMANN, R. "Vector-to-Closest-Point Octree for Surface Ray-Casting". *Vision, Modeling & Visualization*. Ed. by BOMMES, D., RITSCHEL, T., and SCHULTZ, T. The Eurographics Association, 2015 4.

[FBW16] FERSTL, F., BÜRGER, K., and WESTERMANN, R. "Streamline Variability Plots for Characterizing the Uncertainty in Vector Field Ensembles". *IEEE Transactions on Visualization and Computer Graphics* 22.1 (Jan. 2016), 767–776 2, 5.

[FC11] FERRANTI, L. and CORTI, S. "New clustering products". *ECMWF Newsletter* 127 (2011), 6–11 2.

[FFST19] FAVELIER, G., FARAJ, N., SUMMA, B., and TIERNY, J. "Persistence Atlas for Critical Point Variability in Ensembles". *IEEE Transactions on Visualization and Computer Graphics* 25.1 (Jan. 2019), 1152–1162 2.

[FKRW16] FERSTL, F., KANZLER, M., RAUTENHAUS, M., and WESTERMANN, R. "Visual Analysis of Spatial Variability and Global Correlations in Ensembles of Iso-Contours". *Computer Graphics Forum (Proc. EuroVis)* 35.3 (2016), 221–230 2, 3.

[FKRW17] FERSTL, F., KANZLER, M., RAUTENHAUS, M., and WESTERMANN, R. "Time-Hierarchical Clustering and Visualization of Weather Forecast Ensembles". *IEEE Transactions on Visualization and Computer Graphics* 23.1 (Jan. 2017), 831–840 2, 7.

[HB19] HENNERMANN, K. and BERRISFORD, P. *ERA5 data documentation*. Accessed 11 June 2019. 2019 6.

[HBS18] HAZARIKA, S., BISWAS, A., and SHEN, H. "Uncertainty Visualization Using Copula-Based Analysis in Mixed Distribution Models". *IEEE Transactions on Visualization and Computer Graphics* 24.1 (Jan. 2018), 934–943 2.

[HDSC19] HAZARIKA, S., DUTTA, S., SHEN, H., and CHEN, J. "CoDDA: A Flexible Copula-based Distribution Driven Analysis Framework for Large-Scale Multivariate Data". *IEEE Transactions on Visualization and Computer Graphics* 25.1 (Jan. 2019), 1214–1224 2.

[JDKW15] JAREMA, M., DEMIR, I., KEHRER, J., and WESTERMANN, R. "Comparative visual analysis of vector field ensembles". *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on*. Oct. 2015, 81–88 2.

[Jr63] JR., J. H. W. "Hierarchical Grouping to Optimize an Objective Function". *Journal of the American Statistical Association* 58.301 (1963), 236–244 2.

[KHS*18] KERN, M., HEWSON, T., SADLO, F., WESTERMANN, R., and RAUTENHAUS, M. "Robust Detection and Visualization of Jet-Stream Core Lines in Atmospheric Flow". *IEEE Transactions on Visualization and Computer Graphics* 24.1 (Jan. 2018), 893–902 1, 2, 7.

[KTB*18] KUMPF, A., TOST, B., BAUMGART, M., RIEMER, M., WESTERMANN, R., and RAUTENHAUS, M. "Visualizing Confidence in Cluster-based Ensemble Weather Forecast Analyses". *IEEE Transactions on Visualization and Computer Graphics* 24.1 (Jan. 2018), 109–119 2, 7.

[LLBP12] LIU, S., LEVINE, J. A., BREMER, P.-T., and PASCUCCI, V. "Gaussian mixture model based volume visualization". *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*. IEEE. 2012, 73–77 2.

[LPK05] LOVE, A. L., PANG, A., and KAO, D. L. "Visualizing spatial multivalue data". *IEEE Computer Graphics and Applications* 25.3 (2005), 69–79 2.

[MH08] MAATEN, L. V. D. and HINTON, G. "Visualizing data using t-SNE". *Journal of Machine Learning Research* 9.Nov (2008), 2579–2605 4.

[MMP*17] MOLNOS, S., MAMDOUH, T., PETRI, S., NOCKE, T., WEINKAUF, T., and COUMOU, D. "A network-based detection scheme for the jet stream core". *Earth System Dynamics* 8.1 (2017), 75–89 2.

[MVv05] MOBERTS, B., VILANOVA, A., and VAN WIJK, J. J. "Evaluation of fiber clustering methods for diffusion tensor imaging". *VIS 05. IEEE Visualization, 2005.* Oct. 2005, 65–72 2, 4.

[MWK14] MIRZARGAR, M., WHITAKER, R. T., and KIRBY, R. M. "Curve boxplot: Generalization of boxplot for ensembles of curves". *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), 2654–2663 2.

[OLK*14] OELTZE, S., LEHMANN, D. J., KUHN, A., JANIGA, G., THEISEL, H., and PREIM, B. "Blood Flow Clustering and Applications inVirtual Stenting of Intracranial Aneurysms". *IEEE Transactions on Visualization and Computer Graphics* 20.5 (May 2014), 686–701 2.

[PH13] PÖTHKOW, K. and HEGE, H.-C. "Nonparametric Models for Uncertainty Visualization". *Computer Graphics Forum* 32.3 (2013), 131–140 2.

[PS08] PEIKERT, R. and SADLO, F. "Height Ridge Computation and Filtering for Visualization". *2008 IEEE Pacific Visualization Symposium*. Mar. 2008, 119–126 5.

[Rei63] REITER, E. R. *Jet-stream meteorology*. 1st. University of Chicago Press, 1963 2.

[RT12] ROSSL, C. and THEISEL, H. "Streamline Embedding for 3D Vector Field Exploration". *IEEE Transactions on Visualization and Computer Graphics* 18.3 (Mar. 2012), 407–420 2.

[SCW*18] SCHÄFLER, A., CRAIG, G., WERNLI, H., et al. "The North Atlantic Waveguide and Downstream Impact Experiment". *Bulletin of the American Meteorological Society* 99.8 (2018), 1607–1637 6.

[SD05] STRONG, C. and DAVIS, R. E. "The surface of maximum wind as an alternative to the isobaric surface for wind climatology". *Geophys. Res. Lett.* 32.4 (Feb. 2005), L04813+ 2.

[SG02] STREHL, A. and GHOSH, J. "Cluster ensembles—a knowledge reuse framework for combining multiple partitions". *Journal of machine learning research* 3.Dec (2002), 583–617 2.

[SSL17] SPENSBERGER, C., SPENGER, T., and LI, C. "Upper Tropospheric Jet Axis Detection and Application to the Boreal Winter 2013/14". *Mon. Wea. Rev.* (Mar. 2017) 2.

[TN14] THOMAS, D. and NATARAJAN, V. "Multiscale Symmetry Detection in Scalar Fields by Clustering Contours". *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), 2427–2436 2.

[Wil11] WILKS, D. S. *Statistical Methods in the Atmospheric Sciences*. 3rd. Academic Press, June 2011 2.

[WLW*17] WANG, K.-C., LU, K., WEI, T.-H., SHAREEF, N., and SHEN, H.-W. "Statistical visualization and analysis of large data using a value-based spatial distribution". *2017 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE. 2017, 161–170 2.

[WMK13] WHITAKER, R. T., MIRZARGAR, M., and KIRBY, R. M. "Contour Boxplots: A Method for Characterizing Uncertainty in Feature Sets from Simulation Ensembles". *IEEE Transactions on Visualization and Computer Graphics* 19.12 (Dec. 2013), 2713–2722 2.

[ZHQL16] ZHANG, H., HOU, Y., QU, D., and LIU, Q. "Correlation Visualization of Time-Varying Patterns for Multi-Variable Data". *IEEE Access* 4 (2016), 4669–4677 2.

A Comparison of Rendering Techniques for 3D Line Sets with Transparency

Michael Kern, Christoph Neuhauser, Torben Maack, Mengjiao Han, Will Usher, Rüdiger Westermann

Abstract—This paper presents a comprehensive study of rendering techniques for 3D line sets with transparency. The rendering of transparent lines is widely used for visualizing trajectories of tracer particles in flow fields. Transparency is then used to fade out lines deemed unimportant, based on, for instance, geometric properties or attributes defined along with them. Accurate blending of transparent lines requires rendering the lines in back-to-front or front-to-back order, yet enforcing this order for space-filling 3D line sets with extremely high-depth complexity becomes challenging. In this paper, we study CPU and GPU rendering techniques for transparent 3D line sets. We compare accurate and approximate techniques using optimized implementations and several benchmark data sets. We discuss the effects of data size and transparency on quality, performance, and memory consumption. Based on our study, we propose two improvements to per-pixel fragment lists and multi-layer alpha blending. The first improves the rendering speed via an improved GPU sorting operation, and the second improves rendering quality via transparency-based bucketing.

Index Terms—Scientific visualization, line rendering, order-independent transparency.

1 Introduction

In many visualization tasks, the need to efficiently display sets of 3D lines is paramount. Applications range from the visualization of pathways of particle tracers in flow fields or over moving vehicles for smart transportation and urban planning, to exploring neural connections in the brain or relations encoded in large graphs and network structures. Prior work such as [3], [12], [19], [27] has shown that transparency, when used carefully to avoid overblurring, can be used effectively to relieve occlusions and to accent important structures while maintaining less important context information. It is particularly useful for exploratory visualization tasks, where users interactively select the strength of transparency and the mapping of data values to transparency.

Rendering transparency, however, introduces a performance penalty. When using transparency, the per-pixel color and opacity contributions need to be blended in correct visibility order, i.e., by using α-blending (where α represents a point's opacity) in either front-to-back or back-to-front order. Rendering techniques can be distinguished as to whether they compute the visibility order exactly or approximately, and how this order is established. Especially for line sets, which have a significantly higher depth complexity than surface or point models, maintaining the visibility order during rendering can become a severe performance bottleneck.

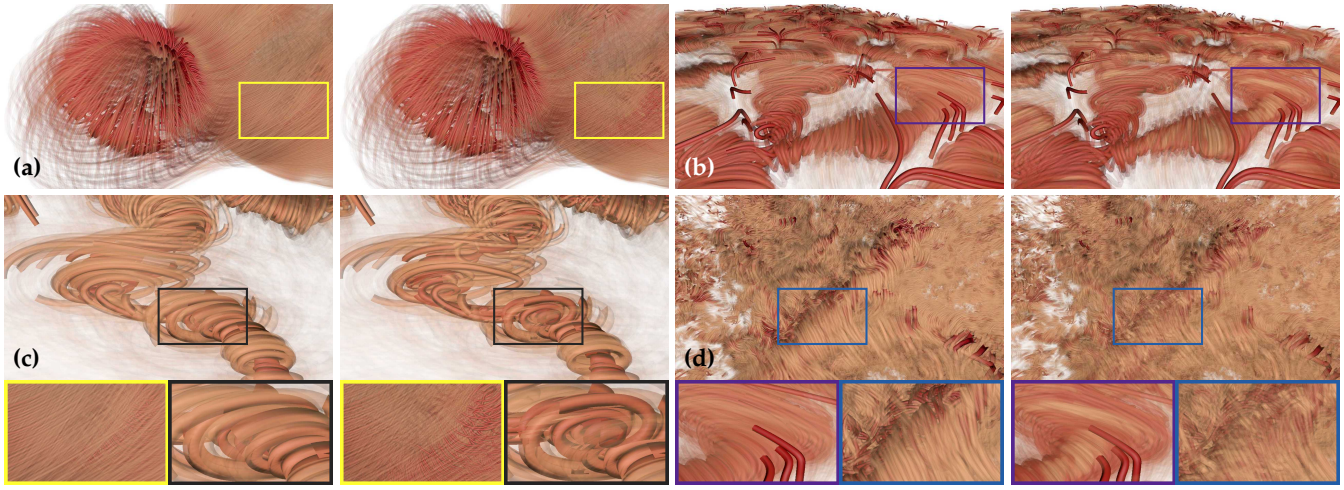
In this study, we evaluate exact and approximate object- and image-order transparency rendering techniques, with intending to analyze the performance of such techniques when used to render line sets with an extremely high depth complexity. Our evaluation includes an in-depth evaluation of model-specific acceleration schemes. We further demonstrate the use of approximate transparency rendering techniques for surface and point models with high depth complexity, though refrain from a detailed performance evaluation on these cases. The latter would require considering specific acceleration structures for such surface or point models, which is beyond the scope of a single paper.

Object-order techniques make use of GPU rasterization. We consider Depth Peeling (DP) [10] and Per-Pixel Linked Lists (LL) [43], both of which can render transparency accurately at the cost of computing or memory. Other object-order techniques use (stochastic) transmittance approximations, where transmittance refers to the multiplicative accumulation of per-fragment transparencies. Of the many different variants of approximate techniques, we selected Multi-Layer Alpha Blending (MLAB) [32] and the most recent Moment-Based Blending Technique (MBOIT) [25] (see Fig. 1 for example images). Both approximate techniques use only small and constant additional buffer resources.

We also evaluate four image-order techniques based on ray-tracing. We consider the Generalized Tubes method [13] as well as Embree's built-in Bezier curve primitives [40] implemented in Intel's OSPRay CPU ray-tracing framework (OSP) [38], a GPU ray-tracer using NVIDIA's RTX ray-tracing interface [26] through the Vulkan API (RTX), and voxel-based GPU line ray-tracing (VRC) [15]. All techniques utilize dedicated data structures to facilitate efficient ray traversal as well as empty space skipping and thus provide effective means to evaluate the capabilities of image-order line rendering.

OSP, RTX, VRC, DP, and LL, despite their algorithmic differences, are all accurate methods and yield the same rendering result. Performance-wise, on the other hand, these techniques differ substantially, and for large data sets some

- *M. Kern, C. Neuhauser, T. Maack, and R. Westermann are with the Computer Graphics & Visualization Group, Technische Universität München, Garching, Germany*
E-mail: {michi.kern, christoph.neuhauser, westermann}@tum.de, torben.maack@mytum.de
- *M. Han and W. Usher are with the Scientific Computing and Imaging Institute, University of Utah, U.S.*
E-mail: {mengjiao, will}@sci.utah.edu



(a) PSNR = 33.41, SSIM = 0.907 **(b)** PSNR = 31.98, SSIM = 0.901 **(c)** PSNR = 34.42, SSIM = 0.913 **(d)** PSNR = 31.10, SSIM = 0.842

Fig. 1: Strengths and weaknesses of transparent line rendering techniques. For each pair, the left image shows the ground truth (GT). Right images show **(a)** approximate blending using MLABDB, **(b)** opacity over-estimation of MBOIT, **(c)** reverse blending order of MLABDB, **(d)** blur effect of MBOIT. Speed-ups to GT rendering technique: (a) 7, (b) 2, (c) 3.5, (d) 4.5.

of them even turn out to be impractical. The main goal of our evaluation study is to shed light on the differences between these techniques and to provide guidelines for selecting a suitable rendering technique for a given application.

Contribution

We provide a qualitative and quantitative comparison of techniques for rendering 3D line sets with transparency. For our evaluation, we have systematically selected a set of techniques that we believe are representative for the different principal approaches that are available today. Even though our evaluation study has been performed solely on 3D line sets, the results are also applicable to other application scenarios where transparency is used to reveal otherwise hidden structures.

Through our study, users and practitioners can gain an understanding of the principal implications of using a certain technique and become aware of their major strengths and limitations with respect to quality, memory requirements, and performance. Since we use a range of different sized data sets with vastly different internal structures, our evaluation hints to specific data-dependencies of certain rendering concepts. We tried to individually select a transparency setting for each data set that reveals important features in a meaningful way. Thus, we consider our results representative of typical use cases of transparent line rendering. For each technique, we also analyze the pre-process that is required to build the data representations needed for rendering and perform a thorough evaluation of rendering performance.

Moreover, we have modified LL to improve scalability with the number of fragments, and MLAB to make it less dependent on the order of fragments per pixel. For LL, we developed GPU-friendly variants of shell-sort and priority-queues through the min-heap data structure, resulting in a performance increase of a factor of 2-3. Our implementation of MLAB uses a discrete set of depth intervals and can considerably reduce the number of incorrectly merged fragments.

We have made our implementations publicly available [17], the test environment using NVIDIA's RTX [20], all data sets [16], and all benchmark results for image quality and performance evaluation. We have also included additional descriptions of how to use the implementations and apply them to other data sets.

2 Related Work

Prior work [22], [42] has compared some of the many different rendering techniques for transparent geometry. These evaluations, however, have mainly focused on the use of techniques for real-time graphics effects in scenes comprised of a few spatially extended and homogeneous transparent objects with rather low depth complexity. Thus, the suitability of techniques for visualization tasks as outlined in our work is difficult to infer from available evaluations. To the best of our knowledge, an evaluation and comparison of techniques for rendering large 3D line sets, including ray-based approaches and scenes with extremely high depth complexity and high-frequency transmittance functions, has not been performed.

2.1 Object-order techniques

Several approaches have been proposed to blend the fragments falling into the same pixel in correct visibility order without having to resort to an explicit sorting of geometry. Everitt et al. [10] presented depth peeling, which renders for each pixel in the i-th rendering pass the i-th closest surface point using a second depth buffer test against the values from the previous pass. In early work by Carpenter [6], the A-buffer was introduced as a data structure that stores the unordered set of fragments falling into each pixel. These fragments are then sorted explicitly based on the stored depth information. Yang et al. [43] used per-pixel linked lists to store a variable number of fragments per pixel on the GPU, after which they are sorted to blend the fragments in correct order. Contrary to the linked lists, the k-Buffer [4]

stores only the k nearest fragments, and merges fragments heuristically if more than k fall into the same pixel.

In scenarios where the k-Buffer is not applicable, fragments have to be blended heuristically. Adaptive Transparency [31] operates on k fragments and aims to store an approximation of the transmittance function per pixel. Alpha blending is then performed in a second pass using this approximation. Maule et al. [21] proposed Hybrid Transparency, which aggregates fragments using a k-Buffer and merges them heuristically with respect to depth and opacity. Even though this approach is order-independent, it is not able to cope with scenes containing many layers. Multi-Layer Alpha Blending (MLAB) [32] is a single-pass technique that uses a fixed number of per-pixel transmittance layers to approximate the transmittance along a view ray. When all layers are occupied and the current fragment creates a new layer, the two most appropriate adjacent layers are merged in turn. Stochastic Transparency [9] uses weights to blend or discard fragments based on opacity. Weighted-Blended Order-Independent Transparency [23] proposed to use weights based on occlusion and distance to the camera to merge fragments.

Recently, Münstermann et al. [25] introduced Moment-Based Order-Independent Transparency (MBOIT). Rojo et al. [3] demonstrated the embedding of importance-based transparency control into MBOIT. MBOIT approximates the transmittance function pixel-wise by power moments or trigonometric moments, and applies logarithmic scaling to the absorbance to enforce order-independency and facilitate additive compositing. Moment-Based transparency builds upon Fourier opacity mapping [14], which represents transmittance as a low-frequency distribution dependant on depth, and approximates these distributions using trigonometric moments, i.e., Fourier coefficients.

Another category of techniques render transparent layers using multiple samples per pixel, for example, Stochastic Layered Alpha Blending [42] and Phenomenological Transparency [24]. The latter technique also incorporates physical processes to create realistic effects of translucent phenomena. However, these techniques significantly increase the number of generated fragments, which is problematic in scenarios where the depth complexity is extremely high. As such, we do not consider them in our study.

We do also not consider particle-based [30] and voxel-based [8], [18] rendering techniques for transparent geometry. Especially when used to render space-filling line sets, these techniques significantly increase the number of rendered primitives or the resolution of the used voxel grid and require substantial modifications to render geometric shapes with fine geometric details and sharp outlines.

2.2 Image-order techniques

Image-order techniques for rendering line primitives make use of ray-tracing. Advances in hardware and software technology have shown the potential of ray-tracing as an alternative to rasterization, especially for high-resolution models with many inherent occlusions. Developments in this field include advanced space partitioning and traversal schemes [35], [37], [41], and optimized GPU implementations [1], [18], [29], to name just a few. Wald et al. [39] proposed the use of ray-tracing in combination with a tree-based search structure for particle locations to efficiently find those particles a ray has to be intersected with. Kanzler et al. [15] built upon voxel ray-tracing and proposed a GPU rendering technique for large 3D line sets with transparency. They use an approximate voxel model for 3D lines, using quantization of line-voxel intersection points to a discrete set of locations on voxel faces. Ray-tracing is then performed using the regular grid as an acceleration structure.

Ray-tracing of line sets can be performed on analytic or polygonal tube models by using common acceleration structures like kD-trees or bounding volume hierarchies to accelerated ray-object intersections. CPU and GPU ray-tracing frameworks like OSPRay [38] and OptiX [29] can be used for this purpose. OSPRay builds on Intel's Embree ray tracing kernels [40] and has built-in support for rendering fixed-radius opaque streamlines and Bézier curve primitives. Han et al. [13] further extended OSPRay with a module for rendering Generalized Tube Primitives, supporting varying radii, bifurcations, and transparency. NVIDIA's RTX ray-tracing through the OptiX interface [26] uses RT cores on current GPUs to perform hardware-accelerated ray-primitive intersection tests. OptiX also provides an interface to implement custom shaders, which, in our current scenario, can also be used to analytically intersect rays with tubes.

3 Line Rendering

We classify line rendering algorithms into two major groups: object-order and image-order. Object-order techniques use rasterization of geometric primitives to let the GPU compute the fragments falling into each pixel in an arbitrary order. Although the order is first given by the order in which rendering calls are issued for each primitive, this order is not given when processing each fragment in the fragment shader stage. For transparency, these techniques use either fragment merge heuristics or 2-pass approaches to ensure (correct) transmittance and visibility. In contrast, image-order techniques use ray-tracing to find the surface points seen through the pixels. The correct visibility order of the points along a ray is established by using a space-partitioning scheme to traverse a ray in front-to-back order through space.

3.1 Object-Order

Object-order techniques can be classified into accurate and approximate techniques. Accurate techniques guarantee the exact visibility order of rendered fragments. Approximate techniques violate this order by blending a fragment's color over a color that already contains the color of a fragment that is closer to the camera. Although approximate techniques typically have bounded memory and rendering constraints, accurate approaches come with either unbounded rasterization load or unbounded memory requirements.

Depth Peeling

Depth Peeling (DP) [10] generates pixel-accurate renderings of transparent geometry by rendering the scene multiple times and using the depth buffer to achieve ordered blending, each transparent layer at a time. DP utilizes the depth

buffer hardware test to successively obtain the next closest layer and performs standard front-to-back blending into the current framebuffer.

A unique property of DP is that it does not require any additional memory besides a second depth buffer. On the other hand, DP needs to "peel" layers, i.e., render the scene as many times as the depth complexity of the scene. In our application scenario, where the scenes are comprised of many thousands of thin and often space-filling objects, a huge number of rendering passes typically has to be performed. As indicated in Fig. 2, terminating the blend passes after a fixed number of times is dangerous, because deep layers with high opacity can contribute significantly to the final pixel color.

Due to the high rendering cost of DP, its performance is typically about 1-2 orders of magnitude below that of all other alternatives we consider. Therefore, we decided to use DP solely for generating ground truth images of transparent lines, against which the results of other techniques are compared.

Fig. 2: Intermediate results of depth peeling for layers 1, 10, and 50 of a semi-transparent line set. Note that even at layer 50 notable differences appear in the final result.

Per-Pixel Linked Lists

While DP has bounded memory constraints and unbounded rasterization load, Per-Pixel Linked Lists (LL) [43] come with bounded rasterization load yet unbounded memory requirements. LL renders the scene only once, and stores all generated fragments in linked lists over all pixels. Then, for every pixel, a pixel shader is invoked which traverses the list and stores all fragments belonging to that pixel in a GPU buffer resource. The fragments are then sorted wrt. their depth. Finally, the fragments are blended in sorted order to produce the final pixel color. Besides the global fragment buffer, LL requires a head buffer that stores, for every pixel, the offsets to the first fragment in the linked list, and an atomic counter that tracks the number of inserted fragments to enable concurrent gathering of new fragments into the fragment buffer. LL assumes that the GPU buffer resource is large enough to store all rendered fragments; otherwise, it fails to correctly render the scene. To reduce the memory requirements, one commonly stores fragment colors in 32-bit unsigned integers, with 8 bits per color and α channel.

Even if the available GPU memory is large enough, which is often the case on high-end GPUs, sorting the many hundreds or even thousands of fragments per pixel can become a performance bottleneck. Although simple sorting algorithms such as bubble sort or insertion sort are suitable for small numbers of fragments, they do not scale well to large numbers of fragments. To address this limitation, we have incorporated alternative sorting algorithms that achieve better scalability and can be implemented on top of GPU-friendly data structures, i.e., GPU versions of shell-sort and priority-queue through the min-heap data structure.

Shell-sort [33] is an in-place sorting algorithm based on insertion-sort. It is specifically designed to achieve improved sorting performance for large arrays by exchanging elements that are far apart from each other. Shell-sort subdivides the array into k subarrays by sorting each k-th element of the array via insertion sort. k in this context defines the offset between elements and is hence called the gap. The gap is iteratively decremented in l iterations using a pre-defined sequence $(k_1, k_2, ..., k_l)$ of l numbers. The a_i-th element is then sorted in the i-th iteration. Note that k_l is always 1 since every element needs to be sorted in the last iteration. In our work, we used $l = 4$ and a gap sequence of $(24, 9, 4, 1)$ (based on Table 1 in [7]) to efficiently order elements with an average depth complexity of 124 elements per node.

Priority-queues are implemented with the min-heap data structure. A min-heap is a full binary tree where each node contains a key defining the priority (or order) of the element. For each parent node, the key of its children is either equal or smaller than its own key. Heap data structures are commonly implemented as binary trees. In our case, the depth value of each fragment represents the key. That is, after each insert operation, the root node is the currently closest element in the min-heap.

Upon insertion of all fragments in the heap, the next closest fragment is iteratively obtained by removing the root node from the heap until the heap is empty. Root removal is implemented by setting the element with the least priority as the new root and sinking it down until it is correctly sorted. This process takes $O(\log n)$ time for a heap with n elements. Since the root has to be obtained n times, the total time complexity is $O(n \cdot (\log n))$, which is faster than the depth complexity of the previously mentioned sorting algorithms.

Both sorting algorithms have been embedded into LL to improve its performance. Fig. 3 shows performance graphs for renderings of the aneurysm data set from many different views. As can be seen, shell-sort and priority-queues significantly improve the sorting performance by a factor of 2 to 3 on average and keep the sorting time almost constant over all frames. Due to the slightly better performance of priority-queues for other data sets, we decided to use this version of LL in our evaluations.

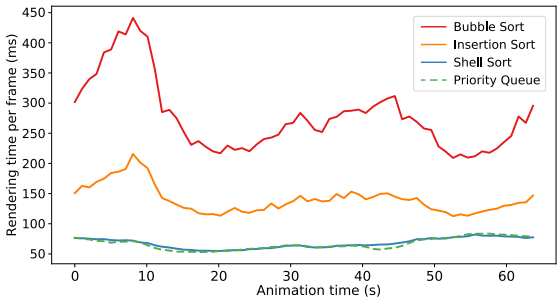


Fig. 3: Rendering times for a flight around the ANEURYSM data set using LL and different sorting algorithms.

Multi-Layer Alpha Blending

Multi-Layer Alpha Blending (MLAB) [32] is a single-pass technique. It belongs to the class of transparency rendering

techniques that are bounded in both memory consumption and rendering load. This class of techniques does not perform exact visibility sorting of fragments but strives to approximate the transmittance and color. MLAB does this by heuristically merging fragments into a small number of layers, a so-called blending array, that are finally merged into the pixel color.

The blending array consists of k buffers, into which incoming fragments are merged. Each of the k buffers stores the blended colors and transmittances of the fragments merged into it, as well as a depth value. Each fragment is placed into a corresponding buffer based on its depth. When all layers are occupied and the current fragment creates a new layer, the two most appropriate adjacent layers are merged in turn.

We chose MLAB because of its simplicity, performance, and low memory consumption. On the other hand, it is clear that with only k layers—eight by default in our current implementation—MLAB is not always able to accurately reconstruct the colors and transmittances for scenes with high depth complexity. In particular, the quality of MLAB is highly dependent on the order in which fragments are generated, as the outcome of the heuristic merge operation depends highly on this order. Some of the inaccuracies shown in Fig. 1 are due to this dependency.

Additionally, MLAB is not frame-to-frame-coherent if the order in which fragments are rendered to the intermediate buffers is not guaranteed over time, resulting in flickering artifacts during animation. We prevent this error by explicitly enabling order-preserving pixel synchronization (see [11]) so that fragments are processed in the order primitives were issued. Note that with pixel synchronization enabled, we did not experience any loss or increase in performance.

Multi Layer Alpha Blending with Depth Bucketing

To make MLAB less dependent on the order in which fragments are generated, we propose a variant that considers a discrete set of depth intervals. We call this approach MLAB with depth bucketing (MLABDB). The general idea underlying MLABDB is to discretize the scene into k disjoint buckets that perform MLAB independently for the corresponding depth interval. Each fragment is thus assigned to a bucket by means of its depth value and merged heuristically into the local corresponding color and transmittance buffer. Since the buckets are already sorted wrt. depth, blending can finally be performed by blending the buckets' values in front-to-back order.

However, only discretizing the scene into buckets of equal intervals produced images with less quality and visible artifacts. MLAB itself is not order-independent and yields different results per pixel depending on the order of fragments for each bucket and pixel, resulting in artifacts. To avoid these artifacts, in MLABDB we segment the scene into two buckets and set the boundaries of the buckets heuristically with respect to opacity (compare Fig. 4).

MLABDB requires two rendering passes to obtain the final color. In the first pass, the boundaries of the two buckets are determined. For each gathered fragment, the first fragment with opacity α greater or equal a user-defined threshold τ_α is maintained. The depth value z_{min} of this

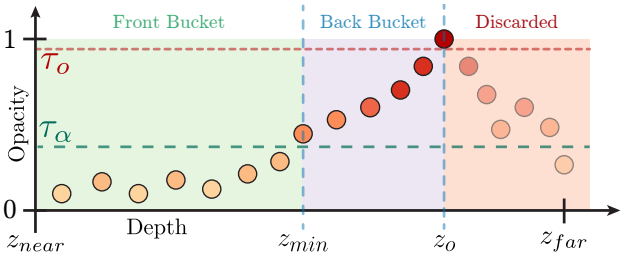


Fig. 4: Fragments with opacity, ordered by depth. MLABDB searches for the first fragment with $\alpha \geq \tau_\alpha$ to obtain depth boundary z_{min} of the front bucket. Back bucket bounds are z_{min} and z_o, with z_o the depth of the first opaque fragment. Fragments with $z > z_o$ are discarded.

fragment represents the upper depth boundary for the first bucket, called front bucket. Since fragments behind opaque lines should not be merged the first opaque fragment with $\alpha \geq \tau_o$ is preserved. Its depth values z_o and z_{min} define the upper and lower boundary of the second bucket, called the back bucket. In the second pass, all incoming fragments are assigned to the corresponding bucket by using their depths, and heuristic merges are performed independently for each bucket. Fragments with a depth value greater than z_o are discarded. For the front bucket, we found that $n = 1$ or $n = 2$ layers were suitable to gather the fragments and avoid order-dependent problems of MLAB, under the assumptions that all fragments of low opacity contribute equally to the image. The back bucket uses a blending array with four or five layers. As demonstrated in Fig. 1c and further evaluated in Sec. 4, MLABDB can, in many cases, considerably improve the quality of MLAB (compare Fig. 5). On the other hand, since it requires more operations, it is slightly less efficient than MLAB. Note that thresholds need to be set carefully, as visual artifacts can occur as seen in Fig. 11(a) and discussed in Sec. 4.5.

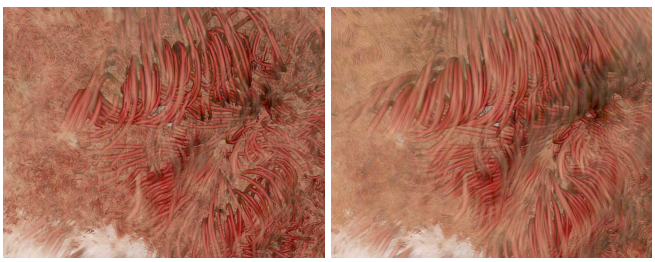


Fig. 5: Difference between MLAB (left) and our modified variant MLABDB (right). MLAB reveals interior lines erroneously due to wrong blending order. MLABDB renders correctly in this scenario.

Moment-Based Order Independent Transparency

Moment-Based Order Independent Transparency (MBOIT) [25] is another variant of transparency rendering techniques with bounded memory and rendering constraints. It builds upon either power moments or trigonometric moments to approximate the transmittance function per pixel in a stochastic way. Power moments are used in statistics to reconstruct or approximate functions such as the mean and standard deviation of arbitrarily sampled random distributions. In addition, MBOIT operates on the logarithm of the

transmittances per fragment to enable order-independent additive blending. The transmittance of the n-th fragment at depth z_f and opacity α_f is given by

$$T(z_f) = \prod_{l=0}^{n-1}(1-\alpha_l), z_l < z_f \quad (1)$$

The absorbance can then be defined in the logarithmic domain as

$$A(z_f) = -\ln T(z_f) = \sum_{l=0}^{n-1} -ln(1-\alpha_l) \quad (2)$$

The absorbance can be interpreted as a cumulative distribution function of the transmittance at each layer, given that for all l translucent fragments it holds that $z_l < z_f$. The distribution can be described as

$$Z := \sum_{l=0}^{n-1} -ln(1-\alpha_l) \cdot \delta_{z_l}, \quad (3)$$

where δ_{z_l} is the Dirac-δ function. Using a power moments generating function b: $[-1,1] \rightarrow \mathbb{R}^{m+1}, b(z) = (1, z, z^2, z^3, \cdots, b^m)^T$, for m power moments the transmittance is given by

$$b := E_z(b) = \sum_{l=0}^{n-1} -ln(1-\alpha_l) \cdot b(z_l). \quad (4)$$

MBOIT requires two passes to compute the final color. In the first pass, the m power moments are computed that are required to reconstruct the transmittance function. The first pass requires storing m floating point values per pixel, we use four in our experiments. In the second rendering pass, the transmittance of each fragment is reconstructed using the pre-computed power moments via

$$T(z_f, b) = \exp(-A(z_f, b)). \quad (5)$$

The real absorbance of the fragment is estimated by computing its lower and upper bounds and interpolating in-between these bounds with an interpolating factor $\beta = 0.1$. This factor was determined by testing multiple values and settling for the one giving the best results. As the quality of reconstruction further degrades with large depth value ranges, the depth values are transformed to logarithmic scale [25]. The final color can then be computed using the total absorbance, stored in the first power moment b_0, and the reconstructed transmittance $T(z_f, b)$ (cf. eq. 2 in [25]).

Münstermann et al. [25] pointed out that this is problematic for scenes with intersecting geometry and large depth ranges, and this turns out to be especially problematic in the situation where many changes in the transparencies along one single view ray occur (see discussion in Sec. 4.5).

3.2 Image-Order

Image-order techniques guarantee exact visibility order of the surface points along the view rays. They utilize a search structure to efficiently find the objects that need to be tested. Therefore, they often come with increased, yet per-frame constant memory requirements. On the other hand, they have unbounded rendering constraints, since the number of ray-object intersection tests depends on the view direction.

Voxel-Based Ray-Casting

VRC is an image-order line rendering technique. It builds upon the voxelization of a line set into a regular voxel grid and performs ray-casting in this grid with analytical ray-tube intersections to correctly blend all intersection points. For discretizing the lines into the voxel grid (i.e., curve voxelization), each line is subdivided into a set of line segments by clipping the line at the voxel boundaries (see Fig. 6). To obtain a compact representation of these segments, their endpoints are quantized based on a uniform subdivision of the voxel faces (i.e., line discretization).

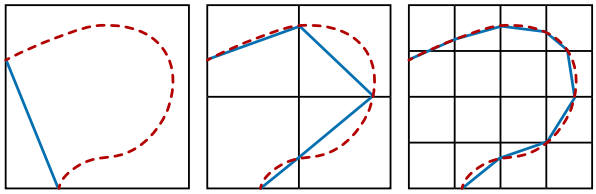


Fig. 6: The original curve (red dotted line) is discretized into a voxel grid of 1, 4, and 16 voxels, respectively, from left to right. Per voxel lines are clipped against the voxel faces and linearly connected. The blue curve represents the approximated original curve at the given grid resolution.

For every pixel, a ray is cast through the voxel grid, going from voxel face to voxel face using a digital differential analyzer algorithm. Whenever a voxel is hit, it is determined how many lines are stored in that voxel. If a voxel is empty, it is skipped; otherwise, the ray is intersected against the tubes corresponding to each line. If multiple intersections with the tubes are found, they are computed and then sorted in place in the correct visibility order. Since tubes can overlap into adjacent voxels, neighboring voxels also need to be taken into account for intersection testing.

A potential weakness of VRC is the approximation quality. Curve voxelization and line quantization introduce an approximation error, which increases with decreasing grid resolution and coarser discretization of voxel faces. Conversely, higher grid resolutions and finer discretizations yield better approximations, but can significantly increase the memory required to store the voxel representation.

OSPRay CPU Ray Tracing

OSPRay is used to evaluate the performance of CPU ray-tracing for transparent line rendering. Within OSPRay, there are three options for representing line primitives. OSPRay's built-in streamlines represent the lines as a combination of analytic cylinder and sphere primitives, suitable for rendering opaque streamlines with a constant radius. For smoother higher-order curves or transparency, OSPRay can also use Embree's built-in Bézier curve primitive directly. Finally, the Generalized Tube Primitive module [13] extends OSPRay's original streamline approach to support varying radii, bifurcations, and correct transparency. The Generalized Tubes module represents the streamlines as a combination of spheres, cylinders and cone stumps, and employs a constructive solid geometry intersection test to ensure correct transparency. Although this CSG-based intersection comes at some cost, it is required to avoid showing interior surfaces from intersections with the constituent primitives.

To render the primitives, we use OSPRay's built-in scientific visualization renderer, which supports illumination effects such as shadows and ambient occlusion.

It is of course also possible to tessellate the tube primitives and render them in OSPRay as a triangle mesh. In our testing, we found that when using a very low-quality tessellation, the triangle mesh outperformed the Generalized Tubes for transparent geometry, due to the removal of the CSG traversal. However, even with a low-quality tessellation, the memory consumed by the triangle mesh is of concern, moreover, tessellating to a level of detail that matches the quality of the Generalized Tubes or Bézier curves will require significantly more primitives, impacting performance. For small- to medium- sized data sets, triangulation may be a reasonable approach.

RTX Ray Tracing

RTX is used to evaluate the performance of GPU ray-tracing for line rendering. On the RTX architecture, dedicated hardware, the RT cores, are used to accelerate the traversal of bounding volume hierarchies—utilizing axis-aligned bounding boxes—and the execution of ray-triangle intersection tests.

As the maximum recursion depth on current RTX hardware (32) is too low for data sets with high depth complexity, we opted for an iterative approach. We also note that a recursive approach is likely to be far more expensive than an iterative one. Our first approach used any hit shaders to accumulate fragments along the rays. However, this can provide only an approximate result, as any hit shaders are not guaranteed to be run in a strict front-to-back order. Thus, we did not pursue this approach further.

Instead, in our experiments we utilize a closest hit shader in combination with a loop in the ray generation shader that blends the fragments returned by it in front-to-back order (cf. Fig. 7). Intersection sorting is thus done entirely by the acceleration structure traversal unit.

The closest hit shader also returns its hit distance along the ray, so that the ray generation shader can start the next ray right after the last hit using a very small offset to avoid intersecting the same primitive again. The loop is terminated by either a sentinel value returned from the miss shader, run when no primitive is hit, or a zero transmittance value. Although iterative next-hit traversals could, in theory, fail to find all intersection hits (see Wald et al. [36]), we have not experienced this problem in our experiments.

The RTX framework can also trace against custom geometry using an intersection shader, which we utilize to perform analytic intersection tests against the tube representations for each line. A ray is first intersected with an infinite tube, and the intersection points are then clipped against the two planes delimiting the tube segment. To correctly interpolate the vertex attributes in the closest hit shader, both planes are intersected with a line parallel to the tube and through the clipped point. The position of the clipped point on this line segment is then mapped to $[0; 1]$ and used as interpolation factor in the closest hit shader.

Interestingly, the analytic ray intersection tests are about a factor of two slower than ray-triangle intersection tests, even though the lower primitive count leads to a significantly smaller memory footprint. The high performance for triangles is likely attributable to the hardware acceleration of triangle intersection testing on the RTX hardware, and do not consider analytical tests in the remainder of the evaluation for RTX.

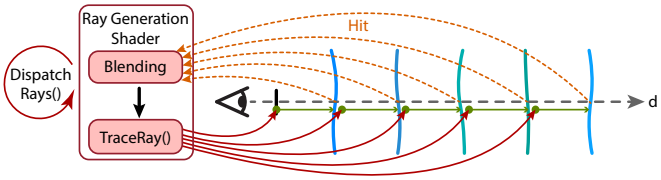


Fig. 7: Illustration of iterative ray-casting using the RTX framework. Blue-colored paths represent line primitives of the data set. At each frame, the ray generation shader is called once and is responsible for iteratively blending over all fragments and issuing new rays (TraceRay()) at intersection point t_i (green arrows). During ray traversal, the intersection point with primitive closest to the viewer is computed. On each hit, color and opacity of line is obtained and sent back to the ray generation shader (orange arrows).

4 Evaluation

We evaluate and compare all selected line rendering techniques regarding memory consumption, performance, and quality. All GPU techniques were run on a standard desktop PC with Intel Xeon processor, 32 GB RAM, and an NVIDIA Geforce TITAN RTX with 24 GB VRAM. CPU ray-tracing was performed on a system with 2 Intel Xeon E5-2640 CPUs at 2.4 GHz and 3.4 GHz boost frequency with 40 CPU threads in total. We used the Vulkan SDK 1.1.129 with the extension VK_NV_ray_tracing to implement RTX ray tracing and conducted the performance tests using the NVIDIA driver 441.87. Both CPU and GPU architectures come at roughly the same price, making the comparison fair in terms of financial investment. Furthermore, all performance measurements (using data sets that fit into memory) were carried out on an NVIDIA Geforce RTX 2060 SUPER with 8 GB VRAM and a single Intel i7-5930K CPU with 12 threads. The performance scale-down compared to the measurements in Sec. 4.4 was roughly a factor of 1.6 – 3 and 2, respectively. All images were rendered at a viewport resolution of 1920×1080 for performance and 1280×720 for image quality. When statistics are given for flights around a data set, the camera parameters were set so that most of the viewport is covered by that data set and the entire viewport is covered in zoom-in scenarios. Ground truth images are generated via DP, yet we do not consider DP any further due to the limitations discussed in the introduction.

4.1 Data Sets

Our experiments were performed on data sets with vastly different numbers of lines and line density. For each data set, we selected meaningful transparency assignments, e.g., by mapping physical parameters along the lines or geometric line curvature to transparency. The following data sets were used:

- **ANEURYSM:** 9,200 randomly seeded streamlines in the interior of an aneurysm [5], and advected up to

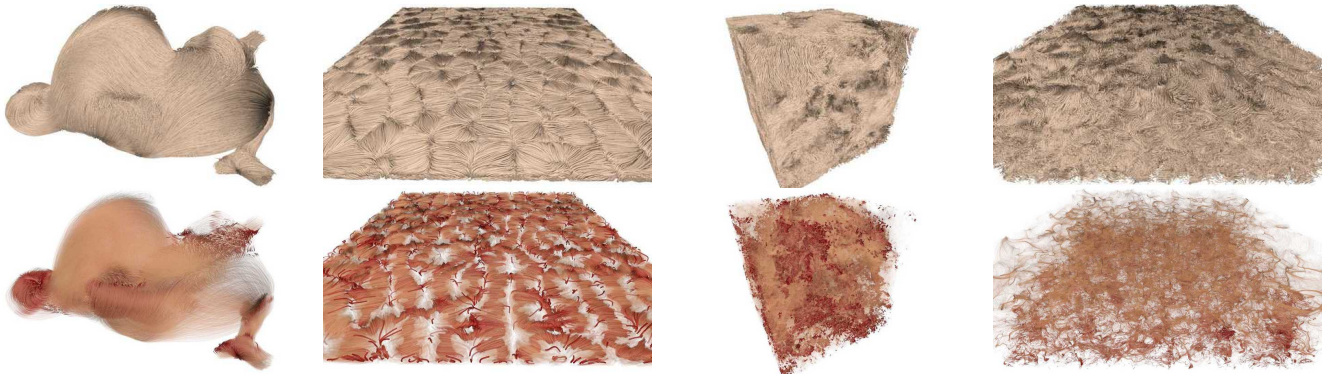


Fig. 8: Data sets used in our experiments. Top: Opaque line rendering. Bottom: Transparent line rendering. From left to right: ANEURYSM, CONVECTION, TURBULENCE, and CLOUDS. Transparency greatly aids the ability to visualize important features in the data.

the vascular wall. Vorticity in the flow field from which the lines were extracted is mapped to line transparency.

- **CONVECTION:** 100,000 short streamlines that were uniformly seeded in a Rayleigh-Bernard convection between a heated bottom wall and a cold top wall [28]. Transparency is assigned according to line curvature.
- **TURBULENCE:** 80,000 long streamlines generated in a forced turbulence field of resolution 1024^3 [2]. Transparency is proportional to the maximum λ_2 vortex measure along a line.
- **CLOUDS:** 400,000 seeded streamlines in a cloud-resolving boundary layer simulation (UCLA-LES, see [34]) using a large eddy simulation (LES). Streamline integration was conducted on a voxel grid with resolution 384×384×130. The magnitude of vorticity along each streamline is mapped to transparency.

Fig. 8 shows all data sets, rendered via opaque and transparent lines. Most data sets are very dense, yet by mapping the selected parameters linearly to transparency, important interior structures can be revealed. Table 1 gives further information on the number of line segments that need to be rendered as tubes, the memory that is required by the initial line representation, and the memory requirements of the internal data representation used by each technique, as explained in the next subsection.

4.2 Data Preparation and Model Representation

All rasterization-based line rendering techniques and RTX render the tubes as triangle meshes. Therefore, each connected sequence of lines first needs to be converted into a set of triangulated tubes that are stitched together to form a closed mesh. This pre-process is performed on the GPU. The meshes are stored in a triangle and shared vertex representation with normal buffer, where each index and vector component is represented by a 32-bit value. In an additional attribute buffer, 16-bit per-vertex attribute values are stored. During rendering, these values are mapped to transparency and color. To construct the tubes, for every vertex shared by two lines, the average of the lines' tangent vectors is computed. At the first and last vertex, respectively, the average is just the tangent of the first and last line segment. Three vertices are generated and placed uniformly on a circle around the vertex in the plane orthogonal to the average tangent and containing this vertex. The three vertices from the line start and end points are then connected to form a closed set of tubes. We use the same circle radius for all tubes. The vectors from the initial vertex to the new ones are used as per-vertex normals. From our experiments, we found that three vertices along a circle radius are sufficient to achieve good results from each view direction.

The resulting buffers are used directly as input for LL, MLAB, MBOIT, and RTX. It is worth noting that all rasterization-based techniques can also generate the polygon models on the fly during rendering in a geometry shader, or use a pixel shader that takes the line information as input and analytically tests for intersections with the corresponding tube. However, since rendering the precomputed geometry is up to a factor of 2 faster, we do not consider on the fly generation in our evaluation.

RTX requires another pre-process to build an AABB hierarchy from the given polygon model. For triangle geometry, the RTX framework supports only a few position formats, and raw data must be converted if it does not already match. For custom geometry, the API requires conservative estimates of the AABB of each primitive. Construction of the AABB hierarchy is performed on the GPU via the API. RTX then generates a tree structure, that is traversed by every ray until reaching the leaf nodes where ray-triangle tests are performed.

Since VRC cannot handle polygon models but tests analytically against the tubes during ray-casting, a voxel-based renderable line representation is first built and uploaded to the GPU. For ANEURYSM and CONVECTION, we used an optimized voxel grid resolution of 113×110×128 (x,y,z-dimension) and 128×8×128, respectively, and a quantization level of 16. For TURBULENCE and CLOUDS, we increased the resolutions to 256^3 and 512^3, respectively, and used a quantization level of 32. Grid resolutions were chosen to reduce the probability that lines fall onto each other and become indistinguishable.

OSP constructs a bounding volume hierarchy using Embree [40]. From the input line data, we build the Generalized Tubes geometry, which consists of a set of analytic spheres, cylinders, and cone stumps, the union of which forms the tube. These primitives are passed to Embree as a user

TABLE 1: Data statistics and memory requirements. Number of line segments in millions (LS), line model size in MB (LM), size of renderable representation in GB (primitive buffers, acceleration structures), and build times in seconds for rasterization-based techniques, OSP, RTX, and VRC.

Data Set	LS	LM	Render. Repr. (GB)				Build Times (s)			
ANEURYSM	2.3	34	0.39	0.38	2.04	0.05	0.8	0.2	0.6	3.6
CONVECTION	9.9	151	1.67	1.62	5.27	0.04	3.2	0.3	2.8	11.1
TURBULENCE	17.5	267	3.01	3.01	9.38	0.50	7.7	1.1	5.1	23.6
CLOUDS	39.6	610	4.63	6.12	14.5	0.89	13.5	4.5	9.5	42.1

geometry, over which it will construct a BVH. As with RTX, the user geometry must provide a conservative bounds estimate to the BVH builder. In our evaluation, we found that Embree's Bézier curves provided better performance for transparent tubes, and in this case, we switch to use Embree's Round Bézier curves, available through OSPRay's "curves" geometry type. Embree then builds a BVH over the curve primitives as before. The curve primitive is built into Embree, and additional optimizations to the BVH quality may be applied during the build, that are not available for user geometry such as our Generalized Tubes. During rendering, Embree traverses packets of rays in SIMD through the BVH until reaching a leaf node, where intersection tests are performed with Generalized Tubes or Bézier curves.

For all data sets and rendering techniques, Table 1 lists the number of line segments to be rendered (LS), the memory requirement to render all line models (LM), the memory requirement of the used renderable representations (primitive buffers and acceleration structure) on the GPU or CPU, and the time required to build these representations. Table 1 indicates that VRC performs better than rasterization-based approaches regarding memory requirement. In general, the polygon model requires much more memory than the voxel grid used by VRC, in some cases more than 10 times. The build times, on the other hand, are about a factor of 4 faster compared to VRC. We attribute this difference in build times to the fact that building the voxel representation requires far more arithmetic and memory scan operations for clipping lines at voxel boundaries, counting how many line segments fall into a voxel, and computing indices into per-voxel memory containers. Rasterization-based approaches, on the other hand, require only simple index arithmetic once the number of lines and the vertices per line is known.

We find that RTX consumes significantly more memory than the other alternatives, partly because of the larger number of triangles that is finally stored in the BVH acceleration structure. In addition to that, space-partitioning schemes for dense line sets become increasingly inefficient and run into the problem of either clipping lines at the boundaries, thus duplicating vertex information, or using overlaps, which significantly increases the number of regions to be tested. OSP achieves the best performance for building the acceleration structures since it considers only the initial line segments and comes with a highly optimized multi-threaded BVH build routine provided by Embree.

4.3 Per-Frame Memory Requirements

To analyze the additional memory consumption of each technique during rendering, we render the models along pre-recorded paths around them, including two zoom-ins. For ANEURYSM and TURBULENCE, the graphs in Fig. 9 show the memory requirements per frame (not including the internal model representation) by each technique.

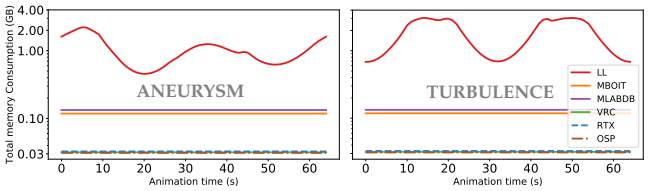


Fig. 9: Memory consumption (in addition to internal model representation) of techniques for different views of ANEURYSM and TURBULENCE.

OSP, RTX, and VRC do not require any additional memory beyond the internal renderable model representation. MLABDB and MBOIT require an additional per-pixel buffer with k=5 or k=4 layers, respectively. MBOIT stores four 32-bit floating point power moments in these layers. The memory consumption of this buffer is negligible compared to that of the renderable representation. LL, on the other hand, needs to keep a buffer with as many entries as the maximum number of fragments that can be generated for any of the possible views. Due to performance issues, this buffer is usually allocated once in a pre-process, using a prescribed maximum number of fragments. The memory consumption of LL can exceed the available GPU memory, especially when rendering at higher viewport resolutions.

In general, if the data set is too large to fit into available GPU memory (4 – 8 GB VRAM on recent commodity graphics cards) rendering the entire line set per frame is not possible anymore. Here, object-order techniques can simply split up line sets into chunks and render those separately at each frame. VRC can split the voxel-based representation into chunks and proceed the same way. RTX can generate chunks of the data and creates an AABB hierarchy for each chunk individually. This requires the traversal of several AABBs at the same time which can lead to a drop in performance. For LL, memory requirements can be reduced by using screen-space partitioning, so that only subsets of fragment lists have to be stored per pixel at once.

4.4 Rendering Performance

Each data set was rendered three times along the pre-recorded flight paths, each time with different transparency settings and zoom levels. In this way, the dependencies between performance and the amount of transparency can be analyzed. We investigate the rendering of opaque lines, semi-transparent lines with an assignment of transparency that gives meaningful results, and lines with constant low transparency (e.g. below 0.15). Even though the latter setting, in general, does not produce meaningful results but mostly blurs out directional information, it is used to demonstrate how either technique behaves in this worst-case scenario. Performance measures are given in Fig. 10.

The rendering times of MLABDB and MBOIT are almost constant for different transparency settings since both techniques always render the entire data set and cannot exploit early-out strategies to skip fragments that are occluded by opaque ones. Rendering performance is mainly affected by

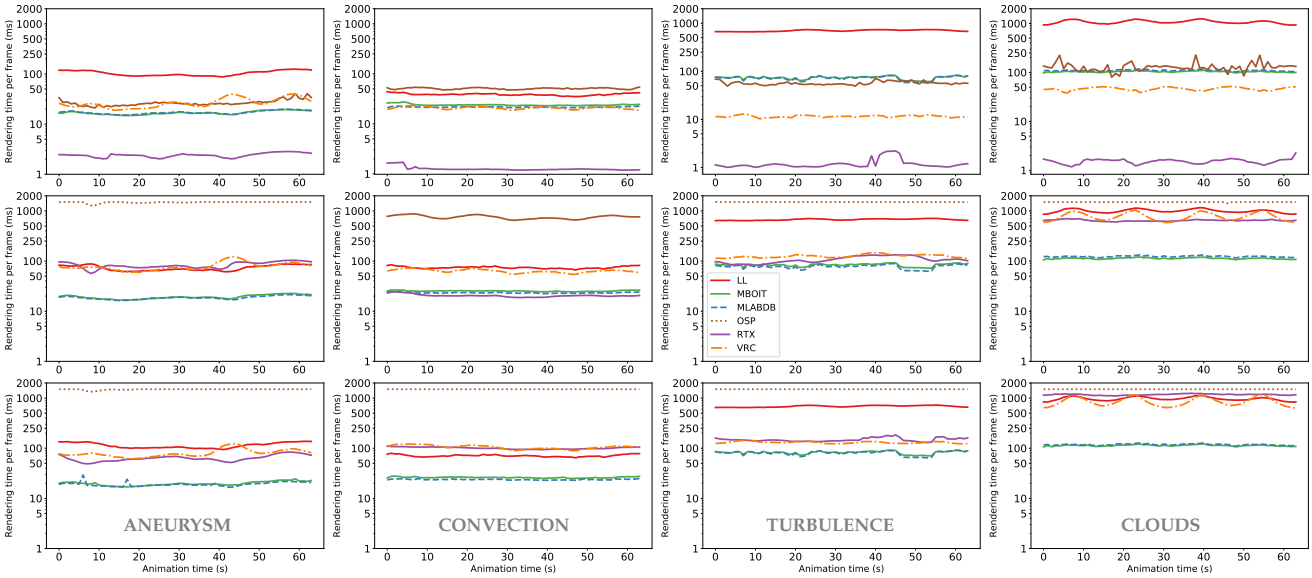


Fig. 10: Rendering performance of all techniques using opaque lines (first row), transparent lines with opaque features of interest (middle row), and highly transparent lines (last row). The 1st, 2nd, 3rd, and 4th columns are for ANEURYSM, CONVECTION, TURBULENCE, and CLOUDS, respectively. For OSP, the line is dotted if rendering time exceeds 1500ms.

the number of lines to be rendered and limited by the polygon throughput of the GPU. Over time, depending on the orientation of lines, different amounts of fragments are generated, which results in slightly varying rendering times along the flight paths. In contrast, the performance of LL is highly dependent on the number of generated fragments. Although LL renders the entire line set once, similar to MLABDB and MBOIT, the generated fragments need to be stored and sorted. Thus, MLABDB and MBOIT render significantly faster than LL, especially for larger data sets. Over time, LL shows performance variations similar to those of MLABDB and MBOIT.

Since image-order techniques can effectively employ early-ray termination, their performance depends strongly on the transparency setting. When highly opaque lines are rendered (see first column in Fig. 10), the performance of RTX, OSP, and VRC is similar or even faster than that of the rasterization-based techniques. The performance of OSP is usually below that of RTX and VRC, as the CPU does not provide any hardware acceleration for ray tracing.

For opaque lines, CONVECTION seems to be an outlier regarding the relative performance differences between OSP and the other techniques. OSP uses Embree's BVH builder, yet since a user geometry is used, Embree is not able to split the geometry to reduce the amount of overlap between BVH nodes. Densely overlapping data sets with long line segments will result in a poorer quality BVH, with more overlap between the nodes. However, RTX uses a triangulated representation and can still perform these spatial splits. For the CONVECTION, since the rolls are laid out in a flat sheet, OSP ends up traversing and intersecting a large amount of the BVH and most of the primitives in aggregate over the image. In the remaining data sets, the higher amount of occlusion helps reduce the amount of traversal needed. Especially on TURBULENCE and CLOUDS, even though there are a large number of lines, OSP only sees the box exterior for opaque lines and traverses very little of the data. Data sets with transparent lines do not benefit from the higher level of occlusion, and a large amount of the data must be traversed.

With increasing transparency, OSP falls behind the other techniques – rendering required more than 2000ms and 6000-10000ms to complete for TURBULENCE and CLOUDS, respectively – as the renderer must now traverse much further into the data. Consequently, more tree-traversal operations and intersections tests need to be computed. Although the same holds for RTX, its rendering times can compete with the performance of approximate techniques for line sets up to 100K, potentially due to the hardware acceleration provided for ray traversal and triangle intersection. The worst-case scenario for OSP and RTX is the highly transparent case, where the majority of view rays have to be traversed until they leave the domain, due to the level of transparency. Thus, the performance drops significantly for large data sets.

It is interesting to note that LL, in many scenarios, can render very efficiently due to the GPU's capability to sort the fragments for many pixels in parallel. The more transparent the fragments are, and the less effectively image-order techniques can exploit early-ray termination, the better the relative performance of LL becomes.

The evaluation shows that, in particular for larger data sets, VRC renders slower than the approximate techniques. This performance difference is mainly due to the traversal of the voxel grid, which is not supported by an acceleration structure to enable empty space skipping, and sorting of multiple ray-tube intersections in the same voxel. The relative performance of VRC, on the other hand, is not much affected by increasing transparency. Although, in this case, many more intersection tests need to be performed, GPU-based voxel traversal can be performed very efficiently and does not impact performance as severely as BVH traversal. VRC outperforms LL by about a factor of 4 and higher for large line sets such as TURBULENCE (cf. last column in

Fig. 10). Again, for ANEURYSM and CONVECTION with many empty regions that need to be traversed on the finest voxel level, the relative performance of VRC compared to the rasterization-based approaches decreases. For CLOUDS, traversing large voxel grids (512^3) greatly reduces VRC's performance and leads to rendering times similar to LL when looking from a diagonal angle into the line set.

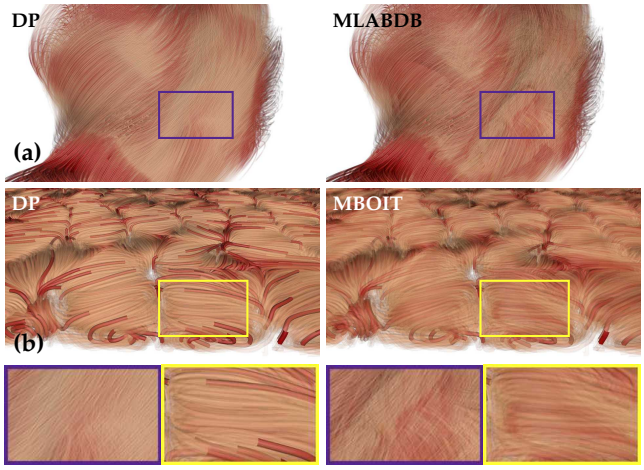
4.5 Image Quality

LL, VRC, OSP, and RTX simulate the effect of transparency accurately. VRC introduces errors due to the voxelization and quantization of lines into a regular voxel grid; however, the visibility order of lines in the grid is handled correctly. In the worst-case, multiple lines can fall on top of each other in the grid, resulting in an incorrect blending order. In our experiments, we did not perceive visual artifacts caused by this effect.

Inaccuracies in MLAB are caused by lines that are not rendered in correct visibility order, and that are merged heuristically using a limited number of transmittance layers. For scenes with high depth complexity, and even when low transparency is used, the accumulation of errors leads to visible artifacts. Most prominent are errors caused by incorrect merging of fragments with high opacity, i.e., when two such fragments are merged in the wrong order into the same transmittance layer (Fig. 1(c) and Fig. 5). If lines are by chance rendered in the correct visibility order, or few opaque fragments are blended into different transmittance layers, MLAB can nevertheless generate accurate results (see Fig. 1(a) for an example). The view-dependent nature of MLAB, i.e., errors can suddenly appear or disappear depending on whether the rendering order matches the current visibility order, makes it less time coherent.

MLABDB can avoid the order-induced artifacts introduced by single MLAB when rendering opaque or nearly opaque lines. However, as mentioned in Sec. 3.1, thresholding has to be done carefully. In Fig. 11(a), bucketing leads to hard cuts in color, revealing the depth segmentation of the line set. This artifact occurs primarily when using transfer functions with sudden opacity changes, which conflicts with MLABDB's assumption of many transparent layers occluding opaque ones. Even though these artifacts can be avoided by manually adapting the threshold for the front bucket according to the selected transfer function, this kind of user intervention is not practical in general.

MBOIT replaces the transmittance function along a line of sight by a low-frequency approximation. Thus, two major types of artifacts can occur: First, as shown in Fig. 1(b) and Fig. 11(b), the accumulated opacity of multiple transparent lines is either highly overestimated or underestimated. These over- and under-estimations can lead to misinterpretations of the visualization, as translucent or opaque lines can appear prominent or be missing in the final image. These errors are due to sudden changes in opacity when mapping the importance of features with step-functions, meaning that MBOIT cannot accurately handle hard transitions in the mapped opacity. Second, since a low-frequency approximation is used, MBOIT tends to smooth out the transmittance distribution across the pixel image. As shown in Fig. 1(d) and in Fig. 11(b), sharp edges between lines with higher opacity are not preserved. This effect, in particular, can hamper a more detailed analysis of the directional structure of important lines, and it tends to smooth out the directional information in important regions.



(a) PSNR = 31.15, SSIM = 0.841 **(b)** PSNR = 29.80, SSIM = 0.82

Fig. 11: **(a)** Ground truth (left) and MLABDB (right). MLABDB can produce hard visual "cuts" due to depth segmentation. **(b)** Ground truth (left) and MBOIT (right). MBOIT tends to blur out features and underestimate opacity, erroneously revealing interior lines.

4.6 Quantitative Assessment

To further quantify the error that is introduced by the different line rendering techniques, all data sets are rendered along the pre-recorded flight paths using the transparency settings described before. Lines are illuminated by a headlight and colored via Blinn-Phong shading. For each image, the Peak-Signal-To-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [44] are computed between the ground truth rendering using DP, and MLABDB, MBOIT, and VRC, respectively. We do not consider LL, RTX, and OSP, since they correctly simulate transparency. For each setting, the accuracy measurements are plotted as line graphs over time. Renderings using semi-transparent settings are shown in Fig. 12, all other settings are shown in Fig. A.1. An in-detail discussion and all results are provided in Appendix A.

4.7 Visual Quality vs. Per-Pixel Error

Interestingly, when looking at images where the SSIM and PSNR values show a lower quality of VRC and higher quality of MBOIT and MLABDB, these differences are not reflected in the visual quality of the results. Since transparency is handled correctly by VRC, even the close-up views appear similar to the ground truth, and even visible artifacts introduced by the alternative methods do not manifest.

In this section, we analyze in further detail the relations between visual image quality and per-pixel error. For each data set, we analyze two views: one view where all techniques come visually close to the ground truth while producing only a small number of pixel-wise errors (case A), and one that reveals typical artifacts of MBOIT and

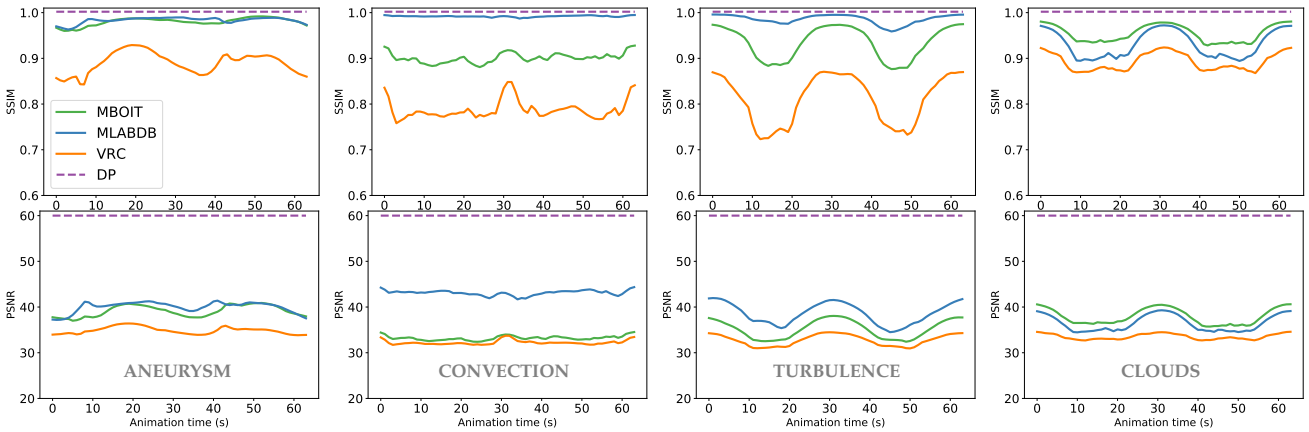


Fig. 12: Error metrics of all techniques using transparent lines with opaque features of interest. The 1st, 2nd, 3rd, and 4th columns are for ANEURYSM, CONVECTION, TURBULENCE, and CLOUDS, respectively. The First row shows SSIM for each technique, the second shows PSNR. A higher value is better.

MLABDB using a most meaningful transparency settings with sharp transitions and alternating high transparency and opacity (case B).

Besides an image-to-image comparison, the analysis is additionally supported by visualizations of the absolute per-pixel differences to the ground truth. These plots are grayscale images with black regions highlighting large color differences. Significant differences in images are marked by colored rectangles and supported by close-up views. Image comparisons of all cases (Fig. A.x) are given in Appendix A.

Fig. A.2 depicts a scenario where the viewer is looking through the entire ANEURYSM data set with alternating opaque lines (red-colored) and transparent lines (orange-ocher). For case A (cf. Fig. A.2(a), corresponding to frame 5 of the first column in Fig. 12), we observe that all techniques are close to the ground truth image with the best image produced by MLABDB. MBOIT overestimates the transparency of few transparent fragments occluding opaque lines highlighted by per-pixel error plots (cf. black regions in Fig. A.2(a) (MBOIT)). VRC produces high pixel errors in regions close to the viewer since line inaccuracies affect larger areas of pixels. However, the quality of VRC becomes better with larger distance to the camera, leading to results indistinguishable from the ground truth. Wrt. case B (cf. Fig. A.2(b)), the quality of both MBOIT and MLABDB is worse than VRC. MBOIT struggles to approximate sharp transitions in transparency leading to high over- and under-estimations, whereas MLABDB is not able to correctly merge opaque fragments. Bucketing is impossible here, leading to visual artifacts. However, VRC remains stable and, besides line inaccuracies, is very close to the ground truth.

Another example is given in Fig. A.3 when looking from a diagonal angle into the entire CONVECTION line set, with the number of transparent lines increasing with the distance to the viewer. For case A (cf. Fig. A.3(a)), all techniques are visually close to the ground truth with MLABDB working best. Here, MBOIT exhibits small errors in transmittance approximation, as many opaque lines are occluded by transparent ones. These errors propagate towards the background as the number of fragments increases with distance, leading to further image quality degradation. With opaque lines more present in this case, VRC produces a number of wrong line silhouettes due to curve discretization, emphasized in per-pixel error plots along the line edges. For case B, the quality of both MLABDB and MBOIT decreases with larger distance (cf. Fig. A.3(b)). In particular, MLABDB produces more per-pixel color inconsistencies, depicted by high noise in error plots, toward the background as more and more fragments are merged incorrectly. On the other hand, MBOIT has difficulty coping with sharp transitions between transparent and opaque fragments. Interestingly, the image quality of VRC is independent of the viewer's distance or angle, and line inaccuracies do not accumulate with increasing distance.

Fig. A.4 demonstrates the differences of image quality for zoom-out and close-up views. Here, case A (cf. Fig. A.4(a)) represents a zoom-out view that corresponds to frame eight of the third column in Fig. 12. All techniques are able to properly render TURBULENCE and are visually indistinguishable from the ground truth, although MLABDB shows some weaknesses in rendering transparent regions due to incorrect fragment merges. However, these pixel errors hardly affect the overall quality of the image. Wrt. VRC, line inaccuracies are not present here since lines are highly transparent and line edges are not emphasized. Case B (cf. Fig. A.4(b)) shows the impact of zoom-ins on the quality of all techniques. MLABDB properly renders opaque lines (red-colored tubes), but incorrectly merges fragments in regions with a large number of highly transparent lines, leading to a wrong colored region (orange instead of ocher) after blending. MBOIT is able to approximate transmittance in transparent regions but fails to display sharp opaque lines where opaque and transparent lines are close by, leading to blurred outline structures in the final image. Per-pixel error plots for VRC reveal some line inaccuracies but demonstrate that, overall, a good image quality is achieved by VRC even for this large data set.

The last example demonstrates the impact of large, dense line data with many layers per pixel (> 10000 at maximum), using high transparency in Fig. A.5(a) and semi-transparent opacity settings in Fig. 13 and Fig. A.5(b). In the first scenario, all techniques are able to properly render the data set. However, MLABDB produces a few wrongly colored features due to false fragment merging. MBOIT works prop-

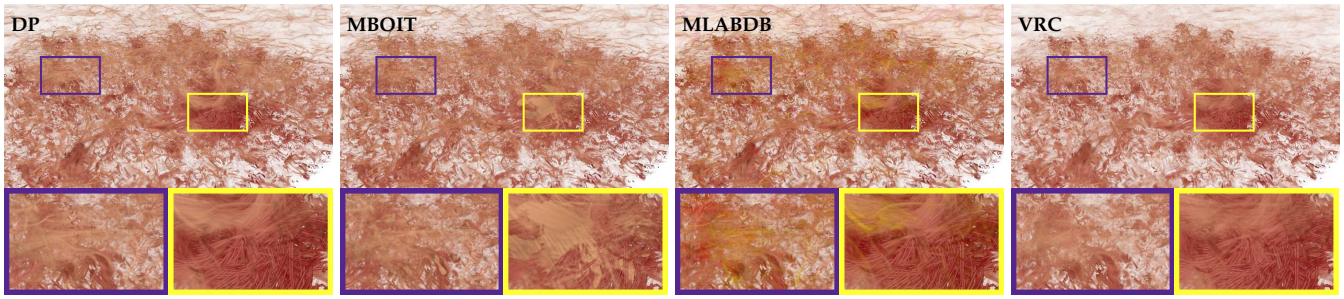


Fig. 13: From left to right: CLOUDS rendered with DP, MBOIT, MLABDB, VRC. Images show renderings using sharp transitions between low and high transparency. Blue and yellow rectangles highlight differences between techniques; close-up views are shown below each image.

erly for this case, and produces few over- or underestimations. VRC, on the other hand, tends to underestimate the actual transmittance as some lines fall into the same cell (disadvantage of VRC explained in Sec. 3.2). Weaknesses of all techniques are even more pronounced in the second scenario; for example, there are more wrongly colored features produced by MLABDB. Due to its line approximation, VRC also struggles to fully reconstruct the actual transmittance. MBOIT works best for CLOUDS, as there are only a few overestimates of the actual transmittance.

Since MLABDB and MBOIT do not depend on any primitive type, we also investigated the image quality of our proposed approximate techniques when rendering transparent triangle surfaces and point cloud data. The visualization of these types of data with transparency can, similar to line sets, lead to complex rendering scenarios with many transparent layers and multiple occlusions. However, for those data types, MLABDB and MBOIT showed similar characteristics as for line sets during rendering (see Appendix B).

In summary, although all techniques show weaknesses some weaknesses in some cases resulting in pixel-errors, they are able to render transparent data sets with high depth complexity at high image quality. Moreover, OSP, RTX, and VRC are temporally stable for all transparency settings and data sets. VRC, despite line inaccuracies, comes very close to the ground truth.

5 Discussion

In the following sections, we discuss the major characteristics of all rendering techniques, as given in Tab. 2 and also present the outcome of an informal user study to shed light on the perception of the errors that are introduced by object-order approximate techniques.

5.1 Object-Order

Equipped with dedicated GPU-friendly sorting algorithms and data structures, LL shows good rendering performance for all but the largest data sets. LL was never slower than a factor of 4-5 compared to approximate techniques for small data sets. For dense data sets with high depth complexity of more than a thousand layers, however, the required GPU buffers can easily exceed available GPU memory, especially for resolutions above 1080p.

Approximate rasterization-based techniques are very fast, work with rendering constraints, and support rendering on hardware with bounded memory. Rendering constraints for MLABDB and MBOIT involve, for each pixel, the maximum number of fragments stored or model parameters to approximate the transmittance function. These constraints implicitly keep the memory consumption constant over time.

MLABDB and MBOIT provide good quality in many real-world scenarios, particularly in scenes where features of interest are rendered opaque and remaining lines are mapped to high transparency. Introduced artifacts are often subtle and local, yet in some views they can cause artifacts that even give a wrong understanding of the line structures. Approximate techniques also fail to maintain time coherence, as their per-pixel rendering outcome is dependent on the transmittance function along each pixel. These errors are especially frequent if high-frequency transfer functions for transparency are used, which can lead to distracting rendering artifacts. Although approximation errors can be reduced by, i.e., increasing the number of nodes of MLABDB's blending arrays (in example to more than 15 layers) or the number of power moments, such settings considerably reduce rendering performance and increase memory consumption.

Informal User Study

In a simple user study, users were asked to give their assessment of the quality differences between MLABDB and MBOIT, to further shed light on the perceptional differences between these approximate variants. We recruited 24 participants, comprised of 19 computer science students and 5 computer science PhD students, all having a background in computer graphics. The participants were selected to have no color vision deficiency. The students were exposed to the application of line rendering for the first time. None of them knew the visualized vector fields and line sets beforehand. The study was performed using the desktop PC described above. We showed the users the tool and let them work with a data set not included in the study. Then, we performed two different experiments:

- To each user, we showed a sequence of eight tri-sets of renderings: the ground truth image first, and then the same view rendered with MLABDB and MBOIT (showing their typical artifacts) side by side.

TABLE 2: Comparison table of all techniques wrt. type of order, render accuracy, performance, render constraints (RenderC), support of bound memory (BoundM), temporal coherency (Temp), changes in color along each pixel (ChColor), dense (large) data sets (Dense), semi-transparent transparency settings (STrans), and support of global illumination (GI) effects. Symbols + and – represent high or low performance/quality, respectively (two repeating symbols indicate very high / very low performances). / represents ratings for opaque / semi-transparent (left) and highly transparent lines (right).

Technique	Order	Accuracy	Perform.	RenderC	BoundM	Temp	ChColor	Dense	STrans	GI
DP	Object	Exact	– –	✘	✔	✔	+ +	– –	+ +	✘
LL	Object	Exact	+/–	✘	✘	✔	+ +	– –	+ +	✘
MLABDB	Object	Approx.	+ +	✔	✔	✘	–	–	+	✘
MBOIT	Object	Approx.	+ +	✔	✔	✘	–/+	+	–	✘
VRC	Image	Approx.	+/–	✔	✔	✔	+ +	–	+ +	✔
OSP	Image	Exact.	+/– –	✘	✘	✔	+ +	– –	+/–	✔
RTX	Image	Exact	+ +/–	✘	✘	✔	+ +	+/–	+ +	✔

- Each user carried out three interactive sessions with two data sets, one minute each. First, the same data set was visualized, starting with MLABDB and then using MBOIT; then the experiment was repeated in reverse order using a different data set.

Users were then asked to rate the visual quality of the still images and the animations. For both experiments, users could select either MBOIT or MLABDB as the best, or "undecided". We had three renderings where 70% of the users selected MBOIT, three renderings where 63% selected for MLAB, and two renderings where 65% of the users were undecided. In addition to the comparison of MBOIT and MLABDB, we asked the users to rate the image quality of still images as "no difference" (good) to the ground truth, "acceptable" (acc), or "non-acceptable" (non-acc). For MBOIT, 41% rated it as good, 46% as acceptable, and 13% as non-acceptable. For MLABDB, 44% rated it as good, 32% as acc, and 24% as non-acc. To assess the image quality over time for each technique, we asked users in a second experiment to rate the quality of the videos similar to image quality. For MBOIT, 62% rated it as good quality, 35% as acceptable quality, and 3% as non-acc. For MLABDB, 45% of users rated it as good, 47% as acc, and 8% as non-acc.

About why they scored the renderings as good, acceptable, or non-acceptable, users mentioned that wrong color outputs and rendering order artifacts (line features falsely hidden) were the most disturbing, as well as the hard and abrupt changes produced by MLABDB or suddenly disappearing features (referred to as "popping" or "flickering") produced by MBOIT during an animation. Some users argued that sometimes even a wrong impression was suggested by both techniques in still images (see Fig. 1 and Fig. 11). In animations, most users did not consider these effects as negative, due to the possibility to interact with the data and, thus, reveal missing information.

To conclude, MLABDB can be recommended for non-dense data or a small number of different colors per pixel while using semi-transparent transparency settings. MBOIT can be suggested for large and dense data with a high variation of color per pixel and transparency settings with smooth transitions. Also, both techniques can be applied to triangular meshes or point cloud data (cf. Fig. B.1 and Fig. B.2, Appendix B) since rasterization-based approaches operate on any input geometry.

5.2 Image-Order

Image-order techniques should be preferred when rather opaque structures are rendered since they can effectively employ early ray termination. If transparency is used too aggressively, the time required to traverse the acceleration structures can increase significantly. The run-time performance varies strongly depending on the selected view and does not scale well with an increasing number of transparent layers.

In general, RTX performs better than OSP for all transparency settings, but OSP required less memory for acceleration structures and less time than RTX to complete builds. Surprisingly, our RTX solution was able to achieve real-time rendering performance for all line data sets, including CLOUDS for semi-transparent settings. For opaque and semi-transparent settings, its rendering times were superior to VRC and OSP. In comparison to OSP, best rendering-times were achieved with VRC and RTX throughout all transparency settings. VRC's performance was slightly superior to RTX for large data sets rendered with high transparency. Results produced by VRC are hardly distinguishable from the ground truth, especially in scenes where the camera is far from the data set, or the entire data set is seen at once through the current viewport (zoomed-out views).

In terms of memory consumption, VRC is recommended if memory is limited due to rendering constraints, which includes the finite size of the voxel grid, a constant line quantization level, and a fixed number of lines covered per cell. Although OSP is generally limited by the amount of RAM, RTX requires more than twice as much VRAM as the memory size of the model's renderable representation to build acceleration structures and usually requires 3 times more memory to render the models. Both OSP and RTX currently support only 32-bit integer values to address primitives on the hardware. As such, large data sets must be chunked beforehand into 4GB or less to be rendered using these methods.

6 CONCLUSION AND OUTLOOK

In this work, we have discussed and analyzed different rendering techniques for large 3D line sets with transparency wrt. memory consumption, performance, and quality. The major findings of our study are that a) approximate techniques can give acceptable quality at high speed and low

memory consumption in many use cases, and b) ray-based approaches offer high quality and often at speeds similar to approximate techniques, besides the most extreme cases with overall low transparency. On the other hand, these techniques can require huge memory resources and considerable pre-processing time.

However, regardless of the technique used, transparent line renderings likely fail to communicate spatial relations when large numbers of transparent lines are rendered. In these cases, global illumination effects such as soft shadows and ambient occlusion can help to significantly improve the user's perception [15]. Such effects can be integrated in a straightforward way into image-order approaches by tracing secondary rays. The integration into object-order approaches is more difficult, and can be achieved only with substantial algorithmic changes, and changes to the data structures used. If high-quality rendering for large line sets is desired, we believe that image-order approaches should be favored over object-order approaches.

With the current power of RTX GPU hardware, it will be interesting in the future to combine both transparency rendering and global illumination effects to enhance the visual perception of complex data. Further user studies have to be conducted to shed light on the question of whether transparency rendering of large, dense line data sets is beneficial to the user's perception, or hampers interpretation of trends in the data, and how this may interact with global illumination effects. In terms of interpretation of dense line sets, it would also be interesting to compare transparency rendering techniques with approaches that heuristically filter line sets and render features-of-interest completely opaque.

7 Acknowledgements

We would like to thank Max Bandle and Mathias Kanzler, Technical University of Munich, for their support concerning the efficient implementation of per-pixel fragment sorting and voxel-based line rendering, respectively. Thanks to Tobias Günther from ETH Zürich for providing access to the cloud simulation data which is provided by DKRZ and MPI-M. We thank Tony Saad and Josh McConnell at the University of Utah CCMSC for the Uintah simulation. The Richtmyer-Meskhov data is courtesy of Lawrence Livermore National Laboratory.

This evaluation study has been done within the subproject "Visualization of coherence and variation in meteorological dynamics" of the Transregional Collaborative Research Center SFB/TRR 165 "Waves to Weather" funded by the German Research Foundation (DFG).

References

[1] T. Aila and S. Laine, "Understanding the efficiency of ray traversal on gpus," in *Proc. High-Performance Graphics*, 2009, pp. 145–149.

[2] H. Aluie, G. Eyink, E. Vishniac, K. Kanov, R. Burns, C. Meneveau1, A. Szalay, and S. Chen, "Forced mhd turbulence data set," turbulence.pha.jhu.edu/docs/README-MHD.pdf, 2013, [Online; accessed 21-March-2019].

[3] I. Baeza Rojo, M. Gross, and T. Guenther, "Fourier opacity optimization for scalable exploration," *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2019.

[4] L. Bavoil, S. P. Callahan, A. Lefohn, J. a. L. D. Comba, and C. T. Silva, "Multi-fragment effects on the gpu using the k-buffer," in *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, ser. I3D '07. New York, NY, USA: ACM, 2007, pp. 97–104.

[5] G. Byrne, F. Mut, and J. Cebral, "Quantifying the large-scale hemodynamics of intracranial aneurysms," *American Journal of Neuroradiology*, vol. 35, no. 2, pp. 333–338, 2014.

[6] L. Carpenter, "The a -buffer, an antialiased hidden surface method," *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, pp. 103–108, Jan. 1984.

[7] M. Ciura, "Best increments for the average case of shellsort," in *Proceedings of the 13th International Symposium on Fundamentals of Computation Theory*, ser. FCT '01. London, UK, UK: Springer-Verlag, 2001, pp. 106–117.

[8] C. Crassin, F. Neyret, S. Lefebvre, and E. Eisemann, "Gigavoxels: Ray-guided streaming for efficient and detailed voxel rendering," in *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games*, ser. I3D '09. New York, NY, USA: ACM, 2009, pp. 15–22.

[9] E. Enderton, E. Sintorn, P. Shirley, and D. Luebke, "Stochastic transparency," in *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '10. New York, NY, USA: ACM, 2010, pp. 157–164.

[10] C. Everitt, "Interactive order-independent transparency," *NVIDIA Corporation*, vol. 2, 10 2001.

[11] S. Grajewski, "Fragment shader interlock," www.khronos.org/registry/OpenGL/extensions/ARB/ARB_fragment_shader_interlock.txt, 2015, [Online; accessed 07-December-2019].

[12] T. Günther, C. Rössl, and H. Theisel, "Opacity optimization for 3d line fields," *ACM Trans. Graph.*, vol. 32, no. 4, Jul. 2013.

[13] M. Han, I. Wald, W. Usher, Q. Wu, F. Wang, V. Pascucci, C. D. Hansen, and C. R. Johnson, "Ray Tracing Generalized Tube Primitives: Method and Applications," *Computer Graphics Forum*, 2019.

[14] J. Jansen and L. Bavoil, "Fourier opacity mapping," in *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '10. New York, NY, USA: ACM, 2010, pp. 165–172.

[15] M. Kanzler, M. Rautenhaus, and R. Westermann, "A voxel-based rendering pipeline for large 3d line sets," *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2018.

[16] M. Kern, "Large 3d line sets for opacity-based rendering," Feb. 2020, [Stored and published by Zenodo].

[17] M. Kern and C. Neuhauser, "chrismile/PixelSyncOIT: CPU and GPU algorithms for large space-filling 3D line sets with transparency," github.com/chrismile/PixelSyncOIT, Feb. 2020, [Stored and published by Zenodo; accessed 05-February-2019].

[18] S. Laine and T. Karras, "Efficient sparse voxel octrees," in *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, 2010, pp. 55–63.

[19] K. Lawonn, T. Günther, and B. Preim, "Coherent View-Dependent Streamlines for Understanding Blood Flow," in *EuroVis - Short Papers*. The Eurographics Association, 2014.

[20] T. Maack and M. Kern, "Rtx ray tracing of transparent 3d line sets," Feb. 2020, [Stored and published by Zenodo].

[21] M. Maule, J. a. Comba, R. Torchelsen, and R. Bastos, "Hybrid transparency," in *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '13. New York, NY, USA: ACM, 2013, pp. 103–118.

[22] M. Maule, J. a. L. D. Comba, R. P. Torchelsen, and R. Bastos, "Technical section: A survey of raster-based transparency techniques," *Comput. Graph.*, vol. 35, no. 6, pp. 1023–1034, Dec. 2011.

[23] M. McGuire and L. Bavoil, "Weighted blended order-independent transparency," *Journal of Computer Graphics Techniques (JCGT)*, vol. 2, no. 2, pp. 122–141, December 2013.

[24] M. McGuire and M. Mara, "Phenomenological transparency," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 5, pp. 1465–1478, May 2017.

[25] C. Münstermann, S. Krumpen, R. Klein, and C. Peters, "Moment-based order-independent transparency," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 1, no. 1, pp. 7:1–7:20, May 2018.

[26] NVIDIA, "Nvidia rtx ray tracing developer documentation," developer.nvidia.com/rtx/raytracing, 2018, [Online; accessed 21-March-2019].

[27] S. Oeltze, D. J. Lehmann, A. Kuhn, G. Janiga, H. Theisel, and B. Preim, "Blood flow clustering and applications invirtual stenting of intracranial aneurysms," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 5, pp. 686–701, May 2014.

[28] A. Pandey, J. D. Scheel, and J. Schumacher, "Turbulent superstructures in rayleigh-bénard convection," *Nature Communications*, vol. 9, no. 1, p. 2118, 2018.
[29] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, and M. Stich, "Optix: A general purpose ray tracing engine," *ACM Transactions on Graphics*, August 2010.
[30] N. Sakamoto and K. Koyamada, "Stoachastic approach for integrated rendering of volumes and semi-transparent surfaces," in *Proceedings - 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, SCC 2012*, 11 2012, pp. 176–185.
[31] M. Salvi, J. Montgomery, and A. Lefohn, "Adaptive transparency," in *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*, ser. HPG '11. New York, NY, USA: ACM, 2011, pp. 119–126.
[32] M. Salvi and K. Vaidyanathan, "Multi-layer alpha blending," in *Proceedings of the 18th Meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '14. New York, NY, USA: ACM, 2014, pp. 151–158.
[33] D. L. Shell, "A high-speed sorting procedure," *Commun. ACM*, vol. 2, no. 7, pp. 30–32, Jul. 1959.
[34] B. Stevens, "Introduction to ucla-les," www.mpimet.mpg.de/fileadmin/atmosphaere/herz/les_doc.pdf, 2013, [Online; accessed 01-December-2019].
[35] I. Wald, G. Johnson, J. Amstutz, C. Brownlee, A. Knoll, J. Jeffers, J. Gunther, and P. Navratil, "Ospray - a cpu ray tracing framework for scientific visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 931–940, 2017.
[36] I. Wald, J. Amstutz, and C. Benthin, "Robust iterative find-next-hit ray traversal," in *Proceedings of the Symposium on Parallel Graphics and Visualization*, ser. EGPGV '18. Goslar Germany, Germany: Eurographics Association, 2018, pp. 25–32.
[37] I. Wald, T. Ize, A. Kensler, A. Knoll, and S. G. Parker, "Ray tracing animated scenes using coherent grid traversal," *ACM Transactions on Graphics*, pp. 485–493, 2006, http://doi.acm.org/10.1145/1141911.1141913.
[38] I. Wald, G. P. Johnson, J. Amstutz, C. Brownlee, A. Knoll, J. Jeffers, J. Günther, and P. Navrátil, "OSPRay – A CPU Ray Tracing Framework for Scientific Visualization," *IEEE Transactions on Visualization and Computer Graphics*, 2017.
[39] I. Wald, A. Knoll, G. P. Johnson, W. Usher, V. Pascucci, and M. E. Papka, "Cpu ray tracing large particle data with balanced p-k-d trees." in *IEEE SciVis*, 2015.
[40] I. Wald, S. Woop, C. Benthin, G. S. Johnson, and M. Ernst, "Embree: A kernel framework for efficient cpu ray tracing," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 143:1–143:8, Jul. 2014.
[41] S. Woop, G. Marmitt, and P. Slusallek, "B-KD Trees for hardware accelerated ray tracing of dynamic scenes," in *GH '06: Proceedings of the 21st ACM SIGGRAPH/EUROGRAPHICS symposium on Graphics hardware*. New York, NY, USA: ACM, 2006, pp. 67–77, http://doi.acm.org/10.1145/1283900.1283912.
[42] C. Wyman, "Stochastic layered alpha blending," in *ACM SIGGRAPH 2016 Talks*, ser. SIGGRAPH '16. New York, NY, USA: ACM, 2016, pp. 37:1–37:2.
[43] J. C. Yang, J. Hensley, H. Grün, and N. Thibieroz, "Real-time concurrent linked list construction on the gpu," in *Proceedings of the 21st Eurographics Conference on Rendering*, ser. EGSR'10. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2010, pp. 1297–1304.
[44] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

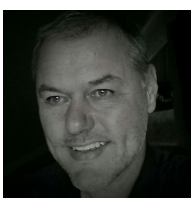
Michael Kern is a PhD candidate at the Computer Graphics and Visualization Group at the Technical University of Munich (TUM). He obtained his M.Sc. in computer science from TUM in 2016. He did an 6-months internship at the Scientific Computing and Imaging Institute at the University of Utah in 2015 with focus on the visualization of biological data. His general research interests include scientific visualization, ensemble uncertainty analysis, and GPU computing.

Christoph Neuhauser is a graduate research assistant at the Computer Graphics and Visualization Group at the Technical University of Munich. He received his B.Sc. in computer science from TUM in 2019. Major interests in research comprise scientific visualization and real-time rendering.

Torben Maack is an undergraduate student at the Technical University of Munich. He received his B.Sc. in computer science at the TUM in 2019. His research interests incorporate areas in real-time rendering and high performance computing.

Mengjiao Han is a graduate research assistant at the Scientific Computing and Imaging Institute at the University of Utah. She obtained her B.Sc. in electronic bio-engineering from the Beijing University of Post and Telecommunication in 2010. She finally received her M.Sc. in computer science in 2014 at the University of Delaware. Her major interests in research are scientific data visualization for large-scale data, parallel rendering and real-time ray tracing.

Will Usher is a graduate research assistant at the Scientific Computing and Imaging Institute at the University of Utah. He finished his B.Sc. in physics with a minor in computer science at the University of California, Riverside in 2014. His research interests cover a range of areas in real-time ray tracing, physically based rendering, virtual reality, and high performance GPU computing.



Rüdiger Westermann studied computer science at the Technical University Darmstadt and received his Ph.D. in computer science from the University of Dortmund, both in Germany. In 2002, he was appointed the chair of Computer Graphics and Visualization at TUM. His research interests comprise scalable data visualization and simulation algorithms, GPU computing, real-time rendering of large data, and uncertainty visualization.

APPENDIX A
QUANTITATIVE ASSESSMENT OF IMAGE QUALITY

PSNR measures the ratio between the power of a signal and the power of noise distorting the signal in decibel (dB). It is computed on a logarithmic scale as

$$\mathrm{PSNR}(O_t^R, O_t^G) = -10\log_{10}(||O_t^R - O_t^G||_2^2), \qquad (1)$$

where O_t^R and O_t^G are the rendering results and the ground truth image, respectively. The PSNR for the ground truth is clamped to a high value of 60dB.

SSIM is a perception-based model to measure the similarity of two images wrt. structural information. In contrast to PSNR, SSIM highlights regions of structural (dis-) similarity and enables a detailed analysis of rendering stability and approximation errors wrt. transparency and fragment depth. SSIM is defined as

$$\mathrm{SSIM}(O_t^R, O_t^G) = \frac{(2\mu_R\mu_G + c_1)(2\sigma_{R,G} + c_2)}{(\mu_R^2 + \mu_G^2 + c_1)(\sigma_R^2 + \sigma_G^2 + c_2)}, \qquad (2)$$

where μ_R and μ_G are the average values of O_t^R and O_t^G, σ_R^2 and σ_G^2 are the variances of O_t^H and O_t^G, $\sigma_{R,G}$ is the covariance between O_t^H and O_t^G, and c_1 and c_2 are two small constants to avoid division by zero. We compute SSIM pixel-wise using a Gaussian sub-window of 11x11 to weight surrounding pixels. The SSIM value of an image (1 for the ground truth) is the mean over all pixel-wise SSIM values.

The SSIM and PSNR plots in Fig. A.1 indicate that MBOIT and MLABDB consistently achieve high image quality. For the smaller data sets, the quality of both techniques doesn't seem to be strongly view dependent, yet during close-ups (cf. frames $10-20$ and $40-50$) the quality decreases significantly when rendering large line sets like TURBULENCE and CLOUDS. When zooming into the data, artifacts along a few pixels are now spread across an ever larger area in pixel space, resulting in decreasing SSIM and PSNR values.

For highly opaque lines, MLABDB always comes very close to the ground truth. This result shows that discarding all transparent fragments behind opaque fragments, similar to early ray termination, helps to improve image quality and stability. Nevertheless, bucketing does not seem to work well when high transparency is used, since the correspondence between opacity and depth distance is increasingly lost (see 3rd and 6th rows in Fig. A.1). In this case, MLABDB operates like MLAB, inheriting the weaknesses mentioned in Sec. 4.5.

Wrt. SSIM, MBOIT performs slightly worse than MLABDB for lines with low and medium transparency due to either transparency over- or underestimation (cf. first two rows for PSNR and SSIM in Fig. A.1). However, for highly transparent lines, MBOIT can accurately simulate the transmittance function for each data set. Furthermore, it shows low variation in image quality during animations.

Regarding the PSNR values, less noise is introduced by MLABDB for highly opaque lines due to bucketing, whereas more noise is produced by MBOIT and VRC with consistently lower PSNR values for all data sets (cf. first two rows for PSNR in Fig. A.1). Whereas VRC has low PSNR values due to line inaccuracies and silhouette errors, low PSNR values for MBOIT are due to pixel-wise color distortions, originating from high variation in the transmittance, making it harder for MBOIT to properly reconstruct the transmittance. On the other hand, with increasing transparency and smoother transfer functions, the quality of MBOIT highly improves, and it is able to outperform MLABDB for such settings (compare green PSNR curves for each column in Fig. A.1 with best PSNR values in the sixth row). For high transparency, PSNR also indicates that merge heuristics in MLABDB perform worse and produce high noise in the image due to incorrect blending and color inconsistencies. These errors are even more pronounced in close-up views where more pixels are affected by incorrect blending, yielding worst PSNR values for TURBULENCE throughout all techniques.

The quality of VRC is consistently below that of MBOIT and MLABDB, showing even more severe variations when zooming into the data sets. The reason lies in the line discretization used by VRC, which introduces inaccuracies especially along the line silhouettes, i.e., a line is either missed or erroneously hit. In both cases, the pixel value can be very different from the ground truth, decreasing the measured image quality. Even though fewer lines are seen when zooming into the data, the differences now affect more and more pixels, and thus increasingly affect the image quality. With higher line transparency, the line contours are also increasingly blurred out, rendering these artifacts less pronounced and resulting in higher SSIM and PSNR. In the following, we show images of all rendering techniques using high transparency or semi-transparent rendering settings for ANEURYSM (cf. Fig. A.2), CONVECTION (Fig. A.3), TURBULENCE (Fig. A.4), and CLOUDS (Fig. A.5). The figures are composed of rendering results from Depth Peeling (ground truth) and each approximate technique (MLABDB, MBOIT, and VRC).

Per-pixel absolute error images are included to further highlight differences between each approximate technique and the ground truth image. Pixel-wise errors are computed using the absolute difference between the ground truth and the rendering for each channel and converted to an inverted gray scale image afterwards. Here, white color means no difference, and black represents highest possible error in the image.

Furthermore, we show the depth complexity of each view to depict the number of layers per pixel for each scene, where black represents 0 fragments and bright cyan is the maximum number of layers in the current view. This is achieved by counting the number of total fragments per pixel using atomic operations on the GPU. For our data set and a viewport of 1920×1080, the total number of fragments varied from 30 million (CONVECTION) to 280 million (CLOUDS) fragments during animation. The maximum number of layers ranged from 140 (ANEURYSM, CONVECTION) to 1000 (TURBULENCE) and even 9000 (CLOUDS).

Furthermore, we compute error metrics between the rendered image of each approximate technique and the ground truth in scenes with opaque, semi-transparent, and highly transparent render settings for all four data sets used in the paper. The plots in Fig. A.1 show SSIM (first three rows) and PSNR (last three rows) values of renderings over time produced during a pre-recorded flight with two zoom-ins.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TVCG.2020.2975795

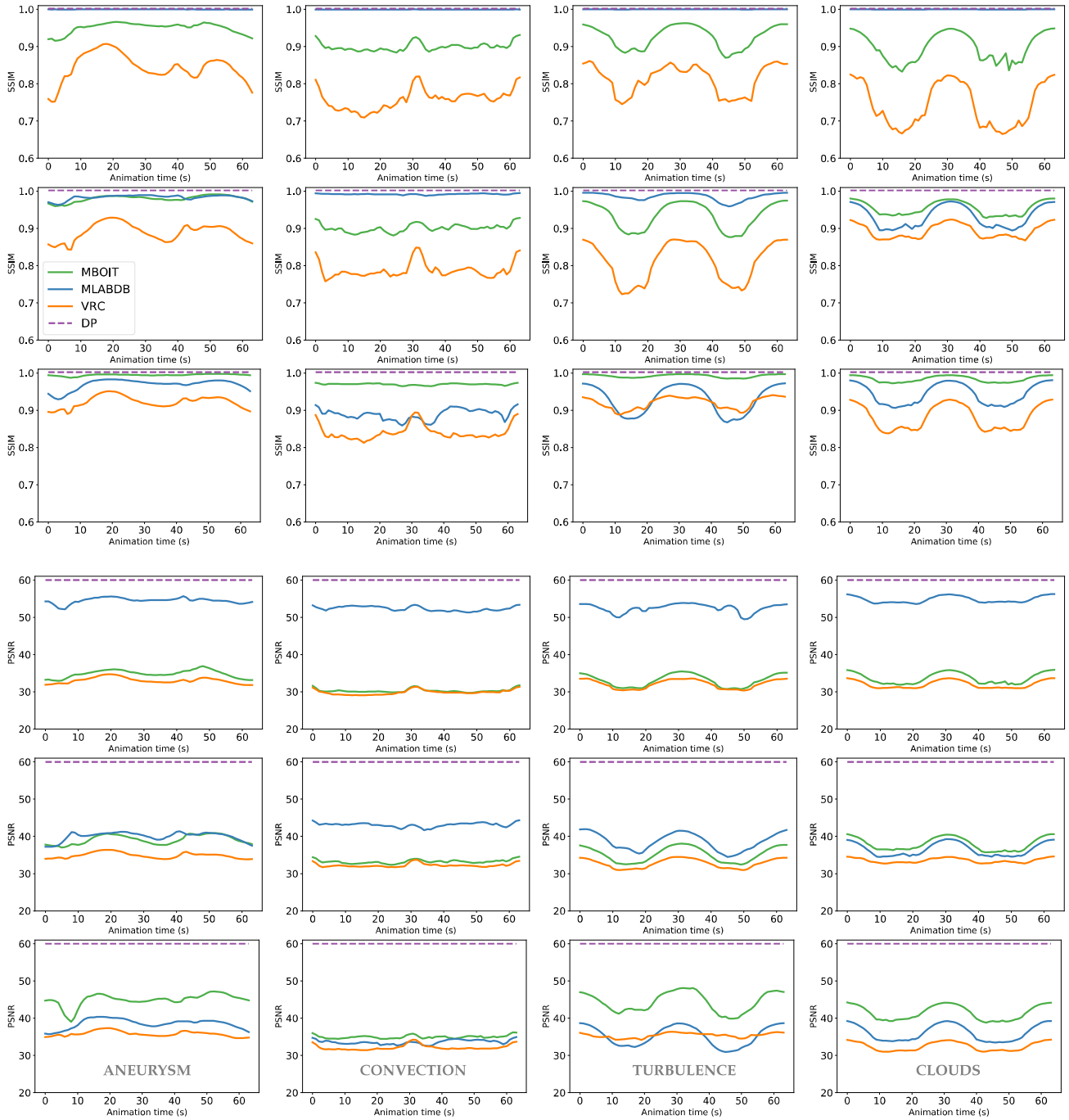


Fig. A.1: Error metrics of all techniques using opaque lines (1st and 4th row), transparent lines with opaque features of interest (2nd and 5th row), and highly transparent lines (3rd and 6th row). Columns are for ANEURYSM, CONVECTION, TURBULENCE, and CLOUDS, respectively. First three rows show SSIM for each technique, the last three rows show PSNR. A higher value is better.

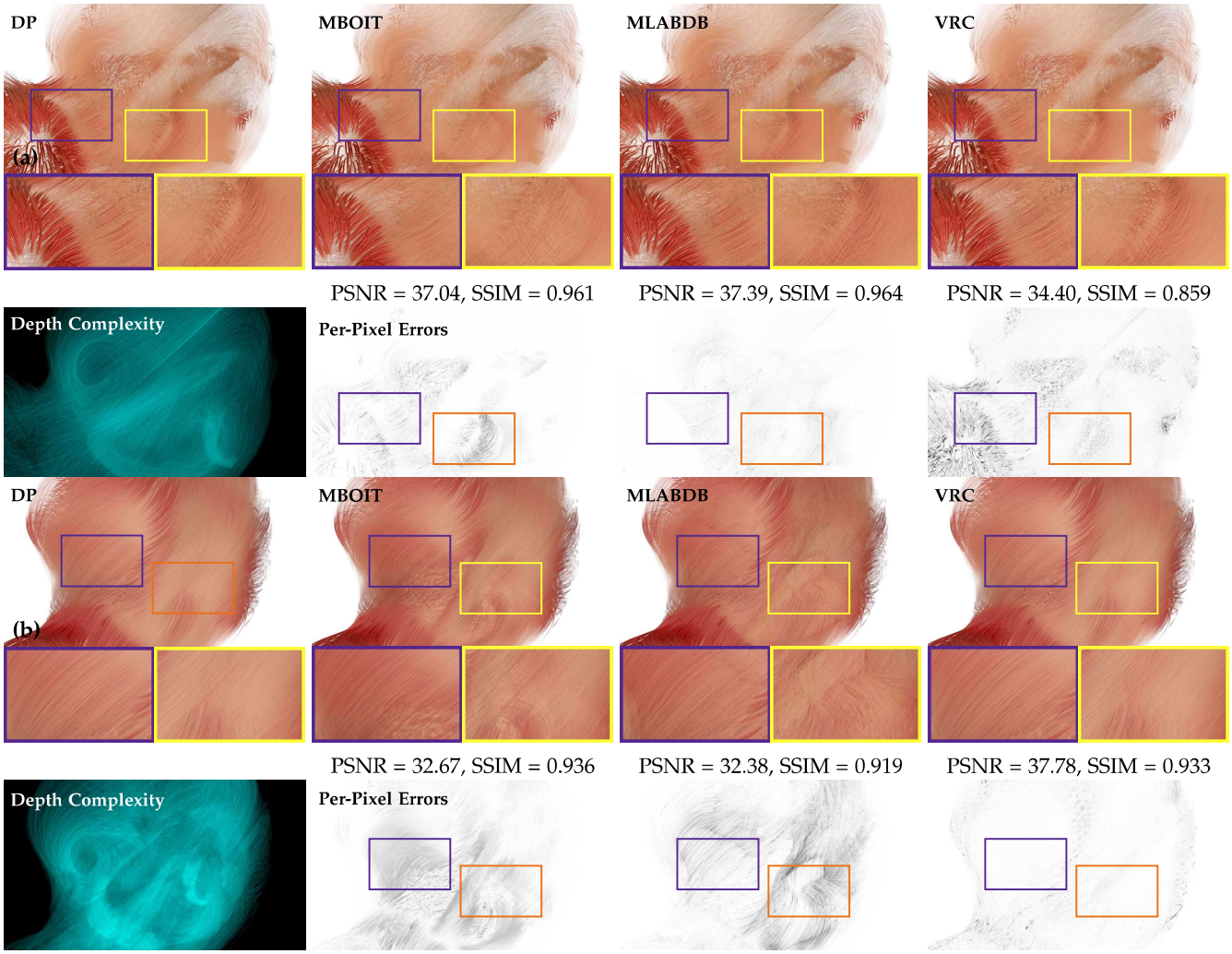


Fig. A.2: From left to right: ANEURYSM rendered with DP, MBOIT, MLABDB, VRC. 1st and 3rd row: Renderings using overall medium transparency with low-frequency variation and detailed views **(a)**, and with sharp transition between low and high transparency **(b)**. Blue and yellow rectangles highlight differences between techniques, close-up views are shown below each image. 2nd and 4th row: show depth complexity per pixel and per-pixel absolute differences to the ground truth (DP). Depth complexity: 62 – 76 million fragments in total, with max. 156 (a) and 174 (b) layers.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TVCG.2020.2975795

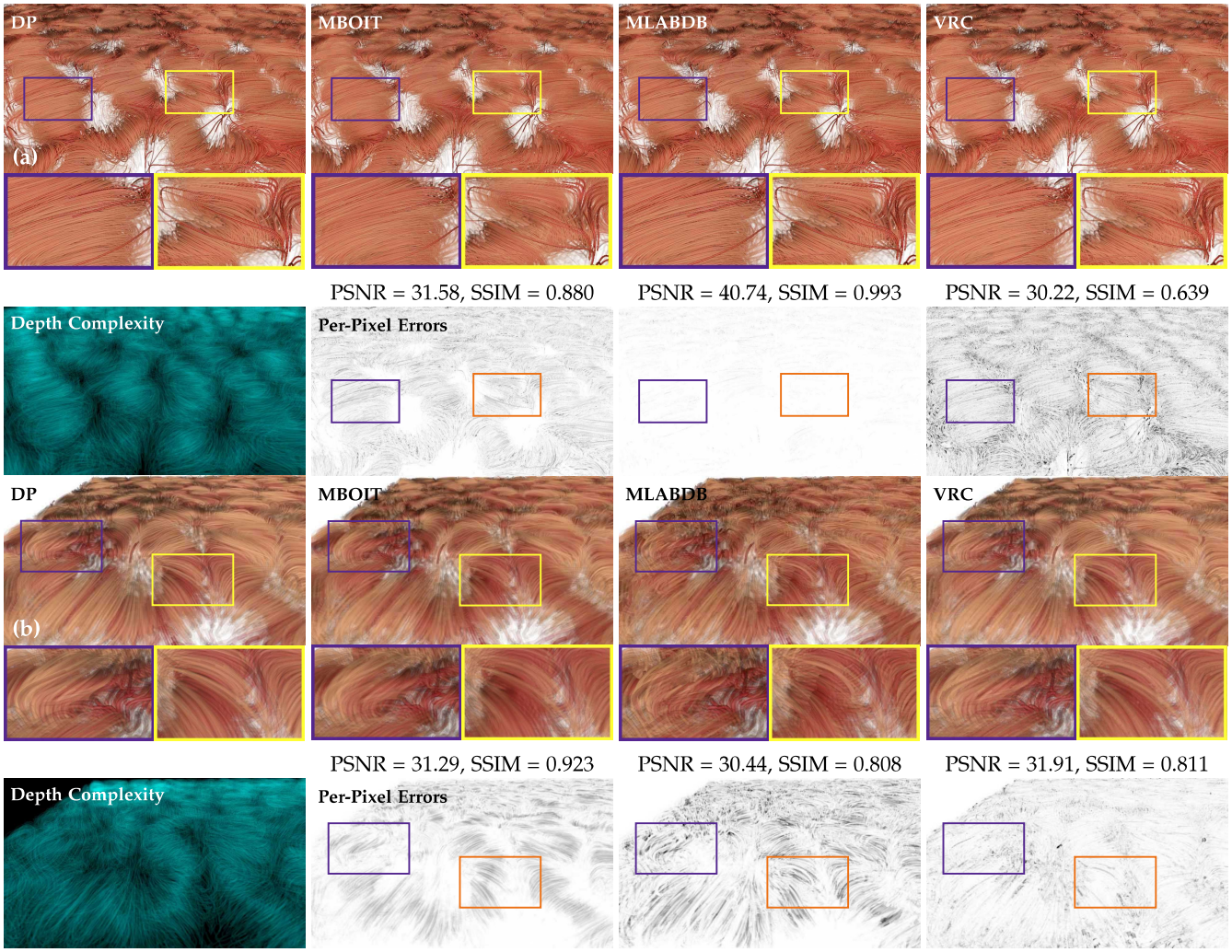


Fig. A.3: From left to right: CONVECTION rendered with DP, MBOIT, MLABDB, and VRC. Same transparency and rendering settings are used as in Fig. A.2. Depth complexity: 33 – 38 million fragments in total, with max. 140 (a) and 142 (b) layers.

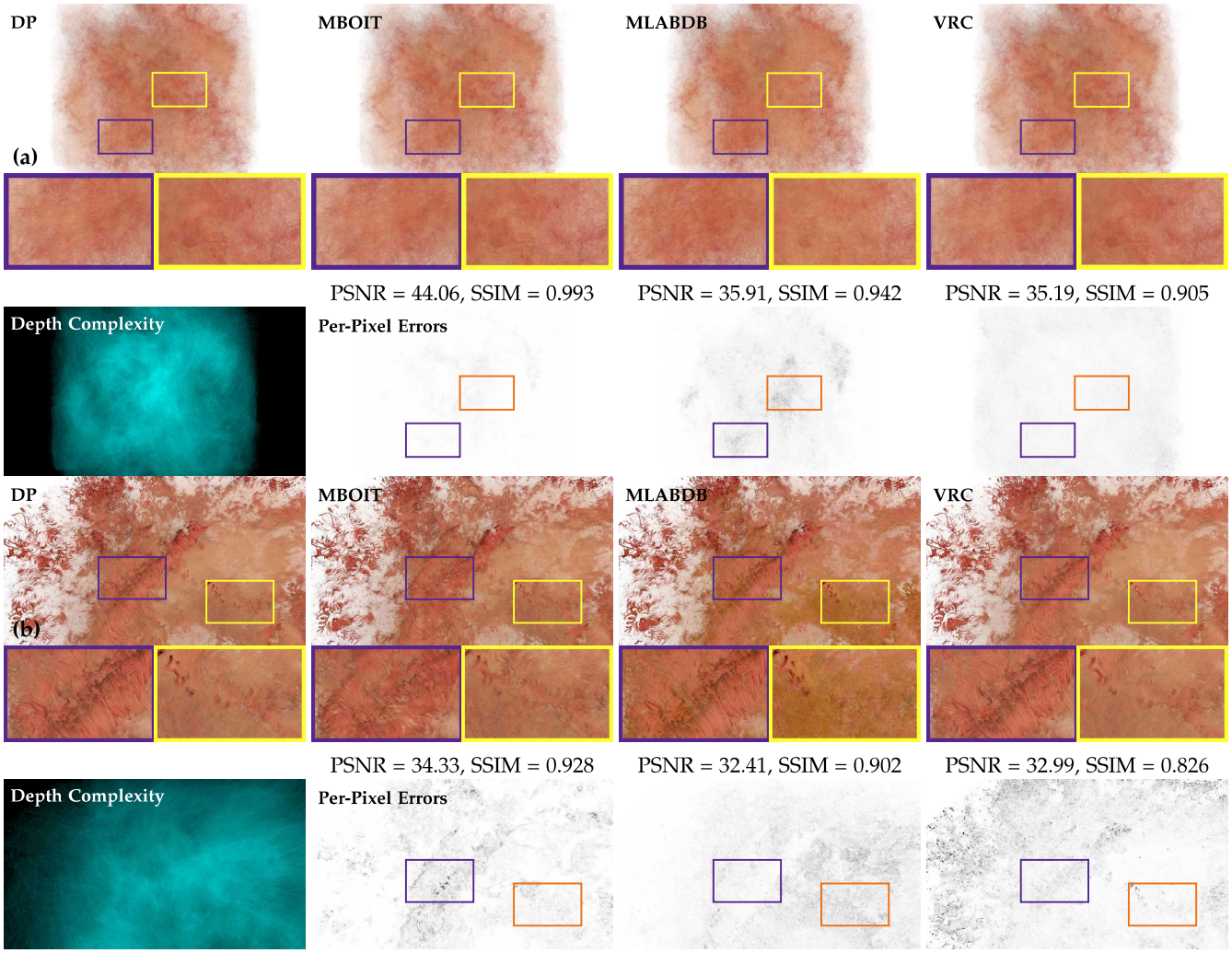


Fig. A.4: From left to right: TURBULENCE rendered with DP, MBOIT, MLABDB, and VRC. Same transparency and rendering settings are used as in Fig. A.2. **(a)** represents a view rendering the entire line set, whereas a close-up view is shown in **(b)**. Depth complexity: 91 – 197 million fragments in total, with max. 1310 (a) and 1460 (b) layers.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TVCG.2020.2975795

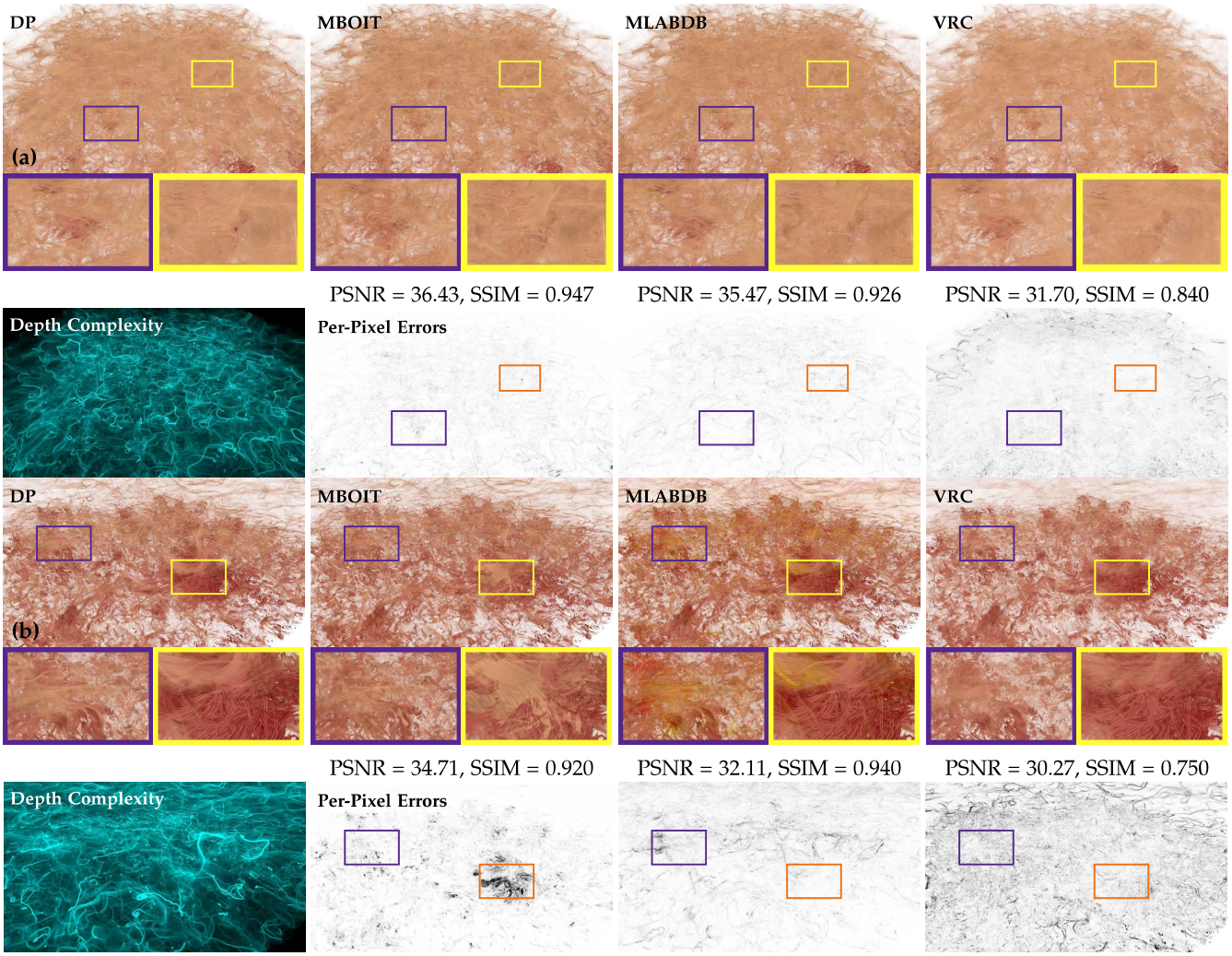


Fig. A.5: From left to right: CLOUDS rendered with DP, MBOIT, MLABDB, VRC. Same transparency and rendering settings are used as in Fig. A.2. Depth complexity: 87 - 135 million fragments in total with max. 7400 (a) and 8650 (b) layers.

APPENDIX B
SURFACE AND POINT DATA

Besides line rendering, we additionally demonstrate the application of the approximate techniques MLABDB and MBOIT to render transparent surface and point models. Fig. B.1 shows an isosurface in a Richtmyer-Meshkov instability[1] extracted with VTK and comprised of 516 million triangles. Constant transparency is used for all triangles. Fig. B.2 shows images of 55 million particles from a coal particle combustion simulation in Uintah[2] rendered with screen-oriented 2D splats. The velocity of each particle is mapped to transparency and color (from beige to red to blue, with blue being the highest).

In all figures, we show the rendering outcome of DP (ground truth), MLABDB, and MBOIT. In addition, we highlight the differences between the approximation and ground truth with (a) detailed views below all rendering algorithms, (b) the number of fragments per pixel (depth complexity) to relate per-pixel errors to the complexity of the scene, and (c) per-pixel errors from the ground truth.

For both data sets, MLABDB and MBOIT produce similar artifacts as for line sets, discussed in Appendix A and Sec. 4.7. As seen in Fig. B.1, MBOIT leads to blur effects, especially transitions between neighboring isosurfaces vanish with higher transparency. MLABDB, on the other hand, is better at preserving small geometric details. For instance, it retains the sharpness of (small) surface contours in regions of high surface variation with multiple transparent layers.

However, for our point cloud data set with a myriad of transparent surfaces — in our test more than 1000 transparent layers — MLABDB is not able to fully reconstruct the correct color due to accumulated errors caused by incorrect fragment merges, leading to blurry or wrongly colored features and misinterpretation of the data. This effect is even more prominent when mapping attributes with more than two colors (mapping from beige to red to blue, cf. Fig. B.2). Note that the depth complexity and per-pixel error images also clearly demonstrate that approximation errors of MLABDB increase with more layers. Since MLABDB is not order-independent, it also fails to preserve the correct visibility order of fragments, causing hidden features to suddenly appear in the final image (cf. yellow rectangles in Fig. B.2).

MBOIT, in contrast, can properly handle transmittance and color in this scenario and, besides small approximation errors, is able to preserve the correct visual appearance of features in the data (compare the detailed views in Fig. B.2). In both detailed views, MBOIT is able to reconstruct the sharpness of red colored particles (highlighted in the purple view) and does not cause hidden interior features to incorrectly become visible on the particle combustion data set (yellow view).

In summary, these findings reflect the results observed when rendering large line sets such as TURBULENCE and CLOUDS, which can be taken from green and blue curves in PSNR and SSIM plots (for all transparent rendering settings) and per-data image comparisons in Appendix A.

1. R. Cohen, W. Dannevik, A. Dimits, D. Eliason, A. Mirin, Y. Zhou, D. Porter, and P. Woodward, "Three-dimensional simulation of a richtmyer-meshkov instability with a two-scale initial perturbation," Physics of Fluids, vol. 14, no. 10, pp. 3692–3709, 1 2002.

2. M. Berzins, J. Luitjens, Q. Meng, T. Harman, C. A. Wight, and J. R.Peterson, "Uintah: A scalable framework for hazard analysis," in Proceedings of the 2010 TeraGrid Conference, ser. TG '10. New York, NY, USA: ACM, 2010, pp. 3:1–3:8

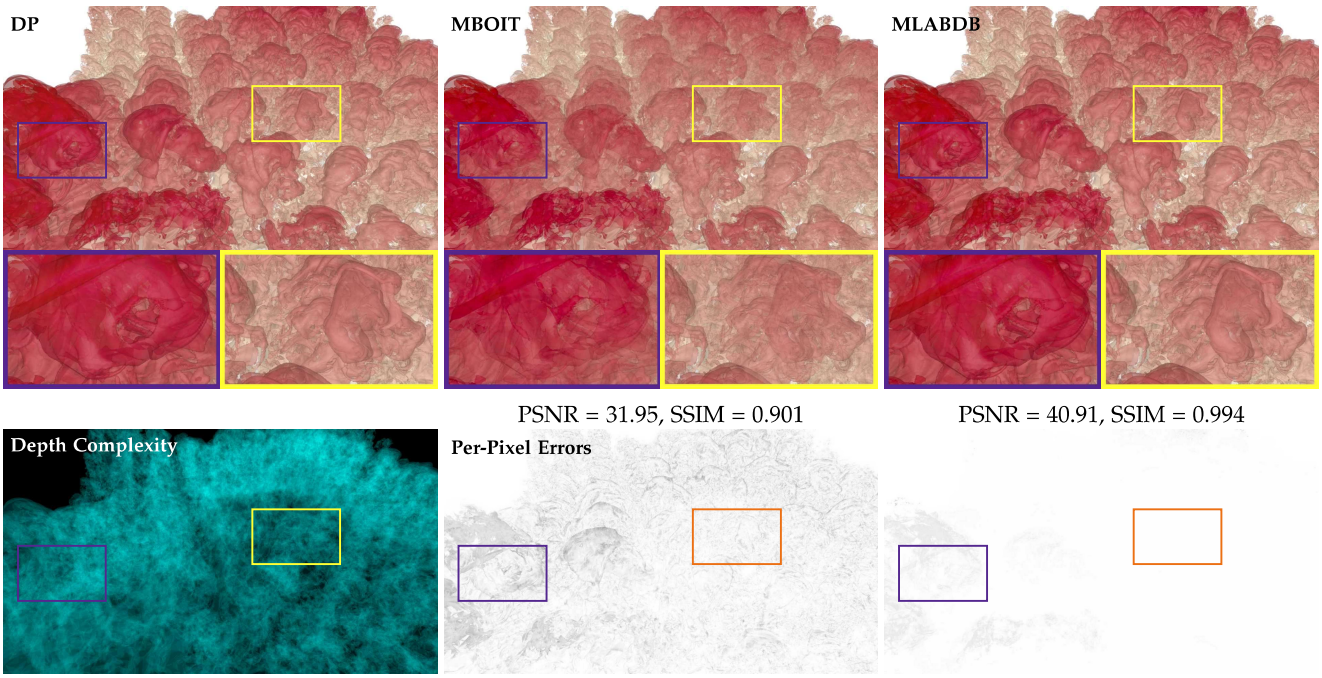


Fig. B.1: DP, MLABDB, and MBOIT used to render isosurfaces (triangle meshes) from the Richtmyer-Meshkov data set. Detailed views and error metrics are shown below each of the rendered images. Depth complexity: 31.1 million fragments in total with max. 90 layers.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TVCG.2020.2975795

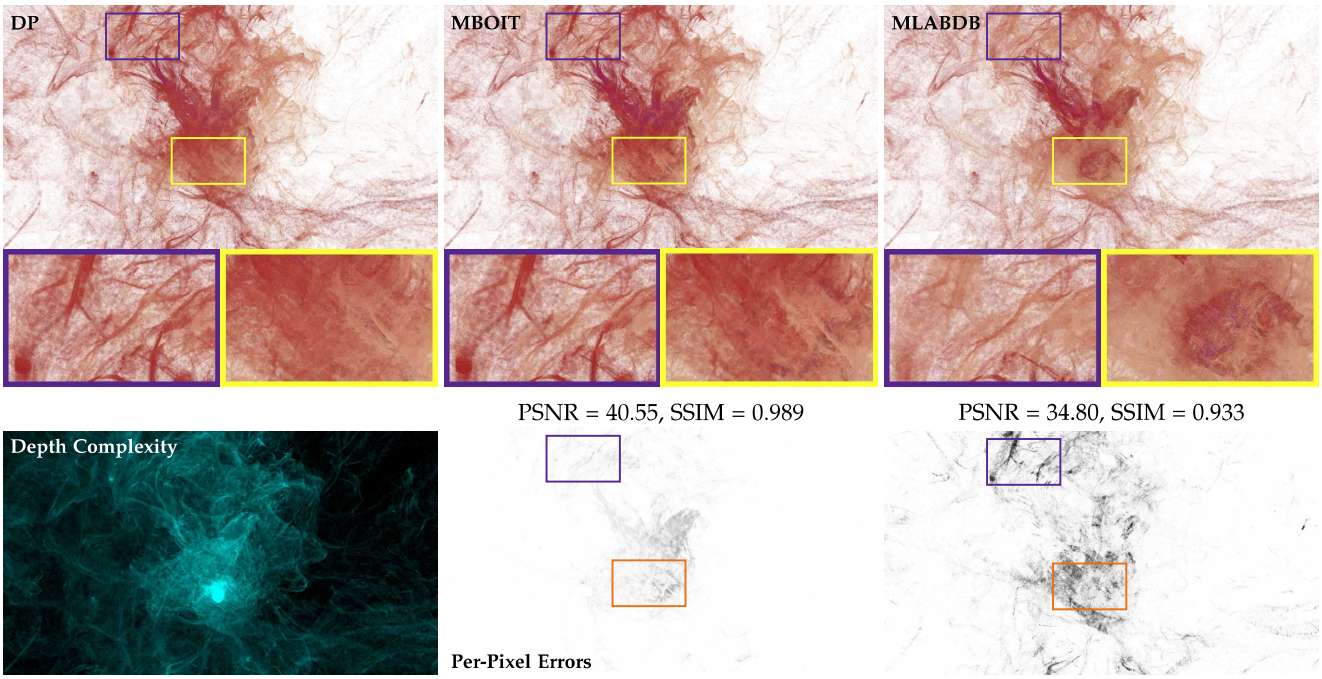


Fig. B.2: DP, MLABDB, and MBOIT are used to render point clouds (2D view-oriented splats) from a coal combustion particle simulation. Detailed views and error metrics are shown below each renderings. Depth complexity: 42 million fragments in total with max. 1720 layers.